ADVANCES IN CYBER-PHYSICAL SYSTEMS

Vol. 4, No. 1, 2019

IMPLEMENTING QUANTUM FOURIER TRANSFORM IN A DIGITAL QUANTUM COPROCESSOR

Valerii Hlukhov

Lviv Polytechnic National University, 12, Bandera Str, Lviv, 79013, Ukraine.
Author's e-mail*: glukhov@polynet.lviv.ua*

Submitted on 24.06.2019

***Abstract*: In this paper, the digital quantum coprocessor has been checked for the possibility of quantum Fourier transform, which is the main quantum operation of the Shor's algorithm. To do this, the model of the 4-qubit coprocessor has been created, its work has been simulated and it has been implemented in FPGA.**

***Index Terms*: digital quantum coprocessor, digital qubit, quantum Fourier transform**

I. INTRODUCTION

Classic quantum computers (QC) are analog computing devices [1]. It is a computing device that uses the phenomena of quantum mechanics (quantum superposition, quantum entanglement) to transfer and process data. A quantum computer (as opposed to a normal classical computer) does not operate with bits (capable of accepting either 0 or 1), but qubits, which have values of both 0 and 1 simultaneously (quantum superposition). At the end of the problem-solving process, this superposition collapses with some probability into one of the two classical states, 0 or 1.

The digital quantum computer that can be implemented in FPGA is described in [2]. A digital quantum coprocessor operates under the control of a classical computer and together they are digital quantum computer. The digital quantum coprocessor is a set of digital units called digital qubits that have multi-bit data input and single-bit data output each. There is a pseudo random number generator (PRNG) in the digital qubit to generate the probabilistic output bit.

For some tasks, the number of qubits in a computer is a polynomial function of the task complexity. For example, algorithms are known for factoring an n-bit integer using just over 2n qubits (Shor's algorithm [3], [4]).

This is estimated to take at least 4000 qubits (but could be more depending upon the algorithm used [5]) to factor a 2048 bit number in order to break RSA encryption with a 2048 bit key.

Real quantum computers are analog and probabilistic devices. Their implementation is a very difficult and expensive task. All this underlines the relevance of work on the study of the characteristics of quantum computers, the creation of their digital models and digital samples, as well as the training of specialists to work with them.

In this paper, the digital quantum coprocessor is checked for the possibility of quantum Fourier transform, which is the main quantum operation of the Shor's algorithm. To do this, the model of the 4-qubit coprocessor was created, his work was simulated and it was implemented in FPGA.

II. QUANTUM COMPUTER BASIS

If a classic computer at any moment can be in exactly one of the states $|0\rangle, |1\rangle, ..., |N-1\rangle$ (Dirac notation), QC at each moment is simultaneously in all these basic states, and in each $|j\rangle$ state – with its complex amplitude l_j. This quantum state is called "quantum superposition" of these classical states and is denoted as the wave function $|y\rangle = \sum_{j=0}^{N-1} l_i |j\rangle$. The probability p_j of obtaining a state $|j\rangle$ as a result of measurement is equal to $p_j = l_j^2$. In this case, the sum of all probabilities $P = \sum_{j=0}^{N-1} l_i^2 = 1$ [6].

The quantum state $|y\rangle$ can change in time in two fundamentally different ways: by unitary quantum operation and by measurement [7].

Any unitary transformation of the wave function $|y\rangle$ can be represented as a simple displacement of a point over the surface or sphere of unit radius (Bloch sphere for complex amplitudes, Fig. 1, [8]), or a circle of unit radius (for real amplitudes). In the last case, the position of the vector at an angle of 0 radian corresponds to the state $|0\rangle$, and at an angle $\pi/2$ radian – state $|1\rangle$). It is possible to realize the calculation of wave functions by digital methods – by software or hardware (for example, on the FPGA).

The state of one qubit is written in the QC as $A|0\rangle + B|1\rangle$ – this is called a quantum superposition. For the Bloch sphere (Fig. 1) the amplitudes $A = \cos(q/2)$ and $B = e^{ij}\sin(q/2)$, for a unit circle – $A = \cos q$, $p_0 = \cos^2 q$ and $B = \sin q$, $p_1 = \sin^2 q$ respectively. Then, for the unit circle $\langle\Psi| = \cos q\,|0\rangle + \sin q\,|1\rangle$. If the qubit is in a state $\langle\Psi| = 1/\sqrt{2}\,|0\rangle + 1/\sqrt{2}\,|1\rangle$, then as a

result of the measurement with probability $p_0 = \left(1/\sqrt{2}\right)^2 = 1/2$ it can be determined as $|0\rangle$ and with the same probability as $|1\rangle$ (it is the undefined state of qubit). The indefinite state of two qubits is written as $\frac{1}{2}|00\rangle + \frac{1}{2}|01\rangle + \frac{1}{2}|10\rangle + \frac{1}{2}|11\rangle$ – two qubits can be in each of their 4 states with probability $p_0 = (1/2)^2 = 1/4$.

