Embedded Calibration for Cryogenic Quantum Devices using a Singleton Sugeno Fuzzy Controller

Che-Ping Lin
Independent Researcher
Hsinchu City, Taiwan

ABSTRACT
Calibration loops are essential for maintaining cryogenic quantum devices under drift, stochastic readout variation, and device mismatch. This study presents an embedded closed-loop calibration architecture using a singleton Sugeno fuzzy controller with fixed-size arithmetic, bounded updates, and a lightweight telemetry interface. The emphasis is deterministic local execution rather than host-side optimization: each iteration performs measurement accumulation, probability estimation, error computation, fuzzy inference, and parameter update along a repeatable path. An executable Verilog-based evaluation model is used to evaluate convergence behavior, monitor-mode operation, and sensitivity to injected drift. The measured results show fast transition into monitor mode, fixed calibrated-parameter retention after convergence, and observable monitor-metric shifts under positive and negative perturbations. These results support the feasibility of compact embedded calibration for cryogenic quantum devices while keeping the controller structure simple, interpretable, and implementation-oriented.

General Terms
Algorithms, Design, Verification, Reliability.

Keywords
Quantum calibration, fuzzy logic, cryogenic control, embedded control, deterministic closed-loop calibration.

1. INTRODUCTION
Cryogenic quantum devices require repeated calibration to maintain target operating points in the presence of drift, noise, and device variability. As quantum hardware scales, calibration must be performed with bounded latency and limited control overhead rather than relying exclusively on slow host-side optimization loops. Recent work has emphasized scalable in-situ calibration under repetitive error-detection operation, motivating compact embedded calibration loops for quantum hardware [4].

This study investigates a singleton Sugeno fuzzy controller as an embedded calibration primitive. The main contributions are: (i) an embedded calibration loop with a lightweight configuration and telemetry interface; (ii) a deterministic singleton Sugeno controller with a compact 3x3 rule table and bounded parameter updates; and (iii) an executable Verilog-based prototype and simulation harness used to quantify convergence, monitor-mode behavior, and drift sensitivity. Rather than claiming superiority over all alternative calibration strategies, the present study focuses on a simple controller structure that is interpretable, hardware-friendly, and suitable for local execution inside a cryogenic control stack.

2. Embedded Calibration Controller Architecture
This section describes an implementation-agnostic architecture for running calibration loops without a host PC in the critical path. The goal is to keep the per-iteration latency bounded and reproducible while allowing a supervisor to schedule calibration windows across multiple channels. Integrated cryogenic quantum-control system designs and reconfigurable cryogenic control platforms illustrate the value of pushing control and configuration closer to the device, which aligns with the hostless partitioning adopted here [5] [6]. In addition, cryogenic CMOS control electronics enabling multi-qubit operations motivate a lightweight, deterministic controller interface that can be realized in hardware [7].

2.1 Functional blocks
The architecture is decomposed into five blocks: (i) a supervisor/scheduler, (ii) a measurement statistics estimator, (iii) an error computation unit, (iv) a decision logic block (singleton Sugeno fuzzy inference with a small rule table), and (v) a parameter update and actuation stage. The decision logic executes once per iteration using fixed-size arithmetic and a small rule table (for the fuzzy case).

