Quantum circuit design for computer-assisted Shor's algorithm

Chi-Chuan Hwang
Department of Engineering Science
National Cheng Kung University
Tainan City, 701, Taiwan
email: *chchwang@mail.ncku.edu.tw*

Chu-Yuan Tseng
Department of Engineering Science
National Cheng Kung University
Tainan City, 701, Taiwan
email: *jimmy0608861997jimmy@gmail.com*

Cheng-Fang Su
Department of Applied Mathematics
National Yang Ming Chiao Tung University
Hsinchu City, 30010, Taiwan
email: *scf1204@nycu.edu.tw*

September 11, 2021

Abstract

We successfully construct the quantum universal gate for Shors algorithm and derive the cost of this quantum circuit to estimate the complexity. In our circuit design, several modules are developed to perform integer operations such as addition and controlled addition on a quantum computer. These integer operations are achieved by using single-qubit logic gates and CNOT logic gates that are then combined into the quantum circuit for Shors algorithm. To reduce the number of qubits requires to decompose composite numbers, we adopt an adder using quantum Fourier transform and introduce a semi-classical quantum computer model to handle the multiplication operation. Using our circuit design to crack the widely used 1024-bit RSA encryption, both the space complexity and time complexity are 10^{14} approximately. In this case, the entire decomposition requires approximately $520,000$ qubits. Finally, we implement Shors factorization of the composite number 15 through IBM's platform. If the hardware can be improved in the future, the quantum circuit design proposed in this paper can be used to decompose larger composite numbers.

1 Introduction

Quantum information science (QIS) is an emerging field that seeks to store and process information with qubits. QIS is very different from classic information science (CIS), given the characteristics of the hardware of a quantum computer. Quantum parallelism[1] and quantum entanglement[2] are the two major characteristics related to qubits that allow QIS to have better performance in dealing with certain problems than traditional methods. For example, quantum entanglement can be adopted as the theoretical basis for applications such as quantum cryptography[3] and quantum key distribution[4][5]. Owing to these characteristics, a quantum computer can be hundreds of times more efficient than a traditional

computer with regard to certain problems and can even handle problems that cannot be solved with classical computer.

The origin of quantum computers can be traced back to 1980 when Paul Benioff described the first quantum mechanical model of a computer[6]. In his study, Benioff described the Schrödinger equation for the Turing machine and demonstrated that computers could operate under the laws of quantum mechanics[7], thereby laying an important foundation for quantum computing. In the same year, Tommaso Toffoli introduced the reversible Toffoli gate[8] that, together with the NOT and XOR gates, provides universal gates for classical reversible computing. Finally, in 1985, David Deutsch described the first universal quantum computer. Similar to a universal Turing machine capable to effectively simulate any other Turing machine, a universal quantum computer can also simulate any other quantum computer with polynomials.

In 1994, Peter Shor obtained a quantum algorithm for cracking RSA encryption to solve the following problem: Given a positive integer N, obtain its prime factors. This is the first specific algorithm for quantum computing[9]. Solving the prime factorization of large numbers is a difficult problem for traditional computers. Compared with classical methods, factoring large numbers is exponentially faster with this algorithm on an envisioned quantum computer. In theory, Shors algorithm can crack several cryptosystems that are currently in use. Therefore, this theory drew great attention from researchers worldwide on quantum computing, which has continued to be improved and refined by researchers to date. Shor proposed the first quantum error-correcting code in 1995[10]. In the following year, David P. DiVincenzo from IBM stated the minimum requirements for creating a quantum computer[11]. In 1998, the first nuclear magnetic resonance (NMR) computer with three qubits was developed. In the same year, the Grover algorithm was implemented for the first time on an NMR computer.

In 2001, Knill, Laflamme, and Milburn observed that optical quantum computing was possible using single-photon sources, linear optical elements, and single-photon detectors, thus opening a door for exploring linear optical quantum computing. In the same year, Raussendorf and Briegel proposed quantum computing based on measurement[12]. In 2004, the first functional NMR quantum computer in a pure state was demonstrated by the University of Oxford and the University of York. In 2006, Christoph Boehme from the University of Utah demonstrated that reading data stored as nuclear spins with a phosphorus-and-silicon quantum computer was feasible. Another research (2008) demonstrated that image storage might enable better quantum storage of qubits [13]. 1QB Information Technologies, Inc. (1QBit) is the worlds first dedicated quantum computing software company established in 2012. In 2015, the first quantum logic gate in silicon was successfully developed. In 2016, IBM launched "IBM Quantum Experience", which is an online interface for superconducting quantum computing and is used for fast releasing new protocols in quantum information processing. In 2017, IBM demonstrated a functional quantum computer that could handle 50 qubits and preserve the quantum state for 90 ms, extending the application of quantum computing in other fields. In 2012, a study published in Nature Photonics proposed a method[14] for reducing the number of qubits in Shors algorithm based on the characteristics of photons, and the number of qubits required by the register was reduced to a single qubit.

