ECCentric: An Empirical Analysis of Quantum Error Correction Codes

ALEKSANDRA ŚWIERKOWSKA, Technical University of Munich, Germany
JANNIK PFLIEGER, Technical University of Munich, Germany
EMMANOUIL GIORTAMIS, Technical University of Munich, Germany
PRAMOD BHATOTIA, Technical University of Munich, Germany

Quantum Error Correction (QEC) is essential for building scalable quantum computers, but a lack of systematic, end-to-end evaluation methods makes it difficult to assess how different QEC codes perform under realistic conditions. The vast diversity of codes, an expansive experimental search space, and the absence of a standardized framework prevent a thorough, holistic analysis. To address this, we introduce ECCentric, an end-to-end benchmarking framework designed to systematically evaluate QEC codes across the full quantum computing stack. ECCentric is designed to be modular, extensible, and general, allowing for a comprehensive analysis of QEC code families under varying hardware topologies, noise models, and compilation strategies.

Using ECCentric, we conduct the first systematic benchmarking of major QEC code families against realistic, mid-term quantum device parameters. Our empirical analysis reveals that intra-QPU execution significantly outperforms distributed methods, that qubit connectivity is a far more critical factor for reducing logical errors than increasing code distance, and that compiler overhead remains a major source of error. Furthermore, our findings suggest that trapped-ion architectures with qubit shuttling are the most promising near-term platforms and that on noisy devices, a strategic and selective application of QEC is necessary to avoid introducing more errors than are corrected. This study provides crucial, actionable insights for both hardware designers and practitioners, guiding the development of fault-tolerant quantum systems.

CCS Concepts: • **Hardware** → **Quantum error correction and fault tolerance**; Quantum technologies; • **Computer systems organization** → **Quantum computing**; • **General and reference** → *Evaluation*; *Empirical studies.*

Additional Key Words and Phrases: quantum computing, quantum error correction, benchmarking

ACM Reference Format:
Aleksandra Świerkowska, Jannik Pflieger, Emmanouil Giortamis, and Pramod Bhatotia. 2026. ECCentric: An Empirical Analysis of Quantum Error Correction Codes. *Proc. ACM Meas. Anal. Comput. Syst.* 10, 2, Article 37 (June 2026), 33 pages. https://doi.org/10.1145/3805635

1 Introduction

Context and motivation. While the promise of quantum computing [9, 58, 101, 118] has led to the rapid development and accessibility of Quantum Processing Units (QPUs) [11, 12, 56, 71, 105, 107], achieving true quantum advantage is fundamentally limited by persistent hardware noise [101].

Quantum Error Correction (QEC) offers the only known path to fault tolerance, a prerequisite for building scalable quantum computers capable of solving classically intractable problems [4, 46, 52]. It encodes a single logical qubit into a redundant, entangled system of multiple noisy physical qubits [98, 101]. Once such an encoded circuit is transpiled into the hardware's native gate set and

Authors' Contact Information: Aleksandra Świerkowska, aleksandra.swierkowska@tum.de, Technical University of Munich, Germany; Jannik Pflieger, Technical University of Munich, Germany; Emmanouil Giortamis, Technical University of Munich, Germany; Pramod Bhatotia, Technical University of Munich, Germany.

ACM 2476-1249/2026/6-ART37
https://doi.org/10.1145/3805635

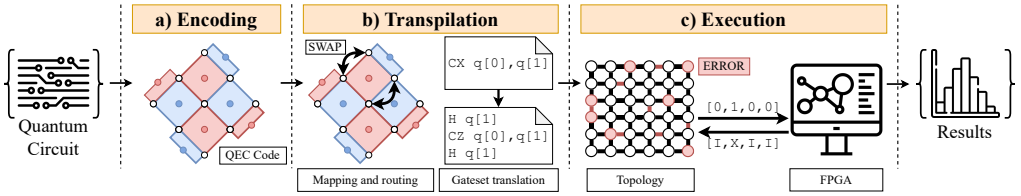


Fig. 2. Workflow of a quantum circuit protected with QEC, showing how mapping and routing, translation, and decoding are handled during execution.

adapted to its topology (Fig. 2), it is executed on the device, where syndrome measurements are continuously performed, allowing the decoder to correct errors in real time [16].

Notably, a diverse family of QEC codes exists, each with unique resource overheads, operational assumptions, and performance characteristics. Their suitability heavily depends on the underlying physical qubit technology and its noise model [26, 46].

Research gap and challenges. Despite the variety of QEC codes, there's a significant research gap: a lack of a systematic, end-to-end methodology for rigorously comparing them. This makes it challenging to accurately assess a code's performance under the specific, realistic noise and software constraints of a particular QPU. A comprehensive practical evaluation needs to consider the entire execution pipeline, as each stage can influence the code's effectiveness.

Unfortunately, current QEC evaluations are often ad hoc and limited, either focusing on a small number of codes with oversimplified noise models [36, 45, 67, 69] or analyzing components such as decoders in isolation [17, 32, 37, 43, 60, 82, 127, 139]. As a result, no existing benchmarking framework provides an end-to-end, systematic evaluation. Current tools exhibit a distinct dichotomy: fast, specialized simulators, such as Stim [49], are hardware-agnostic and rely on simplistic error models, whereas general-purpose frameworks, such as Qiskit [73], lack the scalability needed to simulate large, fault-tolerant codes [46]. These limitations create a major blind spot in practical quantum computing.

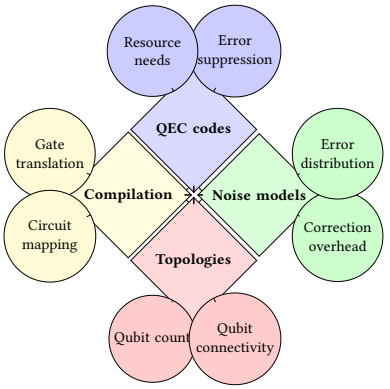


Fig. 1. Our comprehensive framework provides a structured empirical analysis of QEC landscape. We examine the QEC field across four key dimensions, addressing eight critical research questions.

To effectively address this research gap, we must confront three fundamental challenges. *Firstly, the diverse QEC landscape:* A vast number of codes exist, each with unique overheads, error thresholds, and decoding complexities, making it difficult to compare their effectiveness across different hardware and noise models [24, 33, 37, 74, 129, 135]. *Secondly, the vast experimental search space:* A code's performance is determined by a multi-dimensional interplay of QPU topology, physical noise, and compiler artifacts, rendering an exhaustive search computationally intractable [18, 127, 134, 136]. *Finally, the lack of a suitable benchmarking framework:* Existing tools are either too slow for large-scale studies (e.g., Qiskit [73]) or too hardware-agnostic for realistic evaluation (e.g., Stim [49]). A viable solution must be general, supporting a wide spectrum of code families and QPU architectures; extensible, to easily integrate new decoders and evolving noise models; and modular, to enable the isolation and analysis of specific stages within the QEC pipeline, from encoding to decoding.

Research question and our approach. These challenges lead to our central research question: *How can we systematically evaluate the suitability of quantum error correction (QEC) codes for practical applications on current and near-term quantum devices?*

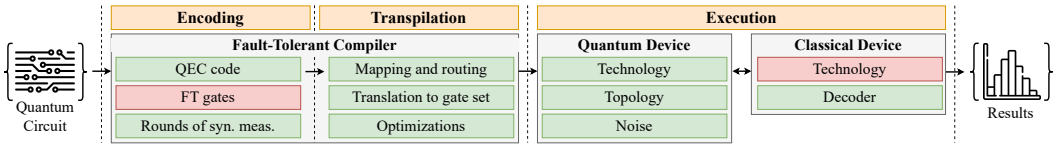


Fig. 3. Lifetime of a quantum circuit protected with QEC code. *Our analysis covers the elements in green.*

To answer this question, we propose a two-staged approach. First, we impose structure on the diverse and rapidly evolving QEC landscape by creating a systematic taxonomy of QEC codes. From this taxonomy, we have identified five primary families of codes and selected a representative member from each to ensure our analysis is both fair and comprehensive.

Building on this foundational taxonomy, we introduce ECCentric, a new, end-to-end benchmarking framework designed for the systematic evaluation of QEC codes on current and near-term quantum devices. ECCentric is general, capable of supporting a wide array of QEC codes, QPU technologies, and compiler back-ends. It is modular, allowing researchers to customize the evaluation pipeline and scope of their experiments. And it is extensible, making it easy to integrate new code, hardware features, and compilation strategies as they emerge.

Leveraging the full capabilities of the ECCentric framework, we structure our investigation around four key dimensions (presented in Fig. 1) that influence QEC performance, defining two targeted research questions for each to systematically assess their impact. These four dimensions include *QEC codes*, where we select six representative codes from all major families using our QEC code taxonomy; *QPU noise models*, where we include both ideal and realistic models derived from current (e.g., Google Willow [3]) and next-generation (e.g., Quantinuum Apollo [105] and IBM Flamingo [107]) quantum devices; *QPU topologies*, where we explore a range of layouts, from abstract to the complex architectures of existing hardware (e.g., Google Willow [3] and Inflection [111]), varying qubit count and connectivity to isolate the impact of topology; and finally, *quantum compilation*, where we investigate how different quantum compilers (e.g., Qiskit [73] and TKET [120]) and their internal stages, such as mapping and routing heuristics (e.g., SABRE [88]), affect QEC performance.

Our key findings, implications, and lessons learned. Our systematic evaluation reveals several critical insights and implications for both hardware and software development:

- *Fault-tolerance with Trapped-Ion is near:* Our analysis of projected hardware roadmaps shows that the planned error rates and qubit shuttling capabilities of these devices should eliminate logical errors across most evaluated codes within the next five years [105].
- *Intra-QPU execution is superior:* Our results show that distributing codes across QPUs increases the logical error rate by 69.14% compared to single-QPU execution. However, while link fidelity is a factor, the principal issue is the low number of connections, and hardware development should prioritize increasing the cross-QPU links rather than solely focusing on their fidelity.
- *Prioritize connectivity over code distance:* Our experiments show that increasing code distance is often ineffective, increasing the logical error rate on average by 0.012 with only 30.95% of adjustments yielding meaningful improvement. In contrast, improving connectivity provides substantial gains: moving from a grid to a fully connected topology reduces the logical error rate by 81.9%, while devices with qubit shuttling outperform those without by 45.48%. Thus, hardware manufacturers should prioritize improving qubit connectivity.
- *QEC-aware compilation is essential:* The compilation process introduces significant overhead that can undermine the benefits of error correction. We find that mapping and routing add an average of 134.095% more two-qubit gates, while optimized translation still adds 3.166 extra gates per original gate. Mitigating this requires QEC-aware compilers that optimize at the logical-qubit

Table 1. Overview of quantum technologies. *Quantinuum (Q.) Apollo [105] data are extrapolated from H2 device specifications [106]. For Infleqtion, reset error is included in readout error, leakage in gate error, and SPAM is approximated as measurement plus reset, as it is not directly reported [111].*

Characteristic	**Superconducting**		**Trapped-Ion**	**Neutral Atoms**
Qubit Type	Transmon qubits [114]		Trapped ions [28]	Rydberg atoms [62]
Topology	heavy-hex [107], grid [3]		race track [96], grid [105]	2D array [111] arbitrary 3D geometry [15]
Mid-Circuit Qubit Movement	No		Yes [96]	Yes [19]
Coherence Time	10 - 100 μs [81]		> 4s [105, 130]	1 - 60s [132]
Operation Speed	12 - 25 ns [10]		1-10 μs [116]	0.4 - 2 μs [132]
Real Error Rates	**Willow** [3]	**Flamingo** [2]	**Q. Apollo** [105, 106]	**Infleqtion** [111]
SQ	6.2×10^{-4}	2.5×10^{-4}	8.0×10^{-6}	9.8×10^{-4}
2Q	2.8×10^{-3}	2.0×10^{-3}	1.4×10^{-4}	6.5×10^{-3}
SPAM	9.5×10^{-3}	2.5×10^{-4}	1.33×10^{-4}	4.0×10^{-3}
Idle	9.0×10^{-3}	-	5.3×10^{-5}	-
Crosstalk	5.5×10^{-4}	-	6.3×10^{-7}	-
Leakage	2.5×10^{-4}	-	4.3×10^{-5}	-

level, for example, by prioritizing efficient routing to busy ancilla qubits and canceling redundant gate sequences across repeated QEC cycles.

- *Heterogeneity is not a primary concern:* We find the variability in individual qubit quality or in readout correctness has a negligible impact on performance, changing the logical error rate by 0.02 on average. This suggests that complex, variance-aware compilation strategies might be an unnecessary burden. This frees compiler developers to de-prioritize these intricate techniques and instead pursue more straightforward and effective designs.
- *Apply QEC strategically, not universally:* On noisy, near-term devices, the indiscriminate application of QEC can be counterproductive. Our results show that at a physical two-qubit error rate of 0.002, most of our evaluated codes fail, and at 0.004, none remain effective, introducing more errors than they correct. This necessitates a selective approach where QEC is only used for operations or qubits where its benefits outweigh its overhead.

Our contributions. Overall, to the best of our knowledge, this paper provides the first systematic benchmarking of multiple QEC codes, focusing on their practical application on real-world hardware, compilers, and noise models. Our contributions are summarized as follows:

- *Experimental QEC code analysis:* A systematic benchmarking of the major Quantum Error Correction (QEC) code families under realistic experimental conditions.
- *ECCentric framework:* The design of a modular benchmarking framework that can be readily extended to incorporate additional codes, devices, noise models, and decoders.
- *QEC code taxonomy:* The introduction of a structured taxonomy for QEC codes, clearly presenting the differences and shared characteristics among the primary families.
- *Concatenated Steane code extension:* The first extension of the [[49,1,9]] concatenated Steane code [99] to [[373,1,27]] concatenated Steane code.

2 Quantum Error Correction (QEC)

QEC is a technique of protecting quantum information from noise by encoding it into entangled states of multiple qubits, i.e., logical qubits [98]. Preparing and running such protected circuits, however, requires a complex toolchain that extends well beyond standard compilation. To the best of our knowledge, no fully integrated end-to-end compiler for QEC exists, but building on existing frameworks [84, 131], we illustrate in Fig. 3 the stages of compiling and executing a quantum circuit with QEC, each of which can introduce overhead and weaken protection:

Encoding. To provide protection for a quantum circuit (Fig. 2(a)), we start by choosing a QEC code matching the constraints of the device regarding topology and noise profile, and preparing logical qubits by entangling them with themselves and with ancilla qubits accordingly. Next, we translate the gates in the circuit into fault-tolerant (FT) gates able to manipulate logical qubits without falsely marking errors [98]. While some gates can be translated with relatively low overhead, they alone cannot form a universal set [40]. Missing gates need to be added with costly techniques such as code switching [8], lattice surgery [65], or magic state distillation [25]. At present, no general tool can automatically translate an arbitrary circuit into a fully encoded form with support for these gates. Afterwards, we insert rounds of syndrome measurement periodically into the circuit, i.e., additional stages of measurement done on the ancilla qubits entangled with data qubits, which detect whether errors have occurred [98].

Transpilation. Now we need to adapt the circuit to the target device [93]. Transpilation involves multiple stages [7, 78, 109]: Firstly, we map the logical qubits of the circuit to physical qubits of the device on which they will be executed [89]. To make two-qubit gate execution possible, qubits involved in them need to be placed on physically adjacent hardware. However, as that is not always possible, we follow this stage with routing, during which we insert a gate sequence (typically an additional swapping operation) to move together the quantum states that must interact [136]. To keep overhead low, we aim for routing sequences that minimize the number of movements. We then translate each gate in the circuit into an equivalent operation from the elementary set supported by the target device [48], as we illustrate in Fig. 2(b). Since mapping the large connection graph, inserting SWAP operations, and decomposing unsupported gates produce substantial overhead, we interleave those mandatory stages with optimization passes, which aggressively prune the length and the gate count of the circuit [76], e.g., by removing mutually cancelling gates.

Execution. The promise of quantum computing [58, 119] has spurred the development of diverse hardware platforms [28, 62, 68], each with distinct trade-offs in coherence times, gate speed, connectivity, and noise profiles, as summarized in Tab. 1. Superconducting devices, the most common platform, offer fast gates but are constrained by limited qubit lifetimes and fixed, sparse connectivity, making them particularly susceptible to idling errors, crosstalk between neighboring qubits, and more gate errors introduced by additional SWAP operations [3, 68]. In contrast, trapped-ion and neutral atoms devices provide longer coherence and can achieve dynamic, all-to-all connectivity via mid-circuit qubit movement, but are affected by errors arising from qubit transport and imperfect control lasers [19, 28, 96, 111, 130]. Each platform's unique error characteristics are a critical factor for the effectiveness of QEC [32].

During circuit execution, those errors can prevent us from obtaining meaningful results. To handle them, we periodically take syndrome measurements and send the results to a classical processor, commonly an FPGA [90] or GPU [44], where a decoder, usually tailored to the chosen QEC code, locates errors and selects the appropriate corrections, trading off precision and speed to apply them before qubits decohere [16]. The performance depends on the specific combination of QEC code, decoding algorithm, and hardware, especially regarding latency. This process, which we illustrate in Fig. 2(c), is essential to suppress noise and ultimately enable effective circuit execution.

Table 2. Research questions we pursue in this work, their relations to the considered problem dimensions, and the sections where they are discussed.

