RESEARCHING POSSIBLE INTERPRETATIONS AND IMPLEMENTATIONS OF CLASSICAL LOGIC ELEMENTS AND ALGORITHMS IN QUANTUM CIRCUITS

Cesar Borisovich Pronin

Andrey Vladimirovich Ostroukh

Moscow Automobile and Road Construction State Technical University (MADI)., 64, Leningradsky prospect, Moscow, Russian Federation, postal code: 125319

Abstract: This paper contains a research about using known classical circuit concepts as a part of quantum algorithms, keeping the benefits of both types of circuits. This was established by interpreting classical logic elements with quantum gates. Results of this research can stimulate the implementation of quantum computing in various fields of economy.

Key Words: quantum bits; quantum circuits; quantum computing; interpreting classical cirquits; logic elements.

Introduction

One of the key differences of quantum circuits from classical logic circuits, is the reversibility of operations performed by logic gates.

A quantum register's state is represented by a quantum wave funcion $|\psi\rangle$. This wave fuction is represented by a vector on a Bloch sphere. Quantum gates turn this vector, around different axis of the Bloch sphere.

Because of that, if a quantum gate is repeated enough times for the total rotation to reach 360 degrees (or 2π) from the starting point of the rotation, it is considered, that the initial operation was reversed and doesn't affect the following quantum circuit. In terms of rotation matrices, a gate is repeated until the total rotation matrix becomes an identity matrix, that wouldn't affect the state of a quantum register[1].

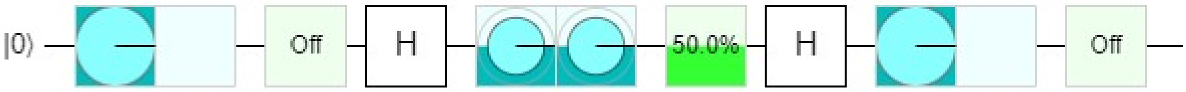


Fig. 1. Reversibility of a Hadamard quantum gate

Classical logic elements are irreversible, thus they arent directlty applicable in quantum circuits. Despite that, classical gates could be interpreted via quantum logic elements, keeping in mind their architectural differences.

Lets look at the most basic logic element – the NOT gate. In quantum computing its closest counterpart is the Pauli-X gate.

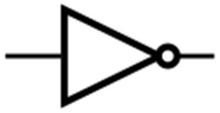


Fig. 2. Classical NOT gate, circuit representation [2]

Tab. 1. Classical NOT gate, truith table [2]

Input A	NOT A
0	1
1	0

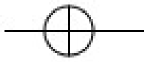


Fig. 3. Pauli-X quantum gate, circuit representation

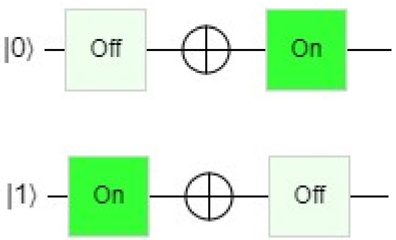


Fig. 4. Inversion of input states perforemed by a Pauli-X gate

To interpret other classical logic elements, we'll add control gates to the Pauli-X gate. The goal of the following circuits will consist of placing quantum gates in a way to simulate the truith table of the required classical logic gate.

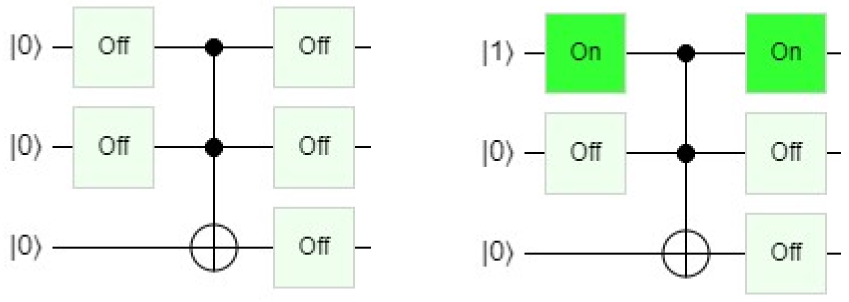
Tab. 2. Classical gate truith tables

Input A	Input B	A AND B	A OR B	A XOR B
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Interpreting the classical AND gate

Fig. 5. Classical AND gate, circuit representation [2]

The quantum counterpart of a classical AND gate is the Toffoli gate[1,3,4], the operation of the controlled Pauli-X gate depends on the values of attached control qubits.