Shors algorithm is a classic and significant algorithm in quantum computing. As quantum computers and related technologies continue to develop, this algorithm will greatly impact long-standing cryptosystems. Moreover, as the first algorithm designed for quantum computing, it considers the characteristics of a quantum computer as part of its theoretical basis to fully exert the strengths of the quantum computer, providing inspiration and guidance for developing quantum algorithms in the future. In 2013, Gamel, Omar, and James[15] designed a quantum circuit for factoring composite numbers 15, 21 and 33. In the same year, Geller, Michael R., and Zhou[16] proposed quantum circuits for factoring composite numbers 51 and 85. Their main idea is first to determine the period based on the results of x and $a^x \bmod N$, establish a truth table of x and $a^x \bmod N$ and finally design a quantum circuit that decomposes the composite number N according to the truth table. Despite these advances, the above quantum circuits are all simplified circuits and therefore lack the quantum circuits for Shors algorithm that can actually be implemented.

This research aims to construct a quantum circuit that can actually be implemented for Shors algorithm, with the shortest possible circuit depth and the smallest possible number of qubits. In addition to the algorithm mentioned in the literature[17], this research adopted a traditional computer to help implement multiplication so that the quantum computer can implement Shors algorithm at a lower cost. To enable operations on a real quantum computer, circuit functions required for Shors algorithm were constructed step by step using universal logic gates, including quantum Fourier transform (QFT), inverse QFT, and adders.

This paper is organized into five sections. Section 1 introduces the development of quantum computers and related research on Shors algorithm and quantum computers, as well as the rationale and objectives of this research. Section 2 describes the procedure of RSA encryption and Shors algorithm, steps of adding a classical computer to the circuit, and differences between the fully quantum computing model and the semi-classical quantum computing model. Section 3 presents a detailed procedure for constructing various circuit modules for Shors algorithm and the advantages of using quantum adders based on QFT. Section 4 introduces how to use basic logic gates to construct the required modules, as well as how to implement the circuit for factoring composite number 15 via IBMs platform. Section 5 provides concluding statements.

2 RSA and Shors Algorithm

RSA encryption[18][19][20] is based on prime factorization. By providing a big composite number, hackers cannot crack within the effective time of the secret key, thus achieving secure encryption. To crack RSA encryption, the most critical step is to decompose N into the product of two prime factors, which is exactly what Shors algorithm seeks to achieve. Therefore, this section will introduce the principle and procedure of Shors algorithm.

2.1 Principle and procedure of Shors algorithm

When decomposing a composite number using Shors algorithm, the first step is to guess a value smaller than N and determine if a and N are coprime. When a is not relatively prime

to N, we use their greatest common factor to decompose N, which rarely happens. After obtaining a suitable a relatively prime to N and less than N, we then need to obtain the period of the function $a^x \bmod N$. This part is extremely crucial in Shors algorithm and is the only part that uses the quantum computer, described in detail in the next section. After obtaining r, if r is odd or $a^{r/2} \equiv 1(\bmod N)$, the value of a should be re-determined until it is satisfied that r is an even number and $a^{r/2} \neq 1(\bmod N)$. After obtaining a suitable r, we can then obtain p and q with the following two equations:

$$p = \gcd(a^{r/2} + 1, N)\,; \quad q = \gcd(a^{r/2} - 1, N)\,.$$

2.2 Quantum period-finding subroutine

The subroutine in Fig.1 can obtain the period of $a^x \bmod N$ with a quantum computer. The quantum circuits[21] for the period-finding subroutine contain 6 steps. The first step is to obtain the initial quantum state. The second step is to add the Hadamard logic gate to Register A to form a superposition of states. The third step is to calculate $f(x) = a^x \bmod N$. The fourth step is to measure register B, which causes Register A to collapse in a specific location; The fifth step is the inverse QFT, and the last step is quantum measurement. After the period-finding subroutine, an arithmetic periodic pulse will be obtained.

