**PROBLEMY MECHATRONIKI**
**UZBROJENIE, LOTNICTWO, INŻYNIERIA BEZPIECZEŃSTWA**

**PROBLEMS OF MECHATRONICS**
**ARMAMENT, AVIATION, SAFETY ENGINEERING**

**ISSN 2081-5891; E-ISSN 2720-5266** **https://promechjournal.pl/**

*Research paper*

# Test Environment for Verifying Quantum Algorithms and Controlling SINARA Real-time Modules

Michał P. WYROSTKIEWICZ* (wyrostkiewiczm@witu.mil.pl)
Tomasz KUCZERSKI (kuczerskit@witu.mil.pl)
Dariusz GIBALSKI (gibalskid@witu.mil.pl)

*Corresponding author
ORCID: https://orcid.org/0000-0003-3680-429X

*Military Institute of Armament Technology,*
*7 Prymasa Stefana Wyszyńskiego Str., 05-220 Zielonka, Poland*

*Received: August 1, 2022 / Revised: September 8, 2022 / Accepted: October 13, 2022 / Published: March 31, 2024.*

**2024,** 15 (1), 55-64; https://doi.org/10.5604/01.3001.0054.4488

**Cite: Chicago Style**

Wyrostkiewicz, P. Michał, Tomasz Kuczerski, and Dariusz Gibalski. 2024. "Test Environment for Verifying Quantum Algorithms and Controlling SINARA Real-time Modules". *Problemy mechatroniki. Uzbrojenie, lotnictwo, inżynieria bezpieczeństwa* / *Probl. Mechatronics. Armament Aviat. Saf. Eng.* 15 (1) : 55-64. https://doi.org/10.5604/01.3001.0054.4488

**Abstract.** This paper describes the process of creating and using a quantum computing environment. It also discusses the manner in which first quantum experiments using mathematical models, simulators or commercially available quantum computers may be conducted. In addition, it explains how the available quantum algorithms may be understood and applied. Finally, the role of real-time systems and their control in the architecture of a quantum computer is described.
**Keywords:** quantum computing, quantum experiments, quantum algorithms, quantum computer simulator, quantum computer, real-time system

## 1. INTRODUCTION

Since the discovery of quantum physics - an area which has changed our understanding of the universe - quantum mechanics has become an accurate and experimentally consistent scientific theory [1]. Thanks to quantum physics effects such as entanglement or superposition, quantum computers are capable of performing calculations that are not possible in the world of classical mechanics, and hence enjoy a powerful advantage over classic computers [2]. Currently, they are mainly used in applications in which the use of classical algorithms that work in theory calls for significant computational or time-related resources, often reaching thousands of years, which makes such an approach completely impractical. However, the first steps in the field of quantum algorithms may seem overwhelming. This is because software engineers are typically not trained in topics such as transistor physics or computational complexity theory, whose philosophy is much closer to that of a quantum computer. Therefore, this article reviews various simulation environments, detailing their specific advantages. The basic issues of quantum computing, as well as those related to designing and using quantum algorithms are presented as well. In addition, a set of Sinara cards with DRTIO Artiq real-time software is described, which is used to control individual hardware modules.

## 2. CREATION OF EXECUTION ENVIRONMENTS AND THEIR ADVANTAGES

The point of departure is to find the right tool that will be used for achieving the objective. There are many alternatives to quantum programming, such as Qiskit [3], Cirq [4], PyQuil [5] and ProjectQ [6], with Qiskit, Cirq, and PyQuil being better suited to use actual quantum devices.

### 2.1. Qiskit – IBM

Through the IBM Quantum Experience initiative, IBM has been providing all users, since 2016, with remote access to a number of superconductivity-based quantum computers.

It was the first platform to allow access to their computers, and it is the one on which the most documented experiments have been conducted to date.

