ARTERY: Fast Quantum Feedback using Branch Prediction

Wuwei Tian
Zhejiang University
College of Computer Science, ACES Laboratory
Hangzhou, Zhejiang, China
sonder@zju.edu.cn

Liqiang Lu*
Zhejiang University
College of Computer Science, ACES Laboratory
Hangzhou, Zhejiang, China
liqianglu@zju.edu.cn

Siwei Tan
Zhejiang University
School of Software Technology, ACES Laboratory
Hangzhou, Zhejiang, China
siweitan@zju.edu.cn

Yun Liang
Peking University
School of Integrated Circuit
Beijing, China
ericlyun@pku.edu.cn

Tingting Li
Zhejiang University
College of Computer Science, ACES Laboratory
Hangzhou, Zhejiang, China
litt2020@zju.edu.cn

Kaiwen Zhou
Zhejiang University
College of Computer Science, ACES Laboratory
Hangzhou, Zhejiang, China
kaiwenzhou@zju.edu.cn

Xinghui Jia
Zhejiang University
College of Computer Science, ACES Laboratory
Hangzhou, Zhejiang, China
jxhhhh@zju.edu.cn

Jianwei Yin
Zhejiang University
College of Computer Science, ACES Laboratory
Hangzhou, Zhejiang, China
zjuyjw@zju.edu.cn

Abstract

Quantum feedback makes the execution of dynamic quantum circuits possible and is widely used in quantum algorithms. However, due to the inherent computation and transmission cost, the latency of the quantum feedback becomes a considerable burden on the current quantum algorithm. The dynamic property of the feedback also makes the gates blocked until the feedback is finished. In this paper, we propose ARTERY, which uses branch prediction to support instruction pre-execution and speed up the feedback. ARTERY integrates historical statistics of branches and a real-time readout pulse analysis to predict the branch. With this idea, we build up a reconciled branch predictor that concatenates the historical statistics of branches and a real-time branch circuit speculation obtained from the readout-pulse trajectory predictor. We further explore the implementation of peripheral hardware for feedback, including a scalable inter-FPGA connection via the backplane, a feedback trigger mechanism for dynamic instruction timing, and an adaptive pulse sampling technique to maximize the hardware bandwidth. ARTERY accelerates quantum feedback process by 2.07× compared to the state-of-the-art method, with over 90% prediction accuracy, achieving 1.24× fidelity improvement.

*Corresponding Author

ISCA '25, Tokyo, Japan

ACM ISBN 979-8-4007-1261-6/25/06
https://doi.org/10.1145/3695053.3731086

CCS Concepts

• **Hardware → Quantum technologies**; • **Computer systems organization → Real-time system architecture**.

Keywords

Quantum Feedback, Branch Prediction

ACM Reference Format:
Wuwei Tian, Liqiang Lu, Siwei Tan, Yun Liang, Tingting Li, Kaiwen Zhou, Xinghui Jia, and Jianwei Yin. 2025. ARTERY: Fast Quantum Feedback using Branch Prediction. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture (ISCA '25), June 21–25, 2025, Tokyo, Japan.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3695053.3731086

1 Introduction

Quantum feedback is necessary for many quantum algorithms, such as fast reset [46, 74], quantum state teleportation [50], and quantum error correction [10]. These algorithms utilize the ability of feedback to dynamically change the operations in the quantum program based on the mid-measurement result, which improves the flexibility of the program [65]. Feedback is a key component in error correction, usually taking more than 70% of time for readout and qubit repair [28, 42, 47, 52]. Since qubits rely on correction to mitigate error in the current noisy hardware [1, 11, 12, 19, 43, 59, 64, 66], feedback will frequently occur on in future quantum applications.

However, similar to conditional judgment in classical CPU feedback, quantum feedback entails an extremely high computational overhead in quantum programs. Moreover, since gates after the feedback are non-deterministic, the computation is blocked at the feedback, which further increases the latency. Specifically, the feedback is implemented via a readout on the quantum processor, followed by classical processing on the FPGA boards that determines

the subsequent quantum operations. The readout and reset with classical processing require 500 ns and 160 ns in Google's quantum error correction experiment [42], individually. The total latency is 26.4 times longer than a single gate operation. During this long feedback period, qubits are exposed to various sources of error, leading to a high error rate. For example, on Sycamore hardware, feedback incurs an error rate of more than 2%. Such an error rate is far above the threshold for quantum error correction, resulting in a significant correction overhead.

Hardware-level optimization on quantum processors or peripheral hardware has been applied to speed up the feedback [9, 14, 16, 48, 61, 67, 68, 70, 71, 73]. Nevertheless, they are limited by hardware-imposed lower bounds in the readout and classical processing. For readout, the latency is constrained by the required qubit lifetime, which, if reduced, would lower overall computational accuracy [15, 18, 22, 67]. For example, Walter et al. [67] achieve an 88 ns readout latency by optimizing the chip fabrication. Further reducing the readout latency requires an increase in the coupling strength between the readout resonator and the qubit, which exposes qubits to the external environment and therefore reduces the qubit lifetime (i.e., T1) [21]. For classical processing, prior works [7, 8, 62, 71] accelerate the state classification and pulse preparation on the FPGA through parallelism [71] and caching techniques [7, 8, 62]. However, the processing unit (i.e., state classifier, ADC, and DAC) in the FPGA has an inherent latency due to clock frequency and essential calculations, resulting in a minimum overall latency of 150 ns [20].

Although quantum feedback faces the latency wall, a key technique is to exploit branch prediction (BP), an essential component of CPUs, for quantum feedback. A typical BP mechanism predicts the most likely branch and pre-execute instructions under this branch. It can significantly reduce pipeline stalls for successful speculations. However, for unsuccessful predictions, the CPU incurs additional instructions to recover the computation. The effectiveness of BP is largely determined by the accuracy of the prediction. However, the classical BP methodologies [6, 35, 49, 53] can not be directly applied to quantum workloads for two reasons, necessitating a new quantum BP design. First, qubits can stay in the superposition state, exhibiting higher randomness in the readout, while classical BP is designed for deterministic outcomes. For example, when measuring a qubit state with a 50% probability of $|0\rangle$ and a 50% probability of $|1\rangle$, adopting deterministic speculation provides no benefit. Second, the quantum readout is a long, continuous process, providing intermediate readout results for prediction, while classical estimations are discrete and cannot utilize this information.

In this paper, we propose ARTERY, a BP design for quantum feedback, including a fast quantum speculation methodology and a physical design that optimizes the synchronization and scalability of the classical feedback hardware. We first introduce a BP mechanism for quantum feedback, which consists of rules for identifying the pre-executable instructions and the recovery strategy for incorrect predictions. Then, we present our speculation methodology, which combines the historical results of prior shots and a dynamic trajectory classifier to estimate the probability of choosing each branch in the early stages of feedback. When the prediction probability reaches a confidence threshold, the program pre-executes gates to eliminate the blocking. Our prediction achieves an accuracy of more than 90%, benchmarked by current quantum algorithms.

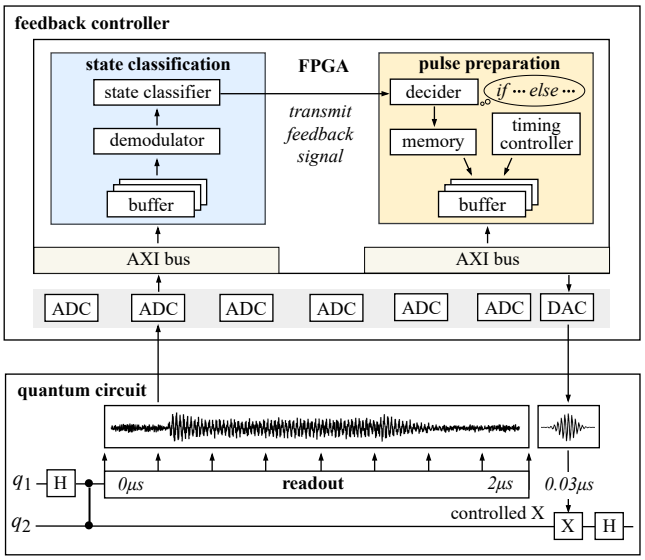


Figure 1: Basic architecture of quantum feedback.

In the physical implementation, we design a hardware architecture for fast feedback based on ARTERY, including hardware units for state classification and pulse preparation. Three optimizations are introduced to improve the synchronization and scalability. First, we develop a hardware interconnection scheme employing hierarchical communication, which shortens the critical path for feedback signal transmission. Second, to reduce the idle time for unit communication, we design a timing control scheme with feedback triggers to orchestrate the units. Third, we implement an on-chip pulse sampling method to alleviate bandwidth pressure and maximize the integration density of DAC channels on a single FPGA. The contributions of this paper are summarized as follows:

- We propose a fast quantum feedback mechanism using branch prediction, which addresses the inherent latency limitations of current quantum hardware, thereby benefiting various quantum algorithms, such as quantum error correction and fast reset.
- We propose the first branch prediction method specifically for quantum programs. Using historical results and dynamic qubit trajectories, achieving over 90% accuracy.
- We propose a set of hardware-level optimization methods, including an adaptive timing control method, a hardware interconnection scheme, and adaptive pulse sampling to enhance the synchronization and scalability of the feedback system.

Experiments suggest that ARTERY achieves 2.07× acceleration in feedback compared to the state-of-the-art feedback method [20], driving the average latency from 2.15 μs to 1.04 μs, achieving 1.24× fidelity improvement.

2 Background

2.1 Quantum Feedback Architecture

Figure 1 shows the typical quantum feedback on quantum devices with two stages. First, a qubit is read. Second, the readout information is sent to a classical feedback controller, which classifies the qubit state and determines the branch 0 or 1 of the quantum

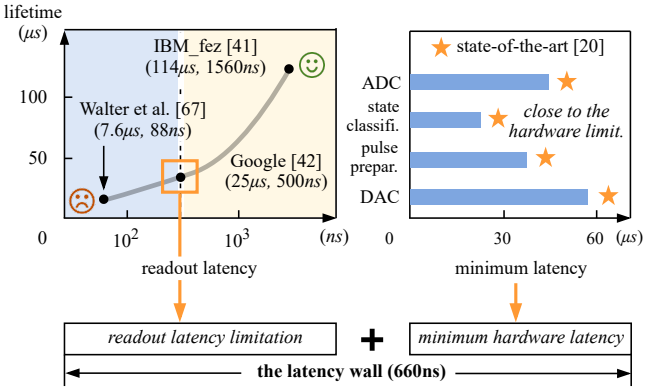


Figure 2: Latency breakdown of quantum feedback.

program based on the qubit state. The physical implementation of this paper is based on the superconducting quantum hardware. However, the mechanism of ARTERY can be generalized to other physical implementations, such as the neutral atom [69], and the trapped-ion hardware [5, 30, 37].

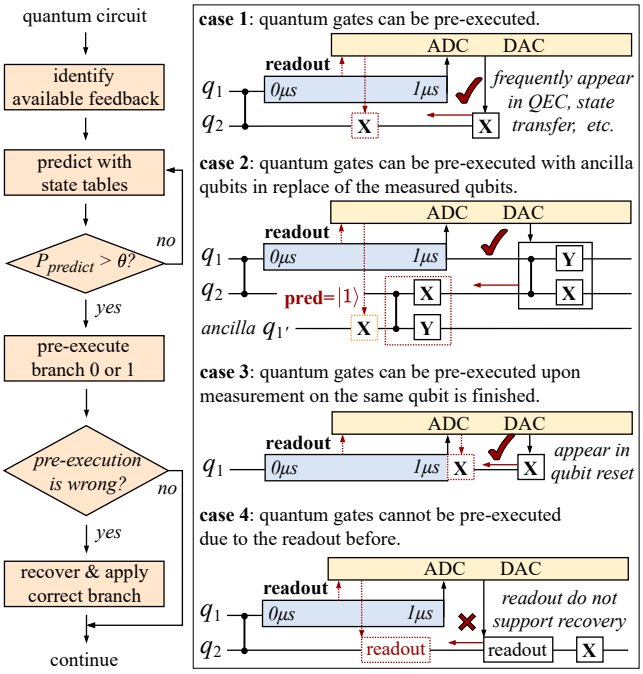
Figure 1 presents the architecture of the classical feedback controller to perform branching. It includes ADC modules to capture the pulse form qubits as readout information. The information is processed by an FPGA, which contains two key units. The state classification unit demodulates the pulse and classifies the qubit state. The pulse preparation reads the pulse data of the chosen branch from the memory. The pulses pass through a timing controller for synchronization and to the buffer, which are then sent by DAC to the quantum processor.

