# Implementing multi-controlled X gates using the quantum Fourier transform

Vladimir V. Arsoski<sup>1\*</sup>

<sup>1\*</sup>The Department of Microelectronics and Technical Physics, School of Electrical Engineering - University of Belgrade, Bulevar kralja Aleksandra 73, Belgrade, P.O. Box 35–54, Serbia.

Corresponding author(s). E-mail(s): vladimir.arsoski@etf.bg.ac.rs;

#### Abstract

Quantum computing has the potential to solve many complex algorithms in the domains of optimization, arithmetics, structural search, financial risk analysis, machine learning, image processing, and others. Quantum circuits built to implement these algorithms usually require multi-controlled gates as fundamental building blocks, where the multi-controlled Toffoli stands out as the primary example. For implementation in quantum hardware, these gates should be decomposed into many elementary gates, which results in a large depth of the final quantum circuit. However, even moderately deep quantum circuits have low fidelity due to decoherence effects and, thus, may return an almost perfectly uniform distribution of the output results. This paper proposes a different approach for efficient cost multi-controlled gates implementation using the quantum Fourier transform. We show how the depth of the circuit can be significantly reduced using only a few ancilla qubits, making our approach viable for application to noisy intermediate-scale quantum computers. This quantum arithmetic-based approach can be efficiently used to implement many complex quantum gates.

**Keywords:** Quantum computing, Quantum algorithms, Multi-controlled gates, Quantum Fourier transform, Ancilla qubits

 $\mathbf{MSC\ Classification:}\ 03\mathrm{G}12\ ,\,81\mathrm{P}68$ 

#### 1 Introduction

Recent progress in the fabrication of quantum computers brings the accelerated development of quantum algorithms that can use the laws of quantum mechanics to solve some tasks faster than classical algorithms. It is well-known that the classes of problems related to quantum chemistry and quantum physics are very challenging for classical computation [1]. When solved using quantum computers, they can achieve up to exponential speedup [2, 3]. However, disciplines such as machine learning, finance, cryptography, optimization, image processing, and linear algebra can also benefit from quantum advantage [4–11]. Many of these algorithms target the desired state for which multi-controlled (MC) gates are used. The most famous one is Grover's search algorithm [12] that utilizes MC gates in diffusion operator. For that reason, efficient implementation of MC gates becomes essential for applying these algorithms to the genuine quantum device.

A systematic approach for decomposing multi-controlled gates is described in Ref. [13]. This approach was subsequently optimized by alternative implementations, such as using  $C - R_x$  rotations [14, 15], partial removal of some gates from the circuit at the price of phase relativization [14, 16], or using ancilla qubits [17]. It is worth noting that the multi-controlled X (MCX) gate implementation can be straightforwardly generalized to implement an arbitrary multi-controlled single-qubit unitary gate [13, 18].

Our approach is inspired by the quantum Fourier transform (QFT) [18] and its approximate (AQFT) form [19]. It was shown that many useful arithmetic operations may be approximated using the QFT [20–22]. We will extend this set by MCX gates. The paper is organized as follows: section 2 provides a brief description of theoretical preliminaries for MCX implementation, section 3 gives implementation details and methods for optimization using ancilla qubits, and section 4 concludes the paper. The proof of equivalence of our implementation and the standard one is given in Appendix A

#### 2 Theoretical preliminaries

When viewed in the computational Z-basis, an n-qubit multi-controlled-NOT gate performs a conditional  $\pi$ -rotation on the most significant (target) qubit around the x-axis based on the state of  $n_c = n - 1$  lower qubits. We should note that the choice of basis states is not unique since any two orthogonal vectors of a qubit system may be used as the computational basis states. Nonetheless, using the proper set of rotations, the computational basis can be converted to the desired.

For simplicity, we will work with the qubits  $|a_k\rangle$  in a pure state. Then  $|a\rangle = |a_n\rangle \otimes |a_{n_c}\rangle = |a_n\rangle \otimes |a_{n-1}\rangle \otimes ... \otimes |a_2\rangle \otimes |a_1\rangle \in Z_{2^n}$  is related to the binary representation of integer number a, where integer  $a_{n_c} = a - a_n \cdot 2^{n-1}$ . In the classical case, an arithmetic operation performed on the integer a stored in the n-bit register will update it to a value that is congruent modulo  $2^n$ . Increment of a by one results in the value of the highest bit given by the Boolean expression  $\overline{x}^{n_c} = a_n^{C^{n_c}X} = (a_1 \cdot a_2 \cdot \cdots a_{n-1}) \oplus a_n$ , where the lower  $n_c$  bits upgrade to a binary representation of  $a_{n_c} + 1 \mod 2^{n_c}$ . Applying a subsequent decrement by one only to the lower  $n_c$  bits restores them to initial values,

where the highest bit remains unchanged, thus storing the result of the controlled-NOT operation. From the standpoint of quantum computation viewed in the  $\{|0\rangle, |1\rangle\}$ -basis, this operation corresponds to the  $n_c$ -controlled X operation, usually denoted by  $C^{n_c}X$ . Therefore, we will use this simple logic for an alternative MCX gate.

The method employed for addition computes QFT on the first addend and then, based on the second addend, evolves it into QFT of the sum. Applying the inverse QFT recovers the sum to the computational basis. If we want to add a priory known classical value to quantum data, we only have to implement a series of corresponding rotations [20]. To implement increment/decrement by one, we use a small set of rotations that can be executed simultaneously in a single time slice. The  $QFT_n$  maps n-qubit state  $|a\rangle$  to:

$$QFT_n|a\rangle = |\widetilde{a}\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{i2\pi ak/N} |k\rangle = |\phi_n(a)\rangle \otimes |\phi_{n-2}(a)\rangle \otimes \cdots \otimes |\phi_1(a)\rangle, \quad (1)$$

where  $N = 2^n$  and

$$|\phi_k(a)\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i2\pi a/2^k}|1\rangle). \tag{2}$$

The Fourier transform is carried through applying the Hadamard gate to each qubit k, followed by a sequence of two-qubit  $C_j - R_{k,k-j+1}$  gates that conditionally apply rotations  $R_{k-j+1}$  to the  $k^{\text{th}}$  qubit based on lower qubits (j < k) [20]. A rotation of a single qubit by an angle  $2\pi/2^m$ , where  $m \in \mathbb{Z}$ , is represented by the matrix:

$$R_m = \begin{bmatrix} 1 & 0 \\ 0 & e^{i2\pi/2^m} \end{bmatrix}. \tag{3}$$

An increment/decrement of a by one in the transform range is achieved by applying an unconditional rotation  $R_m$  to each qubit m. The n-qubit phase gate acting on  $|\tilde{a}\rangle$  is:

$$P_{\pm 1,n}|\widetilde{a}\rangle = R_n^{\pm 1} \otimes \cdots \otimes R_1^{\pm 1}|\widetilde{a}\rangle = |\widetilde{a} \pm \widetilde{1} \bmod \widetilde{N}\rangle. \tag{4}$$

The result  $|\tilde{a} \pm \tilde{1} \mod \tilde{N}\rangle$  is returned to the computational basis using the inverse of the  $QFT_n$   $(IQFT_n)$ :

$$IQFT|\widetilde{a} \pm \widetilde{1} \bmod \widetilde{N}\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{-i2\pi(a\pm 1)k/N} |k\rangle = |a \pm 1 \bmod N\rangle.$$
 (5)