Currently, each user has access to six real quantum computers (two 7-qubit computers and four 5-qubit computers) as well as five simulators with a higher number of qubits. Qiskit, on the other hand, is an open-source project developed in Python that consists of the following specialized modules:

- Qiskit Terra: contains basic modules and is used for porting functionalities to the OpenQASM platform for interacting with real quantum computers.
- Qiskit Ignis: provides tools for quantum noise characterization, parameterization of the equipment used, etc.
- Qiskit Aer: is responsible for implementing a simulator using quantum noise modeling.
- Qiskit Aqua: provides a low level of abstraction for quantum gates, making it possible to use previously imported algorithms to build high-level applications.
- Qiskit IBMQ Provider: provides remote access to IBM quantum computers

## 2.2. Cirq – Google

Cirq is an open-source project developed by Google to design and simulate quantum circuits. It is a Python library designed specifically for NISQ (Noisy Intermediate-Scale Quantum) systems. Cirq differs from other solutions by seeking to make the details of the hardware available to developers, rather than creating a layer of abstraction. Operations performed in Cirq, on the other hand, are intended to program circuit design functions which are entered as an input argument to a locally operated 25-qubit simulator, a quantum computer or another algorithm analysis system.

## 2.3. PyQuil – Rigetti Computing

Rigetti Computing has developed the "Forest" platform that allows developers to simulate quantum circuits using the Quantum Instruction Language (Quil) module. Forest allows circuits consisting of up to 16 qubits to be executed both in the simulator containing the platform, a QVM (Quantum Virtual Machine), and in a real QPU (Quantum Processing Unit) known as Aspen-11. The Foresta simulator, on the other hand, focuses on incremental optimization of the program intended to run on real hardware. To this end, QVM implements a series of debugging levels, from the highest to the lowest level of ideality, which are ultimately fed into the quantum computer. The QPU is accessed via the Rigetti Quantum Cloud Services platform which provides remote access to the quantum chip. To run the algorithms on a quantum computer, both time and computing resources must be reserved.

Reservation of at least one part of the processor, referred to as the grid, which defines the connection between two or more QPUs, is made by using purchased credits. After querying the platform about access time and access window duration, the program returns various available grids, specifying the number of qubits.

### 2.4. ProjectQ – ETH Zurich

ProjectQ, like other platforms, is a Python-based open-source system. ProjectQ allows algorithms to run on real quantum computers, and integrates a C++-based simulator with up to 27 qubits. ProjectQ's syntax is quite simple compared to other platforms and delegates the complexity of translation to a lower level of the compiler which is responsible for generating execution code. One feature worth highlighting is the graphical circuit display function implemented by ProjectQ, which is done via LaTeX. For this purpose, Circuit Drawer is declared as a backend. Additionally, ProjectQ has a very useful resource counter for multi-element circuits.

## 3. QUANTUM ALGORITHMS

A quantum algorithm [7] is a sequence of operations on a quantum system, allowed by quantum mechanics (quantum gates) which, by transforming the input data encoded in the initial state, allow to obtain the desired result.
Examples of the best-known quantum algorithms include:

- Deutsch-Jozsa algorithm (distinguishing between constant and balanced functions)
- Shor (factorization) algorithm
- Grover algorithm (database search)
- Quantum Fast Fourier transform
- Quantum phase estimation algorithm

Quantum gates are most often written in a symbolic form (by assigning an appropriate gate symbol specifying its function) or in the form of a matrix. Through their planned use, they enable a controlled evolution of the quantum system and since each such operation is reversible, the evolution of the entire system is reversible as well.
In order to achieve the goal of performing a quantum algorithm, the designer tries to choose the right sequence of gates based, usually, on the following sequence:

- Initialization of qubits up to a fixed initial value (usually $|00...0\rangle$ ) with erasure of the history of their previous states.
- Implementation, on qubits, of a quantum gate sequence suitable for a given algorithm (planned evolution of the quantum register).