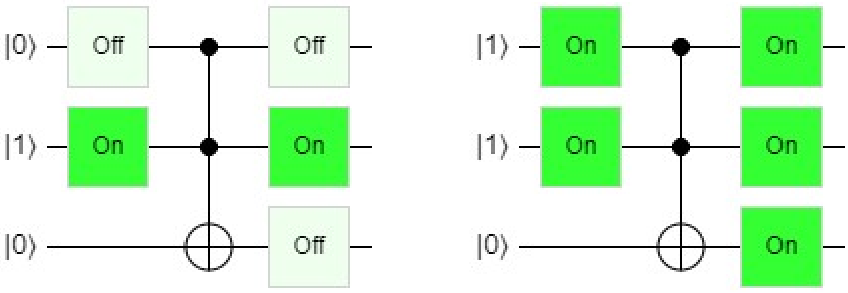


Fig. 6. The Toffoli gate (Quantum AND gate), with different input values

Interpreting the classical OR gate

Fig. 7. Classical OR gate, circuit representation [2]

Using the quantum AND gate and De Morgan's laws:

$$\overline{A \lor B} = \bar{A} \land \bar{B}$$

$$\overline{A \land B} = \bar{A} \lor \bar{B}$$

It is possible to construct a circuit for $A \lor B$:

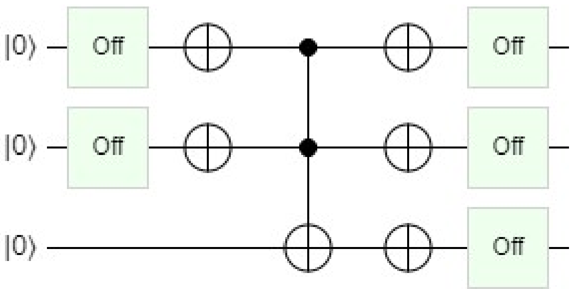


Fig. 8. Quantum OR gate, constructed via De Morgan's laws [3,5]

This circuit could be further visually simplified by using anti-controls, as represented below, with multiple different input states.

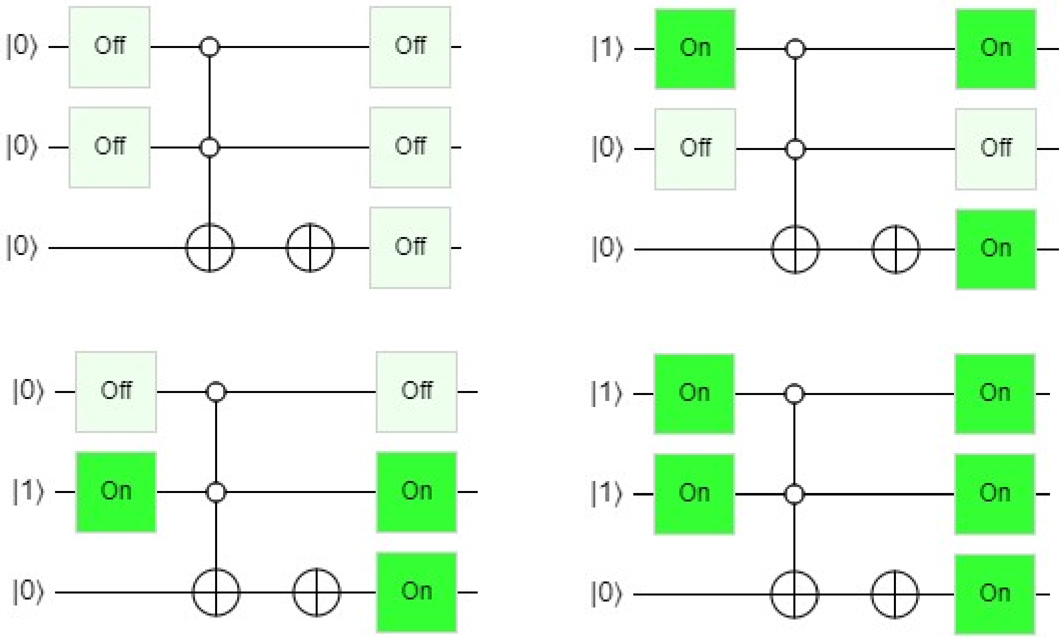


Fig. 9. Quantum OR gate tested with different input states

Interpreting the classical XOR gate [4]

Fig. 10. Classical XOR gate circuit representation [2]