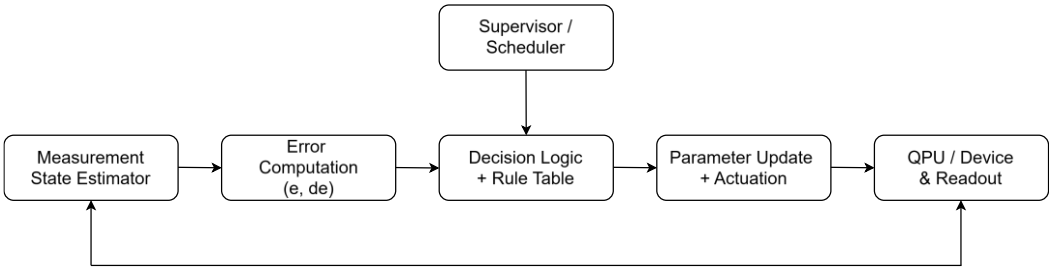

Figure 1. Embedded calibration controller architecture

2.2 Configuration and telemetry interface
A lightweight configuration and telemetry interface is assumed to program the 3x3 singleton table, target and limit registers, and operating modes, while exposing $p_{\text{meas}}(k)$, $e(k)$, $de(k)$, $\Delta(k)$, $\text{amp}(k)$, and optional status flags for tuning and debug.

2.3 Deterministic execution

Each iteration follows the same execution path: accumulate N_{total} shots, compute p_{meas}, compute (e, de), evaluate the controller, then update amp. No history-dependent optimization or variable-size inference is used, which keeps the worst-case latency predictable.

3. Singleton Sugeno Fuzzy Calibration Algorithm

All-RF calibration workflows have highlighted the need for simple on-device update rules with bounded per-iteration cost; a compact singleton Sugeno formulation is therefore used to map (e, de) directly to a bounded amplitude step $\triangle(k)$ [8].

3.1 Measurement and error signals

For each calibration iteration k, the controller collects N_total measurement shots and counts the number of logical-1 outcomes N_1. The measured success probability is estimated as:

$$p_{\text{meas}}(k) = \frac{N_1}{N_{\text{total}}}$$

Given a target probability p_target, the error and error difference are defined as:

$$e(k) = p_{\text{target}} - p_{\text{meas}}(k)$$
$$de(k) = e(k) - e(k-1)$$

3.2 Membership functions

Both e and de are fuzzified into three linguistic terms: Negative (N), Zero (Z), and Positive (P). Fixed symmetric triangular membership functions are implemented in fixed-point arithmetic, consistent with classical fuzzy-set formulations [1]. The implementation model computes $\mu_N(x)$, $\mu_Z(x)$ and $\mu_P(x)$directly from the signed error inputs.

Let x denote the normalized input (either e or de). As illustrated in Figure 2, the input is mapped into three symmetric triangular membership functions $\mu_N(x)$, $\mu_Z(x)$ and $\mu_P(x)$ over the range $[-1,1]$. The Zero term $\mu_Z(x)$ peaks at $x = 0$ and decreases linearly to zero at $x = \pm 1$, while $\mu_N(x)$ and $\mu_P(x)$ dominate the negative and positive regions, respectively, with linear overlaps around the boundaries. Adjacent terms intersect at $x = \pm 0.5$ (i.e., $\mu_N(-0.5) = \mu_Z(-0.5)$ and $\mu_Z(0.5) = \mu_P(0.5)$, and all membership values are clipped to $[0,1]$outside the range $[-1,1]$. This overlap yields smooth transitions in rule weights when e or de changes slightly, which is beneficial under fixed-point quantization and measurement jitter. In the Verilog-based implementation, these memberships are computed using simple piecewise-linear operations (comparison and subtraction) without lookup tables, enabling deterministic and low-cost hardware realization.

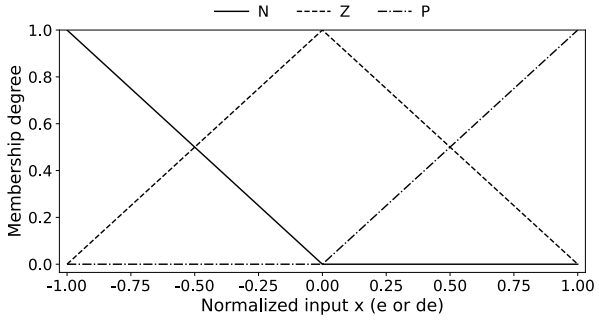

Figure 2. Symmetric triangular membership functions used for fuzzification of e and de.

