

Low-latency machine learning FPGA accelerator for multi-qubit-state discrimination

Pradeep Kumar Gautam, Shantharam Kalipatnapu, Shankaranarayanan H, Ujjawal Singhal, Benjamin Lienhard, Vibhor Singh and Chetan Singh Thakur

Abstract—Measuring a qubit state is a fundamental yet error-prone operation in quantum computing. These errors can arise from various sources, such as crosstalk, spontaneous state transitions, and excitations caused by the readout pulse. Here, we utilize an integrated approach to deploy neural networks onto field-programmable gate arrays (FPGA). We demonstrate that implementing a fully connected neural network accelerator for multi-qubit readout is advantageous, balancing computational complexity with low latency requirements without significant loss in accuracy. The neural network is implemented by quantizing weights, activation functions, and inputs. The hardware accelerator performs frequency-multiplexed readout of five superconducting qubits in less than 50 ns on a radio frequency system on chip (RFSoc) ZCU111 FPGA, marking the advent of RFSoc-based low-latency multi-qubit readout using neural networks. These modules can be implemented and integrated into existing quantum control and readout platforms, making the RFSoc ZCU111 ready for experimental deployment.

Index Terms—Quantum computing, superconducting qubits, multi-qubit readout, FPGA, RFSoc, machine learning, neural networks

I. INTRODUCTION

QUANTUM processors are expected to solve specific computational tasks significantly more efficiently than their classical counterparts [1]–[5]. However, errors are inevitable during the operation of quantum processors due to the inherent instabilities of qubits and the challenges in controlling and reading out their quantum states, especially as quantum processors scale up in the number of qubits [6].

This work was supported by the Centre for Excellence in Quantum Technology (CoE-QT) through the Ministry of Electronics and Information Technology (MeitY) and Quantum Enabled Science and Technology (QuEST) Program within the Department of Science and Technology (DST), Government of India (GoI). The work of Vibhor Singh and Chetan Singh Thakur was supported by the Institute of Eminence (IoE) Scheme, Government of India.

Pradeep Kumar Gautam is with Department of Electronic Systems Engineering, Indian Institute of Science, Bangalore 560012, India and Defence Research and Development Organisation, Bangalore 560093, India.

Shantharam Kalipatnapu, Shankaranarayanan H and Chetan Singh Thakur (Corresponding Author) are with Department of Electronic Systems Engineering, Indian Institute of Science, Bangalore 560012, India. (E-mail: csthakur@iisc.ac.in)

Benjamin Lienhard is with Department of Chemistry, Princeton University, Princeton, NJ 08544, USA and Department of Electrical and Computer Engineering, Princeton University, Princeton, NJ 08544, USA.

Ujjawal Singhal and Vibhor Singh (Corresponding Author) are with the Department of Physics, Indian Institute of Science, Bangalore 560012, India. (E-mail: vsingh@iisc.ac.in)

(Corresponding authors: Vibhor Singh; Chetan Singh Thakur)

Quantum error correction (QEC) schemes address errors by redundantly encoding quantum information and performing repeated measurements to detect and correct errors during computation [7]. The successful execution of QEC algorithms relies on constantly monitoring a group of qubits and swiftly invoking corrective action [8]. To reduce resource requirements, especially from a readout perspective, multi-qubit readout is often achieved using frequency-division multiplexing [9]. Various methods, including matched filters, support vector machines (SVM), and neural networks (NN), have been used to process these frequency-multiplexed signals for inferring qubit states [10]–[14].

When implementing such methods, minimizing latency in post-processing for qubit-state inference is crucial for enabling on-the-fly corrective actions in QEC schemes. NN-based state discriminators outperform traditional signal processing techniques in post-processing readout signals, as demonstrated on various qubit platforms [13], [15]–[17]. Additionally, NN-based discriminators scale efficiently, as they do not require qubit-specific processing, such as digital demodulation, for each qubit [13].

Scalable field-programmable gate array (FPGA) systems, particularly radio frequency system on chip (RFSoc) solutions for controlling and reading out individual qubits in quantum computing, have recently gained significant attention [18]–[21]. In this context, NN-based state discriminators could significantly improve traditional signal processing solutions on FPGAs for frequency-multiplexed readout, enhancing throughput and reducing latency for real-time applications. However, a significant challenge is the substantial resource requirements for implementing NNs on hardware platforms like FPGAs [14].

Overcoming these challenges can enhance readout signal post-processing, allow for additional error correction cycles in QEC protocols, and facilitate the integration of reinforcement learning agents into more complex systems [22]. Quantization effectively reduces a model’s storage and computational demands by representing parameters in low-precision fixed-point formats [23], [24]. Quantization-aware training (QAT) has been an effective method to quantize down model parameters to binary precision [25], [26].

In this work, we tackle the aforementioned challenges and demonstrate that we can design ultra-low latency NN-based qubit-state discriminators by employing QAT and automated flows for mapping NN onto an FPGA, we can design ultra-low latency, NN-based qubit-state discriminators. We present an integrated approach for an NN-based qubit-state discriminator

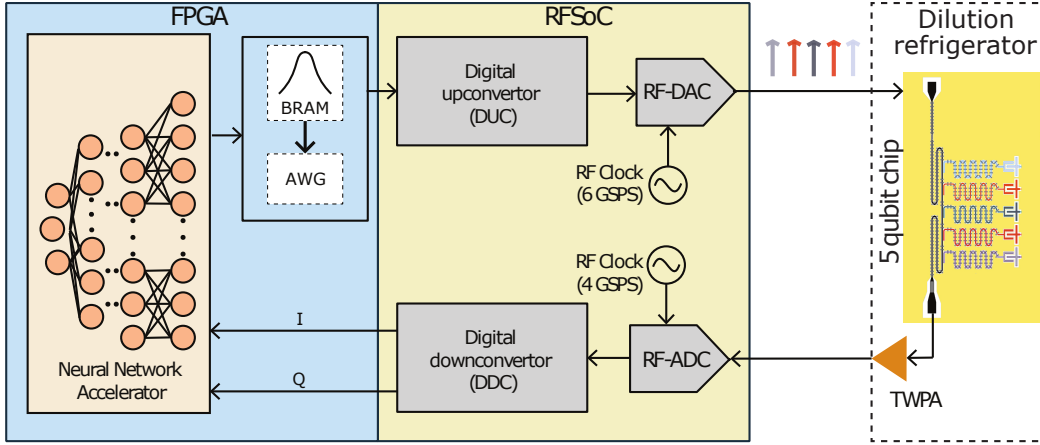


Fig. 1: Block diagram of a superconducting quantum processor interfacing with an FPGA system for qubit control and readout. The FPGA is part of the ZCU111 RFSoc evaluation kit. The quantum processor, which contains five superconducting qubits, is housed in a dilution refrigerator (for more details on the quantum processor and experimental setup, consult Ref. [13]). RF digital-to-analog converter (RF-DAC) process control signals for the qubits. The readout signal combines all five readout tones and is transmitted through a single feedline. This signal is first amplified by, among others, a traveling-wave parametric amplifier (TWPA) and then digitized by an RF analog-to-digital converter (RF-ADC). The RF-ADC converts the frequency-multiplexed readout signal into in-phase (I) and quadrature (Q) components. A machine learning accelerator for qubit-state discrimination subsequently processes these components. Based on the inferred qubit states, a feedback signal may be generated to drive the subsequent control pulse generation logic.

design, covering everything from training to FPGA implementation, using tools such as Brevitas [27] and FINN-R [25]. This approach enables qubit-state inference with latencies ranging from 24.03 ns to 47.12 ns, significantly reducing the signal processing delays that have previously limited the number of QEC cycle repetitions.