To perform the  $n_c$ -controlled X operation, we increment the n-qubit data by one, obtaining the desired result stored in the highest qubit. Then, we restore the control qubits to their original value ("uncomputation") by carrying out a decrement by one on the data stored in the register composed of the lower  $n_c$  qubits. The complete sequence is as follows:

$$|a\rangle \xrightarrow{QFT_n} |\widetilde{a}\rangle \xrightarrow{P_{+1,n}} |\widetilde{a}+\widetilde{1} \bmod \widetilde{N}\rangle \xrightarrow{IQFT_n} |a+1 \bmod N\rangle$$

$$= |\overline{x}^{n_c}\rangle \otimes |a_{n_c} + 1 \mod \frac{N}{2}\rangle$$

$$\xrightarrow{I \otimes QFT_{n_c}} |\overline{x}^{n_c}\rangle \otimes |\widetilde{a}_{n_c} + \widetilde{1} \mod \frac{\widetilde{N}}{2}\rangle$$

$$\xrightarrow{I \otimes P_{-1,n_c}} |\overline{x}^{n_c}\rangle \otimes |\widetilde{a}_{n_c} \mod \frac{\widetilde{N}}{2}\rangle$$

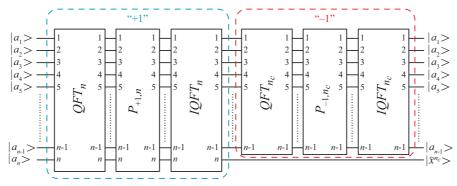
$$\xrightarrow{I \otimes IQFT_{n_c}} |\overline{x}^{n_c}\rangle \otimes |a_{n_c}\rangle. \tag{6}$$

Although our implementation seems quite different from the standard MCX circuit, it is essential to note that they share the same unitary operator, as shown in the Appendix A.

#### 3 Implementation and discussion

To implement circuits, Qiskit (ver. 0.42.1) [23] is used. Initially, we analyze circuits that do not require ancilla qubits. Eq. 6 gives details on the gates needed to implement an MCX. A schematic diagram of the n-qubit MCX circuit is displayed in Fig. 1. First, the QFT is appended to the n-qubit input. The ordering of control qubits in the  $QFT_n$  circuit is irrelevant, but the most significant qubit should be the controlled (output) qubit. In succeeding subcircuits, this arrangement of qubits must be respected. The increment by one is achieved by applying a series of phase gates, as described by Eq. 4. Since these phase gates act on different qubits, they should be executed simultaneously, that is, at a single time slice. The IQFT reverts the incremented value to the computational basis, thereby setting the result of the MCX operation in the most significant qubit. The operator, that executes the increment by one, is given by the  $IQFT_n P_{+1,n} QFT_n$  (it is labeled by "+1" and framed by a blue dashed line in Fig. 1). Then, we apply the inverse operator to the control qubits  $IQFT_{n_c}$   $P_{-1,n_c}$   $QFT_{n_c}$ (labeled by "-1" and framed by a red dashed line in Fig. 1). It carries out the decrement by one, thus restoring their initial value. It is important to note that the proper ordering of qubits for this inverse operation should match the ordering of the lowest  $n_c = n - 1$  qubits.

The number of gates needed for a standard QFT is n(n+1)/2 [18, 20]. After closely examining the schematic diagram of the QFT [20], it becomes clear that every wireline starts with the Hadamard gate, followed by a series of conditional rotations controlled by the lower-order qubits that have not yet undergone the Hadamard gate. This setup imposes specific restrictions for parallel execution. The diagram in Fig. 2(a) shows the six-qubit QFT with clustered gates that can be executed simultaneously in a quantum computer architecture that supports interaction between arbitrary pairs of qubits. We will be using the term "fully-connected" (FC) when referring to this architecture. Only gates acting on different qubits can be executed at a single time slice. To perform the QFT in the FC architecture (QFT-FC), each wireline  $|a_k\rangle$ , except for the one carrying the most significant qubit, must be appended with the  $C_k - R_{k+1,2}$  gate. Thereby, the  $k^{\text{th}}$  qubit will control the  $R_2$  rotation applied to the  $(k+1)^{\text{th}}$  qubit. Once this is done, the Hadamard gate can act on the  $k^{\text{th}}$  qubit. Notice that these two gates cannot be executed simultaneously as both act on the same  $k^{\text{th}}$  qubit. In



**Fig. 1** A schematic diagram of n-qibit QFT-based MCX gate. QFT, P, and IQFT denote the quantum Fourier transform (Eq. (1)), the phase gates (Eq. (4)), and the inverse of QFT (Eq. (5)), respectively. The part of the quantum circuit labeled "+1"/"-1" for increment/decrement is framed by a blue/red dashed line.

the single odd time slice, we group the Hadamard gate acting on the  $k^{\rm th}$  qubit with  $C_{k-p}-R_{k+p,2p+1}$  gates, where  $0. The subsequent even time slice concentrates <math>C_{k-q}-R_{k+q-1,2q}$  gates, where  $0< q \leq \min\{k-1,n-k+1\}$ . It is straightforward to conclude that at most  $\lceil n/2 \rceil$  gates will be executed simultaneously and that the QFT can be performed in  $f_{t,slots}^{QFT}(n)=2n-1$  time slots, provided that the quantum hardware implements an arbitrary controlled rotation as a basic operation. This number of time slots is the lower bound for the QFT's time complexity. From Fig. 2(a), one may find that QFT uses  $f_H^{QFT}(n)=n$  Hadamard gates and  $f_{C-R}^{QFT}(n)=n(n-1)/2$  controlled rotations. The total number of non-elementary gates is  $f_{gates}^{QFT}(n)=n(n+1)/2$ . To implement MCX in FC architecture, we use n-qubit and (n-1)-qubit QFT-P-IQFT block of gates. The number of  $R_m$  gates in P is  $f_{gates}^{P}(n)=n$  and all can be executed simultaneously. The total number of non-elementary gates used is  $f_{gates}^{MCX-FC}(n)=2*f_{gates}^{QFT}(n)+f_{gates}^{P}(n)+2*f_{gates}^{QFT}(n-1)+f_{gates}^{P}(n-1)=2n^2+2n-1$ . The number of time slots for n-qubit MCX execution is  $f_{solts}^{MCX-FC}(n)=2*f_{solts}^{QFT}(n)+2*f_{solts}^{QFT}(n-1)+2=(8n-6)$ . To optimize circuit complexity one may want to consider using an approximate

To optimize circuit complexity one may want to consider using an approximate QFT (AQFT). From Eq. 3, we infer that the rotation angle gets very small as m gets large. Thus, the rotation matrix  $R_m$  approaches the identity matrix. Additionally, it is crucial to consider whether these rotation gates can be executed with a certain level of tolerance in practical applications. Without such precision, the effectiveness of these gates can be in question. Studies have shown that a truncated QFT may provide greater accuracy than a full QFT in the presence of decoherence [19]. The optimal value for m needed is estimated to  $\lceil \log_2 n \rceil$ . This will reduce the number of essential operations required to  $f_{qates}^{AQFT} = (2n - \lceil \log_2 n \rceil)(\lceil \log_2 n \rceil - 1)/2 \approx n\lceil \log_2 n \rceil$  [19, 20], while at most  $\lceil \lceil \log_2 n \rceil/2 \rceil$  gates will be executed simultaneously. Therefore, using AQFT will not reduce the minimum circuit depth but only the total number of gates needed and the maximum number of gates executed simultaneously. Thus, (8n-6) is the lower bound for the time complexity estimation. The total number of non-elementary gates used for AQFT-based n-qubit MCX is  $f_{gates}^{AMCX-FC}(n) =$ 

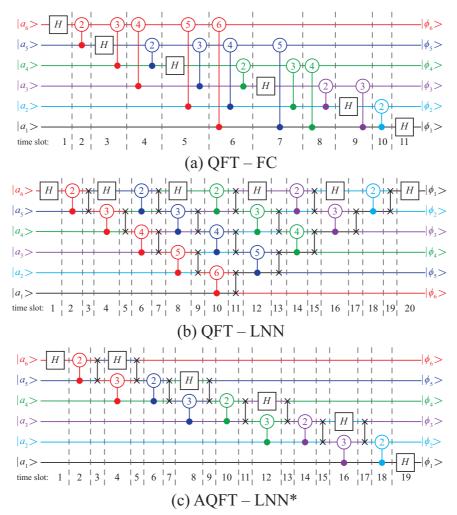


Fig. 2 A schematic diagram of the six-qubit (a) QFT in the fully connected (QFT-FC), (b) QFT in the linear-nearest-neighbor (QFT-LNN), and (c) optimized AQFT in the LNN (AQFT-LNN\*) quantum computer architecture. Controlled  $C_j - R_{jl,m}$  rotations are denoted with a line pointing from the control  $j^{\rm th}$  qubit to the target  $j^{\rm th}$  qubit, where it ends with a circle. This circle indicates rotation gate  $R_m$  with index m inscribed in it. To make it easier to visualize, rotational gates applied to different qubits have different colors. Gates are divided into time slots by vertical gray dashed lines, where the index of the time slot is explicitly denoted at the bottom. The standard procedure to get approximate forms of circuits displayed in panels (a) and (b) is to remove controlled rotations denoted by index m > 3.