- Achievement of the intended state of qubits and their measurement, reduction of all their wave functions, while returning a binary measurement result (0 or 1). It follows from the quantum laws that at this point all the properties of the register achieved through its quantum evolution are destroyed.
- To obtain the required accuracy of the results, the sequence is repeated several times, resulting in statistics for each qubit that must be interpreted in the context of the implemented algorithm.

The advantages of quantum algorithms can be presented using the example of factorization. Currently, GNFS (General Number Field Sieve) is the best factorization algorithm, having a complexity of $\exp(O(n^{1/3}(\log n)^{2/3}))$, while the Shor algorithm, given a sufficient number of qubits, is able to solve such cases with a complexity of $O(n^3)$. When comparing, roughly, their operating times (table 1):

Table 1. Comparison of the number of time steps required for factorization of a number depending on its length [7]

| Number of digits of a natural number | Shor quantum algorithm | GNFS algorithm |
|---|---|---|
| 100 | 1E+06 | ~4E+05 |
| 1,000 | 1E+09 | ~5E+15 |
| 10,000 | 1E+12 | ~1E+43 |

one may notice a much lower increase in the computation time required for factorization. Working on a number comprising 10,000 digits, a classical computer performing 1E+17 operations per second would take about 3E+17 years to complete the task, while a quantum computer with a 1MHz processor would take only 11 days to do so. In addition, issues related to quantum algorithms are attracting increasing interest, as evidenced by the rapidly increasing number of scientific papers on the subject. In recent years, their number has increased from 278 (2015) to as many as 434 [8]

## 3.1. What is a qubit

Qubits or qbits are the basic concept of quantum computing. It may be considered a specific, minimal case of a Schrödinger equation solution, where the wave function assumes only two states representing, for example, the spin of an electron or the polarization of a photon. The essential similarity between a bit and a qubit comes down to the phenomena that take place when reading a bit and measuring a qubit. In both cases, the final result is a value of 0 or 1, interpreted as TRUE or FALSE.

Qubit, on the other hand, remains in a superposition of quantum states before it is measured, not representing one particular value. It is therefore simultaneously present in state $|0\rangle$ and $|1\rangle$. The extent to which the $|0\rangle$ state is preferred over the $|1\rangle$ state is usually the target qubit reading and, therefore, the result of quantum computing. However, multiple measurements establishing different state ratios are necessary to accurately determine the value of the result.

Nevertheless, the reading of a qubit's state involves destroying its superposition state by reducing the qubit's wave function from a complex form to a single state corresponding to the measurement result. Thus, a qubit behaves quite differently during its evolution that stems from it being influenced by quantum gates, compared to its behavior during reading, which distinguishes it from a bit that assumes a fixed value of 0 or 1 at all times.

## 3.2. Basic quantum gates

Quantum gates [9] are the primary tool with which quantum computers perform their calculations. Such a gate may have the form of any process that affects the final state of one or more qubits in a planned and controlled manner.

By determining a baseline consisting of basic quantum gates, it is possible to construct any other quantum gate based thereon. Such basic gates are:

- Pauli-x gate (sigma-x)
- Hadamard gate
- Phase gate
- CNOT gate

### 3.2.1. Pauli-X gate

The Pauli-X gate is a single-qubit gate (also called a sigma-x gate or NOT gate) that transforms the source qubit on the Bloch sphere by $\pi$ radians with respect to the X axis (swapping the probability amplitudes of the quantum states $|0\rangle$ and $|1\rangle$ with each other for a given qubit).
The matrix of a quantum NOT gate can be written in the following form:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

### 3.2.2. Hadamard gate

The Hadamard gate is responsible for summing and differentiating states $|0\rangle$ and $|1\rangle$ of qubits. Its matrix can be written in the following form:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

The operation of this gate impacts the basic states of the qubit in the following manner:

$$|0\rangle \rightarrow \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$
$$|1\rangle \rightarrow \frac{|1\rangle - |0\rangle}{\sqrt{2}}$$