The article is organized as follows: Section II covers the preliminaries. Section III details the design methodology and implementation of NN-based qubit-state discriminators. Section IV presents the results and compares them with state-of-the-art methods. Finally, Section V provides the conclusion of the work.

II. SUPERCONDUCTING QUBIT READOUT

A typical schematic of the experimental setup used to interface with a quantum processor comprising superconducting qubits is shown in Fig. 1. The setup includes a dilution refrigerator maintained at a temperature of around 20 mK housing a five-qubit quantum processor and an RFSoc board at room temperature used to generate microwave pulses for qubit control and readout and post-process readout signals. Specifically, the readout probe pulse is passed through a radio frequency (RF) digital-to-analog converter (RF-DAC) and, after interacting with the quantum processor to acquire a qubit-state-specific signal characteristic, is fed into a high-speed RF analog-to-digital converter (RF-ADC). The RF-ADC and RF-DAC form the interface between the analog signals to and from the quantum processor and the digital RFSoc processing elements.

We use the SQ-CARS architecture developed on the Xilinx RFSoc ZCU111 [21]. The RFSoc device XCZU28DR

includes eight high-precision, low-power RF-DACs and RF-ADCs with maximum sampling rates of 6.554 GSPS and 4.096 GSPS, respectively. These data converters are configurable and integrated with the programmable logic resources of the RFSoc via AXI interfaces, which are standard for exchanging data between connected components.

The superconducting transmon qubits studied here have frequencies between 4.3 GHz to 5.2 GHz and energy relaxation times from 7 μ s to 40 μ s. Detailed characterization of the device can be found in Ref. [13].

For the readout of all qubits, a frequency-multiplexed pulse with a duration of around 1 μ s, consisting of superimposed signals at intermediate frequencies (IF) of 64.729 MHz, 25.366 MHz, 24.79 MHz, 70.269 MHz, and 127.282 MHz, is upconverted to the readout resonators' frequencies in the GHz-range using a local oscillator frequency of 7.127 GHz. The same local oscillator is used for down-conversion of the readout pulse after acquiring a qubit-state-specific phase shift for further digitization at a sampling rate of 500 MHz and post-processing, such as qubit-state discrimination.

The performance metric used to analyze the multi-qubit-state discrimination performance is F_{GM} , the geometric mean fidelity of the five qubits. The fidelity for the i -th qubit, F_i , and F_{GM} are defined as:

$$F_i = 1 - \frac{[P(0_i|\pi_i) + P(1_i|\emptyset_i)]}{2} \quad (1)$$

$$F_{GM} = \sqrt[5]{F_1 F_2 F_3 F_4 F_5}, \quad (2)$$

where $P(0_i|\pi_i)$ and $P(1_i|\emptyset_i)$ represent the conditional probabilities of assigning the ground (excited) state with label 0 (1) to qubit i when it is prepared in the excited (ground) state.

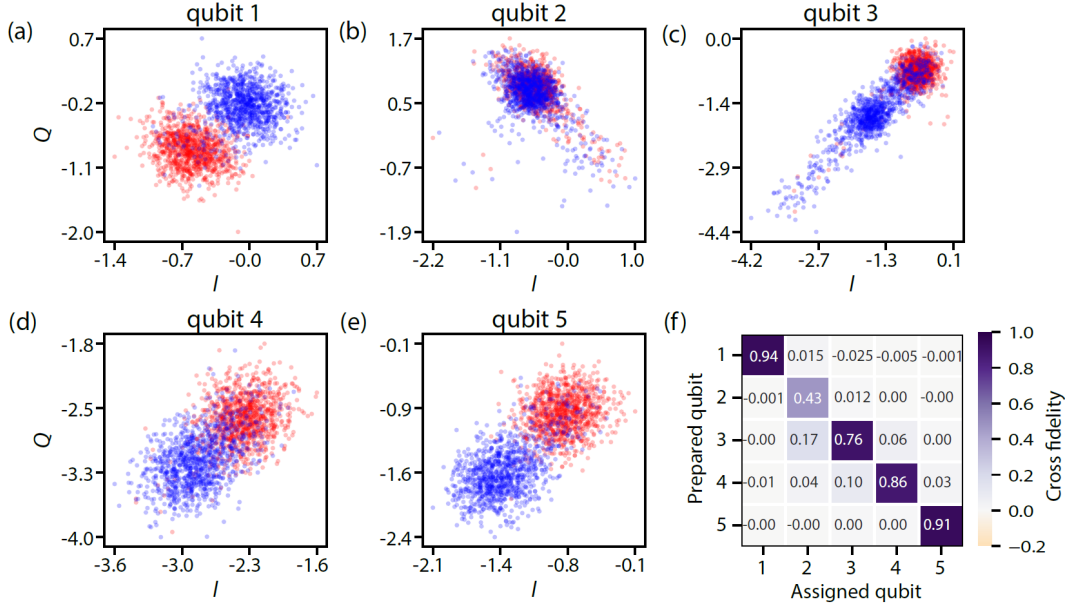


Fig. 2: Characteristics of qubit-specific single-shot readout traces. Panels (a-e) show the state discrimination of integrated single-shot traces for all five superconducting qubits, with red (blue) markers indicating the inferred ground (excited) state of each qubit. Panel (f) presents the cross-fidelity matrix, calculated using matched filters to infer the qubit states.

Multi-qubit processors often face the challenge of crosstalk, where signals—for control and readout—interfere with one another. Cross-fidelity is a key metric for assessing the performance of multi-qubit state discriminators amidst crosstalk, which quantifies the correlation between the assignment fidelities of individual qubits [28]. The cross-fidelity between two qubits is defined as:

$$F_{ij}^{CF} = 1 - [P(1_i|\emptyset_j) + P(0_i|\pi_j)], \quad (3)$$

where $P(1_i|\emptyset_j)$ ($P(0_i|\pi_j)$) represents the preparation of qubit j in the ground (excited) state and subsequent detection of qubit i in the excited (ground) state. Positive (negative) off-diagonal elements represent a correlation (anti-correlation) between qubits. Here, in the experimental data, the observed correlations are primarily caused by readout crosstalk [13].

Fig. 2(a)-(e) displays the results of integrated single-shot measurements in the IQ -plane for all five superconducting qubits. The cross-fidelity based on the matched filter state discriminator is shown in Fig. 2(f). The off-diagonal colored cells indicate the correlation strength.

III. LOW-LATENCY NEURAL NETWORK STATE DISCRIMINATOR

NN-based qubit-state discriminators have proven effective in mitigating readout crosstalk in multi-qubit systems [13]. However, their integration onto dedicated hardware, such as RFSocS, has not been accomplished yet due to the excessive storage and computational demands associated with these discriminators [14].

Quantization effectively reduces the storage and computational demands of models by representing parameters in low-precision fixed-point formats, often with minimal impact on accuracy [23], [24]. When employing QAT, low-precision

quantized models, including those with binary precision, generally maintain high accuracy [25], [26]. We utilize Brevitas' QAT [27] to leverage arbitrary bit-width and mixed precision within the PyTorch framework [29].