2.2 Breakdown of Quantum Feedback

Figure 2 presents the breakdown of latency in the feedback, suggesting a 660 ns latency wall according to current quantum hardware. Specifically, for the quantum processor, reducing the readout latency requires increasing the coupling strength between the readout resonator and the qubit, which also decreases the qubit lifetime (i.e., T1). The left part of Figure 2 presents the relation between the lifetime and the readout latency on different quantum processor designs [41, 42, 67]. For example, Walter et al. [67] achieves the smallest readout latency of 88 ns, while it also has the smallest lifetime of 7.6 μs. To ensure a useful lifetime, the minimum readout latency is limited to 500 ns by Google [42]. On the other hand, the current feedback controller involves ADC processing, state classification, pulse preparation, and DAC processing, as shown in the right part of Figure 2. Each has a minimum latency of 44 ns, 24 ns, 36 ns, and 56 ns, respectively. This overall leads to the latency wall of 660 ns (500 ns of readout + 160 ns of feedback hardware), which can hardly be optimized by hardware-level optimization. Moreover, under the current sequential feedback mechanism, we can observe that the state-of-the-art feedback controller [20] is close to the minimum latency, showing a small optimization space.

3 Overview

ARTERY speeds up the feedback by pre-executing the quantum gates with a new branch prediction algorithm. Figure 3 (a) presents



(a) ARTERY workflow. (b) Example of quantum gates that can be or cannot be pre-executed.

Figure 3: Overview of ARTERY.

the workflow of ARTERY. Given a quantum circuit, ARTERY first identifies available feedback, allowing instruction pre-execution. It then proceeds to program execution. During the readout of the feedback, ARTERY iteratively analyzes the probabilities of branches based on the historical branches in prior shots. When the probability of a branch exceeds a threshold θ, ARTERY pre-executes the quantum gates of these branches. When the readout is finished, ARTERY knows the correctness of the prediction. If the prediction is incorrect, a recovery is required. Specifically, since quantum circuits are reversible, ARTERY applies the reversed quantum gates to cancel the pre-executed gates and then execute the correct branch.

According to the quantum mechanism, all the quantum gates can be pre-executed. We summarize gate pre-execution in the following situations, shown in Figure 3 (b):

(1) In case 1, before the X gate on q_2, there are no other gates or readouts. And the X gate, therefore, can be pre-executed. This situation frequently appears in applying correction gates on the data qubit in feedback-based quantum error correction such as magic state injection (construction of logic T gate) [17, 23]. It also appears in quantum state transfer such as remote entanglement gates construction [4].

(2) In case 2, the feedback gates involve two-qubit gates that rely on the operation of the read qubit, q_1. It cannot be simply pre-executed on q_1 as the read qubit is occupied by the readout operation. However, an ancillary qubit q_1' will help. After readout, q_1 will collapse to classical states, which can be pre-prepared on q_1', then the rest of the gates operated on q_1 can be pre-executed on q_1'. By the way, q_1 can be recycled after

the readout operation is completed, thereby minimizing qubit resource waste to the greatest extent.

(3) In case 3, for scenarios where feedback must operate on the read qubit, such as qubit reset [74], we must wait for the completion of the reading process before applying the feedback gate. In contrast to traditional methods, we can employ prediction techniques to apply the feedback gate at the end of the readout operation, thereby eliminating hardware processing latency, which exceeds 100 ns.

(4) In case 4, if the feedback operation is readouts on another qubit q_2, we cannot apply pre-execution as the readout is not reversible. Once a prediction error occurs, it is unrealistic to recover the state of q_2, as q_2 has already collapsed to a classical state due to the readout operation.

Gate pre-execution essentially involves altering the temporal ordering of operations within the directed acyclic graph (DAG) of the quantum circuit. These four cases essentially represent a DAG constraint analysis for quantum feedback pre-execution. For example, in case 1, the feedback operation on Q2 is independent and unconstrained, and the error state recovery process under the assumption of dynamic circuit execution is also unconstrained, allowing execution in advance. Cases 2 and 3, however, are subject to the same constraint, i.e., operations cannot be performed on a busy qubit, and thus quantum gates must be pre-executed on auxiliary qubits to transition to a valid state. Cases 1-3 encompass almost all quantum feedback situations, allowing branch prediction to accelerate the majority of feedback applications. Case 4 is constrained by the "irreversibility of readout" logic, making it impossible to recover the state prior to prediction, and therefore, quantum gates cannot be executed in advance.

Real-time feedback and Pauli Frame error correction are orthogonal methods in QEC. In the Pauli Frame scheme [45, 57], the syndrome can benefit from latency reduction through prediction-based active feedback (Figure 3 (b), case 3). Some studies suggest that real-time feedback can map data qubits to low-energy state to prevent error accumulation and error propagation [2, 38], allowing logical T-gates to be implemented [17, 40]. In this manner, data qubit can also benefit from pre-correction (Figure 3 (b), case 1). This case can also apply to the pre-execution of branching circuits for modern long-distance practical circuits such as quantum random walk (QRW) [51] and quantum state teleportation (QST) [4, 55] performed on physical qubits.

Although the readout in the feedback causes quantum collapse to all qubits, when the readout qubits are entangled with other qubits, the mathematical foundation of the gate pre-execution lies in the fact that the Hamiltonian of the readout does not operate on the pre-executed qubits. When the branch is determined, the operator of the readout and the subsequent quantum gates commute. In other words, the order of the readout and branch does not change the final result. We put specific proofs of it in the Appendix, to prove that the pre-execution operation is equivalent to the feedback operation.

4 Branch Prediction Methodology

As a motivational example, Figure 4 illustrates the readout outcome statistics of the prior and posterior shots during the execution of a quantum feedback program, using quantum random walk (QRW) as

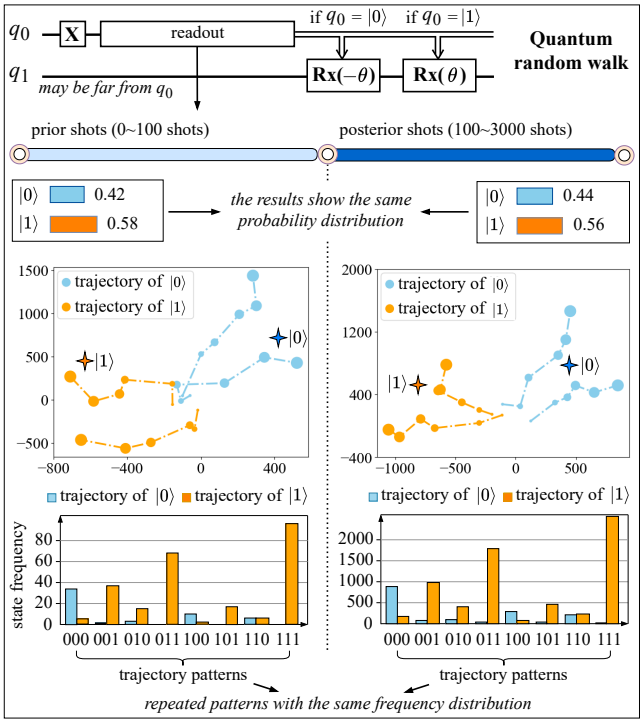
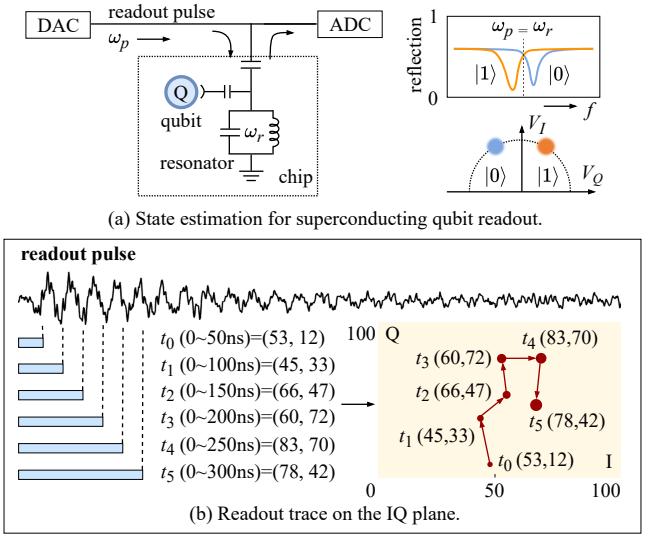


Figure 4: Motivational example for ARTERY.

an example. For any given feedback program, the readout distributions of the prior and posterior shots follow a very similar pattern, e.g., (0.42, 0.58), (0.44, 0.56). The figure also displays example readout trajectories for prior and posterior shots. Larger points indicate later times. The data corresponds to a readout pulse of 2000 ns, with IQ coordinates calculated every 400 ns. The figure shows that the trajectory for $|0\rangle$ generally trends toward the $|0\rangle$ center of the fixed coordinate for most of the time period, while the trajectory for $|1\rangle$ follows a similar pattern. The figure presents statistical *trajectory states*, revealing repeated patterns in both prior and posterior shots, with similar frequency distributions. This observation allows us to leverage the frequency of state occurrences to predict the readout results of subsequent shots, using the following BP method.

ARTERY adopts a branch prediction method inspired by the classical method. As shown in Figure 6 (a), A classical BP method [54] uses historical branches to predict the possible branch under the current state, which is stored in a state table consisting of $< state, branch >$ pairs. *branch* of each state in the state table is the most frequent branch under this state, which is dynamically updated during the CPU execution. A feature based on the temporary dependency of the branches in the CPU is characterized as a current state, which consists of the recent k branches of prior feedback. The prediction is performed as a table matching to find the most possible direction. For example, when the recent 2 directions are "1, 1", the state is "11". For the table with pair $< 11, 1 >$, the prediction is 1.

However, the branch of the quantum feedback shows higher randomness due to the quantum superposition state. Besides, BP on the classical CPU assumes that the branches are temporarily



(a) State estimation for superconducting qubit readout.

(b) Readout trace on the IQ plane.

Figure 5: Mechanism of qubit state classification.

dependent; we can, therefore, use branches of different programs to predict the current branch, but the branches of different quantum programs are independent. ARTERY adopts a combination model to predict the branch. Specifically, a quantum program usually has multiple shots. Besides, a readout has two possible outcomes: 0 or 1. We only need to predict the probability of reading 1, $P_{predict\_1}$, as $P_{predict\_0} = 1 - P_{predict\_1}$ ARTERY uses two key features to model the probability for each shot:

(1) the historical branch distribution of this feedback, which includes the statistical probability of reading 1 ($P_{history\_1}$);
(2) the probability of reading 1 (i.e., P_{read_1}) at an intermediate moment of the readout process in the current shot.