### 3.2.3. Phase gate

The phase gate is a universal quantum gate responsible for changing the phase of a qubit by a value of $\varphi$. In general form, its matrix takes the following form:

$$R_{\varphi} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{bmatrix}$$

Specifically, cases for the following angles are considered$\varphi$:

$$R_{\varphi=\pi} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = Z$$
$$R_{\varphi=\frac{\pi}{2}} = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{2}} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} = \sqrt{Z} = S$$

### 3.2.4. CNOT gate

CNOT (Controlled NOT) gate is a two-qubit gate operating on a quantum register consisting of two qubits. Since its effect consists in entangling or disentangling qubits, it is extremely important in the construction of quantum circuits. The CNOT gate reverses the value of the target qubit only when the control qubit is in the quantum state $|1\rangle$:

| Before | | After | |
|---|---|---|---|
| $\|0\rangle$ | $\|0\rangle$ | $\|0\rangle$ | $\|0\rangle$ |
| $\|0\rangle$ | $\|1\rangle$ | $\|0\rangle$ | $\|1\rangle$ |
| $\|1\rangle$ | $\|0\rangle$ | $\|1\rangle$ | $\|1\rangle$ |
| $\|1\rangle$ | $\|1\rangle$ | $\|1\rangle$ | $\|0\rangle$ |

And in the matrix form:

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

## 3.3. Quantum Fourier Transform

The Quantum Fourier Transform is used in a wide range of both quantum and hybrid algorithms. Although it is the quantum equivalent of the Discrete Fourier Transform, its application is influenced by how classical problems are translated into their quantum form [10]

### 3.3.1. Theoretical introduction

The Quantum Fourier Transform (QFT) [10] consists in quantum mapping of the Discrete Fourier Transform that calculates the amplitudes of a wave function. It is a fundamental component of many other algorithms, such as the Shor algorithm or the quantum phase estimation algorithm. In a similar way as the Discrete Fourier Transform, it acts on vector $(x_0, x_1, \dots, x_N)$ mapping it to vector $(y_0, y_1, \dots, y_N)$, according to the following formula:

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \omega_N^{jk}$$

where $\omega_N^{jk} = e^{2\pi i \frac{jk}{N}}$

The Quantum Fourier Transform impacts the quantum state in a corresponding manner,$\sum_{i=0}^{N-1} x_i |i\rangle$ mapping it to a quantum state $\sum_{i=0}^{N-1} y_i |i\rangle$ described as a map:

$$|x\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega_N^{xy} |y\rangle$$

or in the form of a unitary matrix:

$$U_{QFT} = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \omega_N^{xy} |y\rangle\langle x|$$

which, when simplified, gives the following output:

$$|y_1 y_2 \dots y_N\rangle = QFT(|x_1 x_2 \dots x_N\rangle) = \frac{1}{\sqrt{2^N}} \bigotimes_{j=1}^{N} \left(|0\rangle + e^{2\pi i \frac{x}{2^j}} |1\rangle\right)$$

where $\otimes$ stands for the Kronecker (tensor) product [10]

### 3.3.2. Implementation

The system implementing the Quantum Fourier Transform for n qubits uses Hadamard H gates (described in Section 3.1.2) and controlled phase change gates $CR_m$ whose general matrix representations are as follows:

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \qquad CR_m = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & e^{\frac{2\pi i}{2^m}}\end{bmatrix}$$

After the number is transformed into binary format, it can undergo QFT operations according to a scheme built with the Qiskit library used for quantum computing [Figure 1]

## 4. SINARA REAL-TIME SYSTEMS

Sinara is an open-source ecosystem used in experiments and quantum physics applications. The advantages it offers are the collaborative operation of multiple-need centers, access to all schemes and a wide field of applications. The FPGA (Field Programmable Gate Array) boards of such a design are created specifically for the needs of laboratories and centers using this platform. Recently, it has enjoyed applications in both superconducting [11] and trapped-ion quantum computer designs.

