Accurate Leakage Speculation for Quantum Error Correction

Chaithanya Naik Mude
University of Wisconsin-Madison
Madison, WI, USA
cmude@wisc.edu

Swamit Tannu
University of Wisconsin-Madison
Madison, WI, USA
swamit@cs.wisc.edu

Abstract

Quantum Error Correction (QEC) protects qubits against bit- and phase-flip errors in the $|0\rangle$ /$|1\rangle$ subspace, but physical qubits can also leak into higher energy levels (e.g., $|2\rangle$). Leakage is especially harmful, as it corrupts all subsequent syndrome measurements and can spread to neighboring qubits. Detecting leakage on data qubits is particularly challenging, since they are never measured directly during QEC cycles. Prior work, such as ERASER [43], addresses this by inferring leakage from syndrome patterns using a fixed heuristic. However, this approach often misclassifies benign syndromes, triggering excessive leakage-reduction circuits (LRCs). Because LRCs are themselves noisy and slow, these false triggers lengthen QEC cycles and inflate logical error rates.

We propose GLADIATOR, a general and adaptable leakage speculation framework that works across surface code, color code, and qLDPC codes. Offline, GLADIATOR builds a code-aware error-propagation graph calibrated to device data. Online, it classifies each syndrome in a few nanoseconds and schedules LRC only when the observed pattern is provably leakage-dominated. This precise speculation eliminates up to 3× (and on average 2×) unnecessary LRCs, shortens QEC cycles, and suppresses false positives at their source. Evaluated on standard fault-tolerant benchmarks, GLADIATOR delivers 1.7×–3.9× speedups and 16% reduction in logical error rate, advancing the efficiency of fault-tolerant quantum computing.

CCS Concepts

• **Hardware → Quantum error correction and fault tolerance**.

Keywords

QEC, Leakage Errors, QLDPC, Surface Codes, Color Codes

ACM Reference Format:
Chaithanya Naik Mude and Swamit Tannu. 2025. Accurate Leakage Speculation for Quantum Error Correction. In *58th IEEE/ACM International Symposium on Microarchitecture (MICRO '25), October 18–22, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3725843.3756053

1 Introduction

Quantum computers promise significant speedups over classical computers for quantum simulations, unstructured search, and factorization but are limited by their sensitivity to errors, as quantum information is delicate. The qubits used to store quantum information are highly error-prone, limiting the potential applications of quantum computing. Quantum error correction (QEC) codes can be leveraged to detect and correct the physical errors to enable fault-tolerant memory and computation. In QEC, a large number of noisy physical qubits are used to encode the quantum information into logical qubits, which can detect and correct errors.

Albeit at small scales, many industry and academic labs [2, 6, 7, 45] are actively demonstrating the efficacy of different QEC protocols on a variety of quantum hardware platforms. Some of these experimental demonstrations, especially on superconducting architectures [33, 38], are revealing significant challenges in enabling effective quantum error correction. Leakage errors, for instance, pose a significant challenge to enabling effective QEC protocols.

QEC protocols, while effective for errors within the standard computational basis of qubits ($|0\rangle$, $|1\rangle$), can not detect errors due to leaked states ($|L\rangle$), where qubits occupy higher energy levels ($|2\rangle$, $|3\rangle$, etc.). Such leakage extends beyond intended computational states, posing significant challenges to QEC's effectiveness in error correction as the leaked qubits result in malfunctioning gates corrupting syndrome generation during QEC, leading to faulty detection and correction. Recent experimental studies by IBM [40] and Google QuantumAI [2, 3, 38] highlight this issue, showing leakage errors limiting QEC codes' ability to detect and correct errors.

To improve the effectiveness of QEC at scale, we need techniques to detect and remove the correlated error caused by leakage, especially on data qubits, which are not measured directly and known to accumulate leakage errors [3, 33, 34]. By executing additional operations known as Leakage Reduction Circuit (LRC) gadgets, leaked qubits are forced to return to the computational subspace. Prior works [4, 5, 11, 19, 24, 30, 34] have focused on device-specific mitigation strategies requiring specialized hardware for implementing LRCs, while other methods use more general-purpose gates such as SWAP gates to implement LRCs [10, 32].

Though these LRC operations significantly reduce leakage effects, they increase QEC's cycle latency and resource overhead. For example, a recent work, ERASER [43], highlighted the inefficiencies in unconditional applications of LRCs and developed a speculation-based leakage detection technique using syndrome measurements to insert LRCs judiciously, as adding LRCs constitutes doing additional physical operations, increasing the gate errors and even increasing the probability of leakage itself. ERASER observes that when a data qubit is leaked, it manifests as a syndrome pattern where atleast 50% of the parity qubits are flipped. Whenever such a pattern is encountered, ERASER speculatively applies the LRCs. Furthermore, ERASER uses a *Multi Level Readout (MLR)* on all parity qubits to detect leaked parity qubits and trigger the application of LRCs. However, we observe several limitations of ERASER.

MICRO '25, Seoul, Republic of Korea

ACM ISBN 979-8-4007-1573-0/2025/10
https://doi.org/10.1145/3725843.3756053

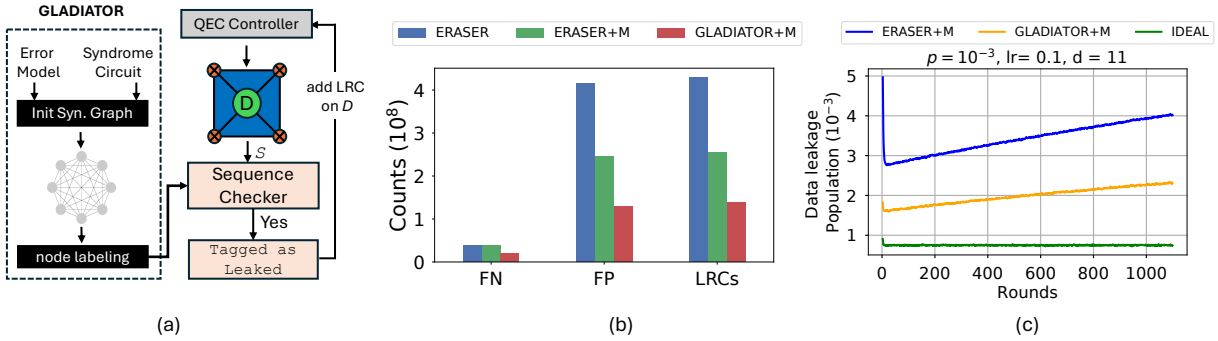

Figure 1: (a) Proposed GLADIATOR Design. (b) Comparing the effectiveness of GLADIATOR and ERASER using False Negative (FN), False Positive (FP), and LRC utilization. (c) The leakage population for surface code with code distance d =11, when executed for $100d$ QEC rounds, assuming the probability of non-leakage error $p_e = 10^{-3}$ and probability of leakage is $p_{leak} = lr \times p_e = 10^{-4}$.

High False Positive Rate. ERASER's strategy leads to a significant rate of false positives due to its assumption that 50% of bit flips would mainly arise from leakage. In reality, many of these bit flips result from other types of errors occurring in computational basis. This high false positive rate degrades reliability, adding unnecessary LRCs introducing additional gate and leakage errors.

Growing Leakage Population. Accumulated leakage can severely impair QEC, potentially making it ineffective because leakage paralyzes syndrome measurement [15]. Ideally, a leakage speculation policy should stabilize or reduce leakage levels over time. Due to imperfection in qubit devices, the insertion of unnecessary LRCs can increase leakage. Although ERASER effectively reduces leakage by strategically applying LRCs, our tests over 100 QEC cycles reveal a continuous rise in leakage population when ERASER is used. This increase is primarily due to ERASER's high rate of false positives, which mistakenly flag qubits as leaky, leading to unnecessary LRC insertions. The speculation strategy ERASER uses is rigid and ineffective for long-running QEC cycles, which need to withstand variations in qubit characteristics, and leakage accumulation.

Generalizability is essential for scalable leakage mitigation. ERASER's heuristic-based approach exploits the symmetry and regularity of the surface code, where each data qubit interacts with four ancilla qubits, yielding rich syndrome information. However, ERASER's strategy does not generalize to other QEC codes, such as color codes or quantum LDPC codes, where syndrome information per data qubit is sparse, and the stabilizer connectivity is irregular and often asymmetric. In these settings, accurately inferring leakage becomes significantly harder due to the limited and noisy observables. The challenge is exacerbated in high-rate codes, where each syndrome bit is influenced by fewer qubits. Furthermore, alternative approaches like walking surface code [14] rely on planar, cyclic role exchanges that are incompatible with non-planar topologies and irregular interaction graphs required by codes like qLDPC or hypergraph product codes (HGP) [22].

GLADIATOR. We propose a general design framework, GLADIATOR, that can guide the leakage speculation strategy by using the error model and error correcting code to estimate syndromes that are most likely caused by the data qubit leakage. We observe that the patterns ERASER flags as leakage may also result from non-leakage errors caused by imperfect gates and measurements, which can approximately be ten times more likely than leakage. Our characterization of IBM hardware shows that with leakage on the data qubit, the CNOT gates malfunction, producing independent random bit flips; therefore, certain syndrome patterns are more likely to be caused by leakage-driven random bit flips than consistent bit flips caused by non-leakage errors. For example, pattern "0011" is more likely to be caused by non-leakage (i.e, data qubit excitation), while the pattern "1001" most likely indicates a leakage.

As shown in Figure 1(a), we formulate the leakage diagnosis using syndrome patterns as a graph labeling problem. First, we construct a leakage graph in which a node represents a 4-bit syndrome pattern associated with a data qubit measured during the surface code syndrome generation cycle. The nodes are connected via directed and weighted edges. The directionality of edges indicates the possibility of a syndrome pattern in the source node transforming into a target pattern due to leakage, with the transformation likelihood determining the weight of these edges. Similarly, we build the syndrome transformation graph due to non-leakage errors, where we only account for transformation results from non-leakage errors. We merge both the non-leakage and leakage graphs so that the edges are adjusted to reflect the combined probabilities of these transitions. Using relative transition rates, we label syndrome nodes as leakage or non-leakage nodes. Our graph-based approach allows for the systematic construction of a diagnosis framework, where nodes are labeled based on their probability of representing leakage or non-leakage scenarios, using experimental data to guide the decision-making process. Under identical error configurations, ERASER flags *11/16* syndrome patterns as leakage-causing patterns, whereas GLADIATOR flags only *8/16* as the most likely leakage patterns. This reduces the false positive rate by 1.91× as shown in Figure 1(b), thereby reducing the data leakage population by 1.73× as shown in Figure 1(c), for code distance $d = 11$ in $100d$ rounds.

Speculation Inaccuracy. We decrease the total counts of false positives and false negatives combined by around 3.11× for GLADIATOR over ERASER. For the MLR-enhanced version of ERASER, i.e, ERASER+M around 1.92× with GLADIATOR+M for d=11.

Leakage Population. GLADIATOR+M produces 1.73× less data leakages compared to ERASER+M, after 100 QEC cycles for $d = 11$.

Resource Overheads Our method uses less than 0.1% LUTs at even code distance 25. Compared to ERASER, it achieves at least

17× reduction in FPGA resource usage across code distances 5–25. Our generalizable approach ensures principled and robust leakage mitigation strategies. The key differences between the prior work ERASER+M and our strategies, STAGGERING and GLADIATOR+M, primarily lie in their approach to leverage syndrome-based speculation, and Multi-Level Readout (MLR) as summarized in Table 1.

Table 1: Leakage Mitigation: GLADIATOR vs. Prior Work

Method	Speculation	Rounds	MLR	Adaptability
Always-LRC	No	0	No	No
STAGGERING (OURS)	No	0	No	No
DQLR[33]	No	0	Yes	No
ERASER+M [43]	Yes	1	Yes	No
GLADIATOR+M (OURS)	Yes	1	Yes	Yes
GLADIATOR-D+M (OURS)	Yes	2	Yes	Yes

2 Background and Motivation

2.1 Noisy Qubits and Leakage Errors

Qubits, by design, are anharmonic oscillators intended to function as two-level systems suitable for quantum computing. Unfortunately, they inherently possess multiple energy levels, complicating their operation within the desired computational subspace. As shown in Figure 2(a), this subspace typically comprises the two lowest energy states, labeled as $|0\rangle$ and $|1\rangle$. Qubits are highly sensitive to errors and can be broadly categorized into two types:

Intra-subspace errors. These occur within the computational subspace itself. Errors of this type involve transitions between the $|0\rangle$ and $|1\rangle$ states or create superpositions of these states and can be corrected by standard quantum error correction (QEC) techniques.

Leakage errors. Unlike intra-subspace errors, leakage errors involve transitions of a qubit's state to energy levels outside of the designated computational subspace, such as $|2\rangle$, $|3\rangle$, or higher. These errors effectively "hide" the qubit state from the standard error correction mechanisms. Addressing leakage requires specialized techniques to detect when a qubit exits the computational subspace and implement strategies to return it to the correct operational state.

