Implementation of Shor's Algorithm in a Digital Quantum Coprocessor

Valeriy Hlukhov [1]

[1] *Lviv Polytechnic National University, 12 Bandera str., Lviv, 29013, Ukraine*

Abstract

The advantages of digital quantum coprocessors include a larger quantum volume, normal operating conditions, the presence of memory, the presence of a tested and reliable element base on which they can be implemented, and the availability of technology for using this element base. The element base refers to field programmable gate arrays (FPGAs). The paper presents the principles of building digital quantum gates, digital qubits and both homogeneous and heterogeneous digital quantum coprocessors. The capabilities of real quantum computers are usually illustrated by performing factorization of the number 15 using Shor's algorithm. This paper describes the implementation of quantum Shor's algorithm for factorizing the number 15 in a digital quantum coprocessor, which is implemented in FPGA. The difference between a real quantum coprocessor and a digital one is shown. A technique for determining the characteristics of a digital quantum coprocessor is described. Its probabilistic characteristics are also given.

Keywords

Digital qubit, digital quantum coprocessor, heterogeneous coprocessor, homogeneous coprocessor, Shor's algorithm, FPGA

1. Introduction

A quantum computer is a heterogeneous device [1] that consists of a classical control computer and its quantum accelerator [2] - a quantum coprocessor. Real quantum coprocessors are analog and probabilistic devices. They consist of qubits, quantum gates provide a change in their states. A classical computer controls the operation of a quantum coprocessor, checks the correctness of the results of its work, and in case of an incorrect result, it restarts the coprocessor to work.

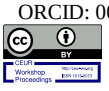
The possibility of creating logical (digital) probabilistic devices that can work according to the same formulas as real quantum coprocessors and can implement quantum algorithms is shown in previous works [3], [4]. The possibility of creating digital quantum gates, digital qubits and, based on them, digital quantum coprocessors is shown. The hardware base for digital quantum coprocessors is FPGA. Unlike real quantum coprocessors, digital ones operate at normal temperatures (like classical computers) and have a larger quantum volume. This makes the development of such coprocessors actual and important.

Quantum computers possible field use is large numbers factorization [5], [6]. This operation is used to hack information security systems that use public key algorithms, such as RSA [7]. Shor's algorithm [8] is used for this. The main elements of the quantum coprocessor that implements Shor's algorithm are Hadamard elements, quantum Fourier transform, modular exponentiation [9], and qubit state meters. In real quantum coprocessors, these elements (except for meters) consist of qubits; changes in their states are provided by quantum gates.

In previous works, the implementation of digital Hadamard elements and digital quantum Fourier transform on FPGAs was shown [3], [4]. In one FPGA, it is possible to create a quantum Fourier transform from thousands of digital qubits. The internal state of a digital qubit can be represented by binary code θ in the range from 0.00...0 to 1.00...0. Also, options for encoding the states of digital qubits with binary codes of

ISIT 2021: II International Scientific and Practical Conference «Intellectual Systems and Information Technologies», September 13–19, 2021, Odesa, Ukraine

EMAIL: Glukhov@polynet.lviv.ua (A. 1)
ORCID: 0000-0002-0542-7447 (A. 1)



various lengths - from 3 to 32 bits were considered [3], [4].

Simulation of individual quantum gates is used to simulate quantum algorithms. To simulate the reversibility of a qubit, models of reversible logic architecture [10] and gates [11] have been developed. In this work, more complex logic circuits are simulated to ensure reversibility of quantum circuits.

2. Purpose of work

The aim of the work is to show the possibility of performing quantum algorithms (using the example of factoring the number 15 by Shor's factorization algorithm) in a digital quantum coprocessor implemented on the FPGA. For this, the possibility of implementing modular exponentiation on the FPGA and the possibility of the effect of the results of this operation on the states of digital qubits is shown, which allows us to determine the period of $y = a^x mod M$ function (determining the period of a function is the main task of a quantum coprocessor in the implementation of Shor's algorithm).

3. Qubit

Qubit quantum state $|\psi\rangle$ can be represented (Figure 2) as a simple displacement of end point of unit radius [12]. The probability p_j of obtaining state $|j\rangle$ as a result of quantum state $|\psi\rangle$ measurement is equal to $p_j = \lambda_j^2$. In this case, the sum of all probabilities $P = \sum_{j=0}^{N-1} \lambda_i^2 = 1$. In unit circle (Figure 2) which is used in [4] $p_0 = cos^2\theta$ and $p_1 = sin^2\theta$ respectively.

4. Digital gates, qubits and quantum coprocessors

A digital quantum gate that is used to change the state of a digital qubit can be represented as a logic circuit Figure 1.

The digital quantum gate includes an ALU, a comparator, and a pipelined register.