3.3 Rule base and singleton consequents

The controller uses a 3x3 rule base. Each rule selects a singleton consequent D_{ij} (a signed fixed-point value) stored in a 9-entry table. Each D_{ij} represents a signed update step $\triangle$amp (in fixed-point), where the sign encodes the correction direction and the magnitude is bounded by $\triangle_{max}$ the clip operator. Table 1 summarizes the rule indexing used by the implementation model. This structure follows a compact Takagi-Sugeno style formulation with singleton consequents [2].

Table 1. Singleton Sugeno rule table (3×3)

e \ de	N	Z	P
N	D0	D1	D2
Z	D3	D4	D5
P	D6	D7	D8

3.4 Inference and defuzzification

Rule weights are computed as the product of membership values:

$$w_{ij}(k) = \mu_i\big(e(k)\big) \cdot \mu_j\big(de(k)\big)$$

The output update $\Delta(k)$ is the weighted average of singleton consequents:

$$\Delta(k) = \frac{\sum_{i,j} w_{ij}(k) \cdot D_{ij}}{\sum_{i,j} w_{ij}(k)}$$

A small sequential divider is used in implementation to compute the ratio. When the denominator is zero, Δ(k) is forced to zero.

3.5 Parameter update with bounds

The calibrated parameter amp is updated as:

$$\text{amp}(k+1) = \text{sat}\big(\text{amp}(k) + \text{clip}(\Delta(k), -\Delta_{max}, \Delta_{max})\big)$$

where clip limits the step size and sat enforces a configurable operating range.

Neural-network-based calibration (e.g., CNN/MLP controllers) can approximate complex nonlinear mappings, but typically requires substantially larger parameter storage and arithmetic throughput. For embedded calibration loops where deterministic latency, small memory footprint, and predictable execution are key, the proposed singleton fuzzy controller provides a practical nonlinear alternative without the overhead of large weight matrices. To highlight suitability for hostless embedded calibration, Table 2 summarizes the per-iteration computational footprint in terms of storage, arithmetic, and update-latency predictability.

Table 2. Per-iteration computational footprint

Method	Rule/Weight Memory	Multiplications	Division	Latency Predictability
No calibration	None	0	No	Deterministic
P	None	1	No	Deterministic
PD	None	2	No	Deterministic
Proposed Fuzzy	9 singleton entries	Constant (small)	Yes (bounded)	Deterministic

Unlike host-driven optimization or NN-based controllers whose inference cost scales with model size, the proposed Sugeno fuzzy update uses a fixed 3×3 rule base and bounded arithmetic, yielding deterministic per-iteration latency and small on-chip storage.

4. DESIGN RATIONALE AND DETERMINISTIC EXECUTION ANALYSIS

The proposed singleton Sugeno controller is motivated by the need for deterministic and bounded calibration updates in embedded quantum-control environments. In contrast to variable-latency host-side search procedures, the presented loop keeps the decision path fixed at every iteration and uses a compact rule table with bounded arithmetic.

Learning-based and optimization-heavy calibration strategies remain important for high-dimensional device tuning, but they typically involve larger model state, variable iteration cost, or additional supervisory software. The controller studied here instead targets a small-footprint local loop whose behavior can be analyzed directly from measurement statistics, rule activations, and parameter-update traces, improving reproducibility and debugging.

The bounded update mechanism further enhances stability. By enforcing a configurable step limit Δ_{max} and saturation constraints on the control parameter, the architecture prevents excessive corrections and keeps the closed-loop behavior interpretable for embedded deployment. Step limiting and saturation are common safeguards in practical feedback control to avoid over-correction and actuator saturation [3].

From an architectural perspective, the separation between measurement statistics, decision logic, and parameter update blocks supports modular verification and reproducibility. Each iteration follows an identical execution path, reinforcing deterministic behavior. This property is particularly relevant for scalable quantum systems where calibration routines must be replicated across multiple control channels.