 $2*f_{gates}^{AQFT}(n)+f_{gates}^{P}(n)+2*f_{gates}^{AQFT}(n-1)+f_{gates}^{P}(n-1)$  which is the lower bound for the space complexity calculation.

Still, in many architectures, each qubit has a finite number of neighbors bounded by a number l [24]. To apply conditional rotation between two qubits, they must be neighbors. If  $[\log_2 n] > l$ , we need to add SWAP gates, which increases the circuit depth. In recent studies, the lower bound of time slices needed for the QFT execution

is estimated to be [(2+2/l)n+O(1)] [25]. The most restrictive case is for the linearnearest-neighbor (LNN) computer architecture, where two-qubit gates are allowed only between qubits whose subscript values differ by one (l=2). To make qubits nearest neighbors, we must swap some pairs of qubits before controlled rotation can be applied. A schematic diagram of the six-qubit QFT in the LNN architecture (QFT-LNN) is displayed in Fig. 2(b). Following the one wireline, only the first rotation  $C_{n-1} - R_{n,2}$ gate can be applied without swapping qubits since acting on the nearest neighbors. The following controlled rotation should be executed on the next-nearest-neighbors, so we swap  $n^{\text{th}}$  and  $(n-1)^{\text{th}}$  qubit. Therefore, formerly  $(n-1)^{\text{th}}$  and  $(n-2)^{\text{th}}$  qubits are no longer neighbors, so we have to perform another swap before the second set of rotations, and so on. Notice that each set of elementary gates executed at a single time slice in Fig. 2(a) has a corresponding pair of time slices in Fig. 2(b) that are used to swap qubits and perform the equivalent set of operations, respectively. Consequently, we will have additional  $f_{t,slots}^{SWAP(QFT)}(n) = (2n-3)$  time slots for swapping qubits with  $f_{gates}^{SWAP(QFT)}(n) = (n-1)(n-2)/2$  SWAP gates. Depending on the basis used for calculation, each SWAP gate is implemented using a few basis gates. Notice that qubit ordering is reversed. When implementing MCX, instead of using additional  $\lfloor n/2 \rfloor$ SWAP gates at QFT(IQFT) output, the subcircuits succeeding a QFT or IQFT should be appended with inputs swapped. Alternatively, QFTs and IQFTs inputs should have reverse ordering. To implement n-qubit MCX in LNN architecture, the number of time slots required is  $f_{t.slots}^{MCX-LNN}(n) = f_{t.slots}^{MCX-FC}(n) + 2 * f_{t.slots}^{SWAP(QFT)}(n) + 2$  $f_{t.slots}^{SWAP(QFT)}(n-1) = 16n-22$  which is the upper bound for the time complexity estimation. The number of non-elementary gates is  $f_{gates}^{MCX-LNN}(n) = f_{gates}^{MCX-FC}(n) + \frac{1}{2} f_{gates}^{MCX-FC}(n)$  $2*f_{gates}^{SWAP(QFT)}(n) + 2*f_{gates}^{SWAP(QFT)}(n-1) = 4n^2 - 10n + 7 \text{ which will be the upper states}$ bound to estimate the space complexity.

Again, the question is whether the circuit can benefit from approximation. The straightforward procedure to obtain an AQFT in the LNN (AQFT-LNN) is to remove controlled rotations with an index  $m > [\log_2 n]$ , which will reduce the number of controlled phases as for the AQFT-FC. The circuit will have the depth and the number of SWAP gates used as the QFT-LNN. Attempting to reduce the number of SWAP operations will increase the depth of the circuit in return. Therefore, we consider the former method optimal. The exception is n < 6 when we perform only  $C - R_2$  gates on the adjacent qubits. Thus, we do not need SWAP gates, so the AQFT-FC and AQFT-LNN will match. The second exception is  $6 \le n < 12$  when we need  $C - R_2$ and  $C - R_3$  gates. To perform consecutive  $C - R_2$  and  $C - R_3$  rotations, we have to swap one of the next-nearest-neighbors qubits to do  $C-R_3$  rotation and then swap back to restore qubit ordering for the following  $C-R_2$  rotation. A schematic diagram of the optimized six-qubit AQFT in the LNN architecture (AQFT-LNN\*) is displayed in Fig. 2(c). This implementation is less complex than the AQFT-LNN obtained from the circuit in Fig. 2(b) by removing controlled rotations denoted by m > 3. Moreover, qubits ordering is not reversed.

Up to this point, we assumed that a quantum computer inherently implements the single-qubit  $R_m$  and the two-qubit  $C - R_m$  rotation, thus making them part of the universal gate set. In a genuine quantum device, this basis set (that is, the native

gate set - NGS) is formed from the single- and two-qubit gates that have high fidelity and are usually the best choice for standard quantum circuit implementations. For example, superconducting hardware uses the single-qubit  $R_z$  gate and the two-qubit C-X gate rather than  $R_m$  and  $C-R_m$ .

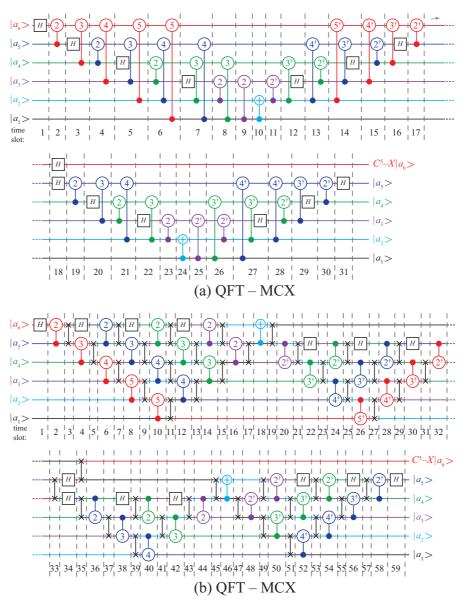


Fig. 3 Optimized QFT-based MCX in (a) FC, and (b) LNN architecture.

A realistic assessment of the time and space complexities can be found from the circuit decomposed in the NGS. Detailed optimization and decomposition are elaborated in Appendix B. An optimized MCX circuits in the FC and LNN architectures are shown in Fig. 3(a) and (b), respectively. Due to the merging controlled-phase and increment/decrement phase gates, the circuit reduces to (8n-17) time slots in the FC architecture, as explained in Appendix B. At most (8n-20) additional SWAP-gate slots are required for implementation in a qubits network with linear connectivity, so (16n-37) time slots are needed in the LNN architecture. Counting in the NGS, the depth of MCX is (32n-80) in the FC and (56n-146) in the LNN (for n>3). For n=3 we use at most two SWAP gates in the LNN. If the target wireline can be placed between control wirelines in the Toffoli gate, we can avoid using SWAP gates. Therefore, the time complexity of the Toffoli gate in the FC and LNN will be the same.