ALU transforms the code of the previous state DataIn of the qubit under the influence of the Instruction with the possible use of the measured state $|Q_I\rangle$ of the neighboring qubit (or states of qubits). The new DataO status code is compared in a comparator with the random variable Asin_f to obtain the measured state of the qubit $|Q_O\rangle'$. The output of the gate is the qubit state code DataOut and the measured state $|Q_O\rangle$ of the qubit, which are taken from the output of the pipeline register.

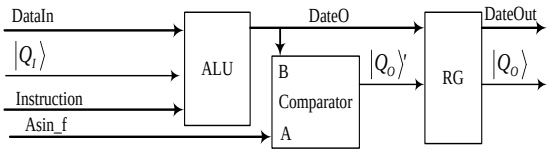


Figure 1: A digital quantum gate QGate [1]

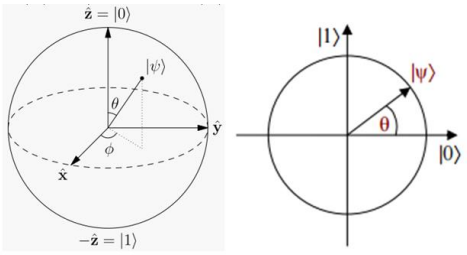


Figure 2: Bloch sphere for qubit complex amplitudes (left) and a unit circle for real ones (right)

In a heterogeneous digital quantum coprocessor, a random variable at the input of each digital quantum gate is generated by a separate pseudo-random code generator (PRNG) and a Read-Only-Memory (ROM) based functional converter. The converter changes the random variable A according to the formula $Asin\_f = D = arcsin\sqrt{A}$ (Figure 3).

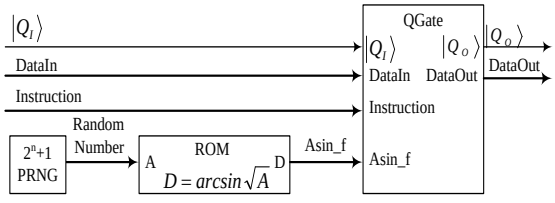


Figure 3: A digital quantum cell DQCell for heterogonous digital quantum coprocessor [3]

Digital qubit circuit for a heterogeneous coprocessor Figure 4 will represent a series connection of several digital quantum cells Figure 3.

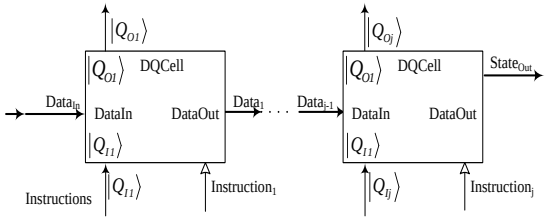


Figure 4: A digital quantum qubit as line (QLine) of DQCells for heterogonous digital quantum coprocessor

Digital qubit circuit for a homogeneous coprocessor Figure 5 has only one difference in comparison with the circuit in Figure 4: all random variables Asin_f for each digital quantum gate are generated using one pseudo-random code generator and one functional converter.

A schematic diagram of a digital quantum coprocessor is shown in Figure 6. In heterogeneous coprocessor, the number of pseudo-random code generators and functional transformers coincides with the number of digital quantum gates. Both oscillators and transformers are located near the digital quantum gates (Figure 3) inside the digital qubits of the Qline circuit in Figure 6.

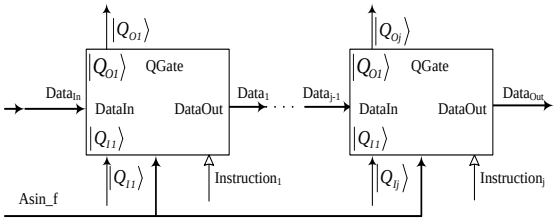


Figure 5: A digital quantum qubit as line (QLine) of DQGates for homogenous digital quantum coprocessor

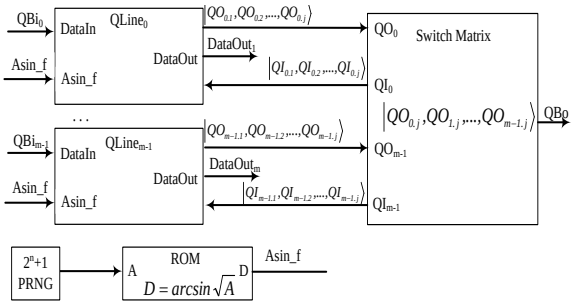


Figure 6: A generalized functional diagram of a digital quantum coprocessor

5. Shor's algorithm

In Shor's algorithm [8], the problem of factorizing the number M is reduced to the problem of determining the period r of the function $y = a^x mod M$, which is calculated by the controlled units CU (Figure 7), where a is an arbitrary integer. This is precisely the problem that a quantum computer solves. It is shown that the greatest common divisor $GCD(a^{r/2}+1, M)$ can be a divisor of the number M. The subsequent finding of the greatest common divisor is performed by a classical computer.