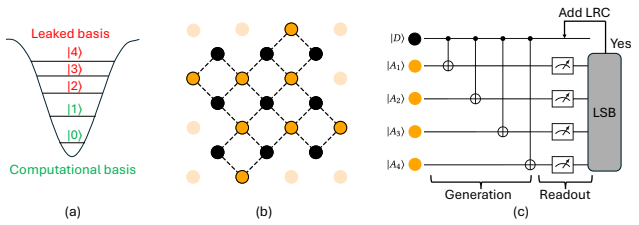

Figure 2: (a) Qubit device with computational basis corresponding to $|0\rangle$,$|1\rangle$ and leaked basis corresponds to higher energy quantum levels. (b) Surface code of distance 3, with black qubits corresponds to data qubits and others corresponding to the parity qubits. (c) Syndrome Generation for LRC insertion using Leakage Speculation Block (LSB).

This might involve additional leakage reduction circuits, specifically designed to monitor and correct leakage. Other strategies include the integration of qudit-error correcting codes [27, 29, 36].

2.2 Impact of Leakage on QEC

Quantum error-correcting codes reduce error rates by encoding a logical qubit across many physical qubits and continuously measuring parity checks. Surface codes is a leading approach, using $2d^2-1$ qubits for a distance-d code with d^2 data qubits and d^2-1 parity qubits. Data qubits are entangled with parity qubits via stabilizer circuits to detect and correct bit-flip (X), phase-flip (Z), or combined errors. A distance-d code can correct up to $\frac{d-1}{2}$ errors.

Leakage errors excite qubits to higher energy states. In this scenario, the traditional syndrome extraction circuits used in QEC fail to detect the leakage, as their design does not account for these leaked states. Therefore, even though the syndrome extraction circuits are run multiple times, the iterative process cannot identify or correct leakage errors, resulting in the accumulation of leakage errors and catastrophic failures [11, 15, 41].

Furthermore, the two-qubit gates such as CZ and CX, used in the syndrome generator circuits malfunction in the presence of leakage [16], further complicating the error correction process. This malfunctioning alters the expected behavior of the gates, leading to incorrect or incomplete syndrome generation. Thus, without modifications or additional mechanisms specifically targeting leakage, standard QEC methods remain ineffective against leakage errors, as detailed in Section 2.3, where we will discuss leakage injection experiments conducted on IBM systems. To mitigate the impact of leakage on QEC, the ERASER+M framework employs a leakage speculation block (LSB) to analyze syndrome measurements, detect leakage, and apply leakage reduction circuits to restore the qubit to one of its computational states, as illustrated in Figure 2(c).

2.3 Leakage Characterization on IBM Systems

QEC depends on reliable gate operations. However, leaked qubits disrupt gate behavior, undermining QEC efficacy. Prior work [3, 12] has shown that QEC performance is tightly coupled to the hardware's error profile. In superconducting qubits, leakage is rare ($\sim 10^{-3}$) and stochastic [11], making systematic analysis difficult.

To study leakage-induced faults in syndrome generation, we conduct experiments on IBM quantum systems using a targeted leakage injection method. Specifically, we initialize qubits in the $|2\rangle$ (leaked) state and repeatedly run syndrome generation circuits. This setup lets us observe how leakage propagates between data and parity qubits and track leakage dynamics over multiple QEC cycles. Due to hardware constraints, we focus on the core component of syndrome circuits—the CNOT gate. Figure 3(a) shows when the control qubit is leaked, the target qubit toggles between $|0\rangle$ and $|1\rangle$, effectively producing a mixed state, inducing a 50% bit-flip error.

Leakage disrupts CNOT operations and must be mitigated for QEC to remain effective.

To examine how leakage accumulates, we repeat CNOT operations with and without injecting leakage. This emulates multiple QEC rounds. As shown in Fig. 3(c), leakage population grows over time when initialized, and remains low without injection, highlighting how a single leaked qubit can spread leakage.

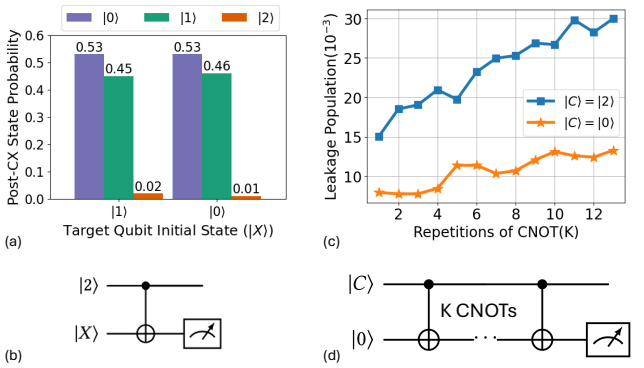

Figure 3: Leakage Injection Experiment on IBM hardware- (a) The probability of the measured state when circuit (b), a single physical CNOT with one of the qubits leaked, is executed. (c) The leakage population of measured qubit when circuit. (d) with one physical CNOT is repeatedly executed. We ran 10,000 shots using Qiskit Pulse [20, 37].

2.4 Taxonomy of Leakage Reduction Circuits

Leakage Reduction Circuits (LRCs) are physical operations that convert leakage errors into Pauli errors that QEC can detect. LRCs fall into four categories: *reset-based*, *specialized hardware*, *qubit role exchange*, and *leakage-to-erasure conversion*.

Reset-Based LRCs. These approaches use SWAP gates or Reset protocols [32] to offload leakage to ancilla qubits for conditional reset. While hardware-friendly, they introduce circuit depth overhead and depend on high-fidelity leakage detection; misidentification can exacerbate logical error rates (LER).

Specialized Hardware LRCs. Techniques like Data Qubit Leakage Removal (DQLR) [33] coherently transfer leakage population via a Leakage-iSWAP gate to a fast-reset qubit. Although effective, DQLR requires precise control, custom pulses, and complex calibration, and remains non-fault-tolerant.

Qubit Role Exchange. The walking surface code [14] alternates data and ancilla roles across QEC rounds, enabling staggered resets. While this avoids specialized hardware, it incurs time overhead and *does not generalize well beyond surface codes.*

Leakage-to-Erasure Conversion. Some platforms convert leakage into detectable erasures by encoding in metastable subspaces or using fluorescence-based detection [25, 44]. Despite high conversion rates, residual leakage and imperfect resets persist.

While LRC techniques vary in mechanism and overhead, they are often applied naively in every QEC round. However, LRCs are *not fault-tolerant*, as excessive application introduces gate and leakage errors, while infrequent application allows leakage to accumulate. To be effective, LRC scheduling must satisfy three key criteria:

- **Accuracy:** Minimize false positives (unnecessary resets) and false negatives (missed leakage).
- **Generalizability:** Maintain efficacy across QEC codes (e.g., surface codes, color codes, HGP, BPC codes).
- **Adaptability:** Remain effective under variable noise models and hardware yield variations [26, 39].

3 Limitations of Prior Work

3.1 Open-Loop vs. Closed-Loop LRC Scheduling

Leakage mitigation strategies fundamentally differ in how they decide when to apply Leakage Reduction Circuits (LRCs). We categorize these approaches into two paradigms:

Open-Loop Scheduling applies LRCs at predetermined intervals without detecting actual leakage events. These methods are hardware-friendly and require minimal real-time decision-making, but risk applying unnecessary LRCs when no leakage is present.

Closed-Loop Scheduling applies LRCs based on inferred leakage from syndrome measurements. While offering higher precision by targeting actual leakage events, these methods require reliable leakage detection and low-latency control decisions.

The key trade-off lies between simplicity and precision, as open-loop methods execute a large number of unnecessary LRCs, while closed-loop methods risk missing leakage events or making incorrect inferences from noisy syndrome data.

3.2 ERASER: Motivating Closed-Loop Control

ERASER [43] introduced the first closed-loop leakage mitigation framework, demonstrating significant improvements over naive open-loop baselines. The core insight was that leaked qubits corrupt syndrome measurements in predictable patterns, enabling leakage inference from stabilizer measurement anomalies.

ERASER's Detection Strategy: ERASER infers data qubit leakage by monitoring downstream effects on syndrome measurements. A leaked data qubit corrupts CNOT gate behavior, producing random measurement outcomes that typically cause approximately 50% of connected syndrome bits to flip. ERASER triggers LRCs when either: (1) 50% or more of connected syndrome bits flip, or (2) Multi-Level Readout (MLR) on parity qubits directly detects leakage.

Baseline Comparison: ERASER demonstrated its effectiveness against *Always-LRC*, an open-loop policy that applies LRCs to all qubits simultaneously after every QEC round. This comparison showed substantial improvements in logical error rates, establishing the value of selective LRC application.

3.3 ERASER's Design Limitations

Our analysis reveals two critical limitations in ERASER's approach that motivate the need for more sophisticated closed-loop methods:

Limitation 1: High False Positive Rate: ERASER's rigid 50% threshold assumption leads to frequent misclassification of standard gate error patterns as leakage events. Table 2 shows that many syndrome patterns attributed to leakage are more likely caused by non-leakage errors, resulting in unnecessary LRC applications that can introduce additional operational and leakage errors.

Limitation 2: Inadequate Leakage Population Control: The high false positive rate prevents effective leakage population management. Excessive LRC applications not only waste resources but can paradoxically increase leakage population over time due to LRC-induced errors, as evidenced by the gradual increase in leakage levels shown in Table 2.

Limitation 3: Code-Specific Heuristics and Poor Generalization: ERASER's detection strategy relies on code-specific assumptions that limit its applicability beyond surface codes. The

Table 2: Leakage Detection Efficacy of ERASER(ER)

Metrics	Always LRC	ER	ER+M	M	Staggered	Ours
FN	1	3.9	3.8	6.3	1.6	3.2
FP	1	0.06	0.04	0.05	0.5	0.021
LRCs	1	0.06	0.04	0.05	0.5	0.022
Leak-70(10^{-3})	4.36	4.19	2.97	6.01	4.87	3.24
Leak-700(10^{-3})	6.73	4.94	3.78	6.87	4.54	3.35

50% threshold heuristic exploits the symmetric structure of surface codes, where data qubits are typically connected to 3-4 ancilla qubits, making the "half-flip" pattern a reasonable indicator of leakage-induced CNOT corruption.

However, this heuristic fails to generalize to other quantum error correction codes with different connectivity patterns and structural properties. In color codes, for example, data qubits are connected to only 2-3 ancilla qubits, significantly altering the expected syndrome patterns from leaked qubits. Similarly, qLDPC codes exhibit irregular connectivity graphs that break the symmetry assumptions underlying ERASER's detection logic.

Our experimental analysis reveals that when applied to color codes, ERASER's heuristic becomes overly sensitive, triggering LRCs for almost all error patterns, therefore, effectively reducing to the *Always-LRC* baseline that ERASER originally demonstrated to be suboptimal. This poor generalization severely limits ERASER's practical applicability as the quantum computing field explores diverse QEC code families beyond surface codes.

These three fundamental limitations demonstrate the need for more sophisticated approaches that can achieve ERASER's selective application benefits while dramatically reducing false positive rates and generalizing across diverse QEC code structures.

3.4 Gaps in ERASER's Analysis

While ERASER's closed-loop approach represents a significant advance, its experimental evaluation has notable gaps that limit our understanding of the open-loop vs. closed-loop trade-offs:

Weak Open-Loop Baseline: ERASER only compared against *Always-LRC*, which simultaneously resets all qubits—a policy known to increase correlated errors and gate overhead. This naive baseline does not represent the best possible open-loop strategy.

Missing Structured Open-Loop Analysis: ERASER did not explore whether more intelligent open-loop scheduling could narrow the performance gap. Specifically, a spatially staggered LRC application could reduce correlated errors while maintaining simplicity.

Incomplete MLR-only Evaluation: The original ERASER work did not evaluate against MLR-only detection (using only parity qubit measurements), making it difficult to assess the value of syndrome-based inference. To address these analytical gaps, we study the Staggered application of LRCs with MLR.

3.5 Reducing Leakage via Staggered LRCs

We propose *Staggered Always-LRC*, a structured open-loop policy that schedules LRCs as an n-coloring problem on the qubit interaction graph. No adjacent or diagonally neighboring qubits share the same color, and each color group is reset in a round-robin fashion across QEC rounds as shown in Figure 4(a) with color groups represented by dotted lines. This spatial staggering minimizes correlated faults while maintaining the simplicity of open-loop control. While Staggered Always-LRC does not universally outperform ERASER, it significantly narrows the gap in low leakage mobility scenarios. At low mobility ($\approx 1\%$), the LER gap between Staggered Always-LRC and ERASER shrinks by 2× compared to high mobility ($\approx 10\%$).