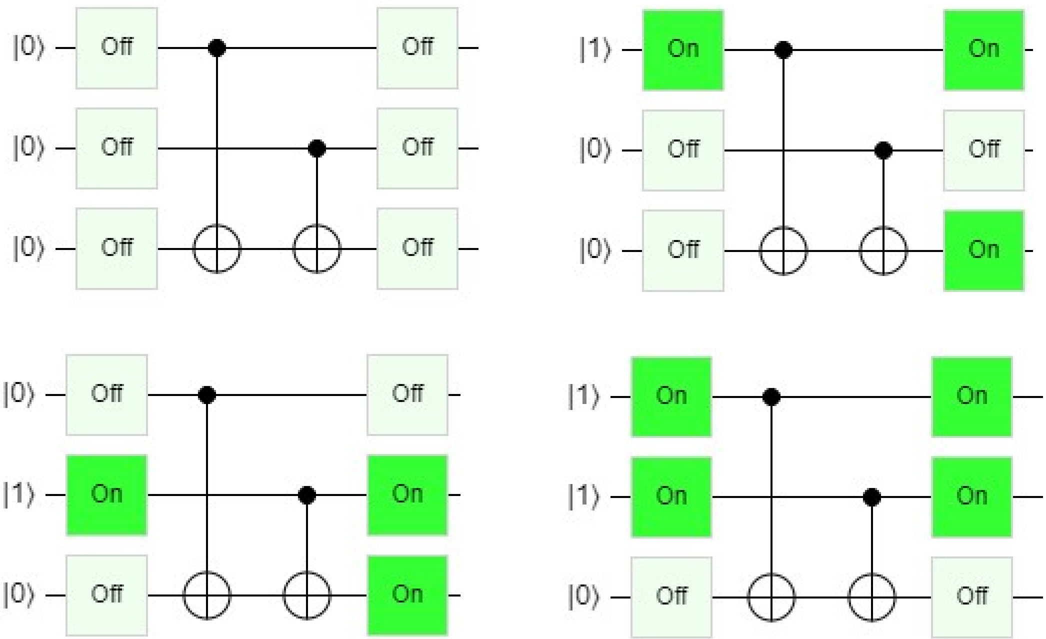


Fig. 11. Quantum XOR gate with different input values

Interpreting algorithms based on classical logic elements

After implementing classical logic gates, the next step would be interpreting algorithms based on these classical logic elements. As an example lets look at a classical single bit adder circuit with a carry bit.

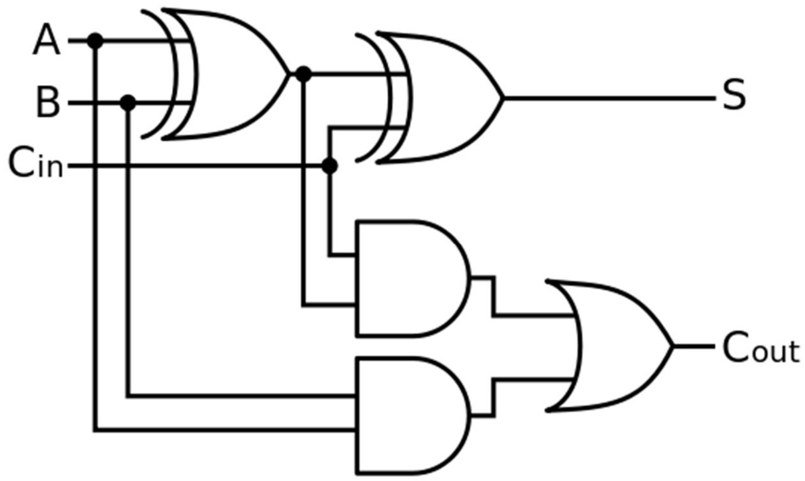


Fig. 12. Classical adder circuit with a carry bit[6]

Label blocks, representing different bits and operations are attached to some qubits and gates on figure 13, they are given only for reference and they do not affect the quantum circuit:

Bit1, Bit2 – bits, which values need to be summed.

Carry (C-In, C-out) – a carry bit (for times when $1_2 + 1_2 = 10_2$), is used as a buffer for connecting multiple adders to find sums of numbers, that consist of multiple bits.

Sum – bit with the resulting value.