FC-MCX uses (4n-10) Hadamard gates,  $(2n^2-6n+3)$  controlled phase gates, and two C-NOTs. The number of elementary gates in H and  $C-R_m$  is 3 and 5, respectively. Two  $R_z$  gates annihilate between  $C_j-R_m$  and  $C_{j-1}-R_m^{\dagger}$  acting on the same target qubit, as discussed in Appendix B. Therefore, we will use 2(n-2) elementary gates less in "+1" and 2(n-3) in "-1". The total number of elementary gates is  $(10n^2-22n-5)$ . If we simplify the circuit using approximate QFT, the best solution is to keep  $C-R_m$  gates with m value up to  $[\log_2 n]$  after merging controlled phase gates in QFT and IQFT. Thereby, there will be a few more  $C-R_m$  gates than if we approximate QFT, IQFT, and P separately. This way, the approximation will be better. Thus, we use approximate "+1" and "-1" circuits rather than approximate QFT(IQFT). The number of  $C-R_m$  gates will reduce to  $2([\log_2 n]-1)(2(n-1)-[\log_2 n])$  for n>3, and the Toffoli gate will use only three  $C-R_2$  gates. Approximate MCX in the FC architecture will use  $10([\log_2 n]-1)(2(n-1)-[\log_2 n])+10n-23$  elementary gates which is the space complexity lower bound.

LNN-MCX uses at most  $(2n^2-6n+6)$  swap gates. In adjacent SWAP and C-NOT, two C-NOTs annihilate. Therefore, the number of C-NOTs used for swapping is  $6n^2-18n+14$ . Assuming that there must be a defined arrangement (ascending or descending) of qubits, the number of elementary gates used in the LNN architecture is  $(16n^2-40n+9)$ , which is the space complexity upper bound. If the approximate form is used, the number of elementary gates reduces to  $10([\log_2 n]-1)(2(n-1)-[\log_2 n])+6n^2-8n-9$ .

To compare the time and space complexities of different implementations, we plot the circuit depth and the number of gates needed as a function of the number of qubits in Figs. 4(a) and (b), respectively. For all considered implementations, the number of the time slices depends linearly on the number of qubits in the MCX, as displayed in Fig. 4(a). QFT-based and AQFT-based implementations are executed in an equal number of time slices. However, the time slice count depends on the number of SWAP gates needed for implementation in a particular computer architecture. The LNN uses the most SWAP gates, while FC architecture doesn't require any. The number of time slices needed to execute MCX in the FC and LNN set the lower and the upper bound for the time complexity, respectively.

Also, we compare the number of elementary gates  $(N_{e.g.})$  required for different implementations. For the QFT-based n-qubit MCX, there is a square increase in the

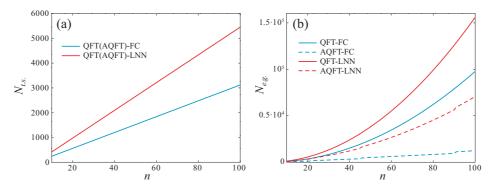


Fig. 4 (a) The number of time slices  $(N_{t.s.})$ , and (b) the number of elementary gates  $(N_{e.g.})$  as a function of the number of qubits in the MCX circuit implemented in the fully connected (FC - blue line) and the linear-nearest-neighbor (LNN - red line) architecture. Solid (dashed) lines are used for the QFT(AQFT)-based MCX circuit.

number of elementary gates with n for both FC and LNN implementation, as displayed by solid lines in Fig. 4(b). The QFT-LNN has a higher positive slope since there is also an additional square increase in the number of SWAP gates with n. This implementation uses the highest number of elementary gates and is considered the upper bound in the gates count. The number of rotation gates in AQFT-based implementations is a function of  $[\log_2 n]$ , which is a stair-step function. It is reflected in a stair-wise dependence of the number of elementary gates on n. This is easy to comprehend because there is a steep change in the number of controlled phase gates used in an AQFT with  $[\log_2 n]$ . Since there is no need for SWAP gates in the AQFT-FC, the number of elementary gates is the smallest, thus setting the lower bound. The AQFT-LNN implementation additionally uses  $O(n^2)$  SWAP gates, which result in a larger count of elementary gates to the number of rotation gates required. Therefore, it is hard to notice that the increase in  $N_{e.g.}$  is stair-wise, which can be inferred from the dashed red line in Fig. 4(b). This differs from the AQFT-FC (dashed blue line), where  $N_{e.g.}$  is a linearly increasing piecewise function with n.

Low quantum circuit complexity can lead to more efficient quantum computation that is less prone to errors. One way to reduce the depth of a circuit is by utilizing ancilla qubits. Suppose we have a certain number (r) of ancilla qubits that are in the initial state  $|0\rangle$  (see Fig. 5). We divide the control qubits into r equal groups, each containing  $\Delta n_c$  qubits. In the general case, we may have  $n_r = n_c - r \cdot \Delta n_c$  extra control qubits, which we'll address later. For each group, we use QFT-based "increment by one circuit" on  $\Delta n_c$  qubits and one ancilla as the "carry out" qubit. The operation will switch ancilla qubit to  $|1\rangle$  if all  $\Delta n_c$  qubits are also in the state  $|1\rangle$ . These increment operations are executed in parallel, applying all  $QFT_{\Delta n_c+1}(IQFT_{\Delta n_c+1})$  simultaneously and executing  $P_{+1,\Delta n_c+1}$  at a single time slice (note that we do not need to implement  $P_{\pm 1}$  separately due to the optimization described in Appendix B). Therefore, dividing control qubits into equal-sized groups is optimal, which was our initial assumption. These operations, which perform switching the state of ancilla qubit to  $|1\rangle$  if the state of  $\Delta n_c$  control qubits is  $|1\rangle$ , are grouped in a single block labeled  $ANC_{+1}$ , as displayed in Fig. 5.

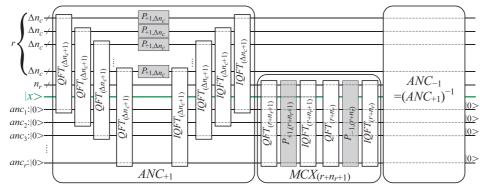


Fig. 5 A schematic diagram of QFT-based MCX circuit using r ancilla qubits. The left part of the circuit, labeled by  $ANC_{+1}$ , performs increment by one on sets of  $\Delta n_c$  controlled qubits and one ancilla qubit. In this step, the values of r registers each containing  $r\Delta n_c$  control qubits are incremented, while ancillae are used as the "carry out" qubits. All  $QFT_{(\Delta n_c+1)}$  are run in parallel, thus limiting the number of time slices to the value needed for executing one QFT. The same parallelization applies to IQFTs, while all phase gates can be executed in a single time slice. The central part of the circuit, denoted by  $MCX_{r+n_r+1}$ , performs X(NOT) operation to the target  $|x\rangle$  qubit controlled by the group of r ancilla qubits and  $n_r$  remnant control qubits. The right part of the circuit,  $ANC_{-1}$ , is the inverse of  $ANC_{+1}$ . It executes decrement by one to groups of joined  $\Delta n_c$  control and ancilla qubits, thus restoring ancillae to their initial state  $|0\rangle$ . In each subcircuit schematic block (QFT, IQFT, or P), dashed gray lines indicate which qubits are used.

Afterward, we execute the operation controlled by the ancilla qubits and the remaining control qubits to the most significant qubit  $|x\rangle$  (see Fig. 5). Note that this MCX operation is implemented in a way described in the explanation regarding Fig. 1. The group of subcircuits used for MCX is denoted by  $MCX_{(r+n_r+1)}$  in the middle section of Fig. 5. Finally, we apply the inverse of the  $ANC_{+1}$  operation (denoted by  $ANC_{-1}$ ), executing "decrement by one" to r batches of joined  $\Delta n_c$  controls with each ancilla. Thereby, ancillae are restored to their initial state  $|0\rangle$ . For successful "uncomputation" of ancilla qubits, the ordering of qubits in  $ANC_{+1}$  and  $ANC_{-1}$  must match.