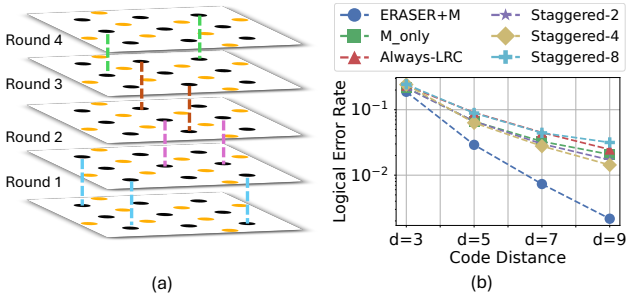

Figure 4: (a) Qubit coloring (Dotted lines as color groups) for round-robin LRC scheduling in Staggered-LRC. (b) Logical Error Rate (LER) across open-loop policies and ERASER+M.

Key Insight: When leakage mobility, which is the probability of leakage transport between qubits is low, and LRC gate errors are modest, structured open-loop strategies can be competitive with closed-loop methods, especially at smaller code distances.

While improved open-loop scheduling narrows the gap, closed-loop methods still offer advantages in scenarios with higher leakage rates or mobility. However, ERASER's heuristic-based approach suffers from high false positive rates.

We introduce GLADIATOR, a principled closed-loop framework that models leakage speculation as a graph labeling problem. Rather than relying on simple bit-flip-count-based heuristics, GLADIATOR uses probabilistic inference to distinguish leakage-induced syndrome patterns from those caused by standard gate errors.

4 GLADIATOR

To address the limitations of ERASER, we propose a more accurate, adaptable, and generalizable approach to leakage speculation: *Graphical Model for Leakage Detection in Syndrome Generator (*GLADIATOR*)*. This section outlines GLADIATOR's key insights and design.

4.1 Not All Bit-flips are caused by Leakage

A leakage error in a data qubit can induce specific bit-flip patterns in the measured syndrome. ERASER uses 50% or more bit flips as the leakage signature for speculation. However, these bit flips can also result from non-leakage errors due to imperfect gates and measurements, especially when non-leakage errors are 10× more likely than leakage errors. As illustrated in Figure 5(a), most patterns flagged by ERASER are probably due to non-leakage errors.

We evaluate ERASER's performance across eleven syndrome patterns, as depicted in Figure 5(a). We observe that ERASER's straightforward design fails to distinguish between different patterns effectively. Some patterns are much more likely to be caused by leakage

than non-leakage errors. Therefore, we can classify these patterns into two groups, as, those requiring immediate mitigation and those that can be postponed without immediate intervention, allowing them to be addressed in the later QEC rounds.

Leakage and non-leakage errors can cause similar flips in the syndrome, complicating the leakage diagnosis. However, we can leverage the relative difference in the leakage and non-leakage event probabilities to help estimate the likelihood of error syndrome caused by leakage or non-leakage errors. Consider a syndrome with bit-flip pattern 0011 in the surface code, which could arise from multiple scenarios: *(1)* Leakage Error on data qubit, causing 50% bitflips due to malfunctioning CNOT gates, *(2)* "X" Error on data qubit, introduced after the second CNOT of the stabilizer circuit, *(3)* Two independent bit flips during measurement, or *(4)* Two independent errors on the last two qubits during the reset operation. Among these, leakage and "X" errors on data qubit are first-order effects because they involve a single error event; the probability of these events is $c.P_e$, where c is a constant and P_e is the error probability. In contrast, scenarios involving two independent errors due to measurement or reset are second-order effects, requiring two concurrent error events with $c.P_e^2$ probability, making them substantially less likely, where leakage errors occur with probability P_{leak} and non-leakage errors with P_e. We assume the leakage occurs with probability $\mathbf{P_{leak}} = \mathbf{lr} \times \mathbf{P_e}$, where $\mathbf{lr}$, leakage ratio, is the scaling factor to represent intensity of leakage over other errors.

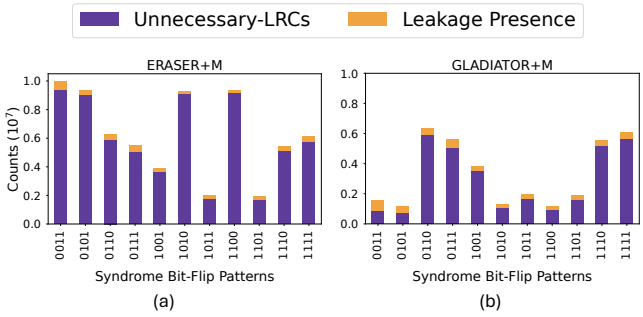

Figure 5: Unnecessary LRCs and leakage instances of (a) ERASER+M (b) GLADIATOR+M for 4-bit ERASER patterns

In comparison, consider syndrome 0110. For the 0110 pattern, all non-leakage errors causes are second-order events, except where the "X" Error on the data qubit generates the 1111 syndrome pattern and then followed by two measurement errors resulting in the 0110 pattern. On the other hand, if the data qubit leaks before the first CNOT or leaks before the second CNOT, we will observe the 0110 patterns with $\frac{1}{16} \times P_{leak}$ and $\frac{1}{8} \times P_{leak}$, respectively, which gives the combined probability of 0.02% for $lr = 0.1$ and $P_e = 10^{-3}$ as shown in Figure 6(a), similarly, the probability that this syndrome is caused by non-leakage event is about $3.P_e^2 + P_e^3 = 10^{-6}$, Making leakage the primary cause to observe this pattern. We leverage this insight to determine the set of bit-flip patterns that most likely to indicate a leakage presence in the data qubit.

4.2 GLADIATOR Design

We formulate leakage diagnosis as a graph labeling problem. In this model, each node represents a syndrome pattern. The edges between these nodes are weighted and directed, indicating how likely it is for one syndrome pattern to transform into another, a transformation driven by leakage or non-leakage errors. Our goal is to label the nodes of this graph as likely leakage and non-leakage patterns at post-qubit calibration and use the critical leakage patterns in the QEC controller to trigger LRCs judiciously at runtime.[1]

Building Leakage and Non-Leakage Graphs. The leakage graph shows the likelihood of leakage transforming one syndrome pattern into another. We build this leakage graph by starting with the base node, which is a syndrome pattern without any leakage, and we grow the graph by adding syndrome nodes that are transformed due to leakage error. This process is illustrated in Figure 6(b). We will use the base node 0000, a state without any error for simplicity. During the generation and measurement of syndromes, any operation can introduce leakage. Our experiments on IBM hardware show that leakage on control during CNOT can result in random bit-flips on target, which is a primary means of detecting data leakage using parity qubit measurement. Figure 6(a), shows the two qubit gates involved in surface code circuit with data qubit as D and the parity qubits in order as A1, A2, A3, and A4. As shown in Figure 6(a), leakage on data qubits just before the measurement cannot be detected until next round. However, if leakage occurs before `CNOT (D, A4)` (light-pink), we observe 50% bitflips in A4 due to malfunctioning CNOT, resulting in 0001 and 0000 syndromes. In comparison, if leakage is injected before `CNOT (D, A3)` (light-orange), we will observe 50% bit flips in both A3 and A4, resulting in 0000, 0001, 0010, 0011 syndrome patterns. This process can be represented as a directed and weighted graph shown in Figure 6(b), where edge weights are a product of the base state probability (prior) and probability of leakage-driven transformation. The leakage events annotated in Figure 6(a) are color-coded to match the syndrome patterns discussed above.

In Figure 6(b), repeated nodes are merged by adding the edge weights corresponding to the pair of nodes. Figure 6(b) presents only a small number of edges for clarity. Next, we build leakage graphs starting with other base states that may have a non-leakage error, as shown in Figure 6(b). We merge all the individual graphs by summing edge weights for repeating source and target nodes.

We use a similar methodology to build a non-leakage graph as the leakage graph. In the figure 6(b), we show how the base node "0000" can possibly be transformed into syndrome patterns due to first-order and second-order non-leakage errors. The non-leakage graph is reduced by merging repeating source and target node pairs and summing the edge weights, similar to the leakage graph's reduction. Using experimentally verified leakage (section 2.3) and non-leakage error models, we can estimate how leakage errors are expressed in the syndrome measurements.

[1]Our approach parallels the standard decoders in inferring the most likely error by analyzing graph weights. However, most conventional decoders either lack awareness of leakage errors altogether or require multiple rounds of syndrome information before inferring the faults due to leakage. Our experiments highlights the cost of delaying detection of leakage faults, as leaked states persist across cycles, accumulate, and can propagate to neighboring qubits, ultimately causing logical errors. To prevent leakage accumulation, it is essential to quickly and accurately classify and detect leakage faults within two rounds from the occurrence of leakage.

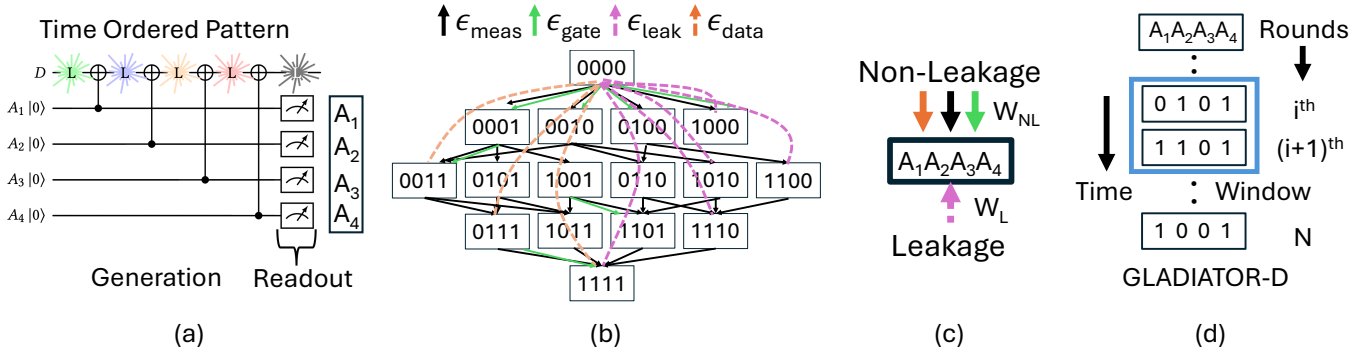

Figure 6: Overview of GLADIATOR. (a) Stabilizer circuit produces time-ordered syndrome patterns (b) A graphical model captures transitions between patterns based on error probabilities (c) Each node's incoming and outgoing edge weights reflect likelihood of leakage over non-leakage for the pattern (d) GLADIATOR-D uses a two-round windowed history for robust leakage speculation.

Graph Merging and Node Labeling. We merge the reduced leakage and non-leakage graphs; while merging, we merge the nodes of two graphs and keep all the non-leakage edges (colored ϵ_{gate} (green), $\epsilon_{measure}$ (black) and ϵ_{data}(orange)) and leakage edges (colored $\epsilon_{leakage}$(pink)). The final graph would be an all-to-all connected graph, with each node having both incoming and outgoing leakage and non-leakage edges, as shown in Figure 6(c). To label the node, we iterate over all nodes and sum the weights of all incoming non-leakage edges to create a super edge W_L such that $W_L = \sum w_l^i$as shown in Figure 6(c). On any node, if the leakage (pink) super-node weight (W_L), which represents the cumulative likelihood that the syndrome is produced by leakage error, is greater than the weight of the non-leakage super-node (W_{NL}), i.e, cumulative likelihood that the syndrome is result of non-leakage error, by a threshold factor, we mark that node thereby the syndrome as leakage pattern for immediate mitigation in the next subsequent round.

GLADIATOR operates in two stages: (1) Offline Stage: we analyze the target QEC circuit once, build an error-propagation graph, and weight its edges with calibration data (leakage rate, non-leakage noise, readout error). The result is a lookup table of syndrome patterns that strongly indicate leakage. (2) Online Stage, i.e, during QEC cycle, the syndrome bit pattern is used to query the table, and schedules an LRC only when the pattern is flagged as "leakage".

4.3 GLADIATOR - Speculation Efficacy

Efficacy. Figure 5(a-b) compares the LRCs inserted for observed syndrome patterns with leakage (golden bar) and without the leakage (purple bar). Compared to ERASER, both the GLADIATOR designs significantly improve the accuracy and thereby dramatically lower the number of LRCs used. *Efficacy of* GLADIATOR *is rooted in sophisticated leakage classification strategy. For instance,* ERASER *flags 11/16 4-bit syndrome patterns as leaked, including frequently occurring patterns such as "0011", most likely caused by non-leakage errors. In comparison,* GLADIATOR, *flags 7/16 syndrome patterns excluding such frequently occurring patterns due to non-leakage errors.*

Tunable Features for New QEC Codes and Dynamic Errors. GLADIATOR incorporate leakage propagation, error dynamics and stabilizer execution to build the graphical model as shown in Figure 6(b) to map the syndrome pattern transitions under both Pauli and leakage errors during the QEC code execution. Nodes represent syndrome patterns, while edges represent error combinations that transition one syndrome pattern to another. Non-leakage faults yield deterministic transitions, while leakage faults produce multiple possible outcomes incorporating leakage propagation dynamics (Section 2.3). Each incoming edge encodes a distinct error path, with tunable weights, updated from calibrated error rates.

