FPGA-based QPU interface unit for fast qubit control and readout

Project introduction and applied basic theory

Christian Križan[1,3] and Mats Tholén[2,3]

[1] Quantum Technology Laboratory, MC2, Chalmers University of Technology
`krizan@chalmers.se`

[2] Intermodulation Products AB

[3] WACQT, Wallenberg Centre for Quantum Technology

Abstract. This document outlines the 2020 DAT096 project *FPGA-based QPU interface unit for fast qubit control and readout*, and provides the reader with an applied theory set useful for the overall picture.

1 Welcome to the second quantum revolution

It should come as no surprise to you that quantum technology as a whole has initiated an unprecedented growth spur during the last decade. Quantum computing shows good potential to vastly outperform its classical counterpart at specific computational tasks. However, the control of a quantum computer relies on classical electronics, in many cases high-end FPGAs with additional microwave engineering hardware. Interfacing to a quantum processor (QPU) using said FPGAs in turn allow for flexible experimentation in digital signal processing and dataflow management of quantum computations, hence this project in embedded electronic system design. Your task for the upcoming months, will be to develop a QPU interface unit - capable of executing operator-specified quantum computations on a live QPU.

1.1 Background and brief applied theory

The forte of a quantum processor lies in its computational basis. The extension beyond the classical boolean logic allows a quantum computer to for instance solve **any** NP-complete problem, provided that can be mapped onto the so-called ground state of an Ising Hamiltonian [1]. n qubits in turn also represent 2^n mutually orthogonal quantum states in Hilbert space [2], meaning that the qubits represent an exponentially growing vector space. Even though this comparison is somewhat over-simplified, it is not strictly incorrect to claim that the QPU non-linearly gains additional parallelism for every additional qubit. Meaning in turn that there is a strive for higher-qubit QPUs, resulting in increased demand on massive-parallel interface units. Thus, a very active branch of quantum computing treats the problem of qubit interfacing, commonly denoted control and readout.

Qubit control is analogous to setting a qubit to logic state $|0\rangle$, $|1\rangle$, or some more exotic logic state such as the superposition of $|0\rangle$ and $|1\rangle$ (for instance, $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$). Qubit readout corresponds to interrogating the qubit, performing some non-quantum processing on the acquired information and deciding whether the observed qubit was in state $|0\rangle$ or $|1\rangle$. As mentioned, qubit control and readout is done in the classical (= non-quantum) realm.

Typically, more non-exotic computers are attached to the I/O of a quantum processor, and thus control everything that is sent onto or from the QPU itself. The signals sent to the QPU are typically results of some layer higher up in the software stack, such as a quantum circuit compiler. A compiled quantum program consists of a set of microwave pulse signals, which are sent to an FPGA configured as an AWG (arbitrary waveform generator). Additional DACs/ADCs will carry this signal through a long setup of attenuators, mixers, amplifiers and filters onto the QPU core itself. As the signal is modified by the QPU core, it travels back to a DAC, FPGA, and finally to some specified destination such as a host PC.

At MC2, Chalmers, there are likewise QPUs with FPGA interfaces available for purposes akin to those outlined in the background. Which brings us to you, the reader.

1.2 This project's existence

This project consists of developing an FPGA-based QPU interfacing unit, which will be used to execute quantum calculations on a live QPU. By parts, the unit will consist of an 8-channel AWG (arbitrary waveform generator), an 8-channel sampling digitiser, and an Ethernet control interface. This will be achieved using four major components, each consisting of several VHDL modules.

1. An Ethernet interface to a host PC.
2. A central datastream controller.
3. An upconversion stage to a DAC.
4. A downconversion stage from an ADC.

A detailed specification of these modules is given in a separate document. In a very general and brief description: the Ethernet interface will receive data packages from a host PC, and stream these onto the central stream controller. The stream controller will send said stream onto the upconversion DAC while simultaneously reading the downconversion stream from the ADC. Parts of this read stream will be sent back to the host PC via the Ethernet controller. All of these major components feature additional specifications relating to their submodules and their functions.

Do note that the interface unit is executive; it does not compile quantum algorithms into quantum gates, it does not optimise said gate sequence using quantum circuit minimisation, it does not perform gate sequence lookup to acquire microwave pulses that are to be sent onto the DAC. These operations are often better done higher up in the software stack, a consensus reached by the quantum computing community at large as recently as in the last few years.

The main challenge of this project, apart from knowledge acquiral and implementation of necessary modules, will likely lie in the massive dataflow handling as the developed modules are scaled up to the target FPGA.

1.3 Target delivery to the product owner

The goal of this project ('the product') is to deliver a package of VHDL modules which when synthesised generates a QPU interface unit for qubit control and readout. The target FPGA is the Xilinx Zync Ultrascale+ RFSoC, although every single module has been specifically designed in such a way that you should be able to design and test them using the Nexys 4 DDR development board. The delivery package is expected to contain supportive documentation onto the synthesised product's design, function and operation. Also, said package is expected to include a usable copy of assisting programs that are required to run and reconfigure the device. Typically, this might include some Python script written to convert filter tap values into BRAM content injected into the FPGA during synthesis. The source code of such programs is explicitly not required, although their runnability is. A 'finished' major component is expected to be tested, verified and working.

The ultimate seal of approval is given to the HDL delivery which passes the final functional verification. Ie. the HDL is actively running an interface to a live quantum processor at QTL, MC2, and can be used to run quantum experiments by its operator.

