Fast Spin Qubit Readout using an RFSoC FPGA Platform

ET4300: Master's thesis

Sameer Khan

Delft University of Technology

TU Delft

Fast Spin Qubit Readout using an RFSoC FPGA Platform

by

Sameer Khan

to obtain the degree of Master of Science
in Electrical Engineering
at the Delft University of Technology,
to be defended publicly on Monday August 25, 2025 at 02:00 PM.

Student number: 5937469
Project duration: November 10, 2024 – August 25, 2025
Thesis committee: A. Chatterjee, TU Delft, Supervisor
M. Babaie, TU Delft, Thesis Advisor
J. S. S. M. Wong, TU Delft

Cover: Crop of a fully unrolled schematic of a Neural Network on FPGA
Style: TU Delft Report Style, with modifications by Daan Zwaneveld

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Acknowledgements

This thesis concisely summarises the research learnings and results of my educational, insightful and fun year-long time in the Chatterjee Lab. Joining and contributing to a research lab at its setup has been a learning and engaging experience. I express my deepest gratitude to my supervisor, Anasua Chatterjee, for providing me with the opportunity to experiment freely with the project she proposed. Her guidance and constant encouragement helped me navigate the conjoined mess of Vivado, OPX, FPGAs, spin qubits and measurement software.

The Chatterjee Lab has been an incredible team from whom I learned a multitude of things besides physics, electronics, and, of course, quantum computing. Thanks to Torbjørn for helping me get started with understanding readout electronics and software, which formed the basis of my thesis, to Miguel and Praveen for the help in understanding the plethora of equipment and processes involved in the lab, to Anton for further specific assistance with the spin qubit experiments and theory and finally to Rouven for the guidance with theory, machine learning which was new to me and analysis of the results. I, of course, cannot forget my fellow master's, Julian, who makes killer cappuccinos. The master's room was always fun with Rohan and Wiggert's continuous banter.

Two years in the Netherlands would have been very bland if not for all my friends. Somehow, all the group chats have ended up being related to food. I'm very grateful to all of them in 'Khana Khazana', 'Tomatos', 'Taco Tuesdays on weekends' and 'Family dinners'. Also, I cannot wait to meet all of my friends back home, whom I miss dearly.

I sincerely thank QuTech Academy for enabling me to pursue this master's degree without worrying about finances. The Vivado licences from the AMD University Donation program played a vital role in the project. I would also like to thank Raymond Vermeulen (QuTech's Electronics team) for help with the hardware, Sho Uemura (QICK) for answering questions about QICK and the RFSoC boards, and Giuseppe Guglielmo (hls4ml) for tips regarding QICK and hls4ml.

Finally, none of this would have been possible without the immense support from my parents and my brother. I couldn't be any luckier.

Sameer Khan
Delft, August 2025

Abstract

Fast and accurate readout of qubits is essential for feedback and error-correction protocols in scalable quantum processors. Current methods of readout in spin qubit processors require long times due to noise and drift present. The timing budget for feedback control and correction is further strained by the transfer of data to the host and back to the controller.

This thesis explores two complementary methods to speed up the readout: first, readout electronics integrated with programmable logic on the same boards reduce the latency of communication between the host and measurement setup, and second, a framework to reduce measurement times is devised by executing Machine Learning (ML) enhanced classification deployed on the same programmable logic.

We demonstrate the utility of the Radio-Frequency System on Chip (RFSoC) board for measurements of gate-defined semiconductor spin qubit devices. The design also incorporates an ML-accelerator for readout with a simulated latency of 58 ns for the classification of states.

This work enables the implementation of sophisticated readout techniques close to data acquisition electronics, which allows experiments that are not possible on conventional control electronics, which are either host-centric, requiring extensive calibration, or have block-box designs restricting hardware reprogrammability for specific experiments.

Contents

List of Figures

1

Introduction

Quantum computers, when sufficiently scaled, are poised to solve several computationally complex problems like large integer factoring [1], unstructured search [2], etc., while also enabling simulations of quantum physical phenomena [3], and solving optimisation problems [4], [5]. This was confirmed with the development of several algorithms, beginning in the 1990s, which started the drive to build a functional quantum computer. With the accumulated technological advancements of more than five decades since Moore's Law [6], semiconductor electronics design and development have reached immense scalability and reliability [7]. Building with this mature technology, backed by the promising properties of semiconductors themselves as platforms for quantum computers, presents a viable approach to realising quantum computers.

Operations in quantum computers, similar to computation in their classical counterparts, often involve starting in a defined state, its manipulation and subsequent measurement. The success or *fidelity* of these operations has improved significantly since spin qubits in semiconductors were first demonstrated [8], with recent results showing more than 99% fidelities in initialisation [9], single [10] and two-qubit [11] gate operations and measurement [9]. Practical applications of quantum computing, however, require thousands of logical qubits comprising millions of individually addressable physical qubits [12]–[15]. Logical qubits build fault-tolerant quantum processors by application of error correction protocols on a number of physical qubits encoding them [16], [17].

1.1. Motivation and Related Works

Developing a fault-tolerant quantum processor requires elements like fast readout, feedback and error correction, which themselves need fast readout. This is being approached from different directions, including:

- *Better materials* - less noise in enhanced materials leads to better Signal to Noise Ratio (SNR), giving better accuracy in shorter integration times [18].
- *Different readout mechanisms* - new mechanisms like latched readout and double latched readout already improve the fidelity and speed. Improved tuning strategies for sweet spot points also help in the improvement [19], [20].
- *Closer electronics*- the readout electronics can be moved closer to the processor for reduced trip delays [13], [21].
- *Fast inference using less data*- machine learning techniques can be used to infer the readout result from a smaller number of measurement results, leading to faster classification of states.

The use of Field Programmable Gate Arrays (FPGAs) for the digital signal processing of qubit readout signals and state-dependent feedback has been done for superconducting qubits regularly [22]–[26]. These systems enabled feedback based on a real-time digital demodulation of the dispersive readout signal. The move from analog feedback and signal processing to digital signal processing enhanced flexibility and reduced latency. Latencies as low as 219 ns for the complete feedback based on qubit state have been shown [22].

Application of Machine Learning (ML) to combat qubit readout errors has resulted in improved measurement accuracy [27]–[29]. Implementation of these solutions on hardware is resource-limited due to large models, which use significant resources. [30]–[32] have shown lightweight neural networks which are scalable and efficient for qubit readout.

The previous works require different systems for data acquisition (Arbitrary Waveform Generators, Digital to Analog and Analog to Digital Converters, Mixers, Oscillators, etc.) and data processing (Field Programmable Gate Arrays or host CPUs). With the increasing complexity of qubit devices, integrated control and processing solutions are needed. Radio Frequency System on Chip systems, which combine the signal generation and processing systems, have been utilised for quantum controllers [33], [34]. Removing the latency from the input signal (at the ADC) to the ML-assisted readout block or instrument will increase the readout speed additionally. [35] and [36] show the incorporation of these accelerators for superconducting qubits with the complete control and readout using the control firmware of [33] and [34], respectively.

The work done during this thesis focuses on utilising these techniques to classify spin qubit states using fewer data samples. A lightweight neural-network accelerated readout scheme is implemented on an open-source quantum control hardware-software system.

1.2. Outline

This thesis is organised as follows:

- *Chapter 2: Background*
  The background supporting the development of this work is described in this chapter, starting with spin qubits, the principles behind their measurement, both physical and electrical, and a discussion of the System on Chip (SoC) board utilised for the realisation of the methods developed in this work.
- *Chapter 3: Co-design of Neural Networks for fast readout*
  This chapter outlines the workflow developed to design neural networks capable of reading qubit states faster. The design decisions at each step and the constraints which needed to be met are discussed.
- *Chapter 4: Readout Accelerator*
  With the neural network firmware files generated by the workflow of Chapter 3, this chapter outlines the architecture of the readout accelerator block designed with these firmware files. After the simulation results of the accelerator IP (Intellectual Property) block are discussed, its integration in the qubit controller firmware is shown. Finally, the software modifications required in the different layers are discussed.
- *Chapter 5: Setup, Experiments and Results*
  This chapter discusses the different experiments done to validate the working of the RFSoC and QICK for spin qubit measurements.
- *Chapter 6: Conclusion and Outlook*
  This final chapter summarises the work done in this thesis, along with the contributions made during its undertaking period. The limitations and scope of further work are also discussed.

2

Background

Processing information using quantum systems, 'Quantum Computing', involves designing and working with numerous systems. Electronic or optical systems are used to define the quantum system. Additional electrical instruments are needed for control and manipulation. This chapter provides an overview of the background on how quantum systems are defined and studied in gate-defined quantum dots in semiconductor heterostructure materials.

2.1. Spins as qubits

To choose from the many possible quantum systems realisable in hardware, DiVincenzo listed five criteria to assess their suitability [37]. Semiconductor spin qubits proposed by Loss and DiVincenzo in [38] fulfil these criteria well.

1. *A scalable physical system with well-characterized qubits:*
   The spin of a single electron forms a natural and well-defined two-level quantum system, with the spin-up and spin-down states serving as the logical $|0\rangle$ and $|1\rangle$. These spin qubits are implemented in semiconductor materials using lithographically defined gates, forming quantum dots. This offers compatibility with existing fabrication technologies and scalability through multi-dot architectures.
2. *The ability to initialise the state of the qubits to a simple fiducial state:*
   The electron can be reliably initialised into a known spin state using spin-selective tunnelling or energy relaxation in a magnetic field, thereby preparing the qubit in a well-defined starting state suitable for computation.
3. *Long relevant decoherence times, much longer than the gate operation time:*
   Single-electron spin qubits exhibit long coherence times, particularly in materials with low nuclear spin density such as isotopically purified silicon. These long coherence times allow for the execution of many quantum gate operations before decoherence significantly affects the quantum state.
4. *A universal set of quantum gates:*
   High-fidelity single-qubit and two-qubit gates have been demonstrated in various spin qubit platforms, with reported fidelities exceeding 99%. These include electrically driven spin resonance for single-qubit rotations and exchange-based interactions for two-qubit entangling gates, enabling universal quantum control.
5. *A qubit-specific measurement capability:*
   The spin state of the electron can be read out using reliable techniques such as spin-to-charge conversion followed by charge detection with a nearby quantum point contact or single-electron transistor. These methods enable accurate projective measurement of individual qubits, satisfying the readout requirement.

Other types of spin qubits are based on trapping more than one electron to define a single qubit - singlet-triplet qubits use two spins, exchange-only qubits use three spins, with further variations based on the specific spatial and spin combinations of the electrons.

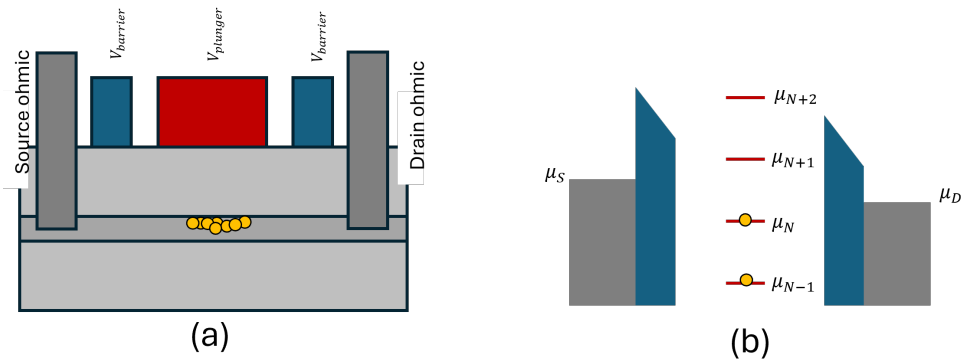

Figure 2.1: (a)Schematic representation of a quantum dot using a semiconductor heterostructure. The quantum dot is present under the red plunger gate, with blue-coloured barrier gates between the plunger and the reservoirs in grey. (b) Quantised energy levels of the charges trapped in the quantum dot. Adapted from [48]

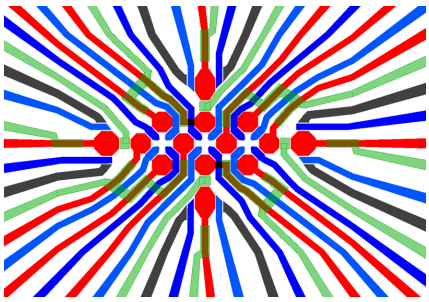

Figure 2.2: Schematic of the 10-dot device used in the experiments from [49]. The colour-coded gates are barriers: blue, plungers: red, reservoirs: grey and screening gates: green.

Around the same time, the first demonstrations of electron spin manipulations were shown [39], [40], theoretical studies indicated that holes in valence bands could implement spin qubits [41], [42]. Now holes have been studied in different materials [43] [44], [45], and hole spin qubits in Si and Ge have shown rapid progress in catching up to the state-of-the-art electron spin qubits [46].

2.1.1. Devices to trap and manipulate charge spins - Gate-defined Quantum Dots

Before the spins of electrons or holes can be used as qubits, a well-defined quantum system must be prepared. One such system is manufactured using semiconductor quantum dot devices.

When semiconductor materials with different band gaps are brought together by growing on top of one another (planar heterostructure material), it is possible to trap charges at the interface. These charges trapped in two dimensions form what is often called a two-dimensional electron (or hole) gas or 2-DEG (DHG). Using electrodes placed on top of the heterostructure, the electrostatic potential in the other two dimensions can be modulated to confine the charge carriers further. This congregation of a few charge carriers confined in all three dimensions is a quantum dot. The spatial confinement of charge carriers leads to the quantisation of their energies [47]. These discrete energy levels dictate the occupation and flow of charges via the quantum dot.

The gates used to create the quantum dots are shown in Fig. 2.1. The plunger gates (red) control the electrochemical potential of the dot and the number of charges it can hold. The barrier gates (blue) are used to change the confinement of the charges and also control the tunnel rates between the dots and any nearby reservoirs of charges. Reservoirs are regions of the 2DEG which have practically infinite charges. In the device design of Fig. 2.2, the ohmic contacts (grey) act as reservoirs since they are directly connected to the metal electrodes on the top of the device and can supply charges directly to the 2DEG. For a completely depleted device, the charges from these ohmics are the only charges in the 2DEG.

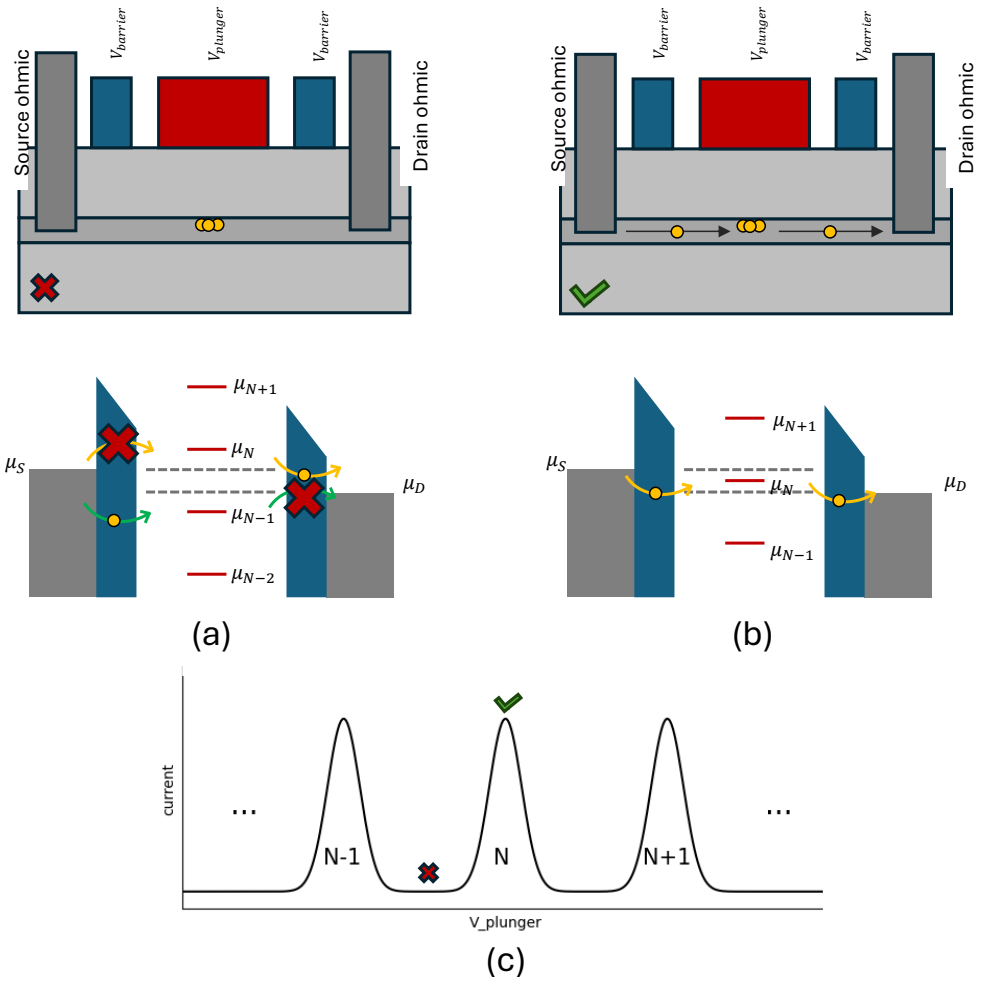

Figure 2.3: Quantised energy levels in the quantum dot and transport of charges across the dot. The colours are matched to the plunger, barrier and ohmics in Fig. 2.2. When none of the plunger's quantised energy levels lie between the reservoir levels (case (a) marked by ×), the current across the sensor drops as seen in (c). When one of the levels aligns(marked by ✓), there is an increase in the current corresponding to (b). Adapted from [48]

2.1.2. Sensing charges with current

The barriers, plungers, and reservoirs all change and control the occupancy of charges in quantum dots defined by the gates. Configuring or distributing charges across these quantum dots to utilise them effectively requires methods to determine their occupation.

Consider the sketch shown in Fig. 2.3 where a quantum dot is coupled to two reservoirs - a source and a drain. The reservoirs are connected via ohmic electrodes to the measurement equipment. In this arrangement, a quantised energy level of the dot aligned between the potentials of the source and the drain allows current to flow through the dot (Fig. 2.3(b)). However, if there is no dot state between the source and drain potentials, the current flow is blocked (Fig. 2.3(a)). This configuration of the energy levels is referred to as the 'Coulomb blockade' of the current. As the plunger gate voltage is swept, the dot's energy levels shift, aligning with the bias window and producing peaks in the measured current, as illustrated in Fig. 2.3(c)

The energy level of the sensor dot is defined by the number of its charges and its local potential landscape. The local potential landscape is influenced by the nearby electrode potentials and the occupancy of the nearby dots as well. Thus, the current across the sensor dot can be used to sense the nearby charge motions in different dots as seen in Fig. 2.4. This 'Charge sensing' in gate electrode-defined two-dimensional devices is based on the capacitive coupling of the other dots and gates to the sensor dot and each other [51].