For newer codes with n-bit detector patterns, the transition graph is constructed by enumerating all possible errors before each gate in the stabilizer circuit, connecting base nodes to reachable detector flip patterns through annotated error edges similar to Figure 6 (b). Recalibration updates only the edge weights, while preserving the underlying error paths in the graph structure, and allows GLADIATOR to adapt seamlessly to time-varying noise characteristics and hardware-specific error profiles. As the system evolves, the framework allows for incorporating additional noise sources as error transitions, added as edges to the graphical model during recalibration. GLADIATOR-D+M extends this framework using a two-round syndrome history as in Figure 6(d), to infer leakage and apply LRCs every round (except the first) in a sliding window fashion.

Large-scale quantum systems [1–3] exhibit considerable device variability, system drift and correlated noise, which can obscure the distinction between leakage-induced and standard Pauli errors. Unlike ERASER with fixed policy, GLADIATOR design with tunable graphical model delivers robust performance across a wide range of error profiles as discussed in section 7.5, positioning GLADIATOR as a scalable and adaptable solution for leakage mitigation in diverse and dynamic QEC scenarios on quantum systems.

4.4 Overheads of Deploying GLADIATOR

Figure 7 illustrates the GLADIATOR microarchitecture. During each QEC cycle, syndrome data is processed by a parity adjacency generator to produce a syndrome pattern per data qubit. This pattern is evaluated by a sequence checker, which matches it against high-probability leakage signatures from GLADIATOR. If either a match is found or if the associated parity qubit is leaked, then the LRC

scheduler triggers a leakage reduction circuit in the next round and notifies the QEC controller. Since common leakage patterns are stable across error rates, a single sequence checker can be efficiently shared across multiple data qubits.

Data-Parity Adjacency Generator. The design and functionality is illustrated in Figure 7. Syndrome Data corresponds to measurement data from all the parity qubits. But the patterns deciphered by GLADIATOR pertains to individual data qubit. The leakage speculation aims to make predictions for each of the nine data qubits marked as D_i in Figure 7, by using the parity qubit A_i measurement flips which interact with the data qubit. For example, the data qubit D_5 is connected to parity qubits (A_3, A_4, A_6, A_5). Upon measuring these parity qubits, we will get a four-bit string, say "1000", which is sent for sequence checking, which outputs a one-bit label, "0" as leakage not present, and "1" leakage present.

Note that not all data qubits have four adjacent parity qubits; corner qubits such as D_3 have just two parity qubits (A_1, A_4). To accommodate cases with two and three-parity qubits, we expand the sequence checker to match patterns for 2-bit, 3-bit, and 4-bit data-parity input to accommodate all possible syndrome patterns. For the patterns with 4-bit we prepend a index tag with "0", 3-bit with a "10" and 2-bit with "110" to make them all 5-bit patterns $(x_4x_3x_2x_1x_0)$. This uniform representation simplifies processing while supporting all possible syndrome patterns. A network of multiplexers forms the core functionality of the Data-Parity Adjacency Generator.

Sequence Checker. GLADIATOR detects leakage by matching a 5-bit data-parity syndrome pattern against pre-defined Boolean templates using combinational logic. This enables fast, parallel evaluation with minimal hardware overhead. In contrast, ERASER relies on a hand-crafted finite state machine (FSM) to track syndrome history and trigger LRCs, resulting in high LUT usage that scales poorly with code distance.

GLADIATOR encodes all the common leakage patterns observed in surface codes as compact Boolean expressions. These patterns represent stable syndrome-flip signatures caused by persistent leakage. After minimization, the pattern set reduces to:

$$(x_0 \wedge x_1 \wedge x_4) \vee (x_0 \wedge x_2 \wedge x_3) \vee (x_2 \wedge x_3 \wedge x_4)$$
$$\vee (x_2 \wedge x_3 \wedge \neg x_1) \vee (x_2 \wedge x_4 \wedge \neg x_0 \wedge \neg x_1)$$

This logic requires at most 10 LUTs per data qubit and can be evaluated within 1 ns. To support code distance d, leakage detection must complete for all d^2 data qubits within 100 ns—the approximate latency of four CNOTs on superconducting platforms like Google's [3]. To meet this deadline, GLADIATOR replicates the sequence checker, therefore total LUTs per logical qubit:

$$\text{LUTs}_{\text{total}} = 10 \times \left\lceil \frac{d^2}{100} \right\rceil.$$

For example, at $d = 25$, GLADIATOR requires only 70 LUTs to detect leakage across all 625 data qubits in real time, demonstrating both scalability and low hardware cost. As shown in Table 3, GLADIATOR achieves a 17× **to** 80× **reduction** in resource usage across **code distances 5 to 25**. We use the same Kintex UltraScale+ FPGA (xcku3p-ffvd900-3-e) as ERASER [43], and re-synthesize ERASER's design to support larger distances for a fair comparison.

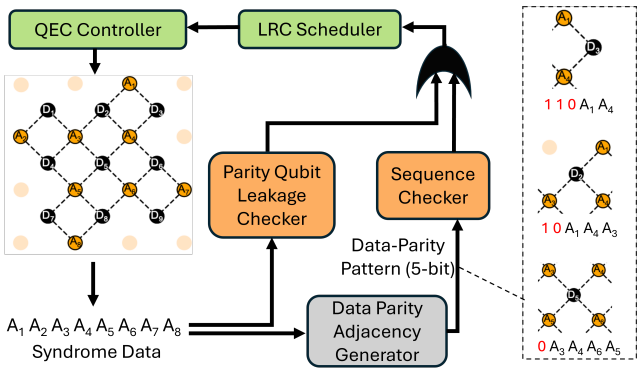

Figure 7: Overview of GLADIATOR's microarchitecture

Table 3: LUTs Per Logical Qubit on Kintex UltraScale+ FPGA

Method	d = 5	d = 9	d = 13	d = 17	d = 21	d = 25
GLADIATOR	10	10	20	30	50	70
ERASER	177	633	1382	2434	3786	5393
Relative Reduction	17.7x	63.3x	69.1x	81.1x	75.7x	77.0x

LRC scheduling decisions must be made before the next round of error correction completes[2]. Like ERASER, GLADIATOR meets this timing constraint, but does so using 17× to 80× fewer LUTs. This efficiency comes from using combinational logic to match 5-bit syndrome patterns against minimized Boolean templates, enabling fast, parallel detection with minimal overhead.

5 Generalizability of GLADIATOR

5.1 Leakage Detection Beyond Surface Codes

Color codes [9, 42, 46], QLDPC codes [7, 8, 22] are emerging as a compelling candidate for quantum error correction, offering resource-efficient encoding of quantum information with significantly fewer qubits. The triangular 6.6.6 color code, as shown in Figure 8(b), constructed on a hexagonal lattice with a three-colorable structure, enables efficient fault-tolerant implementation due to its symmetrical stabilizer operations [23]. For instance, a code distance-7 color code 6.6.6 requires only 37 qubits compared to 97 qubits for a distance-7 surface code. This substantial reduction makes color codes particularly attractive for quantum systems with limited qubits.

Recent advancements by the Google QuantumAI team highlight the use of color codes in magic state cultivation [18], showcasing their potential to reduce overhead in resource-intensive quantum algorithms. However, color codes introduce unique challenges for leakage detection due to their smaller syndrome bit patterns. In surface codes, each data qubit connects to four parity qubits, generating 4-bit syndrome patterns. In contrast, color codes often connect data qubits to fewer parity qubits, resulting in 3-bit patterns, with edge and corner qubits producing only 2-bit or 1-bit patterns, as illustrated in Figure 8(a). This limited information from single-round syndrome measurements reduces the accuracy of GLADIATOR, leading to unnecessary applications of leakage reduction circuits.

[2]Speculative scheduling and execution of LRCs makes synchronization [31] necessary

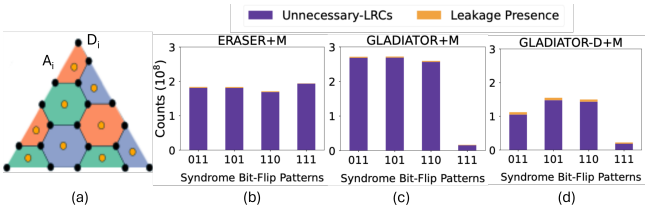

Figure 8: (a) Color Code (d=5), with data and parity qubits in black and orange. LRCs distribution (b) ERASER+M (c) GLADIATOR+M (d) GLADIATOR-D+M for 3-bit ERASER patterns

To address this limitation from syndrome measurements for leakage speculation, we introduce GLADIATOR-D, that incorporates temporal information by leveraging an additional round of measurements. By analyzing dependent syndrome patterns across time, as shown in Figure 8(d), GLADIATOR-D enhances leakage speculation accuracy, reducing misclassifications and minimizing unnecessary LRCs. This integration of spatial and temporal data is particularly effective for complex QEC codes like color codes, where spatial information alone is insufficient. Leveraging temporal correlations, GLADIATOR-D ensures precise leakage detection and robust mitigation, aiding in advancing fault-tolerant quantum computing.

5.2 Leveraging Syndrome Temporal Correlation

Leakage speculation relies on syndrome patterns to detect leakage events, as leakage errors often persist across multiple QEC cycles. Deferring diagnosis by one additional QEC round can significantly improve detection accuracy by leveraging the evolution of consecutive syndrome patterns. Syndrome s_1 from round 1 influences the subsequent syndrome s_2 in round 2, depending on whether the error is caused by leakage or a non-leakage event.

For instance, when the first-round syndrome pattern s_1, is "0011," ERASER would typically insert an LRC, while GLADIATOR might not. This initial decision could be incorrect, as both leakage and non-leakage errors can produce the "0011" pattern. However, observing the pattern evolution can clarify the situation. If "0011" transitions to "0011(s_1) 1111(s_2)" by next round, this pattern likely indicates a non-leakage X-error, such as one affecting the data qubit after the second CNOT in round 1. Conversely, if it evolves into "0011 0101," it suggests leakage, as a leaked qubit causes random bit-flips that a single non-leakage error cannot explain.

Deferring leakage detection by one round significantly reduces false positives. *For example,* GLADIATOR-D, *which uses two rounds (8-bit syndromes), flags 70 out of 256 patterns as leakage, compared to 121 out of 256 flagged by* ERASER .

Deferring leakage speculation to gather more information is particularly effective for color codes, which have limited syndrome information due to smaller patterns: 2-bit for edge qubits, 1-bit for corner qubits, and 3-bit for the rest. This limitation causes ERASER to over-apply Leakage Reduction Circuits (LRCs), as shown in Figure 8(b). For example, out of all 3-bit patterns, ERASER flags 4 out of 8 patterns as leakage, while GLADIATOR flags 3, offering only slight improvement shown in Figure 8(c). By leveraging temporal correlations, GLADIATOR-D captures leakage propagation effects more effectively. It flags 11 out of 64 patterns compared to 16 flagged by ERASER , as shown in Figure 8(d). For instance, in the 3-bit pattern "010," observing its evolution to "010 111" suggests a non-leakage error, while "010 110" indicates leakage. This approach enhances the resilience of color codes by reducing unnecessary LRCs.

Design modifications. To support GLADIATOR-D, which leverages multi-round syndrome history, we extend the design to enable deferred leakage detection. This version builds leakage and non-leakage graphs from two consecutive QEC rounds, expanding the input to the sequence checker from 5-bit to 10-bit patterns. These extended patterns, generated by the Data Parity Adjacency Generator, improves the leakage diagnosis to mitigate leakage while increasing LUT overhead by at most 4×.

6 Methodology

Circuit Noise Model. We assumes a physical error rate of p, accounting for several key error sources. Data qubit depolarization occurs at the start of each QEC round with a probability p. We apply readout errors during qubit measurement, while gate operation faults apply depolarizing errors to qubit operands after each *CNOT* or H gate, both with a probability p. Initialization errors occur during qubit resets, also modeled with a probability p.

Leakage Errors and Propagation. Leakage errors, primarily caused by two-qubit gates such as CNOT operations, include environment driven leakage injected into data qubits at the start of each QEC round with a probability p_l, and leakage from gate operations occurring with the same probability. *Leakage propagation* is modeled with a 10% probability, transferring leakage to the target qubit during CNOT operations from control qubits. In the remaining 90% of cases, a random Pauli error (I, X, Y, Z) is applied to the target qubit to model random bit-flip errors due to *gate-operation induced leakage.* Additionally, *environment-induced leakage* is modeled by injecting leakage into data qubits at the beginning of each QEC round with a probability p_l. Our IBM hardware characterization in Section 2.3, confirms similar leakage transport and bit-flip effects.

Modeling Error Profiles. We extend the ERASER framework by introducing the leakage ratio ($lr = \frac{p_l}{p}$) to quantify the intensity of leakage errors relative to non-leakage errors. Similarly we use *mlr* for Multi-Level Readout to model readout errors for leaked states as $mlr \times p$. For evaluations, we assume $lr = 0.1$ and $mlr = 10$, representing a 10× higher readout error for $|2\rangle$.[3] We test across $lr = 0.01, 0.1, 1$ to ensure adaptability across diverse noise conditions.