When using auxiliary qubits, it is crucial to find an optimal implementation. It is easy to comprehend that multiple ancilla qubits allow independent parts of the algorithm to be executed simultaneously. In our case, parallelization is possible in  $ANC_{\pm 1}$ . Therefore, we divide the algorithm into two parts. One part is doing  $ANC_{+1}$  and  $ANC_{-1}$  with groups of control qubits that we can split among ancilla qubits and process them in parallel, while the other computes MCX operation  $(MCX_{r+n_r+1})$  on the remaining control qubits, the ancilla qubits, and one "MCX output" qubit (denoted by  $|x\rangle$  in Fig. 5). Each part has to execute pairs of QFT, IQFT, and P gates. The complexity of both parts is constrained by the QFT implementation.  $MCX_{r+n_r+1}$  has a lower complexity than a single cluster in  $ANC_{\pm 1}$  of the same size since it doesn't have to "uncompute" the most significant qubit.

First, we explore how the dividing of control qubits into clusters influences the circuit complexity. Increasing the number of control qubits in the cluster, the depth of the circuit will decrease. This is easy to follow since an increase of  $\Delta n_c$  by one means the algorithm parallelized the processing of additional r control qubits in  $ANC_{\pm 1}$ .

The depth of QFTs(IQFTs) in  $ANC_{\pm 1}$  will enlarge due to only one additional qubit in each cluster. At the same time, there will be r qubits less in MCX. Consequently, the depth of the circuit will linearly decrease with  $\Delta n_c$ . The minimum depth of the circuit is when the sum of  $ANC_{\pm 1}$  and  $MCX_{(r+n_r+1)}$  depths is minimal, which is for:

$$\Delta n_{c,\text{max}} = \left[\frac{n_c}{r}\right]. \tag{7}$$

This is considered the maximum number of control qubits in a cluster.

However, the number of elementary gates depends nonlinearly on  $\Delta n_c$ . We will find the optimal cluster size to reach the minimum number of elementary gates building the circuit. When using r ancilla qubits, we form r groups containing  $\Delta n_c$  control qubits plus one ancilla qubit to which we apply increment or decrement operations in parallel. Thereby, we are dealing with  $r\Delta n_c + r = n_c - n_r + r$  qubits in  $ANC_{\pm 1}$ . The remaining  $n_r$  control qubits, r ancilla qubits, and one the most significant qubits are used for executing  $MCX_{(r+n_r+1)}$ . It is straightforward to show that the minimum number of gates used is when MCX and a single cluster in  $ANC_{\pm 1}$  have approximately equal depths and numbers of elementary gates. To find the appropriate clusters size, which is  $(\Delta n_c + 1)$ , we have to divide  $((n_c - n_r + r) + (n_r + r + 1))$  into (r + 1) approximately equal-sized groups. For the number of ancilla qubits constrained to r, provided we have to use them all, the optimal value for  $\Delta n_c$  is:

$$\Delta n_{c,\text{opt}} = \left[ \frac{n_c + r}{r + 1} \right]. \tag{8}$$

An increase in the number of ancilla qubits permits higher parallelization that will reduce execution time. We will find the minimum number of ancilla qubits that will provide an optimal circuit. If the number of auxiliary qubits is not constrained, we can split all control qubits to approximately equal size clusters when  $n_c \approx r^2$ , thus  $r_{\rm opt} = [\sqrt{n_c}]$ . In this case,  $\Delta n_{c,\rm opt}$  and  $\Delta n_{c,\rm max}$  are equal or differ by at most one that follows from Eqs. (7) and (8). The ideal scenario is when the number of control qubits has a whole number as its square root and if we assume that the number of ancilla qubits used is equal to the square root of the number of control qubits. As a consequence,  $\Delta n_{c,\rm opt}(\sqrt{n_c}) = \Delta n_{c,\rm max}(\sqrt{n_c})$ .

For the number of auxiliary qubits above the optimal, control qubits are all divided into clusters where most count  $\Delta n_{c,\max}^{r>r_{\rm opt}}=\lceil n_c/r\rceil$  control qubits, while others have one less. One may note that clusters in  $ANC_{\pm 1}$  are equal in size only when  $n_c$  is divisible by r. The depth of  $ANC_{\pm 1}$  is determined by the size of a larger cluster  $\Delta n_{c,\max}^{r>r_{\rm opt}}$  that is decreasing stair-step function. On the other hand, the depth of the MCX increases linearly with r since we need to implement an  $MCX_{(r+1)}$  circuit. Therefore, the time complexity will increase linearly except if the increase of r by one results in the decrease of cluster size in  $ANC_{\pm 1}$  by one, when the depth does not change.

The number of elementary gates to implement the QFT(IQFT) in MCX is proportional to  $r^2$ , so the number of gates needed for MCX is  $O(r^2)$ . The QFT(IQFT) in  $ANC_{\pm 1}$  uses approximately  $O((\Delta n_{c,\max}^{r>r_{\text{opt}}})^2)$  gates. Considering r clusters in  $ANC_{\pm 1}$ ,

the number of elementary gates decreases with r as  $O(n_c^2/r)$ . Based on a simple analysis, we conclude that there is a relatively small variation in the number of elementary gates with r when using a large number of ancillas.

In various architectures used for quantum computing, ranging from the most efficient FC to the most restricted LNN, there is a similar change in circuits complexity with the number of ancilla qubits used. We will analyze only the most complex QFT-LNN-based implementation of the 101-qubit MCX. In Fig. 6(a) we show the dependence of the number of time slices  $(N_{t.s.})$  and elementary gates used  $(N_{e.g.})$  on the number of control qubits in a cluster  $(\Delta n_c)$  in the MCX that uses r=5 auxiliary qubits. Considering the number of control and ancilla qubits, a cluster can contain at most  $\Delta n_{c,\text{max}} = 20$  control qubits. One may notice a linear decrease in the number of time slices with  $\Delta n_c$ , where the minimum time complexity is reached for the maximum cluster size (see blue line in Fig. 6(a)). However, the dependence of  $N_{e.g.}$  on the  $\Delta n_c$  is nonlinear (see red line in Fig. 6(a)), as explained afore. The minimum number of gates used is for  $\Delta n_c = 17$ , which also follows from Eq. (8).

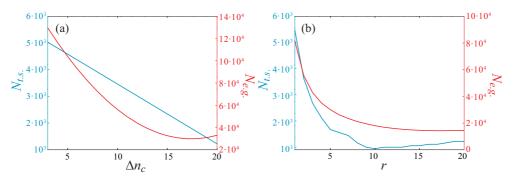


Fig. 6 The dependence of the number of time slices  $N_{t.s.}$  (left axis, blue line) and the number of elementary gates used  $N_{e.g.}$  (right axis, red line) on (a) the number of control qubits in a cluster  $\Delta n_c$  using five ancillas and (b) the number of auxiliary qubits r using the optimal cluster size given by Eq. (8), in the 101-qubit QFT-based MCX.

The dependence of time slices and elementary gates needed on the number of ancilla qubits ranging from 1 to 20 is shown in Fig. 6(b). Both  $N_{t.s.}$  and  $N_{e.g.}$  start decreasing as r increases, as indicated by the blue and red line in Fig. 6(b), respectively. For  $r = r_{opt} = \sqrt{n_c} = 10$ , there is a minimum of time slices while the slope of  $N_{e.g.}$  becomes very small. For  $r > r_{\rm opt}$ , the depth linearly increases with r except for values when cluster size in  $ANC_{\pm 1}$  decreases by one. This is manifested in narrow horizontal lines in  $N_{t.s.}$ , which is evident in Fig. 6(b). On the other hand, the number of elementary gates required slowly decreases as r increases above  $r_{\rm opt}$ . To conclude, an increase in r above the optimal value diminishingly influences a decrease in the number of elementary gates required at the price of increasing time slices needed for execution and using an unnecessarily large number of auxiliary qubits. Therefore, we consider  $r_{\rm opt}$  the best number of auxiliary qubits to use.