The action of the gate on a specific quantum state is found by multiplying the vector $|ab\rangle$, which represents the state, by the matrix U, representing the gate, $U|ab\rangle$ [9].

QC and classical computers tend to work together**.** It should be noted that in algorithms for QC a significant part of the calculations is performed by classical computers. For example, in Shor's algorithm [3], [4], the share of classic computers is the tabulation of the function $f(x) = t^x \bmod M$, where t is a prime number, M is a large number that needs to be factorized, x = 0, 1, …, N-1, N is about 2 times bigger than M. Also, a classical computer is engaged in final processing of the results and checking their reliability and suitability (QC is a probabilistic computer, and it gives the correct result with a certain probability).

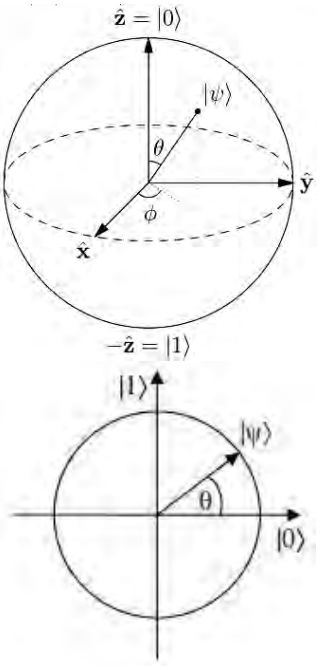


Fig. 1. Representation of a qubit in the form of a Bloch sphere for complex amplitudes (left) and a unit circle for real ones (right)

III. QUANTUM GATES

Quantum gate (quantum logic element) is the basic element of a quantum computer, which transforms the input states of qubits in the output ones according to a certain law. It differs from ordinary logic gates by working with qubits, and, therefore, obeys quantum logic. Quantum gates, unlike many classical ones, are always reversible.

There is a small set of quantum gates (conditional vector rotation and Hadamard transform) that creates a basis that allows the creation of circuits for quantum Fourier transformation [10].

In a gate-based approach to quantum computing, each primitive operation (gate) is performed by precisely changing the Hamiltonian of one or more qubits for the specific amount of time required to achieve the desired transformation. This is done by changing the physical environment, for example, via a laser pulse or application of some other electromagnetic field, depending on the way in which the qubits are built. Since these primitive operations are analogous to logic gates in classical computing, systems built using this approach are called “digital quantum computers” [11]. However, we must remember that in this case so called “digital quantum computers” are analog computer really unlike the real quantum computers described in this article and in [2].

There are software tools that allow to create circuits from quantum gates and simulate their work [12].

IV. THE DIGITAL QUANTUM COMPUTER

A true quantum computer is an analog probabilistic computer. Its schemes consist only of gates. They have no memory elements. Therefore, there are no quantum programs. There are software tools that allow to create circuits from quantum gates and simulate their work according to circuits’ description in C-like language [12]. This is very similar to the design of programmable logic circuits (FPGAs) and, especially, to the design of analogue programmable circuits.

Only a classic computer that controls a quantum computer performs real programs. Therefore, the quantum computer is actually a coprocessor with respect to the classic computer. The quantum computer functional scheme is induced in Fig. 2 [2].

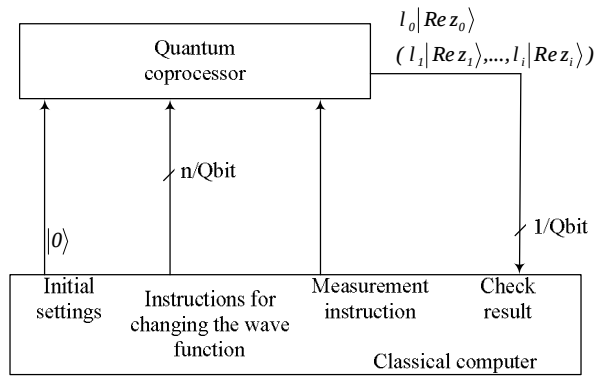


Fig. 2. A digital quantum computer

The digital quantum coprocessor consists of several qubits or it can be assumed that several single-chip

digital quantum coprocessors are connected to one classical computer (Fig. 3).

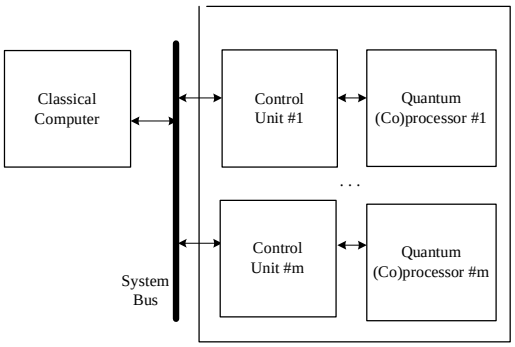


Fig. 3. A quantum computer with multiple digital quantum coprocessors

A classical computer manages the work of a quantum coprocessor, provides it with an input data, stream of instructions and checks the result of its work. Some options for constructing digital quantum computers are discussed in [13]. A generalized functional scheme of a quantum coprocessor is given in Fig. 4 [2]. Digital quantum coprocessor in Fig. 4 is represented as a set of finite-state machines (FSMs), one of them is a controller and one or more implement the functions of a digital qubit. The connection between the latters can be carried out through the pipeline registers (pRG).