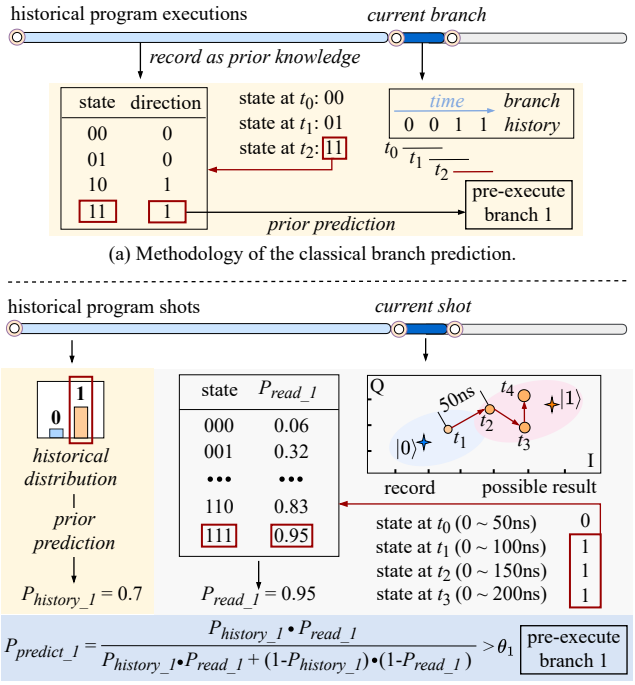
Since readout in the quantum system is a continuous process, we can estimate P_{read_1} at the intermediate time point of this process. As shown in Figure 5 (a), on the superconducting quantum hardware, the readout is implemented with a readout resonator coupled to the qubit [22, 24]. The state of each qubit is measured by sending the readout pulses to the readout resonator and identifying the frequency shift of the pulse, called *dispersive shift*.

A portion of the readout pulse can differentiate frequency shifts via I and Q values, with longer pulses offering higher accuracy. At time point t_i, the readout controller captures a segment of pulse, with complex amplitudes $pulse = [a_1, a_2, \cdots, a_L]$, where L is a specified window length. The I and Q values are calculated following the demodulation equation:

$$I = \frac{1}{L+1}\sum_{i=1}^{L}(a_i.real \cdot cos(\omega i) + a_i.imag \cdot sin(\omega i))$$

$$Q = \frac{1}{L+1}\sum_{i=1}^{L}(a_i.imag \cdot cos(\omega i) - a_i.real \cdot sin(\omega i))$$

where $a_i.real$ and $a_i.imag$ are the real part and the imaginary part of the amplitude a_i, respectively. As presented in Figure 5 (a), I and Q values of different states (i.e., 0 or 1) are in different clusters in the IQ plane, which is used to classify the readout outcome. For



(a) Methodology of the classical branch prediction.

(b) Methodology of ARTERY inspired by classical branch prediction.

Figure 6: Comparison of classical and quantum branch prediction methodologies.

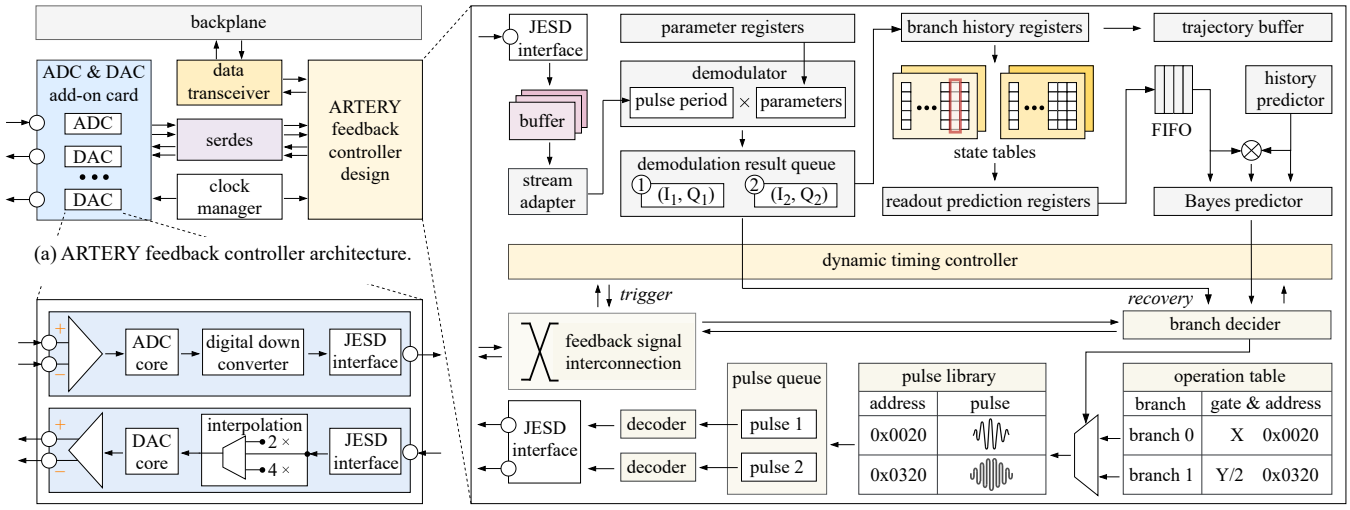
example, Figure 5 (b) presents the trajectory of I and Q values at time points t_0 to t_5.

Since I and Q values at different time points form a trajectory, the estimation is performed by comparing this trajectory with a pre-generated table. Specifically, at time point t_i, we demodulate the captured readout pulse, obtaining the IQ values. Then, by calculating the distances between IQ values and the center of 0 or 1, we can identify the most probable state at this time point. However, there are noises during the readout, leading to data jitter at the time points. We record the most probable states of k recent time points as a trajectory and compare this trajectory to a $< trajectory, P_{read\_1} >$ table, which records P_{read_1} under different trajectories. k is a user-defined parameter determined based on the granularity of the estimation. In addition, the $< states, P_{read\_1} >$ table is pre-generated when the quantum hardware is initialized, and the probability in the state table remains unchanged during the execution of the feedback program. However, we dynamically update the probabilities among different quantum feedback programs based on previous execution results to enhance the accuracy of predictions.

We combine the historical probability and the probability based on the intermediate readout pulse by a Bayesian model to estimate the overall probability of the:

$$P_{predict\_1} = \frac{P_{history\_1} \cdot P_{read\_1}}{P_{history\_1 \cdot read\_1} + (1 - P_{history\_1}) \cdot (1 - P_{read\_1})}$$

For example, as shown in Figure 6, the historical probability $P_{history\_1}$ of using branch 1 is 0.7. The prediction iteratively analyzes the readout pulse and records the trajectory of possible readout results. The state is regarded as the recent 3 results, 111.



(a) ARTERY feedback controller architecture.

(b) Modules for pulse generation and capture.

(c) Processing units of feedback controller.

Figure 7: Hardware design of ARTERY.

According to the $< states, P_{read\_1} >$ table, the probability of reading 1 in the current shot is $P_{read\_1} = 0.95$. Combining $P_{history\_1}$ and P_{read_1}, the overall of predicted probability of using branch 1 is $P_{predict\_1} = \frac{0.7 \cdot 0.95}{0.7 \cdot P_{read\_1} + (1-0.7) \cdot (1-0.95)} = 0.97$. Pre-defined probability thresholds θ_0 and θ_1 are employed to decide whether to take gate pre-execution. If $P_{predict\_0} < \theta_0$, the feedback will take branch 0, otherwise branch 1 when $P_{predict\_1} > \theta_1$.

The ARTERY algorithm is fast in implementation and accurate in the estimation. Clearly, in terms of implementation, updating the historical outcome distribution is performed after each prediction, leading to no latency. In P_{read_1} estimation, computing the distance and the table matching has $\mathcal{O}(1)$ complexity. In terms of accuracy, the prediction uses both the historical outcomes and the current readout state. Therefore, for feedback with uniform qubit state (50% 0 and 50% 1), the probability can be estimated based on the current readout state. On the contrary, when the readout has a high error rate, the accurate estimation is achieved by the historical outcomes.

5 ARTERY Implementation

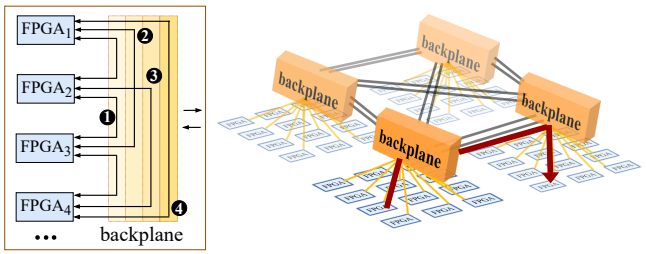
5.1 Hardware Design

Figure 7 (a) provides an overview of ARTERY's feedback controller architecture. In this design, the clock management module generates the reference clock for the ARTERY system and provides synchronized clocks for the ADC and DAC modules. The ADC and DAC modules, responsible for pulse reception and transmission, are integrated into an add-on card. These chips are connected to the ARTERY design via serdes for high-speed pulse data transfer. The ARTERY hardware system connects to the backplate through the data transceiver, enabling the reception and transmission of feedback signals from other FPGAs.

As shown in Figure 7 (b), the input pulses are captured by the ADC core and the digital down converter to obtain low-frequency pulses. Additionally, the output pulses go through data interpolation and are then generated into analog pulses by the DAC core. The digital down converter and interpolation modules are quite time-consuming, accounting for most of the latency of ADC and DAC processing. The key processing units of the feedback controller are arranged as illustrated in Figure 7 (c). The top and the bottom of the figure represent state classification and pulse preparation modules, respectively, which are separated by the dynamic timing controller in the middle.

State classification. The input readout pulse is initially received, buffered, and then adjusted with a stream width adapter. Subsequently, two stream adapters handle the input stream. One generates the real part of the pulse, while the other handles the imaginary part. The adapter collects pulse data within the window length (t_1, t_2) and sends it to the demodulator. The pulse is then processed along with pre-stored parameters for demodulation. The demodulated results are pushed into a demodulation result queue with a depth of $\frac{pulse\ length}{window\ length}$, providing the IQ coordinates of the pulse. The branch history registers with a width of k records the preliminary classification of the quantum state at time t_2. Simultaneously, the register records are synchronized with the readout trajectory buffer to update the probabilities in the state table as soon as the readout is finished. Whenever the values in the branch history register are updated, P_{read} can be obtained via the state table, which is implemented using BRAM and occupies a max memory size of $2^{k-3}(k+16)$ Bytes, where k is the number of branch registers. The Bayesian prediction model consists of a multiplier and a FIFO for storing readout predictions, which outputs the $P_{predict}$ after three cycles. The prediction result is simultaneously sent to the timing controller for dynamic timing control and to the branch decider to determine whether it exceeds the pre-set thresholds θ_0 or θ_1.

Pulse Preparation. The branch decider is designed to analyze branch signals and prefetch quantum gate instructions from the operation table. The pulse library, implemented as a lookup table, stores pre-encoded pulses for branch circuits. Upon querying the library using an address, the pulses are retrieved and sent to the



(a) Wire connections of controllers in the backplane. (b) Wire connections across the backplanes.

Figure 8: Interconnected backplane to speed up the communication.

decoder. The decoded pulse is then transmitted to the DACs via the JESD interface, completing a full feedback cycle.

Additionally, for inter-FPGA feedback signals, transmission occurs through the predictor-interconnection-data transceiver-backplate pathway. This process also incorporates trigger signals issued by the dynamic timing controller to ensure synchronization of inter-FPGA communication. This design enables a scalable approach to branch prediction.