A. Neural Network Model Optimization and Quantization

The potential for mixed-precision quantization of inputs, weights, and activations down to one bit within a large NN design necessitates careful architectural choices. To streamline design exploration, we adopted a strategy similar to that in Ref. [30] and started with a base model presented in Ref. [13]. This model has an input size of 1000, which includes 500 samples of each of the in-phase (I) and quadrature (Q) components corresponding to $1\ \mu\text{s}$ of measurement time. The model features three hidden layers with 1000, 500, and 250 nodes, respectively, and an output layer with 32 nodes.

Initially, we adjusted the input feature size and the number of nodes in each hidden layer to the nearest powers of two. We then systematically reduced the input size and the number of nodes in the hidden layers. The number of output nodes is kept equal to the number of qubits rather than the total number of possible state combinations. This approach exponentially reduces the size of the output layer, using only five nodes (one for each qubit) instead of 32 nodes, representing the 2^N possible state combinations [13], [14]. The training was conducted for 50 epochs with a linearly decaying learning rate starting at 10^{-3} .

The various model archetypes are represented as $N_I \times N_{H_1} \times \dots \times N_{H_k} \times N_O$, where N_I , N_{H_i} , and N_O denote the number of nodes in the input layer, i^{th} hidden layer ($i \in \mathbb{Z}^+$), and output layer, respectively.

Each hidden layer includes a linear transformation followed by batch normalization, a dropout layer, and a rectified linear

unit (ReLU). The model inputs are derived from boxcar operations on I and Q samples by reducing the input feature size from 1024 to 512. We also examined the effect of model size on fidelity. Different architectures were trained using complete floating-point precision.

Fig. 3(a) illustrates the combined effects of model size variation and input feature size on the geometric mean fidelity, F_{GM} . It appears that reducing the input feature size has a minimal impact on fidelity, a maximum of 0.3%. Similarly, decreasing the number of hidden layers to 2 and a reduction in nodes composing the hidden layers only marginally affect F_{GM} . These results suggest that the model size can be carefully reduced without a significant readout fidelity loss. We selected the configuration $512 \times 64 \times 5$, as further reduction in model size leads to a rapid decline in F_{GM} .

We perform quantization of the models to identify the optimal representation for input, weights, and activations. The training is conducted using the same hyperparameters as the earlier approach. We vary the bit-widths of input, weights, and activations from 8-bit to 2-bit and examine their effect on the F_{GM} , as shown in Fig. 3(b) for the model configuration $512 \times 64 \times 5$. The results indicate that F_{GM} degrades for input quantization below 4-bit. Specifically, F_{GM} drops to approximately 0.7 for 2-bit input quantization. This suggests that at least 4-bit representation is necessary for accurate state discrimination in a five-qubit system. Additionally, the fidelity does not significantly vary with changes in input bit-width when activation and weight bit-widths are kept equal.

Fig. 3(c) displays the variation in F_{GM} for mixed precision quantization of weights and activations, ranging from 8-bit to 2-bit. The results indicate that mixed precision does not significantly impact fidelity, consistent with the trend observed in Fig. 3(b). However, a model with binary quantization shows a notable decrease in accuracy, with fidelity dropping by 4-21%, as illustrated in Fig. 3(d).

Fig. 3(e) explores the effect of network parameter count on F_{GM} across different hidden layer sizes. It reveals that the fidelity remains relatively stable regardless of the hidden layer configuration when the number of parameters exceeds 10^4 . Therefore, networks with fewer hidden layers are advisable to achieve lower latency while maintaining a high readout fidelity.

Based on the above-reported studies, we choose a quantization scheme of 4-bit for the input and 2-bit for both weights and activations. The resulting cross-fidelity matrix is computed and displayed in Fig. 3(f). This matrix indicates that the quantized neural network (QNN) significantly outperforms the matched filter discriminator and reduces crosstalk.

B. FPGA Acceleration

Low-latency readout is a primary requirement for mid-circuit measurements and is essential to QEC. To achieve low-latency readout, the dataflow architecture-based framework of FINN-R [25] stands out. FINN-R accepts models in the open NN exchange (ONNX) format with embedded FINN-specific metadata, which can be exported using the Brevitas library in the PyTorch environment. The model is then transformed

into a streaming dataflow graph, with each node represented as a Xilinx high-level synthesis (HLS) callable function. Nodes without equivalent HLS functions would need to run on the processing system of the FPGA, resulting in increased latency.

FINN-R implements quantization and matrix multiplication of low-precision data using multi-vector threshold units (MVTU), with one such unit used for each layer of the QNN. Each unit consists of several processing elements with multiple input lanes, similar to a *single instruction multiple data* architecture. The Appendix A provides more details on the FINN-R flow.

Trained and quantized models are adapted for varying levels of parallelization, constrained by resource availability, HLS conversion capabilities, and AXI-stream connection width. Layers that exceed a single MVTU unit are time-multiplexed, which impacts computation latency. A significant hurdle in achieving complete parallelism for moderate-size models is the limitation imposed by the connection width of the AXI-stream interconnect [31].

To address this constraint, we have designed a novel architecture to maximize the inherent parallelism of FPGAs. The modified hardware architecture is based on a NN with a configuration of $512 \times 64 \times 5$ as described earlier. The first hidden layer of the model, consisting of 64 nodes, is divided into eight equal segments, each containing eight nodes. The 512 nodes of the input layer are connected to all segments, effectively maintaining 64 nodes in the first hidden layer. These segmented nodes can operate in parallel on the FPGA, eliminating the need for time-multiplexing and reducing total latency. As a result, the connection requirement for the first hidden layer decreases from 32,768 to just 4,096 per segment. The outputs of these segments are concatenated using a *Concat* layer. The model architecture is illustrated in Fig. 4(a).

However, the generated ONNX model cannot be fully converted to a streaming dataflow-compliant representation using the default FINN-R flow due to the presence of the *Concat* layer and non-uniform multiplication nodes preceding it. To achieve a fully dataflow-convertible model, we inserted uniform *QuantIdentity* layers before the *Concat* layer and introduced additional model graph modification steps within the default transformation process of the FINN-R flow. This novel architecture fully parallelizes the FPGA implementation of the model and reduces the latency by eliminating the need for time multiplexing. This approach effectively achieves ultra-low latency on FPGAs when working with large neural networks. Fig. 4(b) displays the modified NN design generated by FINN-R.

As an alternative approach to attain low latency, we have also implemented a deeper NN that processes the input in a piecewise manner [22]. In this configuration, only the last layer contributes to the latency of qubit-state discrimination, allowing for a deeper network with small hidden layers. The architecture for the model with a size of $256 \times 128 \times 128 \times 128 \times 128 \times 5$ is displayed in Fig. 4(c).