The programmable switch matrix of the coprocessor makes it similar to the Complex Programmable Logic Device (CPLD [14], Fig. 5 [2]). The inputs of the matrix are the outputs of all digital qubits ($Result_1$, …, $Result_n$), from the side of the classical computer the matrix can be programmed, so that the inputs of any qubit were fed out of any other qubit ($Result_p$, …, $Result_q$).

The simplest version of a digital quantum coprocessor on a FPGA has only one digital qubit, which consistently performs operations that correspond to operations of quantum gates. The classical computer determines the sequence of these operations. The more complex coprocessor has a chain of several digital qubits, as well as several qubit chains (Fig. 4 – a coprocessor with one chain). Data transfer from one qubit to another can be done through pipeline registers.

A classical analog qubit forms only one output bit. A digital qubit as a finite state machine was described in [2]; digital qubits without loopbacks and with pipeline register and flip-flop on the outputs (Fig. 8) are used in this work.

Unlike the analog qubit with a single-bit output, you can also organize a multi-bit output in the digital qubit, which will precisely determine its current state (Fig. 8). The use of a multi-bit output by a classical computer allows you to arrange interruptions of quantum programs and call quantum procedures and functions (programs, procedures, functions, and interruptions that are executed and processed by a classical computer in the process of controlling a digital quantum coprocessor, will be called quantum ones).

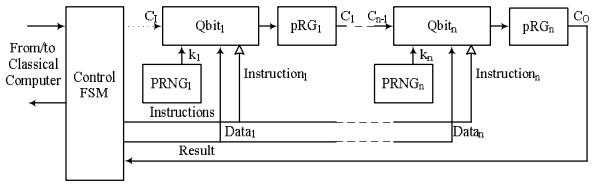


Fig. 4. A generalized functional diagram of a digital quantum coprocessor

V. GOAL OF THE WORK

In the previous paper [2], the possibility of generating correct probabilistic results by digital qubits and a digital quantum coprocessor was investigated. The ability of a digital quantum coprocessor to perform more complex algorithm (quantum Fourier transform), which is used in the Shor's algorithm, was considered.

VI. DIGITAL QUANTUM COPROCESSOR FOR REAL AMPLITUDES

To explain the principles of digital quantum coprocessor design, it will be firstly implemented for the case of real amplitudes when the behavior of a single qubit is described by the movement of the radius vector in the unit circle (Fig. 1).

In this paper, to represent the position of the vector, a polar coordinate system is selected (it is necessary to specify and define only one coordinate – the angle θ (Fig. 6), in contrast to the Cartesian coordinate system, where it is necessary to specify two coordinates, x and y).

The distance on the Bloch sphere from state $|0\rangle$ to state $|1\rangle$ is the same as from $|1\rangle$ to $|0\rangle$ and equal to π radians. In the unit circle, when when we move counter clockwise these distances are different: the distance from state $|0\rangle$ to state $|1\rangle$ is π/2 radians, and from state $|1\rangle$ to state $|0\rangle$ is 3π/2 radians. To eliminate this skewing it is necessary, when transforming algorithms describing the displacement of a vector in the Bloch sphere into algorithms for the unit circle (for example, when correcting the algorithm of the quantum Fourier transform), to reduce the phase shift and use only the first quadrant of the unit circle (Fig. 6).

VII. A MODIFIED DIGITAL QUBIT

There are several approaches to design quantum computers FPGAs ([13], [15]). The peculiarity of [2] and this paper is the implementation of a digital quantum coprocessor and the use of polar coordinates to represent the position of the vector on a unit circle. For the realization of probabilistic functions, a generator of pseudorandom codes with the period 2^{32}-1 [16], [17] is used. The functional converter is added to the output of the generator of random codes [22] (it is a converter $D = \arcsin\sqrt{A}$, Fig. 7). Functional converters can be created according to well-known solutions [18], [19].

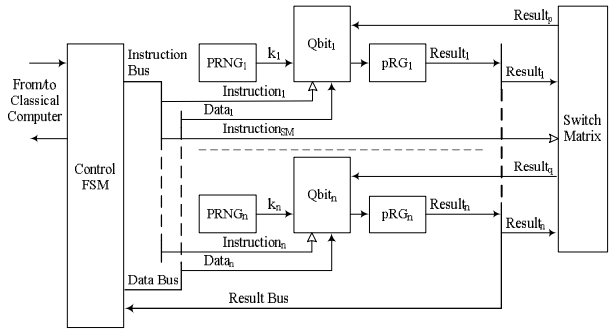


Fig. 5. A generalized functional diagram of a digital quantum coprocessor with a switching matrix

The codes used for angle range are shown in Fig. 6.