The proposed calibration framework is organized as a high-level embedded control architecture rather than a hardware-specific implementation. The architecture consists of five conceptual blocks: (1) a supervisor that schedules calibration windows, (2) a measurement statistics estimator that computes the empirical success probability from shot outcomes, (3) an error computation unit that derives the instantaneous error $e(k)$ and its first-order difference $de(k)$, (4) a singleton Sugeno fuzzy decision block that generates a bounded update $\Delta(k)$ based on $(e,\ de)$, and (5) a parameter update stage that applies $\Delta(k)$ to the control variable under configurable limits.

This decomposition separates measurement processing, decision logic, and parameter actuation at an architectural level. The design can be mapped to an embedded controller, microcontroller, or application-specific control plane, but the algorithmic structure remains independent of any specific register-transfer or circuit-level realization.

A lightweight configuration interface is assumed for rule table programming, parameter readback, and operating-mode control (manual versus autonomous). However, the architectural description intentionally remains at a system level to emphasize algorithmic structure, deterministic execution flow, and modular separation of concerns.

Compared with host-driven calibration, where measurement data are transferred to a general-purpose processor for optimization, the proposed architecture embeds the full decision loop locally. This reduces communication overhead, preserves a deterministic iteration structure, and aligns with broader efforts toward scalable cryogenic and autonomous quantum-control infrastructures.

5. Experimental Setup and Results

5.1 Plant model and noise injection

To evaluate closed-loop behavior in a controlled setting, a simplified success-probability plant model is used:

$$p_{\text{meas}}(k) = \frac{1}{2} + \alpha \cdot \left(\text{amp}(k) - \text{amp}^{*}\right) + n(k)$$

where alpha is a gain constant, amp* is the target operating point, and n(k) denotes bounded pseudo-random additive noise in fixed-point arithmetic. In the Verilog-based evaluation model, this perturbation is implemented as an approximately uniform bounded term of about plus or minus 0.02 before clipping to the interval [0,1]. The noise term $n(k)$ is generated by a seeded pseudo-random source in the testbench and updated once per poll(k) in fixed-point arithmetic. After addition, $p_{\text{meas}}(k)$ is clipped to [0,1] to avoid invalid probabilities. Although simplified, this model captures the key interaction between a tunable control parameter and noisy measurement outcomes while remaining easy to reproduce. Learning-based online autotuning has also been explored for quantum-dot devices; a deterministic plant model is used to isolate the embedded controller dynamics and reproducibility [9].

5.2 Evaluation methodology and metrics

Evaluation is based on the executable Verilog model and simulation harness rather than an offline fit. Each poll records BUSY, CONV, UPD, p_{meas}, amp, and the monitor metric. In the implemented controller model, convergence is asserted either by an absolute residual threshold or by repeated steady-plateau detection on the window-averaged absolute error. The

implementation uses $E_{\text{tol}} = 0.125$, $DE_{\text{tol}} = 0.0625$, and a repeated-hit criterion before monitor mode is entered.

Convergence is evaluated on the window-averaged absolute error $|e|$ using a fixed window length W=16. A convergence hit is counted when $|e| \le E_{\text{tol}}$ and $|de| \le DE_{\text{tol}}$, monitor mode is entered only after H consecutive hits. In addition, a steady-plateau condition is used to avoid false convergence under quantization noise, where $\big||e|(k) - |e|(k-1)\big| \le \varepsilon$ must hold for H (hit-count) consecutive polls.

To probe monitor sensitivity, a controlled drift is injected from $k = 800$ to $k = 1199$ through simulation command-line parameters. Positive and negative perturbations are both evaluated, and the monitor metric is compared before, during, and after the programmed disturbance window. Within the controller model, the drift monitor compares the current window-averaged absolute error against a latched monitor baseline and asserts restart only after repeated deviation detection. Specifically, a baseline $\overline{|e|}_{\text{base}}$ is latched at monitor entry, and restart is asserted only when $\overline{|e|} - \overline{|e|}_{\text{base}} \ge T_{\text{rst}}$ for R consecutive polls (restart persistence). Here, T_{rst} denotes the restart threshold applied to the deviation of the monitor metric from its latched baseline in monitor mode. To make the evaluation setup more explicit, the main simulation conditions used in the Verilog-based prototype are summarized in Table 3. These conditions define the noise setting, monitoring window, and injected-drift configuration used for the convergence and drift-sensitivity experiments.