The phenomenon of capacitive coupling between dots is not limited to the pair of a sensor and a dot. All

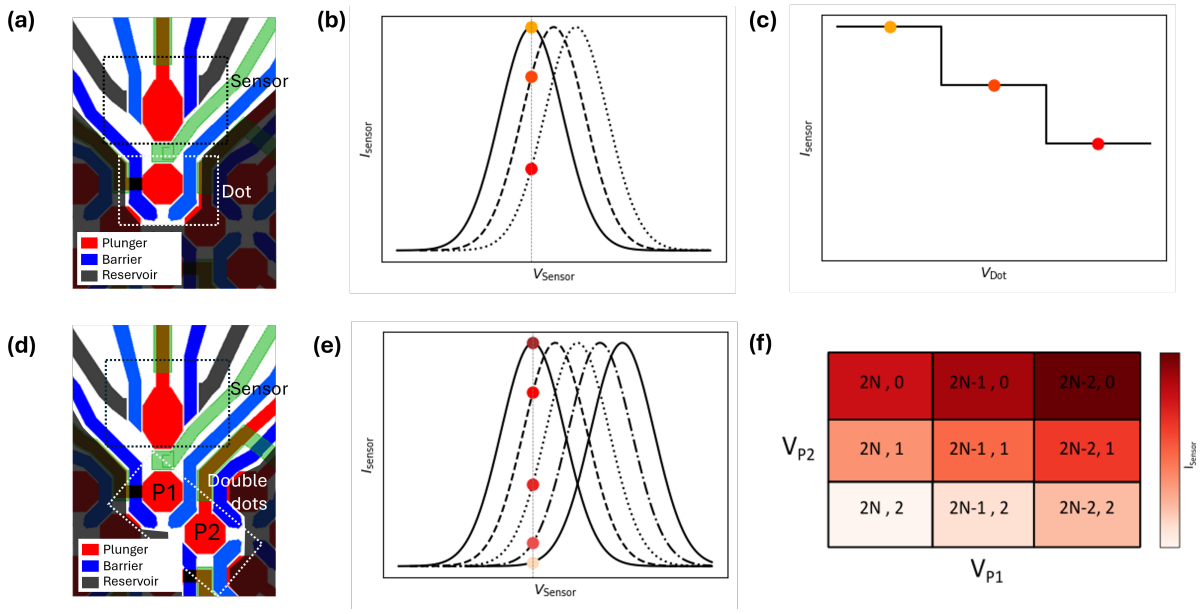

Figure 2.4: (a,b,c) correspond to sensing the occupation of one dot with the nearby charge sensor (a). When the voltage of the dot changes more than the addition energy (c), the occupancy of the dot changes, which capacitively affects the sensor plunger potential. As discussed in Fig. 2.3, this leads to a change in the current through the sensor (b). In (d,e,f), two dots are sensed using a single sensor again. Extending to sweeps of two gate voltages, the same principle applies - a change in occupation of dots under the plungers leads to a change in current via the capacitively coupled sensor. Adapted from [48], [50]

dots capacitively couple to each other [52], [53]. The effect of the barrier potential between two dots on their coupling is shown in Fig. 2.5. When the dots are not coupled to each other, changing the plunger potential only changes the occupation of the respective dot. With moderate coupling, changing a single plunger potential also affects the occupation in the coupled dot. When the dots are extensively coupled, they essentially form a single dot.

While the sensor dots are very sensitive to the charge occupation in the discussed devices due to their sharp transconductance (as seen in Fig. 2.3 (c)), measuring current across the sensor is limited by the small bandwidth of the circuit formed by the current amplifier and the capacitance of the cable to it. This limits the charge sensitivity of the sensor, which is the charge measured in the span of a unit time interval. Achieving a high SNR thus requires current measurements to be done over longer acquisition times.

2.1.3. Reflectometry for sensing

Radio Frequency (RF) reflectometry offers a fast and sensitive method for charge sensing in quantum dot and single-electron transistor (SET) systems [54], [55], overcoming the limitations of current-based measurements that require lengthy integration times. In particular, fast readout is essential in quantum information applications, where measurement times must be shorter than the characteristic timescales of quantum processes such as spin relaxation [19], [56], [57].

In RF reflectometry, a radio-frequency signal is applied to one of the sensor dot ohmic electrodes. The signal is injected through a transmission line and partially reflected due to the mismatch between the cable's characteristic impedance Z_0 and the impedance of the loading circuit Z_{load}, which is formed by the sensor impedance Z_S and the matching circuit comprised of the capacitor C_P, and the inductor L_C. (Fig. 2.6). The sensor impedance, in turn, depends on the potential landscape of the sensor and thus is used to sense charges close to the sensor.

The reflection coefficient Γ then is:

$$\Gamma = \frac{Z_{load} - Z_0}{Z_{load} + Z_0}$$

When the sample impedance Z_S changes, Z_{load} changes accordingly as per $Z_{load} = Z_S + j\omega L_C + \frac{1}{j\omega C_P}$ (ideally when there are no parasitic resistances or capacitances). This, in turn, changes Γ. When a voltage

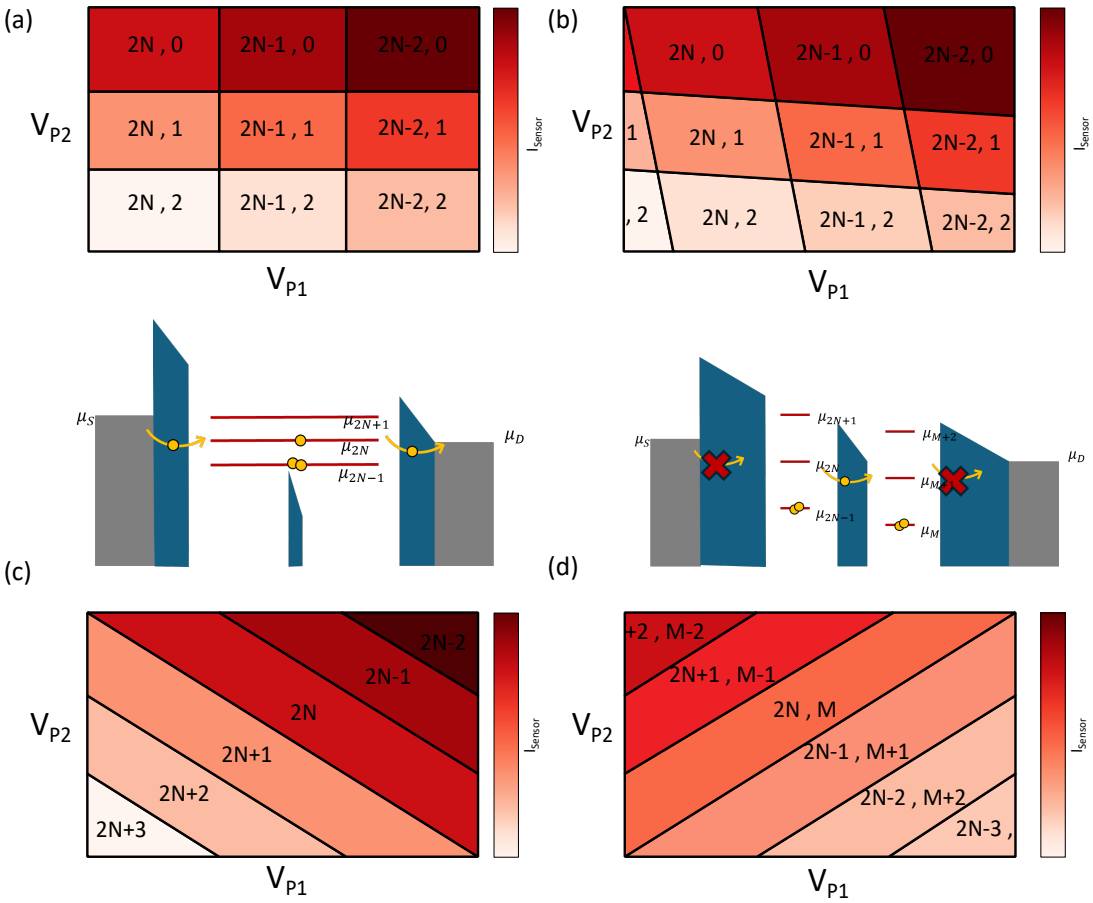

Figure 2.5: (a) Charge stability diagram for dots which are completely uncoupled. (b) With increased coupling, the effect of one dot plunger is visible on the measurement sweep of the other dot. (c) The result of an overly coupled pair of dots. These effectively form a single large dot. (d) With very large barriers separating the dots from the reservoirs, sweeping plungers move charges from one dot to another. Adapted from [48], [50]

signal at frequency ω is sent via the source $V_{in}(t) = V_A cos\omega t$, the complex reflected signal received at the receiver is $V_{rec} = \Gamma V_{in}$. Only the real part of V_{rec} is measured which is $V_{meas} = |\Gamma| V_A cos(\omega t + \angle\Gamma)$. This can be rewritten as

$$\begin{aligned} V_{rec}(t) &= |\Gamma| V_A \cos(\angle\Gamma) \cos\omega t - |\Gamma| V_A \sin(\angle\Gamma) \sin\omega t \\ V_{rec}(t) &= V_I \cos\omega t - V_Q \sin\omega t \end{aligned} \tag{2.1}$$

Where the quadratures V_I and V_Q are defined as

$$\begin{aligned} V_I &= |\Gamma| V_A \cos(\angle\Gamma) \\ V_Q &= |\Gamma| V_A \sin(\angle\Gamma) \end{aligned} \tag{2.2}$$

V_I and V_Q are also called the "in-phase" and "out-of-phase" components. The "IQ" quadrature representation thus encapsulates the load impedance in the reflection coefficient Γ. The IQ components are extracted from the received signal by demodulating with a mixer ($V_{LO} = cos\omega t$) and filtering with a low-pass filter. After the mixer, the signal is

$$V_{IF}(t) = V_{rec}(t) \cdot V_{LO}(t) \tag{2.3}$$

which gives

$$(V_I \cos\omega t \cdot \cos\omega t) + (-V_Q \sin\omega t \cdot \cos\omega t) = (\frac{V_I}{2} + \frac{V_I}{2}\cos 2\omega t) + (\frac{V_Q}{2}\sin 2\omega t). \tag{2.4}$$

The low-pass filter removes the 2ω frequency components, leaving the I quadrature component, $\frac{V_I}{2}$. Similarly the phase shifted $V_{LO}(-sin\omega t)$ gives the Q component:

$$(V_I \cos\omega t \cdot (-\sin\omega t)) + (-V_Q \sin\omega t \cdot (-\sin\omega t)) = \frac{V_I}{2}\sin(2\omega t) + (\frac{V_Q}{2} - \frac{V_Q}{2}\cos 2\omega t) \tag{2.5}$$

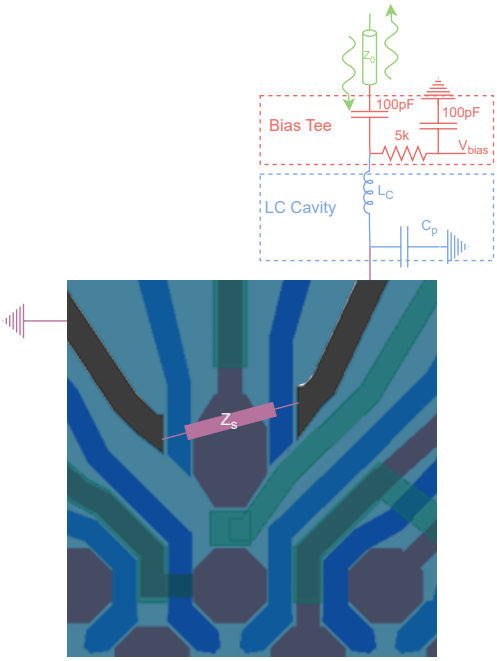

Figure 2.6: Reflectometry circuit representation. The signal is sent via the transmission line having characteristic impedance Z_0, the bias tee is used to set the DC bias across the reservoirs, and finally, the LC cavity is used to match the circuit impedance $Z_{load} = Z_S + j\omega L_C + \frac{1}{j\omega C_P}$ at the frequency of the signal.

The resulting $V_I/2$ and $V_Q/2$ from the low-pass filter are essentially instantaneous voltages and thus significantly prone to the noise from the preceding stages. Averaging these instantaneous outputs greatly reduces the effect of the prevalent white and $1/f$ noise and increases the signal-to-noise ratio (SNR) [58], [59].

$$\begin{aligned} I &= \frac{1}{T}\int_{t_0}^{t_0+T} \frac{V_I}{2} dt \\ Q &= \frac{1}{T}\int_{t_0}^{t_0+T} \frac{V_Q}{2} dt \end{aligned} \tag{2.6}$$

2.1.4. Measuring spin of charges

The spin of charge particles couples very weakly to the environment [47] [60]. This coupling, characterised by the magnetic dipole moment μ_B of the charge, is very small, which makes it hard to measure the spin even with extremely large magnetic fields ($E = \boldsymbol{\mu} \cdot \mathbf{B}$). This small coupling, however, also makes the spins more immune to noise from their environment, thus making promising qubits.

To sense spins, they are converted to charges via spin-to-charge conversion. This is achieved through the spin-selective transport of charges across a (2N,0) - (2N-1,1) transition as seen in Fig. 2.7. Pauli exclusion principle forbids the occupation of the same spin states of charges in one dot. This is utilised to find the spin state of the charges in the two dots by pulsing the dots across the transition from (2N-1,1) to (2N,0). If the singular charge, which is to be transported, has the same spin as the last charge present in the other dot, it is blocked. This shows up as an extending trapezoid in the (2N,0) region [61].

By encoding the spin state configurations as qubit basis states, they can be effectively read by Pauli spin blockade [59], [61].

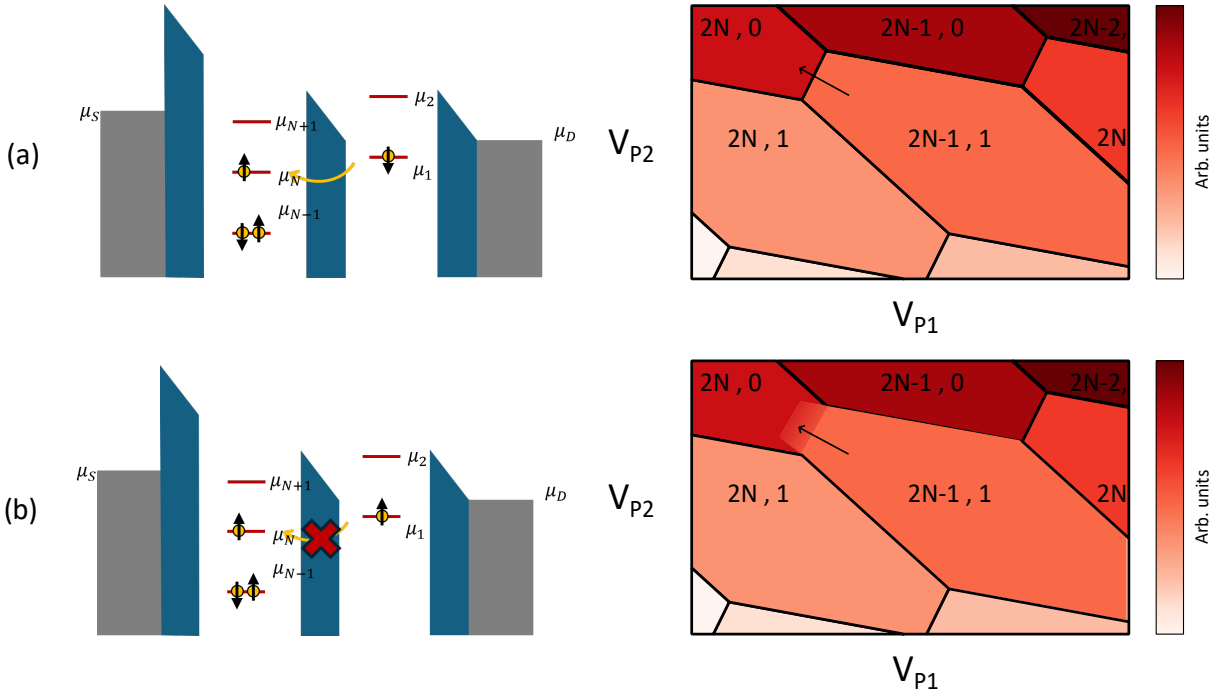

Figure 2.7: The spin state of charge in one dot is measured with the help of the spin in another dot, which is usually known. When the spin to be measured is the same as the known spin in the 'helper' dot, the Pauli exclusion principle prevents the charge from moving into the helper dot, even if it is energetically favourable, as shown in (b). This results in a blockade in the sensor response, which emerges as an extended trapezoid across the intended transition in the charge stability diagram. Conversely, when the spin states are different, as in (a), the charge can transition to the helper dot, and the sensor response corresponds to the new configuration.

2.2. Electronics for measurements

The manipulation and readout of quantum devices requires the use of classical electronic controllers to synthesise a large number of control and measurement signals. Historically, expensive general-purpose equipment involving complex setups of oscillators, Arbitrary Waveform Generators (AWGs), mixers and multiplexers in early setups in academic and commercial research labs. Fig. 2.8 (a) shows such a representative setup. The microwave RF pulse is generated by upconverting an envelope signal from an AWG Digital to Analog Converter (DAC) with a Local Oscillator (LO). This is then attenuated before getting reflected back by the matching circuit discussed earlier. The reflected signal power is directed to the reflection line, where it is amplified, downconverted and filtered before being sampled by an Analog to Digital Converter (DAC).

With the fidelity of state-of-the-art quantum processors increasing over the years, increasingly accurate electrical signal control is required so that the control signals do not become a bottleneck. Hence, tailor-made room-temperature controllers have been introduced by conventional electronic lab equipment makers, such as Keysight [62] and Zurich Instruments [63], and by dedicated quantum control electronic start-ups, including Quantum Machines [64] and QBlox [65]. The tailor-made controllers comprise FPGA-controlled DAC signal generators for the IF envelopes, which are upconverted using analog RF sources. These setups are still proprietary and expensive, making them unfeasible for small academic labs and startups.

2.2.1. RFSoC and QICK

DACs capable of synthesising RF signals directly have enabled FPGA manufacturers to integrate them along with fast sampling ADCs, FPGA logic and a microprocessor in the same package [66]. These affordable boards have been utilised by several academic laboratories as homemade quantum controllers [33], [34], [67], [68].