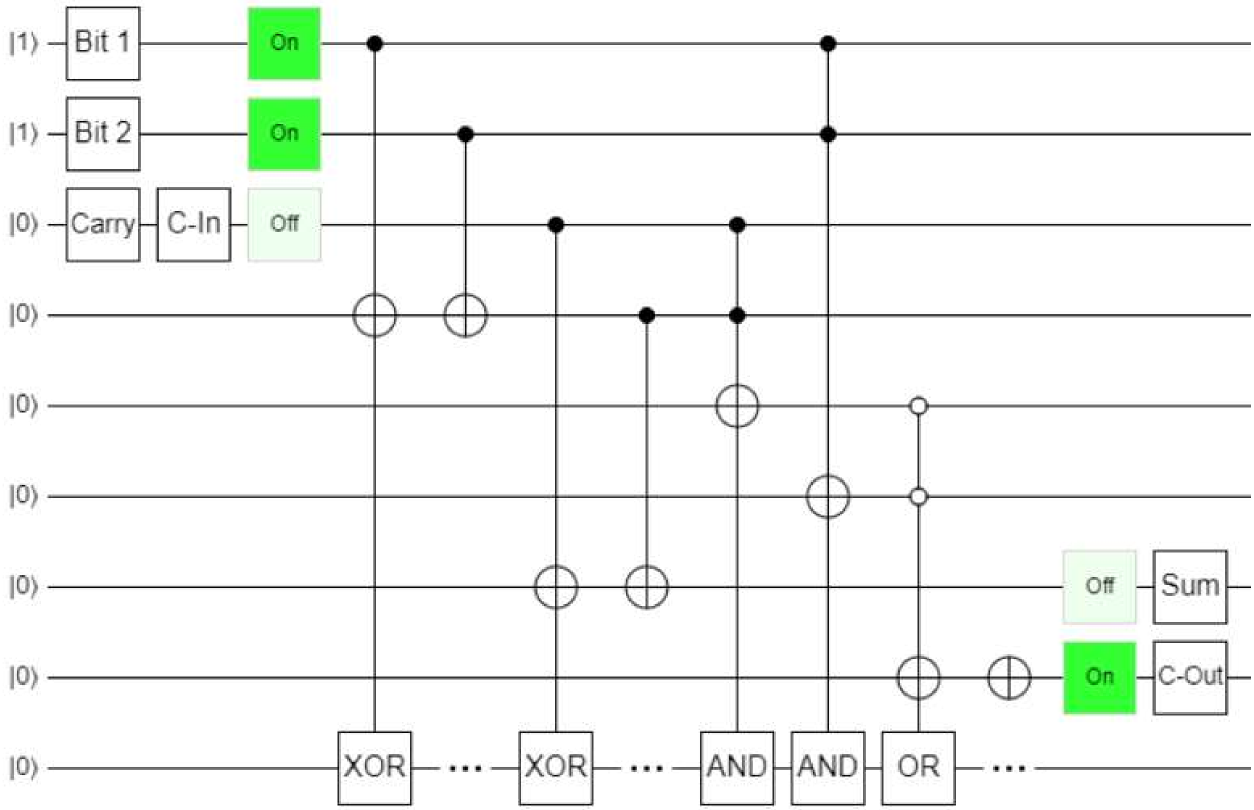


Fig. 13. Quantum circuit of the given classical adder [6]

Now, using the same approach, lets try interpreting a 2-bit binnary multiplier.

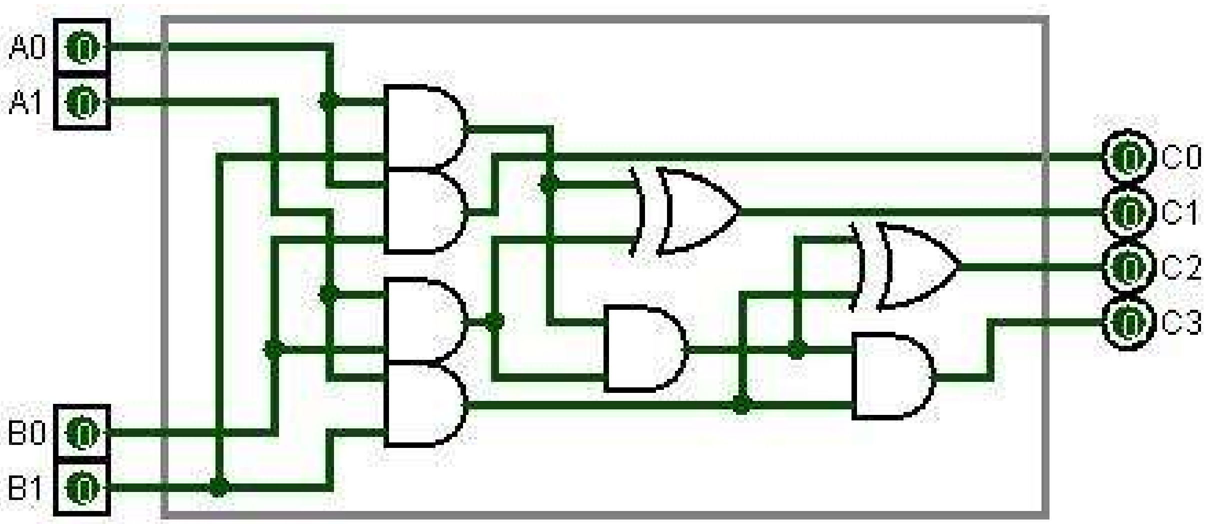


Fig. 14. Classical circuit of a 2-bit binnary multiplier [7]

The goal of this circuit is to get a 4-digit binnary number (MR 1-4), as a result of multipying two 2-digit binnary numbers.