Among the above steps, the construction of U for calculating $a^x \bmod N$ is the most challenging. Its circuit implementation is also the most important part of this paper. The method for constructing U will be described in Section 3.

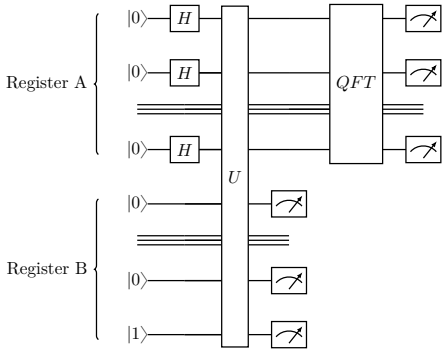


Figure 1: Quantum subroutine in Shors algorithm

2.3 Quantum circuit for Shors algorithm

To obtain U for Shors algorithm, the inputs and outputs should satisfy the setting in Fig.2.

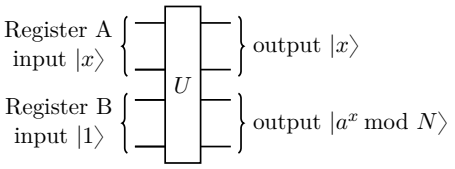


Figure 2: Black box diagram of Shors algorithm circuit

Let us first discuss the calculation of $a^x \bmod N$. It is decomposed as follows:

$$\begin{aligned} & a^x \bmod N \\ =& a^{(x_1 2^{n-1} + x_2 2^{n-2} + \cdots + x_{n-1} 2^1 + x_n 2^0)} \bmod N \\ =& (a^{2^0} \bmod N)^{x_n} \cdot (a^{2^1} \bmod N)^{x_{n-1}} \cdots (a^{2^{n-2}} \bmod N)^{x_2} \cdot (a^{2^{n-1}} \bmod N)^{x_1} \\ =& [a^{2^{n-1}} (a^{2^{n-2}} \cdots (a^{2^1} (a^{2^0} \bmod N)^{x_n} \bmod N)^{x_{n-1}} \cdots \quad \bmod N)^{x_2} \bmod N]^{x_1}, \end{aligned} \tag{1}$$

where $x_i = 0$ or $x_i = 1$. From the above equation, the circuit for Shors algorithm can then be structured as shown in Fig.3, where the inputs and outputs of U_k are as Fig.4. In Section 3, we will construct small modules based on the above diagram.

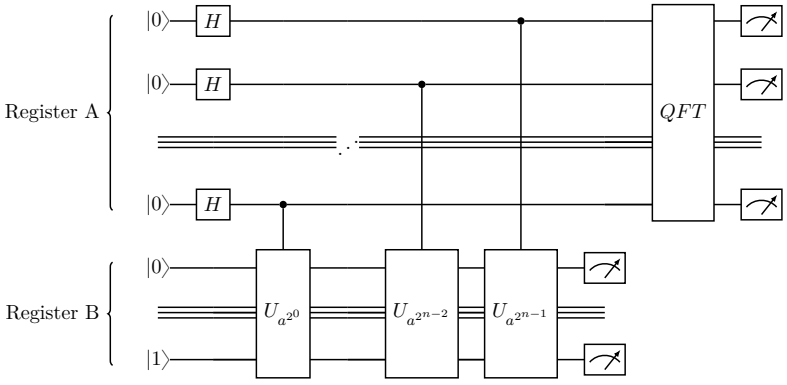


Figure 3: Diagram of the circuit for Shors algorithm

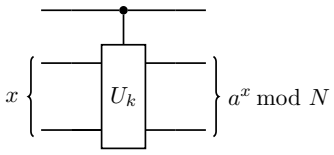


Figure 4: Diagram of the U module

2.4 Models for fully quantum computer and semi-classical quantum computer

According to (1), we should prepare the required qubits and logic gates for k_1 to k_{2n} in this model. Models for fully and semi-classical quantum computers are also compared in this subsection. First, the fully computer model is shown as Fig.5.