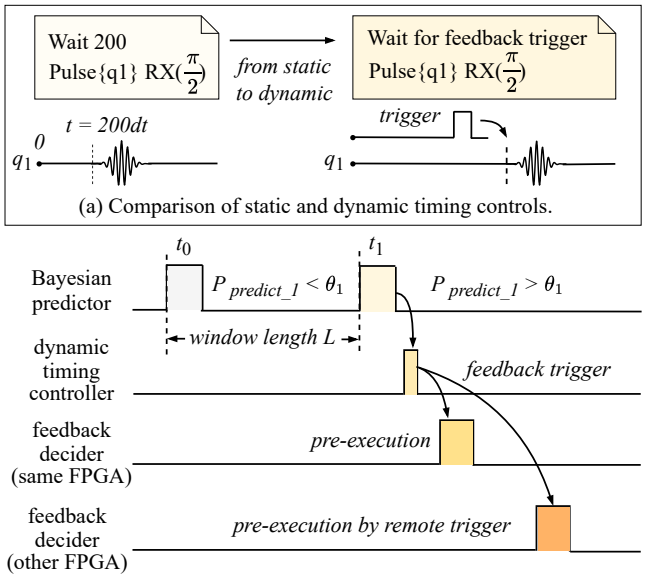
5.2 Scalable Controller Interconnection

Considering the scalability of quantum feedback, the primary issue lies in maintaining real-time feedback across the entire quantum chip, where the branch operations are controlled by nearby or far-away qubits. As shown in Figure 8 (a), the backplane architecture designed to interconnect multiple FPGA boards supports full connectivity, which features a layered structure, enabling high-speed point-to-point transmission between FPGAs with non-overlapping transmission paths within each layer. To ensure sufficient signal fan-out and enable feedback control between any qubit pair on-chip while minimizing overall transmission latency, the backplanes are connected as an efficient routing network for feedback signals.

As illustrated in Figure 8 (b), the transmission is organized into three levels. The first level handles feedback within the same FPGA, where signals are transmitted directly with minimal latency. The second level manages feedback between qubits under the same backplane, which offers a direct connection between FPGAs. The third level is activated to deal with feedback across backplanes to handle the longer-distance feedback. This distributed multi-level control is the optimal transmission architecture because it prioritizes lower-latency paths for most feedback operations, reserving higher-latency paths for only the most necessary communications.

5.3 Dynamic Timing

Conditional quantum execution. ARTERY employs Bayesian analysis based on branch prediction, where the timing of branch direction predictions is inherently uncertain. To support the dynamic execution of branch circuits, the conventional static timing scheme should be updated to conditional execution with dynamic timing. Designing an on-chip dynamic timing control scheme is required to modify timing based on predicted quantum states, allowing for low-latency execution without misalignment by adjusting execution windows to minimize idle time for the entire quantum system.



(b) Sequence diagram of the feedback trigger.

Figure 9: Feedback trigger mechanism to enable dynamic timing control.

As illustrated in Figure 9 (a), the execution of quantum instructions typically requires shifting the execution of branch circuit instructions from being constrained by a fixed time schedule to being governed by feedback triggers. For example, in static timing, an $RX(\frac{\pi}{2})$ pulse is executed at a fixed time point, such as $t = 200dt$. In contrast, under dynamic timing, instructions are conditionally executed: upon receiving a feedback trigger, the controller immediately issues the RX pulse for execution.

Feedback trigger. The Figure 9 (b) illustrates the working strategy of the feedback trigger. At time t_0, the Bayesian predictor generates a probability prediction $P_{\text{predict\_1}}$, which is below the threshold θ_1. As a result, the dynamic timing controller does not issue the feedback trigger, and the branch circuit remains inactive. After a time window of length L, at t_1, the Bayesian predictor updates its prediction, which now exceeds the threshold θ_1, meeting the conditions for executing the branch circuit. Consequently, the dynamic timing controller issues a feedback trigger. If the feedback process is executed on the same FPGA, the feedback decider initiates the branch circuit execution immediately upon receiving the feedback trigger. In the case of inter-FPGA feedback, the feedback trigger is transmitted through the backplane to the feedback decider on a different FPGA. Upon receiving the remote trigger, the other FPGA proceeds with the pre-execution of the branch circuit.

5.4 Adaptive Pulse Sampling

The signal transmission speed between DAC and ADC within an FPGA is significantly faster compared to the transmission between FPGAs via serdes (e.g. $4ns$ vs. $48ns$). However, the number of DACs and ADCs that can be connected with a single FPGA is limited due to Advanced Extensible Interface (AXI) bandwidth constraints. Quantum pulses usually contain a large number of '0' signals for idle operation and are easy to compress [32]. To address this, the

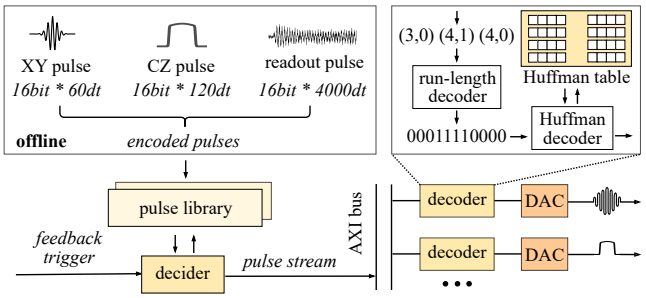


Figure 10: Workflow of the adaptive pulse sampling.

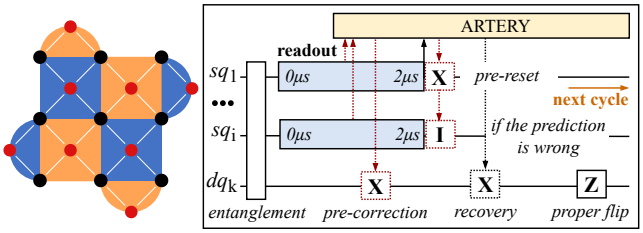
ARTERY design incorporates an adaptive sampling rate method that optimizes on-chip bandwidth usage. By dynamically adjusting the sampling rate based on compressed pulses, the design reduces the data transmission load, allowing the maximization of the integration density of DACs on each FPGA.

When deploying control pulses of the branch circuit, ARTERY generates high sampling rate analog-to-digital conversion data through pulse encoding. Before transmitting data to the DAC module, a decoding unit is designed to handle the encoded pulses from the pulse library. As shown in Figure 10, consider an example where the DAC has a resolution of 16 bits and a sampling rate of 2 GSPS (2G samples per second). For a basic gate set comprising RX, RY, RZ, and CZ, the pulse data required for circuit execution includes a 30 ns XY pulse, a 60 ns CZ pulse, and a 2 μs readout pulse. After encoding, these pulses are stored in the pulse library. When the decider receives a feedback trigger, it retrieves the required pre-encoded pulses from the pulse library. The pulse stream is then transmitted over the AXI bus and sent to the decoder for processing. The decoder utilizes a combination of run-length decoding and Huffman decoding. The pulses are first decoded using the run-length decoder, and then the original pulses are reconstructed using the Huffman table. Once decoded, the pulse is converted back to its original form and adjusted to the required DAC sampling rate. Finally, the decoded pulses are sent to the DAC modules for further processing and conversion into analog signals, ensuring more integration density of DACs on each FPGA under the same transmission bandwidth.

6 Evaluation

6.1 Experiment Setup

Platforms. We use a self-developed quantum processor with 18 Xmon-qubits, arranged in 3×6 grid topology. The qubit relaxation time (T1) ranges from 110μs to 140μs. The basis gates of the device are RX, RY, RZ, and CZ gates, where RZ gates are implemented as virtual gates [33]. The single-qubit gates, two-qubit gates, and readout are calibrated to reach the fidelities of 99.94%, 99.7%, and 99.0%, respectively. The calibrated parameter set of pulses to reach the best fidelity are recorded for pulse generation with ARTERY. 3 qubits share the same readout line using frequency-multiplexing. The duration of the readout pulse is 2μs for all of the qubits. We collect the readout pulses and branch results on this quantum device to perform evaluation. The dataset contains 4,000 readout pulses and the corresponding readout pulses. For each benchmark, we set 1, 000 sequences as training datasets for parameter training and the other for latency testing.



(a) Surface code scheme. (b) Error pre-correction and active reset in QEC circuit.

Figure 11: Example of QEC circuit.

Configurations of ARTERY feedback controller. We simulate the hardware controller with an FPGA carrying 16 DACs and 4 ADCs. We set Xilinx Zynq MPSOC `xczu15eg-ffvb1156-2-i` and `xczu9eg-ffvb1156-2-i` as the backend FPGAs connected with AD9164, AD9680 from Analog Devices as the backend DACs and ADCs, programmed by High-level Synthesis (HLS) and Verilog. The FPGA operates at 250MHz, and the interpolation of DAC is set to 2×, with a numerically controlled oscillator bypassed. We set the default window length in the demodulation to 30 ns for readout pulse demodulation. The default number of branch history registers is set to 6. The communication latency through serdes among FPGAs is 48 ns. The DAC sampling rate is set to 4 GSPS, with the ADC sampling rate fixed to 1 GSPS.

Software configurations. All programs are implemented with `Python` (3.9.13) and the `Numpy` package (1.23.1). We use `Qiskit` to simulate the noisy execution results. We also use `Pymatching` package to pre-generate a lookup table for decoding. All software experiments are performed on an AMD EPYC 2.25GHz 64-core CPU with 1TB of memory.

Baseline. We compare ARTERY against state-of-the-art feedback acceleration approaches, including QubiC [20] (implemented similarly on Google Sycamore [42]), HERQULES [31] (implemented with feedback, and the window length is set to 30 ns), Salathe et al. [48], and Reuer et al. [44].

Benchmark. The evaluation is performed on 6 algorithms, including the quantum error correction (QEC) [10] with surface code, quantum random walk (QRW) [51], quantum remote CNOT gate construction (RCNOT) [4], deterministic quantum teleportation [55], repeat-until-success based quantum neural network (RUS-QNN) [36] and active qubit reset [46, 74]. We use the QEC circuit in [10] with a code distance of 3, which is repeated in 1 to 30 cycles. For the QRW circuit, we evaluate a two-qubit circuit ranging from 1 to 25 steps. The physical distances of the RCNOT circuit and DQT circuit are both set to 1-6. The RUS-QNN circuit is set to repeat, ranging from 1 to 6 cycles. The reset operation is set to be performed on 1-25 qubits simultaneously. We also build a set of random benchmarking circuits, where we randomly add 25-150 gates before and after the feedback operation. Except for the QEC experiment, where the initial state is set to $|0\rangle$, $|+\rangle$ and $|1\rangle$, all other benchmarks adopt random initial states to ensure high accuracy.

6.2 Example: quantum error correction

Figure 11 (a) illustrates the surface code scheme used for validation (distance = 3), where the black circles represent data qubits and the red circles represent syndromes. Figure 11 (b) shows the execution of the QEC circuit integrated with ARTERY, where sq_i

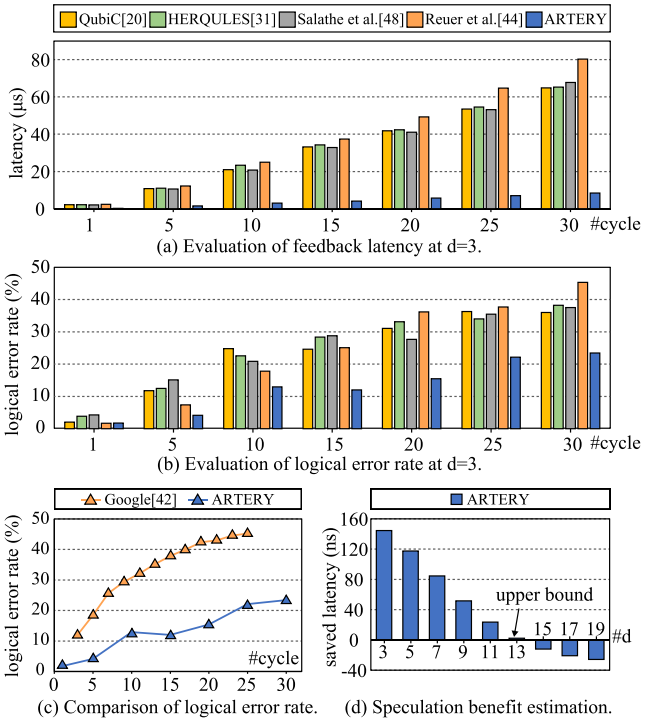


Figure 12: Evaluation of feedback latency and fidelity of QEC.

refers to the syndrome and dq_k refers to a data qubit. In each cycle, the syndrome's readout uses branch prediction to acquire readout results. Once ARTERY obtains all results from syndromes, they are decoded by a real-time decoder, and a pre-correction operation is executed. Next, ARTERY applies an X gate on the data qubit (dq_k) to flip the physical qubit before the syndrome readout is finished. As soon as the syndrome readout concludes, the pre-reset is performed based on the prior prediction, eliminating the need to wait for the hardware processing. If the previous prediction of the data qubit is correct, the program immediately proceeds to the next cycle. Otherwise, recovery is performed on the data qubit (via a second X gate on dq_k), followed by the proper operation (a final Z gate on dq_k).