Bit1, Bit2 – bits, storing the first number

Bit3, Bit4 – bits, storing the second number

MR 1-4 – bits, storing the result of the multiplication

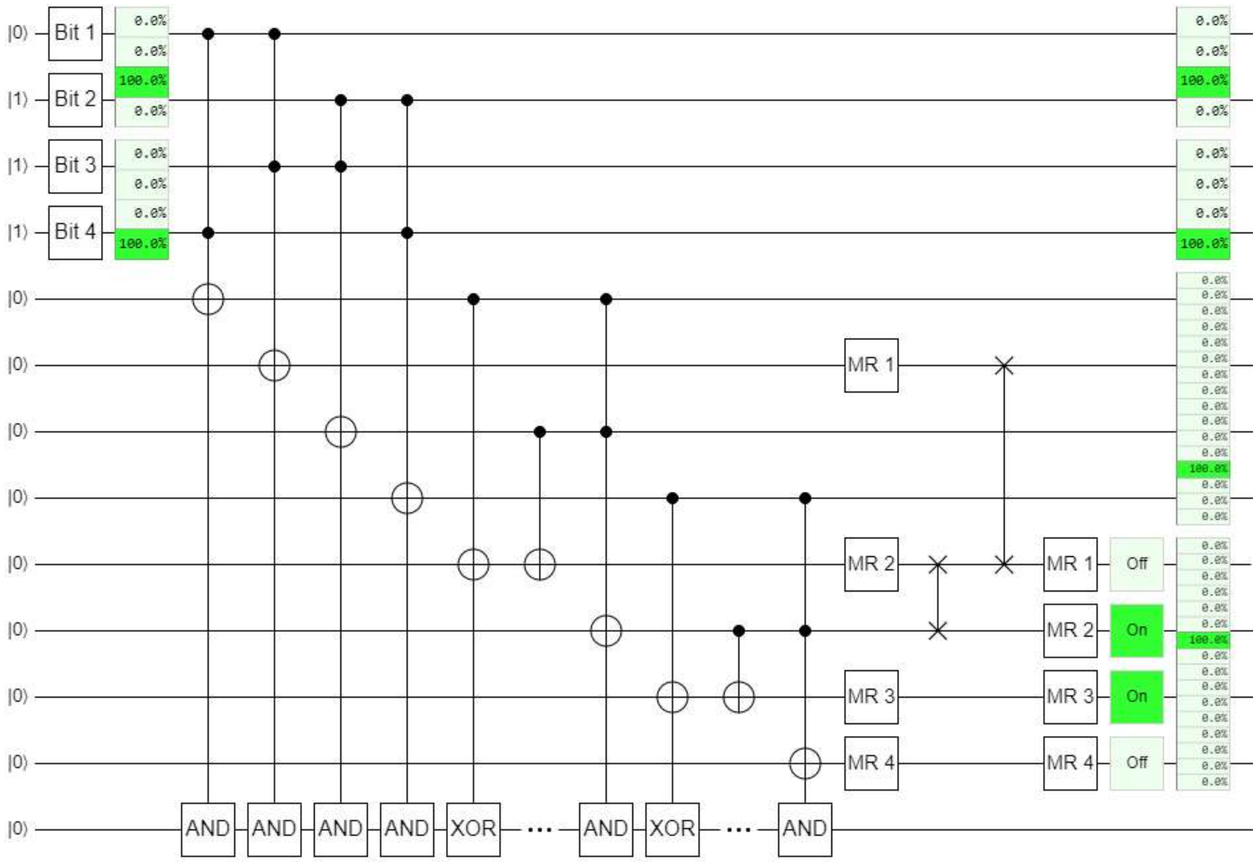


Fig. 15. Quantum circuit of the given 2-bit binnary multiplier

Further optimisation of the cirquit on fig. 15, will consist of aligning the ouput qubits, to remove unnessesary SWAP gates and for utilizing the reversibility principle for clearing bits, that don't store the output or input values, as seen on fig. 16.