Sec.	Research Question	Codes	Topologies	Noise	Compilation
§4	**RQ#1**: Will the effectiveness of the QEC codes noticeably change when their distance expands?	✓	✓	✗	✗
	RQ#2: Does higher qubit connectivity always improve the effectiveness of QEC codes?	✓	✓	✗	✓
	RQ#3: Does qubit quality variance critically impact logical error rates beyond the effect of the mean?	✗	✗	✓	✓
	RQ#4: Does a QEC code's performance scaling differ between single-QPU and distributed execution?	✓	✓	✗	✗
	RQ#5: Which quantum technology provides favorable error rates and hardware features for effective QEC?	✓	✓	✓	✗
§5	**RQ#6**: How do mapping and routing compilation stages influence the effectiveness of QEC codes?	✓	✗	✗	✓
	RQ#7: How does the translation stage influence the effectiveness of QEC codes?	✓	✗	✗	✓
§6	**RQ#8**: Which decoders achieve the best performance across QEC code families?	✓	✗	✓	✗
	RQ#9: Is applying QEC always beneficial, or can the QEC overhead become a net source of error?	✓	✗	✓	✗

Takeaway: Effective QEC performance depends on the entire quantum computing stack, as every stage from compilation to execution introduces potential overhead and errors. *First*, encoding and translation decisions, such as the frequency of syndrome measurements and the conversion to fault-tolerant gates, create significant qubit and gate overhead [61, 85]. *Second*, the transpilation of large QEC circuits adds substantial routing overhead, with each extra gate introducing more noise that degrades the code's effectiveness [42, 100]. *Last*, the hardware and decoder are critical; a device's topology and noise profile determine a code's viability, while a slow or inaccurate decoder can introduce more errors than it corrects [18, 37, 127].

3 Overview

In this section, we discuss the scope of our study along with its limitations and introduce the research questions we aim to address (§ 3.1). Then, we present a systematic taxonomy of QEC codes to enable their methodical exploration (§ 3.2). Finally, we propose our novel benchmarking framework (§ 3.3), which we later use for the experimental evaluation (§ 4, § 5, § 6).

3.1 Research Scope

Each of the steps introduced in § 2 requires in-depth exploration to evaluate its impact on error correction. Taken together, these steps create a combinatorial explosion of factors, making exhaustive analysis infeasible within a single study. To manage this complexity, we slightly narrow the scope. We exclude the construction of FT logical gates, as, to the best of our knowledge, the only existing framework supporting them is limited to the surface code and still under active development [1], and developing a general, scalable FT gate constructor would constitute a substantial standalone contribution. FT gates would require far more resources and greatly increase circuit depth, resulting in higher logical error rates. We also omit classical decoder latency, which depends strongly on the code, decoder, and hardware platform, making any arbitrary artificial value unsuitable for general analysis. Since real-time decoding is planned to operate under 1 μs, the resulting decoherence errors in a real system would be only marginally higher than those observed here [16]. We further discuss both limitations in § 7. Additionally, because circuit optimization techniques generally

improve performance by reducing gate overhead, we treat them as a secondary consideration during compilation.

Even with these restrictions, the remaining scope remains substantial. To approach it systematically, we organize our work around four key dimensions: QEC codes, quantum compilation, quantum device noise models, and quantum device topology. Based on these dimensions, we formulate nine research questions, summarized in Tab. 2, which probe their impact on QEC effectiveness. We select these questions to ensure comprehensive coverage while focusing on practical aspects of compilation and execution, reflecting current development directions and targeting well-known bottlenecks (e.g., limited device size or transpilation overhead). By addressing them experimentally, we aim to generate practical insights that guide real-world QEC implementations.

3.2 Quantum Error Correction Codes Taxonomy

To enable a fair and systematic evaluation across the vast landscape of QEC codes, we first introduce a taxonomy to structure our analysis. As shown in Tab. 3, we organize codes into four main families based on their core principles: subsystem stabilizer, QLDPC, concatenated, and topological codes. We describe each family through its core principles and highlight important variations where relevant:

Table 3. Taxonomy of main QEC code families.

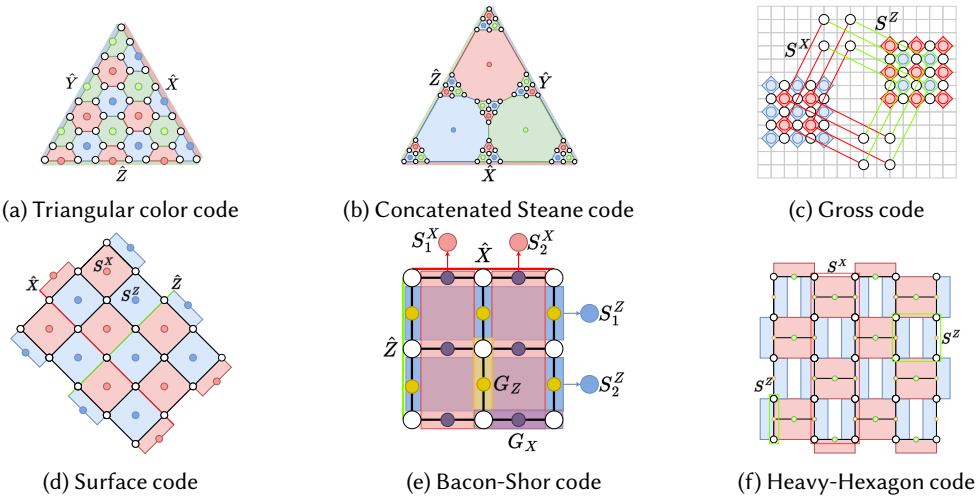
Subsystem			
Stabilizer			
Concatenated	**QLDPC**	**Topological**	**Subsystem Stabilizer**
Concatenated Steane	BB (Gross)	Surface, Color	Bacon-Shor
		Heavy-hex	

Stabilizer codes. Stabilizer codes are built upon a group of mutually commuting Pauli operators $\mathcal{S}$, where a state $|\psi\rangle$ satisfies $S|\psi\rangle = |\psi\rangle$ for all $S \in \mathcal{S}$ [57]. Errors are detected through syndrome extraction: data qubits are entangled with ancillas prepared in known states, which are then measured to reveal occurring errors [112]. Their compatibility with efficient classical simulation has made stabilizer codes the foundation of most QEC schemes.

To describe specific code instances of stabilizer codes, we use the standard $[[n,k,d]]$ notation, where n is the number of physical qubits used to encode k logical qubits, and d is the code distance, which is the minimal number of physical qubits that must be corrupted before an error becomes undetectable [112]. Notably, n does not include ancilla qubits required for syndrome measurement.

Subsystem stabilizer codes. Subsystem codes generalize stabilizer codes by encoding logical information into a subsystem instead of a subspace, introducing additional degrees of freedom, called gauge qubits, which do not influence the logical information and can be viewed as non-essential bits [6]. In practice, attention centers on the *subsystem stabilizer* subclass, which preserves the stabilizer formalism for efficient simulation while using gauge operators to replace some stabilizers with lower-weight checks, simplifying syndrome extraction [63]. An example of a subsystem stabilizer code is the *Bacon-Shor code* [13]. Defined on a rectangular lattice of size $m_1 \times m_2$, it has parameters $[[m_1 m_2, 1, \min(m_1, m_2)]]$ [41], and is particularly well-suited for trapped-ion devices, where under realistic noise models it can outperform the state-of-the art code with larger distance [38].

Topological codes. Topological codes encode information in global features of a lattice, making them intrinsically robust against local errors [21, 79]. Their local stabilizers correspond to lattice features like vertices or plaquettes, and their distance grows with the lattice size. Among topological codes, *surface codes* [46] are the most studied and considered as state-of-the-art [140], typically defined on a 2D square lattice. A prominent variant, the *rotated surface code*, reduces the physical qubits overhead [66] while preserving the code's core properties. Another key subclass is *color codes* [22], defined on lattices with colored plaquettes so that no two adjacent plaquettes share the same color, with each color corresponding to a subset of stabilizers. A notable example is the *triangular color code*, built on a 6.6.6 hexagonal lattice with three plaquette colors, which combines strong



(a) Triangular color code (b) Concatenated Steane code (c) Gross code

(d) Surface code (e) Bacon-Shor code (f) Heavy-Hexagon code

Fig. 4. Visualization of the six QEC codes explored in this work. *White dots represent data qubits, yellow and purple dots indicate gauge qubits, and remaining dots correspond to additional ancilla qubits.* $\hat{X}$, $\hat{Y}$, *and* $\hat{Z}$ *symbolize logical operators,* G_X *and* G_Z *gauge operators. Field colors indicate stabilizers* S^X *and* S^Z.

error suppression with a versatile set of low-overhead fault-tolerant gates, making it a promising candidate for scalable quantum computing [21, 83].

Quantum low-density parity-check (QLDPC) codes. QLDPC codes are stabilizer codes with sparse parity-check matrices, meaning that each stabilizer involves only a small subset of qubits [26]. This sparsity reduces measurement complexity but at the cost of long-range connectivity. The key advantage of QLDPC codes lies in their efficient scaling, maintaining a constant logical-to-physical qubit ratio (encoding rate) as the code distance grows. A prominent subclass of QLDPC codes is *bivariate bicycle (BB) codes* [24], which are constructed from bivariate polynomials with regular stabilizer patterns, preserving a good encoding rate while achieving even higher physical error rates they can tolerate (error thresholds). A notable example is the *gross code* [[144,12,12]], which encodes 12 logical qubits using 288 physical qubits, achieving performance comparable to a surface code that would require thousands of qubits [24].

Concatenated codes. Concatenated codes encode quantum information by nesting one code within another [80]. Specifically, an outer code encodes the logical qubits, and each physical qubit of this code is further encoded using an inner code. This layered structure allows flexible combinations of codes to improve error correction. A notable example of a concatenated code is the *concatenated Steane code* [99], denoted as $[[7^m,1,3^m]]$, able to achieve even lower logical error rates than the triangular color code. It is constructed by recursively concatenating the standard Steane code [[7,1,3]] [121] with itself $m-1$ times. However, a common drawback of concatenated codes is that their encoding rate decreases exponentially with the concatenation level.

Architecture-specific codes. Additionally, although we do not distinguish them as a family, some QEC codes are designed to combine different characteristics to meet the constraints of specific quantum hardware. A prominent example is the *heavy-hexagon code*, introduced as a sparse alternative to the surface code [31] and developed for the heavy-hex topology (e.g., IBM Heron [2]). It blends topological features, using a lattice-based layout, with subsystem stabilizer properties, leveraging gauge operators to reduce qubit connectivity.

For our study, we select representative codes from each major family: the Bacon-Shor, rotated surface, triangular color, gross, and concatenated Steane codes, each chosen as a high-performing

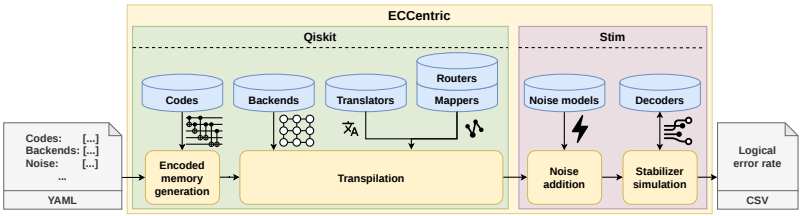


Fig. 5. Architecture of the ECCentric framework. *Our design of a fully modular and extensible setup, enabling systematic study of the different aspects of QEC realization.*

exemplar of its category. We specifically include both the state-of-the-art surface code and the color code, as the latter has promising properties for practical implementations. Additionally, we include the heavy-hex code to test how a topology-specific code performs under general conditions. The structures of these codes are illustrated in Fig. 4.

3.3 The ECCentric Framework

To experimentally explore our research questions while systematically exploring the important parameters, we now require a high-performance and modular framework that incorporates different compilation stages, topology constraints, and detailed noise models. However, existing tools either support detailed compilation and execution (e.g., Qiskit [73]) or are fast and memory-efficient enough for QEC simulation (e.g., Stim [49]), with no single framework fully meeting our needs. To address this gap, we develop ECCentric, our dedicated, fully modular, extensible framework for systematic evaluation of the QEC codes, which we present in Fig. 5.

Encoded quantum memory generation. We first use pre-existing libraries [49–51, 55, 86, 99] to generate quantum memory circuits, i.e., circuits representing one or more logical qubits encoded using our selected codes, not supporting FT logical gates, as discussed in § 7. Generation allows arbitrary error-correction rounds and flexible code distance. For the surface, color, Bacon–Shor, and heavy-hex codes, we only generate odd-distance instances to avoid overhead or complications from majority-voting syndrome extraction [27, 30, 46], while the gross code has a fixed distance of 12 [24]. We further extend the Steane code with an additional concatenation layer to produce a [[343,1,27]] code and provide it at distances 3, 9, and 27. We generate all circuits in the Stim format [49] and subsequently translate them into Qiskit [73] to enable execution.

Translation. Next, circuits are transpiled to the native gate sets of target devices using high-performance translators, including Qiskit [73], TKET [120], and BQSKit [137] translators, selected for their performance [97]. By default, we only use the translation functionalities of these SDKs. For full simulation, the circuits must conform to the gate set supported by Stim, so we perform an additional translation during the mapping and routing stage to guarantee that any gates not supported by Stim are correctly converted before execution.

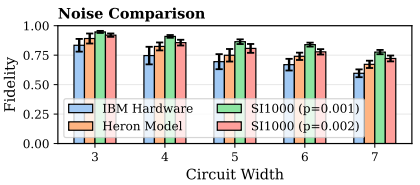


Fig. 6. Comparison of real hardware, the Heron noise model, and standard artificial noise models. *Greater similarity to hardware indicates a more realistic model. We run 10 circuits per width with 1000 shots and report mean fidelity relative to a noiseless simulation.*

Mapping and routing. We define all backends as custom artificial devices based on Qiskit's BackendV2 [73], specifying their topology and, where applicable, qubit properties, such as T_1 and T_2 times. We support both synthetic topologies and models of real devices: Google Willow [3], Quantinuum Apollo [105], Infleqtion [111], IBM Flamingo [107], and IBM Nighthawk [108]. To enable execution of codes, we scale some devices up to reflect mid-term projections, such as Google Willow (3×,

315 qubits) and Infleqtion (16×, 384 qubits), while limiting others like Apollo to 768 qubits for simulation feasibility.

For devices supporting mid-circuit movement, we model shuttling by adding temporary, full-connectivity links with distinct durations and error rates to represent a qubit moving to its target for a remote gate and then returning. We then use the Qiskit Transpiler [109] to determine the initial layout and perform routing while also exploring all available parameters: the initial layout strategies ("Trivial," "Dense," "Lookahead," and "Sabre") and routing algorithms ("Basic," "Stochastic," and "Sabre"). We also implement a qubit tracker to maintain the mapping between physical and logical qubits throughout the circuits, to ensure accurate noise modeling throughout this process.

Representation change. To enable efficient stabilizer simulation, we translate the circuits into the Stim format. Since Qiskit circuits do not include Stim-specific gates (e.g., MRX), we represent these operations as sequences of standard gates in the resulting Stim circuits. We verify the correctness of this translation both manually and automatically across a range of code samples.

Noise addition. After translation, we inject noise gate-by-gate to ensure compatibility with our stabilizer simulation. Our framework supports ① single- and two-qubit gate errors, ② idle errors (modeled either with a constant probability or derived from qubit-specific T_1/T_2 times and gate durations), ③ leakage with propagation, ④ crosstalk, ⑤ reset errors, ⑥ readout errors, ⑦ qubit shuttling errors (modeled as decoherence proportional to movement time [20]), and ⑧ inter-chip gate errors, extending the approach of [54]. We provide both standard models, such as a modified SI1000 [54], and realistic, device-specific models for platforms including Infleqtion, Google Willow, IBM Flamingo, and Quantinuum Apollo (Tab. 1). Fig. 6 illustrates how closely our framework can approximate an actual device.

Stabilizer simulation. Finally, we run a stabilizer simulation using Stim [49], which tracks only the stabilizers rather than full quantum states, allowing efficient, exact simulation without approximations, to sample how physical errors propagate and which syndromes they trigger. We compile these samples into detector error models [39] and feed them to a decoder, which infers the most likely error pattern and applies corrections. We provide two general-purpose decoders: minimum-weight perfect matching (MWPM) [64] and belief propagation with order statistic decoding (BP-OSD) [113], selected for their effectiveness across all codes we study and their well-established open-source implementations. We describe the decoders and justify our choices in more detail in Appendix 6.1. The logical error rate is then estimated as the fraction of runs where errors remain uncorrected.

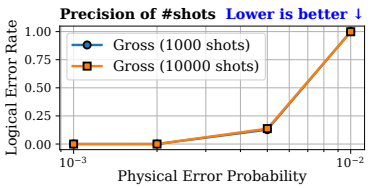


Fig. 7. Effect of the number of shots on logical error rate precision. *Logical error rate of the gross code under the SI1000 noise model for 1000 and 10000 shots.*

Experimental setup. Unless stated otherwise, we use circuits at the maximal code distance supported by the topology, with a number of error-correction rounds equal to the code distance. We run each experiment for 1000 shots, giving confidence intervals of ≈0.004 for logical error rates expected at error probabilities around 0.004, which are at most this value in low-error regimes, while higher rates indicate code failure. This shot count is sufficient to distinguish these regimes and assess the suitability of each code without incurring high computational cost. In Fig. 7, we show that for the gross code, which has the most gates and therefore experiences more opportunities for errors, the logical error rates for 1000 and 10000 shots overlap, indicating that 1000 shots already provide sufficient statistical resolution across this error range.