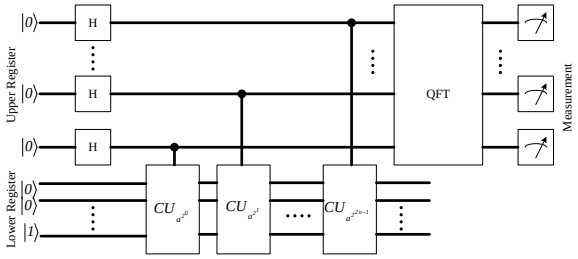


Figure 7: Shor's algorithm implementation in real quantum computer

If $n = \lceil log_2 N \rceil$ is the required number of bits to represent the number N to be factored than the upper quantum register in Figure 7 requires at least $2n$ qubits, because Shor's algorithm requires x to take values between 0 and at least N^2 and the modular exponentiation function can be written [9] as

$$a^x \bmod M = (a^{2^0} \bmod M)^{x_0} (a^{2^1} \bmod M)^{x_1} \cdots (a^{2^{2n-1}} \bmod M)^{x_{2n-1}} \quad (1)$$

Figure 7 of Shor's algorithm implementation illustrates quantum superiority very well. If we take only the upper part (Figure 8) of Figure 7 diagram, then the quantum Fourier transform will determine the frequency of the white noise that the Hadamard elements create. After the initial reset, each Hadamard element transfers the qubit to the neutral position, when the angle $\theta = \pi/4$ and the state of each qubit with the same probability $p_0 = p_1 = 0{,}5$ can be measured both as 0 and as 1. And the measured state of the upper register in Figure 8 can take any value from 0 to 2^{2n}-1. The state spectrum of the upper register will include all 2^{2n} states.

Figure 8: White noise $|X\rangle$ generated by a quantum circuit

Let's conduct a thought experiment - imagine that in some way with a period *t* we find out the state of the upper register without changing states of its qubits. Each time we will receive a new state code, the possible codes will be in the range from 0 to 2^{2n}-1. Now imagine that *t* runs to 0. Then at each moment of time the state of the upper register will contain all codes in the range from 0 to 2^{2n}-1.

If, on the other hand, modular exponentiation is performed over the outputs of the upper register (CU in Figure 7), then at each "moment" only states that give the same result of modular exponentiation at the output of the CU be in the spectrum of upper register states. That is, at each "moment" of time, the spectrum of states will be different. For example, for the function $y = 2^x mod15$ spectrum is presented in Figure 9.

At some "moment" at the output of CU there will be a result $y = 1$, then in the spectrum of upper register states there will be 0, 4, 8, 12, … codes (Figure 10). The distance between the same codes, that is, the period of $y = 2^x mod15$ function will be equal to $r = 4$. This period will be determined using the quantum Fourier transform (as the reciprocal of the repetition rate *F* of the extracted codes $r = 1/F$). At another "moment", the CU output will have the result $y = 4$, then in the spectrum of upper register states there will be 2, 6, 10, 14, ... codes (Figure 11).

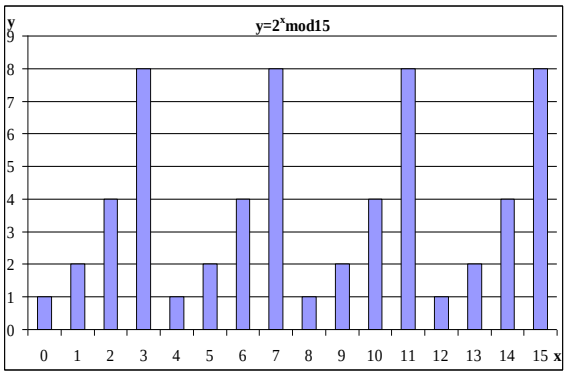


Figure 9: Modular exponentiation

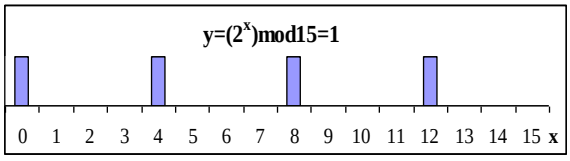


Figure 10: X codes for which $y = 2^x mod15 = 1$

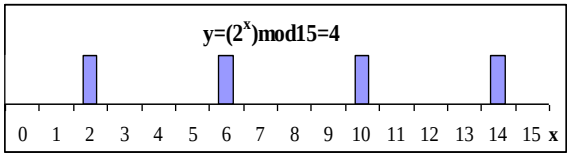


Figure 11: X codes for which $y = 2^x mod15 = 4$

But the period of $y = 2^x mod15$ function will still be $r = 4$. And it will again be determined using the quantum Fourier transform. Whatever the result at the CU output, in the corresponding spectrum of states there will be only codes that allow determining the period of the function $y = 2^x mod15$ in one measurement and this period will be $r = 4$. Determining the period of a function in one measurement illustrates quantum superiority. This does not take into account the time to create a quantum circuit Figure 7.