As shown in Fig. 2.8 (b), the SoC replaces the separate DAC, ADC, mixers, filter and the LO. Further, the calibration that needs to be done for these separate general-purpose instruments is significantly reduced

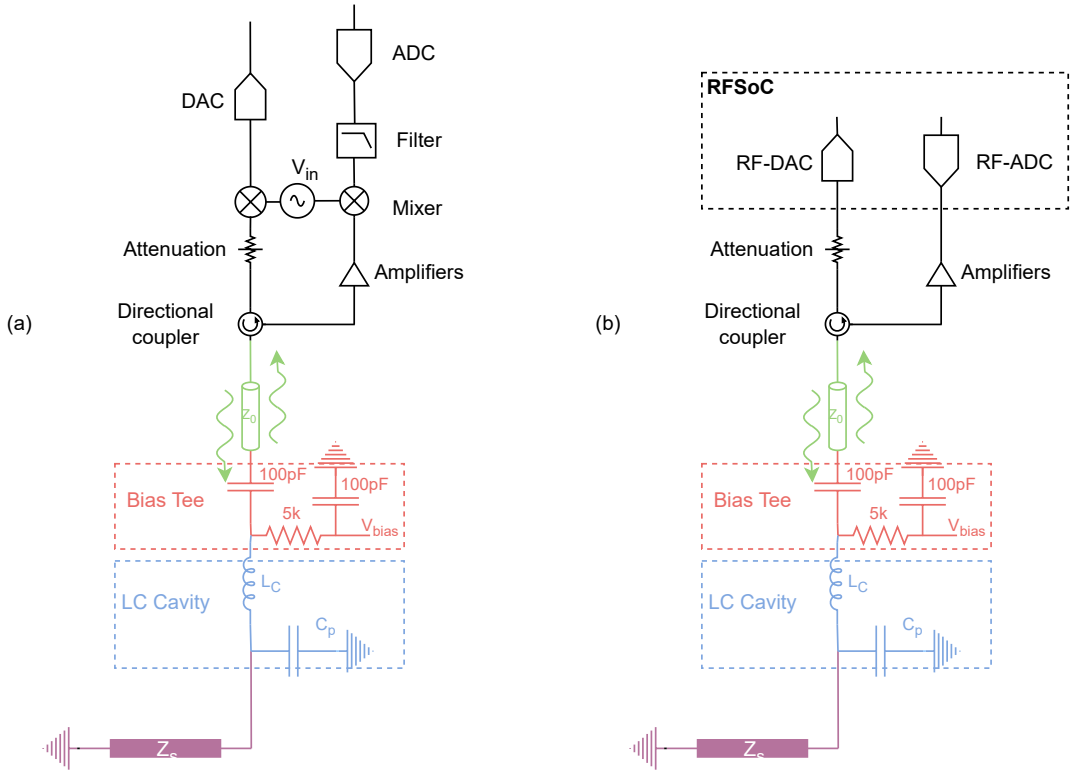

Figure 2.8: The reflectometry setup from Fig. 2.6 is updated with the electronics of the reflectometry chain. (a) shows the typical reflectometry setup with general-purpose instruments used in experiments. (b) shows the Radio Frequency System on Chip board, which replaces many of the general-purpose instruments used for reflectometry purposes.

by implementing them on a single board. The RFSoC 4x2 board from RealDigital [69] is shown in Fig. 2.9 with the RFSoC, DACs, ADCs, and the clocking chips (with the LOs) called out. Use of this board for RF reflectometry can be examined through the study of the QICK controller from [34], which is the only controller supporting this board from the controllers discussed before. The high-level schematic of the QICK design is shown in Fig. 2.10.

To send RF pulses, parameters are sent from the processing system (Zynq, which receives them from the host) to the signal generator blocks (QICK design). This block computes the I and Q components of the waveform, mixes them with the upconversion tone and passes the outputs to the DAC, which generates the excitation on the reflectometry input port of the setup.

The readout block does the readout of the reflected signal. Once the ADC samples the signal, it is digitally downconverted and filtered, saving the IQ quadratures for each time sample. The buffers (not shown in the schematic) save and average these to minimise the noise. The saved IQ values for a readout pulse are transmitted to the timed processor (tProc) and the host. All the timings to synchronise the waveforms are handled by the tProc.

While the DACs and ADCs are already implemented as blocks on the RFSoC, the tProc, signal generator, and readout blocks are custom designs. They are implemented as IP blocks and then used in the QICK block design (see Fig. A.1 for complete design). The block design is implemented on the FPGA resources available on the RFSoC (Table 2.1).

FPGA or Field Programmable Gate Arrays consist of arrays of configurable logic blocks (CLBs) or Slices, which are interconnected by programmable routing networks. Each slice in the Ultrascale+ RFSoC family of FPGAs has eight Lookup Table (LUT) and 16 Flip-Flop (FF) resources. The LUTs and Digital Signal Processing blocks (present separately from CLBs) are the computation resources, whereas FFs and Random Access Memory (RAM) blocks are the memory resources of the FPGA.

Combinations of these compute and memory resources, interconnected by routes, implement the

functionality of blocks - tProcessor, signal generators and readout. Hence, the total resources available on an FPGA affect the performance of the design. Designs utilising most resources can experience congestion in their processing pipelines due to the routing of a large number of slices, which creates long datapaths in the interconnects and results in signal delays and reduced timing margins. This effectively reduces the maximum achievable clock frequency and the throughput of the design.

Ensuring that resource utilisation remains within reasonable bounds, so clock frequency does not degrade in the design, will be a key constraint in the subsequent design work presented in the next chapter.

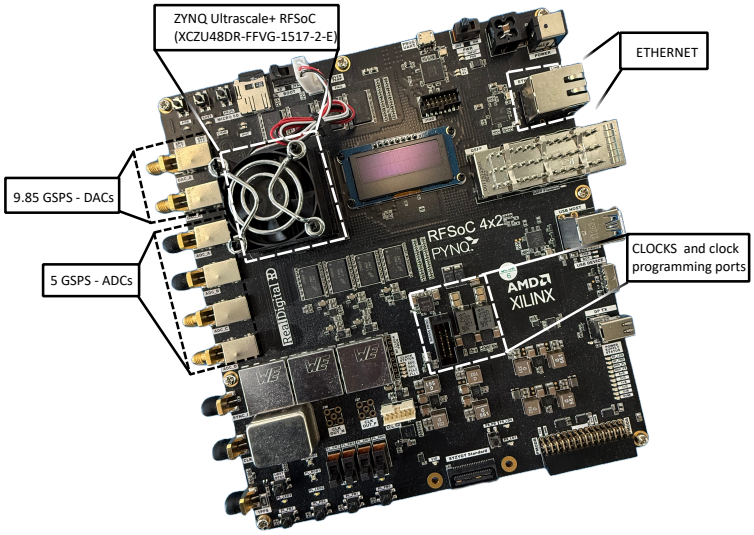

Figure 2.9: RFSoC 4x2 board with AMD/Xilinx XCZU48DR RFSoC (mounted under the fan assembly), 2 RF-DACs and 4 RF-ADCs mounted to output ports via decoupling capacitors and microwave baluns (underneath RF shields for isolation), ethernet port for host connectivity and programmable clocks

Resource	Availability
BRAM	2160
FF	850560
LUT	425280
DSP	4272

Table 2.1: Programmable logic resources available on the XCZU48DR. BRAM – Block RAM; FF – Flip-flops; LUT – Look-up table; DSP – Digital Signal Processing block.

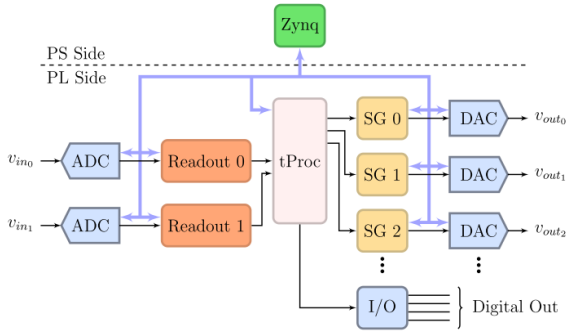

Figure 2.10: The firmware schematic representation of the QICK from [34], which shows the custom tProcessor (tProc), the Signal Generators (SG) and Readout blocks implemented on the FPGA logic. ADCs and DACs are Xilinx/AMD IPs. IOs are part of the SoC. The Zynq processor is responsible for communication between the controller on the logic and the host computer controlling the experiments.

3

Co-design of Neural Networks for fast readout

This chapter presents the co-design of compact neural networks with RFSoC hardware for fast single-shot readout of spin qubits. Simulated noisy data is used to generate labelled readout signal trajectories, which train and test a feed-forward fully connected neural network. With training and testing on the simulated signals, co-design is done with the constraints of FPGA resource availability and fixed-point representation in hardware. A resource-light and classification-accurate hardware design, which will be put in the quantum controller 'QICK', is obtained.

3.1. Neural Networks

Neural networks are machine learning models loosely inspired by how the human brain works. Nodes in a neural network are analogous to neurons in the brain. They learn about patterns in data via training and are then used for processing data in intended applications. They are often used in tasks which require classification, recognition, forecasting, etc.

There are numerous architectures for neural networks: feedforward neural networks (FNNs), convolutional neural networks (CNNs), recurrent neural networks (RNNs) and residual neural networks (ResNets), amongst many more [70]. The choice of an architecture depends upon the input types, output requirements, application and the resource availability for deployment of the application [71].

The classification we are concerned with in this work is of 1-dimensional time series data. This motivates us to work with feed-forward neural networks for their simplicity. Other networks, such as RNNs, which are designed to work with serial data [72], could be investigated in the future for implementation.

Fig. 3.1 depicts a simple feedforward neural network, also called a multilayer perceptron (MLP). It has an input layer comprising five nodes, two hidden layers with three nodes each and an output layer of two nodes. All the nodes from one layer are connected to all the nodes in the subsequent layer. This is an example of a fully connected feedforward network. It is called feedforward because during application or inference, when the network processes data, information flows only in one direction (forward).

Neural networks first have to be trained to make the required predictions. Labelled data, corresponding to the data that needs to be eventually classified, is fed to the network during training with initial random weights and biases. Neural networks process inputs by performing multiplication by weights and addition of bias for each neuron, followed by computing the result using an activation function. These activation functions are required to introduce a non-linearity in the overall network output. Two examples of such functions: Rectified Linear Unit (ReLU) and Softmax are shown in Fig. Training is typically done by defining a loss function, and its value is found for each classification performed on the training dataset. Then the network is traversed in reverse (back propagated) to adjust the weights and biases to minimise the loss (gradient of loss w.r.t. each weight and bias is checked, called gradient descent). Once training is complete, we have a trained neural network which can classify data accordingly.

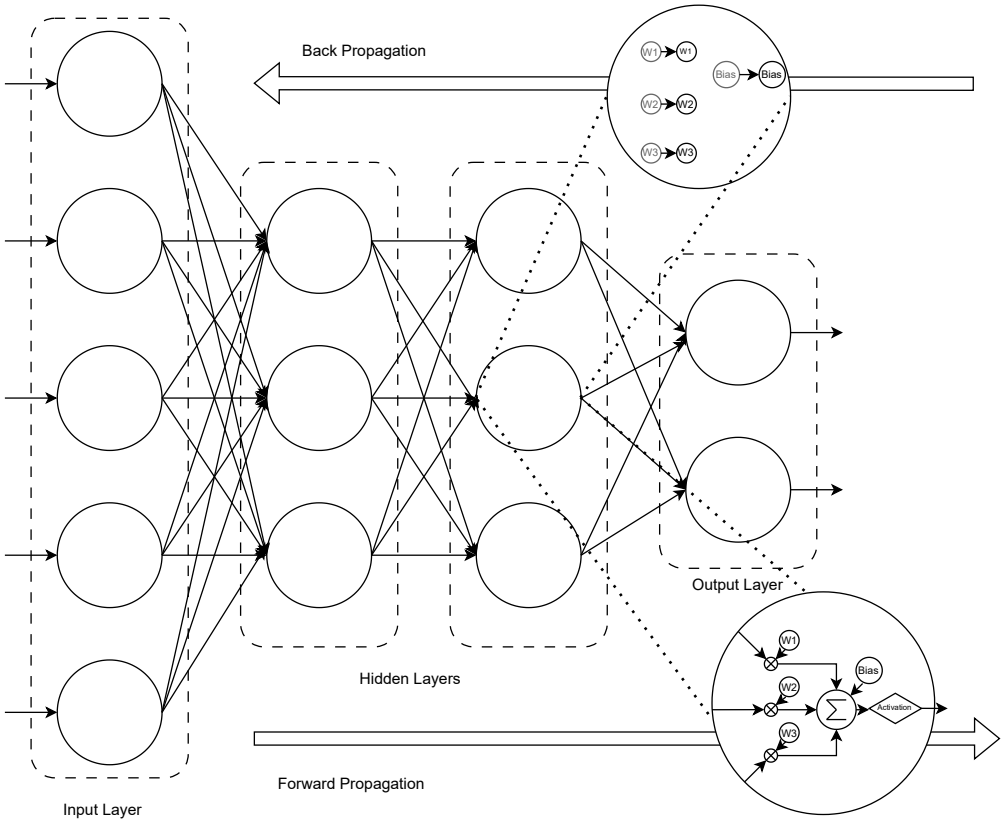

Figure 3.1: Schematic of a simple feedforward neural network (FNN). In forward propagation, each node multiplies the input values by the corresponding node weights, adds a bias and gives the corresponding value of an activation function as its output. During back propagation, the weights and biases are updated using gradient descent to reduce the loss incrementally.

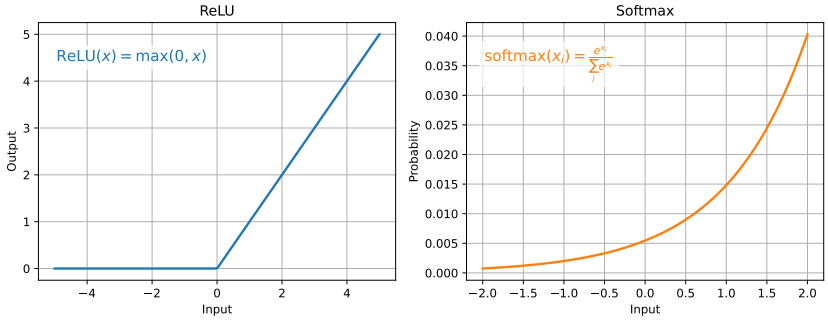

Figure 3.2: Activation function examples in Neural Networks. A ReLU and Softmax activation function outputs are shown with their expressions.

As discussed in 2.1.4, RF reflectometry is used to measure the state of a spin qubit. The microwave signal data corresponding to the qubit states is thereby used as the input for the neural network classifier. To design and test neural networks, a readout simulator for spin qubits was developed to study qubit dynamics and noise, which has been used to train a simple FNN discussed subsequently in the chapter. The simulator based on continuous measurement theory is discussed in the next section.

3.2. Readout signal simulator

Projective measurements of quantum systems collapse the state to the eigenstates. Instead of collapsing the state at once to retrieve state information, weak continuous measurements probe the system and extract the information partially over time [73]. This allows continuous evaluation of the system dynamics

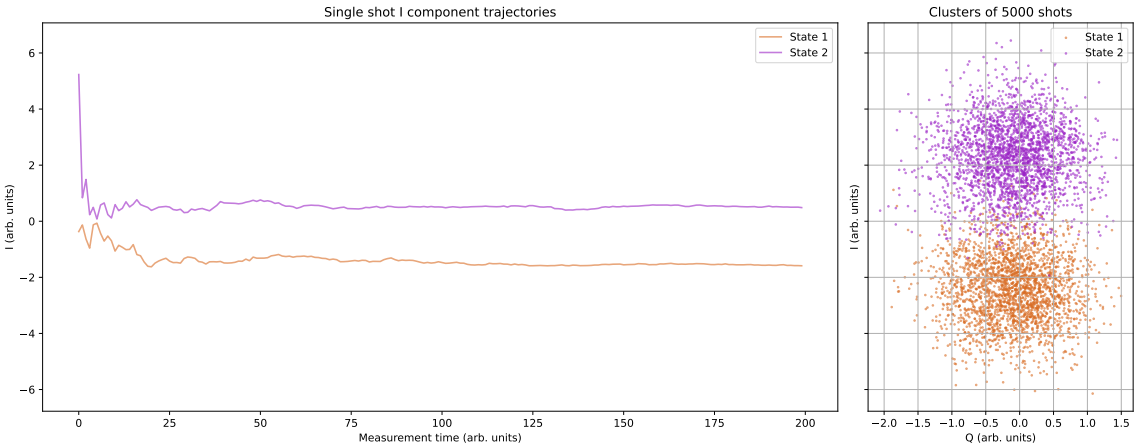

Figure 3.3: The simulated readout signals showing single-shot time traces for the two states. The mean clusters for 5000 shots are also shown.

of the measurement. This is especially useful for adaptive protocols, which are of interest to this work. From [73], the stochastic master equation framework gives way to a description of the state information current as a function of the state expectation, noise and the measurement rate and efficiency.

$$dI(t) = 2\sqrt{\eta\gamma}\,\langle\sigma_z\rangle(t)\,dt + dW(t) \tag{3.1}$$

dW is a Wiener increment or simply white noise increment, γ is the rate of measurement and η the measurement efficiency. The state of the system is encoded in the expectation value $\langle\sigma_z\rangle$. An observable time series can then be simulated by defining the system state and the noise increments. Two instances of time series single-shot trajectories, one for each of two system states, are shown in Fig. 3.3 along with the clusters showing 5000 such single shots averaged over the measurement time.

In the next section, the simulated readout data is used to train and test simple neural networks and deploy them on an FPGA.

3.3. Co-designing Neural Networks for FPGAs

FPGAs are great platforms for implementing neural networks. Architectures to process extensive data in parallel can be easily implemented and executed on FPGAs. This is especially ideal for neural networks, which involve multiple simultaneous matrix-multiplication-like computations. Additionally, the ability to implement different precision arithmetic, like 8-bit, 16-bit, etc., reduces latency, resource usage and power consumption.

In this work, we use FPGAs for implementing neural networks to reduce the latency of acquiring data from the device and processing it in near real-time. We have discussed in Chapter 2 the benefits of using RFSoC boards for replacing AWGs, mixers and LOs with a single system for simpler control and measurement setup. The FPGA in the RFSoC also allows for incorporating the neural network processing part in the same system, reducing the data transfer latency between different parts of the system. In addition to reducing the data transfer latency, the inference step of the neural network itself is faster on the FPGA parallel logic compared to serial CPU designs. This, however, can also be resolved by using GPUs for this step.

We utilise an open-source Python package `hls4ml` [74] to create Verilog firmware code for neural network models. The reasons for choosing hls4ml are two-fold. Firstly, support for hls4ml depends on its support for HLS backends like Vivado and Vitis HLS, Intel HLS, etc., which means broad support for different FPGA options. Second, HLS4ML was developed for processing high data rate processing involved in collision detection in the CERN Large Hadron Collider (LHC) [75]. Tens of terabytes of data are generated every second because of collisions occurring every 25ns. To determine if collision data needs to be saved, deep learning is employed on an FPGA with a 100 ns latency. Low-latency implementations are highly

Activation functions:
64 node hidden layer: ReLu
2 node output layer: Softmax
Training Parameters:
Optimiser: Adam
Learning Rate: 0.0001
Loss Function: Categorical Cross-Entropy
Batch Size: 1024
Epochs: 10

Table 3.1: Model configuration for 128x64x2 FNN.

relevant to the use case of discriminating qubit states, as the lifetimes of specific features are of the order of microseconds [22], [23], [31].

Models defined in Keras, PyTorch or ONNX can be passed to hls4ml, which first converts them to high-level synthesis code (based on C/C++). This can be converted to FPGA firmware using any of the supported HLS compilers - AMD/Xilinx, Intel/Altera, Catapult, etc.

3.3.1. NN model architecture

The first model we work with is a 128x64x2 FNN, which corresponds to a 128-node input layer, a 64-node hidden layer, and a two-node output layer, the two corresponding qubit states. To speed up the readout, fewer samples should be used to classify the state. With this objective, an arbitrary-sized input layer of 100 nodes was chosen. However, since the RFSoC ADC generates eight samples per clock, the input layer size was changed to 128 to use all the inputs available in integer-numbered clock cycles. Similarly, the size of the hidden layer is chosen arbitrarily in the first iteration, with updates based on feedback from the results further in the workflow. The complete parameter set of the 128x64x2 FNN is listed in Table 3.1. The network model is defined in Keras and trained on a 4000-shot dataset from the simulator described earlier, with each shot being a time series output 200 samples long.

The accuracy the model achieves on the test set of the data is 93.7%. It is then synthesised into the HLS model with the default fixed-point precision settings of hls4ml.