Since the QFT is an essential subcircuit used in our approach, one may wonder if there is another way to improve its efficiency other than AQFT. The state-of-the-art

optimization [26] shows that the number of CNOT gates used in the LNN QFT can be reduced to about 40% compared to previously known LNN QFT. Nevertheless, this approach increases circuit depth which cancels the benefit of a decrease in the number of elementary gates. If we can use an arbitrarily large number of ancilla qubits, we might consider parallelizing QFT, as discussed in Ref. [27]. This approach imposes the upper bound of  $O(\log_2 n)$  for the time complexity and  $O(n(\log_2 n)^2 \log_2 \log_2 n)$  on the circuit size in n-qubit QFT. Although this method significantly reduces circuit depths, it considerably increases the number of qubits used making it less scalable for implementation in current noisy intermediate-scale quantum (NISQ) devices. Therefore, we consider AQFT to be the optimal choice.

The possibility for simplifying three-qubit controlled gates, if arbitrary phase shifts of qubit states are permitted, was first elaborated in Ref. [13]. This approach is acceptable if the gate is part of an operation that straightforwardly maps some classical mathematical expression to reversible computation or if gates can be arranged to cancel out this extra phase. In general nonclassical unitary operation, this extra phase is "dangerous". Saeedi and Pedram first recognized the advantages of using a relative phase quantum circuit for implementing n-qubit Toffoli gates [14]. This approach is further elaborated in Ref. [16]. Several recent state-of-the-art implementations are based on optimizing this approach [15, 17, 28]. All these implementations result in depths of MC gates that are linear on the number of qubits and use a relatively small number of elementary gates. The QFT-based implementation also has a linear depth and more importantly, does not exhibit phase relativization.

One of the most efficient and systematic procedures to decompose multi-controlled unitary gates (the linear-depth decomposition - LDD) is presented in Ref. [15]. The authors calculated the lower bound for the number of time slots for execution to be (8n-20) in the FC architecture. It is one of the best implementations concerning the prediction of the MCX complexity. This is comparable to the number of time slots (8n-17) in our approach. To implement n-qubit MCX the authors used  $C-X^{1/2^{m-1}}=C-R_x(\pi/2^{m-1})$  gates as a part of the universal gate set. However, the decompositions of  $C-R_x(\pi/2^{m-1})$  in the NGS use 11 (for m=2) and 12 (for m>2) elementary gates (two C-X, four  $\sqrt{X}$  and the rest are  $R_z$ ), respectively. Both gates need 11 time sequences to execute. This is approximately twice what we have for  $C-R_m$ . We infer that our approach can outperform this state-of-the-art one.

The authors demonstrated the proof-of-principle using IBM's quantum cloud platform. For LDD-MCX implementation ibm\_hanoi, a 27-qubit Falcon r5.11 processor, was used. One should note that our implementation also uses the Falcon's family native gate set. They demonstrated that the quantum circuit depth of their MCX implementation increases much more slowly with the number of qubits than in other known methods. Up to the 6 qubits, LDD-MCX yields relatively deep circuits. For a number of qubits above 6, their implementation almost exponentially increases the advantage compared to the standard one [15].

We transpiled our circuit using the local simulator and compared circuit depths with state-of-the-art implementation publicly available on the GitHub page [29]. This comparison is displayed in Fig. 7. We used optimization level 3 in Qiskit's transpile function. In Ref. [15] optimization level 2 is used. A higher optimization level does

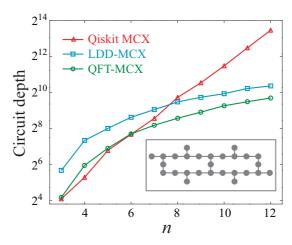


Fig. 7 The dependence of circuit depths on the number of qubits in default Qiskit ( $\triangle$ ), LDD-MCX from Ref. [15] ( $\square$ ), and QFT-MCX ( $\bigcirc$ ). A schematic view of qubits connectivity in the ibm\_hanoi quantum processor is shown in the insert.

not significantly affect the LDD-based and QFT-based implementations but results in a more optimal default Qiskit MCX circuit. As suggested above, our approach is approximately twice as efficient as the state-of-the-art one, which can be inferred by comparing the green ( $\bigcirc$ -markers) and blue ( $\square$ -markers) lines in Fig. 7. Up to the 6-qubit MCX, the default Qiskit implementation is comparable to the QFT-based, but outperforms the LDD circuits. However, the default Qiskit implementation shows an exponential ( $\sim 2^n$ ) increase in the circuit depth with the number of qubits (red line with  $\triangle$  markers in Fig. 7). For the optimization level used, when the number of qubits n > 7 in LDD-MCX or n > 6 in QFT-MCX, both become significantly less complex than the standard implementation. Since the LDD-MCX and QFT-MCX are linear depth circuits, they both gain an exponential advantage over the default MCX above this critical number of qubits. A more detailed analysis can show that the complexity of the QFT-based MCX gates is close to the upper bound. This is due to relatively sparse qubit connectivity in superconducting hardware. The schematic view of qubit connectivity in ibm\_hanoi device is displayed in the insert of Fig. 7.

Based on the LDD-MCX and QFT-MCX decompositions in the NGS, we concluded that the LDD-MCX uses twice the number of elementary gates as the QFT-MCX. These elementary gates have a high, but finite fidelity. Increasing the number of elementary gates used in a quantum computation increases an error in the computation process. Therefore, our circuit has a higher fidelity.

We should note that the LDD-MCX implementation in Ref. [15] does not use auxiliary qubits. This circuit can be simplified using ancilla qubits applying the method explained for the QFT-MCX. Since clusters in both implementations will be the same size (counting in a basis set with non-elementary gates), the advantage of our implementation will be preserved even when comparing the LDD-based and QFT-based implementations that use auxiliary qubits.

#### 4 Conclusions

In this paper, we proposed a new multi-controlled X (MCX) gate implementation based on the quantum Fourier transform (QFT). Circuit implementation that does not use auxiliary qubits is explained in detail. The most connected/restricted quantum computer architecture was analyzed to obtain the lower/upper bound for the time and space complexities. There is a linear increase in the number of time slices and a quadratic in the number of required elementary gates with the number of qubits in MCX. The number of gates significantly reduces when using the approximate implementation of the QFT, while there is no improvement in the time complexity of the circuit.

When using ancilla qubits, a circuit is divided into two functional parts. One performs in parallel increment/decrement by one on clusters composed of control qubits and one ancilla, while the other executes MCX operation on remnant control qubits, ancillas, and one output qubit. For a fixed number of auxiliary qubits used, the time complexity linearly decreases with the control qubits count in equal-size clusters. Constraining the number of required elementary gates to the minimum, the expression for the optimal cluster size was found. If the number of auxiliary qubits is not constrained, the optimal number that should be used is equal to the rounded square root value of the number of control qubits. Above this optimal number of ancillae, the time complexity of circuits increases whilst the number of gates insignificantly reduces.

Similar optimizations concerning the clustering of control qubits and selecting the proper number of ancilla qubits can be employed in other MCX implementations. Moreover, it was inferred that many exotic quantum gates can be implemented using quantum arithmetics. In cases where it is necessary to implement more complex arithmetic operations for a gate, the QFT-based approach can be of great advantage.

We compared our implementation with the state-of-the-art one. QFT-based MCX has a smaller complexity and higher fidelity than the most efficient existing linear depth MCX circuit.

**Acknowledgements.** This work was financially supported by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia under contract number: 451-03-65/2024-03/200103.