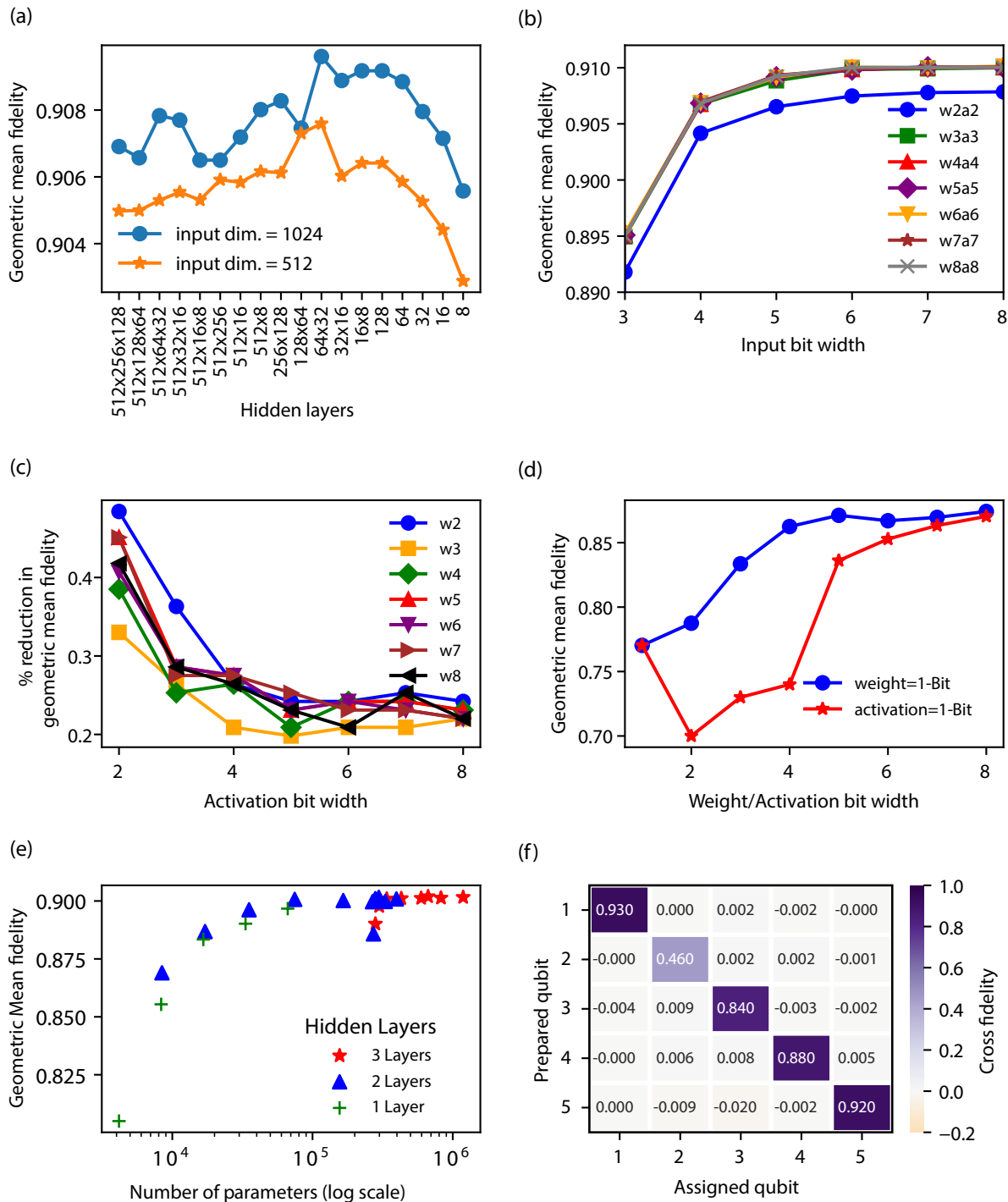


Fig. 3: Effects of neural network (NN) architecture parameters on readout fidelity. (a) Fidelity for various NN architectures with input feature sizes of 1024 and 512. The horizontal axis represents the dimensions of the hidden layers. (b) Impact on geometric mean fidelity F_{GM} with varying input bit quantization sizes. The labels ‘w#a#’ indicate the quantization of weights and activations where # indicates the number of bits used for quantization. (c) Effects of mixed quantization of weights and activations on fidelity. (d) Impact of mixed quantization of weights and activations with binarized input. The blue (red) curve shows fidelity variation with activation (weight) bit width for weights quantized to a single bit. (e) Effect of model parameters and depth on readout fidelity. The input, weights, and activations are quantized to 4, 2, and 2 bits, respectively. (f) Cross-fidelity matrix for the quantized NN. For panels (b), (c), (d), and (f), the NN architecture is $512 \times 64 \times 5$.