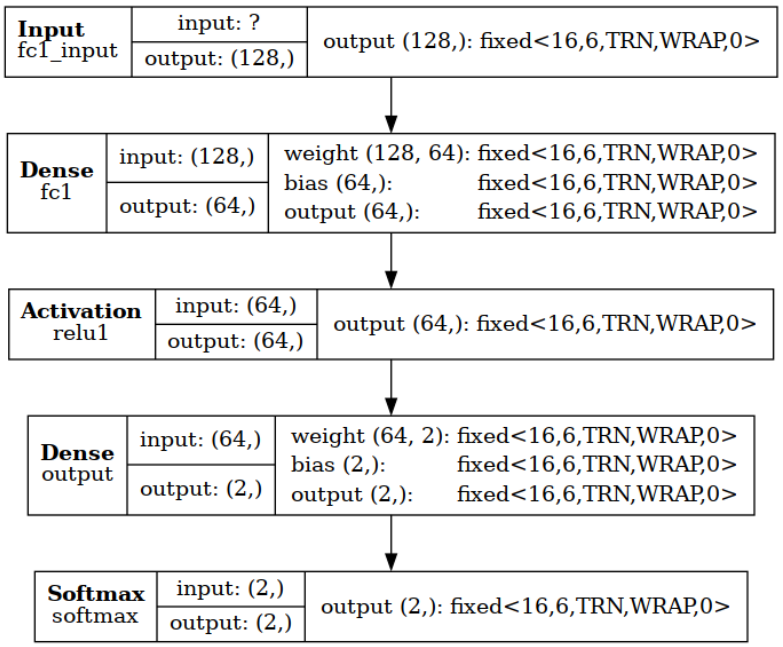

Figure 3.4: Visualisation of the HLS model architecture for 128x64x2 FNN. The floating point inputs, weights, biases and outputs are all converted to fixed point precision of <16,6,TRN,WRAP,0>. The "?" in the first layer is inconsequential, as the input layer merely defines the input dimensions that will be passed to the remaining layers.

The fixed-point precision allows the HLS (C++) simulations to match bit accuracy, quantisation and overflow behaviours of hardware. The HLS model architecture of the network is depicted in Fig. 3.4. Here fixed<16,6,TRN,WRAP,0> datatype identifier corresponds to a fixed-type 16-bit variable with six integer bits including the sign bit (and the remaining 10 bits signifying the fractional part), TRN means truncate towards minus infinity, i.e. discards the fractional bits beyond 10 and rounds towards negative infinity, WRAP corresponds to wrap around in case of overflow operations and the last zero is the number of saturation bits in overflow wrap modes. The various identifier fields and options are covered in the AMD HLS user guide [76]. The precision of each layer can thus be specified to fine-tune the fixed-point representation of the respective parameters of the layer.

The synthesised HLS model achieves an accuracy of 93.8%, which is quite close to the Keras model. The confusion matrices, which represent the performance of both models' tests, are shown in Fig. 3.5. The next step is using Vitis HLS to build the model firmware. This is wrapped in the build function of hls4ml model object. The resource usage and latency of the firmware are estimated using the Vitis tool reports before using the firmware files for FPGA block designs. For the 128x64x2 FNN, this resource usage is listed in the Table. 3.2.

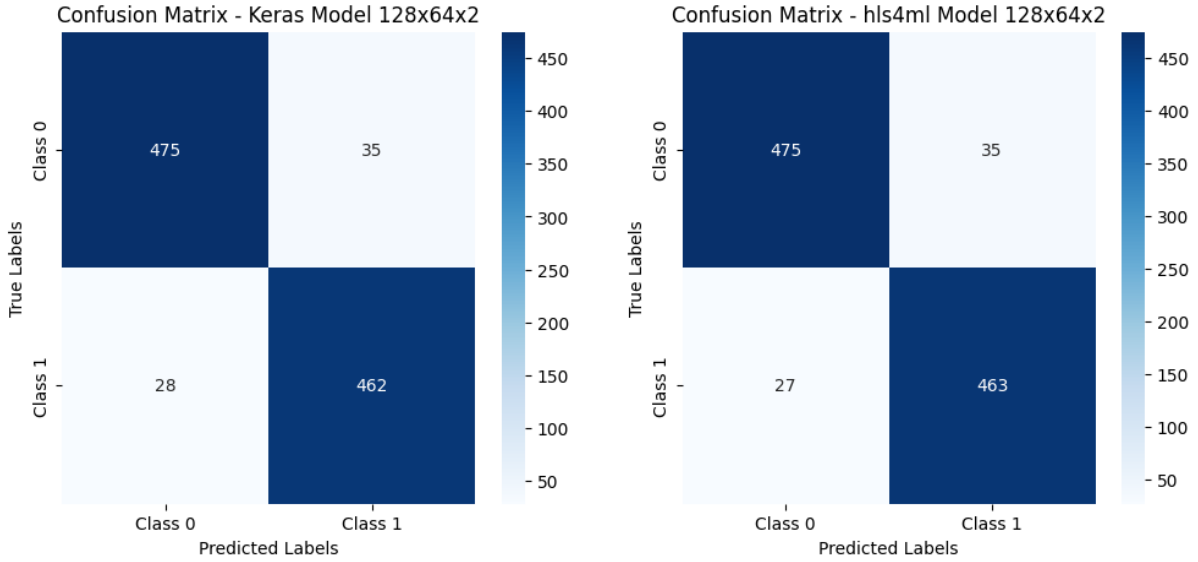

Figure 3.5: Confusion matrices of Keras and hls4ml synthesised model of 128x64x2 FNN. While the accuracies are above 93.5% for both tests, the excessive resource usage evident in Table 3.2 makes this model highly unfeasible.

Finer precision of the weights in the hidden layer leads to overutilisation of the DSP resources. To reduce the DSP resources used, we check if using a smaller network leads to a significant drop in accuracy. The hidden layer size is halved to 32, resulting in a 128x32x2 FNN, which is then trained and tested on the same dataset. The achieved accuracy is 93.3%, and the resource usage is listed in the Table. 3.3.

Name	BRAM	DSP	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	4
FIFO	-	-	-	-
Instance	2	5162	585762	237427
Memory	-	-	-	-
Multiplexer	-	-	-	36
Register	-	-	4163	-
Total	2	5162	589925	237467
Available	2160	4272	850560	425280
Util (%)	~0	120	69	55

Table 3.2: Resource Utilization: 128x64x2 FNN

Name	BRAM	DSP	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	4
FIFO	-	-	-	-
Instance	2	2688	306770	121560
Memory	-	-	-	-
Multiplexer	-	-	-	36
Register	-	-	3139	-
Total	2	2688	309909	121600
Available	2160	4272	850560	425280
Util (%)	~0	62	36	28

Table 3.3: Resource Utilization: 128x32x2 FNN

Further, a 128x16x2 network is also tested and reports an accuracy of 92.7%. However, as seen in Fig. 3.6, the network mislabels state one significantly more than state zero. Thus, a 128x32x2 FNN is chosen for

testing and as the subsequent baseline model.

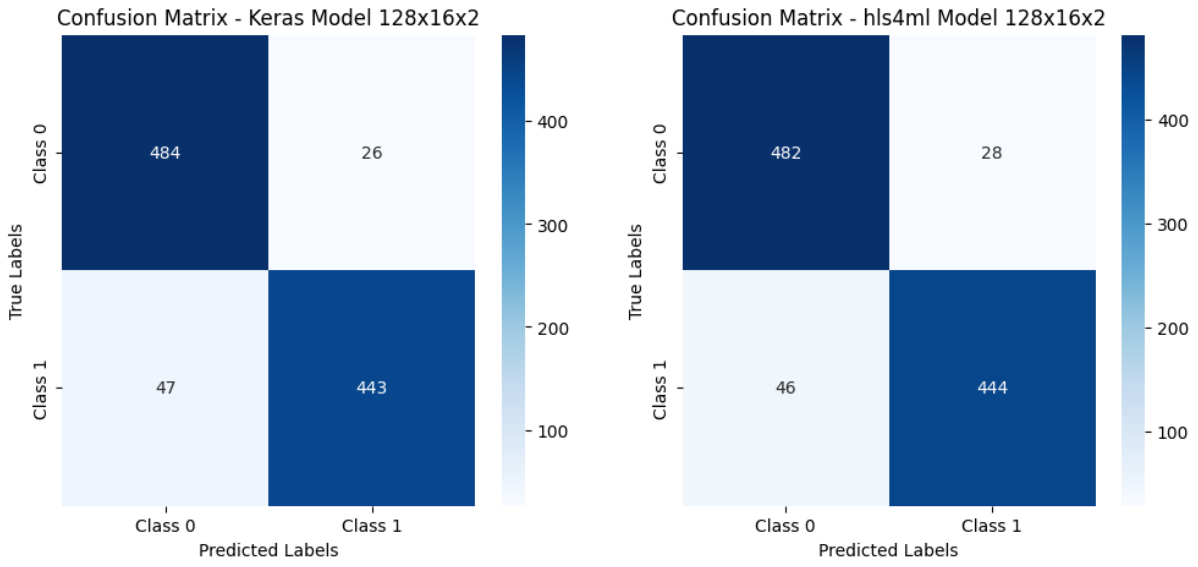

Figure 3.6: Confusion matrices of the Keras model for 128x16x2 FNN. State 1 is mislabeled as State 0 almost twice as much as State 0 being mislabeled as State 1. Due to this skewedness, this model is not used.

With the estimated resource utilisation within the limits of the available resources, we now explore different approaches to further reducing utilisation without letting accuracy drop drastically. Using fewer resources typically leads to faster placement and routing times, hence meeting the timing constraints and allowing for fast clock rate designs. For a neural network design, there are three methods which can be employed in different combinations to reduce the resource usage. The methods and their impact on the design are listed next.

1. *compression*, via pruning less important parameters (typically weights or sometimes entire neurons), which have minimal effect on the output, can be removed. This is useful for reducing the size of large networks, but for smaller networks, it does not significantly affect the size without compromising accuracy.
2. *quantisation*, by reducing the precision of the weights and biases in the network design, a significant reduction in resource usage can be achieved. This can be done to quite some extent without incurring a massive drop in accuracy [77].
3. *parallelisation*, low latency and high throughput are achieved by fully parallelising the multiplication and processing required in the layers' neurons. By reusing multiplication resources in a layer and pipelining the computations, the resources used for multiplication can be reduced at the cost of additional latency.

For the chosen 128x32x2 network, we did not find any improvement in the utilised resource reduction when applying pruning. This was predicted since it is an effective technique for compressing large networks and does not show improvements for small networks. We employ method two by quantising the Keras model with an extension for Quantisation Aware Training (QAT) of models - QKeras. This will be discussed in the following subsection. Reusing multiplication resources is not pursued in favour of generating a low-latency design. The results of method two also lead to a significant reduction in multiplication resources, eliminating the need to reuse them.

3.3.2. Quantisation of NN model

Resource utilisation by the neural network scales quadratically with the operation precisions [36]. Reducing the precision leads to performance degradation, which is addressed by employing QAT, which constrains the neural networks to operate within the precision limits. To design a quantised version of the previous FNN, the model is defined in QKeras with QDense and QActivation layers. QDense layer has quantised weights and biases, while the QActivation layer quantises the ReLu activation. QKeras currently has no quantised version of the Softmax function. The selected NN is then quantised with 6-bit

weights and biases as well as a 6-bit ReLu function. The network is trained on the same dataset with identical parameters as listed in Table 3.1.

When configuring the HLS model of the network with the QKeras frontend, hls4ml requires the fixed-point precisions to be specified at each layer. This is in contrast with the Keras configuration, where this is not required and the precision can be left at the model level.

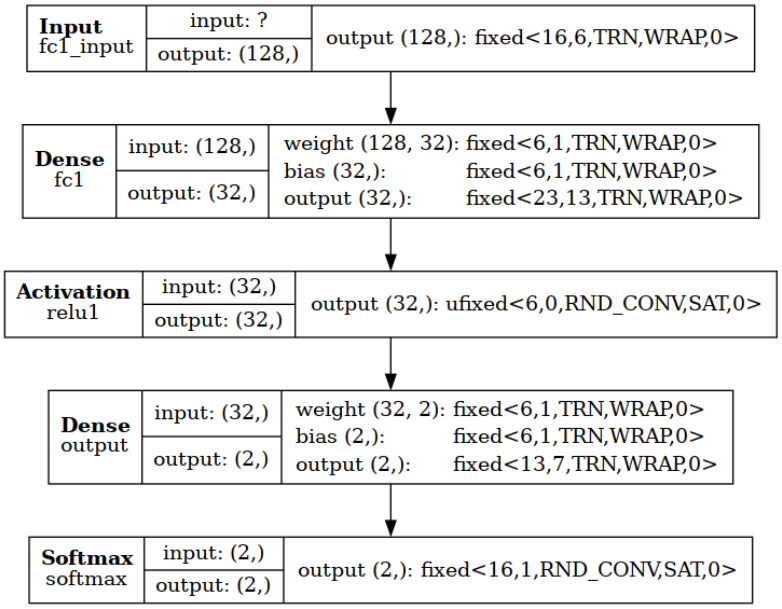

Figure 3.7: HLS model architecture of 128x32x2 QKeras model

Fig. 3.7 shows the layerwise precisions of the HLS architecture for the QKeras model.

1. The input layer in the FPGA block will receive the values from the RFSoC ADC, which has 14-bit resolution. However, the AXI Stream interface for the transfer of this data has a width of 16 bits. Thus, the input layer's precision is set to ap_fixed<16,6> to match the AXI Stream width. The integer part is chosen to accommodate the maximum absolute values in the dataset.
2. The Dense (QDense) layer involves multiplication of the inputs with the weights, followed by their accumulation and then addition of the bias at all the nodes. To accommodate the outputs of the dense layer, we find the maximum output of the above computation for ap_fixed<16,6> inputs multiplied by ap_fixed<6,1> and accumulated for 128 multiplications at each node. The ap_fixed<6,1> bias is added further. Thus, the integer part of the output is well represented by 5+7+1 (integer part of input + exponent of the multiplication by 128 + signed bit), i.e. 13 bits. The output of the Dense layer is then ap_fixed<23,13> by keeping the fractional bits the same.
3. The Activation layer, which follows, ReLu, maintains the same precision as specified in the training, ufixed<6,0>. The ReLu layer has the unsigned bit precision due to the ReLu function itself being non-negative. It should be noted that this is not a leaky ReLu layer, which can have negative outputs.
4. The precision of the next layer outputs, the Output layer, can similarly be found as the Dense layer. With zero integer bits in the input and 32 inputs multiplied by 6-bit signed fractional weights, the output layer can be resolved with ap_fixed<13,7> precision. An additional integer bit is added to account for the addition of the bias. This was not done in the previous layer because the representation of 14-bit ADC inputs with 16-bit representation already had additional bits.
5. Finally, the precision of the Softmax layer is set to ufixed<16,0> to match the outgoing AXI Stream and the range of the Softmax function, i.e. [0,1].

The HLS model achieves an accuracy of 93.2% on the dataset, which is quite close to the 93.3% accuracy of the Keras model with the same architecture. This model is now synthesised with Vitis HLS to generate the firmware files, which will be used for implementing the readout accelerator block in the next Chapter. The estimated resource usage of this block is listed in Table 3.4. Furthermore, Vitis HLS also provides

latency estimates for the block when the targeted clock period provided is 1.8 ns, the clock period of the QICK readout blocks. These are listed in Table 3.5.

Table 3.4: Resource Utilization Estimates for Quantised 128x32x2 FNN

Name	BRAM	DSP	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	4
FIFO	-	-	-	-
Instance	2	2	102028	114198
Memory	-	-	-	-
Multiplexer	-	-	-	36
Register	-	-	2847	-
Total	2	2	104875	114238
Available	2160	4272	850560	425280
Util (%)	~0	~0	12	26

Timing

Clock	Target	Estimated	Uncertainty
ap_clk	1.80 ns	1.311 ns	0.49 ns

Latency

	min	max
Latency (cycles)	32	32
Latency (absolute)	57.600 ns	57.600 ns

Table 3.5: Timing and Latency Estimates for Quantised 128x32x2 FNN

3.4. Workflow developed

This chapter describes the workflow developed to co-design an NN model for implementation on the RFSoC. This workflow can be summarised in Fig. 3.8. The final model built from the Vitis HLS implementation outputs both Verilog and VHDL firmware files, which will be utilised in the next chapter to create the custom readout accelerator IP, which will be subsequently integrated with the QICK design.

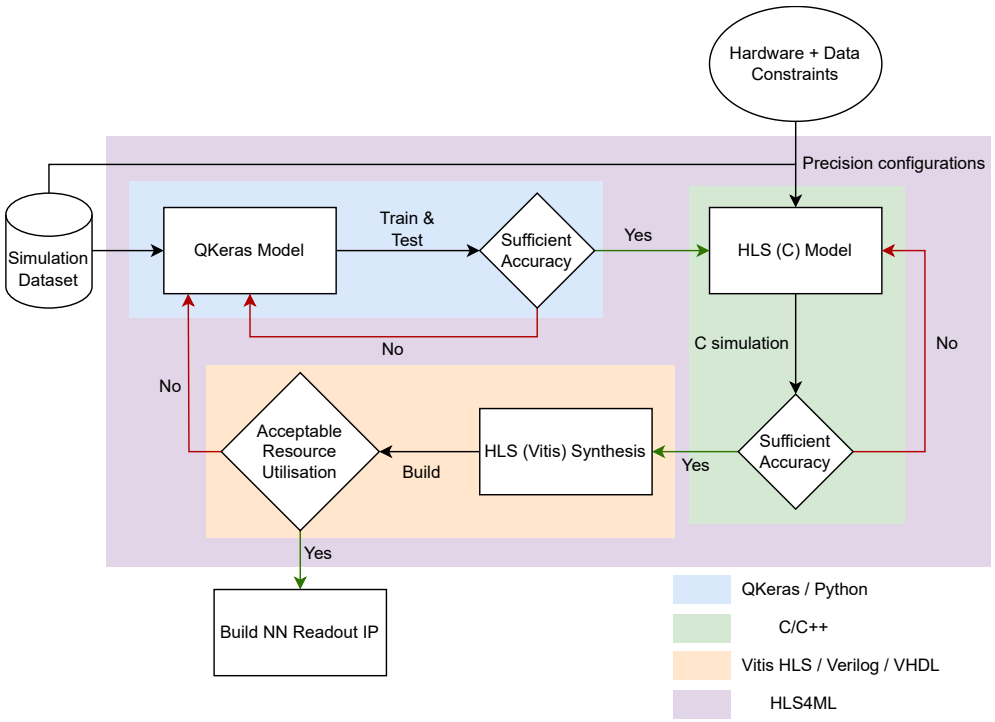

Figure 3.8: Workflow for co-designing a NN for FPGA implementation. Co-designing involves setting the precision of NN parameters constrained to the hardware limits and compressing the network to occupy as few FPGA resources while maintaining sufficient accuracy.

4

Readout Accelerator

In Chapter 3, the firmware files for a 128x32x2 quantised FNN were generated by co-designing it with HLS4ML. This chapter focuses on the design and simulation of a readout accelerator block built around the FNN and its integration with the rest of the QICK firmware. Drivers for accommodating the block in the controller software libraries are added.