Apart from the target goal of this project, finishing any of the four major components outlined in subsection 1.2 results in an actual contribution into quantum computing research after this course is over. We are more interested in knowing where the system components choke and break, rather than operating your modules in our day-to-day experiments.

1.4 Benefactors and project transparency

This subsection has been included as a genuine effort of transparency. Ask yourself, where does your code end up after this project is finished?

The specification has been laid out in such a way that your code is not simply copy-pasteable into a commercial device. However, the extent of your ideas (ie. *we implemented A using method B, which meant that C had to work too hard to be usable for D*) will influence the workflow of Ph.D. students and their research, one of which is actively developing a commercial device.

Neither of us have the intent to grab your code after the project is over and send it to the patent office. Neither do we have the intent to stuff your code into a cruise missile. We **do** have the intent to look at your findings, and put less effort on "getting D to work with C" if A has to be implemented using B (see comparison above).

1.5 Risk mitigation

The quantum processing done at MC2 lies mainly in the superconducting regime. This implies cryogenic temperatures, typically 7 mK at the QPU core. This means that it is very hard to actively break a state-of-the-art QPU core and delay research on a global scale, by for instance applying the output from a very buggy VHDL module onto the DAC.

However, it is not impossible - given free access to the DUT wiring. Which is mainly why final stages of verification towards an actual QPU is to be handled by an experienced operator. There are components inside the quantum processor that are only manufactured in one facility on the planet, and there are other components inside requiring Pentagon-affiliated export control paperwork and about six months lead time.

1.6 Further reading for the interested student

Granted, some of you have likely selected this project out of the sheer interest for quantum computing. The first important factoid to know is that there are many different flavours of quantum processing, that in turn differ substantially more than for instance a MIPS architecture differs from a PowerPC one. Depending on who you ask, this stems from the fact that *any* quantum two-level system could be used for quantum computation in theory. The remainder of this document will outline links to the interested student who wishes to delve deeper into the applied theory of quantum computing.

The master's thesis report *Instrument and measurement automation for classical control of a multi-qubit quantum processor* [3] explains a bit further the underlying theory, very specifically targeting EESD students. Pages 1-31 are relevant whether your interest in quantum processing is high or low. The beginning of chapter 3 includes a circuit diagram outlining the QPU core, very similar to the one you will be operating in the final testing stages of this project.

The article *A Quantum Engineer's Guide to Superconducting Qubits* [4] was written specifically for new engineers in the quantum technology area, and is somewhat focused on explaining the role of concepts known by EESD students. It gives a broad understanding of the current state of the field of quantum computing.

For a beginner friendly entry into the theory, one may take a look at [5] and the introductory chapter of [6].

This dissertation [7] specifically treats quantum calculations from a data processing perspective. Ie. complexity and set theory.

For a classical thought experiment in quantum algorithms, chapter 2 in [8] provides Peter Shor's own thoughts regarding the so-called Shor's algorithm.

The introductory piece on 'Nomenclature and notation' in [9] is useful when looking into articles where editors leave out matrix operations.

For an even further understanding of quantum computing, please help yourself to this list of toolbox terms that are crucial when looking further into quantum computing articles.

- Transmons
- cQED
- Josephson junctions
- Decoherence times T1 and T2
- Energy relaxation time T1
- Dephasing time T2
- SQUID
- iSWAP
- Bloch sphere notation
- Dispersive readout

References

1. A. Bengtsson et al., "Quantum approximate optimization of the exact-cover problem on a superconducting quantum processor", *Physical Review Letters*, Currently unpublished, 2020 [Online]. Available: `https://arxiv.org/abs/1912.10495`.
2. S. Lloyd, "Universal Quantum Simulators", *Science*, vol. 273, no. 5278, pp. 1073-1078, 1996 [Online]. Available: `https://dx.doi.org/10.1126/science.273.5278.1073`
3. C. Križan, "Instrument and measurement automation for classical control of a multi-qubit quantum processor", Master's thesis report. Available: `https://odr.chalmers.se/handle/20.500.12380/300065?locale=sv`

4. P. Krantz, M. Kjaergaard, F. Yan, T. Orlando, S. Gustavsson and W. Oliver, "A Quantum Engineer's Guide to superconducting qubits", *Applied Physics Reviews*, vol. 6, no. 2, p. 021318, 2019 [Online]. Available: `https://dx.doi.org/10.1063/1.5089550`.
5. R. J. Schoelkopf and S. M. Girvin, "Wiring up quantum systems", *Nature*, vol. 451, no. 7179, pp. 664-669, 2008 [Online]. Available: `https://dx.doi.org/10.1038/451664a`
6. I. L. Chuang and Y. Yamamoto, "Simple quantum computer", *Physical Review A*, vol. 52, no. 5, pp. 3489-3496, 1995 [Online]. Available: `https://dx.doi.org/10.1103/physreva.52.3489`
7. M. Dobšíček, "Quantum computing, phase estimation and applications", Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czechia, 2008 [Online]. Available: `https://arxiv.org/pdf/0803.0909.pdf`.
8. P. W. Shor, "Progress in Quantum Algorithms", in *Experimental Aspects of Quantum Computing*, 1st ed., H. O. Everitt, Ed. Research Triangle Gate, NC: Springer Science and Business Media Inc., 2005, pp. 5-12 [Online]. Available: `https://link.springer.com/content/pdf/10.1007%2F0-387-27732-3.pdf`
9. M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*, 10th ed. Cambridge: Cambridge University Press, 2018, pp. 1-11.