The period of the $y = a^x modM$ function can be determined not only by quantum, but also by logical (digital) methods in several clock cycles. For example, in [5], [6] it is proposed to approach the solution of the problem from the other side: fix $r = 2$, and determine the random variable a from the given $r = 2$. With this approach, a quantum computer is no longer needed. But Shor's algorithm is convenient for demonstrating quantum superiority using separate examples - for determining the result in one measurement.

Also, one of the limitations of Shor's algorithm is the requirement for the parity of the period *r*. It was shown in [5], [6] that this condition is optional. The period *r* can be odd if *a* is a square.

Once again, we recall that all the "moments" in a quantum computer are one and the same moment in time (Figure 10, Figure 11).

Attaching (Figure 7) an additional circuit to the upper register changes the state spectrum at any given "moment" in time. This is similar to a high-pass filter in analog technology - connecting a capacitor removes high frequencies from the spectrum, removes interference.

Convenient examples are used to illustrate the quantum superiority in determining the period of

$y = a^x mod M$ function. In the example considered earlier, when M = 15 = 3 * 5, period r is power of 2: $r = 4 = 2^2$ and both factors are Fourier primes, they can be represented as $2^{2^m}+1$: $3 = 2^{2^0}+1, 5 = 2^{2^1}+1$. With such factors, formula (1) will have the form (2):

$$\begin{aligned} a^x \bmod M &= (a^{2^0} \bmod M)^{x_0}(a^{2^1} \bmod M)^{x_1}\cdots(a^{2^{2n-1}} \bmod M)^{x_{2n-1}} = \\ &= 2^x \bmod 15 = (2^{2^0} \bmod 15)^{x_0}(2^{2^1} \bmod 15)^{x_1}\cdots(2^{2^{2n-1}} \bmod 15)^{x_{2n-1}} = \\ &= (2 \bmod 15)^{x_0}(4 \bmod 15)^{x_1}(1 \bmod 15)^{x_1}\cdots(1 \bmod 15)^{x_{2n-1}} = \\ &= (2 \bmod 15)^{x_0}(4 \bmod 15)^{x_1} = 2^{x_0}4^{x_1} \end{aligned} \quad (2)$$

Formula (2) makes it possible to find the period using a digital quantum coprocessor.

The height of the bars in Figure 10, Figure 11 illustrates the probability of receiving the code x as a result of mentally measuring the upper register state of the circuit Figure 7. Thus, Figure 10 corresponds to the upper register in Figure 7 measured $|xx...x00\rangle$ state, and Figure 11 corresponds to its $|xx...x10\rangle$ state.

The implementation of Shor's algorithm in a digital quantum coprocessor is shown in Figure 12.

The lower register in Figure 12 is a classic digital logic circuit, signals at its outputs formation (for the considered example of finding the period of $y = a^x mod M$ function with $a = 2$, $M = 15$) is shown by Table 1 and Table 2. Measuring the states of the developed digital quantum qubits does not change the code of this state, which is indicated by the letter D on the meter symbol in the circuit Figure 12.

The calculated values of $y = a^x mod M$ function transform the state $|X\rangle$ into a state $|X_C\rangle$ correlated with the function values. For the considered example, this transformation is described in Table 3.

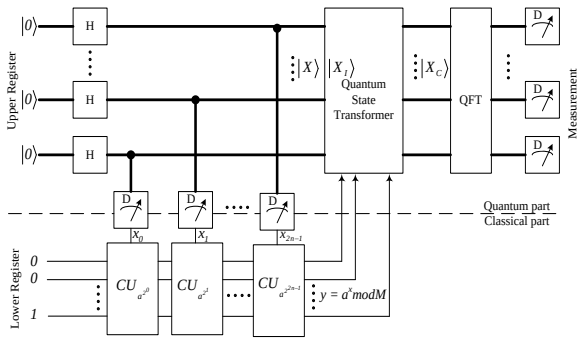


Figure 12: Shor's algorithm implementation in digital quantum coprocessor

Table 1

Controlled Unit $CU_{a^{2^0}} = CU_2$, $a = 2, a^{2^0} = 2$

Control signal	x_0=0 (multiplication by 1)	x_0=1 (multiplication by 2)
Input	0001	0001
Output binary	0001	0010
Output logical	$000\overline{x_0}$	$00x_00$
Output, formula	$00x_0\overline{x_0}$	