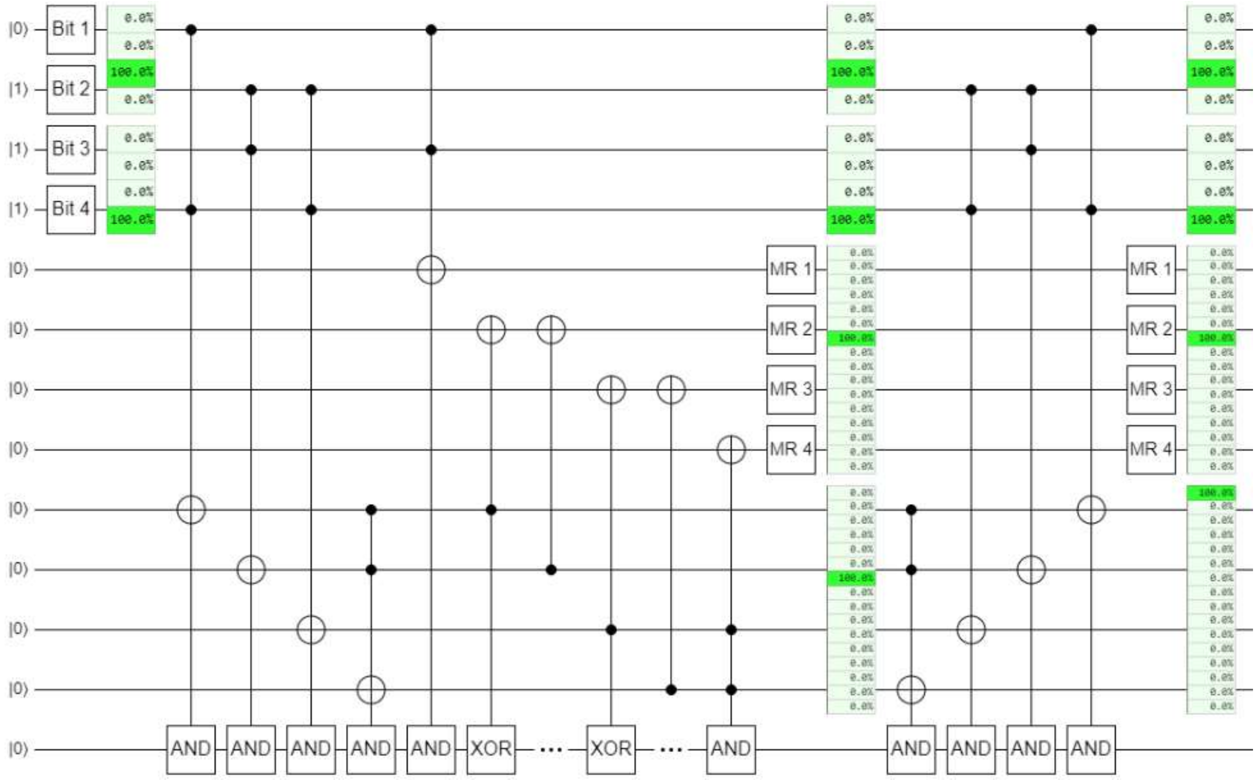


Fig. 16. Optimised quantum circuit of the given 2-bit binnary multiplier

Using classical algorithms as a part of quantum algorithms on the example of Grover's Search Algorithm

Using the interpreted circuits from above and the reversibility principle, lets try to research the practical possibilities of speeding up the problem of finding a certain value, that fits a certain search criteria, from a large array.

To get a potential speedup in the number of iterations needed to complete this task, against a classical algorithm, we started with the Grover Search Algorithm [8], that uses quantum entanglement for simultaneous checking of all input values *"X"* against a given search criteria, in a form of an oracle function.

The equation $X \cdot B = C$ in a form of an oracle function was used as the search criteria. The goal of this algorithm is to find the correct value *"X" (Fig. 17.3, Res 1-2)*.

The multiplier circuit from figure 16 was to form our oracle function [9,10].

The following example shows that the interpreted circuit works with entangled quantum states, is reversible and is ready to work as a part of a quantum circuit (circuit split across figures 17.1-17.3).

Label blocks for circuit on figure 17:

Bit-1, Bit-2 – bits, storing input values of variable X.

Res-1, Res-2 – bits outputting the resulting value X.

Bit-3, Bit-4 – input value of variable B.

C-Bits - input value of variable C.