The commonly used system for communicating with the boards is the ARTIQ control system (Advanced Real-Time Infrastructure for Quantum physics) [12]. It is an open-source project that creates a Python-based high-level programming language which is compiled and executed on dedicated FPGA boards with nanosecond resolution capability and less than one-millisecond latency. The main advantage of using this system is the so-called DRTIO (Distributed Real Time Input-Output) protocol that provides gigabit connectivity and time distribution over a dedicated copper or fiber optic cable. This allows to synchronize the clocks of all devices, ensuring a deterministic phase relationship, and nanosecond resolution of input-output events on all devices used in the experiment.

## 5. CONCLUSIONS

The purpose of the paper was to provide an overview of the available environments for experiments focusing on quantum algorithms, and the methodology for their design, as well as to describe a set of real-time Sinara cards used to control the experiments performed. The authors of the paper hope that the project will contribute to boosting interest in new quantum experiments as well as in developments taking place in this particular field.

## FUNDING

The authors received no financial support for the research, authorship, and/or publication of this article.

## REFERENCES

[1] Nielsen, A. Michael, and Isaac L. Chuang. 2002. "Quantum computation and quantum information". Cambridge University Press.
[2] Bennett, H. Charles, and David P. DiVincenzo. 2000. "Quantum information and computation". *Nature* 404 : 247-255.
[3] *Qiskit.* https://qiskit.org/ (accessed July 8, 2022).
[4] *Cirq*. https://quantumai.google/cirq (accessed July 8, 2022).
[5] *Welcome to the Docs for pyQuil*! https://pyquil-docs.rigetti.com/en/stable/ (accessed July 8, 2022).
[6] *ProjectQ - An open source software framework for quantum computing*. https://github.com/ProjectQ-Framework/ProjectQ (accessed July 8, 2022)
[7] Montanaro, Ashley. 2016. "Quantum algorithms: an overview". *npj Quantum Information* 2 : 15023-1-8.
[8] *Quantum Algorithm Zoo, Algebraic and Number Theoretic Algorithms*. https://quantumalgorithmzoo.org.
[9] Brylinski, Jean-Luc, and Ranee Brylinski. 2002. "Universal quantum gates." Mathematics of quantum computation. Chapman and Hall/CRC. 117-134.
[10] Weinstein, S. Yaakov et al. 2001. "Implementation of the quantum Fourier transform". *Physical review letters* 86.9 : 1889.
[11] Ryan, A. Colm, et al. 2017. "Hardware for dynamic quantum computing." *Review of Scientific Instruments* 88.10 : 104703.
[12] *ARTIQ v6.7646.759f0041 documentation.* http://m-labs.hk/artiq/manual/index.html (accessed July 10, 2022).

# Środowisko testowe do badania algorytmów kwantowych i sterowania modułami czasu rzeczywistego SINARA

Michał P. WYROSTKIEWICZ, Tomasz KUCZERSKI, Dariusz GIBALSKI

*Wojskowy Instytut Techniczny Uzbrojenia*
*ul. Prymasa Stefana Wyszyńskiego 7, 05-220 Zielonka*

**Streszczenie.** Prezentacja o tworzeniu i wykorzystywaniu środowiska do obliczeń kwantowych. Omówione są również, w jaki sposób przeprowadzić pierwsze eksperymenty kwantowe na poziomie matematycznym, symulatora lub dostępnych komercyjnie komputerów kwantowych. Dodatkowo wyjaśnione jest w jaki sposób rozumieć oraz wykorzystywać dostępne algorytmy kwantowe. Na koniec, opisano rolę systemów czasu rzeczywistego i ich sterowania w architekturze komputera kwantowego.
**Słowa kluczowe:** obliczenia kwantowe, eksperymenty kwantowe, algorytmy kwantowe, symulator komputera kwantowego, komputer kwantowy, system czasu rzeczywistego