4.1. Design of NN-accelerated readout block

The firmware files generated using hls4ml in the previous chapter cannot be directly plugged into the QICK design. First, an RTL wrapper for the FNN module is created to bridge module signals to the AXI Stream signals. The FNN input size is 128 samples, while the ADC streams eight samples every clock cycle. Initially, a simple shift register-based serial-to-parallel converter is implemented to save eight samples for 16 clock cycles and subsequently stream the 128 samples to the FNN block. This was replaced in the later designs with the AXI4-Stream Data Width Converter IP from Xilinx, which is better suited to deal with AXI Stream protocols. Slave and Master AXI Stream interface ports are connected to the respective ports to finish the block design of the NN-accelerated Readout block. This block design is shown in Fig. 4.1. An HDL wrapper for the block design is created to facilitate its instantiation in simulation and export as an IP for subsequent use.

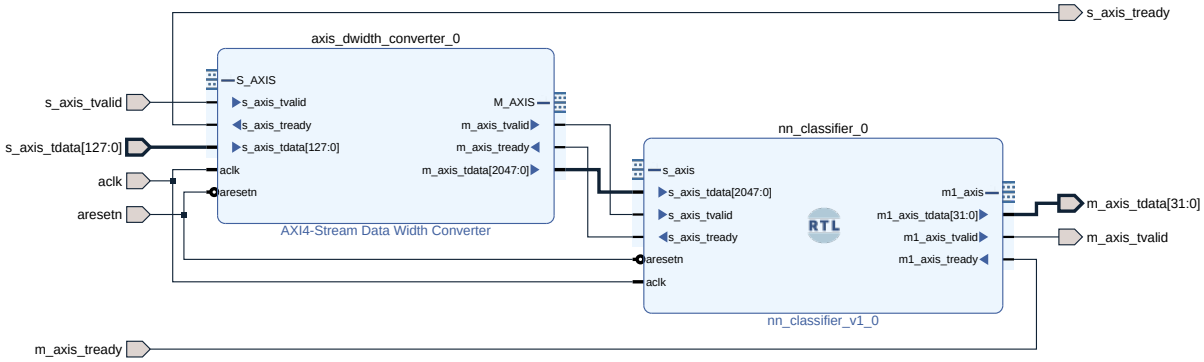

Figure 4.1: Vivado block design of the NN accelerated readout block. An AXIS Datawidth converter is connected to the FNN RTL wrapper. AXIS interface ports are added to facilitate generating a custom IP for use subsequently. *tvalid*, *tready*, and *tdata* ports for both slave and master axis are used for compatibility with the complete block design of QICK.

4.1.1. Verilog simulation of the block

To ensure consistency and direct comparison amongst the QKeras, HLS and HDL implementations of the neural network, the same test data is used in the behavioural simulation of the NN block generated previously.

A test bench is created, which reads the serial test data eight samples at a time from an input text file, streams it using a 1.8 ns period clock and writes the output into another file. Fig. 4.2 shows the simulation waveform. The timings of different events are marked. The first marker signals the start of streaming in of a new set of samples. When 128 samples are collected, the master axis *tvalid* bit of the data width converter is driven.

As expected, this latency is 16 clock cycles, i.e. 28.8 ns, as seen from the second marker. The latency of the *nn_classifier* block is 86.4 - 28.8, i.e. 57.6 ns, which corresponds to 32 clock cycles. This is in agreement with the estimated latency obtained from Vitis HLS earlier. Further, since the readout block processes data serially, the throughput is $1/latency$ of the block, which is $1/86.4GSPS$ or 11.57 MSPS.

Since the test data contains 1000 sets of 128 samples, 16 bits each, manual validation of the outputs from the simulation waveform is not feasible. The output file is parsed to evaluate the performance of the classifier using a confusion matrix. Along with the confusion matrices from QKeras test and HLS C simulation, this confusion matrix is shown in Fig. 4.3. The accuracy achieved in the behavioural simulation is 93.3 %, almost identical to the accuracy of the preceding results. Further, the respective class labels are also quite close. This sufficiently proves the design to function as expected on the FPGA. After exporting the IP block, integration in the QICK block design is done.

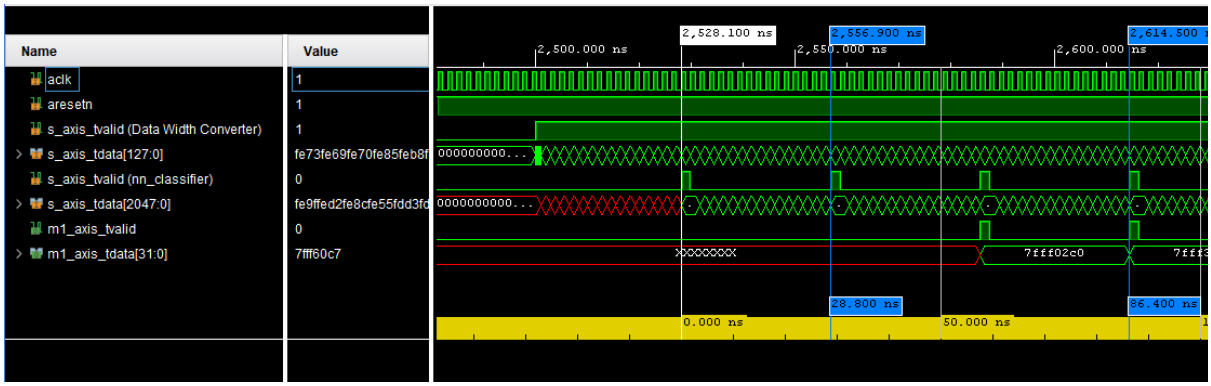

Figure 4.2: Behavioural simulation of the NN accelerator with a custom testbench which feeds the same test data as Keras and C/HLS tests. The latency of the classifier is 32 cycles, whereas the throughput of the complete accelerator is 11.57 outputs per second.

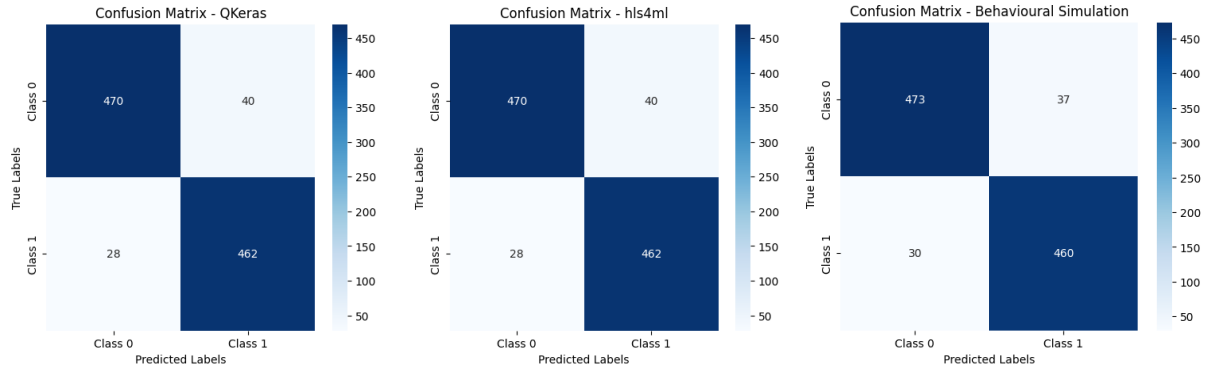

Figure 4.3: Confusion matrices for QKeras, hls4ml model, and the behavioural simulation in Vivado for the FNN 128x32x2 Neural Network design.

4.2. Integration of NN-accelerator in QICK

The separate design and integration of the IP follows the loosely coupled model for hardware accelerators [78]. This approach allows additions to the complete design of QICK without needing to co-design the processor and other cores every time a new type of accelerator block is added or updated.

4.2.1. Design changes required

The requirements for the integration and usage of the accelerator IP design in this thesis are listed below:

1. Raw ADC data must be accessible to the demodulation-based readout block (*axis_readout_v2*) and be saved directly for training and testing purposes.
2. Demodulated I and Q components from the *axis_readout_v2* block should be available at the accelerator IP.
3. Output of the IP should be available at the timed processor as well for implementation of feedback-based protocols.

Based on the above requirements, a conceptual schematic is presented in Fig. 4.4. These changes have thus been incorporated into the block design of QICK with the adjustments required to account for the data bandwidths of the block input and outputs. The complete block design is shown in Fig. 4.5.

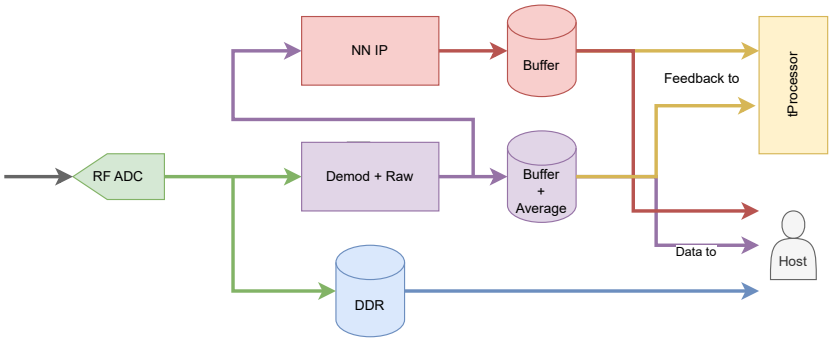

Figure 4.4: Schematic showing the proposed design modifications to incorporate the NN IP. ADC data is broadcast to the demodulation-based readout block, and a large DDR memory for training purposes. The demodulated outputs from the readout block are fed to the NN-IP.

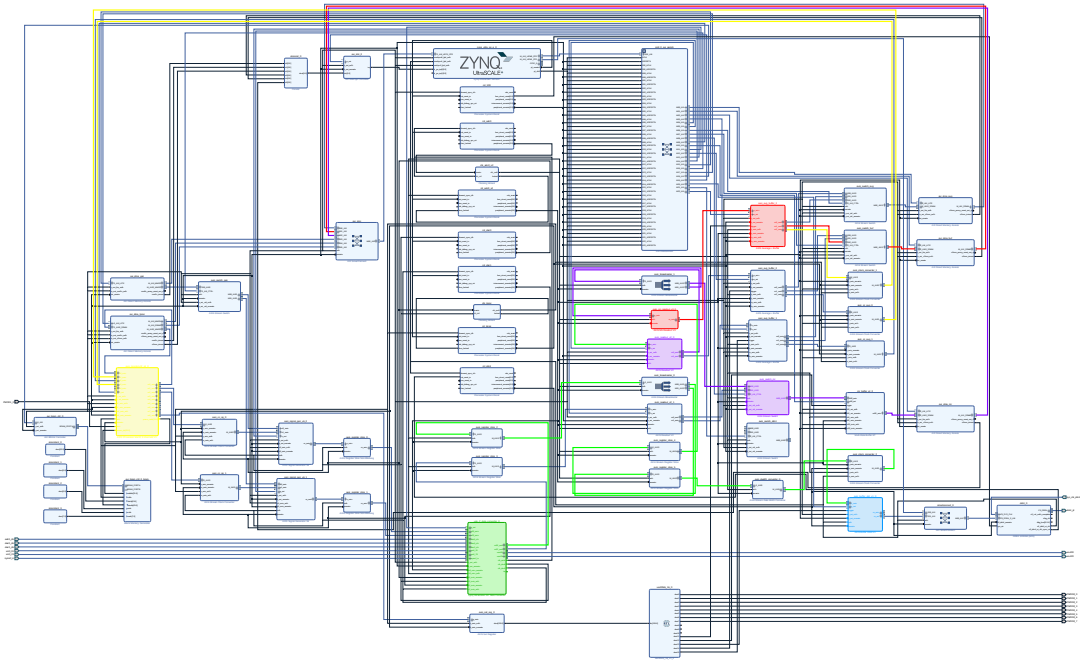

Figure 4.5: Block design of the modified QICK with the respective blocks from Fig. 4.4 highlighted in corresponding colours along with their related datapaths

4.2.2. Latency comparisons

The latency of the complete design in feedback mode can be calculated from the inference time of the accelerator found before and the specifications of the other block from the QICK [34]. The complete path in feedback starts from the readout pulse output from the DAC through a coax cable in loopback mode to the ADC input, where it is demodulated and fed to the readout blocks (the accelerator), from where it is accessed by the timed processor for the feedback conditional pulse firing. The DAC to ADC path latency is 117 ns. This is followed by the latency of the accelerator block, 86.4 ns and the processor's conditional evaluation and instruction, 42 ns. This gives a total latency of 245.4 ns.

This latency is comparable to that advertised by commercial quantum controllers from Quantum Machines, OPX+ with 220ns [64] and Qblox with under 400 ns.

4.2.3. FPGA resources utilised

The resource utilisation of the implemented design is shown in Fig. 4.6. When compared to the estimate obtained from Vitis HLS in 3.4, the actual resource usage is less. This is so because of Vitis' estimation being pessimistic and the resource utilisation itself being optimised in the Vivado implementation process.

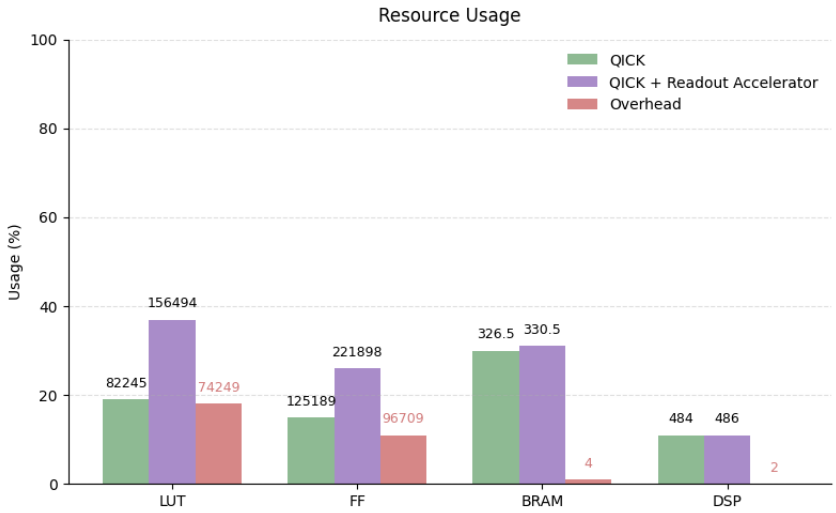

Figure 4.6: Resource utilisation of the complete design on the XCZU48DR-FFVG-1517 FPGA of the RFSoC 4x2. LUT and DSP are the Lookup Tables and Digital Signal Processing units - the computational resources, whereas FF and BRAM are FlipFlops and Block RAMs - the memory resources.

4.2.4. Software Layer changes

To accommodate the change in the QICK design, the software libraries of the controller have to be modified. The necessary modifications are listed below:

- The accelerator IP runs continuously, processing data according to the availability of input signals. However, the outputs are saved only when the buffers depicted in Fig. 4.4 are triggered from the *tProcessor*. The buffer required to be triggered in an experiment is identified by its preceding block. Thus, for the buffer to be usable, the accelerator itself should be identifiable. Hence, the accelerator IP is added to the readout drivers for PYNQ.
- Previously, the DDR buffer was used to store long traces of the demodulated and decimated I and Q components. In this design, however, it is used to store long-time traces of the raw ADC data. Its corresponding driver is updated accordingly. For instance, instead of 512 samples consisting of 256 IQ pairs, 512 samples of 8 raw 16-bit ADC values are processed.

The changes listed above make it possible to use the readout accelerator block to work with the default environment of the QICK controller - the Jupyter notebook or a Pyro instance of the QICK SoC.

5

Experiments and Results

The NN-accelerated QICK system on the RFSoC 4x2 board has been integrated into the instrument stack for spin qubit experiments. The board's capabilities for performing RF reflectometry measurements, ranging from tuning the sensor dots to acquiring double-dot charge stability diagrams, are demonstrated. Further, readout signal classification is performed, and long-time-trace data is acquired for neural network training of the readout signals and also readout noise analysis.

5.1. Experimental Setup

Fig. 5.1 illustrates the path of DC and RF signals used in spin qubit experiments along with the electrical components and instruments. The spin qubit device is bonded to an in-house Printed Circuit Board (PCB), 17-022, which has 96 DC and 32 high-frequency (HF) lines, sufficient for the device currently tested (see Fig. 5.5 (a) for the PCB). An Oxford Instruments ProteoxMX dilution refrigerator is used for cooling the PCB with the device to operating temperature regimes (where $k_B T$ (thermal excitation energy) $\ll E_C$ (the charging energy for quantum dot levels)). It has four plates for thermalising before the final mixing chamber plate, which thermalises to 10mK, is reached (remaining plates are listed in Fig. 5.1).

There are 96 unattenuated DC lines for biasing purposes. These are sufficient for all the DC lines on the PCB. QDAC II and in-house build DAC modules (D5As) are used for bias and are connected to the DC loom via in-house Matrix modules.

There are 28 HF lines, out of which 22 are for control pulses, four are reflectrometry in/out, and the remaining two are used for pulsing cryogenic parametric amplifiers or for biasing gates, as the amplifiers in our setup do not require pulsing. Cryogenic amplifiers amplify the reflected signals on the reflectrometry output lines. In the current setup, one of the reflectometry input lines is further attenuated before being connected to the RFSoC DAC port. One of the output lines is amplified by two amplifiers, a cryo-amplifier from Cosmic Microwave and a room-temperature amplifier, before being connected to the RFSoC ADC coupled port. A few HF lines are connected to the OPX+ for pulsing experiments (not performed in this work) and for benchmarking the suitability of RFSoC 4x2 as a replacement for the OPX+'s readout functionality. The attenuation and amplification of all the lines are shown in Figure 5.1.

All the room temperature electronics are controlled via either SPI (Serial Parallel Interface) through the in-house SPI rack or the QCoDeS station on the host. We developed the driver for the QICK (i.e. QICK hosted on RFSOC 4x2) to be plugged as a QCoDeS instrument on the station.