Table 4. Ratio of logical to physical qubits and number of two-qubit gates based on our implementations of the QEC codes. *Distance 11 for the majority of codes, 12 for the gross code, and 9 for the concatenated Steane with 3 syndrome measurement rounds.*

	Surface	Bacon-Shor	BB (Gross)	C. Steane	Color Code	Heavy-Hex
Net Enc. Rate	$\frac{1}{2d^2-1}$	$\frac{1}{d^2}$	$\frac{k}{2n}$ $(\frac{1}{24})$ [24]	$\frac{1}{2\cdot 7^m}$	$\frac{4}{(3d-1)^2}$ [30]	$\frac{2}{(5d^2-2d-1)}$ [31]
#Phys. Qubits	274	121	288	98	181	291
#2Q	1320	1320	2592	961	1440	1920

4 Hardware Restriction Analysis

In this section, we present a detailed effectiveness analysis of the chosen QEC codes across the hardware-related dimensions: topology size (§ 4.1), topology connectivity (§ 4.2), variance of qubit quality (§ 4.3), distributed device (§ 4.4), and different realistic quantum technologies (§ 4.5).

4.1 Topology Size

We examine how the performance of QEC codes scales with the number of physical qubits. The scale of quantum hardware is rapidly increasing, with roadmaps projecting growth from hundreds of qubits on current devices, such as Google's Willow [3] and IBM's Flamingo [107], to thousands on future Quantinuum systems [105]. This expansion in qubit count enables the usage of larger codes. Conventionally, a larger code distance is expected to yield superior error suppression, as it directly determines the maximum number of correctable errors [112]. Tab. 4 lists the encoding rate of each code, and Fig. 8 shows the maximum possible distance for the Surface and Bacon-Shor codes on devices with 20, 40, and 60 qubits.

Research question and hypothesis. This leads to the question: **RQ# 1**: *Does a QEC code's effectiveness change as its distance increases?* Our hypothesis is that a higher distance is always more beneficial, and thus, the decision on how many qubits to use will lead to a trade-off between using the entire QPU for the best protection of one qubit or having more qubits under less severe protection. Fig. 8 illustrates a key resource trade-off using a 60-qubit device. With the surface code, one could implement a single distance-5 patch or three distance-3 patches. With the Bacon-Shor code, the options include one distance-7 patch, two of distance-5, or six of distance-3.

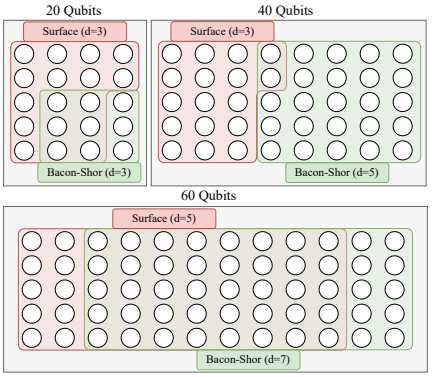


Fig. 8. QEC codes possible on different device sizes. *An overview of the attainable distances for the surface code (in red) and the Bacon–Shor code (in green) on systems with 20, 40, and 60 qubits.*

Methodology. To investigate this trade-off, we simulate our selected QEC codes with 1000 shots at their maximum possible distance, corresponding to the distance number of rounds on an idealized, all-to-all connected 300-500 qubit backend. For comparison, we include the fixed-distance gross and concatenated Steane ([[49,1,9]]) codes as baselines. The simulations employ two error models: the SI1000 model and a more severe uniform error model with an error probability of 0.004.

Results. Our results in Fig. 9 show that a code's distance provides surprisingly little benefit. We observe no consistent improvement with distance; a larger distance on average corresponds to a slight increase in the logical error rate. Notably, at this error probability, only surface and gross code remain effective.

Analysis. We repeat the experiment at two additional error probabilities, 0.002 and 0.008, to examine whether the effect persists closer to and farther from the codes' thresholds. Across all four

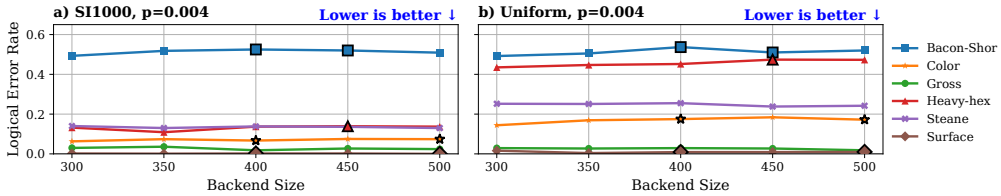


Fig. 9. Logical error rates of different QEC codes at the maximal available code distance across backend sizes under two noise models of error probability 0.004. *Highlighted points mark an increase in the maximal achievable code distance.*

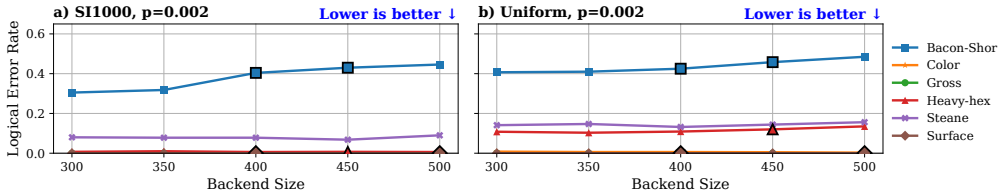


Fig. 10. Logical error rates of different QEC codes at the maximal available code distance across backend sizes under two noise models of error probability 0.002. *Highlighted points mark an increase in the maximal achievable code distance.*

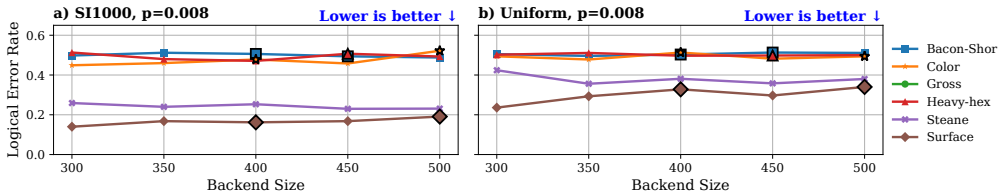


Fig. 11. Logical error rates of different QEC codes at the maximal available code distance across backend sizes under two noise models of error probability 0.008. *Highlighted points mark an increase in the maximal achievable code distance.*

codes and both probabilities, distance variations rarely coincide with the largest deviations from the mean, confirming no consistent trend. At 0.002, all codes except Bacon-Shor perform better, achieving logical error rates of at most 0.156. Color code and gross code produce almost error rate near zero, for surface code we observe no errors. At 0.008, performance degrades across all codes; surface code holds up best under both models, though its logical error rate still exceeds 10%.

Takeaway #1: Our experiments show that increasing code distance does not consistently improve logical performance. The change, on average, raises logical error rates by around 0.012 due to the overhead of additional qubits and gates. This suggests that once a code shows effectiveness, we can use the available device scale to run larger protected circuits instead of raising the distance, enabling more complex algorithms on mid-term hardware.

4.2 Topology Connectivity

Qubit connectivity is a critical factor for QEC because certain hardware topologies (e.g., superconducting) require noisy SWAP gates for routing, which significantly increases error rates [93]. For instance, Figure 12 shows that mapping the Steane code to sparse topologies such as grids or cubes requires additional, error-inducing SWAP gates, an overhead that a fully connected architecture entirely avoids.

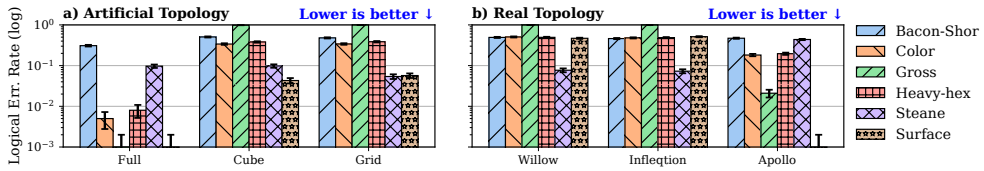


Fig. 13. Logical error rates of different QEC codes under the SI1000 noise model across topologies with varying connectivity. *a) Artificial topologies of 300 qubits with different connectivities and error probability 0.002. b) Realistic topologies with error probability 0.004.*

Research question and hypothesis. These observations raise the following question: **RQ# 2**: *Does higher qubit connectivity always improve the effectiveness of QEC codes?* We hypothesize that higher qubit connectivity reduces SWAP overheads: while a Steane code circuit requires only 49 two-qubit gates on a fully connected device, mapping it to sparser topologies with the Qiskit transpiler [109] adds 54 extra CNOTs on a cube layout and 75 on a grid layout. Since two-qubit gates are a dominant error source [2, 3, 104], this routing overhead will degrade code performance.

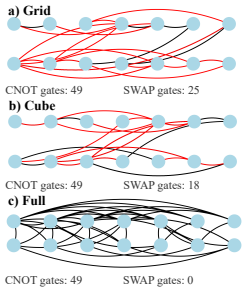


Fig. 12. Connectivity graphs of Steane code $[[7, 1, 3]]$ circuit. *Circuit mapped and routed to various topologies. Black links indicate CNOT connections; red links indicate SWAP connections. Overlapping connections prioritize the noisier SWAP.*

Methodology. To isolate and quantify the impact of connectivity, we simulate each QEC code at its maximum feasible distance on three artificial 300-qubit topologies: a 15×20 grid, a 5×6×10 cuboid, and a fully connected graph. To minimize confounding effects from noise, we use an SI1000 model with a low error probability of 0.002.

Results. The logical error rates, summarized in Fig. 13(a), reveal two distinct trends. First, transitioning from a 2D grid to a 3D cuboid topology yields inconsistent benefits; while the surface code improves by 24.56%, the concatenated Steane performs 81.48% worse. In contrast, adopting a fully connected topology yields substantial and universal improvements, reducing logical errors by an average of 81.92% from a grid. Notably, under full connectivity, we observe no errors for the surface code and the gross code.

Analysis. To validate our findings on realistic hardware, we simulate performance on Google Willow, Quantinuum Apollo, and Infleqtion topologies. Using an SI1000 noise model with a two-qubit error probability of 0.004, the results in Fig. 13(b) confirm that connectivity remains the dominant factor. Apollo, which supports shuttling, achieves an average logical error rate of 0.22, compared to 0.40 on more constrained layouts. The concatenated Steane code delivers the best overall performance with an average error of 0.13; however, we observe only a single error on the surface code on Apollo. Topology dependence is most evident in the gross code: it performs well on Apollo (0.021) but fails on the other platforms.

Takeaway #2: We confirm that connectivity is a crucial factor in the effectiveness of QEC. While transitioning from a 2D grid to a 3D cuboid topology offers inconsistent benefits, moving to an idealized fully connected layout is highly effective, reducing the logical error rate by 81.92% on average. However, such fully connected topologies are physically unrealistic for superconducting qubits due to frequency collisions and crosstalk [138]. Hence, realistic architectures rely on partial-connectivity mechanisms, such as qubit shuttling, which approximate full connectivity over time at the cost of additional latency and noise from qubit movement. With realistic device models, we observe that shuttle capabilities improve performance by an average of 45%. We find that codes exhibit varying sensitivity to these constraints and confirm that high connectivity is essential to unlock the full potential of QEC.

4.3 Variance of Qubit Error Probabilities

We next investigate the impact of non-uniform physical qubit quality on QEC performance. While qubits in trapped-ion and neutral atoms systems are largely uniform [28, 115], superconducting qubits exhibit significant heterogeneity in their decoherence (T_1) and dephasing (T_2) times [125] as well as their readout errors rates [124]. Consequently, relying on device-wide average error rates can be misleading, as a code's performance may be dictated by the specific, lower-quality qubits on which it is executed.

Fig. 14 illustrates T_2 and gate lengths on IBM Heron's `ibm_pittsburgh` [70]. We consider T_2 because it is shorter than T_1. The mean T_2 is 330.14 μs and the median 336.16 μs. However, with the standard deviation of 142.84 μs, there are also qubits of significantly lower lifetimes.

Research question and hypothesis. We raise a question **RQ# 3**: *Is mean qubit quality enough to characterize expected error rates in heterogeneous qubit devices, or does variance also play a critical role?* We hypothesize that reducing the qubit quality variance improves QEC performance by making error patterns more predictable. We show four possible mappings of a distance-3 surface code onto the `ibm_pittsburgh` device in Fig. 14. The choice of physical qubits significantly impacts the average lifetime of the logical qubit; the mean T_2 times for the mappings are 253.00 μs, 355.88 μs, 292.45 μs, and 373.63 μs.

Methodology. We simulate the QEC codes on a 399-qubit all-to-all topology using the IBM Heron noise model to isolate the effects of qubit variability from device size and connectivity constraints. Qubit quality is modeled by sampling T_1 and T_2 values from normal distributions calibrated to IBM Heron data. We explore three variance regimes with standard deviations of 0 μs (none), 60 μs (low), 120 μs (medium), and 180 μs (high), and additionally consider a tenfold prolongation of T_1 and T_2 to study the codes under reduced-noise conditions.

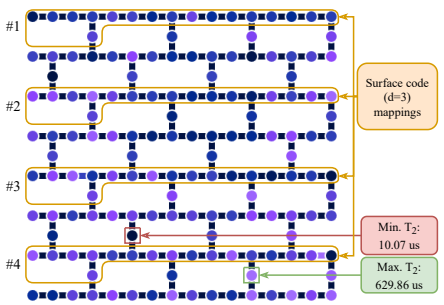


Fig. 14. Qubit topology of the IBM Heron r3 `ibm_pittsburgh` QPU. *Topology of the* `ibm_pittsburgh` *device with qubit* T_2 *dephasing times (μs) shown by color intensity, based on the calibration of 2025-08-29, 07:56:08 [70]. Example mappings of a distance-3 surface code are highlighted with orange frames.*

Results. Fig. 15(a) shows that increasing qubit lifetime variance does not systematically affect logical error rates, with most codes varying by less than 9%. To verify that this lack of correlation is not due to codes being already ineffective under baseline noise, we repeat the simulations with T_1 and T_2 values scaled tenfold to reduce noise, as shown in Fig. 15(b). Even in this high-qubit-lifetime regime, no consistent trend emerges: logical error rates fluctuate non-monotonically, with pronounced oscillations, e.g., the color code's error decreases by 69.64% from mid to high variance.

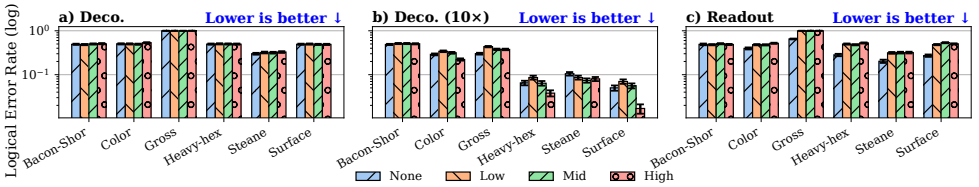


Fig. 15. Influence of qubit quality and readout error variability on the QEC codes. *Comparison of logical error rates on an all-to-all topology for: (a) realistic qubit T_1 and T_2 sampled from normal distributions based on IBM Heron parameters at various variance levels, (b) tenfold prolonged T_1 and T_2, and (c) measurement errors with varying variance from IBM Heron.*

Analysis. However, these results may stem from the relatively small overall influence of decoherence errors. To further explore this effect, we investigate the impact of varying readout errors, another commonly fluctuating source of noise. We model the readout errors using a logarithmic distribution with standard deviations of 0.0, 0.4, 0.7, and 1.0, chosen to best represent the spread of errors observed on the IBM Torino device. As shown in Fig. 15(c), we again observe no clear correlation, with only the Steane code displaying a steady increase in logical error rates across the variation levels (from 0.30 to 0.319).

Takeaway #3: We show that variations in qubit lifetime and readout error do not systematically affect the performance of QEC codes: no consistent correlation is observed, and the average magnitude of differences is 0.02. This is consistent with an important feature of many QEC codes, particularly topological codes as discussed in §3.2, i.e., an immunity to local errors. As long as individual qubits remain below the code's error threshold, local variations in qubit quality have a negligible impact. Our study suggests that incorporating qubit-variance-aware or readout-variance-aware code selection and mapping is unnecessary, which can simplify compiler design.

4.4 Distributed Topology

Next, we explore the effectiveness of QEC codes on distributed devices. Due to scalability limitations of monolithic QPUs, several companies are developing architectures in which multiple QPUs are connected via noisier and slower long-range couplers [107, 117]. For instance, IBM Flamingo consists of three connected IBM Heron QPUs, as we show in Fig. 16. The two-qubit gate error rate for inter-QPU connections is $\sim 10\times$ higher than an intra-QPU connection.

Since the code distance determines the number of corrections [112], we might expect that increasing the distance across distributed QPUs should enhance fault tolerance. However, our study in § 4.1, combined with the noisy nature of inter-QPU connections, challenges this intuition.

Research question and hypothesis. Our next question is **RQ#4:** *Does the effectiveness of a QEC code scale across the QPUs of a distributed architecture similar to when the code runs on a single QPU?* Our results in § 4.3 hint that distributing a QEC code across QPUs will not degrade its performance. We motivate this in Fig. 16, where we scale a surface code from distance $d=3$ on a single 20-qubit QPU to $d=5$ across QPUs, which requires using a few, noisier inter-QPU links. We treat them as high-error outliers, analogous to the low-quality qubits in our previous study, and thus predict that their impact on the logical error rate will be limited.

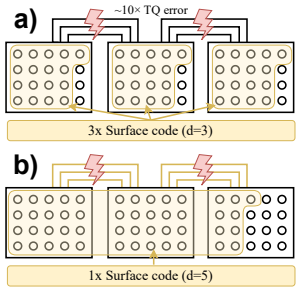


Fig. 16. QEC code placement on a distributed device. *(a) Surface code (d=3) circuits on individual QPUs. (b) Surface code (d=5) circuit spanning the full device.*

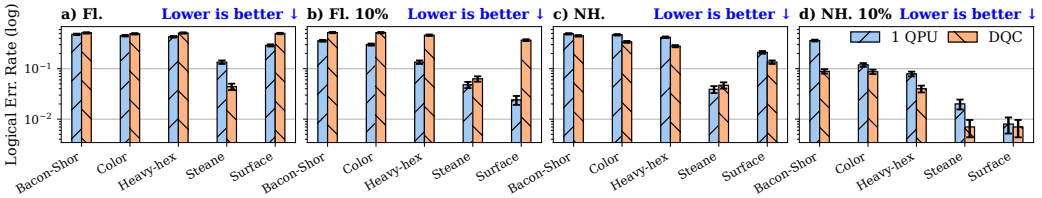


Fig. 17. Performance of QEC codes on a single QPU vs. on the entire DQC architecture of IBM Flamingo (Fl.) and IBM Nighthawk (NH.) with original and tenfold downscaled noise.