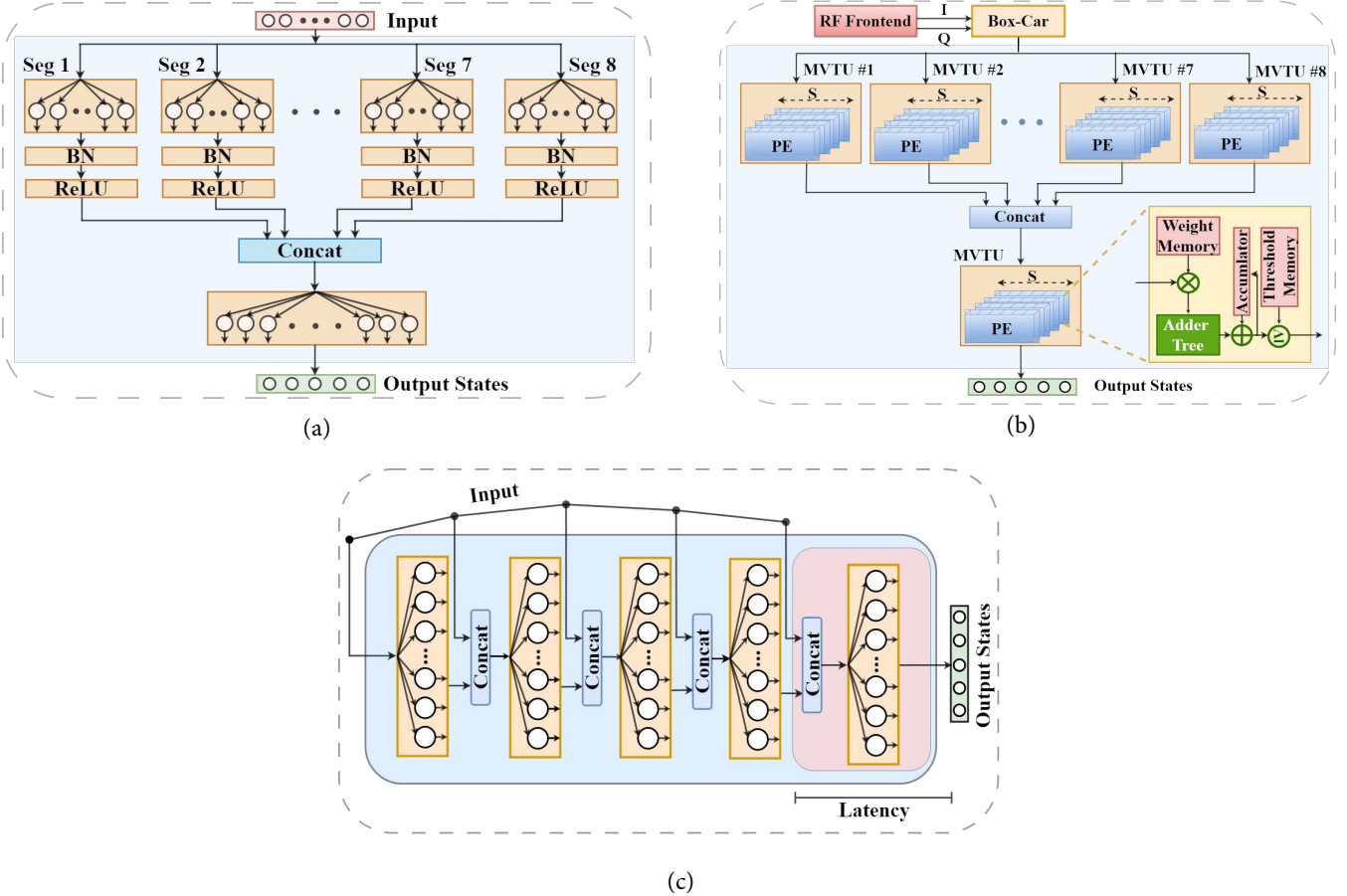


Fig. 4: Quantized neural network (QNN) archetypes. (a) Software model of the QNN $((512 \times 8) \times 8 \times 5)$ for achieving maximum parallelism. The first hidden layer consists of 8 equal segments (Seg 1 to Seg 8), each containing eight nodes in the *Linear* layer, followed by batch normalization (BN) and rectified linear units (ReLU). The output of each segment (1×8) is concatenated using the *Concat* layer, resulting in a size of 1×64 . (b) Fully parallel hardware architecture of the QNN. Each segment of the model shown in panel (a) is implemented as a multi-vector threshold unit, which runs in parallel on hardware. A processing element, consisting of various computation blocks, is shown in the inset. (c) Piecewise layered QNN architecture $256 \times 128 \times 128 \times 128 \times 128 \times 5$. Each layer processes a segment of the input signal along with the output from the previous layer. The input size for the first layer of the model is 1×256 , and subsequent layers receive an input size of 1×128 , composed of 1×64 from the previous layer and 1×64 from the input segment. The last section of the input signal is only fed to the last layer, meaning only the last layer contributes to the latency of the network.

IV. RESULTS AND DISCUSSION

A. Performance and Resource Utilization

To demonstrate the effectiveness of our approach, we implemented the NN model described in Ref. [13]. This network architecture consists of an input dimension of 1,000 nodes, three hidden layers with 1,000, 500, and 250 nodes, and 32 output nodes for each possible state. The total number of learnable parameters sums up to 1,634,782, each requiring 4 bytes of storage in floating-point representation. This translates to 50 MB of storage just for the weights and biases. Using this standard floating-point approach, the implementation exceeds the resource limits of most available FPGA devices, as reported in Ref. [14].

To address this challenge, we adopted the methodology described in the previous section to implement the model in Brevitas and convert it to a dataflow graph using FINN-

R. The input, weight, and activation quantization used are 4, 2, and 2 bits, respectively. The model achieves a geometric mean fidelity of 0.904, which is only 0.9% below the one reported in Ref. [13]. This reduction is within the acceptable range reported by other quantized implementations of standard models [25].

Latency and resource utilization comparisons for various NN model architectures and quantizations are presented in Tab. I. Reducing input size and hidden layers have a significant impact on resource usage and drastically improve latency with a minimal drop in fidelity.

The resource utilization of the base model (Arch-1) on the RFSoc XCZU28DR is only 39%, still leaving enough room for implementing other logic. Arch-4 to Arch-9 have almost the same parameters but vary in architecture and show a similar pattern in resource utilization, except for the binarized

TABLE I: Resource utilization and latency comparison of neural network archetypes.

Model Arch	# of Parameters	Quantization (IN/W/A)	F_{GM}	Resource		Max Freq (MHz)	Latency (Cycles)	Latency (ns)
				LUT (% util)	FF (% util)			
$1000 \times 1000 \times 500 \times 250 \times 32$ (Arch-1)	1,634,782	4/2/2	90.40	167054 (39.28)	135088 (15.88)	243	924	3802.26
$1024 \times 512 \times 256 \times 5$ (Arch-2)	657,413	4/2/2	90.39	58336 (13.72)	46654 (5.48)	275	336	1219.68
$512 \times 128 \times 32 \times 5$ (Arch-3)	69,957	4/2/2	89.85	38689 (9.10)	33358 (3.92)	261	51	195.33
$512 \times 64 \times 32 \times 5$ (Arch-4)	35077	4/2/2	90.11	38087 (8.96)	46064 (5.41)	350	40	114.4
$512 \times 64 \times 5$ (Arch-5)	33217	4/2/2	89.91	43959 (10.37)	53252 (6.26)	260	33	127.05
$512 \times 64 \times 5$ (Arch-6)	33217	4/1/1	80.1	18302 (4.30)	15197 (1.78)	440	17	38.59
$((512 \times 8) \times 8) \times 5$ (Arch-7)	33295	4/2/2	89.72	107026 (25.16)	60696 (7.13)	403	19	47.12
$256 \times 128 \times 128 \times 128 \times 5$ (Arch-8)	42124	4/2/4	89.78	110351 (25.94)	123514 (14.52)	341	11	32.23
$256 \times 128 \times 128 \times 128 \times 5$ (Arch-9)	42124	4/1/4	89.07	104527 (24.58)	95945 (11.28)	374	9	24.03

TABLE II: Latency comparison of discriminator using neural networks with the state-of-the-art.

	Discriminator	# of Parameters	NN Latency (ns)	Resources (LUTs)	Readout Type
Reuer <i>et al.</i> [22]	8-point Boxcar Filter + NN	1891	48	Not Reported	Single
Satvik <i>et al.</i> [14]	Demodulation + Matched Filter + NN	1112	Not Reported	17917	Multiplexed
This Work	2 point Boxcar Filter + NN (Arch-7)	33295	49.6	107026	Multiplexed
	2 Point Boxcar Filter + NN (Arch-9)	42124	26.7	104527	Multiplexed

model of Arch-6, which uses less than about 5% of the look-up tables (LUT)—programmable logic blocks in FPGAs. Arch-7, Arch-8, and Arch-9 exhibit higher utilization as these architectures have been optimized for higher performance.

The novel approach of splitting the first hidden layer of Arch-5 ($512 \times 64 \times 5$) into eight parallel segments results in Arch-7, reducing latency from 33 cycles to 19 cycles, an improvement of 42%. However, this comes with a significant increase in resource consumption, likely due to the additional *QuantIdentity* layers introduced before the *Concat* layer and the heightened level of parallelism. This architecture effectively balances complexity and performance. While binarized models are fast, they yield a significantly reduced fidelity. It is challenging to identify patterns in the maximum operable frequency of the design based on model architecture alone, as this is largely influenced by the Xilinx Vivado tool, which offers limited control over the outcome.

Additionally, we have implemented an ultra-low latency SVM-based state discriminator on FPGA, which is particularly useful for single-qubit state discrimination. This method offers an improved decision boundary compared to a matched filter while maintaining an ultra-light hardware footprint. Further details on the SVM implementation can be found in the Appendix B.

B. Comparison

As an alternative to traditional signal processing, recent work has utilized NNs for their robustness to signal perturbations and their capability for multi-qubit state discrimina-

tion [13], [14]. Maurya *et al.* [14] implemented a deep NN (DNN)-based discriminator for frequency-multiplexed qubit readout. Their approach uses a matched filter for dimensionality reduction before processing the data with a lightweight DNN. However, this approach introduces the additional requirement of qubit-specific demodulation and signal processing, which limits the solution’s scalability. The latency for this approach is not reported and has 1,112 learnable parameters.

