
Fraunhofer
FOKUS
Qrisp
High-level programming framework
for quantum computers

What is Qrisp?

Qrisp is a high-level programming language for working with quantum computers. Qrisp is designed to enable programmers to write complex quantum algorithms with the ease of a modern day programming language while still compiling down to the circuit level.

By automating much of the low-level coding duties, such as gate-by-gate assembly or qubit management, we aim to open this field of research to a broad audience of developers. Qrisp is being developed at Fraunhofer FOKUS funded by the German ministry for economic affairs and climate action.

Framework structure overview

The central data structure for abstract quantum programming is the *QuantumVariable*. The lifetime cycle of *QuantumVariables* and other aspects are managed by the QuantumSession class, which manages the interaction with a QPU at the backend. Due to a sophisticated system for managing *QuantumSessions*, typically the user does not have to think about *QuantumSession* objects and can just use *QuantumVariables*.

In many cases raw *QuantumVariables* are not that helpful as they provide very few advanced data processing capabilities due to their generality. *QuantumVariables* can be thought of as the abstract base class of more specific datatypes.

Qrisp provides 4 advanced quantum data types:

1. *QuantumFloat:* a datatype to represent and process numbers to arbitrary precision
2. *QuantumBool:* a datatype to represent boolean values
3. *QuantumChar:* a datatype to represent characters
4. *QuantumString:* a datatype to represent strings

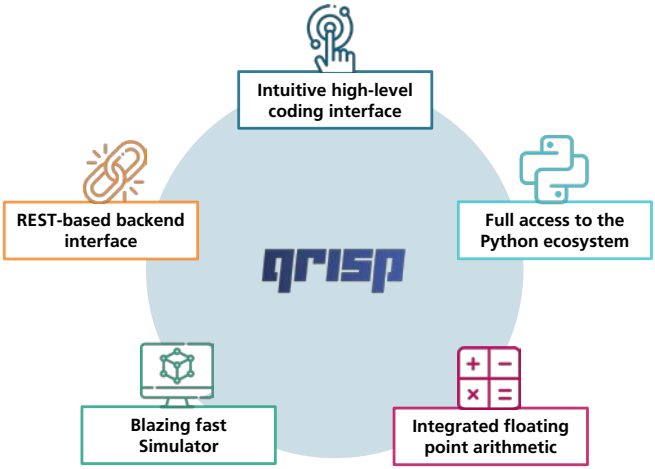


Figure 1: Overview of the Qrisp programming framework

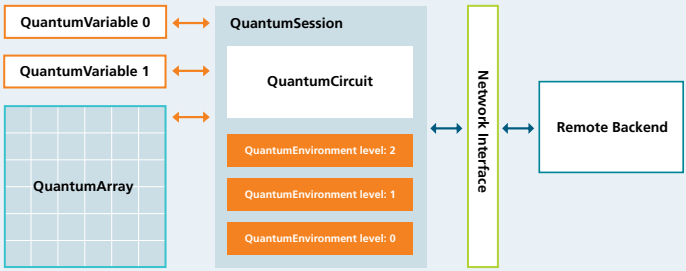


Figure 2: Framework structure overview

QuantumVariables of the same type can be managed in a class called *QuantumArray*. This class provides many convenient and established features like slicing or reshaping.

Using the concept of *Quantum Environments*, it is possible to program using many of the established paradigms from classical computing such as conditional execution of blocks of code (described in *ConditionEnvironment*).

As most of today's research on quantum algorithms has been formulated in terms of quantum circuits, we provide the Circuit Construction module, which allows the construction of *QuantumCircuits*. Constructing *QuantumCircuits* in Qrisp is very similar as in Qiskit since the structure and the naming of the classes and methods are held as close as possible.

To guarantee application-oriented algorithm development at every stage, Qrisp comes with a network interface for addressing remote backends. The way this works is that the backend provider runs a *BackendServe*r on their infrastructure and the user connects via a *BackendClient* object. Note that these classes are only wrappers for an interface generated by state-of-the art interfacing technology. Furthermore, Qrisp supports running it's circuits on the backends of established providers using the *VirtualBackend* class.

»

Algorithm development via manual circuit construction is literally the slowest, least modular and most unstructured approach!«

Raphael Seidel,
Senior Scientist at Business Unit SQC

Why should you use Qrisp over other quantum frameworks?

With Qrisp, you can concentrate on the crucial aspects of your code and reduce the burden of overseeing individual qubits and quantum gates. Due to a sophisticated qubit management system, recycled quantum resources are automatically reused across functions, implying Qrisp code can be modularized effectively. Combined with a typing system, which is smoothly integrated into the Python infrastructure, scalable algorithm development is a straightforward process. Qrisp is lightweight and fast yielding a convenient development workflow.

Conclusion

With Qrisp we open the creation of quantum algorithms to a much broader audience of developers than today. Not only does this lower the entry barrier significantly but open new levels of complexity in quantum algorithms, which might uncover previously unseen quantum advantages. Qrisp is available as an open-source codebase and open for contributions!

More information can be found at:

https://qrisp.eu

https://github.com/fraunhoferfokus/Qrisp

Contact

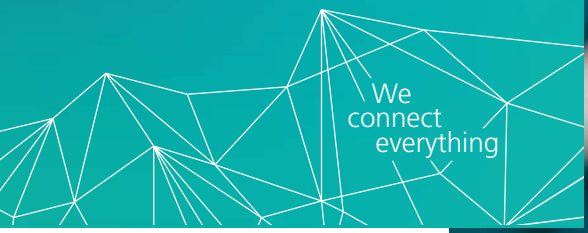
Dr.-Ing. Nikolay Tcholtchev
Head of Quality Engineering for
Urban ICT and Quantum Computing
Business Unit SQC
Tel. +49 30 3463-7175
nikolay.tcholtchev@fokus.fraunhofer.de

Sebastian Bock
Senior Scientist
Business Unit SQC
sebastian.bock@fokus.fraunhofer.de

Raphael Seidel
Senior Scientist
Business Unit SQC
raphael.seidel@fokus.fraunhofer.de

Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31
10589 Berlin

www.fokus.fraunhofer.de/en



mfr | 2309 (Fotos:
istockphoto/ koto_feja