5.2. Experiments

For the experiments performed in this work, the QICK was only used for reflectometry purposes and was not used to drive other gates, like barriers or plungers, because of the hardware restrictions of the RFSoC 4x2 board. It has only two DAC outputs, one of which is being used for reflectometry. Further, the differential outputs of the DACs are already converted to single-ended by a balun on chip, thus not allowing the board to apply a fast DC offset on the HF lines. A more advanced board, AMD Zynq RFSoC

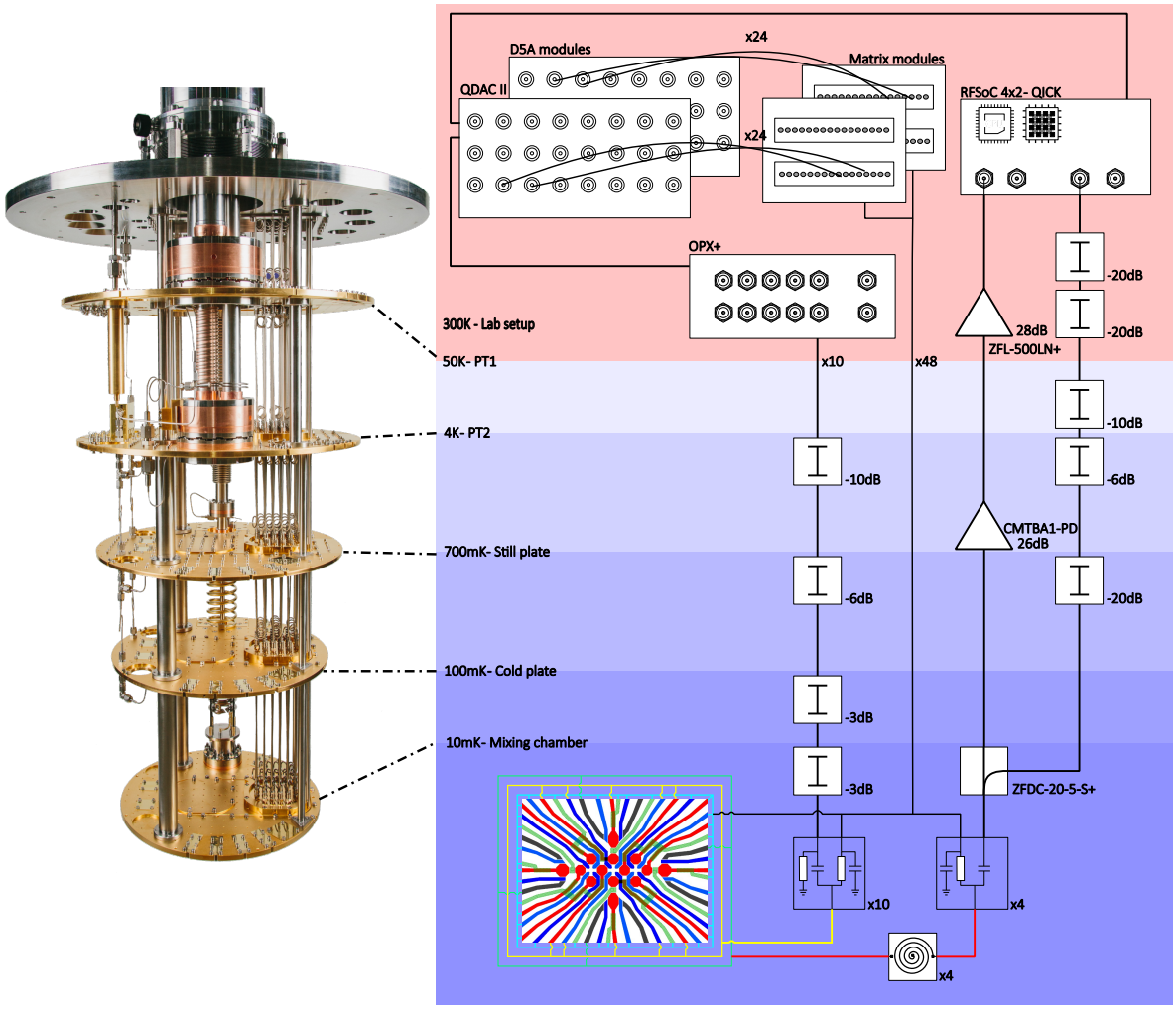

Figure 5.1: Experimental setup for measuring spin qubits with the RFSoC systems. The uncanned Oxford Instruments ProteoxMX cryostat is shown to the left with its different thermalising states marked in the schematic to the right. QDAC-II and D5A modules are connected to the matrix modules, which in turn connect to the fridge's DC lines. The RFSoC and QICK are directly connected via coaxial cables to the HF lines. The attenuators and amplifiers in between the HF lines are shown. Apart from their room temperature counterparts, they are thermalised by anchoring them to the different plates of the cryostat. The schematic is adapted in part from [48]

ZCU216 [79], exists in the lab and will be utilised for further experiments, but was not used for the proof-of-principle readout implementations in this thesis.

QDAC-II was used to sweep the gates, triggering QICK to run the measurement programs and perform readout. The QDAC-II, however, has a low slew rate, thus limiting the experiments that can be done. This is because a slow slew rate means slower ramp speeds, which leads to slow ramping between different operating voltage regions. This slow ramping could lead to a particular state collapsing or decohering before it can be measured or further manipulated. For instance, with this combination of instruments, observing Pauli Spin Blockade, which requires fast pulsing across the (2N-1, 1)-(2N,0) transition, is not feasible.

Therefore, experiments that set the stage for qubit readout but can be performed at these slower ramp rates were chosen - spectrometry of the resonators used for reflectometry, tuning of the sensor dot for subsequent readouts, and obtaining the charge-stability diagrams. done before data was acquired for training and testing the NN-accelerated readout scheme. Since new blocks have been added to the QICK design and the firmware rebuilt, its functionality is validated first.

5.2.1. Testing QICK on the RFSoC 4x2

After the connection to the RFSoC board is established, loopback tests are done by connecting the RFSoC DAC to the ADC, as seen in Fig. 5.2 (a). A simple test is conducted to check the modulation, demodulation, and saving of a square enveloped pulse. Fig. 5.3 (a) is the acquired waveform for one program run and shows the time-series demodulated IQ components for a 100 ns wide pulse with 200 ns of ADC data saved. Running the program 100 times and averaging in software (Fig. 5.3 (b)) demonstrates consistency in the timings and improves the signal by reducing the noise.

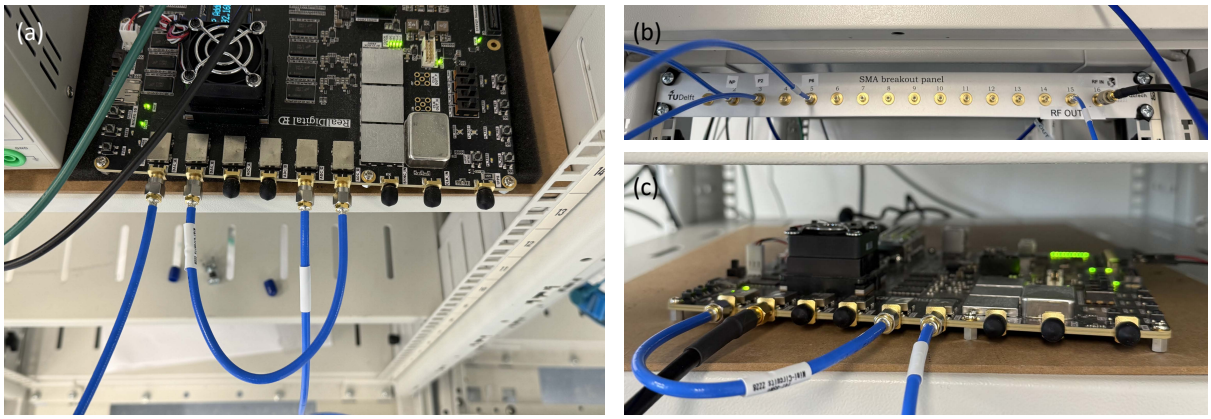

Figure 5.2: The RFSoC 4x2 board setup. (a) shows the loopback connection for validating the signal generation and acquisition. (c) shows the same DAC and ADC connected to the cryostat cabling via the breakout panel in (b)

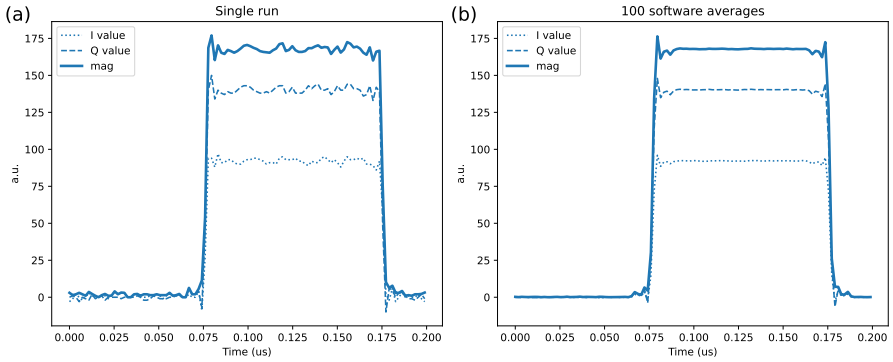

Figure 5.3: (a) shows a loopback test to check the pulse shape and the amplitude. 100 program results are averaged in software (b) to show consistency and reduction in noise.

Averaging in software, however, is slow because the complete program has to be compiled every time on the host and uploaded to the board. This can be improved by compiling the program once and then running a hardware loop to repeat the same compiled program using the timed processor. These repetitions are shown in Fig. 5.4 with timings and amplitudes being consistent across runs.

With the firmware functioning as expected, the same board ports (Fig. 5.2 (c)) are connected to the fridge reflectometry lines (Fig. 5.2 (b)). The DAC output is first attenuated by 40 dB to reduce the quantisation noise which arises at lower amplitudes required for the readout pulses. The output line from the fridge is first fed into a room temperature amplifier before feeding in the QICK ADC.

The first set of experiments aims to determine the frequencies for the reflectometry readout. This also helps find and tune the optimal settings for the amplifiers.

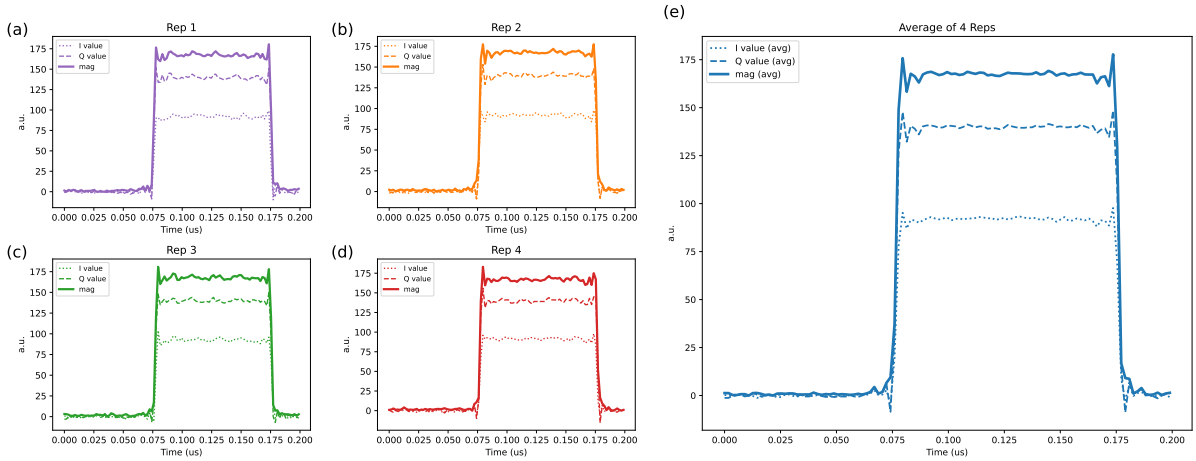

Figure 5.4: Using the timed processor of the QICK to run loops of the acquisition in hardware speeds up the runtime, as the program needs to be compiled and uploaded to the processor only once. (a)-(d) shows the consistency in repetitions while (e) shows the average

5.2.2. Reflectometry characterisation

First, a spectroscopy check is done to identify the resonances of the off-chip inductors, which form the tank circuit and are responsible for frequency selection in the measurements. As we sweep the frequency, as shown in Fig. 5.5 (b), the four resonances observed correspond to the four resonators bonded to an equal number of sensors of the device (5.5 (a)). This also validates the RFSoC's functionality to sweep pulse frequencies.

From the previous runs, it is known that the smallest frequency around 80 MHz corresponds to the northern sensor NS of the device; it can also be found by doing some preliminary experiments. Zooming in, Fig. 5.5 (b), shows the resonance to be at 80.72 MHz. In the following experiments, this value is used as the frequency parameter in the pulse configurations.

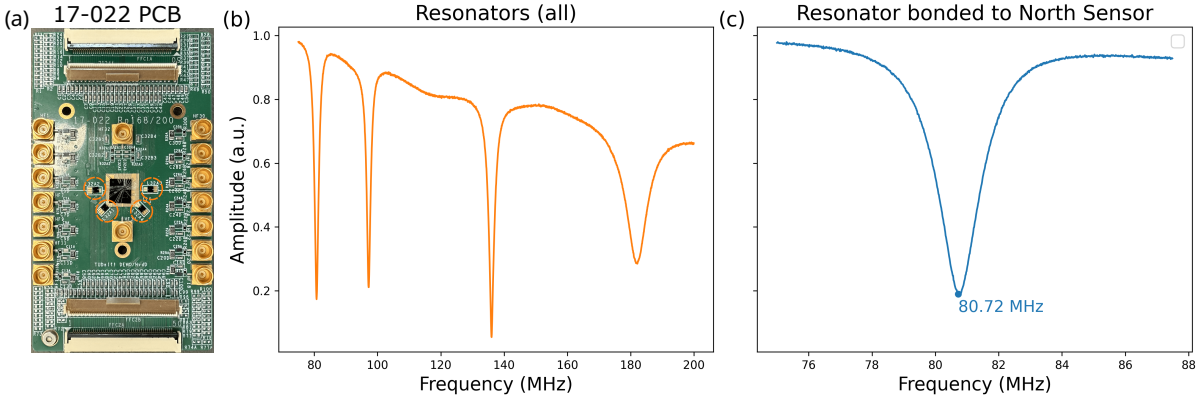

Figure 5.5: Resonator spectroscopy to find the resonances of the four off-chip resonators. (a) shows the four resonators, circled in orange, bonded to the 17-022 PCB (The device and resonators bonded are different from the rest of the experiments; this image is meant to show the PCB and the resonator bonding positions). Resonance peaks corresponding to the actual resonators are shown in (b) with a zoom in (c) on the resonator of interest (bonded to the north sensor ohmic gate)

5.2.3. Tuning the sensor dot using RFSoC

The tuning process begins by verifying the turn-ons and cut-offs of all gates to identify the voltage ranges of interest and ensure the device sustained no damage during the glueing, bonding, and cooling-down in the cryostat. Leakages between the gates and to the ground are also checked. These checks are usually done with current measurements. With these steps complete, we begin tuning the sensor, specifically the north sensor. The gates of interest to this work are depicted in Fig. 5.6 with the greyed out region acting as a reservoir for the two dots of interest.

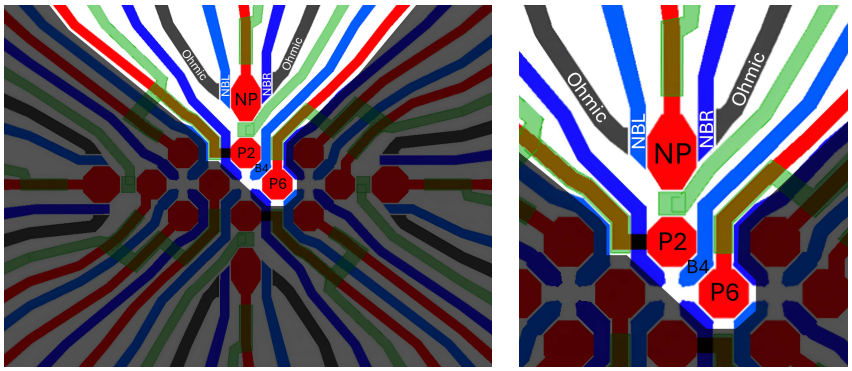

Figure 5.6: The gates of the device used in the experiments are shown. Due to space limitations in connecting the HF lines to the PCB attached inside the fridge puck, only 10 of them could be used. Except for the gates used in the experiments of this thesis, the rest are used as the reservoirs.

By fixing the voltage of the sensor plunger gate (NP) to be sufficiently high in magnitude (within voltage limits before it starts leaking) to trap enough charges underneath, the barriers of the sensor - NBL and NBR are swept. Fig. 5.7 (a) thus shows oscillations corresponding to Coulomb blockades. These are so because changing the barrier potentials would also periodically align the fixed plunger/dot potential in between the barrier potentials, allowing charge transitions across the dot. A NBL-NBR voltage pair on one such oscillation peak, where the reflectometry response amplitude is high, is chosen, and the sensor plunger is swept to check for Coulomb oscillations. As seen in Fig. 5.7 (b), this is indeed a good configuration for the sensor dot. We have now demonstrated that the Coulomb peaks of the sensor dot can be detected with good sensitivity and sharpness; this is essential as these peaks are used for the readout of the qubit states. Next, we will demonstrate that we can use the tuned sensor to tune the charge state of the holes trapped in the double quantum dot.

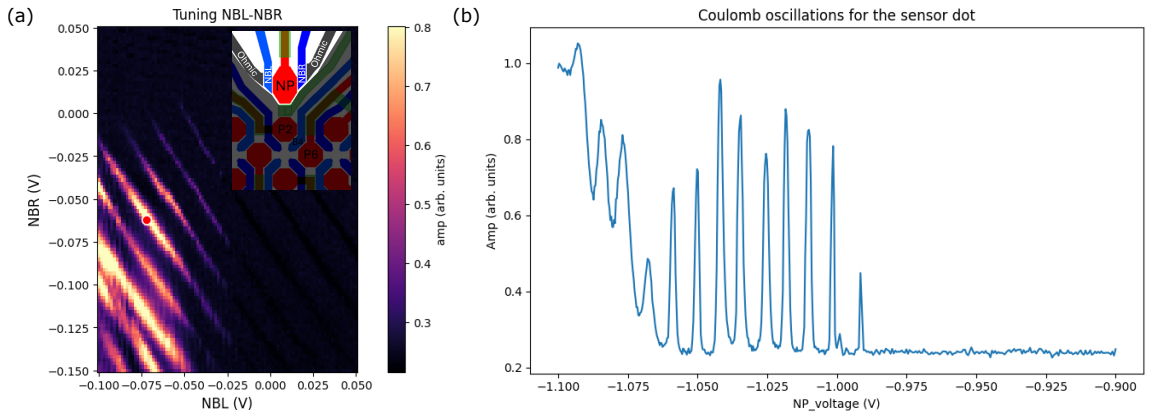

Figure 5.7: The sensor dot is tuned to Coulomb oscillations. (a) shows the reflectometry response when the sensor barriers, NBL and NBR, are swept. The voltage values at the red marker are used when sweeping the sensor plunger NP in (b), which shows good Coulomb oscillations. A voltage value at which the slope of the response in (b) is high is chosen for the plunger NP for best sensitivity.

After sensor tuning, charge stability sweeps for the double-dots formed by plungers P2 and P6 are done.

5.2.4. Double dot Charge Stability Diagrams (CSD) using RFSoC

The charge stability diagrams are obtained by sweeping the plungers P2 and P6 via the QDAC-II virtual sweep functionality. An outer slower loop sweeps P6 while a fast inner loop sweeps P2. The data from QICK is collected one fast loop at a time due to timing mismatches between the data acquisition and sweep triggers. Fig. 5.8 shows the acquired double dot charge stability diagrams.

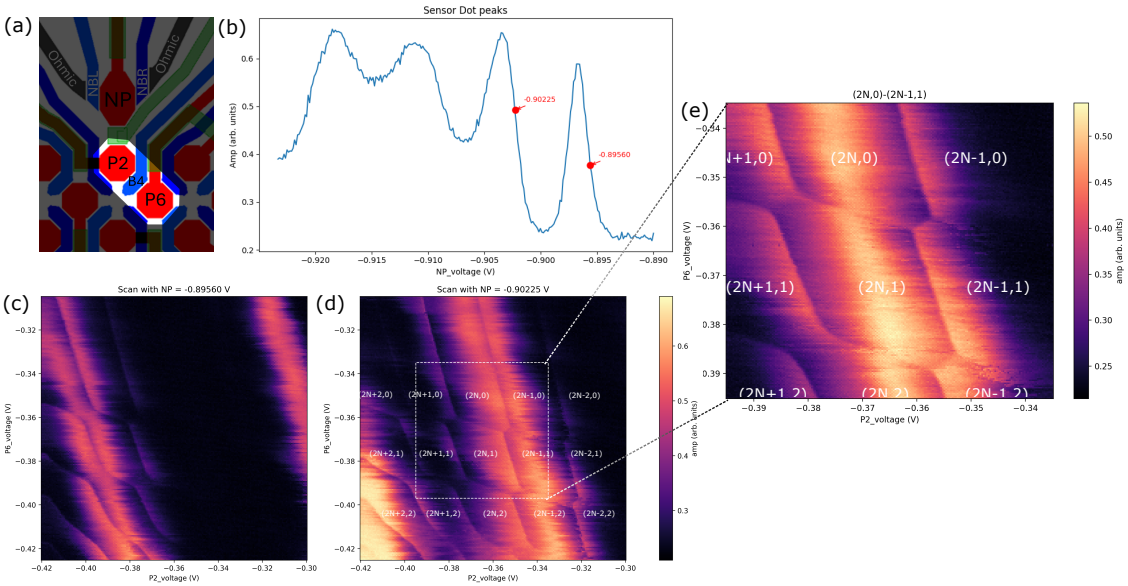

Figure 5.8: P2-P6 Charge Stability Diagrams. (a) shows the gates P2 and P6, which are swept. Maps at two sensor plunger settings in (b) are shown in (c) and (d). As is evident, NP = -0.90225 V gives much better visibility of the transitions. A zoom in on the parameter space of interest is shown in (e).