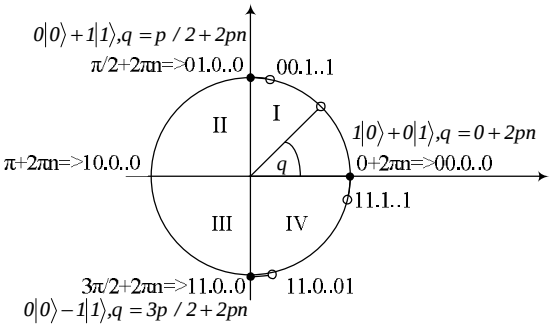


Fig. 6. Position codes of vectors (angle θ codes)

A qubit that can be controlled by a classical computer appears as one pipeline stage (Fig. 8) with the measurement unit, pipeline register and pipeline flip-flop on the outputs. The digital qubit has multi-bit input QI and output QO, single-bit output q and input for conditions Cond, which is used for conditional instructions such as conditional phase shift in digital Fourier transform.

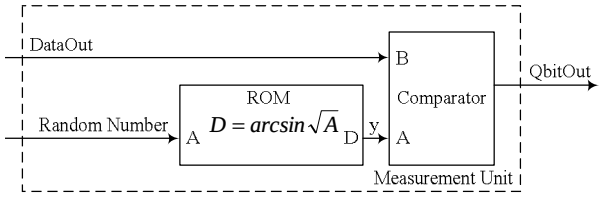


Fig. 7. An alternative way to measure the state of a digital qubit

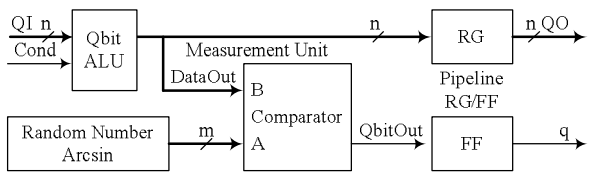


Fig. 8. Digital qubit

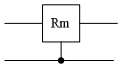
The additional and optional multi-bit output (QO in Fig. 8) allows to interrupt digital qubit control programs performed by a classical computer. This output allows you to read the exact internal state DataOut of the digital qubit, to download the new state S_1 through the QI input, to perform a series of instructions starting from this state S_1, to read the exact state of S_2, in which the qubit will appear after their execution, to save S_2 in the memory of the classic computer, then to read previously saved qubit previous state S_0 from the memory of a classical computer, to load it into a qubit and continue to execute instructions from this state S_0.

VIII. MAIN OPERATIONS THAT ARE REQUIRED TO PERFORM A QUANTUM FOURIER TRANSFORM

The instructions that a digital qubit can perform must be universal – form a functionally complete system of functions, that is, using a set of such instructions, you can recreate the work of any other quantum gate.

Quantum gates [9] perform unitary operations that do not change the vector norm on the Bloch sphere; they only move a point along the sphere.

Only 2 of such operations are required during quantum Fourier transform:

- controlled phase shift R_m 
 ;
- Hadamard transform H 

$$R_m = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{2\pi i/2^m} \end{pmatrix}, \quad H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

IX. QUANTUM FOURIER TRANSFORM

In quantum computing, the quantum Fourier transform (for short: QFT) is a linear transformation on quantum bits, and is the quantum analogue of the inverse discrete Fourier transform. The quantum Fourier transform is a part of many quantum algorithms, notably Shor's algorithm for factoring [10]. The quantum Fourier transform is defined [20] as $\sum_j a_j |j\rangle \to \sum_k \tilde{a}_k |k\rangle$, where $\tilde{a}_k \equiv \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi i j k / N} a_j$.

The quantum gates used in the QFT circuit (Fig. 9) are the Hadamard gate (H) and the controlled phase gate (R_m). The order of the qubit at the output is opposite to their order at the input. Output qubits must be swapped.

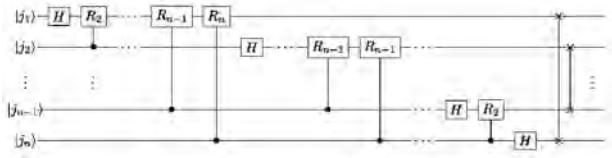


Fig. 9. QFT Circuit implementation [20]

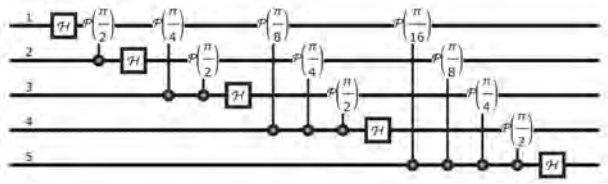


Fig. 10. Simplified drawing of Quantum Fourier Transform (5-qubits) [20]

The task of the discrete quantum Fourier transform is to determine the frequency F of the input register measured states changes. In the Shor's algorithm the result is used to determine the period T of states changes because T=1/F.