Figure 12 (a) presents the evaluation of the FPGA feedback latency of data qubit correction in the QEC circuit. Overall, ARTERY achieves 4.80× speedup in feedback compared to the state-of-the-art method QubiC [20] (used in Google's hardware). As for active reset for syndromes, ARTERY achieves 1.08× faster than QubiC, driving 2.16 μs to 2.01 μs of latency. Considering that 2 μs of the 2.01 μs feedback latency are dedicated to the readout, this result demonstrates that the reset operation begins almost immediately after the readout is completed, thereby allowing more time for gate execution. Additionally, the latency of the QEC cycle depends on the syndrome readout and reset. ARTERY achieves a 1.06× acceleration in end-to-end latency for each cycle, reducing it from 2.45 μs to 2.31 μs. This acceleration is achieved with the readout latency of 2 μs; with faster readouts, the acceleration ratio could be even greater.

As for the simulation of logical errors, since packages like `Stim` do not support feedback operations and dynamic circuit configurations, we use `Qiskit` to construct and simulate a noisy d=3 surface code circuit (including T1, T2, single-qubit gates, two-qubit gates, and readout errors, with parameters consistent with Google [42]). We then construct a dynamic circuit using latency results obtained from ARTERY. Due to limitations in Qiskit's syntax for feedback operations, we replace the real-time decoder with a lookup table, retaining only the feedback operations. To better align with practical experimental requirements, we executed the QEC circuit with 500 repetitions. Figure 12 (b) shows the logical error rate of the circuit under different cycles, with dynamical decoupling [13] added on idle qubits. Compared to QubiC [20], ARTERY achieves a 1.86× reduction in logical error rate. This is due to the fast reset of the syndrome, which reduces the overall cycle latency, while the data qubits, being in a low-energy state due to pre-correction, reduce decoherence errors. It decreases physical qubit errors and therefore improves logical qubit error rates.

Figure 12 (c) compares the logical error rate result in the surface code with d=3 between the noise environment simulation of ARTERY and the state-of-the-art Google's real-world QEC demonstration experiment [42]. ARTERY achieves up to 2.02× improvement in the logical error rate, with a logical error rate of only 22.1% at cycle=25, which is lower than Google's 44.6%. Furthermore, compared to Google's demonstration, the error rate of ARTERY increases more gradually with the number of cycles.

By sampling from the existing syndrome prediction accuracy, we develop a latency error estimation model for larger code distances. In the estimation, any prediction error in a syndrome triggers branch recovery. We estimate the upper bound of ARTERY's benefit in latency reduction for larger code distances. Figure 12 (d) shows the syndrome feedback time saved by ARTERY at each cycle for different code distances. For d=13, ARTERY reaches its upper bound. Although ARTERY can predict the readout results in advance and operate alongside the decoder pipeline, the cost of recovery for wrong predictions limits its effectiveness. For circuits with d>13, the cost of prediction errors will overwhelm the benefits of pre-execution, and ARTERY does not contribute to latency reduction under the current prediction accuracy.

6.3 Evaluation of Feedback Latency

Table 1 presents the average feedback latency performance of ARTERY and other baselines. Overall, ARTERY achieves 2.07× acceleration in feedback latency compared to the state-of-the-art method (QubiC [20]). ARTERY drives the average feedback latency from 2.15 μs to 1.04 μs. The significant acceleration in feedback latency comes from branch prediction and circuit pre-execution, while other feedback methods still wait for readout and hardware processing.

As shown in Table 1, the feedback latency of different methods increases linearly with the number of feedback iterations. Therefore, the more feedback cycles included in the circuit, the more execution time ARTERY can save. Additionally, we observe that ARTERY feedback latency differs in algorithms. For instance, the average feedback latency in the RCNOT circuit is only 0.93 μs, while in the QRW circuit, it can be 1.23 μs. This significant difference is attributed to the varying historical readout probabilities. For example, in QEC, the probability distribution of historical readouts is highly imbalanced, with $P_{history\_1}$ being below 1%. Such a prior

Table 1: Evaluation of feedback latency (μs).

	QRW [51] (#step)				RCNOT [4] (#depth)				RUS - QNN [36] (#cycle)				DQT [55] (#distance)				reset [46, 74]	Random (#gate)			
	1	5	15	25	1	2	3	4	1	2	3	4	1	2	3	4	-	25	50	75	100
QubiC [20]	2.15	10.78	33.26	52.90	2.14	4.36	6.47	8.68	2.14	4.43	6.52	8.77	2.14	4.29	6.51	8.66	2.16	3.12	4.27	5.61	6.62
HERQULES [31]	2.17	10.95	33.96	55.13	2.16	4.39	6.55	8.71	2.17	4.44	6.53	8.69	2.21	4.29	6.54	8.67	2.16	3.16	4.39	5.72	6.69
Salathe et al. [48]	2.12	10.69	33.10	53.40	2.12	4.30	6.42	8.62	2.13	4.31	6.45	8.64	2.11	4.32	6.40	8.59	2.11	3.07	4.18	5.50	6.44
Reuer et al. [44]	2.43	12.15	37.21	64.20	2.40	4.91	7.37	9.86	2.37	4.98	7.36	9.97	2.38	4.86	7.42	9.81	2.38	3.39	4.58	6.01	7.10
ARTERY	**1.23**	**6.12**	**17.98**	**29.82**	**0.93**	**1.85**	**2.68**	**3.39**	**1.12**	**2.45**	**3.69**	**4.72**	**1.07**	**2.20**	**3.41**	**4.64**	**2.01**	**2.34**	**3.31**	**4.06**	**4.77**

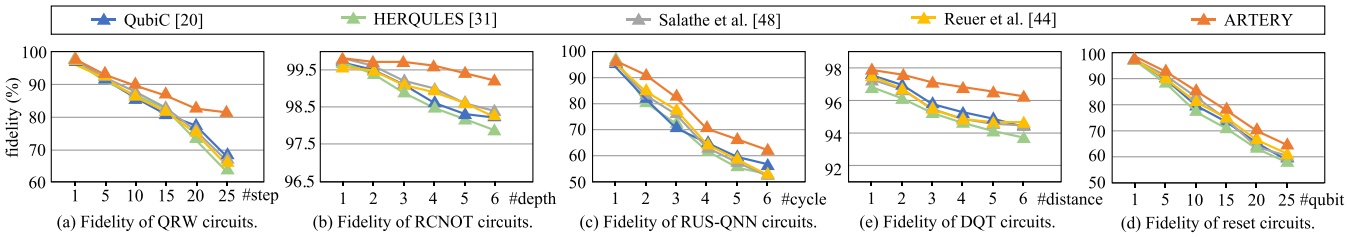


Figure 13: Evaluation of fidelity improvement.

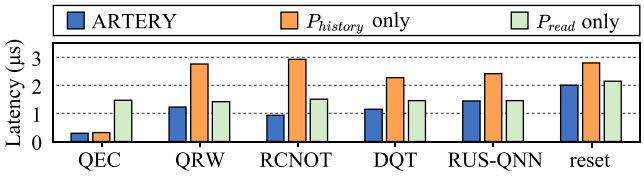


Figure 14: Ablation study of techniques.

probability allows the predictor to output $P_{predict\_1}$ exceeding the tolerance threshold θ_1 shortly after receiving the readout pulse, enabling early branch pre-execution. However, the prior probabilities in the QRW circuit are more random, requiring the analysis of a longer readout pulse before predicting and executing the branch circuit, resulting in a longer feedback latency.

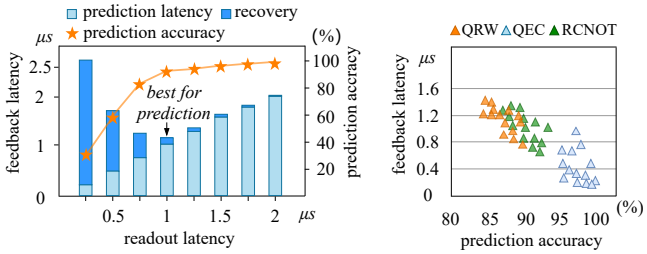
Figure 14 (a) summarizes the average feedback latency when relying solely on historical data versus readout pulse analysis. For the QEC algorithm, using only historical data achieves a prediction accuracy of 0.972 with an average latency of 0.386 μs. In contrast, other algorithms like DQT and RUS-QNN achieve prediction accuracies of only around 0.4 to 0.7, accompanied by significantly higher latencies. When relying solely on readout pulse analysis, the prediction accuracies across benchmarks exceed 0.9, but it still leads to 1.47× longer latency than that of ARTERY. Because of the pre-execution constraint (Figure 3 (b) case 3) for active reset, the feedback latency will never be less than 2 μs.

Figure 13 illustrates the fidelity improvement of QRW [51], RCNOT [4], RUS-QNN [36], DQT [55] and active reset [46, 74]. Including QEC fidelity in Figure 12 (b), overall, ARTERY achieves 1.24×, 1.22×, 1.19× and 1.29× fidelity improvement compared with the QubiC [20], HERQULES [31], Salathe et al. [48] and Reuer et al. [44]. Due to the shorter feedback latency, ARTERY is less affected by relaxation noise, and it becomes more beneficial when dealing with longer circuits. For example, when the QRW circuit reaches the 25th step, the fidelity of the target qubit remains at 80%, whereas the fidelity of other methods drops below 70%. In the case of the DQT algorithm, when a circuit with a distance of 6 (i.e., with 6 ancillary qubits is used for quantum state teleportation), ARTERY achieves a 1.52× improvement in fidelity compared to Salathe et al. [48]. Moreover, according to Figure 13 (b) and Figure 13 (d), as the distance of long-distance entanglement increases, ARTERY demonstrates even more significant improvements in fidelity for the target qubit than other methods. Considering ARTERY's shorter latency and higher fidelity, it proves to be a more efficient approach for facilitating long-distance quantum entanglement.

6.4 Evaluation of ARTERY Prediction

Figure 15 (a) depicts the prediction accuracy and feedback latency during the state classification for a depth=10 RCNOT circuit. As the readout latency increases, the branch prediction accuracy rises rapidly. At 0.75 μs of the readout pulse, the prediction accuracy reaches 82.7%, and at 1 μs, it increases to 90.6%. The rapid classification is attributed to the branch registers and state tables, which record the readout trajectory history and provide read probabilities, enabling the Bayesian model to refine the overall probability and quickly make accurate branch predictions iteratively. In the latter half of the readout pulse analysis, the prediction accuracy stabilizes above 95%, allowing for earlier detection of quantum states compared to a complete readout.

We randomly sample 14 branch prediction results for the feedback process of each benchmark. Figure 15 (b) illustrates the accuracy distribution of branch predictions for different benchmarks. As observed, the prediction accuracy for QEC is relatively high, predominantly around 97.0%, with an average latency of 0.382 μs. In contrast, QRW and RCNOT exhibit accuracy ranges primarily between 84.6% and 93.5%, with corresponding average latencies of 1.227 μs and 0.934 μs, respectively. The higher history probability in QEC enables faster predictions with greater accuracy. On the other hand, QRW and RCNOT rely more heavily on readout information compared to QEC, leading to longer feedback latency.