6. Discussions

Further, Figure 13 - Figure 20 show the results of the quantum part of Shor's algorithm (determining the period of the function $y = a^x mod M$ ($a = 2$, $M = 15$) in the digital quantum coprocessor Figure 7 with 8 digital qubits in the upper register, the width of each qubit is 8 bits. The research was carried out on two types of coprocessors - homogeneous and heterogeneous. For statistic, each study was repeated 4096 times with the same input data. In this case, the frequency of occurrence of each result (the probability of the result) was recorded.

Since the upper register with $2n = 8$ qubits was used, the number of different measured codes at the output of the Fourier quantum transform is $N = 2^{2n} = 256$.

In Figure 13 - Figure 20, these 256 codes are plotted along the horizontal axis, from 0 to 255. The figures show the codes that, as a result of the study, were found most often (high probability states – State_HP). The vertical axis shows the probability (Probability) of the occurrence of the indicated codes, the value of the probability (Value) is indicated in percent.

First of all figures show the results of white noise states generator Figure 8 study, the state of the qubits at the output of the upper register is $|X\rangle = |xxxxxxxx\rangle$. Figure 13 and Figure 14 show how such white noise is perceived by quantum Fourier transformer in homogeneous (Figure 13) and heterogeneous (Figure 14) digital quantum coprocessors. Measurement of such quantum state can give any code in the range 0-255 with equal probability. A heterogeneous coprocessor perceives white noise more correctly.

Table 2

Controlled Unit $CU_{a^{2^1}} = CU_4$, $a = 2, a^{2^1} = 4$

Control signal	x_1=0 (multiplication by 1)	x_1=1 (multiplication by 4)
Input	$00x_0\overline{x_0}$	$00x_0\overline{x_0}$
Output logical	$00(\overline{x_1}x_0)(\overline{x_1}\,\overline{x_0})$	$(x_1x_0)(x_1\overline{x_0})00$
Output, formula	$(x_1x_0)(x_1\overline{x_0})(\overline{x_1}x_0)(\overline{x_1}\,\overline{x_0})$	

Table 3

Controlled Unit $CU_{a^{2^1}} = CU_4$, $a = 2, a^{2^1} = 4$

$\lvert X\rangle = \lvert xxxxxxxx\rangle$. (measured x_1x_0)	y = a^xmodM	$\lvert X_C\rangle$
00	1	$\lvert xx...x00\rangle$
01	2	$\lvert xx...x01\rangle$
10	4	$\lvert xx...x10\rangle$
11	8	$\lvert xx...x11\rangle$

After the quantum Fourier transform, determining the number of repetitions of the measured codes gives the following results: both homogeneous and heterogeneous quantum coprocessors correctly determine that there are no repetitions of codes when carrying out a large number of measurements (the number of repetitions is $F = 0$). A homogeneous coprocessor generates such result with probability of 84.195% (Figure 15), and heterogeneous - with probability 32.805 % (Figure 16).

After confirming correct operation of both digital quantum elements of Hadamard and quantum Fourier transformer, which is also built from digital qubits, studies of Shor's algorithm implementation (Figure 7) were continued. Figure 17 and Figure 18 show how the quantum Fourier transform perceives correlated with $y = a^x mod M$ function upper register state $\lvert X_C\rangle$ in homogeneous (Figure 17) and heterogeneous (Figure 18) digital quantum coprocessors.

And in this case, the heterogeneous coprocessor perceives correlated states more correctly.

$y = 2^x mod 15$ function has period $T = 4$. The discrete Fourier transform should most often form the result $F = N/T = 256/4 = 64$. Despite the difference in the perception of correlated upper register states, both homogeneous and heterogeneous quantum coprocessors correctly determine the repetition rate of codes when carrying out a large number of measurements, they correctly determine the number of repetitions $F = 64$. A homogeneous coprocessor generates such a result with probability 21.439 % (Figure 19), and heterogeneous - with probability 15.341 % (Figure 20).