Methodology. We test whether this intuition holds on the real-life IBM Flamingo topology. To do so, we compare logical error rates achieved by the QEC codes with their maximal possible distance and with distance limited by the size of a single QPU.

Results. In Fig. 17(a), we compare logical error rates for codes executed on a single QPU versus across the full distributed topology (the gross code cannot be run on a single QPU due to its size, so we exclude it). Single-QPU execution performs substantially better, with an average improvement of 13.27%. The surface code benefits the most (41.88%), while the concatenated Steane code is the only code that performs better on the distributed topology, since its effective distance remains unchanged. Because high error rates render most codes ineffective, Fig. 17(b) shows the same comparison under the IBM Flamingo noise model with error rates reduced by a factor of 10. Here, the single-QPU advantage grows to 55.15%, with the surface code again gaining the most (88.63%) and the concatenated Steane code the least (23.81%). These results suggest that limited inter-QPU connectivity, rather than noise levels, primarily drives the performance gap.

Analysis. To test the role of connectivity, we explore the performance on IBM Nighthawk [108], which has the same error rates as Flamingo but higher connectivity (six neighbors per qubit) and the same quality of inter- and intra-QPU couplers. In Fig. 17(c), only the Steane code performs better on a single QPU under normal noise, and under scaled noise (Fig. 17(d)), no code does. All other codes benefit slightly from the larger distributed system, with average gains of just 0.076 (normal) and 0.072 (scaled).

Takeaway #4: We find that QEC codes on a single QPU perform 34.21% better on average than on distributed QPUs with limited inter-QPU connectivity. Our results show that both stronger and more numerous inter-QPU couplers are critical for achieving fault tolerance in distributed architectures.

4.5 Quantum Technologies

To conclude hardware analysis, we investigate how different quantum technologies support the use of QEC codes on mid-term devices. Since each technology realizes qubits in a distinct way, this leads to heterogeneous noise models, gate speeds, and coherence times (§ 2), as shown in Fig. 18.

Research question and hypothesis. This leads us to ask **RQ#5:** *Which quantum technologies are most suitable for effective error correction, given their error rates and hardware restrictions?* We hypothesize that QEC codes are most effective on platforms

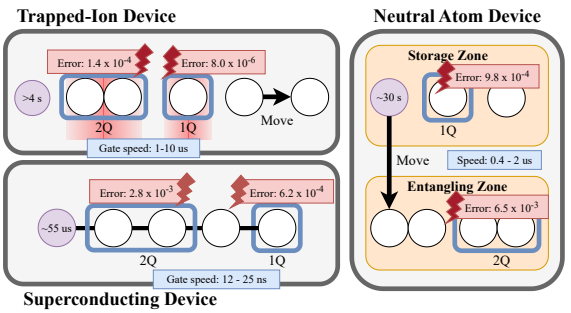


Fig. 18. Overview of quantum technologies. *Gate error rates, coherence times, and gate speeds based on Tab. 1.*

that support shuttling, since enhanced connectivity plays a decisive role in error correction performance as we show in § 4.2. Among these, trapped-ion devices appear particularly promising due to their long coherence times. However, noise rates in realistic mid-term hardware remain above most QEC thresholds, limiting the potential for full fault tolerance. Trapped-ion and neutral atom QPUs support mid-circuit movement and offer a higher ratio of gate speed to coherence time, while superconducting and neutral atoms devices show similar two-qubit error rates, with trapped-ion systems providing an order-of-magnitude improvement. Thus, we expect the mid-term noise levels to be too high for effective QEC.

Methodology. We simulate QEC codes at the maximum distance on mid-term devices for each technology: Google Willow, Quantinuum Apollo, and Infleqtion. We use noise models that capture their corresponding error rates. For devices supporting shuttling, we compare performance with and without it.

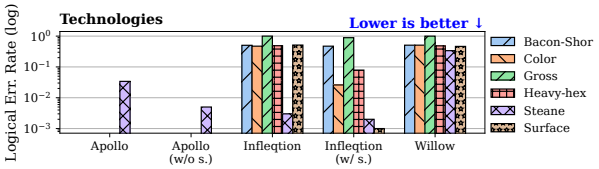


Fig. 19. Effectiveness of codes on realistic mid-term devices, both w/ and w/o mid-circuit qubit movement.

Results. Fig. 19 shows the logical error rate results. Quantinuum Apollo performs best, with average rates of 0.833×10^{-3} without shuttling and 5.67×10^{-3} with, followed by Infleqtion at 4.97×10^{-1} without shuttling and 2.45×10^{-1} with. The gross code fails in all cases except Apollo, where high connectivity and low error rates enable full error suppression. Across technologies, the concatenated Steane code performs best on average (1.11×10^{-1}), although we still observed some errors on Apollo, likely due to the chosen concatenation level. For Apollo, only the concatenated Steane code degrades with shuttling, while all other codes maintain complete reduction. On average, shuttling improves performance from about 0.249 to 0.125, nearly halving the error rate. Projected trapped-ion error rates appear sufficient for practical QEC, as most codes eliminate logical errors on Apollo. On Infleqtion with shuttling, the surface code reaches 1.0×10^{-3}. Although current superconducting devices perform poorly, Fig. 17 shows that a tenfold reduction in IBM Flamingo's physical error rate enables its surface code to reach 9.0×10^{-3}. Our results indicate that if industry roadmaps [105, 107] are met, fault tolerance may be achievable within five years.

Takeaway #5: We observe that shuttling can nearly halve logical error rates. Among the platforms we study, Quantinuum Apollo (trapped-ion) performs the best, often presenting no errors in the sample, while on Infleqtion with shuttling, the surface code brings the logical error rate down to 1.0×10^{-3}. Our results hint that future low-error, high-connectivity hardware will enable near-complete error suppression, putting fault-tolerant QEC within reach.

5 Framework Analysis

In this section, we analyze how certain compilation stages, such as mapping and routing (§ 5.1) and translation to the target gate set (§ 5.2), affect the effectiveness of the protection offered by the chosen QEC codes. Here, we aim to provide insights into the error overhead that must be accounted for when selecting a QEC code for a QPU with a specific noise model and available compiler.

5.1 Mapping and Routing

We investigate the impact of mapping and routing on QEC codes' performance, since these compilation stages substantially increase the number of noisy two-qubit gates [93]. Fig. 20 shows this,

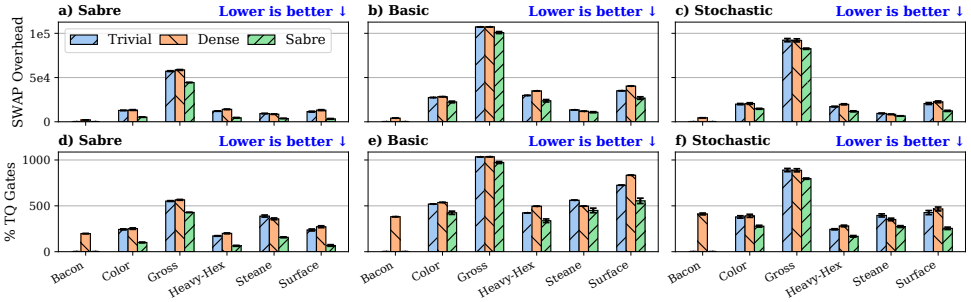


Fig. 21. SWAP overhead (a-c) and SWAP overhead as percentage of original two-qubit gates (d-f) introduced for different QEC codes and routing methods on grid architecture.

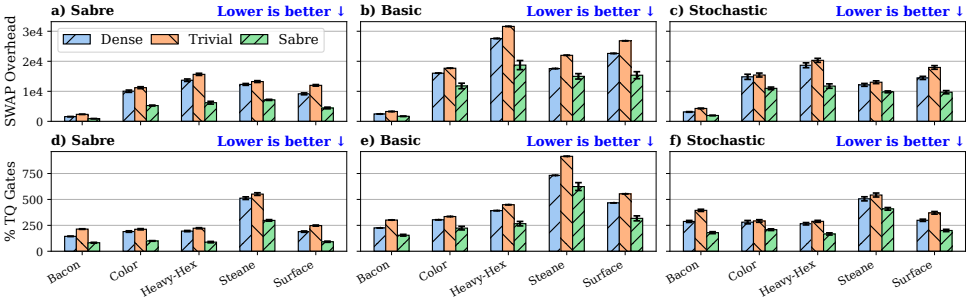


Fig. 22. SWAP overhead (a-c) and SWAP overhead as percentage of original two-qubit gates (d-f) introduced for different QEC codes and routing methods on heavy-hex architecture.

where we map a circuit on a grid topology (a-b). The routing step that follows the mapping adds SWAP gates to resolve the connectivity constraints of the topology (c).

Research question and hypothesis. This motivates the research question: **RQ#6:** *How do the mapping and routing compilation stages influence the effectiveness of QEC codes?* We hypothesize that when mapping and routing heuristics limit SWAP overheads, QEC effectiveness increases. Since many quantum devices do not support SWAPs natively [2, 104], a SWAP is decomposed into three two-qubit gates, effectively tripling the error probability (Fig. 20(c)).

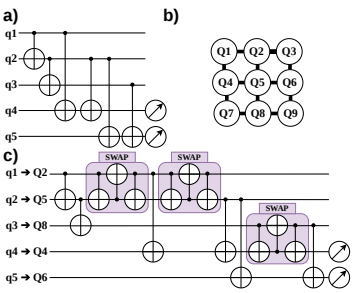


Fig. 20. Circuit mapping and routing on an exemplary topology. *(a) Repetition code (d=3) (b) Grid topology (c) Exemplary mapping of logical to physical qubit, which necessitates additional SWAP gates.*

Methodology. We compile each code 1000 times using all combinations of initial layout strategies and routing heuristics onto a 17×17 grid and the 193-qubit heavy-hex topology, where the gross code is excluded due to prohibitive runtime.

Results. Figures 21(a–c) and 22(a–c) show the SWAP overheads. We exclude the Lookahead mapping from our experiments due to its prohibitively long execution times on larger circuits and topologies. Across all codes, we find that SABRE yields the best results. On average, it introduces 10216.480 SWAP gates per code for grid topology, ranging from 0 for the Bacon–Shor code to 44415.49 for the gross code, and 4789.2756 SWAP gates per code for heavy-hex topology, ranging from 892.139 for the Bacon–Shor code to 7147.208 for the concatenated Steane code. However, even codes typically well-suited for the given topologies, such as the surface code for the grid or the heavy-hex code for the heavy-hex topology, accumulate substantial overhead (3288.283

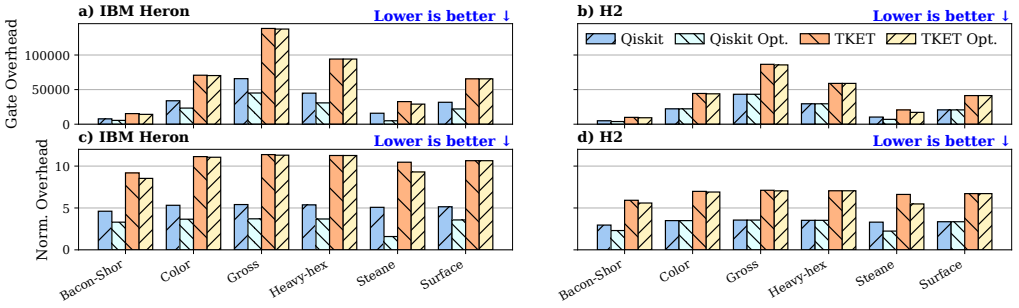


Fig. 24. Gate overhead of converting to IBM Heron and Quantinuum H2 gate sets using TKET and Qiskit.

and 6201.596 additional SWAPs, respectively). Notably, the overhead does not seem consistently proportional across all the codes.

Analysis. Next, we examine how SWAP overhead relates to the number of two-qubit gates in the original circuit, which we show in Figures 21(d–f) and 22(d–f). For the grid topology, SABRE incurs an average overhead of 136.34%, reaching up to 428.39% for the gross code. For the heavy-hex topology, SWAP overhead is around 131.85%. Lastly, we observe that even for codes with similar connectivity needs, the proportions differ significantly. Because of the SWAP decomposition, in fact, we observe an average increase equivalent to 409.02% and 395.55% of the original two-qubit gates, respectively.

Takeaway #6: We show that mapping and routing add 134.095% more two-qubit gates due to SWAP overhead, on average. In practical applications, this corresponds to just over four extra two-qubit gates for every original one, which greatly amplifies two-qubit errors and highlights the need for mapping and routing strategies tailored to QEC to preserve fault tolerance.

5.2 Gateset Translation

We next analyze the error overhead introduced when translating a quantum circuit into a device's native gate set. Each quantum device supports a specific, restrictive set of hardware-level gates [2, 104], requiring compilers to decompose high-level operations into hardware-compliant sequences. Fig. 23 illustrates this process, showing the decomposition required to implement a CNOT gate on a device that does not support it [2].

Research question and hypothesis. This raises a question: **RQ#7:** *How does the translation stage influence the effectiveness of QEC codes?* We hypothesize that different compilers generate hardware-level gate sequences with varying overheads. To illustrate our hypothesis, we consider the example in Fig. 23. The same high-level circuit can be translated in multiple equivalent ways, including with or without optimization. These choices yield different numbers of gates, each contributing to the total error probability.

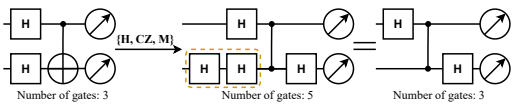


Fig. 23. Example of circuit translation. *Non-optimal and optimal translation into a gate set w/o a CNOT gate.*

Methodology. We compare the number of additional gates introduced when translating each QEC code with maximal distance to the gate sets of IBM Heron and Quantinuum H2 devices. To remove any connectivity constraints, we use a fully connected topology of 300 qubits.

Results. We exclude BQSKit from our results since it removes operations deemed redundant, which interferes with error detection. In Fig. 24(a-b), we show the number of additional gates introduced

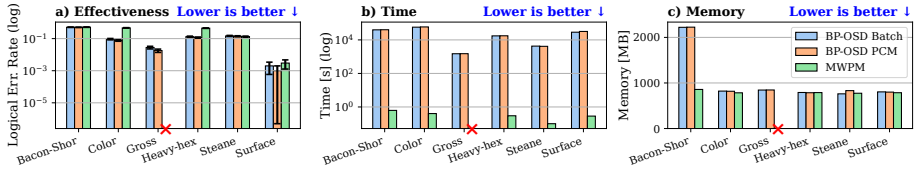


Fig. 26. Comparison of decoders. *a) Effectiveness against the SI1000 noise model with probability 0.004. b) Runtime of the decoders. c) Memory usage of the decoders.*

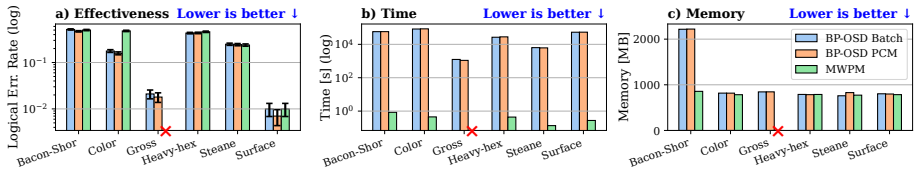


Fig. 27. Comparison of decoders. *a) Effectiveness of different decoders against the uniform noise model with probability 0.004. b) Runtime of the decoders. c) Memory usage of the decoders.*

by translation. The gross code exhibits the highest absolute increase due to its large number of gates. On average, Qiskit introduces 27552 additional gates, while TKET adds over twice as many. At their highest optimization levels, Qiskit reduces the overhead by 21.94%, while TKET at its highest viable level (L2 [103]) achieves just 1.7%. These results demonstrate that optimization mitigates, but cannot fully eliminate, the cost of translation.

Analysis. We investigate whether the overhead is not always proportional to the original gate count. In Fig. 24(c-d), we show that, in contrast to routing, translation overhead scales approximately proportionally with the original gate count. Across all compilers and codes, an average of 6.134 additional gates is introduced per original gate, with TKET contributing 8.704, optimized TKET 8.406, Qiskit 4.261, and optimized Qiskit 3.166. We observe the highest deviation from the average in the Bacon-Shor code, with a normalized overhead of 5.301. This suggests that the majority of codes have similar distributions of gate types, leading to consistent translation scaling.

Takeaway #7: Our investigation shows that even the most optimized translation introduces significant overhead, with an average of 3.166 additional two-qubit gates per original gate. This highlights that, compared to unoptimized circuits, careful compilation is crucial to minimize overhead and limit error accumulation, thereby preserving the effectiveness of QEC codes.

6 Quantum Error Correction Analysis

Finally, we explore the correcting abilities of decoders (§ 6.1) and QEC codes (§ 6.2) themselves.

6.1 Decoders

We examine how the choice of decoder shapes QEC performance, as these algorithms interpret syndrome measurements, predict errors, and propose corrections, with varying accuracy and efficiency [16, 92] (Fig. 25).

Research question and hypothesis. This leads us to raise a question **RQ#8**: *Which decoders achieve the best performance across QEC code families?* We hypothesize that no single decoder consistently dominates, as performance depends on code structure and error propagation.