Table 3. Evaluation conditions used in the Verilog-based prototype

Item	Value	Purpose
Noise model	Bounded pseudo-random additive noise	Emulates measurement perturbation in the plant model
Noise amplitude	Approximately ±0.02	Sets the disturbance level before clipping
Window length	W = 16	Used for convergence and monitor evaluation
Drift injection start	k = 800	Defines the beginning of the programmed drift interval
Drift duration	DRIFT_LEN = 400	Defines the length of the injected drift
Drift cases	+0.3, -0.3	Evaluates monitor response under opposite perturbation directions
Evaluation platform	Verilog model and simulation harness	Provides executable prototype-based measurement

5.3 Results

Figures 3–5 and Table 3 and 4 summarize the measured behavior of the Verilog-based prototype. As listed in Table 4, the closed-loop update converged at $k = 20$ and entered monitor mode at $k = 21$ for both the $+0.3$ and -0.3 drift cases. Thus, the delay between convergence detection and monitor-mode entry was one polling interval. The update cadence was $U = 40$ f, and the converged calibrated parameter was $amp = 2.000$in both runs. After convergence, the calibrated parameter remained unchanged, while the residual stayed bounded at a nonzero level. Under injected drift, the 16-sample moving-average monitor metric changed from its pre-drift baseline in both runs; however, under the current restart threshold, no automatic restart was triggered.

As shown in Figure 3, the calibrated parameter reached its converged value of $amp = 2.000$ at $k = 20$ and remained unchanged after monitor mode was entered at $k = 21$. This result indicates deterministic settling followed by parameter retention in monitor mode, confirming stable operation after convergence.

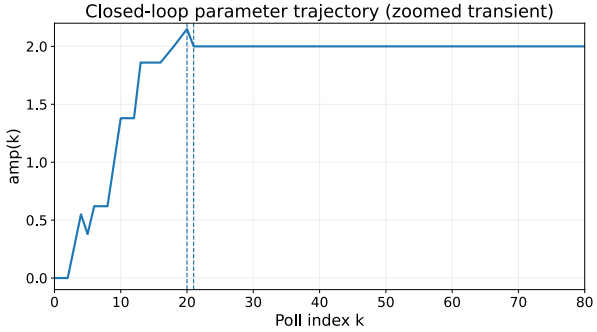

Figure 3. Measured trajectory of the calibrated parameter

As shown in Figure 4, once the loop has converged and entered monitor mode, the post-startup absolute error $|e(k)|$ does not collapse to zero. Instead, it settles into a bounded residual plateau. This behavior is expected in a fixed-point, windowed-measurement setting, where quantized arithmetic and finite-shot measurement noise impose a nonzero error floor even after the calibrated parameter is frozen.

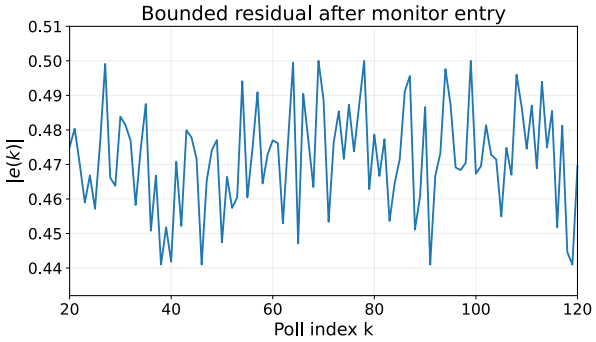

Figure 4. Measured absolute-error trajectory after startup

Figure 5 visualizes the monitor-stage response to injected drift by plotting the 16-sample moving-average of the absolute error, $|e|$, versus the polling index k. The shaded region indicates the interval where drift is actively injected, while the unshaded portions correspond to the pre-drift baseline and the post-drift recovery. Two traces are shown for opposite drift polarities (positive vs. negative). In both cases, the monitor metric departs from the baseline once the drift window begins and remains shifted throughout the injection period, demonstrating sensitivity to slow directional perturbations even after the controller has entered monitor mode. After drift removal, the metric trends back toward the baseline range, indicating that the observed shift is driven by injected drift rather than random fluctuation.