## Appendix A The unitary operator implemented using the QFT-based MCX

We will show that the unitary operator implemented using our approach is the same as one implemented by standard MCX. The circuit implements

$$MCX_n = (I_{2\times 2} \otimes (IQFT_{n_c}P_{+1,n_c}QFT_{n_c}))^{\dagger} \cdot IQFT_nP_{+1,n}QFT_n, \tag{A1}$$

where the number of qubits  $n = n_c + 1$ , and  $N = 2^n$ . Here

$$(IQFT_{n_c}P_{+1,n_c}QFT_{n_c})^{\dagger} = QFT_{n_c}^{\dagger}P_{+1,n_c}^{\dagger}IQFT_{n_c}^{\dagger} = IQFT_{n_c}P_{-1,n_c}QFT_{n_c}, \quad (A2)$$

where  $QFT_{n_c}^{\dagger} = IQFT_{n_c}$  ( $IQFT_{n_c}^{\dagger} = QFT_{n_c}$ ), and  $P_{+1,n_c}^{\dagger} = P_{-1,n_c}$ . At first, we manipulated the unitary matrix representation of the QFT, IQFT, and P for n=3 and n=4 and multiplied matrices according to Eq. A1 to show it correct in these cases. Using inductive reasoning, results were generalized for an arbitrary n. That was very tedious and uneasy to prove. While doing it, we noticed that the result for the  $IQFT_nP_{+1,n}QFT_n$  was evident:

$$IQFT_{n}P_{+1,n}QFT_{n} = \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & \dots & 0 & 0 & 0 \\ \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} N-1 & 1 & 1 & 1 & 1 & 1 \\ \hline 0 & \dots & \dots & \dots & 1 & 1 & 1 & 1 \\ \hline I_{(N-1)\times(N-1)} & 0 & 1 & 1 & 1 & 1 \\ \hline 0 & 0 & 0 & 0 & \dots & 0 & 1 & 1 \end{bmatrix},$$
 (A3)

since it performs N-element circular shifts to the left in the computational Z-basis. Similarly,

$$IQFT_{n_c}P_{-1,n_c}QFT_{n_c} = \begin{bmatrix} 0 & I_{\left(\frac{N}{2}-1\right) \times \left(\frac{N}{2}-1\right)} \\ \frac{\frac{N}{2}-1}{0 \cdot \cdot \cdot \cdot \cdot \cdot \cdot 0} \end{bmatrix}$$
(A4)

executes circular shifts to the right. Substituting Eqs. (A3) and (A4) into Eq. (A1), we obtain

Using the reversed qubits ordering, one may find familiar notation

$$MCX_n^{rev} = \left[ \frac{I_{(N-2)\times(N-2)} \mid 0_{(N-2)\times2}}{0_{2\times(N-2)} \mid X_{2\times2}} \right], \tag{A6}$$

where  $X_{2\times 2}$  is the Pauli-X matrix

$$X_{2\times 2} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \tag{A7}$$

The gates schematics, truth tables, and permutation matrices for different qubits ordering in the 3-qubit case, which is the Toffoli gate, are given in Fig. A1.

Based on quantum arithmetics, even one that is not the QFT-based, many complex gates can be very efficiently implemented.

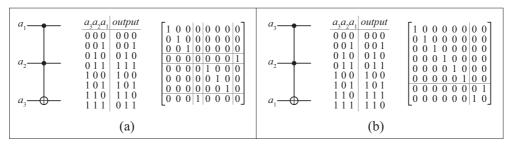


Fig. A1 The Toffoli gate, truth table, and permutation matrix when the target qubit is (a) the most significant and (b) the least significant.

#### Appendix B Optimization and decomposition

In MCX, there are phase  $P_{\pm 1}$  gates between QFT and IQFT. One may note that on the first wireline we have  $HR_1H = HZH = X$ . Using identities displayed in Figs. B.1(a) and (b) we conclude that we don't need to use any gates to implement  $P_{\pm 1}$ . Moreover, consecutive controlled phase gates ending/starting QFT/IQFT and phase gates in P will merge. As a result, we will have 4 fewer time slots each in the "+1" and "-1" blocks, as shown in Figs. B.1(c) and (d), respectively.

The native gate set (NGS) used by superconducting IMB's Falcon processor family is  $\{C-X,R_z,ID,SX=\sqrt{X},X\}$ . One may show that  $R_m=\exp(i\frac{\pi}{2^m})\cdot R_z(\frac{\pi}{2^{m-1}})$ ,  $H=\exp(i\frac{\pi}{4})\cdot R_z(\frac{\pi}{2})\sqrt{X}R_z(\frac{\pi}{2})$  and  $SWAP_{12}=(C_1-X_2)(C_2-X_1)(C_1-X_2)[18]$ . Due to optimization, we don't use single-qubit  $R_m$  gates. Hadamard and SWAP comprise three native gates and use three elementary time intervals to execute. However, we need three  $R_z$  and two C-X gates to implement  $C-R_m$  [13]. It is executed in 5 time sequences, as explicitly shown in Fig. B.2.

We will decompose our circuit into the NGS. This procedure is called *transpilation*. Using the transpiled MCX circuit one may find circuit depth (time for circuit execution) and the number of elementary gates used in genuine quantum computation. The circuit transpilation process involves some optimizations. By merging consecutive elementary gates of the same type or parallelizing the execution of ones acting on different qubits, we can reduce the experimental circuit's time and space complexity. A single-qubit gate can shift between time slices along the wireline up to the slice with different types of single-qubit gates or up to a C-X gate acting on that qubit. In each QFT there are  $C_j - R_{j',m} \cdot C_{j+1} - R_{j',m-1}$  and  $C_{j-1} - R_{j,2} \cdot H_j$  subcircuits. It is easy to notice that  $R_z(\pi/2^{m-1})$  on the  $j'^{\text{th}}$  wireline of  $C_{j+1} - R_{j',m-1}$  can be executed simultaneously with  $R_z(\pi/2^m)$  on the j<sup>th</sup> control line of  $C_j - R_{j',m}$  gate. The position in the time slice where  $R_z$ , located in the fifth time slice of the neighboring gate, can "shift" is indicated by a dotted line in Fig. B.2. Similarly,  $R_z(\pi/2)$  from  $H_i$  can be executed simultaneously with  $R_z(\pi/4)$  on the control line of  $C_{j-1} - R_{j,2}$  (moving to the position in  $C - R_m$  denoted by a dotted line in Fig. B.2). This rule mirrored applies to IQFT, too. Also, there are  $C_{j+1} - R_{j',m}^{\dagger} \cdot C_j - R_{j',m}$  gates between QFT and IQFT. Here, on the  $j'^{\text{th}}$  controlled line  $R_z(\pi/2^m)$  and  $R_z(\pi/2^m)^{\dagger} = R_z(-\pi/2^m)$ annihilate. Using simultaneous executions and gates annihilation, the effective depth of the  $C - R_m$  and H gates is reduced to 4 and 2, respectively.

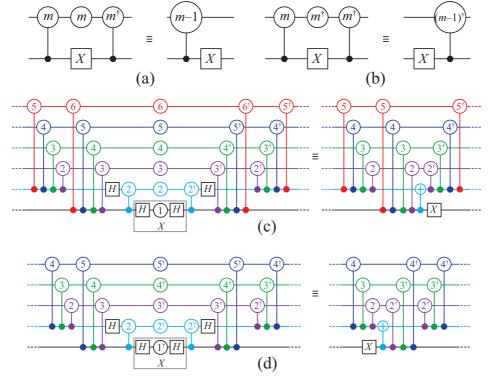


Fig. B.1 The identity used to simplify (a) "+1" and (b) "-1" circuits. Merging of phase gates and controlled phases for 6-qubit MCX in (c) "+1" and (d) "-1" circuits, respectively.