An illustrative example of the quantum Fourier transform applied to a three-qubit system is given in [11]. The results obtained below of the study of the quantum Fourier transform by a digital quantum coprocessor are obtained by a method similar to [11].

X. TESTING THE IMPLEMENTATION OF QUANTUM FOURIER TRANSFORM BY A DIGITAL QUANTUM COPROCESSOR

A detailed diagram of a 4-qubit digital quantum coprocessor that performs discrete Fourier transform is shown in Fig. 11. The diagram corresponds to Fig. 9 and Fig. 10.

The scheme has 4 rows of elements. Each row corresponds to the conversion of one analog or one digital qubit.

The digital quantum coprocessor in Fig. 11 consists of three types of digital quantum gates (Fig. 8) .

ALU in digital quantum gates of type Load translates the input QI code to the pipeline register without conversion.

ALU in digital quantum gates of type Add Const sends the sum QI + Const/2 if Cond = 1 (where Const – $\pi/2$, $\pi/4$ or $\pi/8$) to the pipeline register.

ALU in Had type digital quantum gates sends the input code to the pipeline register after the Hadamar transformation over it.

The number of digital quantum gates in the coprocessor circuit can be reduced to one row, or one column, or even up to one digital quantum gate (one digital qubit). In this case, the ALU in every digital quantum gate will have to perform more than one operation, but a few. In addition, the control scheme in Fig. 4 will be more complicated.

The results of the study are presented in the form of graphs. The graphs (a) show the state spectra of the input 4-qubit register, in graphs (b) – the state spectra of the output 4-qubit register.

The states of the input qubits (in the form of angles that determine the position of the vectors in the unit circle) are set before the start of the study and during the study do not change. The probability of being an input register in one of 16 states is calculated. For example, the probability that the input register is in state $|13\rangle = |q_3 q_2 q_1 q_0\rangle = |1101\rangle$ is $P_{13} = \cos^2 q_3 \cdot \cos^2 q_2 \cdot \cdot \sin^2 q_1 \cdot \cos^2 q_0$, where q_j is the angle under which the vector of j-th qubit q_i is inclined.

The situation is chosen for the study, when one vector is inclined at an angle of 0°, and all other are at an angle of 45°.

We also investigated the situation when all vectors are inclined at an angle of 45°.

The results of the study are presented in Fig. 12, Fig. 13,

Fig. 14, Fig. 15,

Fig. 16.

Results of the study, when $q_3 = q_2 = q_1 = 45°$, $q_0 = 0°$, are shown in Fig. 12. As it can be seen from Fig. 12 (a) the input state spectrum is periodic and consists of 8 periods of probability fluctuations. The quantum coprocessor will mark this phenomenon after quantum Fourier transformation and will give true result 8 (as the number of oscillation periods) with a probability of 63.2 %, or will give false result 0 with less probability 18.1 %, or will give other false results with even lower probabilities.The task of the classical computer is to check each of the quantum coprocessor results.

The output state spectrum was determined by measuring output register states during 10.000 identical experiments with the subsequent calculation of the frequency of occurrence (probability) of each state.

As it can be seen from Fig. 12, Fig. 13,

Fig. 14, Fig. 15,

Fig. 16 in the performed research, the digital quantum coprocessor in general correctly determines the number (correspondingly 4, 2, 1 and 0) of the probability fluctuation periods, that is, it correctly executes a quantum Fourier transform. It indicates that the proposed digital qubites of which it is composed also work appropriately; they can be used to construct other digital quantum coprocessors.

XI. IMPLEMENTATION OF THE DIGITAL COPROCESSOR ON FPGA

Described digital coprocessor for n=13, m=32 in Fig. 6 has been implemented on the FPGA Spartan 6 (Xilinx) [21].

Table 1

Implementation results of 4 digital qubits in Xilinx xc6slx45t, package csg484, speed -3, clock period 5.7 ns, 9-bit qubit, 32-bit qubit's TRNG

Number of	Number	out of	%
Slice Registers:	966	54.576	1
Slice LUTs:	998	27.288	3
Occupied Slices:	361	6.822	5
RAMB8BWERs:	7	232	3