5.3. Testing NN using experimental data

The QDAC-II and RFSoC setup's inability to pulse the gates fast enough to observe PSB has been discussed before. Additionally, the use of other equipment, like the high-frequency commercial FPGA-based instrument OPX1000 from Quantum Machines, for achieving PSB-enabled readout, was still being developed concurrently during this work. Thus, to train, test, and evaluate this work, we decided to concentrate on the sensor dot and its peaks. In essence, PSB and hence the qubit state measurement is the differentiation of the sensor peak signal, i.e. the reflectometry measurements of the sensor peak height. This is a proxy for the qubit measurements; a full cycle would include initialisation of the qubit, spin manipulations, and spin-to-charge conversion, followed by measuring this sensor signal across the transition. While it lacks the physical correlations which arise from the qubit, the classification of the sensor peak levels provides a proxy two-level system to test this work.

Thus, we gather 1000 shots of long-time-traced (10 us) raw ADC and demodulated IQ data at two settings of the sensor, maximising its contrast (in Fig. 5.9 (a)). Each shot comprises 24576 samples of the raw and I/Q components. Integrated traces of single shots for two states are shown in Fig. 5.9 (b). To then validate the separation of the IQ clusters for the two states, the integrated outputs for all shots are shown in Fig. 5.9 (c) for three different integration times.

Fig. 5.9 (c) also shows the accuracy of threshold-based classification of the integrated datasets. The thresholding is done simply by finding the equidistant normal from the means of the two clusters and using it as the threshold value for subsequent measurements. This is a standard single-shot readout characterisation, which needs to be done often to account for the drift of the sensor over longer times. The maximum accuracy is achieved for the longest integration times of 10 us due to averaging noise and better clustering.

With the acquired data, the workflow depicted in Fig. 3.8 is followed to regenerate all the neural network test results. Using the instantaneous outputs, however, leads to an accuracy of < 50% as seen in Fig. 5.10 (a) for the initial 128 samples of I components. Further tests showed that sliding this constant window or

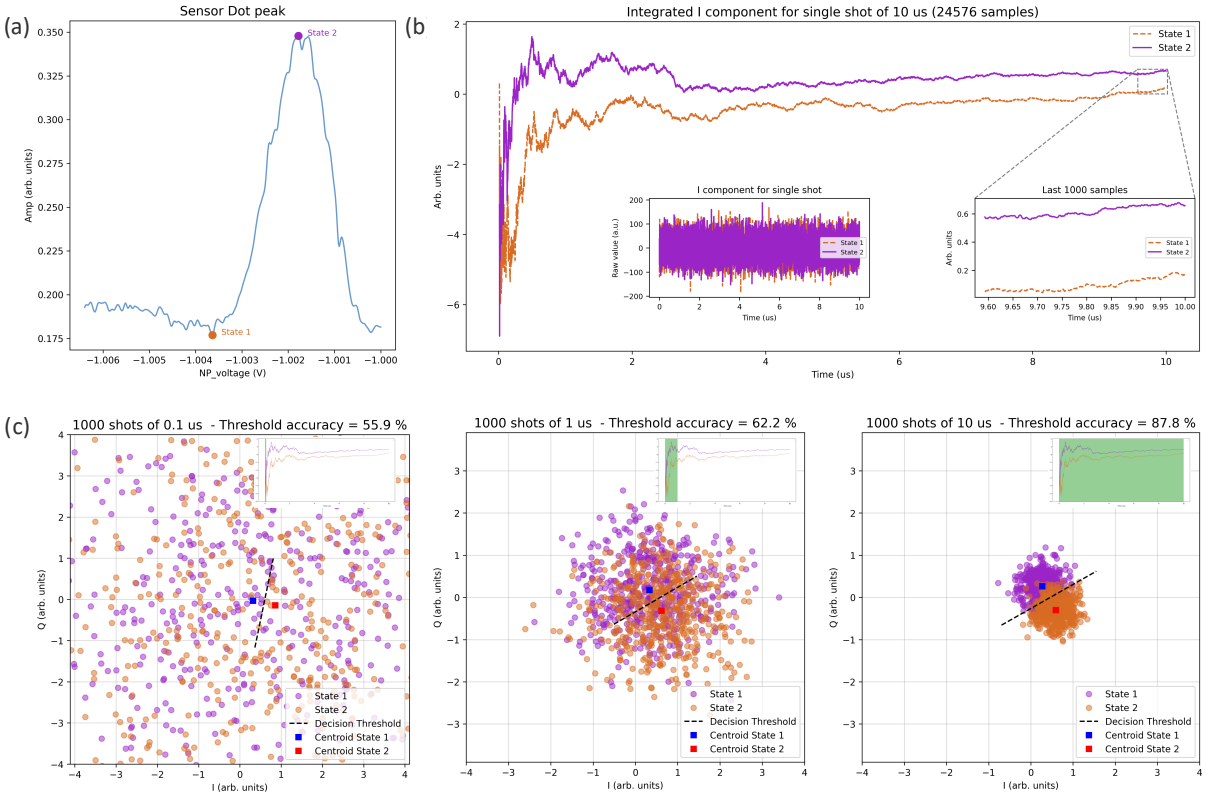

Figure 5.9: Data for training and testing the neural network is acquired. (a) shows the two sensor settings used to acquire 1000 shots of 10 us long time-traces. The sensor is randomly set at one of the values at each of 1000 runs to simulate the output when PSB measurements are done. Complete integrated time traces of two single shots, one for each state, are shown in (b) with the final outputs as well as the complete instantaneous values inset. (c) shows the clusters in the IQ plane of the final integrated values of the 1000 shots for three different integration times - 0.1 us, 1 us and 10 us (also shown inset)

using decimated longer inputs does not improve or degrade this accuracy, leading us to speculate that the network is unable to learn from these small datasets. Larger networks are being investigated, along with further analysis of noise, which could contribute to this behaviour.

On the other hand, using integrated data (Fig. 5.10 (b)), we were able to demonstrate that the NN classification performance is sufficiently close to thresholding seen in Fig. 5.9 (c) when the integration times are equal. The NN size is kept the same by decimating the data accordingly.

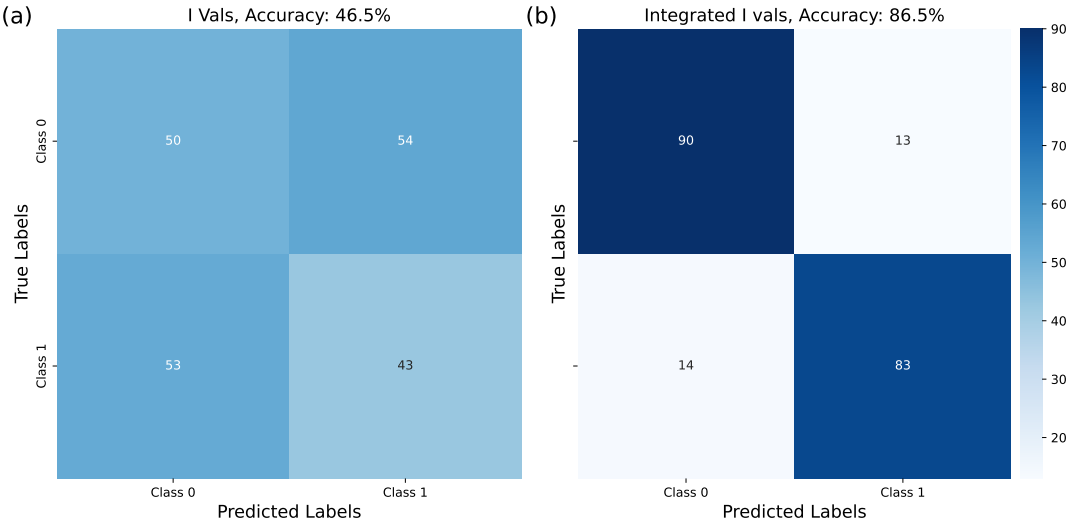

Figure 5.10: Results of NN training on experimental data. (a) shows the outputs for 128 demodulated I samples without any decimation. (b) shows the NN results for the complete sample set of 10 us data.

6

Conclusion and Outlook

6.1. Conclusions

This work focused on improving the readout of spin qubits defined in gate-defined devices on semiconductor heterostructures. In this scope, RFSoC boards, which house both the readout electronics and a programmable FPGA, are used as a single-board solution for acquiring measurement data, processing and classification using ML-based inference.

With the techniques to read the charges and spins reviewed in Chapter 2, we find the RFSoC 4x2 board, with the custom QICK controller employed on the FPGA logic, is sufficient to perform reflectometry measurements of the relevant devices. This has been demonstrated experimentally in Chapter 5 with a spin qubit gate-defined semiconductor device for the first time *to our knowledge*. HRL has released an extension of QICK, spinQICK, for similar spin qubit devices, towards the end of this thesis [80].

Classification of states with fewer measurement samples is shown in Chapter 3 by the use of light FNN models to classify simulated qubit signals. A workflow has been devised and utilised to synthesise a hardware design that maintains classification accuracy while being resource-efficient.

Chapter 4 showcases the development of a readout accelerator IP, its validation on the simulated qubit state data and the subsequent integration in the QICK controller design. This gives, *to our knowledge*, the first ML-accelerated readout controller for gate-defined semiconductor spin qubit devices. This controller design is implemented on the RFSoC 4x2. It has been utilised in the experiments of Chapter 5, including acquiring data for training and testing of the neural network on experimental data, and improving the simulator with noise analysis from the gathered data. Training and testing on the experimental data have shown the effectiveness of the network for the classification of the integrated signal components with accuracy comparable to thresholding techniques.

6.2. Outlook

RFSoC systems provide a wonderful testbed for experimenting with different protocols involving fast acquisition and processing of measurement data. Amongst the numerous possibilities, a few planned further works are listed:

- *New architecture of readout block.* New accelerator block, which collects and integrates the readout signal before classification is required based on the results of the Sec. 5.3.
- *Dynamic weight loading.* The gate-defined semiconductor qubit devices currently suffer from drift in the operating voltages. This requires retraining the neural network to classify the drifted signals. The current design of the readout accelerator requires rebuilding with each retraining to update the parameters on chip. Since the neural network architecture can remain the same, subsequent sets of weights and biases can be uploaded to the FPGA memory and retrieved during computations. Although this eliminates the need to rebuild the complete firmware, it introduces an additional memory-access latency to fetch weights and biases for neural network computations. Further

analysis of this introduced latency will be required when the dynamic loading of weights is possible in the design.

- *Qubit readout.* Readout of spin qubits will require fast control of the gate biasing to pulse between manipulation and readout parameters. To achieve this with the RFSoC board systems, two practical approaches will be considered: integrating the RFSoC 4x2 with a controller like the OPX+, which can pulse the gates while triggering the QICK on the RFSoC to perform readout, or utilising the larger ZCU216 RFSoC board from Xilinx, which features eight DACs with differential outputs. Bypassing the HF baluns, adding DC-coupled amplifiers with the DACs will be used for fast pulsing in PSB measurements.
- *Model extensions.* Recurrent Neural Networks (RNNs), which are suitable for the classification of time-series data, can be implemented using the same workflow to test their accuracy and inference latency.

With faster readout enabled by the RFSoC, feedback-based algorithms would be one of the next logical and desired tools. The architecture for the readout accelerator (Fig. 4.4) already has the data feedback to the timed processor of the controller. Implementation of the algorithms on the processor will pave the way for mid-circuit measurements.

References

[1] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997. DOI: 10.1137/s0097539795293172.

[2] L. K. Grover, "A fast quantum mechanical algorithm for database search," *Proceedings of the Annual ACM Symposium on Theory of Computing*, vol. Part F129452, pp. 212–219, 1996. DOI: 10.1145/237814.237866.

[3] R. P. Feynman, "Simulating physics with computers," *International Journal of Theoretical Physics*, vol. 21, no. 6-7, pp. 467–488, 1982. DOI: 10.1007/BF02650179/METRICS.

[4] A. Aspuru-Guzik, A. D. Dutoi, P. J. Love, and M. Head-Gordon, "Chemistry: Simulated quantum computation of molecular energies," *Science*, vol. 309, no. 5741, pp. 1704–1707, 2005. DOI: 10.1126/SCIENCE.1113479/SUPPL{_}FILE/ASPURU-GUZIK{_}SOM.PDF.

[5] E. Farhi, J. Goldstone, and S. Gutmann, "A Quantum Approximate Optimization Algorithm," pp. 1–16, 2014.

[6] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, pp. 114–117, Apr. 19, 1965.

[7] IEEE International Roadmap for Devices and Systems (IRDS), "Executive summary 2021," 2021. DOI: 10.1109/IRDS54852.2021.00007.

[8] J. M. Elzerman, R. Hanson, L. H. Van Beveren, B. Witkamp, L. M. Vandersypen, and L. P. Kouwenhoven, "Single-shot read-out of an individual electron spin in a quantum dot," *Nature*, vol. 430, no. 6998, pp. 431–435, Jul. 2004. DOI: 10.1038/NATURE02693;KWRD=SCIENCE.

[9] H. Kiyama, D. van Hien, A. Ludwig, A. D. Wieck, and A. Oiwa, "High-fidelity spin readout via the double latching mechanism," *npj Quantum Information*, vol. 10, no. 1, pp. 1–7, 2024. DOI: 10.1038/S41534-024-00882-1;SUBJMETA=1130,2798,2802,483,639,766;KWRD=ELECTRONIC+AND+SPINTRONIC+DEVICES,QUBITS.

[10] W. I. Lawrie, M. Rimbach-Russ, F. v. Riggelen, *et al.*, "Simultaneous single-qubit driving of semiconductor spin qubits at the fault-tolerant threshold," *Nature Communications 2023 14:1*, vol. 14, no. 1, pp. 1–7, 2023. DOI: 10.1038/s41467-023-39334-3.

[11] T. Tanttu, W. H. Lim, J. Y. Huang, *et al.*, "Assessment of the errors of high-fidelity two-qubit gates in silicon quantum dots," *Nature Physics 2024 20:11*, vol. 20, no. 11, pp. 1804–1809, 2024. DOI: 10.1038/s41567-024-02614-w.

[12] A. M. Zwerver, T. Krähenmann, T. F. Watson, *et al.*, "Qubits made by advanced semiconductor manufacturing," *Nature Electronics 2022 5:3*, vol. 5, no. 3, pp. 184–190, 2022. DOI: 10.1038/s41928-022-00727-9.

[13] S. K. Bartee, W. Gilbert, K. Zuo, *et al.*, "Spin-qubit control with a milli-kelvin CMOS chip," *Nature*, vol. 643, no. 8071, pp. 382–387, 2025. DOI: 10.1038/S41586-025-09157-X;TECHMETA=120;SUBJMETA=2802,481,483,639,766,925,927;KWRD=QUANTUM+INFORMATION,QUBITS.

[14] M. Veldhorst, H. G. Eenink, C. H. Yang, and A. S. Dzurak, "Silicon CMOS architecture for a spin-based quantum computer," *Nature Communications*, vol. 8, no. 1, pp. 1–8, 2017. DOI: 10.1038/S41467-017-01905-6;SUBJMETA=2802,481,483,639,766,925,927;KWRD=QUANTUM+INFORMATION,QUBITS.

[15] I. H. Kim, Y. H. Liu, S. Pallister, W. Pol, S. Roberts, and E. Lee, "Fault-tolerant resource estimate for quantum chemical simulations: Case study on Li-ion battery electrolyte molecules," *Physical Review Research*, vol. 4, no. 2, p. 023 019, 2022. DOI: 10.1103/PHYSREVRESEARCH.4.023019/FIGURES/9/MEDIUM.

[16] J. M. Gambetta, J. M. Chow, and M. Steffen, "Building logical qubits in a superconducting quantum computing system," *npj Quantum Information*, vol. 3, no. 1, pp. 1–7, 2017. DOI: 10.1038/S41534-016-0004-0;SUBJMETA=2802,481,483,639,766;KWRD=QUANTUM+INFORMATION,QUBITS.

[17] D. Bluvstein, S. J. Evered, A. A. Geim, *et al.*, "Logical quantum processor based on reconfigurable atom arrays," *Nature 2023 626:7997*, vol. 626, no. 7997, pp. 58–65, 2023. DOI: 10.1038/s41586-023-06927-3.

[18] B. Paquelet Wuetz, D. Degli Esposti, A. M. J. Zwerver, *et al.*, "Reducing charge noise in quantum dots by using thin silicon quantum wells," *Nature Communications*, vol. 14, no. 1, pp. 1–9, 2023. DOI: 10.1038/S41467-023-36951-W;SUBJMETA=1000,1001,1017,119,301,639,766;KWRD=QUANTUM+DOTS,SPINTRONICS.

[19] K. Takeda, A. Noiri, T. Nakajima, *et al.*, "Rapid single-shot parity spin readout in a silicon double quantum dot with fidelity exceeding 99%," *npj Quantum Information*, vol. 10, no. 1, pp. 1–6, 2024. DOI: 10.1038/S41534-024-00813-0;SUBJMETA=2802,481,483,639,766;KWRD=QUANTUM+INFORMATION,QUBITS.

[20] N. W. Hendrickx, L. Massai, M. Mergenthaler, *et al.*, "Sweet-spot operation of a germanium hole spin qubit with highly anisotropic noise sensitivity," *Nature Materials 2024 23:7*, vol. 23, no. 7, pp. 920–927, 2024. DOI: 10.1038/s41563-024-01857-5.

[21] A. Ruffino, T. Y. Yang, J. Michniewicz, Y. Peng, E. Charbon, and M. F. Gonzalez-Zalba, "A cryo-CMOS chip that integrates silicon quantum dots and multiplexed dispersive readout electronics," *Nature Electronics 2021 5:1*, vol. 5, no. 1, pp. 53–59, 2021. DOI: 10.1038/s41928-021-00687-6.

[22] Y. Salathé, P. Kurpiers, T. Karg, *et al.*, "Low-Latency Digital Signal Processing for Feedback and Feedforward in Quantum Computing and Communication," *Physical Review Applied*, vol. 9, no. 3, Sep. 2017. DOI: 10.1103/PhysRevApplied.9.034011.

[23] P. Campagne-Ibarcq, E. Flurin, N. Roch, *et al.*, "Persistent control of a superconducting qubit by stroboscopic measurement feedback," *Physical Review X*, vol. 3, no. 2, p. 021 008, 2013. DOI: 10.1103/PHYSREVX.3.021008/SUPMAT{_}CAMPAGNE{_}PRX.PDF.