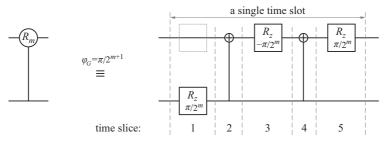


Fig. B.2 The decomposition of  $C-R_m$  gate up to the global phase  $\varphi_G$  in the IBM's Falcon family processor native gate set.

### References

- [1] Feynman, R.P.: Simulating physics with computers. Int. J. Theor. Phys.  $\bf 21(6)$ , 467-488 (1982) https://doi.org/10.1007/BF02650179
- [2] McArdle, S., Endo, S., Aspuru-Guzik, A., Benjamin, S.C., Yuan, X.: Quantum computational chemistry. Rev. Mod. Phys. **92**(1), 015003 1–51 (2020) https://doi.org/10.1103/RevModPhys.92.015003

- [3] Bauer, B., Bravyi,, S., Motta, M., Chan, G.K.-L: Quantum algorithms for quantum chemistry and quantum materials science. Chem. Rev. **120**(22), 12685–12717 (2020) https://doi.org/10.1021/acs.chemrev.9b00829
- [4] Ichikawa, T., Hakoshima, H., Inui, K., Ito, K., Matsuda, R., Mitarai, K., Miyamoto, K., Mizukami, W., Mori, Y., Nakano, Y., Nakayama, A., Okada, K.N., Sugimoto, T., Takahira, S., Takemori, N., Tsukano, S., Ueda, H., Watanabe, R., Yoshida, Y., Fujii, K.: A comprehensive survey on quantum computer usage: How many qubits are employed for what purposes? (2023) Preprint at https://doi.org/10.48550/arXiv.2307.16130
- [5] Schuld, M., Fingerhuth, M., Petruccione, F.: Implementing a distance-based classifier with a quantum interference circuit. EPL 119(6), 60002-1-6 (2017) https://doi.org/10.1209/0295-5075/119/60002
- [6] Rebentrost, P., Mohseni, M., Lloyd, S.: Quantum Support Vector Machine for Big Data Classification. Phys. Rev. Lett. 113(13), 130503-1-5 (2014) https://doi. org/10.1103/PhysRevLett.113.130503
- [7] Li, Y-.C., Zhou, R-.G., Xu, R-.Q., Luo, J., Hu, W-.W.: A quantum deep convolutional neural network for image recognition. Quantum Sci. Technol. 5(4), 44003-1-21 (2020) https://doi.org/10.1088/2058-9565/ab9f93
- [8] Orús, R., Mugel, S., Lizaso, E.: Quantum computing for finance: Overview and prospects. Rev. Phys. 4, 100028-1–12 (2019) https://doi.org/10.1016/j.revip. 2019.100028
- [9] Woerner, S., Egger, D.J.: Quantum risk analysis. Npj Quantum Inf. 5(1), 15-1-8
   (2019) https://doi.org/10.1038/s41534-019-0130-6
- [10] Stamatopoulos, N., Egger, D.J., Sun, Y., Zoufal, C., Iten, R., Shen, N., Woerner, S.: Option Pricing using Quantum Computers. Quantum 4, 291-1-20 (2020) https://doi.org/10.22331/q-2020-07-06-291
- [11] Martin, A., Candelas, B., Rodríguez-Rozas, Á., Martín-Guerrero, J., Chen, X., Lamata, L., Orús, R., Solano, E., Sanz, M.: Toward pricing financial derivatives with an IBM quantum computer. Phys. Rev. Res. **3**(1), 013167-1–12 (2021) https://doi.org/10.1103/PhysRevResearch.3.013167
- [12] Grover, L.K.: A fast quantum mechanical algorithm for database search. Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing STOC'96, Association for Computing Machinery, NewYork, NY, USA, 212–219 (1996) https://doi.org/10.1145/237814.237866
- [13] Barenco, A., Bennett, C.H., Cleve, R., DiVincenzo, D.P., Shor, P., Sleator, T., Smolin, J.A., Weinfurter, H.: Elementary gates for quantum computation. Phys. Rev. A 52(5), 3457–3467 (1995) https://doi.org/10.1103/PhysRevA.52.3457

- [14] Saeedi, M., Pedram, M.: Linear-depth quantum circuits for n-qubit Toffoli gates with no ancilla. Phys. Rev. A 87(6), 062318-1-5 (2013) https://doi.org/10.1103/PhysRevA.87.062318
- [15] Silva, A.J.da, Park, D.K.: Linear-depth quantum circuits for multiqubit controlled gates. Phys. Rev. A 106(4), 042602-1-6 (2022) https://doi.org/10.1103/PhysRevA.106.042602
- [16] Maslov, D.: Advantages of using relative-phase Toffoli gates with an application to multiple control Toffoli optimization. Phys. Rev. A 93(2), 022311-1-12 (2016) https://doi.org/10.1103/PhysRevA.93.022311
- [17] Balauca, S., Arusoaie, A.: Efficient Constructions for Simulating Multi Controlled Quantum Gates. In: Groen, D., de Mulatier, C., Paszynski, M., Krzhizhanovskaya, V.V., Dongarra, J.J., Sloot, P.M.A., (ed.) Computational Science – ICCS 2022, vol. 13353, pp. 179–194. Springer International Publishing, Cham (2022) ISBN 978-3-031-08759-2 https://doi.org/10.1007/978-3-031-08760-8\_16
- [18] Nielsen, M.C., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press, New York (2010) ISBN 978-1-107-00217-3
- [19] Barenco, A., Ekert, A., Suominen, K.-A., Törmä, P.: Approximate quantum Fourier transform and decoherence. Phys. Rev. A **54**(1), 139–146 (1996) https://doi.org/10.1103/PhysRevA.54.139
- [20] Draper, T.G.: Addition on a Quantum Computer. (2000) https://doi.org/10.48550/arXiv.quant-ph/0008033
- [21] Ruiz-Perez, L., Garcia-Escartin, J.C.: Quantum arithmetic with the Quantum Fourier Transform. Quantum Inf. Process. **16**(6), 152-1–14 (2017) https://doi.org/10.1007/s11128-017-1603-1
- [22] Yuan, Y., Wang, C., Wang, B., Chen, Z.-Y., Dou, M.-H., Wu, Y.-C., Guo, G.-P.: An improved QFT-based quantum comparator and extended modular arithmetic using one ancilla qubit. New J. Phys. **25**(10), 103011-1–12 (2023) https://doi.org/10.1088/1367-2630/acfd52
- [23] https://pypi.org/project/qiskit/
- [24] Fowler, A.G., Devitt, S.J., Hollenberg, L.C.L.: Implementation of Shor's algorithm on a linear nearest neighbour qubit array. Quantum Inf. Comput. 4(4), 237–251 (2004) https://doi.org/10.26421/QIC4.4-1
- [25] Maslov, D.: Linear depth stabilizer and quantum Fourier transformation circuits with no auxiliary qubits in finite-neighbor quantum architectures. Phys. Rev. A **76**(5), 052310-1–7 (2007) https://doi.org/10.1103/PhysRevA.76.052310

- [26] Park, B., Ahn, D.: Reducing CNOT count in quantum Fourier transform for the linear nearest-neighbor architecture. Sci. Rep. 13, 8638-1–10 (2023) https://doi.org/10.1038/s41598-023-35625-3
- [27] Cleve, R., Watrous, J.: Fast parallel circuits for the quantum Fourier transform. Proceedings 41st Annual Symposium on Foundations of Computer Science, Redondo Beach, CA, USA, 526–536 (2000) https://doi.org/10.1109/SFCS.2000.892140
- [28] Jun, Y-.M., Choi, I-.C.: Optimal Multi-Bit Toffoli Gate Synthesis. IEEE Access  $\bf 11$ , 27342–27351 (2023) https://doi.org/10.1109/ACCESS.2023.3243798
- [29] https://github.com/qclib/qclib