(a) Evaluation of prediction accuracy and latency on different readout latencies.

(b) Evaluation of prediction accuracy distribution.

Figure 15: Evaluation of branch prediction accuracy.

Table 2: Evaluation of the adaptive pulse sampling.

	Method	Raw pulse	Huffman	Run-length	Huffman & Run-length
Bandwidth (Gb/s)	QEC	64.0	27.5	11.9	**9.9**
	QRW	64.0	28.8	15.6	**13.1**
	RCNOT	64.0	26.4	14.0	**12.2**
#DAC / FPGA	QEC	4.0	9.0	21.0	**25.0**
	QRW	4.0	8.0	16.0	**19.0**
	RCNOT	4.0	9.0	18.0	**20.0**
Latency (ns)	QEC	-	18.9	12.3	**20.7**
	QRW	-	16.4	7.6	**13.5**
	RCNOT	-	17.2	12.5	**14.6**

6.5 Evaluation of Controller Implementation

Table 2 illustrates the bandwidth performance of the decoder when employing three different compression algorithms: Huffman encoding, run-length encoding, and a combined Huffman and run-length approach across three benchmarks. Compared to the original raw pulse, all three algorithms achieve significant improvements in on-chip pulse data transmission bandwidth. Specifically, ARTERY demonstrates an average 4.7× increase in pulse transmission bandwidth using the combined Huffman and run-length encoding, with the QEC benchmark achieving up to a 6.2× enhancement. Huffman encoding and run-length encoding individually yield average bandwidth optimizations of 4.1× and 2.6×, respectively. These improvements are attributed to the sparsity and compressibility of the raw pulses, even after software calibration techniques. While all three compression methods are effective across the benchmarks, their performance varies due to different gates during the execution. For instance, QEC achieves a higher compression ratio compared to QRW, likely because QEC circuits exhibit greater pulse repetitiveness despite a larger coding space. However, the temporal distribution of the pulse tends to be more disordered and hard for run-length encoding. Under a storage constraint of 1.4MB, Huffman demonstrates an advantage over run-length encoding. The combined Huffman and run-length approach further optimizes data transmission bandwidth by first applying Huffman encoding to the pulses, followed by run-length compression. This sequential strategy enables superior bandwidth performance.

Table 2 also shows the number of available DACs on a single FPGA after the bandwidth optimization. For the three benchmarks, the increase in the number of available DACs varies due to the differences in circuit compression ratios. Using the combined Huffman and run-length encoding approach, the number of available DACs can be increased by factors of 6.3, 4, and 5 for QEC, QRW, and RCNOT, respectively. Considering the general-purpose usage, it is optimal to minimize the number of DACs required for communication with the FPGA. Consequently, compared to the original quantum feedback implementation, where a single FPGA could support up to 4 DACs, the pulse decoder method increases this capacity to 16 DACs.

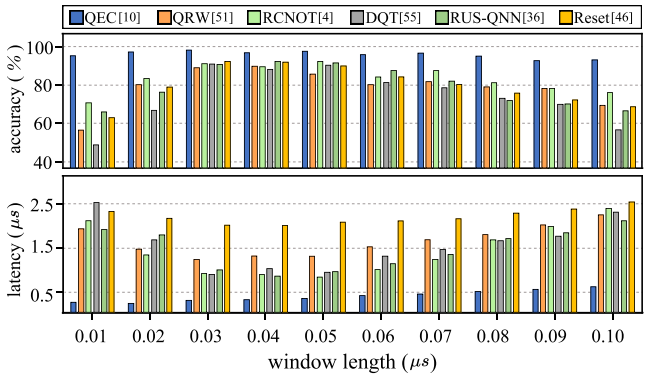


Figure 16: Evaluation of the window length in the demodulation.

The decoder itself introduces a decoding latency, adding to the feedback latency. However, the primary purpose of the decoder is to increase the number of DACs that can be supported by a single FPGA, replacing inter-FPGA transmission latency with shorter on-chip transmissions. This creates a latency trade-off. Table 2 also shows the average latency of different decoding algorithms on three benchmarks when feedback signals must be transmitted across FPGA boards. The combined Huffman and run-length approach achieves an average 1.66× reduction in overall latency. For algorithms like QEC and RCNOT, which involve inter-FPGA feedback communication, the acceleration provided by the decoder is more significant. Notably, for the RCNOT algorithm, which requires coordination among multiple FPGAs, the decoder method achieves a 2.3× acceleration compared to the original implementation. For longer transmission distances, the anticipated reduction in transmission latency would be even greater. In such cases, the decoder enhances bandwidth and adds decoding overhead to the feedback latency. Although there is a trade-off, decoders stay beneficial in superconducting quantum control systems composed of multiple FPGAs.

6.6 Evaluation of Parameter Setting

Figure 16 presents the prediction accuracy and feedback latency when setting different window lengths for segmented demodulation among 6 benchmarks. Both high prediction and lowest average feedback latency are achieved with the window length set to 0.03 μs. Our findings indicate that shorter window lengths fail to capture sufficient information, resulting in reduced accuracy. For example, DQT [55] algorithm shows the longest latency when the window length is set to 0.1 μs. Conversely, longer window lengths decrease the frequency of prediction updates, leading to increased latency,

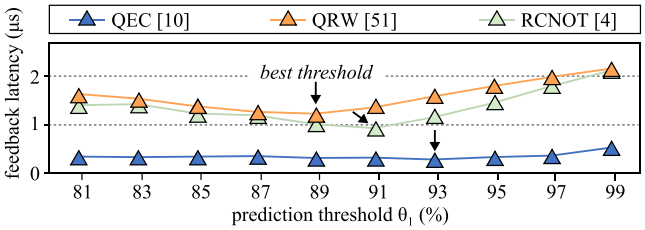


Figure 17: Evaluation of the probability threshold to determine the pre-execution.

especially for QEC [42] circuits. For instance, when the window length is set to 0.1 μs, the feedback latency increases to 2.1×.

Adjusting the tolerance threshold for each benchmark is recommended. Figure 17 shows an example of how to determine the experimental best threshold for RCNOT's branch prediction. We use the training pulse data to evaluate the feedback latency among different tolerance thresholds, obtaining the lowest overall feedback latency. Then, we set 91% to the tolerance threshold to evaluate the test pulse data.

7 Related Work

Current feedback optimization methods mainly focus on the acceleration of feedback calculations on FPGA. To reduce the feedback latency, Salathe et al. [48] use parallel calculations and pipelined processing to accelerate state classification. QubiC [20] implements a pulse table to speed up the pulse preparation and also uses fine-grained DAC optimization to minimize the feedback latency. Guo et al. [16] use parallel computing when demodulating readout IQ pulses. Yang et al. [71] optimizes the state classification by directly removing zero-value points during the mixing step, bypassing the time required for a moving average filter. Reuer et al. [44] generate a deep reinforcement agent for quantum feedback, reaching faster and more accurate readout for quantum feedback. Additionally, Tholén et al. [61] implement an event sequencer to pipeline signal generation, data acquisition, and feedback with precise timing. However, these methods typically yield only modest latency improvements and remain insufficient to overcome the performance bottleneck caused by the *latency wall*.

Prior works also focus on readout latency reduction. Walter et al. [67] minimizes readout latency to 88 ns by increasing the dispersive-interaction strength, choosing an optimal linewidth of the readout resonator. Heinsoo et al. [18] employ individual Purcell filters for each readout resonator to suppress off-resonant driving, minimizing readout latency to 80 ns. However, these methods face a shorter lifetime for qubits, making the quantum processor less practical.

HERQULES [31] and Benjamin et al. [26] employ readout trajectory analysis with FNN and DNN, respectively. However, in Section 4, We apply a new vectorization method that converts readout trajectories into states for feedback branch prediction. Combined with optimized readout analysis methods [60, 72] and pulse generation speedup [29, 62], ARTERY achieves higher accuracy and lower feedback latency. In pulse compression, Jongseok et al. and Dongmoon et al. [34, 39] only compress the pulse between the classical computer and FPGA, while our technique optimizes the communication between the components of FPGAs, which aims to increase the number of available DACs on each FPGA. ARTERY focuses on feedback on physical qubits rather than algorithms executed on logical qubits [3, 27, 56, 58] or non-feedback benchmarks [25, 63]. However, ARTERY benefits error correction itself by accelerating fast reset on syndrome and enabling data qubit pre-execution, providing benefits for more widely-used applications.

8 Conclusion

Readout and feedback latency are the primary sources of latency overhead in mid-circuit measurement processes required by quantum error correction and similar algorithms. Previous efforts, such as leveraging parallel computation or pre-stored pulses, have achieved only marginal latency reductions. However, due to the persistence of the *latency wall*, minimizing feedback time remains a significant challenge. To address the problem, we propose ARTERY to accelerate quantum program feedback via branch prediction. We introduce how to predict the readout result with historical readout results and IQ pulse trajectories. Finally, we propose a hardware design to provide end-to-end acceleration for quantum feedback.

Appendix: Proof of Gate Pre-execution

Consider a quantum system of two qubits (q_1, q_2). At t_0, the qubits are prepared to an initial state $|\psi(0)\rangle$. Originally, at specific times:

- During the period of t_1 to t_2, q_1 is under readout; The readout process can also be regarded as applying an entangling gate with the resonator, represented as $G_{q_1,r}$ operated on t_1.
- the branch circuit (represented as gate A_{q_2}), is executed on q_2 after t_2, based on the readout results of q_1.

let A_{q_2} is the branch circuit, which is applied based on the results of measurements m_1 on q_1. This feedback operation can be represented by

$$(A_{q_2})^{m_1} = \begin{cases} A_{q_2}, & \text{if } m_1 = 1 \\ I, & \text{if } m_1 = 0 \end{cases} \tag{1}$$

where I is the identity operation.

We want to prove that the pre-execution operation is equivalent to the feedback operation. That is to say,

$$\underbrace{(A_{q_2})^{m_1 \oplus m}}_{feedback\ on\ m} G_{q_1,r}(t_1,t_2)(A_{q_2})^m = (A_{q_2})^{m_1} G_{q_1,r}(t_1,t_2) \tag{2}$$

Where m is the predicted results of readout on q_1, and $\oplus$ represents the right multiplication by a measurement operator. Since $G_{q_1,r}(t_1,t_2)$ and A_{q_2} are applied on different qubits, they commute with each other. We have

$$G_{q_1,r}(t_1,t_2)(A_{q_2})^m = (A_{q_2})^m G_{q_1,r}(t_1,t_2) \tag{3}$$

Then, the pre-execution operation(the left of Equation (2)) becomes

$$(A_{q_2})^{m_1 \oplus m} G_{q_1,r}(t_1,t_2)(A_{q_2})^m \tag{4}$$

$$= (A_{q_2})^{m_1 \oplus m} (A_{q_2})^m G_{q_1,r}(t_1,t_2) \tag{5}$$

$$= (A_{q_2})^{m_1 \oplus m \oplus m} G_{q_1,r}(t_1,t_2) \tag{6}$$

Because $m_1 \oplus m \oplus m = m_1 \oplus (m \oplus m) = m_1 \oplus 0 = m_1$, we have

$$(A_{q_2})^{m_1 \oplus m} G_{q_1,r}(t_1,t_2)(A_{q_2})^m = (A_{q_2})^{m_1} G_{q_1,r}(t_1,t_2) \tag{7}$$

Thus, we have proved Equation (2).

This analysis shows that when two quantum gates are applied at different times and operated on the same qubit, they are regarded to be equivalent.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No.62472374) and the Zhejiang Provincial Natural Science Foundation of China under Grant (No.LR25F020002).