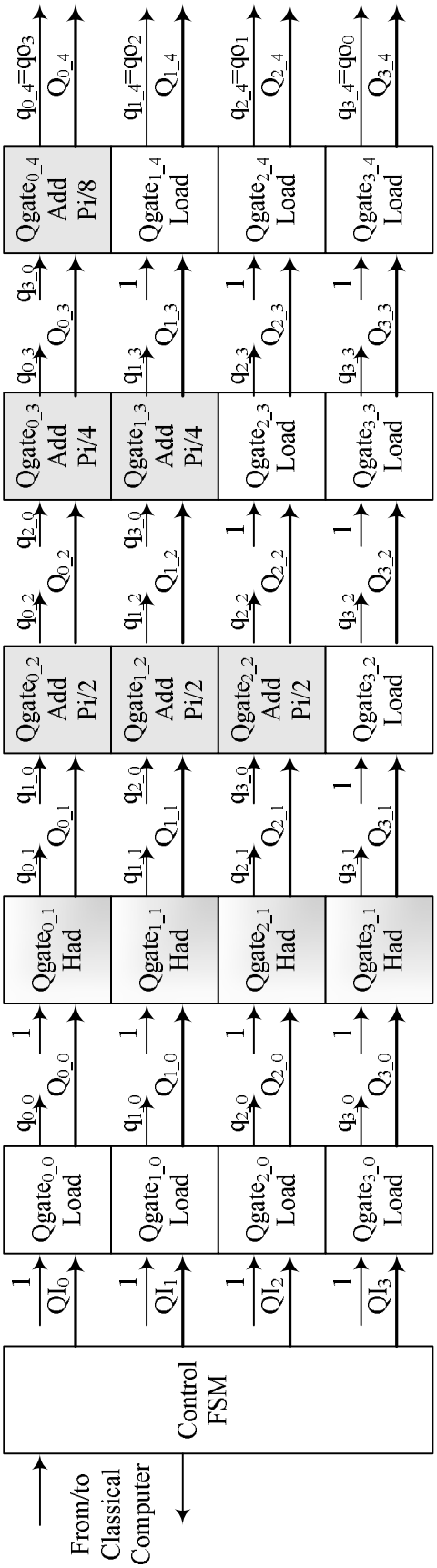
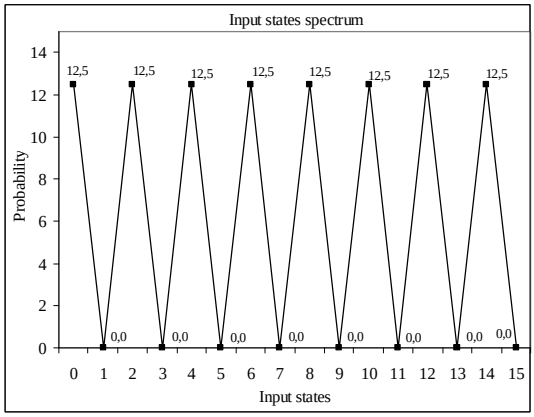
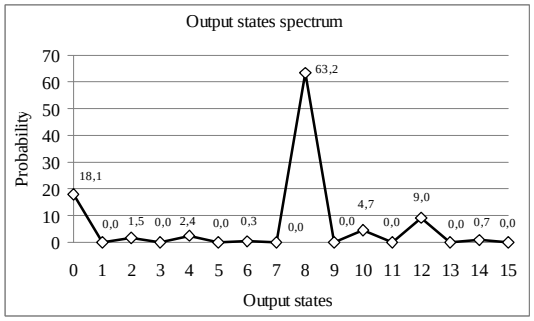


Fig. 11. Digital quantum Fourier transformer

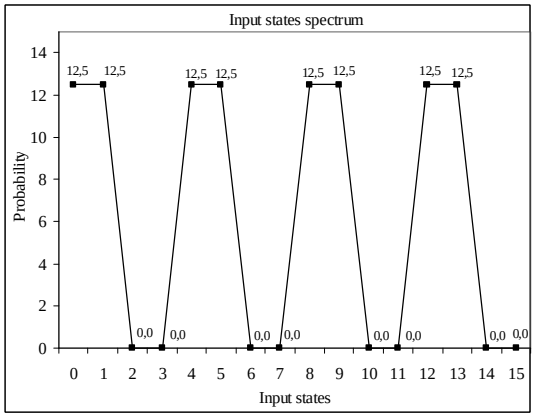


a

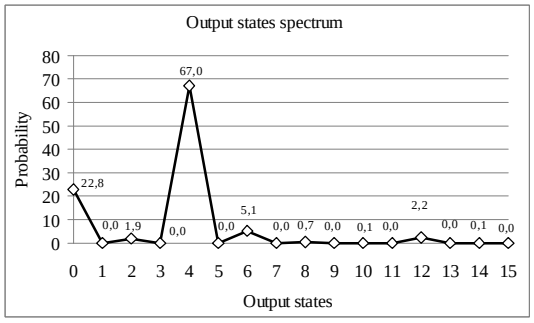


b

Fig. 12. Calculated input (a) and measured output (b) state spectrum ($q_3 = q_2 = q_1 = 45°, q_0 = 0°$)