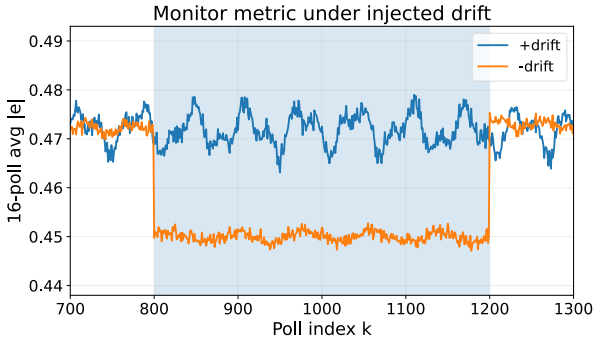

Figure 5. Moving-average monitor metric under positive and negative injected drift

Table 4 consolidates the key convergence and drift indicators extracted from the Verilog simulation logs, enabling a compact, side-by-side comparison of both steady-state behavior and drift response. Specifically, it reports (i) the iteration index at which the parameter update converges (k_{conv}) and the subsequent transition to monitor mode (k_{mon}), (ii) the update cadence (UPD_f) and the converged parameter value (amp_f), and (iii) representative error statistics before drift injection (e_{pre}), during the drift interval (e_{drift}), and after drift ends (e_{post}). By summarizing these fields for both positive and negative drift cases, Table 4 makes it easy to verify that the controller reaches the same stable operating point under nominal conditions and to quantify how the bounded monitor metric shifts under directional perturbations.

Table 4. Measured convergence and drift summary (k = 800, DRIFT_LEN = 400)

Run	k_{conv}	k_{mon}	UPD_f	amp_f	e_{pre}	e_{drift}	e_{post}
+0.3	20	21	40	2.000	0.472	0.450	0.473
-0.3	20	21	40	2.000	0.472	0.470	0.473

The positive and negative drift runs both change the monitor metric relative to the pre-drift baseline, although the shift is modest. This supports the architectural claim that an embedded monitor can detect operating-point movement with bounded local telemetry even when the current restart thresholds are conservative.

5.4 Discussion

The Verilog-based prototype supports a practical interpretation of embedded calibration: a short deterministic tune-up phase followed by monitor-mode operation with observable drift sensitivity. In particular, the measured trajectories in Figs. 3–5 and the summary in Table 4 show that the controller reaches a stable operating point within a bounded number of iterations and then freezes the calibrated parameter, which is desirable for cryogenic control where latency and control overhead must be limited.

A key observation is the bounded, nonzero residual plateau after monitor entry (Figure 4). This behavior is expected in a windowed-measurement setting with fixed-point arithmetic: finite-shot statistics, quantization, and bounded update steps impose an error floor even after the loop has converged. Rather than chasing measurement noise, the controller converges to a robust steady state, which is consistent with the design goal of deterministic execution and hardware-friendly implementation.

The drift experiment (Figure 5) highlights that the lightweight monitor can detect slow directional perturbations without re-entering a full optimization loop. The moving-average metric exhibits a clear level shift during the injected-drift interval for both drift polarities and trends back toward the baseline range after drift removal. This suggests that the monitor stage can provide a practical trigger for maintenance actions (e.g., re-calibration scheduling) under bounded compute and memory budgets.

The rule-based design is intentionally compact. The membership partitioning and inference datapath are fixed and deterministic in the RTL implementation, while the singleton consequents are stored in a small 9-entry table that can be reprogrammed via the register interface. This separation enables deployment across different devices or operating regimes by retuning only the consequent table, without modifying the underlying controller architecture.

