Low Level Interface Phase 1 Definition

Description

An interface document must be defined for the low-level interaction between the Qruise calibration & characterization software and the ZIGE control electronics firmware after consultation with the WP partners. This definition is a starting point for a minimum supported list of operations and will continue to evolve over the next phases as more features are added to both the control hardware/firmware and the interfacing middleware this enabling support for a growing collection of high-level applications.

Version History:

Version	Date	Author	Description
1.0	20.01.2024	Roman Razilov, Anurag Saha Roy, Shai Machnes	Phase 1 Definition following preliminary discussions with WP partners, based on an existing implementation already live and extensively tested

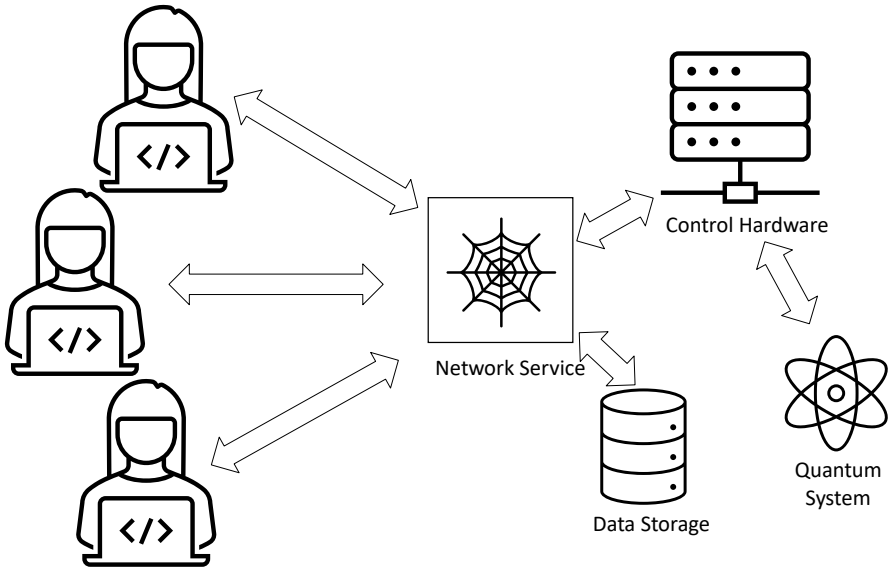
Summary

The key aspects of this document can be summarized in the following few points:

- The low-level interface closely follows the standards set by the OpenQasm3 quantum assembly language [1].
- The implementation of this interface closely follows the API provided by the Qiskit open-source quantum programming framework [2].
- The interface allows for translation of Qruise instructions to LabOneQ [3] experiments via the intermediate OpenQasm/Qiskit layer.
- Active reset operations will be supported as part of the Phase 1 interface.
- No other operations requiring mid-circuit logic will be supported as part of the Phase 1 interface.

General Architecture

The low-level interface at its heart is a server-client architecture. Multiple users with different roles can run experiments on a quantum system remotely, via network protocols. Remote access is essential for cross group collaboration, for physical security and better working environment. The network service installed on a computer (or a computer cluster) exposes web services, giving access to configuration, experiment scheduler, historical data, and parameter history. To run an experiment the server interacts with control hardware to prepare and run the control sequence and measurement. Continuous monitoring data, logs and experiment data stored as plain data objects or in databases appropriate for a task.