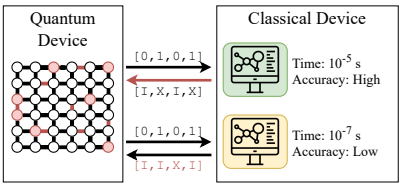


Fig. 25. Decoding algorithms demonstrate a trade-off between speed and accuracy.

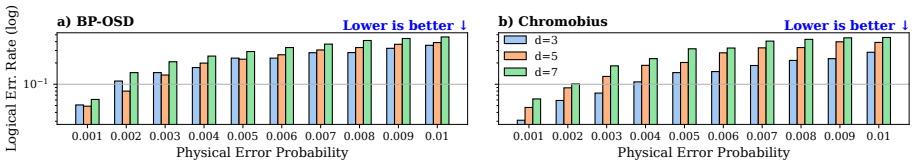


Fig. 28. Difference in logical error rate between BP-OSD and Chromobius for color codes of various distances under the SI1000 error model.

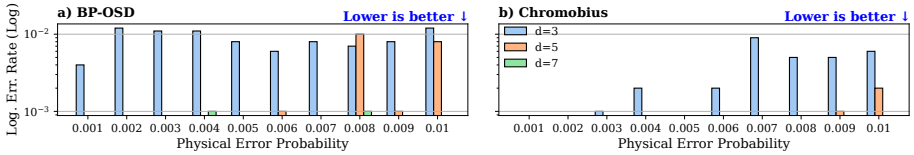


Fig. 29. Difference in logical error rate between BP-OSD and Chromobius for color codes of various distances under the phenomenological error model.

Methodology. We evaluate two open-source decoders: *BP-OSD* [113] and *MWPM* [64], across multiple QEC codes under the SI1000 and the uniform error models ($p = 0.004$). BP-OSD is tested in two variants: parity-check matrix decoding and batched error decoding. In both cases, we use minimum-sum belief propagation with up to 10000 iterations for improved convergence, followed by order-7 OSD, trading increased accuracy for higher computational cost. Each configuration runs on a fully connected 400-qubit topology, using the maximum achievable code distance for each QEC scheme.

Results. Excluding the gross code, for which MWPM fails, the BP-OSD parity-check decoder achieves the best overall performance against the SI1000 noise model across the tested QEC codes, with an average logical error rate of 0.168, outperforming MWPM by 46.66% on average (Fig. 26). MWPM outperforms BP-OSD only for the concatenated Steane code. However, this improved accuracy comes at a substantial computational cost: MWPM requires, on average, only 0.34 s per experiment, whereas BP-OSD requires over 8 h, and MWPM uses 26.89% less memory than the heavier parity-check BP-OSD implementation. Fig. 27 presents the results for the uniform noise model. Here, on average, the BP-OSD decoder with a parity-check matrix achieves the best overall performance across all tested QEC codes, yielding an average logical error rate of 0.2624. It outperforms BP-OSD batch (0.2764) and MWPM (0.3378). Again, the average execution time for BP-OSD algorithms is over 10^5 times longer, while the memory usage is 26.86% smaller for the MWPM than for the less resource-efficient BP-OSD with a parity-check matrix. These results reflect the choice of aggressive BP-OSD settings optimized for decoding accuracy. Reducing the number of BP iterations or the OSD order would significantly shorten the runtime at the expense of performance.

Analysis. General-purpose decoders are essential for cross-code benchmarking but may underperform compared to specialized decoders. We compare the performance of the BP-OSD batch and *Chromobius* [53] decoder tailored for the color code, at distances 3, 5, and 7. Under the SI1000 noise model, Chromobius achieves an average logical error of 0.228 versus BP-OSD's 0.289 (Fig. 28), showing an advantage, but smaller than we anticipated. We further compare the performance of the color code using Chromobius and BP-OSD under a standard, albeit less realistic, phenomenological noise regime, where only noise sources are the depolarization of data qubits between rounds and imperfect measurements [53]. In Fig. 29, we show that under this noise model, Chromobius achieves an average logical error rate of 0.0011, performing approximately 3.3× better than BP-OSD.

Table 5. Observed increase in logical error rate due to decoding.

QEC Code	Bacon–Shor				Heavy–Hex		Con. Steane	
Error prob.	0.004	0.006	0.008	0.010	0.008	0.010	0.001	0.008
ΔLER = Corrected − Uncorrected	+0.003	+0.002	+0.004	+0.005	+0.016	+0.010	+0.014	+0.012

Takeaway #8: BP-OSD achieves the best general decoding performance and, on color code under a realistic noise model, performs only 26.75% worse than a dedicated decoder, making it the most suitable choice for cross-code evaluations.

6.2 Is QEC Essential for Quantum Computing?

We close with a provocative question: is QEC always necessary? Here, we examine errors introduced by the correction procedure itself, highlighting the gap between ideal fault-tolerance and the reality of QPUs suffering from various noise sources (Fig. 30).

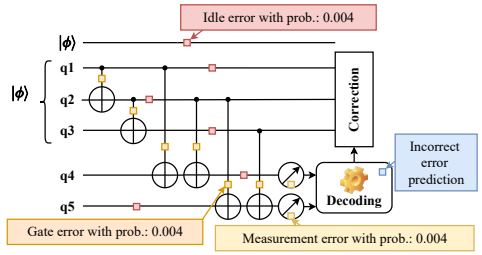


Fig. 30. Example of errors in an unprotected versus a protected qubit. *Comparison of errors in quantum memory w/o protection and w/ a repetition code under an uniform noise model.*

Research question and hypothesis. Thus, we ask: **RQ#9:** *Is the application of QEC always beneficial, or are there regimes in which the overhead of QEC introduces more noise than it suppresses?* We hypothesize that applying QEC can be detrimental as the error probability approaches a code's threshold. Consider a single idle qubit with error probability p_{idle}. Protecting it with a simple repetition code (Fig. 30) introduces additional errors from gates (p_{gate}) and measurements (p_{meas}), yielding a combined error probability of approximately $5 \times p_{idle} + 6 \times p_{gate} + 2 \times p_{meas}$. Once the code's correcting capabilities are exceeded, the correction process itself becomes a net source of faults, making the encoded qubit noisier than the unprotected one. We aim to investigate whether, across different error probabilities and especially near the threshold, this extra overhead causes the logical error rate to grow faster than that of the unprotected qubit.

Methodology. We compare error growth in protected and unprotected circuits by generating the QEC codes at maximal distance, on a fully connected 300-qubit topology, and their corresponding unprotected circuits, consisting of a single idle qubit (12 for the gross code), which we keep idle for as many rounds as their protected counterparts. We add noise using the SI1000 model with two-qubit error probabilities in the 0.001–0.01 range, and record the resulting error rates.

Results. We observe no errors on unprotected qubits at any probability. We present results for protected circuits in Fig. 31. At $p = 0.001$, we observe errors for two codes. At $p = 0.004$, all codes exceed their thresholds. Surface code exhibits the slowest error growth, while the gross code, with the most gates, deteriorates most sharply. We expect that in more constrained topologies, the higher gate count will result in even faster performance degradation.

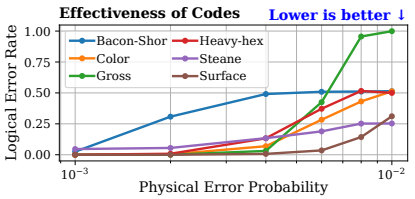


Fig. 31. Logical error rates for different codes under the SI1000 noise model of various probabilities in a realistic setup.

Analysis. To further investigate the introduced overhead, we also isolate the effect of the correction itself. As expected, logical error rates are almost always higher without correction, even after a code's threshold is crossed, confirming the codes' error-suppressing capabilities. However, as shown in Tab. 5, this benefit is not universal: in some cases, incorrect decoder decisions actively degrade performance rather than improve it.

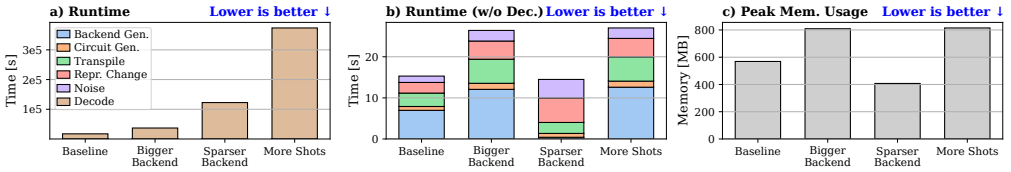


Fig. 32. Runtime per stage and peak memory usage of ECCentric. *Baseline: Running surface code on a fully connected 300-qubit backend (SI1000 noise, error 0.004) with BP-OSD decoder for 1000 shots. Bigger backend: 400 qubits. Sparser backend: grid topology. More shots: 10000 shots.*

Takeaway #9: Our study shows that QEC is not universally beneficial. At $p = 0.001$, we observe no errors only on four of the evaluated codes, and at $p = 0.004$ all codes cross their thresholds, while unprotected qubits still show no observed errors. Although QEC can suppress errors at low rates, once a code's threshold is crossed, its performance deteriorates rapidly, depending on its correcting capabilities and the noise sources in the circuit. These results suggest that QEC should be applied selectively, only where error rates are low or to protect complex circuits rather than quantum memory, to achieve meaningful error suppression.

7 Limitations

ECCentric implementation. Our implementation does not include FT gate constructions, as no general, scalable framework exists across different QEC codes, and practical methods to enable FT gates are still an active area of research aimed at reducing overhead. Once such methods become available, ECCentric will support circuits in this more general FT setting. We also omit realistic decoding delays, since latency depends strongly on the QEC code, decoder, and hardware, making general comparisons difficult. Although we could assume optimistic sub-microsecond latencies, they would have minimal impact and risk introducing code- or hardware-specific bias. Future versions of ECCentric may offer configurable latency models.

ECCentric execution. ECCentric supports exact sampling of QEC-protected quantum memory circuits, but at significant memory and runtime cost. Fig. 32 shows that the decoding stage, during which errors are sampled via stabilizer simulation and a decoding heuristic is applied, dominates runtime, accounting for an average of 99.96% of runtime, as errors are sampled via stabilizer simulation and processed by a decoding heuristic. Since Stim can simulate a surface code ($d = 100$) in 15 s [52], decoding becomes the bottleneck as detection events increase with code distance and additional gates for sparse backends, with shot count determining total runtime. Memory usage remains consistent across execution stages and scales with backend size, connectivity, and shots, reaching up to 815 MB per experiment. Overall, ECCentric execution is constrained by both memory and runtime, which can extend over multiple days.

8 Related Work

Benchmarking in quantum computing. Recent comprehensive surveys [91, 102] highlight the crucial role of benchmarking in guiding the development of the entire quantum computing stack. Established suites now exist for assessing quantum software [87, 110, 126], development kits [97], and physical qubit performance [95]. However, these existing benchmarking studies are QEC-agnostic and thus orthogonal to our work.

Benchmarking in QEC. While systematic evaluations of QEC codes are scarce, existing studies tend to be limited in scope. Much of the prior work focuses on specific QEC-related aspects [29, 82, 123, 128, 136, 139], relies on theoretical analysis with simplified noise models [14, 23, 24, 35, 36, 54, 67, 122], or tests only a single code, topology, or noise model under realistic conditions

[17, 32, 37, 43, 60, 72, 127]. Furthermore, evaluations on real hardware remain rare [45, 69]. In contrast, our work bridges this gap by systematically benchmarking a wide range of QEC codes under realistic, device-specific noise models, while accounting for compilation effects to deliver a comprehensive evaluation of their practical applicability.

Existing frameworks. While several QEC benchmarking frameworks exist, they are often limited. The most established is Stim [49], which enables efficient stabilizer simulations but offers only simple noise models out of the box. Other efforts are either limited in scope [59], deprecated [133], or restricted to specific code families [75, 77]. In contrast, our work builds on Stim's stabilizer simulation to provide a general framework for evaluating a broad spectrum of codes under precise noise models, while incorporating software stack effects and maintaining efficiency.

Surveys. While several surveys of QEC codes exist, they often focus on deep theoretical foundations [47, 74, 94] or target broader audiences [33, 34, 112]. Even the comprehensive Error Correction Zoo [5], a valuable community catalog, uses a non-uniform presentation that hinders direct comparison. In contrast, our work focuses on the practical aspects of QEC and introduces a clear taxonomy designed to highlight the key characteristics and interrelations between different code families, enabling a more direct, comparative analysis.

How our paper differs? This work systematically benchmarks multiple QEC codes with an emphasis on practical application, evaluating them under real devices, compilers, and noise models. To do so, we introduce a framework that integrates diverse codes, architectures, compilers, noise models, and decoders, and is easily extensible as the field evolves. Complementing this, our survey targets a broad audience and introduces a clear taxonomy that highlights code characteristics, overlaps, and differences, clarifying their strengths and constraints.

9 Concluding Remarks

We present a unified benchmarking framework enabling systematic and comprehensive empirical analysis of QEC codes. Our results provide valuable insights into the performance of QEC codes under various hardware- and framework-specific limitations, aiding the pursuit of FT quantum hardware. Future work should investigate factors beyond the scope of this study, including the evaluation of decoders and the translation of quantum circuit gates to their FT counterparts.

Having systematically explored the performance of QEC codes across mid-term devices, topologies, and noise regimes, we now draw practical lessons for hardware design, compilation strategies, and execution of quantum circuits. Our findings highlight which factors matter most for effective error correction and which can be deprioritized.

- **Trapped-ion in the lead:** Predicted trapped-ion devices with shuttling achieve the strongest error suppression, nearly eliminating logical errors. Prioritizing this platform's development is the most direct path to achieving fault-tolerant QEC within five years (**Takeaway #5**).
- **Connectivity outweighs code size:** Enhancing qubit connectivity through features like mid-circuit movement or shuttling dramatically reduces logical errors, proving far more effective than simply increasing code distance (**Takeaways #1, #2, #5**).
- **Variability is less critical:** Variations in qubit or readout quality have minimal impact on logical errors. This suggests that variance-aware mapping or code selection can be de-emphasized, simplifying both compilation and operational planning (**Takeaway #3**).
- **Distributed execution is still challenging:** Single-QPU patches consistently outperform distributed setups due to the limited connectivity of inter-QPU couplers. Increasing the number of links is the key to enabling distributed fault-tolerant QEC (**Takeaway #4**).

- **QEC-aware compilation is crucial:** The compilation process, particularly routing and mapping, can quadruple the two-qubit gate count. QEC-aware compilation is therefore essential to minimize this error-inducing overhead and preserve a code's effectiveness (**Takeaways #6, #7**).
- **QEC is not always helpful:** In our experiments, we found that in regimes where gate errors dominate, applying QEC indiscriminately can actually introduce more errors than it corrects. For near-term devices, this means we need to carefully decide when and where to apply QEC, particularly avoiding protection of idle qubits unless necessary, and focusing on more complex circuits where error correction truly provides a benefit (**Takeaway #9**).

Artifact. ECCentric, along with the entire experimental setup, datasets, and results, is publicly available on Zenodo: https://doi.org/10.5281/zenodo.19241564.

Acknowledgments

We thank Dr. Joschka Roffe for his insights on the taxonomy of QEC codes and Peter Wegmann for his assistance with the implementation. Funded by the Bavarian State Ministry of Science and the Arts as part of the Munich Quantum Valley (MQV), grant number 6090181.

References

[1] 2025. *Topological Quantum Error Correction (TQEC).* https://github.com/tqec/tqec Accessed: 2025-12-17.