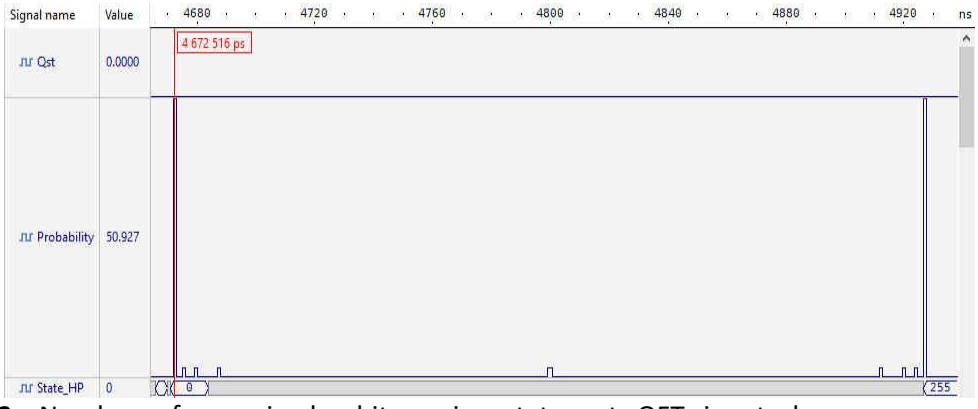


Figure 13: Number of perceived white noise states at QFT input, homogeneous quantum coprocessor

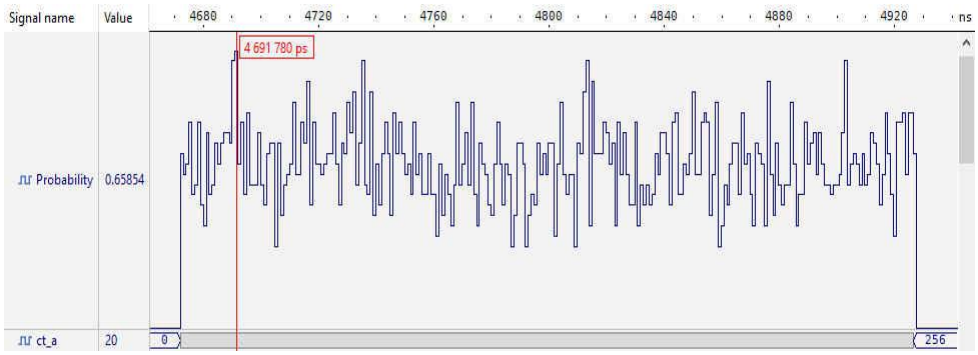


Figure 14: Number of perceived white noise states at QFT input, heterogeneous quantum coprocessor

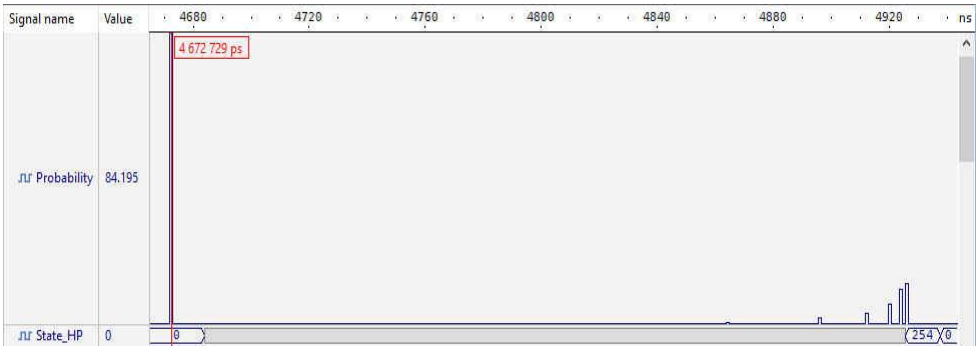


Figure 15: Number of repetitions of white noise states, homogeneous quantum coprocessor

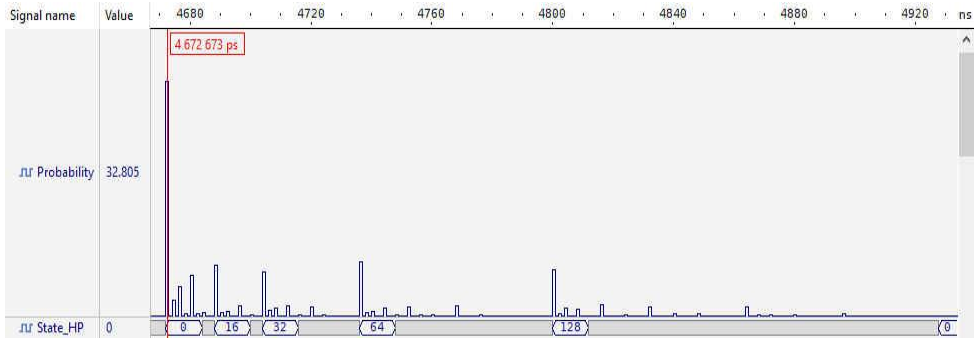


Figure 16: Number of repetitions of white noise states, heterogeneous quantum coprocessor