Experiment types

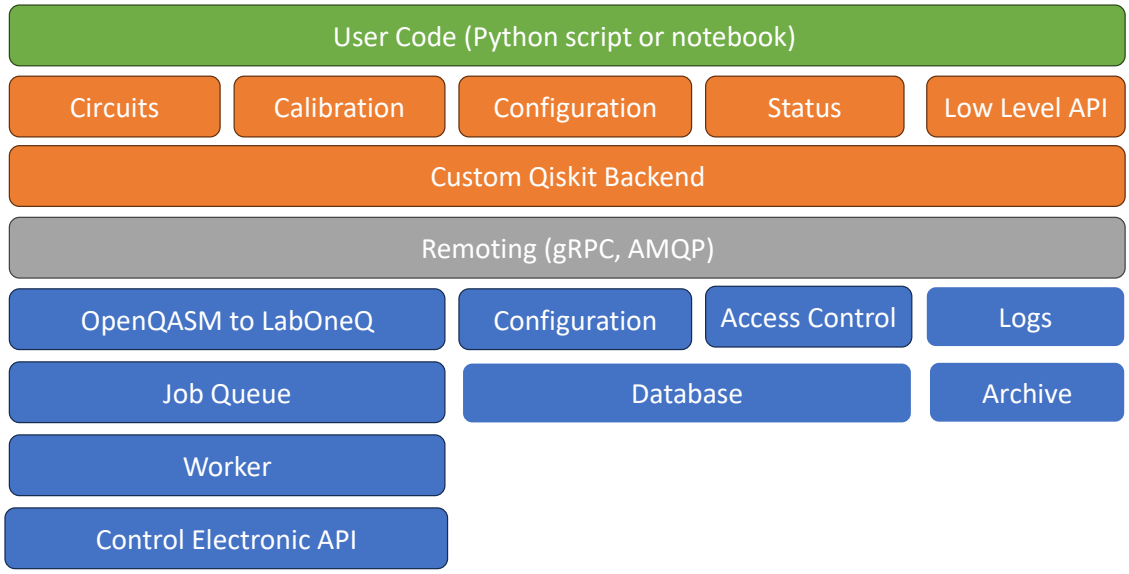
Depending on goal we can identify two types of experiments, performed by users of the quantum computing demonstrators:

1. **Application-oriented**: High-level experiments with execution of quantum circuits on logical and physical qubits: Goal is system benchmarking, compiler development and algorithm verification.
2. **Calibration-oriented**: Low-level experiments, performing relatively local and simple control sequences with variating parameters (i.e. frequency, amplitude, phase, shape, delays). Goal is calibration parameters of the instructions for used to manipulate and measure quantum system state to maximize accuracy and speed.

Architecture layers

The next diagram shows the layered architecture of Qruise Backend API used by remote users. Users define they experiment within a Python code in a Jupyter Notebook or a script. Circuit

and pulse definition uses popular open source Qiskit API (https://www.qiskit.org/), enhanced with Qruise/ZI specific extensions. Role base access control and configuration management, as well as access to historical data uses native Qruise Backend API – designed and developed specially for this purpose.



We choose partial implementation of OpenQASM for high level circuits. These OpenQasm instructions are converted to native LabOneQ experiments for fast and optimized execution on the hardware. The experiment is represented as a set of control sequences including following types of instructions:

- **Quantum Gates:** Basic single and multi-qubit operations like Pauli gates (X, Y, Z), Hadamard (H), CNOT, and various rotation gates (Rx, Ry, Rz). Backend has a native set of gates (i.e. id, sx, x, rz, cx) for every qubit and coupling allowing transpilation of an arbitrary circuit.
- **Measurement Operations:** Instructions to measure the state of qubits.
- **Reset Operations:** To reset qubits to a known state, usually the basis state. Depending on hardware capabilities it can be thermalization, active reset or other reset approaches enabled by the hardware, such es an unconditional reset.
- **Barrier:** A synchronization point in the circuit, preventing certain types of optimizations across the barrier.
- **Custom Gates and Subroutines:** User-defined gates or composite gates built from the standard set.

To leverage on instrument specialities, i.e. hardware supported sweeps and efficient pulse parametrization representations we define pseudo gates.

References

1. OpenQASM 3: A broader and deeper quantum assembly language, arXiv:2104.14722
2. Qiskit: An Open-source Framework for Quantum Computing, DOI: 10.5281/zenodo.2573505
3. Zurich Instruments LabOne Q, https://docs.zhinst.com/labone_q_user_manual/