[24] D. Ristè, M. Dukalski, C. A. Watson, *et al.*, "Deterministic entanglement of superconducting qubits by parity measurement and feedback," *Nature 2013 502:7471*, vol. 502, no. 7471, pp. 350–354, 2013. DOI: 10.1038/nature12513.

[25] D. Ristè and L. Dicarlo, "Digital feedback in superconducting quantum circuits," 2015.

[26] C. A. Ryan, B. R. Johnson, D. Ristè, B. Donovan, and T. A. Ohki, "Hardware for Dynamic Quantum Computing," *Review of Scientific Instruments*, vol. 88, no. 10, 2017. DOI: 10.1063/1.5006525.

[27] 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. 200 501, May 2015. DOI: 10.1103/PHYSREVLETT.114.200501/ML{_}MEAS{_}SUPP{_}PRL.PDF.

[28] 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*, New York, NY, USA: ACM, Jun. 2023, pp. 1–13. DOI: 10.1145/3579371.3589042.

[29] B. Lienhard, A. Vepsäläinen, L. C. Govia, *et al.*, "Deep-neural-network discrimination of multiplexed superconducting-qubit states," *Phys. Rev. Appl.*, vol. 17, p. 014 024, 1 2022. DOI: 10.1103/PhysRevApplied.17.014024.

[30] C. N. Mude, S. Maurya, B. Lienhard, and S. Tannu, "Efficient and Scalable Architectures for Multi-Level Superconducting Qubit Readout," 2024.

[31] P. K. Gautam, S. Kalipatnapu, S. H, *et al.*, "Low-latency machine learning FPGA accelerator for multi-qubit-state discrimination," 2024.

[32] X. Guo, T. Bunarjyan, D. Liu, B. Lienhard, and M. Schulz, "KLiNQ: Knowledge Distillation-Assisted Lightweight Neural Network for Qubit Readout on FPGA," 2025.

[33] Y. Xu, G. Huang, J. Balewski, *et al.*, "Qubic: An open source FPGA-based control and measurement system for superconducting quantum information processors," *IEEE Transactions on Quantum Engineering*, vol. 2, pp. 1–11, 2021. DOI: 10.1109/TQE.2021.3116540.

[34] L. Stefanazzi, K. Treptow, N. Wilcer, *et al.*, "The QICK (Quantum Instrumentation Control Kit): Readout and control for qubits and detectors," Oct. 2021. DOI: 10.1063/5.0076249.

[35] N. R. Vora, Y. Xu, A. Hashim, *et al.*, "ML-Powered FPGA-based Real-Time Quantum State Discrimination Enabling Mid-circuit Measurements," *Proceedings of ACM Conference (Conference'17)*, vol. 1, 2024.

[36] G. Di Guglielmo, B. Du, J. Campos, *et al.*, "End-to-end workflow for machine learning-based qubit readout with QICK and hls4ml," Jan. 2025.

[37] D. P. DiVincenzo and IBM, "The Physical Implementation of Quantum Computation," *Fortschritte der Physik*, vol. 48, no. 9-11, pp. 771–783, Feb. 2000. DOI: 10.1002/1521-3978(200009)48:9/11<771::AID-PROP771>3.0.CO;2-E.

[38] D. Loss and D. P. DiVincenzo, "Quantum computation with quantum dots," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 57, no. 1, pp. 120–126, Jan. 1998. DOI: 10.1103/PhysRevA.57.120.

[39] J. R. Petta, A. C. Johnson, J. M. Taylor, *et al.*, "Coherent Manipulation of," *Science*, vol. 309, no. September, pp. 2180–2184, 2005. DOI: 10.1126/science.1116955.

[40] F. H. Koppens, C. Buizert, K. J. Tielrooij, *et al.*, "Driven coherent oscillations of a single electron spin in a quantum dot," *Nature 2006 442:7104*, vol. 442, no. 7104, pp. 766–771, Aug. 2006. DOI: 10.1038/nature05065.

[41] D. V. Bulaev and D. Loss, "Spin relaxation and decoherence of holes in quantum dots," *Physical Review Letters*, vol. 95, no. 7, p. 076 805, Aug. 2005. DOI: 10.1103/PHYSREVLETT.95.076805/FIGURES/1/MEDIUM.

[42] D. V. Bulaev and D. Loss, "Electric dipole spin resonance for heavy holes in quantum dots," *Physical Review Letters*, vol. 98, no. 9, p. 097 202, Feb. 2007. DOI: 10.1103/PHYSREVLETT.98.097202/FIGURES/3/MEDIUM.

[43] Y. Hu, H. O. Churchill, D. J. Reilly, J. Xiang, C. M. Lieber, and C. M. Marcus, "A Ge/Si heterostructure nanowire-based double quantum dot with integrated charge sensor," *Nature Nanotechnology 2007 2:10*, vol. 2, no. 10, pp. 622–625, Sep. 2007. DOI: 10.1038/nnano.2007.302.

[44] G. Katsaros, P. Spathis, M. Stoffel, *et al.*, "Hybrid superconductor–semiconductor devices made from self-assembled SiGe nanocrystals on silicon," *Nature Nanotechnology 2010 5:6*, vol. 5, no. 6, pp. 458–464, May 2010. DOI: 10.1038/nnano.2010.84.

[45] R. Li, F. E. Hudson, A. S. Dzurak, and A. R. Hamilton, "Pauli Spin Blockade of Heavy Holes in a Silicon Double Quantum Dot," *Nano Letters*, vol. 15, no. 11, pp. 7314–7318, Nov. 2015. DOI: 10.1021/ACS.NANOLETT.5B02561/ASSET/IMAGES/LARGE/NL-2015-02561V{_}0004.JPEG.

[46] G. Scappucci, C. Kloeffel, F. A. Zwanenburg, *et al.*, "The germanium quantum information route," *Nature Reviews Materials 2020 6:10*, vol. 6, no. 10, pp. 926–943, Dec. 2020. DOI: 10.1038/s41578-020-00262-z.

[47] R. Hanson, L. P. Kouwenhoven, J. R. Petta, S. Tarucha, and L. M. Vandersypen, "Spins in few-electron quantum dots," *Reviews of Modern Physics*, vol. 79, no. 4, pp. 1217–1265, 2007. DOI: 10.1103/RevModPhys.79.1217.

[48] W. I. L. Lawrie, "Spin Qubits in Silicon and Germanium," *Citation*, 2022. DOI: 10.4233/uuid:97c4ea24-9672-4e0b-b7a5-e3a48258c871.

[49] C. A. Wang, V. John, H. Tidjani, *et al.*, "Operating semiconductor quantum processors with hopping spins," *Science (New York, N.Y.)*, vol. 385, no. 6707, pp. 447–452, Jul. 2024. DOI: 10.1126/SCIENCE.ADO5915/SUPPL{_}FILE/SCIENCE.ADO5915{_}SM.PDF.

[50] A. Zubchenko, "DEVELOPMENT OF AUTOMATED SPIN QUBIT TUNING PROTOCOLS USING NEURAL NETWORKS MSc esis by," Tech. Rep.

[51] L. P. Kouwenhoven, D. G. Austing, and S. Tarucha, "Few-electron quantum dots," *Reports on Progress in Physics*, vol. 64, no. 6, p. 701, 2001. DOI: 10.1088/0034-4885/64/6/201.

[52] L. DiCarlo, H. J. Lynch, A. C. Johnson, *et al.*, "Differential charge sensing and charge delocalization in a tunable double quantum dot," *Physical Review Letters*, vol. 92, no. 22, p. 226 801, 2004. DOI: 10.1103/PHYSREVLETT.92.226801/FIGURES/4/MEDIUM.

[53] D. R. Ward, D. Kim, D. E. Savage, *et al.*, "State-conditional coherent charge qubit oscillations in a Si/SiGe quadruple quantum dot," *npj Quantum Information 2016 2:1*, vol. 2, no. 1, pp. 1–6, 2016. DOI: 10.1038/npjqi.2016.32.

[54] R. J. Schoelkopf, P. Wahlgren, A. A. Kozhevnikov, P. Delsing, and D. E. Prober, "The radio-frequency single-electron transistor (RF-SET): A fast and ultrasensitive electrometer," *Science*, vol. 280, no. 5367, pp. 1238–1242, 1998. DOI: 10.1126/SCIENCE.280.5367.1238/ASSET/EEEF1CD8-EDD5-4748-9B78-95A7A351D006/ASSETS/GRAPHIC/SE2286534004.JPEG.

[55] F. Vigneau, F. Fedele, A. Chatterjee, *et al.*, "Probing quantum devices with radio-frequency reflectometry," Feb. 2022. DOI: 10.1063/5.0088229.

[56] T. Kobayashi, T. Nakajima, K. Takeda, A. Noiri, J. Yoneda, and S. Tarucha, "Feedback-based active reset of a spin qubit in silicon," *npj Quantum Information*, vol. 9, no. 1, pp. 1–8, 2023. DOI: 10.1038/S41534-023-00719-3;SUBJMETA=166,481,483,639,766,987;KWRD=ELECTRICAL+AND+ELECTRONIC+ENGINEERING,QUANTUM+INFORMATION.

[57] J. Y. Huang, R. Y. Su, W. H. Lim, *et al.*, "High-fidelity spin qubit operation and algorithmic initialization above 1 K," *Nature 2024 627:8005*, vol. 627, no. 8005, pp. 772–777, 2024. DOI: 10.1038/s41586-024-07160-2.

[58] G. Zheng, N. Samkharadze, M. L. Noordam, *et al.*, "Rapid gate-based spin read-out in silicon using an on-chip resonator," *Nature Nanotechnology 2019 14:8*, vol. 14, no. 8, pp. 742–746, 2019. DOI: 10.1038/s41565-019-0488-9.

[59] A. West, B. Hensen, A. Jouan, *et al.*, "Gate-based single-shot readout of spins in silicon," *Nature Nanotechnology*, vol. 14, no. 5, pp. 437–441, 2019. DOI: 10.1038/S41565-019-0400-7;TECHMETA=120,144;SUBJMETA=2802,483,639,766,925,927;KWRD=NANOSCALE+DEVICES,NANOSCIENCE+AND+TECHNOLOGY,QUBITS.

[60] G. Burkard, T. D. Ladd, A. Pan, J. M. Nichol, and J. R. Petta, "Semiconductor spin qubits," *Reviews of Modern Physics*, vol. 95, no. 2, Apr. 2023. DOI: 10.1103/RevModPhys.95.025003.

[61] A. Noiri, K. Takeda, J. Yoneda, T. Nakajima, T. Kodera, and S. Tarucha, "Radio-frequency detected fast charge sensing in undoped silicon quantum dots," *Nano Letters*, vol. 20, no. 2, pp. 947–952, 2020. DOI: 10.1021/acs.nanolett.9b03847.

[62] Keysight Technologies, *Signal generators and signal sources*, https://www.keysight.com/us/en/products/signal-generators-signal-sources.html.

[63] "SHFSG signal generator," Zurich Instruments. (), [Online]. Available: https://www.zhinst.com/europe/en/products/shfsg-signal-generator.

[64] "OPX+: Ultra-fast quantum controller," Quantum Machines. (), [Online]. Available: https://www.quantum-machines.co/products/opx/.

[65] "Cluster: Scalable quantum control hardware," Qblox. (), [Online]. Available: https://www.qblox.com/products#cluster.

[66] AMD. "Xilinx zynq ultrascale+ RFSoCs." (2025), [Online]. Available: https://www.amd.com/en/products/adaptive-socs-and-fpgas/soc/zynq-ultrascale-plus-rfsoc.html.

[67] R. Gebauer, N. Karcher, and O. Sander, "A modular RFSoC-based approach to interface superconducting quantum bits," in *2021 International Conference on Field-Programmable Technology (ICFPT)*, 2021, pp. 1–9. DOI: 10.1109/ICFPT52863.2021.9609909.

[68] K. H. Park, Y. S. Yap, Y. P. Tan, *et al.*, "Icarus-q: Integrated control and readout unit for scalable quantum processors," *Review of Scientific Instruments*, vol. 93, no. 10, p. 104 704, 2022. DOI: 10.1063/5.0081232.

[69] Real Digital, *RFSoC 4×2 board*, Online; accessed 14 August 2025, 2025.

[70] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. DOI: 10.1038/NATURE14539;SUBJMETA=117,639,705;KWRD=COMPUTER+SCIENCE,MATHEMATICS+AND+COMPUTING.

[71] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[72] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. A. Muller, "Deep learning for time series classification: a review," *Data Mining and Knowledge Discovery*, vol. 33, no. 4, pp. 917–963, 2019. DOI: 10.1007/S10618-019-00619-1/FIGURES/16.

[73] G. T. Landi, M. J. Kewming, M. T. Mitchison, and P. P. Potts, "Current Fluctuations in Open Quantum Systems: Bridging the Gap between Quantum Continuous Measurements and Full Counting Statistics," *PRX Quantum*, vol. 5, no. 2, p. 020 201, Apr. 2024. DOI: 10.1103/PRXQUANTUM.5.020201/FIGURES/24/MEDIUM.

[74] FastML Team, *Fastmachinelearning/hls4ml*, version v1.0.0, 2024. DOI: 10.5281/zenodo.1201549.

[75] J. Duarte *et al.*, "Fast inference of deep neural networks in FPGAs for particle physics," *JINST*, vol. 13, no. 07, P07027, 2018. DOI: 10.1088/1748-0221/13/07/P07027. arXiv: 1804.06913 [physics.ins-det].

[76] AMD, "Vitis hls user guide," Advanced Micro Devices, Inc., User Guide UG1399, version v2024.1, 2024.

[77] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *Proceedings of the 32nd International Conference on Machine Learning*, F. Bach and D. Blei, Eds., ser. Proceedings of Machine Learning Research, vol. 37, Lille, France: PMLR, 2015, pp. 1737–1746.

[78] E. G. Cota, P. Mantovani, G. Di Guglielmo, and L. P. Carloni, "An analysis of accelerator coupling in heterogeneous architectures," in *Proceedings of the 52nd Annual Design Automation Conference*, ser. DAC '15, San Francisco, California: Association for Computing Machinery, 2015. DOI: 10.1145/2744769.2744794.

[79] AMD, *Zcu216 evaluation board*, https://www.amd.com/en/products/adaptive-socs-and-fpgas/evaluation-boards/zcu216.html, Accessed: 2025-08-17, 2025.

[80] HRL-Laboratories, *spinQICK: Open-source control library for electrostatically confined spin qubits*, https://github.com/HRL-Laboratories/spinqick, 2025.

A

Appendix

A.1. Block design of standard QICK firmware

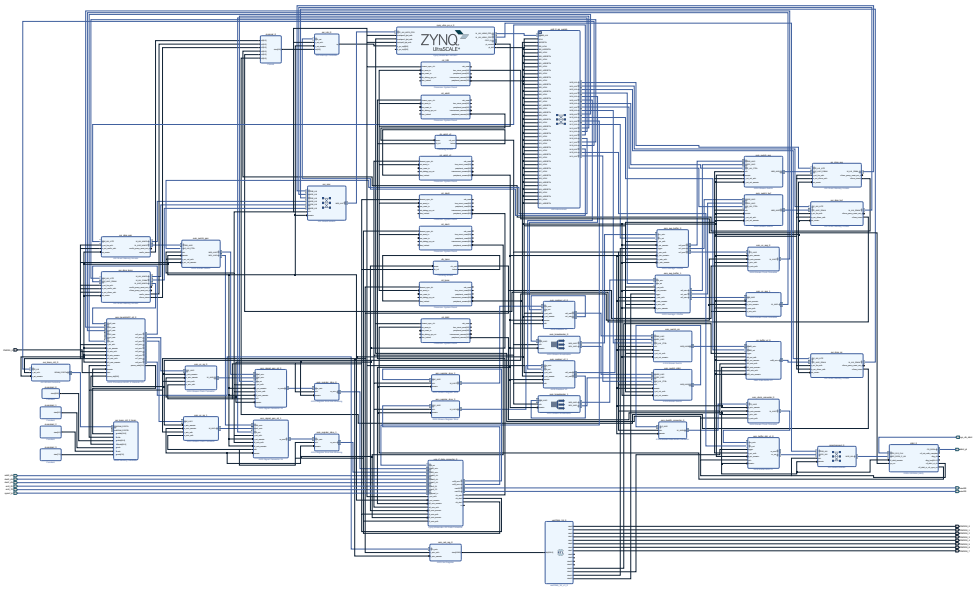

Figure A.1: QICK firmware block design in Vivado

A.2. Architecture for raw data classification

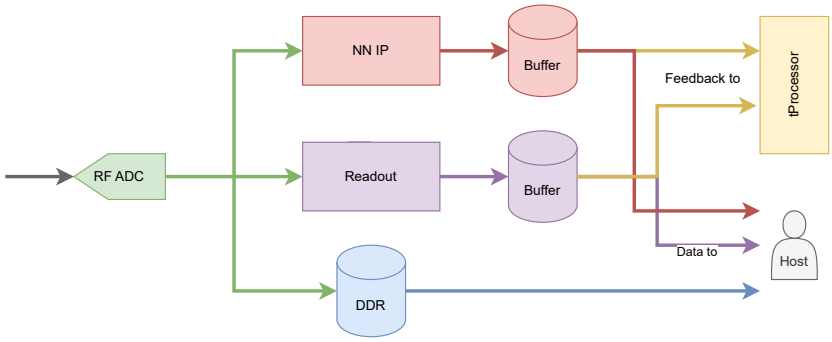

Figure A.2: Schematic showing the proposed design modifications to incorporate the NN IP. ADC data is broadcast to three blocks: NN IP, a demodulation-based readout block, and a large DDR memory for training purposes.

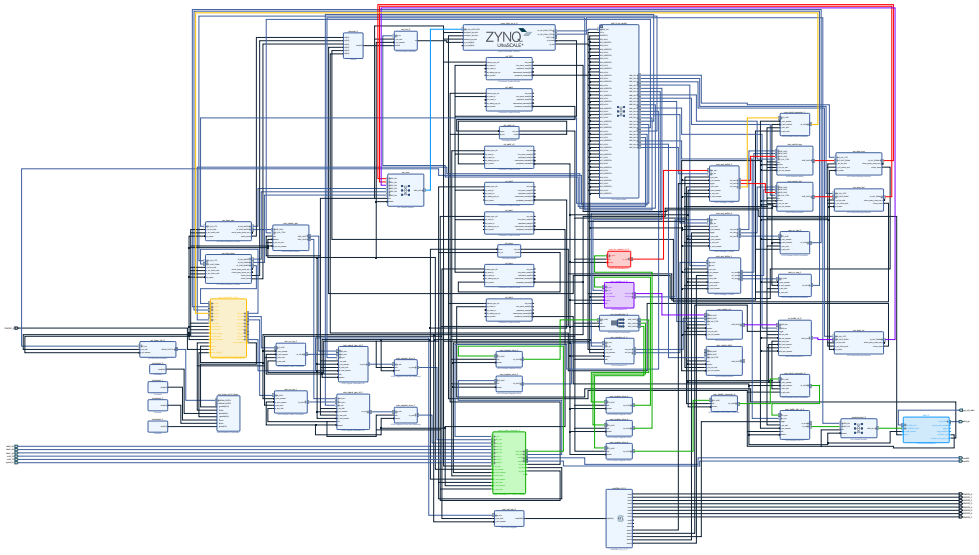

Figure A.3: Block design of the modified QICK with the respective blocks from Fig. A.2 highlighted in corresponding colours along with their related datapaths