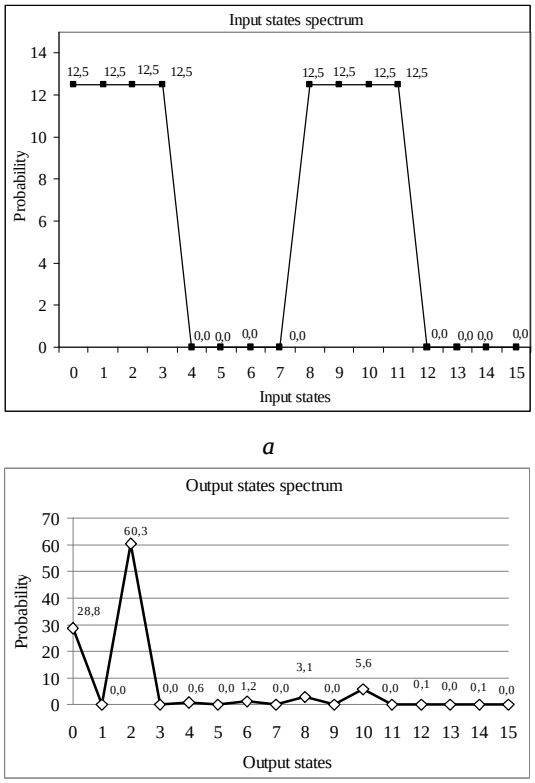


a



b

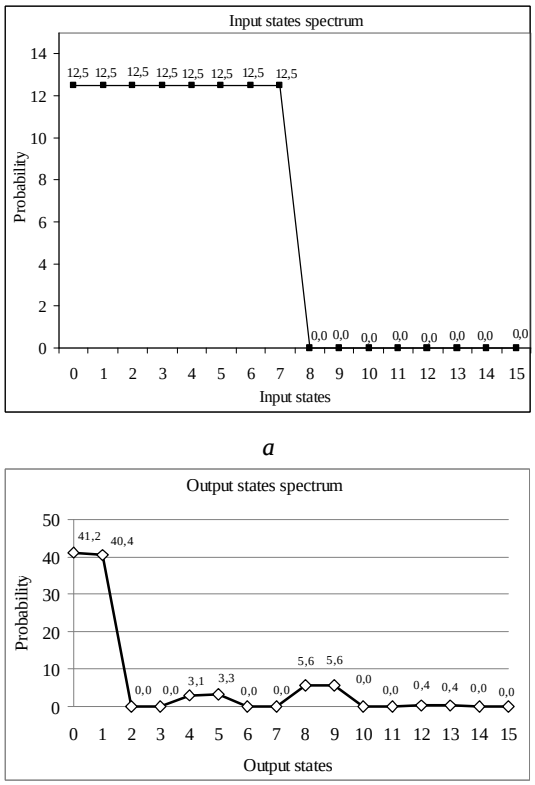
Fig. 13. Calculated input (a) and measured output (b) state spectrum ($q_3 = q_2 = q_0 = 45°, q_1 = 0°$)



a

b

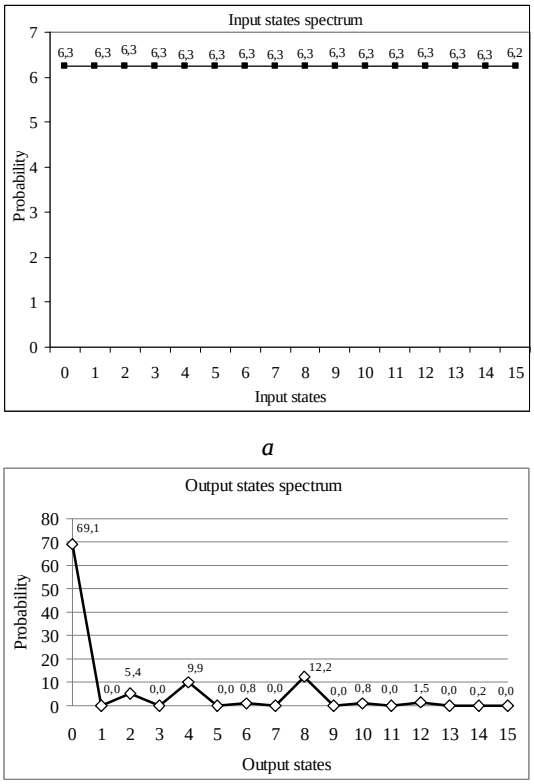
Fig. 14. Calculated input (a) and measured output (b) state spectrum ($q_3 = q_1 = q_0 = 45°, q_2 = 0°$)



a

b

Fig. 15. Calculated input (a) and measured output (b) state spectrum ($q_2 = q_1 = q_0 = 45°, q_3 = 0°$)



a

b

Fig. 16. Calculated input (a) and measured output (b) state spectrum ($q_3 = q_2 = q_1 = q_0 = 45°$)

XII. CONCLUSION

The principles of constructing digital quantum coprocessors to perform a quantum Fourier transform were presented.

Model of the 4-qubit digital quantum coprocessor for its next implementation in the FPGA was created. The model was tested.

In general, proposed models of digital quantum gates, digital qubits that form a digital quantum coprocessor, and the proposed digital quantum coprocessor itself work correctly.