Interplay of Leakage with other Noise Sources. We simulate stabilizer circuits with errors introduced at each step of the QEC cycle. Initialization and leakage errors occur at the start of each round, while CZ gates introduce both depolarizing and leakage errors. Measurement errors include those arising from multi-level measurement in the presence of leakage. This comprehensive modeling captures the interaction of leakage with other noise sources, evaluating their combined impact on QEC performance.

Simulation Framework. We modified the ERASER artifact for our evaluation. We also use Google's Stim [17] simulator to simulate leakage errors through the Tableau simulator.

Quantum Hardware. We conducted leakage characterization on 7-qubit IBM machines (Lagos, Jakarta, Perth) using Qiskit and

[3]This is a conservative assumption. Several experimental works [28, 35] show MLR inaccuracies to be at most 2x worse compared to two-level readout.

Qiskit Pulse [37]. Each experiment used 10,000 shots. Leakage was induced by applying an X gate followed by a calibrated $|1\rangle \rightarrow |2\rangle$ pulse, following IBM's official guide [20]. These experiments were performed before IBM retired pulse-level access in September 2024 [21]. Despite this, the observed leakage effects are consistent with prior studies [3, 11, 15, 16].

Scaling Simulations using Leakage Sampling. As leakage spreads and population grows over time, evaluating performance for larger QEC cycles becomes critical. However, ERASER's methodology is limited to 10 QEC cycles due to the computational cost of capturing meaningful leakage events. For $p_l = 10^{-4}$ and $p = 10^{-3}$ and a code distance $d = 7$, only 0.5% of runs begin with at least one qubit in a leaked state, making it inefficient for long running evaluations. To address this, we introduce leakage sampling, where simulations start with at least one leaked data qubit to focus on scenarios where leakage is actively present.

This method of leakage sampling allows us to extend simulations to 100× more QEC cycles while significantly reducing computational overhead. This method ensures that the performance of speculation policies is evaluated under relevant conditions, particularly when leakage is present. We evaluate our policies over 100 QEC cycles using leakage sampling, with ERASER as the baseline.

7 Evaluations

To assess the efficacy of our design, we utilize - *(1) Data Leakage Population Ratio (DLP):* The average fraction of leaked data qubits per round, computed across all simulation shots. This metric quantifies sustained leakage over n QEC rounds, assessing the effectiveness of leakage mitigation. *(2) Total LRC Usage Rate:* The total number of LRCs applied per round across all simulation runs. *(3) False Negatives and False Positives:* The number of undetected leakages and unnecessary LRC applications.*(4) Logical Error Rate (LER):* The probability of a logical error after decoding the syndromes.

7.1 Speculation Accuracy and LRC usage

The efficacy of leakage speculation hinges on efficient mitigation of leakage through optimal LRC insertion. False Positives (FPs) introduce unnecessary physical errors and increase execution time, while False Negatives (FNs) represent undetected leakages that allow leakage to spread, leading to accumulation. Multi-Level Readout (MLR) lowers FNs by detecting leakage transport, preventing leakage accumulation. MLR errors can introduce FPs unrelated to syndrome patterns, necessitating better MLR accuracy. Together, lower FPs, FNs, and LRC usage directly translate to improved error rates and better overall system performance.

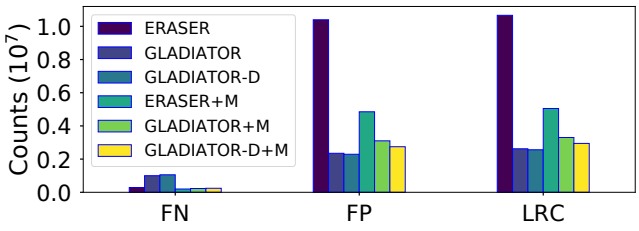

Figure 9: False Negative (FN), False Positive (FP), Leakage Reduction Circuit (LRC) counts with the leakage error as $p_{leak} = 10^{-4}$, gate error p=10^{-3} for surface code (d=7)

GLADIATOR and GLADIATOR-D significantly reduces the false positive rate, i.e, the number of un-necessary LRCs inserted compared to the baseline ERASER. Figure 9 shows the False Negative (FN) rate, False Positive (FP) rate, and total LRCs inserted by different policies for leakage ratio $lr = 0.1$, code distance 7 and error probability 10^{-3}, similar to ERASER. Compared to ERASER+M, GLADIATOR+M reduces the false positive rate by 1.56×, and GLADIATOR-D+M by 1.76× with an increase in false negatives by 1.16×, 1.22×, leading to a reduction in number of LRCs inserted by 1.53× and 1.71× respectively.

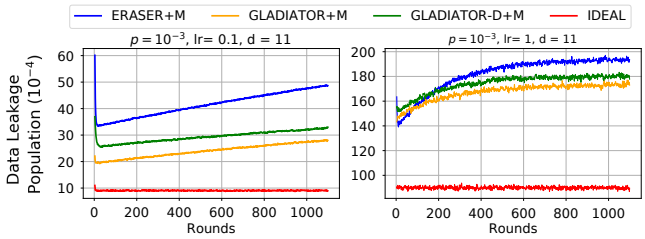

Figure 10: Data leakage (lower is better) for code distance d, leakage ratio lr and run for $100d$ rounds and 10^5 repetitions

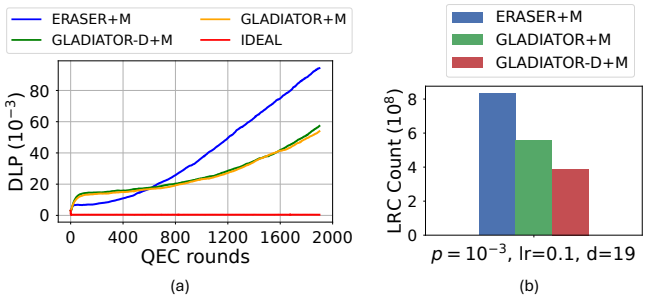

Figure 11: Comparison of (a) Data Leakage Population (b) LRC usage for color code with d=19 run for 100 QEC cycles

7.2 Impact on Data Leakage Population

Short-term simulations over 10 QEC cycles as shown in ERASER+M, fail to reveal the leakage accumulation over extended rounds. To better understand this accumulation, we evaluate the performance of GLADIATOR+M and GLADIATOR-D+M over 100 QEC cycles. For a higher leakage intensity ($lr = 1$), we observe a crossover point between ERASER+M and the proposed policies, GLADIATOR+M and GLADIATOR-D+M, occurring between 100 and 200 rounds, for **surface codes**, as shown in Figure 10. Additionally, Figure 11(a) also shows the increasing gap between GLADIATOR+M and ERASER+M, with increasing QEC rounds for **color codes**. Leakage growth differs over time, with a slower leakage accumulation rate for GLADIATOR-D+M and GLADIATOR+M compared to ERASER+M.

Specifically, for code distance $d = 11$ and $lr = 0.1$, the data leakage population for GLADIATOR-D+M and GLADIATOR+M is 1.47× and

1.73× lower than ERASER+M, respectively, after $100 \times d$ QEC rounds. This demonstrates superior leakage mitigation due to better speculation. The slower rate of leakage accumulation for GLADIATOR-D+M (1.14× lower than GLADIATOR+M) indicates the advantages of deferred speculation for sustaining extended QEC runs of thousands of rounds. While in surface codes GLADIATOR+M reduces leakage more overall, the finer control offered by GLADIATOR-D+M makes it a be alternative for mitigating leakage growth over extended QEC cycles especially for color codes.

7.3 Impact on Logical Error Rate

Figure 12 compares the logical error rate (LER) across three leakage mitigation strategies—GLADIATOR+M , ERASER+M , and ALWAYS-LRC—across multiple surface code distances. We observe that closed-loop approaches significantly outperform open-loop strategies, with GLADIATOR+M achieving up to an order of magnitude lower LER than ALWAYS-LRC at comparable code distances.

We evaluate the suppression factor $\Lambda_{d/(d+2)} = \frac{\epsilon_{d+2}}{\epsilon_d}$, to quantify scalability by capturing the rate of logical error reduction as the code distance increases. For distances $d = \{5, 7, 9\}$, GLADIATOR+M achieves an average Λ of approximately 3.71, compared to 3.38 for ERASER+M —indicating better error suppression with increasing code size. This improvement is closely tied to better leakage control. As leakage error rate decreases, GLADIATOR+M adaptively reduces unnecessary LRC applications, resulting in nearly 2× lower residual leakage population and speculation inaccuracy (see Table 4) translating to improved LER for $p = 10^{-4}$ as shown in Figure 13(a).

Table 4: Total Leakage Equilibrium across Leakage Ratios and Speculation Inaccuracy across Physical Error Rates ($d = 11$)

Method	Leakage Equilibrium			Spec. Inaccuracy	
	lr = 0.01	lr = 0.1	lr = 1.0	p = 10^{-3}	p = 10^{-4}
GLADIATOR+M	0.00148	0.00386	0.01538	0.01335	0.00238
ERASER+M	0.00276	0.00660	0.01782	0.02606	0.00736

To highlight the cost of unmitigated leakage, we include a NO-LRC baseline in Figure 12. Unlike other schemes, its LER increases with code distance, almost 200× GLADIATOR at $d = 9$, to demonstrate leakage accumulation without LRCs degrading QEC.

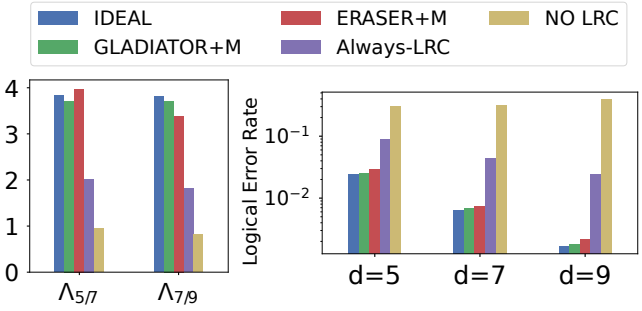

Figure 12: Logical Error Rate (LER) for increasing code distances d =5,7,9 run for $10d$ cycles with $p = 10^{-3}$ and $lr = 0.1$

7.4 Generalizes to Color Codes and LDPC Codes

Table 5 shows that GLADIATOR outperforms ERASER across a range of codes by reducing the number of unnecessary LRCs, lowering residual data-leakage population (DLP), and enabling faster QEC execution. While both methods perform well on the surface code, GLADIATOR still achieves 1.7× fewer LRCs, 1.67× lower DLP, and 1.69× faster QEC execution. These benefits are even more pronounced on Hypergraph Product (HGP) codes, GLADIATOR reduces LRC count and QEC execution time by nearly 4×, and lowers DLP by 1.88×. Fewer LRCs not only reduce circuit depth but also shorten the QEC cycle, as LRCs are typically inserted after entangling gates and extend the duration of quantum error correction rounds.

To quantify QEC performance, we measure the cycle time as the total QEC execution time normalized by the number of rounds and shots. We further compute the average LRC count per round per shot and convert this into an estimated latency overhead, assuming SWAP-based LRC implementations. This allows us to approximate the time cost directly attributable to leakage mitigation. We then compare the normalized QEC cycle times for GLADIATOR and ERASER across different codes. This analysis reveals the overhead reduction achieved by GLADIATOR through more selective LRC application, resulting in faster and more efficient QEC execution.

Unlike ERASER, which is tailored to surface-code symmetry, GLADIATOR generalizes across surface code, color code, Hypergraph Product (HGP) code, and Balanced Product Cyclic (BPC) code using only syndrome history. These results highlight GLADIATOR's effectiveness and broad applicability to heterogeneous quantum error correction codes, the direction pursued by Google, IBM in their push towards large-scale, scalable quantum computing.

Table 5: Reduction Factors for GLADIATOR over ERASER

Metric / Code	Surface Code	Color Code	HGP Code	BPC Code
LRCs	1.71x	1.5x	3.86x	1.51x
DLP	1.67x	1.54x	1.88x	1.02x
QEC Cycle Time	1.69x	1.5x	3.83x	1.505x

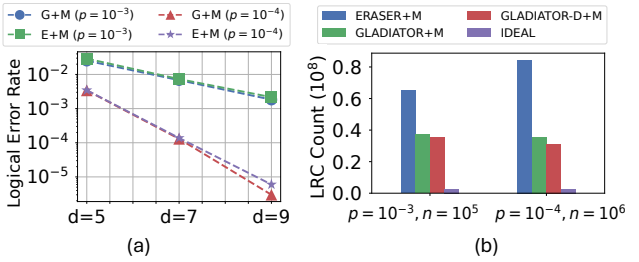

Figure 13: Comparison of (a) Logical error rate (b) LRCs usage for error probability $p = 10^{-3}$ and $p = 10^{-4}$ run for n shots

7.5 Sensitivity Study

Sensitivity to Error Rate: Leakage occurrence directly correlates with LRC usage and LER, with higher p_{leak} leading to increased LRC counts per shot and sustained leakage. Similarly, as operational

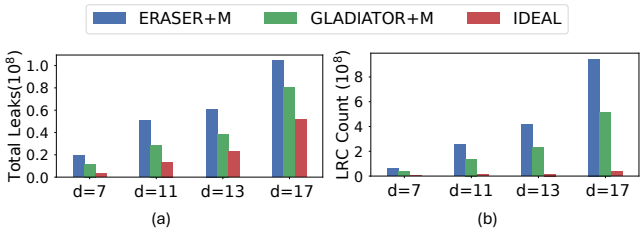

Figure 14: For code distances d = 7,11,13,17 run for $100d$ cycles the plots show a) Total leakages and b) Total LRCs utilized

error probability (p) decreases, both LER (Figure 13(a)) and LRC usage (Figure 13(b)) decreases. Among the evaluated methods for LRC counts, GLADIATOR-D+M shows superior adaptability by leveraging additional rounds for effective leakage management. Table 4 shows the leakage population at the equilibrium state is lower for GLADIATOR+M compared to ERASER+M for different lr.