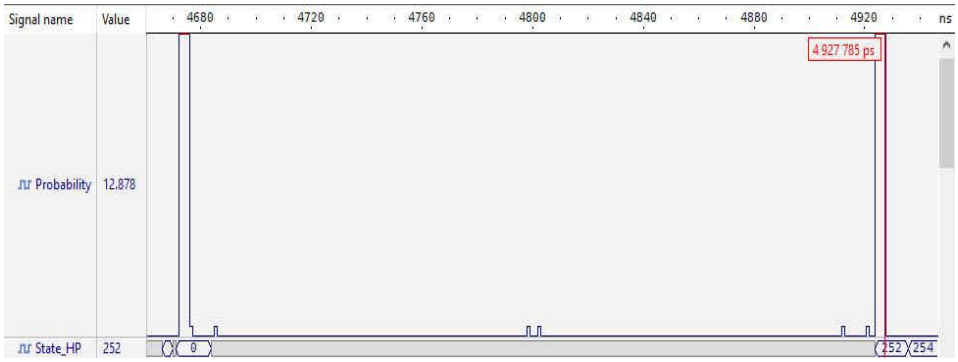


Figure 17: Upper register states that are perceived at the input of QFT and correlated with the function $y = a^x mod M$, homogeneous quantum coprocessor

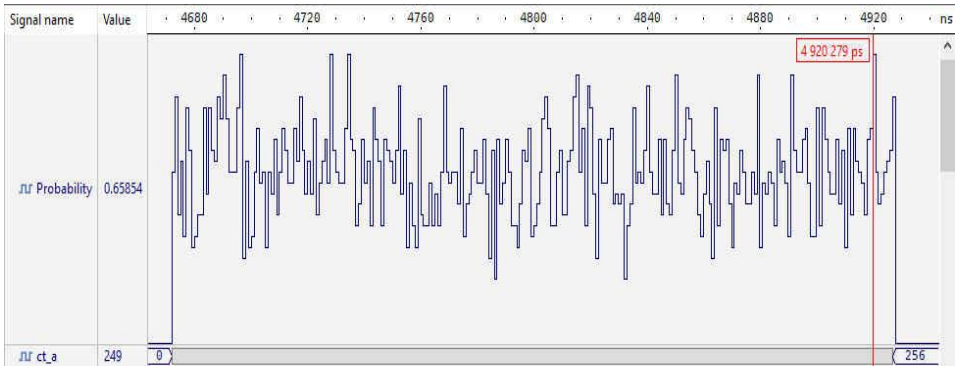


Figure 18: Upper register states that are perceived at the input of QFT and correlated with the function $y = a^x mod M$, heterogeneous quantum coprocessor

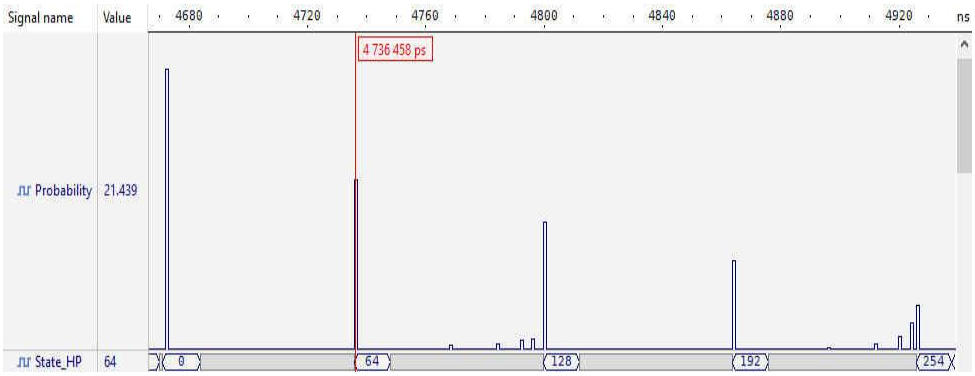


Figure 19: The number of states repetitions when determining the period of a function, homogeneous quantum coprocessor

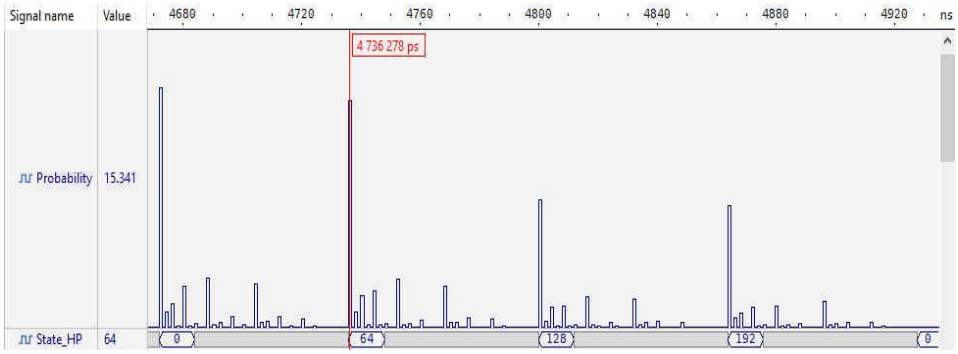


Figure 20: The number of states repetitions when determining the period of a function, heterogeneous quantum coprocessor

The results of Shor's algorithm execution are summarized in the Table 4.