[2] Muhammad AbuGhanem. 2025. IBM quantum computers: evolution, performance, and future directions. *The Journal of Supercomputing* 81, 5 (April 2025). https://doi.org/10.1007/s11227-025-07047-7

[3] Rajeev Acharya, Dmitry A. Abanin, Laleh Aghababaie-Beni, Igor Aleiner, Trond I. Andersen, Markus Ansmann, Frank Arute, Kunal Arya, Abraham Asfaw, Nikita Astrakhantsev, Juan Atalaya, Ryan Babbush, Dave Bacon, Brian Ballard, Joseph C. Bardin, Johannes Bausch, Andreas Bengtsson, Alexander Bilmes, Sam Blackwell, Sergio Boixo, Gina Bortoli, Alexandre Bourassa, Jenna Bovaird, Leon Brill, Michael Broughton, David A. Browne, Brett Buchea, Bob B. Buckley, David A. Buell, Tim Burger, Brian Burkett, Nicholas Bushnell, Anthony Cabrera, Juan Campero, Hung-Shen Chang, Yu Chen, Zijun Chen, Ben Chiaro, Desmond Chik, Charina Chou, Jahan Claes, Agnetta Y. Cleland, Josh Cogan, Roberto Collins, Paul Conner, William Courtney, Alexander L. Crook, Ben Curtin, Sayan Das, Alex Davies, Laura De Lorenzo, Dripto M. Debroy, Sean Demura, Michel Devoret, Agustin Di Paolo, Paul Donohoe, Ilya Drozdov, Andrew Dunsworth, Clint Earle, Thomas Edlich, Alec Eickbusch, Aviv Moshe Elbag, Mahmoud Elzouka, Catherine Erickson, Lara Faoro, Edward Farhi, Vinicius S. Ferreira, Leslie Flores Burgos, Ebrahim Forati, Austin G. Fowler, Brooks Foxen, Suhas Ganjam, Gonzalo Garcia, Robert Gasca, Élie Genois, William Giang, Craig Gidney, Dar Gilboa, Raja Gosula, Alejandro Grajales Dau, Dietrich Graumann, Alex Greene, Jonathan A. Gross, Steve Habegger, John Hall, Michael C. Hamilton, Monica Hansen, Matthew P. Harrigan, Sean D. Harrington, Francisco J. H. Heras, Stephen Heslin, Paula Heu, Oscar Higgott, Gordon Hill, Jeremy Hilton, George Holland, Sabrina Hong, Hsin-Yuan Huang, Ashley Huff, William J. Huggins, Lev B. Ioffe, Sergei V. Isakov, Justin Iveland, Evan Jeffrey, Zhang Jiang, Cody Jones, Stephen Jordan, Chaitali Joshi, Pavol Juhas, Dvir Kafri, Hui Kang, Amir H. Karamlou, Kostyantyn Kechedzhi, Julian Kelly, Trupti Khaire, Tanuj Khattar, Mostafa Khezri, Seon Kim, Paul V. Klimov, Andrey R. Klots, Bryce Kobrin, Pushmeet Kohli, Alexander N. Korotkov, Fedor Kostritsa, Robin Kothari, Borislav Kozlovskii, John Mark Kreikebaum, Vladislav D. Kurilovich, Nathan Lacroix, David Landhuis, Tiano Lange-Dei, Brandon W. Langley, Pavel Laptev, Kim-Ming Lau, Loïck Le Guevel, Justin Ledford, Joonho Lee, Kenny Lee, Yuri D. Lensky, Shannon Leon, Brian J. Lester, Wing Yan Li, Yin Li, Alexander T. Lill, Wayne Liu, William P. Livingston, Aditya Locharla, Erik Lucero, Daniel Lundahl, Aaron Lunt, Sid Madhuk, Fionn D. Malone, Ashley Maloney, Salvatore Mandrà, James Manyika, Leigh S. Martin, Orion Martin, Steven Martin, Cameron Maxfield, Jarrod R. McClean, Matt McEwen, Seneca Meeks, Anthony Megrant, Xiao Mi, Kevin C. Miao, Amanda Mieszala, Reza Molavi, Sebastian Molina, Shirin Montazeri, Alexis Morvan, Ramis Movassagh, Wojciech Mruczkiewicz, Ofer Naaman, Matthew Neeley, Charles Neill, Ani Nersisyan, Hartmut Neven, Michael Newman, Jiun How Ng, Anthony Nguyen, Murray Nguyen, Chia-Hung Ni, Murphy Yuezhen Niu, Thomas E. O'Brien, William D. Oliver, Alex Opremcak, Kristoffer Ottosson, Andre Petukhov, Alex Pizzuto, John Platt, Rebecca Potter, Orion Pritchard, Leonid P. Pryadko, Chris Quintana, Ganesh Ramachandran, Matthew J. Reagor, John Redding, David M. Rhodes, Gabrielle Roberts, Eliott Rosenberg, Emma Rosenfeld, Pedram Roushan, Nicholas C. Rubin, Negar Saei, Daniel Sank, Kannan Sankaragomathi, Kevin J. Satzinger, Henry F. Schurkus, Christopher Schuster, Andrew W. Senior, Michael J. Shearn, Aaron Shorter, Noah Shutty, Vladimir Shvarts, Shraddha Singh, Volodymyr Sivak, Jindra Skruzny, Spencer Small, Vadim Smelyanskiy, W. Clarke Smith, Rolando D. Somma, Sofia Springer, George Sterling, Doug Strain, Jordan Suchard, Aaron Szasz, Alex Sztein, Douglas Thor, Alfredo Torres, M. Mert Torunbalci, Abeer Vaishnav, Justin Vargas, Sergey Vdovichev, Guifre Vidal, Benjamin Villalonga, Catherine Vollgraff Heidweiller, Steven Waltman, Shannon X. Wang, Brayden Ware, Kate Weber, Travis

Weidel, Theodore White, Kristi Wong, Bryan W. K. Woo, Cheng Xing, Z. Jamie Yao, Ping Yeh, Bicheng Ying, Juhwan Yoo, Noureldin Yosri, Grayson Young, Adam Zalcman, Yaxing Zhang, Ningfeng Zhu, Nicholas Zobrist, Google Quantum AI, and Collaborators. 2025. Quantum error correction below the surface code threshold. *Nature* 638, 8052 (01 Feb 2025), 920–926. https://doi.org/10.1038/s41586-024-08449-y

[4] Dorit Aharonov and Michael Ben-Or. 1999. Fault-Tolerant Quantum Computation With Constant Error Rate. arXiv:quant-ph/9906129 [quant-ph] https://arxiv.org/abs/quant-ph/9906129

[5] Victor V. Albert and Philippe Faist (Eds.). 2025. *The Error Correction Zoo.* https://errorcorrectionzoo.org/

[6] Salah A. Aly, Andreas Klappenecker, and Pradeep Kiran Sarvepalli. 2006. Subsystem Codes. arXiv:quant-ph/0610153 [quant-ph] https://arxiv.org/abs/quant-ph/0610153

[7] Matthew Amy and Vlad Gheorghiu. 2020. staq—A full-stack quantum processing toolkit. *Quantum Science and Technology* 5, 3 (Jun 2020), 034016. https://doi.org/10.1088/2058-9565/ab9359

[8] Jonas T. Anderson, Guillaume Duclos-Cianci, and David Poulin. 2014. Fault-Tolerant Conversion between the Steane and Reed-Muller Quantum Codes. *Phys. Rev. Lett.* 113 (Aug 2014), 080501. Issue 8. https://doi.org/10.1103/PhysRevLett.113.080501

[9] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G. S. L. Brandao, David A. Buell, Brian Burkett, Yu Chen, Zijun Chen, Ben Chiaro, Roberto Collins, William Courtney, Andrew Dunsworth, Edward Farhi, Brooks Foxen, Austin Fowler, Craig Gidney, Marissa Giustina, Rob Graff, Keith Guerin, Steve Habegger, Matthew P. Harrigan, Michael J. Hartmann, Alan Ho, Markus Hoffmann, Trent Huang, Travis S. Humble, Sergei V. Isakov, Evan Jeffrey, Zhang Jiang, Dvir Kafri, Kostyantyn Kechedzhi, Julian Kelly, Paul V. Klimov, Sergey Knysh, Alexander Korotkov, Fedor Kostritsa, David Landhuis, Mike Lindmark, Erik Lucero, Dmitry Lyakh, Salvatore Mandrà, Jarrod R. McClean, Matthew McEwen, Anthony Megrant, Xiao Mi, Kristel Michielsen, Masoud Mohseni, Josh Mutus, Ofer Naaman, Matthew Neeley, Charles Neill, Murphy Yuezhen Niu, Eric Ostby, Andre Petukhov, John C. Platt, Chris Quintana, Eleanor G. Rieffel, Pedram Roushan, Nicholas C. Rubin, Daniel Sank, Kevin J. Satzinger, Vadim Smelyanskiy, Kevin J. Sung, Matthew D. Trevithick, Amit Vainsencher, Benjamin Villalonga, Theodore White, Z. Jamie Yao, Ping Yeh, Adam Zalcman, Hartmut Neven, and John M. Martinis. 2019. Quantum supremacy using a programmable superconducting processor. *Nature* 574, 7779 (Oct. 2019), 505–510. https://doi.org/10.1038/s41586-019-1666-5

[10] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G. S. L. Brandao, David A. Buell, Brian Burkett, Yu Chen, Zijun Chen, Ben Chiaro, Roberto Collins, William Courtney, Andrew Dunsworth, Edward Farhi, Brooks Foxen, Austin Fowler, Craig Gidney, Marissa Giustina, Rob Graff, Keith Guerin, Steve Habegger, Matthew P. Harrigan, Michael J. Hartmann, Alan Ho, Markus Hoffmann, Trent Huang, Travis S. Humble, Sergei V. Isakov, Evan Jeffrey, Zhang Jiang, Dvir Kafri, Kostyantyn Kechedzhi, Julian Kelly, Paul V. Klimov, Sergey Knysh, Alexander Korotkov, Fedor Kostritsa, David Landhuis, Mike Lindmark, Erik Lucero, Dmitry Lyakh, Salvatore Mandrà, Jarrod R. McClean, Matthew McEwen, Anthony Megrant, Xiao Mi, Kristel Michielsen, Masoud Mohseni, Josh Mutus, Ofer Naaman, Matthew Neeley, Charles Neill, Murphy Yuezhen Niu, Eric Ostby, Andre Petukhov, John C. Platt, Chris Quintana, Eleanor G. Rieffel, Pedram Roushan, Nicholas C. Rubin, Daniel Sank, Kevin J. Satzinger, Vadim Smelyanskiy, Kevin J. Sung, Matthew D. Trevithick, Amit Vainsencher, Benjamin Villalonga, Theodore White, Z. Jamie Yao, Ping Yeh, Adam Zalcman, Hartmut Neven, and John M. Martinis. 2019. Quantum supremacy using a programmable superconducting processor. *Nature* 574, 7779 (01 Oct 2019), 505–510. https://doi.org/10.1038/s41586-019-1666-5

[11] awsQuantum [n. d.]. AWS Bracket. https://aws.amazon.com/braket/. Accessed: 2025-05-05.

[12] azurequantum [n. d.]. Azure Quantum. https://azure.microsoft.com/en-us/products/quantum. Accessed: 2025-05-05.

[13] Dave Bacon. 2006. Operator quantum error-correcting subsystems for self-correcting quantum memories. *Physical Review A* 73, 1 (Jan. 2006). https://doi.org/10.1103/physreva.73.012340

[14] Jeff P. Barnes, Colin J. Trout, Dennis Lucarelli, and B. D. Clader. 2017. Quantum error-correction failure distributions: Comparison of coherent and stochastic error models. *Phys. Rev. A* 95 (Jun 2017), 062338. Issue 6. https://doi.org/10.1103/PhysRevA.95.062338

[15] Daniel Barredo, Vincent Lienhard, Sylvain de Léséleuc, Thierry Lahaye, and Antoine Browaeys. 2018. Synthetic three-dimensional atomic structures assembled atom by atom. *Nature* 561, 7721 (Sept. 2018), 79–82. https://doi.org/10.1038/s41586-018-0450-2

[16] F Battistel, C Chamberland, K Johar, R W J Overwater, F Sebastiano, L Skoric, Y Ueno, and M Usman. 2023. Real-time decoding for fault-tolerant quantum computing: progress, challenges and outlook. *Nano Futures* 7, 3 (Aug. 2023), 032003. https://doi.org/10.1088/2399-1984/aceba6

[17] César Benito, Esperanza López, Borja Peropadre, and Alejandro Bermudez. 2025. Comparative study of quantum error correction strategies for the heavy-hexagonal lattice. *Quantum* 9 (Feb. 2025), 1623. https://doi.org/10.22331/q-2025-02-06-1623

[18] César Benito, Esperanza López, Borja Peropadre, and Alejandro Bermudez. 2025. Comparative study of quantum error correction strategies for the heavy-hexagonal lattice. *Quantum* 9 (Feb. 2025), 1623. https://doi.org/10.22331/q-2025-02-06-1623

[19] Dolev Bluvstein, Simon J. Evered, Alexandra A. Geim, Sophie H. Li, Hengyun Zhou, Tom Manovitz, Sepehr Ebadi, Madelyn Cain, Marcin Kalinowski, Dominik Hangleiter, J. Pablo Bonilla Ataides, Nishad Maskara, Iris Cong, Xun Gao, Pedro Sales Rodriguez, Thomas Karolyshyn, Giulia Semeghini, Michael J. Gullans, Markus Greiner, Vladan Vuletić, and Mikhail D. Lukin. 2023. Logical quantum processor based on reconfigurable atom arrays. *Nature* 626, 7997 (Dec. 2023), 58–65. https://doi.org/10.1038/s41586-023-06927-3

[20] Dolev Bluvstein, Harry Levine, Giulia Semeghini, Tout T. Wang, Sepehr Ebadi, Marcin Kalinowski, Alexander Keesling, Nishad Maskara, Hannes Pichler, Markus Greiner, Vladan Vuletić, and Mikhail D. Lukin. 2022. A quantum processor based on coherent transport of entangled atom arrays. *Nature* 604, 7906 (April 2022), 451–456. https://doi.org/10.1038/s41586-022-04592-6

[21] H. Bombin. 2013. An Introduction to Topological Quantum Codes. arXiv:1311.0277 [quant-ph] https://arxiv.org/abs/1311.0277

[22] H. Bombin and M. A. Martin-Delgado. 2006. Topological Quantum Distillation. *Physical Review Letters* 97, 18 (Oct. 2006). https://doi.org/10.1103/physrevlett.97.180501

[23] H. Bombin and M. A. Martin-Delgado. 2007. Optimal resources for topological two-dimensional stabilizer codes: Comparative study. *Phys. Rev. A* 76 (Jul 2007), 012305. Issue 1. https://doi.org/10.1103/PhysRevA.76.012305

[24] Sergey Bravyi, Andrew W. Cross, Jay M. Gambetta, Dmitri Maslov, Patrick Rall, and Theodore J. Yoder. 2024. High-threshold and low-overhead fault-tolerant quantum memory. *Nature* 627, 8005 (01 Mar 2024), 778–782. https://doi.org/10.1038/s41586-024-07107-7

[25] Sergey Bravyi and Alexei Kitaev. 2005. Universal quantum computation with ideal Clifford gates and noisy ancillas. *Phys. Rev. A* 71 (Feb 2005), 022316. Issue 2. https://doi.org/10.1103/PhysRevA.71.022316

[26] Nikolas P. Breuckmann and Jens Niklas Eberhardt. 2021. Quantum Low-Density Parity-Check Codes. *PRX Quantum* 2 (Oct 2021), 040101. Issue 4. https://doi.org/10.1103/PRXQuantum.2.040101

[27] Peter Brooks and John Preskill. 2013. Fault-tolerant quantum computation with asymmetric Bacon-Shor codes. *Physical Review A* 87, 3 (March 2013). https://doi.org/10.1103/physreva.87.032310

[28] Colin D. Bruzewicz, John Chiaverini, Robert McConnell, and Jeremy M. Sage. 2019. Trapped-ion quantum computing: Progress and challenges. *Applied Physics Reviews* 6, 2 (May 2019). https://doi.org/10.1063/1.5088164

[29] Christopher Chamberland, Tomas Jochym-O'Connor, and Raymond Laflamme. 2017. Overhead analysis of universal concatenated quantum codes. *Physical Review A* 95, 2 (Feb. 2017). https://doi.org/10.1103/physreva.95.022313

[30] Christopher Chamberland, Aleksander Kubica, Theodore J Yoder, and Guanyu Zhu. 2020. Triangular color codes on trivalent graphs with flag qubits. *New Journal of Physics* 22, 2 (Feb. 2020), 023019. https://doi.org/10.1088/1367-2630/ab68fd

[31] Christopher Chamberland, Guanyu Zhu, Theodore J. Yoder, Jared B. Hertzberg, and Andrew W. Cross. 2020. Topological and Subsystem Codes on Low-Degree Graphs with Flag Qubits. *Physical Review X* 10, 1 (Jan. 2020). https://doi.org/10.1103/physrevx.10.011022

[32] Avimita Chatterjee, Subrata Das, and Swaroop Ghosh. 2025. Q-Pandora Unboxed: Characterizing Resilience of Quantum Error Correction Codes Under Biased Noise. *Applied Sciences* 15, 8 (April 2025), 4555. https://doi.org/10.3390/app15084555

[33] Avimita Chatterjee and Swaroop Ghosh. 2024. Magic Mirror on the Wall, How to Benchmark Quantum Error Correction Codes, Overall? . In *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE Computer Society, Los Alamitos, CA, USA, 356–367. https://doi.org/10.1109/QCE60285.2024.00050

[34] Avimita Chatterjee, Koustubh Phalak, and Swaroop Ghosh. 2023. Quantum Error Correction For Dummies . In *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE Computer Society, Los Alamitos, CA, USA, 70–81. https://doi.org/10.1109/QCE57702.2023.00017

[35] Luis Colmenarez, Ze-Min Huang, Sebastian Diehl, and Markus Müller. 2024. Accurate optimal quantum error correction thresholds from coherent information. *Physical Review Research* 6 (10 2024). https://doi.org/10.1103/PhysRevResearch.6.L042014

[36] Andrew W. Cross, David P. Divincenzo, and Barbara M. Terhal. 2009. A comparative code study for quantum fault tolerance. *Quantum Info. Comput.* 9, 7 (July 2009), 541–572.