Sensitivity to Code Distance: While error rates generally decrease with increasing code distance d, the total leakage does not follow the same trend. Figure 14(a) illustrates that even under an ideal policy, assuming perfect speculation, leakage is still introduced by LRCs and the syndrome generation circuit due to the quadratic growth in the number of qubits and gates required as the code distance scales. Our evaluations show that qubit leakage becomes increasingly pronounced as the code distance grows. This increase in total leakage directly translates to a higher count of LRCs needed for leakage mitigation. LRC usage gap between GLADIATOR+M and ERASER+M widens with increasing code distance, demonstrating the efficiency and scalability of our leakage mitigation strategy.

Impact on Execution Depth. ALWAYS-LRC applies 60 LRCs per round, increasing execution depth by 20% for d=11. GLADIATOR applies only 1.22 LRCs per round ($\approx 50\times$ fewer), resulting in only 0.4% increase in execution depth compare to no LRC. The reduction factor in LRCs per round grows from $\approx 40\times$ at $d = 7$ to $\approx 50\times$ at $d = 17$, showing GLADIATOR is more effective at larger code distances.

7.6 Speculating Leakage Mobility

On real quantum hardware, both how often qubits leak and how easily that leakage spreads (leakage mobility) can vary. This plays a crucial role in how well different mitigation strategies work. For example, walking surface codes use extra operations to move leakage from data qubits to ancilla qubits, where it can be detected and reset. But this only works well when leakage mobility is low and the leakage stays put unless intentionally moved [13].

To adapt to different hardware behaviors, we introduce a practical way to estimate leakage mobility. Our approach combines GLADIATOR's speculative detection, which flags likely leaked data qubits, with multi-level readout (MLR) signals that reveal leakage on ancilla qubits. By measuring how often an ancilla qubit is leaked when its neighboring data qubit is flagged, we estimate the leakage mobility. Based on prior work [13], we use a 5% threshold: if the conditional probability is below 5%, we classify the system as having low mobility; if it's above, we consider it high mobility.

Our method accurately classifies leakage mobility regimes, helping determine whether the system is best suited for open-loop or closed-loop mitigation. Low mobility supports simpler strategies like Staggered Always-LRC or walking codes to prevent leakage buildup. High mobility, on the other hand, favors dynamic, feedback-driven approaches like GLADIATOR.

Table 6: Leakage Mobility Classification via GLADIATOR

Mobility (%)	1.0	2.5	5.0	6.0	9.0
True Regime	Low	Low	High	High	High
Accuracy (%)	100	100	50	100	100

8 Conclusion

Leakage errors pose a fundamental challenge to fault-tolerant quantum computing by corrupting syndrome extraction. While prior mitigation strategies like ERASER apply leakage reduction circuits (LRCs) speculatively using a fixed heuristic, they suffer from high false positive rates, leading to unnecessary LRC insertions that degrade performance and reliability.

This paper introduces GLADIATOR, a principled, model-driven framework that speculates leakage events by analyzing syndrome patterns through an offline-constructed, code-aware error graph informed by device calibration data. GLADIATOR approach generalizes across QEC codes—surface, color, and beyond—adapting to hardware-specific noise profiles without relying on hardcoded rules or code-specific assumptions. Through accurate leakage speculation, GLADIATOR significantly reduces false positives and unnecessary LRCs, achieving up to 3× fewer LRCs, 16% lower logical error rates, and $1.7 - 3.9\times$ shorter QEC cycles, resulting in significant speedups in application runtimes compared to ERASER [43].

Acknowledgments

The authors would like to thank Benjamin Lienhard, Satvik Maurya, and Ayushi Dubal for their feedback on previous versions of this paper. The authors also thank Professor Guri Sohi for the help with the computing infrastructure. This research was supported by an NSF CAREER Award #2340267, and the Office of the Vice Chancellor for Research (OVCR) at the University of Wisconsin-Madison, with the support of Wisconsin Alumni Research Foundation (WARF).

References

[1] Muhammad AbuGhanem. 2025. IBM quantum computers: evolution, performance, and future directions. https://doi.org/10.1007/s11227-025-07047-7

[2] Rajeev Acharya, 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, 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 Mandrá, Leigh S. Martin, Steven Martin, Orion 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, 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, 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, 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, and Nicholas Zobrist. 2024. Quantum error correction below the surface code threshold. arXiv:2408.13687 [quant-ph] https://arxiv.org/abs/2408.13687

[3] Rajeev Acharya, Igor Aleiner, Richard Allen, Trond I. Andersen, Markus Ansmann, Frank Arute, Kunal Arya, Abraham Asfaw, Juan Atalaya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Joao Basso, Andreas Bengtsson, Sergio Boixo, Gina Bortoli, Alexandre Bourassa, Jenna Bovaird, Leon Brill, Michael Broughton, Bob B. Buckley, David A. Buell, Tim Burger, Brian Burkett, Nicholas Bushnell, Yu Chen, Zijun Chen, Ben Chiaro, Josh Cogan, Roberto Collins, Paul Conner, William Courtney, Alexander L. Crook, Ben Curtin, Dripto M. Debroy, Alexander Del Toro Barba, Sean Demura, Andrew Dunsworth, Daniel Eppens, Catherine Erickson, Lara Faoro, Edward Farhi, Reza Fatemi, Leslie Flores Burgos, Ebrahim Forati, Austin G. Fowler, Brooks Foxen, William Giang, Craig Gidney, Dar Gilboa, Marissa Giustina, Alejandro Grajales Dau, Jonathan A. Gross, Steve Habegger, Michael C. Hamilton, Matthew P. Harrigan, Sean D. Harrington, Oscar Higgott, Jeremy Hilton, Markus Hoffmann, Sabrina Hong, Trent Huang, Ashley Huff, William J. Huggins, Lev B. Ioffe, Sergei V. Isakov, Justin Iveland, Evan Jeffrey, Zhang Jiang, Cody Jones, Pavol Juhas, Dvir Kafri, Kostyantyn Kechedzhi, Julian Kelly, Tanuj Khattar, Mostafa Khezri, Mária Kieferová, Seon Kim, Alexei Kitaev, Paul V. Klimov, Andrey R. Klots, Alexander N. Korotkov, Fedor Kostritsa, John Mark Kreikebaum, David Landhuis, Pavel Laptev, Kim-Ming Lau, Lily Laws, Joonho Lee, Kenny Lee, Brian J. Lester, Alexander Lill, Wayne Liu, Aditya Locharla, Erik Lucero, Fionn D. Malone, Jeffrey Marshall, Orion Martin, Jarrod R. McClean, Trevor McCourt, Matt McEwen, Anthony Megrant, Bernardo Meurer Costa, Xiao Mi, Kevin C. Miao, Masoud Mohseni, Shirin Montazeri, Alexis Morvan, Emily Mount, Wojciech Mruczkiewicz, Ofer Naaman, Matthew Neeley, Charles Neill, Ani Nersisyan, Hartmut Neven, Michael Newman, Jiun How Ng, Anthony Nguyen, Murray Nguyen, Murphy Yuezhen Niu, Thomas E. O'Brien, Alex Opremcak, John Platt, Andre Petukhov, Rebecca Potter, Leonid P. Pryadko, Chris Quintana, Pedram Roushan, Nicholas C. Rubin, Negar Saei, Daniel Sank, Kannan Sankaragomathi, Kevin J. Satzinger, Henry F. Schurkus, Christopher Schuster, Michael J. Shearn, Aaron Shorter, Vladimir Shvarts, Jindra Skruzny, Vadim Smelyanskiy, W. Clarke Smith, George Sterling, Doug Strain, Marco Szalay, Alfredo Torres, Guifre Vidal, Benjamin Villalonga, Catherine Vollgraff Heidweiller, Theodore White, Cheng Xing, Z. Jamie Yao, Ping Yeh, Juhwan Yoo, Grayson Young, Adam Zalcman, Yaxing Zhang, Ningfeng Zhu, and Google Quantum AI. 2023. Suppressing quantum errors by scaling a surface code logical qubit. *Nature* 614, 7949 (01 Feb 2023), 676–681. https://doi.org/10.1038/s41586-022-05434-1

[4] Panos Aliferis and Barbara M. Terhal. 2007. Fault-tolerant quantum computation for local leakage faults. *Quantum Info. Comput.* 7, 1 (jan 2007), 139–156.

[5] F. Battistel, B.M. Varbanov, and B.M. Terhal. 2021. Hardware-Efficient Leakage-Reduction Scheme for Quantum Error Correction with Superconducting Transmon Qubits. https://doi.org/10.1103/prxquantum.2.030314

[6] 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

[7] 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

[8] Nikolas P. Breuckmann and Jens Niklas Eberhardt. 2021. Quantum Low-Density Parity-Check Codes. https://doi.org/10.1103/prxquantum.2.040101

[9] Benjamin J. Brown, Naomi H. Nickerson, and Dan E. Browne. 2016. Fault-tolerant error correction with the gauge color code. https://doi.org/10.1038/ncomms12302

[10] Natalie C. Brown and Kenneth R. Brown. 2019. Leakage mitigation for quantum error correction using a mixed qubit scheme. https://doi.org/10.1103/physreva.100.032325

[11] Natalie C. Brown, Andrew Cross, and Kenneth R. Brown. 2020. Critical faults of leakage errors on the surface code. , 286-294 pages. https://doi.org/10.1109/QCE49297.2020.00043

[12] Zhenyu Cai, Michael A. Fogarty, Simon Schaal, Sofia Patomäki, Simon C. Benjamin, and John J. L. Morton. 2019. A Silicon Surface Code Architecture Resilient Against Leakage Errors. *Quantum* 3 (Dec. 2019), 212. https://doi.org/10.22331/q-2019-12-09-212

[13] Joan Camps, Ophelia Crawford, György P. Gehér, Alexander V. Gramolin, Matthew P. Stafford, and Mark Turner. 2024. Leakage Mobility in Superconducting Qubits as a Leakage Reduction Unit. arXiv:2406.04083 [quant-ph] https://arxiv.org/abs/2406.04083

[14] Alec Eickbusch, Matt McEwen, Volodymyr Sivak, Alexandre Bourassa, Juan Atalaya, Jahan Claes, Dvir Kafri, Craig Gidney, Christopher W. Warren, Jonathan Gross, Alex Opremcak, Nicholas Zobrist Kevin C. Miao, Gabrielle Roberts, Kevin J. Satzinger, Andreas Bengtsson, Matthew Neeley, William P. Livingston, Alex Greene, Rajeev, Acharya, Laleh Aghababaie Beni, Georg Aigeldinger, Ross Alcaraz, Trond I. Andersen, Markus Ansmann, Frank, Arute, Kunal Arya, Abraham Asfaw, Ryan Babbush, Brian Ballard, Joseph C. Bardin, Alexander Bilmes, 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, Alexander Del Toro Barba, Sean Demura, Laura De Lorenzo, Agustin Di Paolo, Paul Donohoe, Ilya K. Drozdov, Andrew Dunsworth, 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, Tan Ha, Steve Habegger, Monica Hansen, Matthew P. Harrigan, Sean D. Harrington, Stephen Heslin, Paula Heu, Oscar Higgott, Reno Hiltermann, Jeremy Hilton, Hsin-Yuan Huang, Ashley Huff, William J. Huggins, Evan Jeffrey, Zhang Jiang, Xiaoxuan Jin, Cody Jones, Chaitali Joshi, Pavol Juhas, Andreas Kabel, Hui Kang, Amir, H. Karamlou, Kostyantyn Kechedzhi, Trupti Khaire, Tanuj Khattar, Mostafa Khezri, Seon Kim, Bryce Kobrin, Alexander N. Korotkov, Fedor Kostritsa, John Mark Kreikebaum, Vladislav D. Kurilovich, David Landhuis, Tiano, Lange-Dei, Brandon W. Langley, Kim-Ming Lau, Justin Ledford, Kenny Lee, Brian J. Lester, Loïck Le Guevel, Wing, Yan Li, Alexander T. Lill, Aditya Locharla, Erik Lucero, Daniel Lundahl, Aaron Lunt, Sid Madhuk, Ashley Maloney, Salvatore Mandrà, Leigh S. Martin, Orion Martin, Cameron Maxfield, Jarrod R. McClean, Seneca Meeks, Anthony, Megrant, Reza Molavi, Sebastian Molina, Shirin Montazeri, Ramis Movassagh, Michael Newman, Anthony Nguyen, Murray Nguyen, Chia-Hung Ni, Logan Oas, Raymond Orosco, Kristoffer Ottosson, Alex Pizzuto, Rebecca Potter, Orion Pritchard, Chris Quintana, Ganesh Ramachandran, Matthew J. Reagor, David M. Rhodes, Eliott Rosenberg, Elizabeth Rossi, Kannan Sankaragomathi, Henry F. Schurkus, Michael J. Shearn, Aaron Shorter, Noah Shutty, 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, Adam Zalcman, Yaxing, Zhang, Ningfeng Zhu, Sergio Boixo, Julian Kelly, Vadim Smelyanskiy, Hartmut Neven, Dave Bacon, Zijun Chen, Paul V. Klimov, Pedram Roushan, Charles Neill, Yu Chen, and Alexis Morvan. 2024. Demonstrating dynamic surface codes. arXiv:2412.14360 [quant-ph] https://arxiv.org/abs/2412.14360