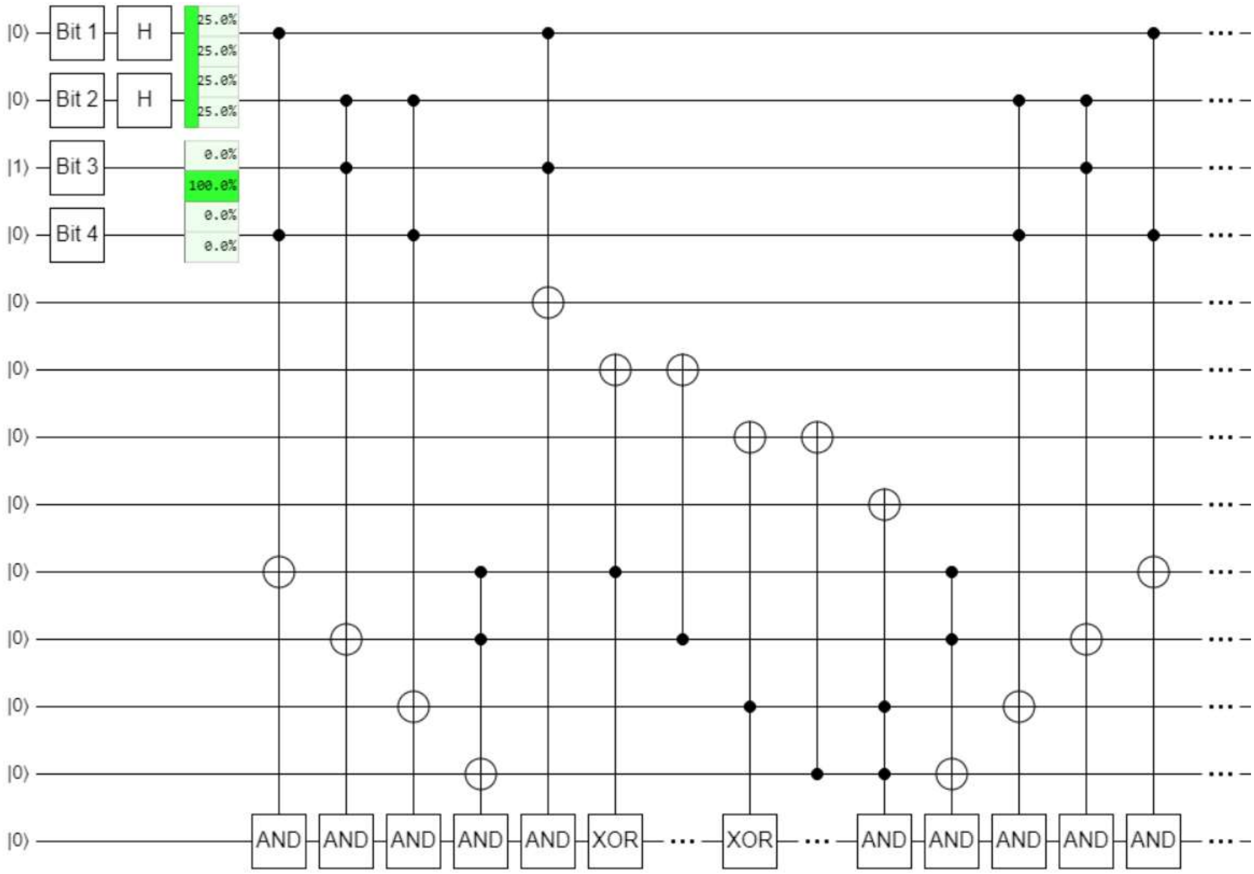


Fig. 17.1. Prepearing the input values of variable X and multiplying

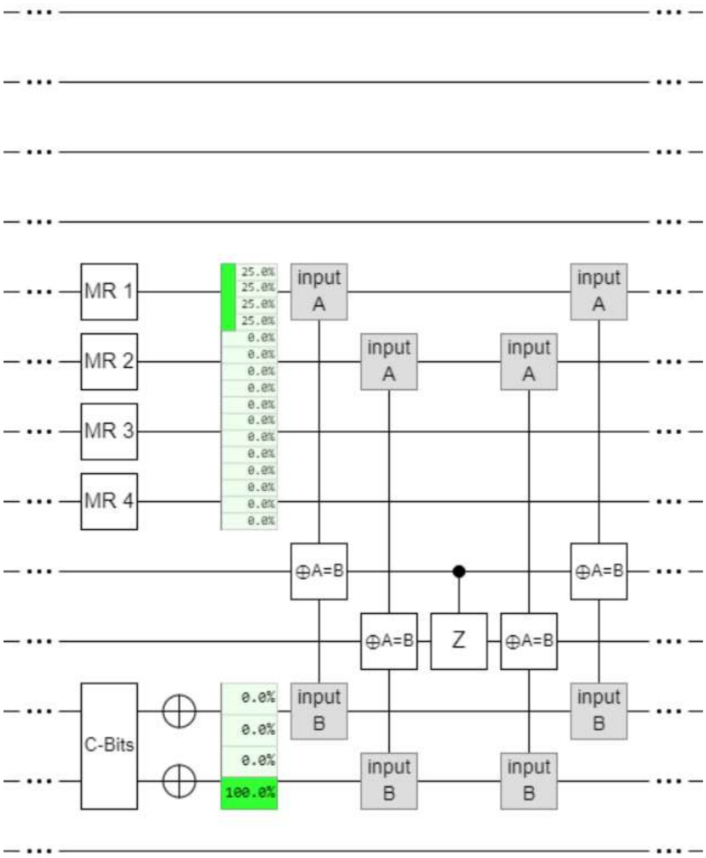


Fig. 17.2. Comparing resulting values to variable C's value and selecting the required value with the CZ gate in the oracle function