Furthermore, Reuer *et al.* [22] describes a feed-forward NN designed for a reinforcement learning agent and state discriminator. This network features 7 hidden layers, each with 20 neurons, and is designed explicitly for single-qubit applications, generating decisions based on readout measurements. The approach has a latency of 48 ns and a total number of learnable parameters of 1,891.

Tab. II summarizes our networks’ latency, resource usage, and number of learnable parameters compared to the state-of-the-art. Our NN qubit-state discriminator, based on Arch-7, Arch-8, and Arch-9 implemented on FPGA, has latencies of 20 cycles, 12 cycles, and 10 cycles, respectively, including 1 cycle for the boxcar operation to provide input to the NN. Consequently, Arch-7, Arch-8, and Arch-9 achieve latencies (including boxcar operation) of 49.6 ns, 35.16 ns, and 26.7 ns, respectively, which are comparable to or better than the state-of-the-art. Meanwhile, the presented approach has 18 to 29 times more learnable parameters, making it more robust and versatile for complex readout scenarios.

In Refs. [14] and [13], the number of output nodes corresponds to the total number of possible state combinations

for N qubits, 2^N . However, using *BCEWithLogitsLoss* as the loss function, the number of output nodes can be reduced to N . This reduction makes scaling the proposed NN qubit-state discriminator scalable in the number of qubits.

Our methodology is scalable and can support the implementation of large DNNs, as demonstrated by our implementation of the model reported in Ref. [13] with 1.6 million parameters. By employing QAT, we have shown that even with 2-bit quantization, the model experiences only a minimal drop in fidelity while achieving ultra-low latency state discrimination, which is crucial for realizing quantum error correction (QEC), as evidenced by Arch-7, Arch-8, and Arch-9.

While Arch-7 effectively balances complexity and latency, some scenarios may require deeper networks. In such cases, Arch-8 and Arch-9 can achieve better latency, regardless of the network depth.

V. CONCLUSION

We present NN accelerators for state discrimination of frequency-multiplexed qubit readout traces of a superconducting multi-qubit processor. Our integrated approach deploys scalable NNs onto FPGAs, offering flexibility, automation, and faster design turnaround times. Adopting this methodology, we proposed ultra-low-latency architectures with latencies below 50 ns. The demonstrated substantial reduction in latency while maintaining qubit-state discrimination fidelity enables the implementation of QEC protocols with more error correction cycles, thereby significantly improving the performance of quantum processors. This marks the advent of low-latency NN architectures on FPGAs that do not require qubit-specific signal processing and can be scaled up as the number of qubits increases.

APPENDIX A

NN-ACCELERATOR DESIGN METHODOLOGY

All models were trained and quantized using PyTorch 1.12.1 and Brevitas 0.9.1. The activation function employed is the rectified linear unit (*ReLU*). We utilized the *Adam* [32] optimizer with a weight decay of 10^{-3} , a learning rate of 10^{-3} , and a batch size of 1024. During quantization-aware training (QAT), *Linear* and *ReLU* layers were replaced with their quantized equivalents, *QuantLinear* and *QuantReLU*, respectively.

A. FINN-R Flow

The conventional methods for implementing NNs on FPGAs typically involve either a hand-crafted custom architecture [22] or high-level synthesis (HLS) [14]. Both approaches restrict flexibility in selecting the NN's architecture and size. Additionally, custom architectures often entail long design turnaround times. Therefore, there is a need for an integrated approach that provides both automation and flexibility in implementing quantized neural networks (QNNs) on FPGAs.

Vitis-AI and FINN-R are leading automation frameworks for mapping QNNs onto FPGAs [33]. Both frameworks use QNNs and fixed-point arithmetic to generate hardware designs, but they differ in their implementation strategies. Vitis-AI

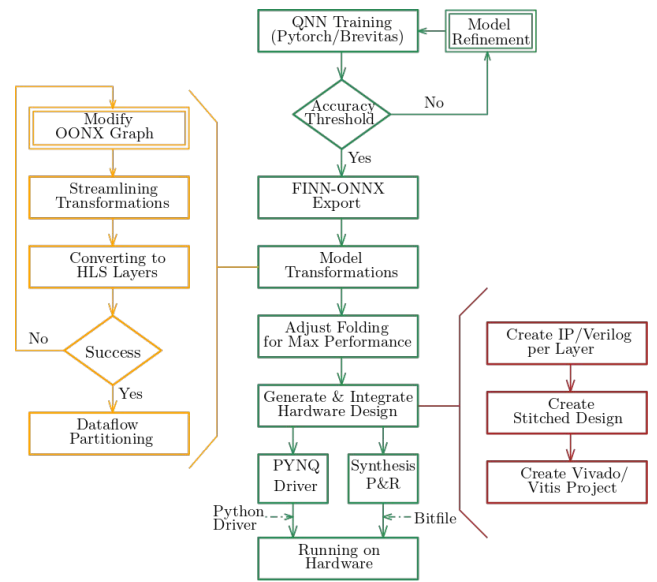


Fig. 5: Flowchart illustrating the end-to-end process of FINN-R. The double-bordered rectangles indicate modifications made to the default FINN-R flow.

employs an overlay-based approach, which is highly scalable due to its use of off-chip memory for model parameter storage. However, this approach may not achieve the same performance levels as dataflow-based architectures. In contrast, FINN-R utilizes a dataflow architecture and relies on on-chip memory, resulting in lower latencies than overlay-based implementations. Given the critical need for low-latency readout in quantum technologies, the dataflow framework provided by FINN-R is the preferred choice for our work. The end-to-end flow of FINN-R is illustrated in Fig. 5.

FINN-R processes models provided in open NN exchange format with FINN-specific metadata, which can be exported using the Brevitas library in a PyTorch environment. The framework transforms the model into a streaming dataflow graph and represents it in an intermediate representation. Nodes in the graph are then replaced with HLS-callable functions, and parallelism is either user-defined or derived based on throughput targets.

Quantization and matrix multiplication of low-precision data are handled by multi-vector threshold units (MVTU), with one unit per layer of the QNN. Each MVTU comprises multiple processing elements that operate similarly to single instruction multiple data architectures. Computation can be time-multiplexed to optimize hardware resource usage or dedicated processing elements and single instruction multiple data lanes can be employed for faster performance at the expense of higher hardware resource consumption. This flexibility allows for a balance between hardware resource usage and latency performance.

The final design is processed using the Xilinx Vivado backend, which generates Verilog code for the targeted FPGA device. The design is exported as intellectual property, enabling the modification and integration into other designs based on

specific requirements. The hardware architectures discussed are implemented on the Xilinx RFSoc ZCU111, with Xilinx Vivado 2022.2 used for design implementation. The synthesis strategy was optimized for high performance, with the primary goal of achieving low latency.

APPENDIX B SVM-BASED QUBIT-STATE DISCRIMINATOR

Common single-qubit state discriminators include the boxcar [34] and matched filter [35]–[37] with subsequent thresholding. Both methods are widely used for superconducting qubit platforms. A matched filter is generally preferred over a boxcar filter because it maximizes the signal-to-noise ratio. Moreover, boxcar filters are susceptible to additive stationary noise. Support vector machines (SVM) provide superior decision boundaries compared to boxcar or matched filters. Particularly, SVMs are more effective for state discrimination in single-qubit systems and frequency-division multiplexed readout, even when qubit-specific processing is involved [11], [13].