7. Implementation

In this example, adding a lower register and an upper register state converter (Figure 12) adds practically nothing to the hardware costs of a discrete Fourier transform implementation (Figure 8). These costs were determined in previous works [3], [4]. It was shown that on one FPGA it is possible to implement the discrete Fourier transform with the number of digital qubits up to 1024.

8. Conclusions

The article shows the possibility of determining the period of the $y = a^x mod M$ function in a digital quantum coprocessor. Determination of the period is necessary for the execution of Shor's factorization algorithm.

Table 4
Probability of correct results, %

Qubits number	8
Qubits width, bit	8
Homogenous coprocessor	21.439
Heterogeneous coprocessor	15.341

The possibility of implementing Shor's factorization algorithm using two types of implemented on FPGA digital quantum coprocessors - homogeneous and heterogeneous - is shown. For the research, the factorization of the number 15 was chosen ($a = 2$, $M = 15$). Determining the period of the $y = a^x mod M$ function is a task of a quantum coprocessor.

The studies were carried out on coprocessors with 8 digital qubits, the state of each qubit was encoded using 8 bits.

A homogeneous digital quantum coprocessor has the best performance: the probability of obtaining a correct result is 21.439%, and that of a heterogeneous one is 15.341%.

The coprocessor is focused on implementation in FPGA. The presented results were obtained after simulating VHDL-descriptions of coprocessors.

The digital quantum coprocessor outputs each subsequent result at the system frequency of the FPGA. In the simulation, this frequency was 1 GHz (period was 1 ns).

9. References

[1] X. Fu et al., "A heterogeneous quantum computer architecture," in 2016 ACM International Conference on Computing Frontiers (CF'16), 323–330, 2016, doi: http://dx.doi.org/10.1145/2903150.2906827.

[2] L. Riesebos et al., "Quantum Accelerated Computer Architectures," 2019 IEEE International Symposium on Circuits and Systems (ISCAS), 2019, pp. 1-4, doi: 10.1109/ISCAS.2019.8702488.

[3] V. Hlukhov. "FPGA-Based Digital Quantum Computer Verification". The 11th IEEE International Conference on Dependable Systems, Services and Technologies, DESSERT'2020. 14-18 May, 2020, Kyiv, Ukraine. In press.

[4] V. Hlukhov, "FPGA-Based Homogeneous and Heterogeneous Digital Quantum Coprocessors", Advances in Science, Technology and Engineering Systems Journal (ASTESJ), 2020, 5(6), 1643-1650, DOI: 10.25046/aj0506195.

[5] J. A. Smolin, G. Smith, A. Vargo, Pretending to factor large numbers on a quantum computer, arXiv: 1301.7007v 1 [quant-ph] (2013). http://arxiv.org/abs/1301.7007.

[6] J. A. Smolin, G. Smith, A. Vargo, Oversimplifying quantum factoring. Nature 499, 163-165 (11 July 2013).

[7] Y. Wang, H. Zhang and H. Wang, "Quantum polynomial-time fixed-point attack for RSA," in China Communications, vol. 15, no. 2, pp. 25-32, Feb. 2018, doi: 10.1109/CC.2018.8300269.

[8] P. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," in 35th Annual Symposium on Foundations of Computer Science, 124–134, 1994, available at: https://www.jstor.org/stable/2653075?seq=1, accessed 25 Nov. 2020.

[9] A. Pavlidis, D. Gizopoulos. Fast Quantum Modular Exponentiation Architecture for Shor's Factorization Algorithm. July 2014. Quantum Information & Computation 14(7&8):0649-0682.

[10] B. Dey, K. Khalil, A. Kumar and M. Bayoumi, "A Reversible-Logic based Architecture for Artificial Neural Network," 2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS), 2020, pp. 505-508, doi: 10.1109/MWSCAS48704.2020.9184662.

[11] Quantum-computing.ibm.com. Shor's algorithm. Reversible classical circuits, 2020. URL: https://quantum-computing.ibm.com/composer/docs/iqx/guide/shors-algorithm.

[12] E. Grumbling, M. Horowitz (eds.). "Quantum computing: progress and prospects". The National Academies of Sciences, Engineering, and Medicine. Washington, DC: National Academies Press. 2019.