[37] Dripto M Debroy, Muyuan Li, Shilin Huang, and Kenneth R Brown. 2020. Logical performance of 9 qubit compass codes in ion traps with crosstalk errors. *Quantum Science and Technology* 5, 3 (apr 2020), 034002. https://doi.org/10.1088/2058-9565/ab7e80

[38] Dripto M Debroy, Muyuan Li, Shilin Huang, and Kenneth R Brown. 2020. Logical performance of 9 qubit compass codes in ion traps with crosstalk errors. *Quantum Science and Technology* 5, 3 (apr 2020), 034002. https://doi.org/10.1088/2058-9565/ab7e80

[39] Peter-Jan H. S. Derks, Alex Townsend-Teague, Ansgar G. Burchards, and Jens Eisert. 2024. Designing fault-tolerant circuits using detector error models. arXiv:2407.13826 [quant-ph] https://arxiv.org/abs/2407.13826

[40] Bryan Eastin and Emanuel Knill. 2009. Restrictions on Transversal Encoded Quantum Gate Sets. *Physical Review Letters* 102, 11 (March 2009). https://doi.org/10.1103/physrevlett.102.110502

[41] Laird Egan, Dripto M. Debroy, Crystal Noel, Andrew Risinger, Daiwei Zhu, Debopriyo Biswas, Michael Newman, Muyuan Li, Kenneth R. Brown, Marko Cetina, and Christopher Monroe. 2021. Fault-Tolerant Operation of a Quantum Error-Correction Code. arXiv:2009.11482 [quant-ph] https://arxiv.org/abs/2009.11482

[42] Pau Escofet, Alejandro Gonzalvo, Eduard Alarcón, Carmen G. Almudéver, and Sergi Abadal. 2024. Route-Forcing: Scalable Quantum Circuit Mapping for Scalable Quantum Computing Architectures. In *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 909–920. https://doi.org/10.1109/qce60285.2024.00110

[43] Jihao Fan. 2020. A Performance Analysis of Quantum Low-Density Parity-Check Codes for Correcting Correlated Errors. *International Journal of Theoretical Physics* 59 (12 2020). https://doi.org/10.1007/s10773-020-04630-x

[44] Oscar Ferraz, Bruno Coutinho, Gabriel Falcao, Marco Gomes, Francisco A. Monteiro, and Vitor Silva. 2025. GPU-Accelerated Syndrome Decoding for Quantum LDPC Codes below the 63 μs Latency Threshold. arXiv:2508.07879 [quant-ph] https://arxiv.org/abs/2508.07879

[45] Regina Finsterhoelzl and Guido Burkard. 2022. Benchmarking quantum error-correcting codes on quasi-linear and central-spin processors. *Quantum Science and Technology* 8, 1 (nov 2022), 015013. https://doi.org/10.1088/2058-9565/aca21f

[46] Austin G. Fowler, Matteo Mariantoni, John M. Martinis, and Andrew N. Cleland. 2012. Surface codes: Towards practical large-scale quantum computation. *Phys. Rev. A* 86 (Sep 2012), 032324. Issue 3. https://doi.org/10.1103/PhysRevA.86.032324

[47] Shayan Garani, Priya Nadkarni, and Ankur Raina. 2023. Theory Behind Quantum Error Correcting Codes: An Overview. *Journal of the Indian Institute of Science* 103 (07 2023). https://doi.org/10.1007/s41745-023-00392-7

[48] Yan Ge, Wu Wenjie, Chen Yuheng, Pan Kaisen, Lu Xudong, Zhou Zixiang, Wang Yuhan, Wang Ruocheng, and Yan Junchi. 2024. Quantum Circuit Synthesis and Compilation Optimization: Overview and Prospects. arXiv:2407.00736 [quant-ph] https://arxiv.org/abs/2407.00736

[49] Craig Gidney. 2021. Stim: a fast stabilizer circuit simulator. *Quantum* 5 (July 2021), 497. https://doi.org/10.22331/q-2021-07-06-497

[50] Craig Gidney. 2023. *heavy-hex-demo*. https://github.com/Strilanc/heavy-hex-demo GitHub repository.

[51] Craig Gidney and Dave Bacon. 2023. Less Bacon More Threshold. arXiv:2305.12046 [quant-ph] https://arxiv.org/abs/2305.12046

[52] Craig Gidney and Martin Ekerå. 2021. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum* 5 (April 2021), 433. https://doi.org/10.22331/q-2021-04-15-433

[53] Craig Gidney and Cody Jones. 2023. New circuits and an open source decoder for the color code. arXiv:2312.08813 [quant-ph] https://arxiv.org/abs/2312.08813

[54] Craig Gidney, Michael Newman, Austin Fowler, and Michael Broughton. 2021. A Fault-Tolerant Honeycomb Memory. *Quantum* 5 (Dec. 2021), 605. https://doi.org/10.22331/q-2021-12-20-605

[55] Anqi Gong, Sebastian Cammerer, and Joseph M. Renes. 2024. Toward Low-latency Iterative Decoding of QLDPC Codes Under Circuit-Level Noise. arXiv:2403.18901 [quant-ph] https://arxiv.org/abs/2403.18901

[56] googleQuantum [n. d.]. Google Quantum Computing Service . https://quantumai.google/cirq/google/concepts. Accessed: 2025-05-05.

[57] Daniel Gottesman. 1997. Stabilizer Codes and Quantum Error Correction. arXiv:quant-ph/9705052 [quant-ph] https://arxiv.org/abs/quant-ph/9705052

[58] Lov K. Grover. 1996. A fast quantum mechanical algorithm for database search. arXiv:quant-ph/9605043 [quant-ph] https://arxiv.org/abs/quant-ph/9605043

[59] Thomas Grurl, Christoph Pichler, Jürgen Fuß, and Robert Wille. 2023. Automatic Implementation and Evaluation of Error-Correcting Codes for Quantum Computing: An Open-Source Framework for Quantum Error Correction. https://doi.org/10.48550/arXiv.2301.05731 International Conference on VLSI Design ; Conference date: 09-01-2023.

[60] Mauricio Gutiérrez and Kenneth R. Brown. 2015. Comparison of a quantum error-correction threshold for exact and approximate errors. *Phys. Rev. A* 91 (Feb 2015), 022335. Issue 2. https://doi.org/10.1103/PhysRevA.91.022335

[61] Tianyi Hao, Amanda Xu, and Swamit Tannu. 2025. Reducing T Gates with Unitary Synthesis. arXiv:2503.15843 [quant-ph] https://arxiv.org/abs/2503.15843

[62] Loïc Henriet, Lucas Beguin, Adrien Signoles, Thierry Lahaye, Antoine Browaeys, Georges-Olivier Reymond, and Christophe Jurczak. 2020. Quantum computing with neutral atoms. *Quantum* 4 (Sept. 2020), 327. https://doi.org/10.22331/q-2020-09-21-327

[63] Oscar Higgott and Nikolas P. Breuckmann. 2021. Subsystem Codes with High Thresholds by Gauge Fixing and Reduced Qubit Overhead. *Physical Review X* 11, 3 (Aug. 2021). https://doi.org/10.1103/physrevx.11.031039

[64] Oscar Higgott and Craig Gidney. 2025. Sparse Blossom: correcting a million errors per core second with minimum-weight matching. *Quantum* 9 (Jan. 2025), 1600. https://doi.org/10.22331/q-2025-01-20-1600
[65] Dominic Horsman, Austin G Fowler, Simon Devitt, and Rodney Van Meter. 2012. Surface code quantum computing by lattice surgery. *New Journal of Physics* 14, 12 (dec 2012), 123011. https://doi.org/10.1088/1367-2630/14/12/123011
[66] Dominic Horsman, Austin G Fowler, Simon Devitt, and Rodney Van Meter. 2012. Surface code quantum computing by lattice surgery. *New Journal of Physics* 14, 12 (Dec. 2012), 123011. https://doi.org/10.1088/1367-2630/14/12/123011
[67] Eric Huang, Andrew C. Doherty, and Steven Flammia. 2019. Performance of quantum error correction with coherent errors. *Physical Review A* 99, 2 (Feb. 2019). https://doi.org/10.1103/physreva.99.022313
[68] He-Liang Huang, Dachao Wu, Daojin Fan, and Xiaobo Zhu. 2020. Superconducting quantum computing: a review. *Science China Information Sciences* 63, 8 (15 Jul 2020), 180501. https://doi.org/10.1007/s11432-020-2881-9
[69] Shilin Huang, Kenneth R. Brown, and Marko Cetina. 2024. Comparing Shor and Steane error correction using the Bacon-Shor code. *Science Advances* 10, 45 (2024), eadp2008. https://doi.org/10.1126/sciadv.adp2008 arXiv:https://www.science.org/doi/pdf/10.1126/sciadv.adp2008
[70] IBM Quantum. 2025. *IBM Quantum*. https://quantum.cloud.ibm.com/ Accessed: 29 August 2025.
[71] ibmQuantum [n. d.]. IBM Quantum. https://www.ibm.com/quantum-computing/. Accessed: 2025-05-05.
[72] Pavithran Iyer, Aditya Jain, Stephen D. Bartlett, and Joseph Emerson. 2022. Efficient diagnostics for quantum error correction. *Phys. Rev. Res.* 4 (Dec 2022), 043218. Issue 4. https://doi.org/10.1103/PhysRevResearch.4.043218
[73] Ali Javadi-Abhari, Matthew Treinish, Kevin Krsulich, Christopher J. Wood, Jake Lishman, Julien Gacon, Simon Martiel, Paul D. Nation, Lev S. Bishop, Andrew W. Cross, Blake R. Johnson, and Jay M. Gambetta. 2024. Quantum computing with Qiskit. arXiv:2405.08810 [quant-ph] https://arxiv.org/abs/2405.08810
[74] Akshaya Jayashankar and Prabha Mandayam. 2022. Quantum Error Correction: Noise-Adapted Techniques and Applications. *Journal of the Indian Institute of Science* 103 (09 2022), 1–16. https://doi.org/10.1007/s41745-022-00332-x
[75] Mingyu Kang, Yingjia Lin, Hanwen Yao, Mert Gökduman, Arianna Meinking, and Kenneth R. Brown. 2025. QUITS: A modular Qldpc code circUIT Simulator. (4 2025). arXiv:2504.02673 [quant-ph]
[76] Krishnageetha Karuppasamy, Varun Puram, Stevens Johnson, and Johnson P. Thomas. 2025. A Comprehensive Review of Quantum Circuit Optimization: Current Trends and Future Directions. *Quantum Reports* 7, 1 (Jan. 2025), 2. https://doi.org/10.3390/quantum7010002
[77] Mitsuki Katsuda, Kosuke Mitarai, and Keisuke Fujii. 2024. Simulation and performance analysis of quantum error correction with a rotated surface code under a realistic noise model. *Phys. Rev. Res.* 6 (Jan 2024), 013024. Issue 1. https://doi.org/10.1103/PhysRevResearch.6.013024
[78] Ercüment Kaya, Jorge Echavarria, Muhammad Nufail Farooqi, Aleksandra Swierkowska, Patrick Hopf, Burak Mete, Lukas Burgholzer, Robert Wille, Laura Schulz, and Martin Schulz. 2024. A Software Platform to Support Disaggregated Quantum Accelerators. In *SC24-W: Workshops of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 1646–1653. https://doi.org/10.1109/SCW63240.2024.00205
[79] A.Yu. Kitaev. 2003. Fault-tolerant quantum computation by anyons. *Annals of Physics* 303, 1 (Jan. 2003), 2–30. https://doi.org/10.1016/s0003-4916(02)00018-0
[80] Emanuel Knill and Raymond Laflamme. 1996. Concatenated Quantum Codes. arXiv:quant-ph/9608012 [quant-ph] https://arxiv.org/abs/quant-ph/9608012
[81] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver. 2019. A quantum engineer's guide to superconducting qubits. *Applied Physics Reviews* 6, 2 (June 2019). https://doi.org/10.1063/1.5089550
[82] Yaniv Kurman, Lior Ella, Ramon Szmuk, Oded Wertheim, Benedikt Dorschner, Sam Stanwyck, and Yonatan Cohen. 2024. Benchmarking the ability of a controller to execute quantum error corrected non-Clifford circuits. arXiv:2311.07121 [quant-ph] https://arxiv.org/abs/2311.07121
[83] Nathan Lacroix, Alexandre Bourassa, Francisco J. H. Heras, Lei M. Zhang, Johannes Bausch, Andrew W. Senior, Thomas Edlich, Noah Shutty, Volodymyr Sivak, Andreas Bengtsson, Matt McEwen, Oscar Higgott, Dvir Kafri, Jahan Claes, Alexis Morvan, Zijun Chen, Adam Zalcman, Sid Madhuk, Rajeev Acharya, Laleh Aghababaie Beni, Georg Aigeldinger, Ross Alcaraz, Trond I. Andersen, Markus Ansmann, Frank Arute, Kunal Arya, Abraham Asfaw, Juan Atalaya, Ryan Babbush, Brian Ballard, Joseph C. Bardin, Alexander Bilmes, Sam Blackwell, Jenna Bovaird, Dylan Bowers, Leon Brill, Michael Broughton, David A. Browne, Brett Buchea, Bob B. Buckley, Tim Burger, Brian Burkett, Nicholas Bushnell, Anthony Cabrera, Juan Campero, Hung-Shen Chang, Ben Chiaro, Liang-Ying Chih, Agnetta Y. Cleland, Josh Cogan, Roberto Collins, Paul Conner, William Courtney, Alexander L. Crook, Ben Curtin, Sayan Das, Sean Demura, Laura De Lorenzo, Agustin Di Paolo, Paul Donohoe, Ilya Drozdov, Andrew Dunsworth, Alec Eickbusch, Aviv Moshe Elbag, Mahmoud Elzouka, Catherine Erickson, Vinicius S. Ferreira, Leslie Flores Burgos, Ebrahim Forati, Austin G. Fowler, Brooks Foxen, Suhas Ganjam, Gonzalo Garcia, Robert Gasca, Élie Genois, William Giang, Dar Gilboa, Raja Gosula, Alejandro Grajales Dau, Dietrich Graumann, Alex Greene, Jonathan A. Gross, Tan Ha, Steve Habegger, Monica Hansen, Matthew P. Harrigan, Sean D. Harrington, Stephen Heslin, Paula Heu, Reno Hiltermann, Jeremy Hilton, Sabrina Hong, Hsin-Yuan Huang, Ashley Huff, William J. Huggins, Evan Jeffrey, Zhang Jiang, Xiaoxuan Jin, Chaitali Joshi, Pavol Juhas,

Andreas Kabel, Hui Kang, Amir H. Karamlou, Kostyantyn Kechedzhi, Trupti Khaire, Tanuj Khattar, Mostafa Khezri, Seon Kim, Paul V. Klimov, Bryce Kobrin, Alexander N. Korotkov, Fedor Kostritsa, John Mark Kreikebaum, Vladislav D. Kurilovich, David Landhuis, Tiano Lange-Dei, Brandon W. Langley, Pavel Laptev, Kim-Ming Lau, Justin Ledford, Kenny Lee, Brian J. Lester, Loïck Le Guevel, Wing Yan Li, Yin Li, Alexander T. Lill, William P. Livingston, Aditya Locharla, Erik Lucero, Daniel Lundahl, Aaron Lunt, Ashley Maloney, Salvatore Mandrà, Leigh S. Martin, Orion Martin, Cameron Maxfield, Jarrod R. McClean, Seneca Meeks, Anthony Megrant, Kevin C. Miao, Reza Molavi, Sebastian Molina, Shirin Montazeri, Ramis Movassagh, Charles Neill, Michael Newman, Anthony Nguyen, Murray Nguyen, Chia-Hung Ni, Murphy Y. Niu, Logan Oas, William D. Oliver, Raymond Orosco, Kristoffer Ottosson, Alex Pizzuto, Rebecca Potter, Orion Pritchard, Chris Quintana, Ganesh Ramachandran, Matthew J. Reagor, Rachel Resnick, David M. Rhodes, Gabrielle Roberts, Eliott Rosenberg, Emma Rosenfeld, Elizabeth Rossi, Pedram Roushan, Kannan Sankaragomathi, Henry F. Schurkus, Michael J. Shearn, Aaron Shorter, Vladimir Shvarts, Spencer Small, W. Clarke Smith, Sofia Springer, George Sterling, Jordan Suchard, Aaron Szasz, Alex Sztein, Douglas Thor, Eifu Tomita, Alfredo Torres, M. Mert Torunbalci, Abeer Vaishnav, Justin Vargas, Sergey Vdovichev, Guifre Vidal, Catherine Vollgraff Heidweiller, Steven Waltman, Jonathan Waltz, Shannon X. Wang, Brayden Ware, Travis Weidel, Theodore White, Kristi Wong, Bryan W. K. Woo, Maddy Woodson, Cheng Xing, Z. Jamie Yao, Ping Yeh, Bicheng Ying, Juhwan Yoo, Noureldin Yosri, Grayson Young, Yaxing Zhang, Ningfeng Zhu, Nicholas Zobrist, Hartmut Neven, Pushmeet Kohli, Alex Davies, Sergio Boixo, Julian Kelly, Cody Jones, Craig Gidney, and Kevin J. Satzinger. 2024. Scaling and logic in the color code on a superconducting quantum processor. arXiv:2412.14256 [quant-ph] https://arxiv.org/abs/2412.14256

[84] Tyler Leblond, Ryan S. Bennink, Justin G. Lietz, and Christopher M. Seck. 2023. TISCC: A Surface Code Compiler and Resource Estimator for Trapped-Ion Processors. In *Proceedings of the SC '23 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis (SC-W 2023)*. ACM, 1426–1435. https://doi.org/10.1145/3624062.3624214

[85] Tyler Leblond, Christopher Dean, George Watkins, and Ryan Bennink. 2024. Realistic Cost to Execute Practical Quantum Circuits using Direct Clifford+T Lattice Surgery Compilation. *ACM Transactions on Quantum Computing* 5, 4 (Oct. 2024), 1–28. https://doi.org/10.1145/3689826

[86] Seok-Hyung Lee, Andrew Li, and Stephen D. Bartlett. 2025. Color code decoder with improved scaling for correcting circuit-level noise. *Quantum* 9 (Jan. 2025), 1609. https://doi.org/10.22331/q-2025-01-27-1609

[87] Ang Li, Samuel Stein, Sriram Krishnamoorthy, and James Ang. 2022. QASMBench: A Low-Level Quantum Benchmark Suite for NISQ Evaluation and Simulation. *ACM Transactions on Quantum Computing* 4 (07 2022). https://doi.org/10.1145/3550488

[88] Gushu Li, Yufei Ding, and Yuan Xie. 2019. Tackling the Qubit Mapping Problem for NISQ-Era Quantum Devices. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems* (Providence, RI, USA) *(ASPLOS '19)*. Association for Computing Machinery, New York, NY, USA, 1001–1014. https://doi.org/10.1145/3297858.3304023

[89] Gushu Li, Yufei Ding, and Yuan Xie. 2019. Tackling the Qubit Mapping Problem for NISQ-Era Quantum Devices. arXiv:1809.02573 [cs.ET] https://arxiv.org/abs/1809.02573

[90] Namitha Liyanage, Yue Wu, Alexander Deters, and Lin Zhong. 2023. Scalable Quantum Error Correction for Surface Codes using FPGA. arXiv:2301.08419 [quant-ph] https://arxiv.org/abs/2301.08419