[15] Austin G. Fowler. 2013. Coping with qubit leakage in topological codes. *Phys. Rev. A* 88 (Oct 2013), 042308. Issue 4. https://doi.org/10.1103/PhysRevA.88.042308

[16] Joydip Ghosh, Austin G. Fowler, John M. Martinis, and Michael R. Geller. 2013. Understanding the effects of leakage in superconducting quantum-error-detection circuits. https://doi.org/10.1103/physreva.88.062329

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

[18] Craig Gidney, Noah Shutty, and Cody Jones. 2024. Magic state cultivation: growing T states as cheap as CNOT gates. arXiv:2409.17595 [quant-ph] https://arxiv.org/abs/2409.17595

[19] D. Hayes, D. Stack, B. Bjork, A.C. Potter, C.H. Baldwin, and R.P. Stutz. 2020. Eliminating Leakage Errors in Hyperfine Qubits. https://doi.org/10.1103/physrevlett.124.170501

[20] IBM Quantum. 2023. Accessing Higher Energy States with Qiskit Pulse. https://github.com/Qiskit/textbook/blob/main/notebooks/quantum-hardware-pulses/accessing_higher_energy_states.ipynb. Accessed: 2025-06-21.

[21] IBM Quantum. 2024. Qiskit 2.0 Migration Guide: Pulse Access Retirement. https://quantum.cloud.ibm.com/docs/en/migration-guides/qiskit-2.0#qiskitpulse. Accessed: 2025-06-21.

[22] Mingyu Kang, Yingjia Lin, Hanwen Yao, Mert Gökduman, Arianna Meinking, and Kenneth R. Brown. 2025. QUITS: A modular Qldpc code circUIT Simulator. arXiv:2504.02673 [quant-ph] https://arxiv.org/abs/2504.02673

[23] Aleksander Kubica and Michael E. Beverland. 2015. Universal transversal gates with color codes: A simplified approach. *Phys. Rev. A* 91 (Mar 2015), 032330. Issue 3. https://doi.org/10.1103/PhysRevA.91.032330

[24] Nathan Lacroix, Luca Hofele, Ants Remm, Othmane Benhayoune-Khadraoui, Alexander McDonald, Ross Shillito, Stefania Lazar, Christoph Hellings, Francois Swiadek, Dante Colao-Zanuz, Alexander Flasby, Mohsen Bahrami Panah, Michael Kerschbaum, Graham J. Norris, Alexandre Blais, Andreas Wallraff, and Sebastian Krinner. 2023. Fast Flux-Activated Leakage Reduction for Superconducting Quantum Circuits. arXiv:2309.07060 [quant-ph]

[25] H. Levine, A. Haim, J. S. C. Hung, N. Alidoust, M. Kalaee, L. DeLorenzo, E. A. Wollack, P. Arrangoiz-Arriola, A. Khalajhedayati, R. Sanil, H. Moradinejad, Y. Vaknin, A. Kubica, D. Hover, S. Aghaeimeibodi, J. A. Alcid, C. Baek, J. Barnett, K. Bawdekar, P. Bienias, H. A. Carson, C. Chen, L. Chen, H. Chinkezian, E. M. Chisholm, A. Clifford, R. Cosmic, N. Crisosto, A. M. Dalzell, E. Davis, J. M. D'Ewart, S. Diez, N. D'Souza, P. T. Dumitrescu, E. Elkhouly, M. T. Fang, Y. Fang, S. Flammia, M. J. Fling, G. Garcia, M. K. Gharzai, A. V. Gorshkov, M. J. Gray, S. Grimberg, A. L. Grimsmo, C. T. Hann, Y. He, S. Heidel, S. Howell, M. Hunt, J. Iverson, I. Jarrige, L. Jiang, W. M. Jones, R. Karabalin, P. J. Karalekas, A. J. Keller, D. Lasi, M. Lee, V. Ly, G. MacCabe, N. Mahuli, G. Marcaud, M. H. Matheny, S. McArdle, G. McCabe, G. Merton, C. Miles, A. Milsted, A. Mishra, L. Moncelsi, M. Naghiloo, K. Noh, E. Oblepias, G. Ortuno, J. C. Owens, J. Pagdilao, A. Panduro, J.-P. Paquette, R. N. Patel, G. Peairs, D. J. Perello, E. C. Peterson, S. Ponte, H. Putterman, G. Refael, P. Reinhold, R. Resnick, O. A. Reyna, R. Rodriguez, J. Rose, A. H. Rubin, M. Runyan, C. A. Ryan, A. Sahmoud, T. Scaffidi, B. Shah, S. Siavoshi, P. Sivarajah, T. Skogland, C.-J. Su, L. J. Swenson, J. Sylvia, S. M. Teo, A. Tomada, G. Torlai, M. Wistrom, K. Zhang, I. Zuk, A. A. Clerk, F. G. S. L. Brandão, A. Retzker, and O. Painter. 2024. Demonstrating a Long-Coherence Dual-Rail Erasure Qubit Using Tunable Transmons. *Phys. Rev. X* 14 (Mar 2024), 011051. Issue 1. https://doi.org/10.1103/PhysRevX.14.011051

[26] Sophia Fuhui Lin, Joshua Viszlai, Kaitlin N. Smith, Gokul Subramanian Ravi, Charles Yuan, Frederic T. Chong, and Benjamin J. Brown. 2024. Codesign of quantum error-correcting codes and modular chiplets in the presence of defects. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2* (La Jolla, CA, USA) *(ASPLOS '24)*. Association for Computing Machinery, New York, NY, USA, 216–231. https://doi.org/10.1145/3620665.3640362

[27] Shiang Yong Looi, Li Yu, Vlad Gheorghiu, and Robert B. Griffiths. 2008. Quantum-error-correcting codes using qudit graph states. https://doi.org/10.1103/physreva.78.042303

[28] Piero Luchi, Paolo E. Trevisanutto, Alessandro Roggero, Jonathan L. DuBois, Yaniv J. Rosen, Francesco Turro, Valentina Amitrano, and Francesco Pederiva. 2023. Enhancing Qubit Readout with Autoencoders. *Phys. Rev. Appl.* 20 (Jul 2023), 014045. Issue 1. https://doi.org/10.1103/PhysRevApplied.20.014045

[29] Yue Ma, Michael Hanks, and M. S. Kim. 2023. Non–Pauli Errors Can Be Efficiently Sampled in Qudit Surface Codes. *Phys. Rev. Lett.* 131 (Nov 2023), 200602. Issue 20. https://doi.org/10.1103/PhysRevLett.131.200602

[30] J. F. Marques, H. Ali, B. M. Varbanov, M. Finkel, H. M. Veen, S. L. M. van der Meer, S. Valles-Sanclemente, N. Muthusubramanian, M. Beekman, N. Haider, B. M. Terhal, and L. DiCarlo. 2023. All-Microwave Leakage Reduction Units for Quantum Error Correction with Superconducting Transmon Qubits. *Phys. Rev. Lett.* 130 (Jun 2023), 250602. Issue 25. https://doi.org/10.1103/PhysRevLett.130.250602

[31] Satvik Maurya and Swamit Tannu. 2025. Synchronization for Fault-Tolerant Quantum Computers. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture (ISCA '25)*. Association for Computing Machinery, New York, NY, USA, 1370–1385. https://doi.org/10.1145/3695053.3730991

[32] M. McEwen, D. Kafri, Z. Chen, J. Atalaya, K. J. Satzinger, C. Quintana, P. V. Klimov, D. Sank, C. Gidney, A. G. Fowler, F. Arute, K. Arya, B. Buckley, B. Burkett, N. Bushnell, B. Chiaro, R. Collins, S. Demura, A. Dunsworth, C. Erickson, B. Foxen, M. Giustina, T. Huang, S. Hong, E. Jeffrey, S. Kim, K. Kechedzhi, F. Kostritsa, P. Laptev, A. Megrant, X. Mi, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Niu, A. Paler, N. Redd, P. Roushan, T. C. White, J. Yao, P. Yeh, A. Zalcman, Yu Chen, V. N. Smelyanskiy, John M. Martinis, H. Neven, J. Kelly, A. N. Korotkov, A. G. Petukhov, and R. Barends. 2021. Removing leakage-induced correlated errors in superconducting quantum error correction. *Nature Communications* 12, 1 (19 Mar 2021), 1761. https://doi.org/10.1038/s41467-021-21982-y

[33] Kevin C. Miao, Matt McEwen, Juan Atalaya, Dvir Kafri, Leonid P. Pryadko, Andreas Bengtsson, Alex Opremcak, Kevin J. Satzinger, Zijun Chen, Paul V. Klimov, Chris Quintana, Rajeev Acharya, Kyle Anderson, Markus Ansmann, Frank Arute, Kunal Arya, Abraham Asfaw, Joseph C. Bardin, Alexandre Bourassa, Jenna Bovaird, Leon Brill, Bob B. Buckley, David A. Buell, Tim Burger, Brian Burkett, Nicholas Bushnell, Juan Campero, Ben Chiaro, Roberto Collins, Paul Conner, Alexander L. Crook, Ben Curtin, Dripto M. Debroy, Sean Demura, Andrew Dunsworth, Catherine Erickson, Reza Fatemi, Vinicius S. Ferreira, Leslie Flores Burgos, Ebrahim Forati, Austin G. Fowler, Brooks Foxen, Gonzalo Garcia, William Giang, Craig Gidney, Marissa Giustina, Raja Gosula, Alejandro Grajales Dau, Jonathan A. Gross, Michael C. Hamilton, Sean D. Harrington, Paula Heu, Jeremy Hilton, Markus R. Hoffmann, Sabrina Hong, Trent Huang, Ashley Huff, Justin Iveland, Evan Jeffrey, Zhang Jiang, Cody Jones, Julian Kelly, Seon Kim, Fedor Kostritsa, John Mark Kreikebaum, David Landhuis, Pavel Laptev, Lily Laws, Kenny Lee, Brian J. Lester, Alexander T. Lill, Wayne Liu, Aditya Locharla, Erik Lucero, Steven Martin, Anthony Megrant, Xiao Mi, Shirin Montazeri, Alexis Morvan, Ofer Naaman, Matthew Neeley, Charles Neill, Ani Nersisyan, Michael Newman, Jiun How Ng, Anthony Nguyen, Murray Nguyen, Rebecca Potter, Charles Rocque, Pedram Roushan, Kannan Sankaragomathi, Henry F. Schurkus, Christopher Schuster, Michael J. Shearn, Aaron Shorter, Noah Shutty, Vladimir Shvarts, Jindra Skruzny, W. Clarke Smith, George Sterling, Marco Szalay, Douglas Thor, Alfredo Torres, Theodore White, Bryan W. K. Woo, Z. Jamie Yao, Ping Yeh, Juhwan Yoo, Grayson Young, Adam Zalcman, Ningfeng Zhu, Nicholas Zobrist, Hartmut Neven, Vadim Smelyanskiy, Andre Petukhov, Alexander N. Korotkov, Daniel Sank, and Yu Chen. 2023. Overcoming leakage in quantum error correction. *Nature Physics* 19, 12 (01 Dec 2023), 1780–1786. https://doi.org/10.1038/s41567-023-02226-w

[34] F. Motzoi, J. M. Gambetta, P. Rebentrost, and F. K. Wilhelm. 2009. Simple Pulses for Elimination of Leakage in Weakly Nonlinear Qubits. *Phys. Rev. Lett.* 103 (Sep 2009), 110501. Issue 11. https://doi.org/10.1103/PhysRevLett.103.110501

[35] Chaithanya Naik Mude, Satvik Maurya, Benjamin Lienhard, and Swamit Tannu. 2025. Efficient and Scalable Architectures for Multi-Level Superconducting Qubit Readout. arXiv:2405.08982 [quant-ph] https://arxiv.org/abs/2405.08982

[36] Priya J. Nadkarni and Shayan Srinivasa Garani. 2021. Quantum error correction architecture for qudit stabilizer codes. *Phys. Rev. A* 103 (Apr 2021), 042420. Issue 4. https://doi.org/10.1103/PhysRevA.103.042420

[37] Qiskit Development Team. 2023. Qiskit: An open-source framework for quantum computing. https://qiskit.org. Accessed: 2025-06-21.