REFERENCES

[1] Valeriy Hlukhov. “Kvantovyy kompyuter kak veroyatnostnyy kompyuter”. Shosta mizhnarodna naukova konferencija “Modeljuvannja-2018”. September 12–14, 2018 Kyiv, Ukraine. Zbirka pracj konferenciji, p. 111–114.

[2] Valerii Hlukhov, Bohdan Havano. FPGA-based Digital Quantum Coprocessor. Advances in Cyber-Physical Systems. Volume 3. Number 2. Lviv Polytechnic National University. 2018. pp. 12–31.

[3] Peter W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, Nov. 20–22, 1994, IEEE Computer Society Press, pp. 124–134.

[4] Shor's algorithm. https: //en.wikipedia.org/wiki/Shor %27s_ algorithm 08.02.2019.

[5] Applying Moore’s Law to Quantum Qubits https://quantumcomputingreport.com/our-take/applying-moores-law-to-quantum-qubits/ Copyright © 2019 Quantum Computing Report, All rights reserved 19.02.2019

[6] Quantum computing. https: //en.wikipedia.org/wiki/Quantum_ computing 08.02.2019.
[7] Qubit. https://en.wikipedia.org/wiki/Qubit. 08.02.2019.
[8] Introduction to Quantum Computing. https://blogs.msdn. microsoft.com/uk_faculty_connection/2018/02/06/introduction-to-quantum-computing/ 08.02.2019.
[9] Quantum logic gate. https: //en.wikipedia.org/ wiki/Quantum_logic_gate 08.02.2019.
[10] Quantum Fourier transform. https://en.wikipedia.org/ wiki/Quantum_Fourier_transform 08.02.2019.
[11] National Academies of Sciences, Engineering, and Medicine. 2019. Quantum Computing: Progress and Prospects. The National Academies Press, Washington, DC. doi: https://doi.org/10.17226/25196.
[12] Welcome to the Microsoft Quantum Development Kit Preview. https://docs.microsoft.com/ru-ru/quantum/?view=qsharp-preview 08.02.2019
[13] M. Khalil-Hani, Y. H. Lee, M. N. Marsono. An Accurate FPGA-Based Hardware Emulation on Quantum Fourier Transform. Proceedings of the 13th Australasian Symposium on Parallel and Distributed Computing (AusPDC 2015), Sydney, Australia, 27–30 January 2015. pp. 23–30.
[14] CPLD. https://www.xilinx.com/products/silicon-devices/cpld/ cpld.html 16.02.2019
[15] Gushanskiy S. M., Pereverzev V. A. Simulation of Quantum Computing using Hardware Cores. Nauchnyy zhurnal KubGAU, №123(09), 2016. pp. http://ej.kubagro.ru/2016/09/pdf/37.pdf
[16] LFSR-Random number generator:: Overview. https://opencores.org/projects/lfsr_randgen 14.02.2019.
[17] Pseudo Random Number Generators as synthesizable VHDL code. https://github.com/jorisvr/vhdl_prng 14.02.2019
[18] Popov B.A.. Tesler G.S. Vychisleniye funktsiy na EVM. Spravochnik. Kiyev: Nauk. Dumka, 1984. 59 p. (In Russian).
[19] V. V. Aristov. Integro-algoritmicheskiye vychisleniya. “Nauk. Dumka”, 1980. 189 p. (In Russian).
[20] Jonathan Hui. QC – Quantum Fourier Transform. https://medium.com/@jonathan_hui/qc-quantum-fourier-transform-45436f90a43 2019.07.07 01:17
[21] Spartan-6 Family Overview. DS160 (v2.0) October 25, 2011. Product Specification. https://www.xilinx.com/support/ documentation/data_sheets/ds160.pdf 14.02.2019
[22] Valeriy Hlukhov, Bohdan Havano. Principles of Digital Quantum Coprocessor Based on a FPGA, which Operates under the Control of a Classical Compute. Advanced Computer Information Technologies Acit 2019. June 5–7, 2019. International Conference. Ceske Budejovice, Czech Republic. Conference Proceedings, pp. 191–194.

Valerii S. Hlukhov is a professor of the Department of Computer Engineering in Lviv Polytechnic National University, Ukraine. He graduated from Lviv Polytechnic Institute with the engineer degree in computer engineering in 1977. In 1991 he obtained his Ph.D. from the Institute of Modeling Problems in Power Engineering of the National Academy of Science of Ukraine. He was recognized for his outstanding contributions into special-purpose computer systems design as a Senior Scientific Researcher in 1995.

He was awarded with the academic degrees of Doctor of Technical Sciences in 2013 at Lviv Polytechnic National University. He became a Professor of Computer Engineering in 2014. He has scientific, academic and hands-on experience in the field of computer systems research and design, proven contribution into IP Cores design methodology and high-performance reconfigurable computer systems design methodology. He is experienced in computer data protection, including cryptographic algorithms, cryptographic processors design and implementation. Mr. Hlukhov is an author of more than 100 scientific papers, patents and monographs.