Finally, the presented plant model is simplified and intended to isolate controller dynamics and reproducibility. Future work includes evaluating the same RTL implementation against more realistic readout/plant models (e.g., nonlinear response and time-correlated noise) and quantifying hardware cost (area, timing, and power) for an ASIC/FPGA implementation.

6. CONCLUSION

This study presented an embedded calibration architecture and a singleton Sugeno fuzzy controller suitable for cryogenic quantum-device control. The executable Verilog-based prototype showed deterministic transition from update mode to monitor mode, bounded steady-state residual behavior, and measurable monitor deviation under injected drift. These results position the approach as a lightweight local-control alternative for calibration paths where deterministic latency and implementation simplicity are prioritized.

Future work may extend the present framework in several directions. First, the same controller may be evaluated with more realistic plant and readout models, including nonlinear transfer characteristics, time-correlated noise, device mismatch, and longer-term drift patterns, in order to better approximate practical cryogenic operating conditions. Second, multi-channel calibration scenarios may be investigated to study how a lightweight embedded controller can be scheduled and scaled across multiple qubits or readout/control paths while maintaining bounded latency and limited supervisory overhead. Third, monitor-mode policies may be refined further, including adaptive restart thresholds, persistence conditions, and drift-trigger criteria, so that the controller can respond more effectively to slow operating-point movement without introducing unnecessary recalibration events.

In addition, neuro-fuzzy or adaptive fuzzy extensions may be explored to reduce human intervention in rule-base design and consequent tuning. Such extensions could improve flexibility across different devices and operating regions while still preserving the compact architecture, deterministic execution style, and relatively low-power characteristics required for practical embedded cryogenic control.

Finally, implementation-level evaluation in FPGA and ASIC targets remains an important next step. Quantifying area, timing, memory footprint, and power consumption would help determine the feasibility of integrating this calibration loop directly into larger cryogenic control systems. In this sense, the proposed local-control primitive may serve as a practical building block for broader efforts toward scalable spin-qubit interfacing and system integration [10].

7. Data Available

All artifacts used in this study are available publicly. The executable Verilog prototype (RTL), simulation testbench/harness, and configuration files in the GitHub repository:

https://github.com/acelin1981/embedded-qcal-sugeno-controller.

8. REFERENCES

[1] L. A. Zadeh, "Fuzzy sets," Information and Control, vol. 8, no. 3, pp. 338-353, 1965.

[2] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," IEEE Transactions on Systems, Man, and Cybernetics, vol. 15, no. 1, pp. 116-132, 1985.

[3] R. C. Dorf and R. H. Bishop, Modern Control Systems, 13th ed., Pearson, 2016.

[4] J. Kelly et al., "Scalable in-situ qubit calibration during repetitive error detection," arXiv:1603.03082, 2016.

[5] M. Prathapan et al., "A system design approach toward integrated cryogenic quantum control systems," arXiv:2211.02081, 2022

[6] H. Homulle, L. Song, X. Xue, J. van Staveren, S. Visser, G. Patra, A. P. M. Kentie, S. K. G. Garits, R. L. M. Op het Veld, D. Schinkel, and F. Sebastiano, "A reconfigurable cryogenic platform for the classical control of scalable quantum computers," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 25, no. 10, pp. 2590-2601, 2017.

[7] D. L. Underwood et al., "Using cryogenic CMOS control electronics to enable a two-qubit cross-resonance gate," PRX Quantum, vol. 4, no. 2, 020358, 2023.

[8] C. Carlsson et al., "Automated all-RF tuning for spin qubit readout and control," arXiv:2506.10834, 2025.

[9] V. Yon et al., "Experimental online quantum dots charge autotuning using neural networks," Nano Letters, vol. 25, no. 10, pp. 3717-3725, 2025.

[10] L. M. K. Vandersypen et al., "Interfacing spin qubits in quantum dots and donors - hot, dense, and coherent," npj Quantum Information, vol. 3, art. 34, 2017