[38] V. V. Sivak, A. Eickbusch, B. Royer, S. Singh, I. Tsioutsios, S. Ganjam, A. Miano, B. L. Brock, A. Z. Ding, L. Frunzio, S. M. Girvin, R. J. Schoelkopf, and M. H. Devoret. 2023. Real-time quantum error correction beyond break-even. *Nature* 616, 7955 (01 Apr 2023), 50–55. https://doi.org/10.1038/s41586-023-05782-6

[39] Kaitlin N. Smith, Gokul Subramanian Ravi, Jonathan M. Baker, and Frederic T. Chong. 2022. Scaling Superconducting Quantum Computers with Chiplet Architectures. , 1092-1109 pages. https://doi.org/10.1109/MICRO56248.2022.00078

[40] Martin Suchara, Andrew W. Cross, and Jay M. Gambetta. 2015. Leakage suppression in the toric code. , 1119-1123 pages. https://doi.org/10.1109/ISIT.2015.7282629

[41] Martin Suchara, Andrew W. Cross, and Jay M. Gambetta. 2015. Leakage suppression in the toric code. , 1119-1123 pages. https://doi.org/10.1109/ISIT.2015.7282629

[42] Yugo Takada and Keisuke Fujii. 2024. Improving Threshold for Fault-Tolerant Color-Code Quantum Computing by Flagged Weight Optimization. https://doi.org/10.1103/prxquantum.5.030352

[43] Suhas Vittal, Poulami Das, and Moinuddin Qureshi. 2023. ERASER: Towards Adaptive Leakage Suppression for Fault-Tolerant Quantum Computing. , 509-525 pages.

[44] Yue Wu, Shimon Kolkowitz, Shruti Puri, and Jeff D. Thompson. 2022. Erasure conversion for fault-tolerant quantum computing in alkaline earth Rydberg atom arrays. https://doi.org/10.1038/s41467-022-32094-6

[45] Qian Xu, J. Pablo Bonilla Ataides, Christopher A. Pattison, Nithin Raveendran, Dolev Bluvstein, Jonathan Wurtz, Bane Vasic, Mikhail D. Lukin, Liang Jiang, and Hengyun Zhou. 2023. Constant-Overhead Fault-Tolerant Quantum Computation with Reconfigurable Atom Arrays. arXiv:2308.08648 [quant-ph]

[46] Jiaxuan Zhang, Yu-Chun Wu, and Guo-Ping Guo. 2024. Facilitating practical fault-tolerant quantum computing based on color codes. *Phys. Rev. Res.* 6 (Jul 2024), 033086. Issue 3. https://doi.org/10.1103/PhysRevResearch.6.033086

A Artifact Appendix

A.1 Abstract

This artifact contains the code to verify the key results of this paper. Specifically, this artifact will generate and plot data for Figures 4(b), 9, 10, 11, 12, 13 and 14. Additionally, the artifact contains code to verify the results of Table 2.

A.2 Artifact check-list (meta-information)

- **Algorithm: GLADIATOR, a leakage-detection algorithm.**
- **Compilation: GCC**
- **Binary: leakage**
- **Metrics: LRC Usage, Logical Error Rate (LER), Data Leakage Population (DLP), False Positivies and False Negatives**
- **Output: Data files and figures.**
- **How much disk space required (approximately)?: Not more than 5GB**
- **How much time is needed to prepare workflow (approximately)?: About a minute**
- **How much time is needed to complete experiments (approximately)?: 15-20hrs**
- **Publicly available?: Yes**
- **Code licenses (if publicly available)?: MIT License**
- **Archived (provide DOI)?: https://doi.org/10.5281/zenodo.16735148**

A.3 Description

A.3.1 How to access. The artifact for this work is available on Zenodo at https://doi.org/10.5281/zenodo.16735148

A.3.2 Software dependencies. The code is built by modifying the artifact from ERASER[43] using CMake v3.20.3. The compiler g++-12 and g++-13 are used for the evaluations, and to parallelize the experiments on computing clusters we use OpenMPI v4.x.x. CMake is used to package all other de- pendencies within the code.

Additionally for plotting the figures, we provide a python notebook, tested using Python v3.12, and the following packages are dependencies: matplotlib v3.6.1, numpy v1.23.4, scipy v1.9.2, and pandas v2.3.0, though it should work fine with newer versions.

A.4 Installation

For creating the data for Figures 4(b), 9, 10, 11, 12, 13 and 14, we encourage to use a build directory 'build' to avoid any issues. The executable leakage can be generated as follows,

```
$ cd build
$ cmake .. -DCMAKE_BUILD_TYPE=Release
$ make -j8
```

A.5 Experiment workflow

We explain how to generate the data for the main insights and results of this work, i.e, Figures 4(b), and from Figure 9 till 14. For each of the Figures, we provide `AE_Figure_<#>.sh` to generate the data for the corresponding figures. Additionaly, we also provide `AE_all_data_for_plots.sh` to generate all the data needed to plot the results, as follows,

```
$ cd leakage
$ ./AE_all_data_for_plots.sh <PROC>
```

where PROC is the number of processors used by OpenMPI to parallelize the experiments. This will create data for plotting in the `leakage/AE_results/ folder`.

Our evaluations are performed at $p = 10^{-3}$ and at $p = 10^{-4}$. We note that due to the lower logical error rate of $p = 10^{-4}$, evaluations will take much longer due to requiring significantly more shots. We recommend 100k shots for $p = 10^{-3}$ and 1M shots for $p = 10^{-4}$, as 100k is insufficient to obtain complete results for LER at $p = 10^{-4}$.

We recommend using a cluster with atleast 32 cores with sufficient memory, as the memory requirement to run experiments with larger distance codes is significant and may need many cores to complete in time. For reference, our evaluations for Figure 14 took total of 8 hours running on a cluster with 64 cores. The memory requirement arises from the need for storing measurement data for each experiment to run LER evaluations. As the code distance (d) increases to 17 and run for 1700 (100d) rounds for 1M shots, the memory requirement increases to store increase over 20GB.

For Figure 14, as run experiments for code distance (d) till 17, and for 100d rounds each. This might lead to heavy memory usage. We would recommend to comment out the lines that uses `stim::write_table_data` in the leakage/src/experiment.cpp and recompile before running the scripts for running leakage population evaluations. For LER evaluations, you need to uncomment these lines again and recompile before running the scripts.

The script `AE_all_data_for_plots_wLER.sh` runs LER evaluations and `AE_all_data_for_plots_wo_LER.sh` runs other evaluations that doesn't require storing measurement data.

To run the data generation seperately for LER evaluations and leakage population evaluations, use the following,

```
$ cd build
$ cmake .. -DCMAKE_BUILD_TYPE=Release
$ make -j8
$ cd ../leakage
$ ./AE_all_data_for_plots_wLER.sh <PROC>
```

To avoid unnecessary usage of memory, comment out the lines with `stim::write_table_data` from leakage/src/experiment.cpp, recompile and then run the other script.

```
$ make -j4
$ cd ../leakage
$ ./AE_all_data_for_plots_wo_LER.sh <PROC>
```

A.6 Evaluation and expected results

The results for LER, LRC and leakage population should be follow the same as the reported values in the paper, perhaps with slight deviations due to randomness.

A.7 Experiment customization

GLADIATOR+M and GLADIATOR-D+M can be modified by modifying fleece.cpp file situated at quarch/src/fleece.cpp. Similarly, experiments can be modified using leakage/src/experiments.cpp.

A.8 Methodology

Submission, reviewing and badging methodology:

- https://www.acm.org/publications/policies/artifact-review-and-badging-current
- https://cTuning.org/ae

B Boolean Patterns for Leakage Detection

To enable early identification of leaked qubits, we derive Boolean patterns based on detector outputs in various code configurations. These patterns are extracted from simulated leakage events and represent recurring structures in the detector graph that correlate with leakage. Below, we describe the methodology and present patterns for BPC codes, standard color codes, and the color code augmented with GLADIATOR-D.

B.1 Pattern Extraction Methodology

To derive accurate Boolean expressions for leakage patterns, we begin by identifying syndrome configurations associated with known leakage events using annotated simulations. Since these patterns vary in length depending on spatial and temporal detector context, we normalize them using an index tagging system that encodes shorter patterns with unique binary prefixes. For example, 6-bit patterns are padded to 7 bits with a leading `0`, 5-bit patterns with `10`, and so on, following a structured prefix scheme. This allows uniform processing while preserving distinctiveness between patterns of different lengths.

Once normalized, we construct a truth table over all possible bit combinations, marking those corresponding to known leakage signatures as true and the rest as false. Each true entry is then converted into a minterm, and symbolic Boolean minimization is applied using SymPy's logic simplification engine. This produces compact, disjunctive normal form (DNF) expressions that generalize well across leakage events while maintaining logical equivalence. The final expressions are validated against the full truth table to ensure correctness and are integrated into our leakage speculation engine for efficient runtime detection.

B.2 Balanced Product Cyclic Code Patterns

The following Boolean expression encodes leakage patterns identified for the BPC code, as described in [22].

$$
\begin{aligned}
&(x_0 \wedge x_1 \wedge x_2 \wedge x_3 \wedge \neg x_5) \vee (x_0 \wedge x_1 \wedge x_2 \wedge x_4 \wedge \neg x_5) \\
&\vee (x_0 \wedge x_1 \wedge x_2 \wedge x_5 \wedge \neg x_6) \vee (x_0 \wedge x_1 \wedge x_3 \wedge x_4 \wedge \neg x_5) \\
&\vee (x_0 \wedge x_1 \wedge x_3 \wedge x_5 \wedge \neg x_6) \vee (x_0 \wedge x_1 \wedge x_4 \wedge x_5 \wedge \neg x_6) \\
&\vee (x_0 \wedge x_2 \wedge x_3 \wedge x_4 \wedge \neg x_5) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge \neg x_5) \\
&\vee (x_0 \wedge x_2 \wedge x_3 \wedge x_5 \wedge \neg x_4 \wedge \neg x_6) \vee (x_0 \wedge x_2 \wedge x_4 \wedge x_5 \wedge \neg x_3 \wedge \neg x_6) \\
&\vee (x_0 \wedge x_3 \wedge x_4 \wedge x_5 \wedge \neg x_2 \wedge \neg x_6) \vee (x_1 \wedge x_2 \wedge x_3 \wedge x_5 \wedge \neg x_4 \wedge \neg x_6) \\
&\vee (x_1 \wedge x_2 \wedge x_4 \wedge x_5 \wedge \neg x_3 \wedge \neg x_6) \vee (x_1 \wedge x_3 \wedge x_4 \wedge x_5 \wedge \neg x_2 \wedge \neg x_6)
\end{aligned}
$$

B.3 Color Code Patterns

Leakage detection patterns for the standard color code are more compact and are given below:

$$(x_0 \wedge x_1 \wedge \neg x_2 \wedge \neg x_3) \vee (x_0 \wedge x_2 \wedge \neg x_1 \wedge \neg x_3) \vee (x_1 \wedge x_2 \wedge \neg x_0 \wedge \neg x_3)$$

B.4 Color Code with GLADIATOR-D

The following patterns were identified in the color code augmented with our speculation method (GLADIATOR-D), which captures more temporally distributed signatures of leakage:

$$
\begin{aligned}
&(x_0 \wedge x_2 \wedge x_5 \wedge \neg x_1 \wedge \neg x_3 \wedge \neg x_6) \vee (x_0 \wedge x_2 \wedge x_5 \wedge \neg x_1 \wedge \neg x_4 \wedge \neg x_6) \\
&\vee (x_1 \wedge x_2 \wedge x_5 \wedge \neg x_0 \wedge \neg x_3 \wedge \neg x_6) \vee (x_1 \wedge x_2 \wedge x_5 \wedge \neg x_0 \wedge \neg x_4 \wedge \neg x_6) \\
&\vee (x_0 \wedge x_1 \wedge x_3 \wedge x_4 \wedge \neg x_2 \wedge \neg x_5 \wedge \neg x_6) \\
&\vee (x_0 \wedge x_1 \wedge x_3 \wedge x_5 \wedge \neg x_2 \wedge \neg x_4 \wedge \neg x_6) \\
&\vee (x_0 \wedge x_1 \wedge x_4 \wedge x_5 \wedge \neg x_2 \wedge \neg x_3 \wedge \neg x_6) \\
&\vee (x_0 \wedge x_2 \wedge x_3 \wedge x_4 \wedge \neg x_1 \wedge \neg x_5 \wedge \neg x_6) \\
&\vee (x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge \neg x_0 \wedge \neg x_5 \wedge \neg x_6)
\end{aligned}
$$