References

[1] Narges Alavisamani, Suhas Vittal, Ramin Ayanzadeh, Poulami Das, and Moinuddin Qureshi. 2024. Promatch: Extending the Reach of Real-Time Quantum Error Correction with Adaptive Predecoding. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*. 818–833.

[2] Christian Kraglund Andersen, Ants Remm, Stefania Lazar, Sebastian Krinner, Johannes Heinsoo, Jean-Claude Besse, Mihai Gabureac, Andreas Wallraff, and Christopher Eichler. 2019. Entanglement stabilization using ancilla-based parity detection and real-time feedback in superconducting circuits. *npj Quantum Information* 5, 1 (2019), 69.

[3] Ryan Babbush, Craig Gidney, Dominic W Berry, Nathan Wiebe, Jarrod McClean, Alexandru Paler, Austin Fowler, and Hartmut Neven. 2018. Encoding electronic spectra in quantum circuits with linear T complexity. *Physical Review X* 8, 4 (2018), 041015.

[4] Elisa Bäumer, Vinay Tripathi, Derek S Wang, Patrick Rall, Edward H Chen, Swarnadeep Majumder, Alireza Seif, and Zlatko K Minev. 2024. Efficient long-range entanglement using dynamic circuits. *PRX Quantum* 5, 3 (2024), 030339.

[5] Colin D Bruzewicz, John Chiaverini, Robert McConnell, and Jeremy M Sage. 2019. Trapped-ion quantum computing: Progress and challenges. *Applied Physics Reviews* 6, 2 (2019).

[6] Po-Yung Chang, Marius Evers, and Yale N Patt. 1997. Improving branch prediction accuracy by reducing pattern history table interference. *International journal of parallel programming* 25 (1997), 339–362.

[7] Yanhao Chen, Yuwei Jin, Fei Hua, Ari Hayes, Ang Li, Yunong Shi, and Eddy Z Zhang. 2023. A pulse generation framework with augmented program-aware basis gates and criticality analysis. In *2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 773–786.

[8] Jinglei Cheng, Haoqing Deng, and Xuehai Qia. 2020. Accqoc: Accelerating quantum optimal control based pulse generation. In *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 543–555.

[9] ID Conway Lamb, JI Colless, JM Hornibrook, SJ Pauka, SJ Waddy, MK Frechtling, and DJ Reilly. 2016. An FPGA-based instrumentation platform for use at deep cryogenic temperatures. *Review of Scientific Instruments* 87, 1 (2016).

[10] Julia Cramer, Norbert Kalb, M Adriaan Rol, Bas Hensen, Machiel S Blok, Matthew Markham, Daniel J Twitchen, Ronald Hanson, and Tim H Taminiau. 2016. Repeated quantum error correction on a continuously encoded qubit by real-time feedback. *Nature communications* 7, 1 (2016), 11526.

[11] Poulami Das, Aditya Locharla, and Cody Jones. 2022. Lilliput: a lightweight low-latency lookup-table decoder for near-term quantum error correction. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*. 541–553.

[12] Poulami Das, Christopher A Pattison, Srilatha Manne, Douglas M Carmean, Krysta M Svore, Moinuddin Qureshi, and Nicolas Delfosse. 2022. Afs: Accurate, fast, and scalable error-decoding for fault-tolerant quantum computers. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 259–273.

[13] Poulami Das, Swamit Tannu, Siddharth Dangwal, and Moinuddin Qureshi. 2021. ADAPT: Mitigating Idling Errors in Qubits via Adaptive Dynamical Decoupling. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture* (Virtual Event, Greece) *(MICRO '21)*. Association for Computing Machinery, New York, NY, USA, 950–962.

[14] Richard Gebauer, Nick Karcher, Daria Gusenkova, Martin Spiecker, Lukas Grünhaupt, Ivan Takmakov, Patrick Winkel, Luca Planat, Nicolas Roch, Wolfgang Wernsdorfer, Alexey Ustinov, Marc Weber, Martin Weides, Ioan Pop, and Oliver Sander. 2020. State preparation of a fluxonium qubit with feedback from a custom FPGA-based platform. In *AIP Conference Proceedings*, Vol. 2241. AIP Publishing.

[15] András M Gunyhó, Suman Kundu, Jian Ma, Wei Liu, Sakari Niemelä, Giacomo Catto, Vasilii Vadimov, Visa Vesterinen, Priyank Singh, Qiming Chen, and Mikko Möttönen. 2024. Single-shot readout of a superconducting qubit using a thermal detector. *Nature Electronics* (2024), 1–11.

[16] Cheng Guo, Jin Lin, Lian-Chen Han, Na Li, Li-Hua Sun, Fu-Tian Liang, Dong-Dong Li, Yu-Huai Li, Ming Gong, Yu Xu, Sheng-Kai Liao, and Cheng-Zhi Peng. 2022. Low-latency readout electronics for dynamic superconducting quantum computing. *AIP Advances* 12, 4 (2022).

[17] Riddhi S Gupta, Neereja Sundaresan, Thomas Alexander, Christopher J Wood, Seth T Merkel, Michael B Healy, Marius Hillenbrand, Tomas Jochym-O'Connor, James R Wootton, Theodore J Yoder, W Andrew Cross, Maika Takita, and J Benjamin Brown. 2024. Encoding a magic state with beyond break-even fidelity. *Nature* 625, 7994 (2024), 259–263.

[18] Johannes Heinsoo, Christian Kraglund Andersen, Ants Remm, Sebastian Krinner, Theodore Walter, Yves Salathé, Simone Gasparinetti, Jean-Claude Besse, Anton Potočnik, Christopher Eichler, and Andreas Wallraff. 2018. Rapid high-fidelity multiplexed readout of superconducting qubits. *Physical Review Applied* 10, 3 (2018), 034040.

[19] Fei Hua, Yanhao Chen, Yuwei Jin, Chi Zhang, Ari Hayes, Youtao Zhang, and Eddy Z Zhang. 2021. Autobraid: A framework for enabling efficient surface code communication in quantum computing. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*. 925–936.

[20] Gang Huang, Yilun Xu, Neelay Fruitwala, Abhi D Rajagopala, Kasra Nowrouzi, Ravi K Naik, David Santiago, and Irfan Siddiqi. 2023. QubiC 2.0: A Flexible Advanced Full Stack Quantum Bit Control System. In *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, Vol. 2. IEEE, 248–249.

[21] He-Liang Huang, Dachao Wu, Daojin Fan, and Xiaobo Zhu. 2020. Superconducting quantum computing: a review. *Science China Information Sciences* 63 (2020), 1–32.

[22] Evan Jeffrey, Daniel Sank, JY Mutus, TC White, J Kelly, R Barends, Y Chen, Z Chen, B Chiaro, A Dunsworth, A Megrant, P O'Malley, C. Neill, P Roushan, A Vainsencher, J Wenner, A Cleland, and J Martinis. 2014. Fast accurate state measurement with superconducting qubits. *Physical review letters* 112, 19 (2014), 190504.

[23] Hansol Kim, Wonjae Choi, Younghun Kim, and Younghun Kwon. 2024. Implementation of Magic State Injection within Heavy-Hexagon Architecture. *arXiv preprint arXiv:2412.15751* (2024).

[24] Jens Koch, Terri M Yu, Jay Gambetta, Andrew A Houck, David I Schuster, Johannes Majer, Alexandre Blais, Michel H Devoret, Steven M Girvin, and Robert J Schoelkopf. 2007. Charge-insensitive qubit design derived from the Cooper pair box. *Physical Review A—Atomic, Molecular, and Optical Physics* 76, 4 (2007), 042319.

[25] Ang Li, Samuel Stein, Sriram Krishnamoorthy, and James Ang. 2023. Qasmbench: A low-level quantum benchmark suite for nisq evaluation and simulation. *ACM Transactions on Quantum Computing* 4, 2 (2023), 1–26.

[26] Benjamin Lienhard, Antti Vepsäläinen, Luke C.G. Govia, Cole R. Hoffer, Jack Y. Qiu, Diego Ristè, Matthew Ware, David Kim, Roni Winik, Alexander Melville, Bethany Niedzielski, Jonilyn Yoder, Guilhem J. Ribeill, Thomas A. Ohki, Hari K. Krovi, Terry P. Orlando, Simon Gustavsson, and William D. Oliver. 2022. Deep-neural-network discrimination of multiplexed superconducting-qubit states. *Physical Review Applied* 17, 1 (2022), 014024.

[27] Daniel Litinski. 2019. A game of surface codes: Large-scale quantum computing with lattice surgery. *Quantum* 3 (2019), 128.

[28] William P Livingston, Machiel S Blok, Emmanuel Flurin, Justin Dressel, Andrew N Jordan, and Irfan Siddiqi. 2022. Experimental demonstration of continuous quantum error correction. *Nature communications* 13, 1 (2022), 2307.

[29] Liqiang Lu, Wuwei Tian, Xinghui Jia, Zixuan Song, Siwei Tan, and Jianwei Yin. 2024. SmartQCache: Fast and Precise Pulse Control With Near-Quantum Cache Design on FPGA. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2024).

[30] Andrii Maksymov, Jason Nguyen, Vandiver Chaplin, Yunseong Nam, and Igor L Markov. 2022. Detecting Qubit-coupling faults in ion-trap quantum computers. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 387–399.

[31] Satvik Maurya, Chaithanya Naik Mude, William D Oliver, Benjamin Lienhard, and Swamit Tannu. 2023. Scaling qubit readout with hardware efficient machine learning architectures. In *Proceedings of the 50th Annual International Symposium on Computer Architecture*. 1–13.

[32] Satvik Maurya and Swamit Tannu. 2022. Compaqt: Compressed waveform memory architecture for scalable qubit control. In *2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 1059–1077.

[33] David C McKay, Christopher J Wood, Sarah Sheldon, Jerry M Chow, and Jay M Gambetta. 2017. Efficient Z gates for quantum computing. *Physical Review A* 96, 2 (2017), 022330.

[34] Dongmoon Min, Junpyo Kim, Junhyuk Choi, Ilkwon Byun, Masamitsu Tanaka, Koji Inoue, and Jangwoo Kim. 2023. Qisim: Architecting 10+ k qubit qc interfaces toward quantum supremacy. In *Proceedings of the 50th Annual International Symposium on Computer Architecture*. 1–16.

[35] Sparsh Mittal. 2019. A survey of techniques for dynamic branch prediction. *Concurrency and Computation: Practice and Experience* 31, 1 (2019), e4666.

[36] MS Moreira, Gian Giacomo Guerreschi, Wouter Vlothuizen, Jorge F Marques, Jeroen van Straten, Shavindra P Premaratne, Xiang Zou, Hany Ali, Nandini Muthusubramanian, Christos Zachariadis, J. van Someren, M. Beekman, N. Haider, A. Bruno, C. G. Almudever, A. Y. Matsuura, and L. DiCarlo. 2023. Realization of a quantum neural network using repeat-until-success circuits in a superconducting quantum processor. *npj Quantum Information* 9, 1 (2023), 118.

[37] Emily Mount, Daniel Gaultney, Geert Vrijsen, Michael Adams, So-Young Baek, Kai Hudek, Louis Isabella, Stephen Crain, Andre van Rynbach, Peter Maunz, and Jungsang Kim. 2016. Scalable digital hardware for a trapped ion quantum computer. *Quantum Information Processing* 15 (2016), 5281–5298.

[38] Thomas E O'Brien, B Tarasinski, and Leo DiCarlo. 2017. Density-matrix simulation of small surface codes under current and projected experimental noise. *npj Quantum Information* 3, 1 (2017), 39.