[91] Jeanette Miriam Lorenz et al. 2025. Systematic benchmarking of quantum computers: status and recommendations. (3 2025). arXiv:2503.04905 [quant-ph]

[92] Arshpreet Singh Maan and Alexandru Paler. 2023. Testing the Accuracy of Surface Code Decoders. arXiv:2311.12503 [quant-ph] https://arxiv.org/abs/2311.12503

[93] Marco Maronese, Lorenzo Moro, Lorenzo Rocutto, and Enrico Prati. 2021. Quantum Compiling. arXiv:2112.00187 [quant-ph] https://arxiv.org/abs/2112.00187

[94] Ryutaroh MATSUMOTO and Manabu HAGIWARA. 2021. A Survey of Quantum Error Correction. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* E104.A (06 2021). https://doi.org/10.1587/transfun.2021EAI0001

[95] David C. McKay, Ian Hincks, Emily J. Pritchett, Malcolm Carroll, Luke C. G. Govia, and Seth T. Merkel. 2023. Benchmarking Quantum Processor Performance at Scale. arXiv:2311.05933 [quant-ph] https://arxiv.org/abs/2311.05933

[96] S. A. Moses, C. H. Baldwin, M. S. Allman, R. Ancona, L. Ascarrunz, C. Barnes, J. Bartolotta, B. Bjork, P. Blanchard, M. Bohn, J. G. Bohnet, N. C. Brown, N. Q. Burdick, W. C. Burton, S. L. Campbell, J. P. Campora, C. Carron, J. Chambers, J. W. Chan, Y. H. Chen, A. Chernoguzov, E. Chertkov, J. Colina, J. P. Curtis, R. Daniel, M. DeCross, D. Deen, C. Delaney, J. M. Dreiling, C. T. Ertsgaard, J. Esposito, B. Estey, M. Fabrikant, C. Figgatt, C. Foltz, M. Foss-Feig, D. Francois, J. P. Gaebler, T. M. Gatterman, C. N. Gilbreth, J. Giles, E. Glynn, A. Hall, A. M. Hankin, A. Hansen, D. Hayes, B. Higashi, I. M. Hoffman, B. Horning, J. J. Hout, R. Jacobs, J. Johansen, L. Jones, J. Karcz, T. Klein, P. Lauria, P. Lee, D. Liefer, S. T. Lu, D. Lucchetti, C. Lytle, A. Malm, M. Matheny, B. Mathewson, K. Mayer, D. B. Miller, M. Mills, B. Neyenhuis, L. Nugent, S. Olson, J. Parks, G. N. Price, Z. Price, M. Pugh, A. Ransford, A. P. Reed, C. Roman, M. Rowe, C. Ryan-Anderson, S. Sanders, J.

Sedlacek, P. Shevchuk, P. Siegfried, T. Skripka, B. Spaun, R. T. Sprenkle, R. P. Stutz, M. Swallows, R. I. Tobey, A. Tran, T. Tran, E. Vogt, C. Volin, J. Walker, A. M. Zolot, and J. M. Pino. 2023. A Race-Track Trapped-Ion Quantum Processor. *Physical Review X* 13, 4 (Dec. 2023). https://doi.org/10.1103/physrevx.13.041052

[97] Paul Nation, Abdullah Ash Saki, Sebastian Brandhofer, Luciano Bello, Shelly Garion, Matthew Treinish, and Ali Javadi-Abhari. 2025. Benchmarking the performance of quantum computing software for quantum circuit creation, manipulation and compilation. *Nature Computational Science* (04 2025), 1–9. https://doi.org/10.1038/s43588-025-00792-y

[98] Michael A. Nielsen and Isaac L. Chuang. 2010. *Quantum Computation and Quantum Information: 10th Anniversary Edition.* Cambridge University Press.

[99] Balint Pato, Theerapat Tansuwannont, and Kenneth R. Brown. 2024. Concatenated Steane code with single-flag syndrome checks. *Physical Review A* 110, 3 (Sept. 2024). https://doi.org/10.1103/physreva.110.032411

[100] Tom Peham, Nina Brandl, Richard Kueng, Robert Wille, and Lukas Burgholzer. 2023. Depth-Optimal Synthesis of Clifford Circuits with SAT Solvers. arXiv:2305.01674 [quant-ph] https://arxiv.org/abs/2305.01674

[101] John Preskill. 2018. Quantum Computing in the NISQ era and beyond. *Quantum* 2 (Aug. 2018), 79. https://doi.org/10.22331/q-2018-08-06-79

[102] Timothy Proctor, Kevin Young, Andrew D. Baczewski, and Robin Blume-Kohout. 2024. Benchmarking quantum computers. arXiv:2407.08828 [quant-ph] https://arxiv.org/abs/2407.08828

[103] Quantinuum. [n. d.]. *pytket-quantinuum API documentation.* https://docs.quantinuum.com/tket/extensions/pytket-quantinuum/

[104] Quantinuum. 2024. *Quantinuum System Model H2 Product Data Sheet.* Product Data Sheet. Quantinuum. Version 2.00.

[105] Quantinuum. 2024. *Quantinuum unveils accelerated roadmap to achieve universal, fully fault-tolerant quantum computing by 2030.* https://www.quantinuum.com/press-releases/quantinuum-unveils-accelerated-roadmap-to-achieve-universal-fault-tolerant-quantum-computing-by-2030

[106] Quantinuum. 2025. *Quantinuum Hardware Specifications.* https://github.com/CQCL/quantinuum-hardware-specifications

[107] IBM Quantum. 2024. *Development & Innovation Roadmap.* https://www.ibm.com/quantum/assets/IBM_Quantum_Developmen_&_Innovation_Roadmap_Explainer_2024-Update.pdf

[108] IBM Quantum. 2025. *Development & Innovation Roadmap.* http://www.ibm.com/downloads/documents/us-en/131cf87ab63319bf

[109] IBM Quantum. 2025. *Qiskit Transpiler Documentation.* https://docs.quantum.ibm.com/api/qiskit/transpiler

[110] Nils Quetschlich, Lukas Burgholzer, and Robert Wille. 2023. MQT Bench: Benchmarking Software and Design Automation Tools for Quantum Computing. *Quantum* 7 (July 2023), 1062. https://doi.org/10.22331/q-2023-07-20-1062

[111] A. G. Radnaev, W. C. Chung, D. C. Cole, D. Mason, T. G. Ballance, M. J. Bedalov, D. A. Belknap, M. R. Berman, M. Blakely, I. L. Bloomfield, P. D. Buttler, C. Campbell, A. Chopinaud, E. Copenhaver, M. K. Dawes, S. Y. Eubanks, A. J. Friss, D. M. Garcia, J. Gilbert, M. Gillette, P. Goiporia, P. Gokhale, J. Goldwin, D. Goodwin, T. M. Graham, CJ Guttormsson, G. T. Hickman, L. Hurtley, M. Iliev, E. B. Jones, R. A. Jones, K. W. Kuper, T. B. Lewis, M. T. Lichtman, F. Majdeteimouri, J. J. Mason, J. K. McMaster, J. A. Miles, P. T. Mitchell, J. D. Murphree, N. A. Neff-Mallon, T. Oh, V. Omole, C. Parlo Simon, N. Pederson, M. A. Perlin, A. Reiter, R. Rines, P. Romlow, A. M. Scott, D. Stiefvater, J. R. Tanner, A. K. Tucker, I. V. Vinogradov, M. L. Warter, M. Yeo, M. Saffman, and T. W. Noel. 2025. A universal neutral-atom quantum computer with individual optical addressing and non-destructive readout. arXiv:2408.08288 [quant-ph] https://arxiv.org/abs/2408.08288

[112] Joschka Roffe. 2019. Quantum error correction: an introductory guide. *Contemporary Physics* 60, 3 (July 2019), 226–245. https://doi.org/10.1080/00107514.2019.1667078

[113] Joschka Roffe, David R. White, Simon Burton, and Earl Campbell. 2020. Decoding across the quantum low-density parity-check code landscape. *Physical Review Research* 2, 4 (Dec. 2020). https://doi.org/10.1103/physrevresearch.2.043423

[114] Thomas E. Roth, Ruichao Ma, and Weng C. Chew. 2023. The Transmon Qubit for Electromagnetics Engineers: An introduction. *IEEE Antennas and Propagation Magazine* 65, 2 (April 2023), 8–20. https://doi.org/10.1109/map.2022.3176593

[115] M Saffman. 2016. Quantum computing with atomic qubits and Rydberg interactions: progress and challenges. *Journal of Physics B: Atomic, Molecular and Optical Physics* 49, 20 (Oct. 2016), 202001. https://doi.org/10.1088/0953-4075/49/20/202001

[116] V. M. Schäfer, C. J. Ballance, K. Thirumalai, L. J. Stephenson, T. G. Ballance, A. M. Steane, and D. M. Lucas. 2018. Fast quantum logic gates with trapped-ion qubits. *Nature* 555, 7694 (01 Mar 2018), 75–78. https://doi.org/10.1038/nature25737

[117] Hassan Shapourian, Eneet Kaur, Troy Sewell, Jiapeng Zhao, Michael Kilzer, Ramana Kompella, and Reza Nejabati. 2025. Quantum Data Center Infrastructures: A Scalable Architectural Design Perspective. arXiv:2501.05598 [quant-ph] https://arxiv.org/abs/2501.05598

[118] P.W. Shor. 1994. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science.* 124–134. https://doi.org/10.1109/SFCS.1994.365700

[119] Peter W. Shor. 1995. Scheme for reducing decoherence in quantum computer memory. *Phys. Rev. A* 52 (Oct 1995), R2493–R2496. Issue 4. https://doi.org/10.1103/PhysRevA.52.R2493

[120] Seyon Sivarajah, Silas Dilkes, Alexander Cowtan, Will Simmons, Alec Edgington, and Ross Duncan. 2020. t|ket⟩: a retargetable compiler for NISQ devices. *Quantum Science and Technology* 6, 1 (Nov. 2020), 014003. https://doi.org/10.1088/2058-9565/ab8e92

[121] A. M. Steane. 1996. Error Correcting Codes in Quantum Theory. *Phys. Rev. Lett.* 77 (Jul 1996), 793–797. Issue 5. https://doi.org/10.1103/PhysRevLett.77.793

[122] Andrew M. Steane. 2003. Overhead and noise threshold of fault-tolerant quantum error correction. *Phys. Rev. A* 68 (Oct 2003), 042322. Issue 4. https://doi.org/10.1103/PhysRevA.68.042322

[123] Martin Suchara, John Kubiatowicz, Arvin Faruque, Frederic T. Chong, Ching-Yi Lai, and Gerardo Paz. 2013. QuRE: The Quantum Resource Estimator toolbox. In *2013 IEEE 31st International Conference on Computer Design (ICCD)*. IEEE, 419–426. https://doi.org/10.1109/iccd.2013.6657074

[124] Swamit S. Tannu and Moinuddin K. Qureshi. 2019. Mitigating Measurement Errors in Quantum Computers by Exploiting State-Dependent Bias. In *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture* (Columbus, OH, USA) *(MICRO '52)*. Association for Computing Machinery, New York, NY, USA, 279–290. https://doi.org/10.1145/3352460.3358265

[125] Swamit S. Tannu and Moinuddin K. Qureshi. 2019. Not All Qubits Are Created Equal: A Case for Variability-Aware Policies for NISQ-Era Quantum Computers. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems* (Providence, RI, USA) *(ASPLOS '19)*. Association for Computing Machinery, New York, NY, USA, 987–999. https://doi.org/10.1145/3297858.3304007

[126] Teague Tomesh, Pranav Gokhale, Victory Omole, Gokul Subramanian Ravi, Kaitlin N. Smith, Joshua Viszlai, Xin-Chuan Wu, Nikos Hardavellas, Margaret R. Martonosi, and Frederic T. Chong. 2022. SupermarQ: A Scalable Quantum Benchmark Suite . In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE Computer Society, Los Alamitos, CA, USA, 587–603. https://doi.org/10.1109/HPCA53966.2022.00050

[127] Yu Tomita and Krysta M. Svore. 2014. Low-distance surface codes under realistic quantum noise. *Physical Review A* 90, 6 (Dec. 2014). https://doi.org/10.1103/physreva.90.062320

[128] Lorenzo Valentini, Diego Forlivesi, and Marco Chiani. 2023. Performance Analysis of Quantum Error-Correcting Surface Codes over Asymmetric Channels. arXiv:2302.13015 [quant-ph] https://arxiv.org/abs/2302.13015

[129] Ke Wang, Zhide Lu, Chuanyu Zhang, Gongyu Liu, Jiachen Chen, Yanzhe Wang, Yaozu Wu, Shibo Xu, Xuhao Zhu, Feitong Jin, Yu Gao, Ziqi Tan, Zhengyi Cui, Ning Wang, Yiren Zou, Aosai Zhang, Tingting Li, Fanhao Shen, Jiarun Zhong, Zehang Bao, Zitian Zhu, Yihang Han, Yiyang He, Jiayuan Shen, Han Wang, Jia-Nan Yang, Zixuan Song, Jinfeng Deng, Hang Dong, Zheng-Zhi Sun, Weikang Li, Qi Ye, Si Jiang, Yixuan Ma, Pei-Xin Shen, Pengfei Zhang, Hekang Li, Qiujiang Guo, Zhen Wang, Chao Song, H. Wang, and Dong-Ling Deng. 2025. Demonstration of low-overhead quantum error correction codes. arXiv:2505.09684 [quant-ph] https://arxiv.org/abs/2505.09684

[130] Pengfei Wang, Chun-Yang Luan, Mu Qiao, Mark Um, Junhua Zhang, Ye Wang, Xiao Yuan, Mile Gu, Jingning Zhang, and Kihwan Kim. 2021. Single ion qubit with estimated coherence time exceeding one hour. *Nature Communications* 12, 1 (11 Jan 2021), 233. https://doi.org/10.1038/s41467-020-20330-w

[131] George Watkins, Hoang Minh Nguyen, Keelan Watkins, Steven Pearce, Hoi-Kwan Lau, and Alexandru Paler. 2024. A High Performance Compiler for Very Large Scale Surface Code Computations. *Quantum* 8 (2024), 1354. https://doi.org/10.22331/q-2024-05-22-1354 arXiv:2302.02459 [quant-ph]

[132] Karen Wintersperger, Florian Dommert, Thomas Ehmer, Andrey Hoursanov, Johannes Klepsch, Wolfgang Mauerer, Georg Reuber, Thomas Strohm, Ming Yin, and Sebastian Luber. 2023. Neutral atom quantum computing hardware: performance and end-user perspective. *EPJ Quantum Technology* 10, 1 (28 Aug 2023), 32. https://doi.org/10.1140/epjqt/s40507-023-00190-1

[133] James R Wootton. 2020. Benchmarking near-term devices with quantum error correction. *Quantum Science and Technology* 5, 4 (jul 2020), 044004. https://doi.org/10.1088/2058-9565/aba038

[134] Xiaosi Xu, Simon C. Benjamin, and Xiao Yuan. 2021. Variational Circuit Compiler for Quantum Error Correction. *Phys. Rev. Appl.* 15 (Mar 2021), 034068. Issue 3. https://doi.org/10.1103/PhysRevApplied.15.034068

[135] Min Ye and Nicolas Delfosse. 2025. Quantum error correction for long chains of trapped ions. arXiv:2503.22071 [quant-ph] https://arxiv.org/abs/2503.22071

[136] Keyi Yin, Hezi Zhang, Xiang Fang, Yunong Shi, Travis S. Humble, Ang Li, and Yufei Ding. 2025. QECC-Synth: A Layout Synthesizer for Quantum Error Correction Codes on Sparse Architectures. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1* (Rotterdam, Netherlands) *(ASPLOS '25)*. Association for Computing Machinery, New York, NY, USA, 876–890. https://doi.org/10.1145/3669940.3707236

[137] Ed Younis, Costin C. Iancu, Wim Lavrijsen, Marc Davis, and Ethan Smith. 2021. Berkeley Quantum Synthesis Toolkit (BQSKit) v1. [Computer Software] https://doi.org/10.11578/dc.20210603.2. https://doi.org/10.11578/dc.20210603.2

[138] Zewen Zhang, Pranav Gokhale, and Jeffrey M. Larson. 2025. Efficient frequency allocation for superconducting quantum processors using improved optimization techniques. *Physical Review A* 111, 1 (Jan. 2025). https://doi.org/10.1103/physreva.111.012619

[139] Yue Zhao. 2024. Benchmarking Machine Learning Models for Quantum Error Correction. arXiv:2311.11167 [quant-ph] https://arxiv.org/abs/2311.11167

[140] Youwei Zhao, Yangsen Ye, He-Liang Huang, Yiming Zhang, Dachao Wu, Huijie Guan, Qingling Zhu, Zuolin Wei, Tan He, Sirui Cao, Fusheng Chen, Tung-Hsun Chung, Hui Deng, Daojin Fan, Ming Gong, Cheng Guo, Shaojun Guo, Lianchen Han, Na Li, Shaowei Li, Yuan Li, Futian Liang, Jin Lin, Haoran Qian, Hao Rong, Hong Su, Lihua Sun, Shiyu Wang, Yulin Wu, Yu Xu, Chong Ying, Jiale Yu, Chen Zha, Kaili Zhang, Yong-Heng Huo, Chao-Yang Lu, Cheng-Zhi Peng, Xiaobo Zhu, and Jian-Wei Pan. 2022. Realization of an Error-Correcting Surface Code with Superconducting Qubits. *Phys. Rev. Lett.* 129 (Jul 2022), 030501. Issue 3. https://doi.org/10.1103/PhysRevLett.129.030501

Received January 2026; revised March 2026; accepted April 2026