The incoming stream of I and Q data from the RF-ADC is digitally demodulated at each qubit-specific intermediate frequency (IF) over the readout integration time. The training dataset consisted of 1,000 random samples from each of 32 possible combinations, with a vector size of 512 for both IQ -data, corresponding to a $1\ \mu\text{s}$ readout trace duration.

The linear support vector classification (LinearSVC) package [38] was utilized to implement the SVM. After training, we derived floating-point weights and biases, which were applied to the test data. The floating-point SVMs outperformed their matched filter counterparts, improving qubit fidelity and achieving a 1.53% increase in the overall geometric mean fidelity of the five qubits.

TABLE III: Geometric mean readout fidelities F_{GM} of all five qubits using matched filters and SVM discriminators.

	F_{GM}
Matched Filter (Float)	0.8846
SVM (Float)	0.8982
SVM (2 multiplier + 8-bit quant)	0.8980
SVM (1 multiplier + 8-bit quant)	0.8985

The weights, along with the input and demodulation parameters, were quantized to eight bits. In the quantized implementation, we have two variants: one uses separate multipliers for demodulation and weight multiplication. At the same time, the other employs a standard multiplier that processes pre-computed fused values of weights and demodulation coefficients, reducing the number of multipliers by one per qubit. Tab. III compares the readout fidelities of this implementation with those of matched filters and various SVM approaches.

The hardware implementation of the SVM is illustrated in Fig. 6. The quantized SVM implementation, which optimizes weight and computation efficiency, outperforms other state-of-the-art traditional signal processing discriminators regarding latency and resource utilization. Tab. IV details a comparison with other leading methods.

Salathe *et al.* [40] proposed a method using demodulation and thresholding, achieving a core processing latency of 30 ns.

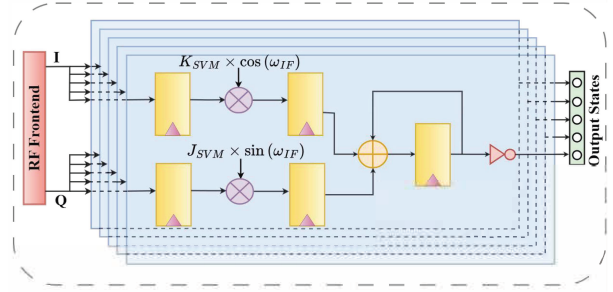


Fig. 6: Signal processing for SVM-based qubit-state discriminator. The hardware architecture is designed to handle the incoming eight-bit I and Q inputs from the RF-ADC. It performs multiply-accumulate operations throughout the readout integration time. The resulting processing latency corresponds to the time required to process the final input sample. The architecture includes five SVM modules, one dedicated to each qubit.

Tholen *et al.* [20] presented an integrated RFSoc solution with a matched filter state discriminator that uses pre-stored samples, resulting in a readout latency of 10 ns. Guo *et al.* [42] demonstrated demodulation and matched filtering with a readout latency of 24 ns. In contrast, our quantized SVM on FPGA achieves a discriminator latency of 5.74 ns and utilizes only 1675 LUTs for the five-qubit system, demonstrating significant performance and resource efficiency improvements.

ACKNOWLEDGMENT

Ujjawal Singhal acknowledges the support received through the Prime Minister’s Research Fellowship (PMRF), GoI. Pradeep Kumar Gautam thanks Ankita Nandi and Vignesh Ramanathan for valuable suggestions on the manuscript.

REFERENCES

- [1] S. Boixo, S. V. Isakov, V. N. Smelyanskiy, R. Babbush, N. Ding, Z. Jiang, M. J. Bremner, J. M. Martinis, and H. Neven, “Characterizing quantum supremacy in near-term devices,” *Nature Physics*, vol. 14, no. 6, pp. 595–600, 2018.
- [2] P. W. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer,” *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999.
- [3] L. K. Grover, “A fast quantum mechanical algorithm for database search,” in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 212–219.
- [4] M. Schuld, I. Sinayskiy, and F. Petruccione, “An introduction to quantum machine learning,” *Contemporary Physics*, vol. 56, no. 2, pp. 172–185, 2015.
- [5] P. J. O’Malley, R. Babbush, I. D. Kivlichan, J. Romero, J. R. McClean, R. Barends, J. Kelly, P. Roushan, A. Tranter, N. Ding *et al.*, “Scalable quantum simulation of molecular energies,” *Physical Review X*, vol. 6, no. 3, p. 031007, 2016.
- [6] R. Barends, J. Kelly, A. Megrant, A. Veitia, D. Sank, E. Jeffrey, T. C. White, J. Mutus, A. G. Fowler, B. Campbell *et al.*, “Superconducting quantum circuits at the surface code threshold for fault tolerance,” *Nature*, vol. 508, no. 7497, pp. 500–503, 2014.
- [7] D. A. Lidar and T. A. Brun, *Quantum error correction*. Cambridge university press, 2013.
- [8] G. Q. Ai, “Exponential suppression of bit or phase errors with cyclic error correction,” *Nature*, vol. 595, no. 7867, pp. 383–387, 2021.

TABLE IV: Latency comparison of SVM-based discriminator with the state-of-the-art.

	Discriminator	Processing Latency Multi-cycle (ns)	Processing Latency Single-cycle (ns)	# LUTs	Readout type
Xiang <i>et al.</i> [39]	Demodulation + Boxcar Filter	32	Not reported	Not Reported	Single
Salathe <i>et al.</i> [40]	Demodulation + Boxcar Filter	30	5.3	509	Single
Yang <i>et al.</i> [41]	Demodulation + Boxcar Filter	20	Not Reported	Not Reported	Single
Guo <i>et al.</i> [42]	Demodulation + Matched Filter	24	Not Reported	Not Reported	Single
Tholén <i>et al.</i> [20]	Demodulation + Matched Filter	10	Not Reported	Not Reported	Single
This work	Demodulation + SVM	5.74	3.67	1675	Multiplexed