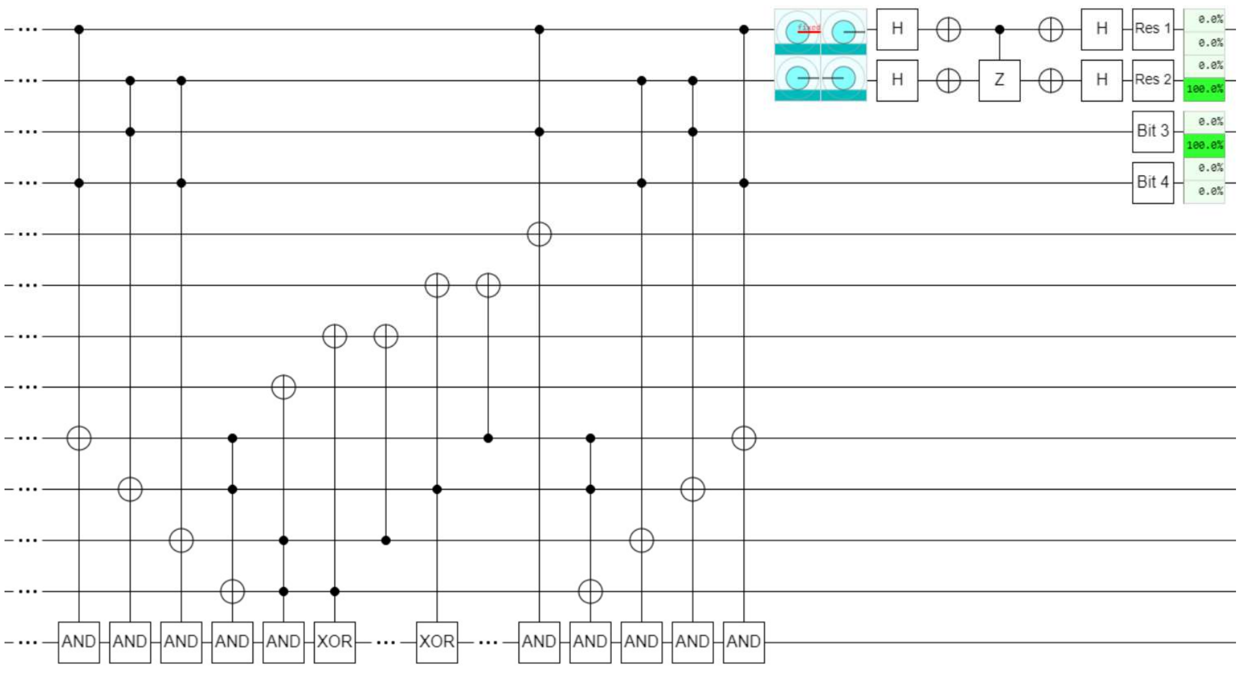


Fig. 17.3. Operation, that reverses the used multiplication function and amplitude amplification

Using the reversibility principle, we remove further influence of the used multiplication and comparation functions (figures 17.2, 17.3). After completing the oracle, the amplitude amplification step, increases the amplitude of the state, selected by the oracle function. This completes the current iteration of this Grover's algorithm and results in a form of an increased probability of measuring our desired value on *Res 1-2* bits (fig. 17.3).

Conclusion

Upon testing, the interpreted algorithms proved to be fully compatible with quantum circuits, retaining their properties even with entangled input states.

Use of interpreted classical circuits in quantum algorithms can, in theory, provide speedup for solving more complex and more specific problems. Using interpreted and already proven classical algorithms inside quantum circuits could save resources in development of larger and more field-specific quantum circuits, which can greatly stimulate the implementation of quantum calculations in various fields of life and economy.

References

1. Quantum Logic gate // Wikipedia - the free encyclopedia, URL: https://en.wikipedia.org/wiki/Quantum_logic_gate.
2. Logic gate // Wikipedia - the free encyclopedia, URL: https://en.wikipedia.org/wiki/Logic_gate
3. Implementing "Classical AND Gate" and "Classical OR Gate" with a quantum circuit // Quantum Computing Stack Exchange. URL: https://quantumcomputing.stackexchange.com/q/5829
4. Classical XOR gate in Quantum Circuit // Quantum Computing Stack Exchange. URL: https://quantumcomputing.stackexchange.com/q/4153
5. Transferring classical OR gate in a quantum gate // Quantum Computing Stack Exchange. URL: https://quantumcomputing.stackexchange.com/q/6119

6. How does a quantum computer do basic math at the hardware level? // Quantum Computing Stack Exchange. URL: https://quantumcomputing.stackexchange.com/q/1289
7. Binary multiplier // Wikipedia. URL: https://en.wikipedia.org/wiki/Binary_multiplier
8. Grover L.K.: A fast quantum mechanical algorithm for database search // Cornell University Library. URL: https://arxiv.org/abs/quant-ph/9605043
9. Cesar B. Pronin, Andrey V. Ostroukh, Researching the Possibilities of Creating Mathematical Oracle Functions for Grover's Quantum Search Algorithm. URL: https://vixra.org/abs/2004.0704
10. Pronin C. B., Ostroukh A.V., Pronin B.V., Vasiliev Y.-, Kotliarskiy E.: Development of a Quantum Algorithm Based on Quantum Parallelism for Finding the Shortest Path in a Graph // APRN Journal of Engineering and Applied Sciences, Vol. 14, No. 4, Feb. 2019, ISSN 1819-6608 – Scopus.
11. Quirk // Quirk – online quantum computer simulator. URL: http://algassert.com/quirk

Author details

Andrey Vladimirovich Ostroukh
Russian Federation, full member of RAE, Doctor of Technical Sciences, Professor, Department «Automated Control Systems».
State Technical University – MADI, 125319, Russian Federation, Moscow, Leningradsky prospekt, 64. Tel.: +7 (499) 151-64-12. http://www.madi.ru
ostroukh@mail.ru

Cesar Borisovich Pronin
Russian Federation, Postgraduate Student, Department «Automated Control Systems».
State Technical University – MADI, 125319, Russian Federation, Moscow, Leningradsky prospekt, 64. Tel.: +7 (499) 151-64-12. http://www.madi.ru
caesarpr12@gmail.com