Architecture of a quantum computing platform

Marat Ismagilov, Rustam Sayfutdinov, Alexander Vasiliev
Kazan Federal University, Russia
E-mail: vav.kpfu@gmail.com

Abstract.
We present an architecture of a cloud-based quantum computing platform. It is based on the original model of a nanophotonic quantum processor with integrated quantum memory and a quantum transistor. At the moment the platform works with an integrated quantum simulator that implements the behavior of the underlying model.

1. Introduction
Nowadays there is an interdisciplinary effort worldwide toward scalable quantum computational devices and quantum communication systems, that promise new possibilities for solving computing-intensive or previously not efficiently solvable problems. However, quantum processors can still be used only in restrictive laboratory conditions. Thus, it was a natural decision to propose a cloud-based access to the quantum computing devices and integrate them with a high-performance computing platform. This led to the new area of quantum programming and a rapidly increasing number of companies (including IBM, Google, Microsoft) developing their own quantum computing platforms. The new quantum programming community is seeking for the optimal programming languages and compiler design for realistic (already created or future) quantum hardware [1].

In this research we develop an architecture of a quantum computing platform with the cloud interface, that is based on the original model of a nanophotonic quantum processor with integrated quantum memory and a quantum transistor [2]. At the moment the platform works with an integrated quantum simulator that implements the behavior of the underlying model [3]. As shown earlier this model is universal, i.e. it is capable of performing arbitrary quantum computations. Additionally, it allows to speed up the implementation of controlled operations, that are heavily used in most multi-qubit algorithms.

The platform has two main components: the server-side framework and the client SDK, which gives the possibility of coding the mixture of quantum and classical algorithms in a conventional environment.

2. Preliminaries
Let us recall the computational model given in [3].

A Quantum Processing Unit (QPU) is a device consisting of 4 modules controlled by a classical device as illustrated in Fig. 1. They include:

- An s-qubit Quantum Memory module, that allows to store quantum states. The contents of the memory is described by the current state $|\psi\rangle \in \mathcal{H}^{2^s}$. The quantum memory module
Figure 1. A model of a Quantum Processing Unit (QPU).

is capable of exchanging qubits with quantum transistor and quantum communication channel.

- A Quantum Transistor is a 3-qubit device capable of performing quantum transformations. There are three types of operations described in a physical architecture: QET, PHASE and CQET [5]. Generally, we can consider an array of quantum transistors working in parallel, but here for simplicity we leave just one module.

- Init is an initialization module that can create a quantum state given its classical description and save it to the quantum memory. Without loss of generality we can consider a single-qubit initialization in state $|0\rangle$, since any other state can be obtained via universal set of operations.

- Measure is a measurement module that implements a photon detecting procedure and gives a classical bit out of qubit being measured.

The proposed model is capable of performing arbitrary quantum computation, since the available set of operations provides a possibility to implement CNOT and any single-qubit operation.

A quantum algorithm in this model can be described by a $(t, s)$ quantum program, which is the sequence of $t$ instructions of several types over $s$-qubit quantum memory. These parameters correspond to the key complexity measures: $t$ is the time of computation and $s$ is the required space.

3. Cloud Architecture for QPU
Based on the proposed QPU model we have constructed a cloud-based quantum computing platform. The platform has two main components: the server-side framework and the client-side software development kit.

3.1. Client SDK
We have created a Software Development Kit (SDK) that communicates with the server-side framework and provides the means for quantum programming at the client side.

The Python-based SDK hides most of the complex details and gives abstractions that make coding quantum algorithms as easy as shown in Listing 1.
3.2. Server-side Framework

The server-side Framework provides an application programming interface (API) for the remote clients of the quantum computing platform. Any incoming quantum program is translated to a lower level of abstraction transforming logical operations and logically encoded qubits into physical ones. The translation is performed based on the formal grammar that uses basic operations of the QPU model as the terminal symbols and the conventional quantum operators as the nonterminal ones.

As mentioned earlier, there are three types of computational operations given by the physical architecture [3]: QET, CQET and PHASE. These operations constitute a universal basis and allow to perform arbitrary quantum computation. In particular, any single-qubit operation can be described using following operators [4]:

\begin{align}
U_1 (\mu) &= \begin{pmatrix} 1 & 0 \\ 0 & e^{i\mu} \end{pmatrix} \sim R_z (\mu), \\
U_2 (\varphi, \mu) &= R_z (\varphi + \frac{\pi}{2}) R_x (\frac{\pi}{2}) R_z (\mu - \frac{\pi}{2}), \\
U_3 (\theta, \varphi, \mu) &= R_z (\varphi + 3\pi) R_x (\pi) R_z (\theta + \pi) R_x (\pi) R_z (\mu),
\end{align}

where \( R_z (\theta) \) can be implemented by QET (-\( \theta \)) and \( R_z (\theta) \) is equivalent (in a logical encoding of qubits \(|0\rangle_L = |01\rangle, |1\rangle_L = |10\rangle \)) to PHASE (\( \theta \)) [5].

An entangling operation is given by the CQET [3], which can be used to implement the widely used CNOT operation.

Altogether, the server-side framework maintains access to the following logical gates:
• $R_x, R_z, \text{CNOT};$
• $U_1, U_2, U_3;$
• $R_y, H, S, S^\dagger, T, T^\dagger, \text{CCNOT, X, Y, Z}.$

4. Conclusion
Thus, we have proposed a quantum computing platform with the cloud interface based on the mathematical model of the Quantum Processing Unit (QPU). This platform contributes to the area of quantum programming providing API to the unique model of a nanophotonic quantum processor with integrated quantum memory and a quantum transistor, that has certain advantages for implementation of controlled transformations.

Acknowledgments
The research was funded by the subsidy allocated to Kazan Federal University for the state assignment in the sphere of scientific activities, project # 1.12878.2018/12.1.

References
[1] Chong F T, Franklin D and Martonosi M 2017 Nature 549 180–7
[2] Moiseev S A and Andrianov S N 2016 Optics and Spectroscopy 121 886–96
[3] Ablayev F, Andrianov S, Fetisov D, Moiseev S, Terentyev A, Urmanchev A and Vasiliev A 2018 AIP Conference Proceedings 1936 020014
[4] Cross A W, Bishop L S, Smolin J A and Gambetta J M 2017 Preprint quant-ph/1707.03429
[5] Ablayev F M, Andrianov S N, Moiseev S A and Vasiliev A V 2013 Lobachevskii Journal of Mathematics 34 291–303