- [9] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Physical Review A*, vol. 86, no. 3, p. 032324, 2012.
- [10] C. A. Ryan, B. R. Johnson, J. M. Gambetta, J. M. Chow, M. P. da Silva, O. E. Dial, and T. A. Ohki, "Tomography via correlation of noisy measurement records," *Physical Review A*, vol. 91, no. 2, p. 022118, 2015. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevA.91.022118>
- [11] E. Magesan, J. M. Gambetta, A. D. Córcoles, and J. M. Chow, "Machine learning for discriminating quantum measurement trajectories and improving readout," *Physical review letters*, vol. 114, no. 20, p. 200501, 2015.
- [12] R. Navarathna, T. Jones, T. Moghaddam, A. Kulikov, R. Beriwal, M. Jerger, P. Pakkiam, and A. Fedorov, "Neural networks for on-the-fly single-shot state classification," *Applied Physics Letters*, vol. 119, no. 11, p. 114003, 09 2021. [Online]. Available: <https://doi.org/10.1063/5.0065011>
- [13] B. Lienhard, A. Vepsäläinen, L. C. Govia, C. R. Hoffer, J. Y. Qiu, D. Risté, M. Ware, D. Kim, R. Winik, A. Melville *et al.*, "Deep-neural-network discrimination of multiplexed superconducting-qubit states," *Physical Review Applied*, vol. 17, no. 1, p. 014024, 2022.
- [14] S. Maurya, C. N. Mude, W. D. Oliver, B. Lienhard, and S. Tannu, "Scaling qubit readout with hardware efficient machine learning architectures," in *Proceedings of the 50th Annual International Symposium on Computer Architecture*, 2023, pp. 1–13.
- [15] A. Seif, K. A. Landsman, N. M. Linke, C. Figgatt, C. Monroe, and M. Hafezi, "Machine learning assisted readout of trapped-ion qubits," *Journal of Physics B: Atomic, Molecular and Optical Physics*, vol. 51, no. 17, p. 174006, 2018.
- [16] Z.-H. Ding, J.-M. Cui, Y.-F. Huang, C.-F. Li, T. Tu, and G.-C. Guo, "Fast high-fidelity readout of a single trapped-ion qubit via machine-learning methods," *Physical Review Applied*, vol. 12, no. 1, p. 014038, 2019.
- [17] Y. Matsumoto, T. Fujita, A. Ludwig, A. D. Wieck, K. Komatani, and A. Oiwa, "Noise-robust classification of single-shot electron spin readouts using a deep neural network," *npj Quantum Information*, vol. 7, no. 1, p. 136, 2021.
- [18] L. Stefanazzi, K. Treptow, N. Wilcer, C. Stoughton, C. Bradford, S. Uemura, S. Zorzetti, S. Montella, G. Cancelo, S. Sussman *et al.*, "The qick (quantum instrumentation control kit): Readout and control for qubits and detectors," *Review of Scientific Instruments*, vol. 93, no. 4, 2022.
- [19] K. H. Park, Y. S. Yap, Y. P. Tan, C. Hufnagel, L. H. Nguyen, K. H. Lau, P. Bore, S. Efthymiou, S. Carrazza, R. P. Budoyo *et al.*, "Icarus-q: Integrated control and readout unit for scalable quantum processors," *Review of Scientific Instruments*, vol. 93, no. 10, 2022.
- [20] M. O. Tholén, R. Borgani, G. R. Di Carlo, A. Bengtsson, C. Križan, M. Kudra, G. Tancredi, J. Bylander, P. Delsing, S. Gasparinetti *et al.*, "Measurement and control of a superconducting quantum processor with a fully integrated radio-frequency system on a chip," *Review of Scientific Instruments*, vol. 93, no. 10, 2022.
- [21] U. Singhal, S. Kalipatnapu, P. K. Gautam, S. Majumder, V. V. L. Pabbisetty, S. Jandhyala, V. Singh, and C. S. Thakur, "Sq-cars: A scalable quantum control and readout system," *IEEE Transactions on Instrumentation and Measurement*, 2023.
- [22] K. Reuer, J. Landgraf, T. Fösel, J. O'Sullivan, L. Beltrán, A. Akin, G. J. Norris, A. Remm, M. Kerschbaum, J.-C. Besse *et al.*, "Realizing a deep reinforcement learning agent for real-time quantum feedback," *Nature Communications*, vol. 14, no. 1, p. 7138, 2023.
- [23] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A survey of quantization methods for efficient neural network inference," in *Low-Power Computer Vision*. Chapman and Hall/CRC, 2022, pp. 291–326.
- [24] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. Van Baalen, and T. Blankevoort, "A white paper on neural network quantization," *arXiv preprint arXiv:2106.08295*, 2021.
- [25] M. Blott, T. B. Preußner, N. J. Fraser, G. Gambardella, K. O'brien, Y. Umuroglu, M. Leeser, and K. Vißers, "Finn-r: An end-to-end deep-learning framework for fast exploration of quantized neural networks," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 11, no. 3, pp. 1–23, 2018.
- [26] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *Journal of machine learning research*, vol. 18, no. 187, pp. 1–30, 2018.
- [27] A. Pappalardo, "Xilinx/brevitas," 2023. [Online]. Available: <https://doi.org/10.5281/zenodo.3333552>
- [28] J. Heinsoo, C. K. Andersen, A. Remm, S. Krinner, T. Walter, Y. Salathé, S. Gasparinetti, J.-C. Besse, A. Potočnik, A. Wallraff, and C. Eichler, "Rapid high-fidelity multiplexed readout of superconducting qubits," *Phys. Rev. Appl.*, vol. 10, p. 034040, Sep 2018. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevApplied.10.034040>
- [29] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.
- [30] W. Sung, S. Shin, and K. Hwang, "Resiliency of deep neural networks under quantization," *arXiv preprint arXiv:1511.06488*, 2015.
- [31] Xilinx, "AXI4-Stream Interconnect v1.1," https://docs.amd.com/v/u/en-US/pg035_axis_interconnect, 2024, [Online; accessed 20-Apr-2022].
- [32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [33] F. Hamanaka, T. Odan, K. Kise, and T. Van Chu, "An exploration of state-of-the-art automation frameworks for fpga-based dnn acceleration," *IEEE Access*, vol. 11, pp. 5701–5713, 2023.
- [34] P. Krantz, A. Bengtsson, M. Simoen, S. Gustavsson, V. Shumeiko, W. Oliver, C. Wilson, P. Delsing, and J. Bylander, "Single-shot readout of a superconducting qubit using a josephson parametric oscillator," *Nature communications*, vol. 7, no. 1, p. 11417, 2016.
- [35] G. Turin, "An introduction to matched filters," *IRE transactions on Information theory*, vol. 6, no. 3, pp. 311–329, 1960.
- [36] C. A. Ryan, B. R. Johnson, J. M. Gambetta, J. M. Chow, M. P. Da Silva, O. E. Dial, and T. A. Ohki, "Tomography via correlation of noisy measurement records," *Physical Review A*, vol. 91, no. 2, p. 022118, 2015.
- [37] J. Heinsoo, C. K. Andersen, A. Remm, S. Krinner, T. Walter, Y. Salathé, S. Gasparinetti, J.-C. Besse, A. Potočnik, A. Wallraff *et al.*, "Rapid high-fidelity multiplexed readout of superconducting qubits," *Physical Review Applied*, vol. 10, no. 3, p. 034040, 2018.
- [38] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler *et al.*, "Api design for machine learning software: experiences from the scikit-learn project," *arXiv preprint arXiv:1309.0238*, 2013.
- [39] L. Xiang, Z. Zong, Z. Sun, Z. Zhan, Y. Fei, Z. Dong, C. Run, Z. Jia, P. Duan, J. Wu *et al.*, "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*, vol. 14, no. 1, p. 014099, 2020.
- [40] Y. Salathé, P. Kurpiers, T. Karg, C. Lang, C. K. Andersen, A. Akin, S. Krinner, C. Eichler, and A. Wallraff, "Low-latency digital signal processing for feedback and feedforward in quantum computing and communication," *Physical Review Applied*, vol. 9, no. 3, p. 034011, 2018.
- [41] Y. Yang, Z. Shen, X. Zhu, Z. Wang, G. Zhang, J. Zhou, X. Jiang, C. Deng, and S. Liu, "Fpga-based electronic system for the control and readout of superconducting quantum processors," *Review of Scientific Instruments*, vol. 93, no. 7, 2022.

- [42] C. Guo, J. Lin, L.-C. Han, N. Li, L.-H. Sun, F.-T. Liang, D.-D. Li, Y.-H. Li, M. Gong, Y. Xu *et al.*, “Low-latency readout electronics for dynamic superconducting quantum computing,” *AIP Advances*, vol. 12, no. 4, 2022.