[39] Jongseok Park, Sushil Subramanian, Lester Lampert, Todor Mladenov, Ilya Klotchkov, Dileep J. Kurian, Esdras Juarez-Hernandez, Brando Perez Esparza, Sirisha Rani Kale, Asma Beevi K. T., Shavindra P. Premaratne, Thomas F. Watson, Satoshi Suzuki, Mustafijur Rahman, Jaykant B. Timbadiya, Saksham Soni, and Stefano Pellerano. 2021. A fully integrated cryo-CMOS SoC for state manipulation, readout, and high-speed gate pulsing of spin qubits. *IEEE Journal of Solid-State Circuits* 56, 11 (2021), 3289–3306.

[40] Lukas Postler, Sascha Heuβen, Ivan Pogorelov, Manuel Rispler, Thomas Feldker, Michael Meth, Christian D Marciniak, Roman Stricker, Martin Ringbauer, Rainer Blatt, Philipp Schindler, Markus Müller, and Thomas Monz. 2022. Demonstration of fault-tolerant universal quantum gate operations. *Nature* 605, 7911 (2022), 675–680.

[41] IBM Fez Processor. 2024. *IBM Quantum Platform.* https://quantum.ibm.com/services/resources?system=ibm_fez

[42] Google quantum AI. 2023. Suppressing quantum errors by scaling a surface code logical qubit. *Nature* 614, 7949 (2023), 676–681.

[43] Gokul Subramanian Ravi, Jonathan M Baker, Arash Fayyazi, Sophia Fuhui Lin, Ali Javadi-Abhari, Massoud Pedram, and Frederic T Chong. 2023. Better than worst-case decoding for quantum error correction. In *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*. 88–102.

[44] Kevin Reuer, Jonas Landgraf, Thomas Fösel, James O'Sullivan, Liberto Beltrán, Abdulkadir Akin, Graham J Norris, Ants Remm, Michael Kerschbaum, Jean-Claude Besse, Florian Marquardt, Andreas Wallraff, and Christopher Eichler. 2023. Realizing a deep reinforcement learning agent for real-time quantum feedback. *Nature Communications* 14, 1 (2023), 7138.

[45] Leon Riesebos, Xiang Fu, Savvas Varsamopoulos, Carmen G Almudever, and Koen Bertels. 2017. Pauli frames for quantum computer architectures. In *Proceedings of the 54th Annual Design Automation Conference 2017*. 1–6.

[46] Diego Ristè, Josephine G van Leeuwen, H-S Ku, Konrad W Lehnert, and Leonardo DiCarlo. 2012. Initialization by measurement of a superconducting quantum bit circuit. *Physical review letters* 109, 5 (2012), 050507.

[47] Ciaran Ryan-Anderson, Justin G Bohnet, Kenny Lee, Daniel Gresh, Aaron Hankin, JP Gaebler, David Francois, Alexander Chernoguzov, Dominic Lucchetti, Natalie C Brown, TM. Gatterman, SK. Halit, K. Gilmore, JA. Gerber, B. Neyenhuis, D. Hayes, and RP. Stutz. 2021. Realization of real-time fault-tolerant quantum error correction. *Physical Review X* 11, 4 (2021), 041058.

[48] Yves Salathé, Philipp Kurpiers, Thomas Karg, Christian Lang, Christian Kraglund Andersen, Abdulkadir Akin, Sebastian Krinner, Christopher Eichler, and Andreas Wallraff. 2018. Low-latency digital signal processing for feedback and feedforward in quantum computing and communication. *Physical Review Applied* 9, 3 (2018), 034011.

[49] André Seznec. 2007. A 256 kbits l-tage branch predictor. *Journal of Instruction-Level Parallelism (JILP) Special Issue: The Second Championship Branch Prediction Competition (CBP-2)* 9 (2007), 1–6.

[50] Si Shen, Chenzhi Yuan, Zichang Zhang, Hao Yu, Ruiming Zhang, Chuanrong Yang, Hao Li, Zhen Wang, You Wang, Guangwei Deng, Haizhi Song, Lixing You, Yunru Fan, Guangcan Guo, and Qiang Zhou. 2023. Hertz-rate metropolitan quantum teleportation. *Light: Science & Applications* 12, 1 (2023), 115.

[51] Neil Shenvi, Julia Kempe, and K Birgitta Whaley. 2003. Quantum random-walk search algorithm. *Physical Review A* 67, 5 (2003), 052307.

[52] VV Sivak, Alec Eickbusch, Baptiste Royer, Shraddha Singh, Ioannis Tsioutsios, Suhas Ganjam, Alessandro Miano, BL Brock, AZ Ding, Luigi Frunzio, SM Girvin, RJ Schoelkopf, and MH. Devoret. 2023. Real-time quantum error correction beyond break-even. *Nature* 616, 7955 (2023), 50–55.

[53] James E Smith. 1998. A study of branch prediction strategies. In *25 years of the international symposia on Computer architecture (selected papers)*. 202–215.

[54] James E Smith. 1998. A study of branch prediction strategies. In *25 years of the international symposia on Computer architecture (selected papers)*. 202–215.

[55] Lars Steffen, Yves Salathe, Markus Oppliger, Philipp Kurpiers, Matthias Baur, Christian Lang, Christopher Eichler, Gabriel Puebla-Hellmann, Arkady Fedorov, and Andreas Wallraff. 2013. Deterministic quantum teleportation with feed-forward in a solid state system. *Nature* 500, 7462 (2013), 319–322.

[56] Samuel Stein, Sara Sussman, Teague Tomesh, Charles Guinn, Esin Tureci, Sophia Fuhui Lin, Wei Tang, James Ang, Srivatsan Chakram, Ang Li, Margaret Martonosi, Fred T. Chong, Andrew A. Houck, Isaac L. Chuang, and Michael Austin DeMarco. 2023. Hetarch: Heterogeneous microarchitectures for superconducting quantum systems. In *Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture*. 539–554.

[57] Yasunari Suzuki, Takanori Sugiyama, Tomochika Arai, Wang Liao, Koji Inoue, and Teruo Tanimoto. 2022. Q3DE: A fault-tolerant quantum computer architecture for multi-bit burst errors by cosmic rays. In *2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 1110–1125.

[58] Mario Szegedy. 2004. Quantum speed-up of Markov chain based algorithms. In *45th Annual IEEE symposium on foundations of computer science*. IEEE, 32–41.

[59] Daniel Bochen Tan, Murphy Yuezhen Niu, and Craig Gidney. 2024. A SAT Scalpel for Lattice Surgery: Representation and Synthesis of Subroutines for Surface-Code Fault-Tolerant Quantum Computing. *arXiv preprint arXiv:2404.18369* (2024).

[60] Siwei Tan, Liqiang Lu, Hanyu Zhang, Jia Yu, Congliang Lang, Yongheng Shang, Xinkui Zhao, Mingshuai Chen, Yun Liang, and Jianwei Yin. 2024. QuFEM: Fast and Accurate Quantum Readout Calibration Using the Finite Element Method. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*. 948–963.

[61] Mats O Tholén, Riccardo Borgani, Giuseppe Ruggero Di Carlo, Andreas Bengtsson, Christian Križan, Marina Kudra, Giovanna Tancredi, Jonas Bylander, Per Delsing, Simone Gasparinetti, and David Haviland. 2022. Measurement and control of a superconducting quantum processor with a fully integrated radio-frequency system on a chip. *Review of Scientific Instruments* 93, 10 (2022).

[62] Wuwei Tian, Xinghui Jia, Siwei Tan, Zixuan Song, Liqiang Lu, and Jianwei Yin. 2023. QPulseLib: Accelerating the Pulse Generation of Quantum Circuit with Reusable Patterns. In *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*. IEEE, 01–09.

[63] Teague Tomesh, Pranav Gokhale, Victory Omole, Gokul Subramanian Ravi, Kaitlin N Smith, Joshua Viszlai, Xin-Chuan Wu, Nikos Hardavellas, Margaret R Martonosi, and Frederic T Chong. 2022. Supermarq: A scalable quantum benchmark suite. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 587–603.

[64] Yosuke Ueno, Masaaki Kondo, Masamitsu Tanaka, Yasunari Suzuki, and Yutaka Tabuchi. 2022. Qulatis: A quantum error correction methodology toward lattice surgery. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 274–287.

[65] Antti Vepsäläinen, Roni Winik, Amir H Karamlou, Jochen Braumüller, Agustin Di Paolo, Youngkyu Sung, Bharath Kannan, Morten Kjaergaard, David K Kim, Alexander J Melville, BM. Niedzielski, Jonilyn. Yoder, Simon Gustavsson, and William Oliver. 2022. Improving qubit coherence using closed-loop feedback. *Nature Communications* 13, 1 (2022), 1932.

[66] Suhas Vittal, Poulami Das, and Moinuddin Qureshi. 2023. Astrea: Accurate quantum error-decoding via practical minimum-weight perfect-matching. In *Proceedings of the 50th Annual International Symposium on Computer Architecture*. 1–16.

[67] Theodore Walter, Philipp Kurpiers, Simone Gasparinetti, Paul Magnard, Anton Potočnik, Yves Salathé, Marek Pechal, Mintu Mondal, Markus Oppliger, Christopher Eichler, and A Wallraff. 2017. Rapid high-fidelity single-shot dispersive readout of superconducting qubits. *Physical Review Applied* 7, 5 (2017), 054020.

[68] Zhan Wang, Hai Yu, Rongli Liu, Xiao Ma, Xueyi Guo, Zhongcheng Xiang, Pengtao Song, Luhong Su, Yirong Jin, and Dongning Zheng. 2021. Hardware for multi-superconducting qubit control and readout. *Chinese Physics B* 30, 11 (2021), 110305.

[69] Karen Wintersperger, Florian Dommert, Thomas Ehmer, Andrey Hoursanov, Johannes Klepsch, Wolfgang Mauerer, Georg Reuber, Thomas Strohm, Ming Yin, and Sebastian Luber. 2023. Neutral atom quantum computing hardware: performance and end-user perspective. *EPJ Quantum Technology* 10, 1 (2023), 32.

[70] Liang Xiang, Zhiwen Zong, Zhenhai Sun, Ze Zhan, Ying Fei, Zhangjingzi Dong, Chongxin Run, Zhilong Jia, Peng Duan, Jianlan Wu, Yi Yin, and Guoping Guo. 2020. Simultaneous feedback and feedforward control and its application to realize a random walk on the bloch sphere in an xmon-superconducting-qubit system. *Physical Review Applied* 14, 1 (2020), 014099.

[71] Yuchen Yang, Zhongtao Shen, Xing Zhu, Ziqi Wang, Gengyan Zhang, Jingwei Zhou, Xun Jiang, Chunqing Deng, and Shubin Liu. 2022. FPGA-based electronic system for the control and readout of superconducting quantum processors. *Review of Scientific Instruments* 93, 7 (2022).

[72] Hanyu Zhang, Liqiang Lu, Siwei Tan, Size Zheng, Jia Yu, and Jianwei Yin. 2024. SpREM: Exploiting Hamming Sparsity for Fast Quantum Readout Error Mitigation. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*. 1–6.

[73] Mengyu Zhang, Lei Xie, Zhenxing Zhang, Qiaonian Yu, Guanglei Xi, Hualiang Zhang, Fuming Liu, Yarui Zheng, Yicong Zheng, and Shengyu Zhang. 2021. Exploiting different levels of parallelism in the quantum control microarchitecture for superconducting qubits. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*. 898–911.

[74] Yu Zhou, Zhenxing Zhang, Zelong Yin, Sainan Huai, Xiu Gu, Xiong Xu, Jonathan Allcock, Fuming Liu, Guanglei Xi, Qiaonian Yu, Hualiang Zhang, Mengyu Zhang, Hekang Li, Xiaohui Song, Zhan Wang, Dongning Zheng, Shuoming An, Yarui Zheng, and Shengyu Zhang. 2021. Rapid and unconditional parametric reset protocol for tunable superconducting qubits. *Nature Communications* 12, 1 (2021), 5924.