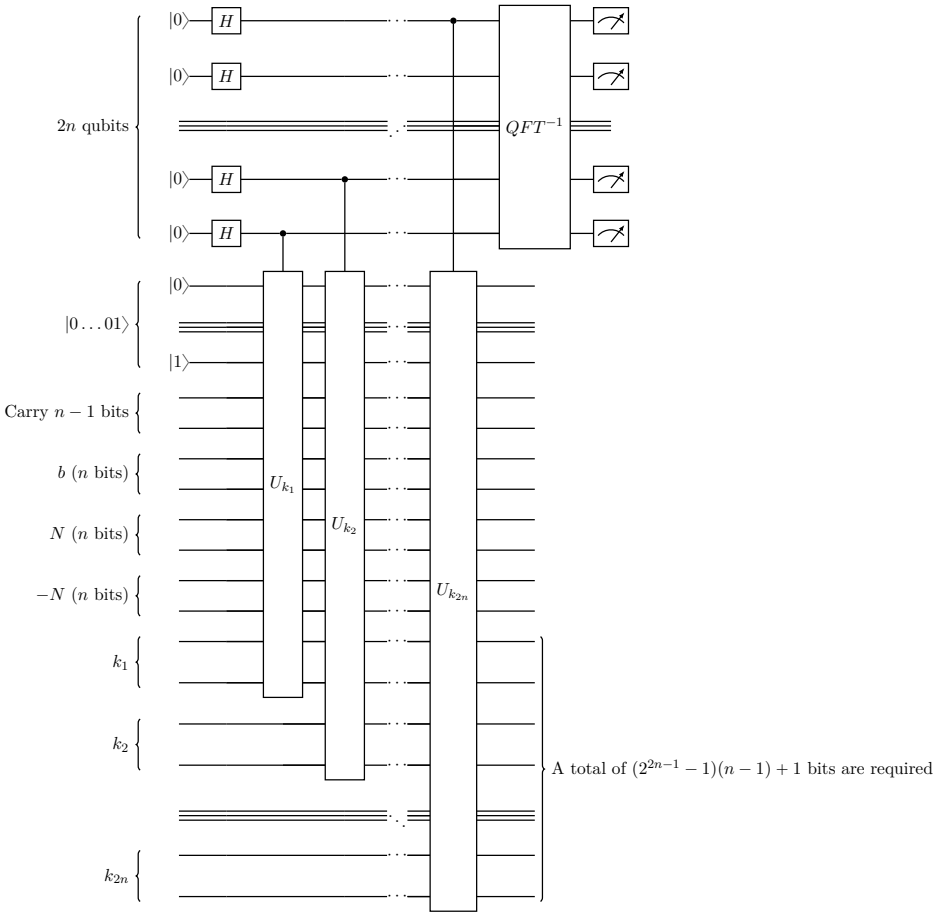


Figure 5: Fully quantum computers

Each multiplication requires $n-1$ bits and (1) has $2^{2n-1}-1$ multiplications. Therefore, based on the number of multiplications, the maximum number of bits required would be $(2^{2n-1}-1)(n-1)$. In addition, the modulo operation requires ± 1 bit. Altogether, we need to prepare a total of $(2^{2n-1}-1)(n-1)+1$ bits. As n increases, the number of bits required increases astronomically, which is why we use $(b+ax) \bmod N$. The model for the semi-classical quantum computing can be obtained from Fig.6. First, k_1 to k_{2n} are first computed using a classical computer, and the results are then used as inputs into the quantum circuit. To input k_1, we prepare n qubits. For k_2, we also prepare n qubits, and so on. The reason is that no matter how big a number is, it must be less than N after

mod N, so we don't need to prepare more bits than N. This is why we use traditional computers to perform multiplication. In other words, we can assume that x is the input to $U_{a^{2^i}}$. Therefore, we define $k_{2i} = a^{2^{2i-1}} \bmod N$. Subsequently, based on the following equation

$$\begin{aligned}(k_i x) \bmod N &= ((a^{2^i} \bmod N) \cdot x) \bmod N \\ &= ((a^{2^i} \bmod N) \cdot (x \bmod N)) \bmod N \\ &= (k_i \cdot (x \bmod N)) \mod N . \end{aligned} \tag{2}$$

We obtain

$$\begin{aligned}&[a^{2^{n-1}}(a^{2^{n-2}} \cdots (a^{2^1}(a^{2^0} \bmod N)^{c_{2n}} \bmod N)^{c_{2n-1}} \cdots \mod N)^{c_2} \bmod N]^{c_1} \\ =&[k_{2n} \cdots (k_2 \cdot (k_1 \bmod)^{c_{2n}} \bmod)^{c_{2n-1}} \cdots \bmod N]^{c_1}\end{aligned} \tag{3}$$

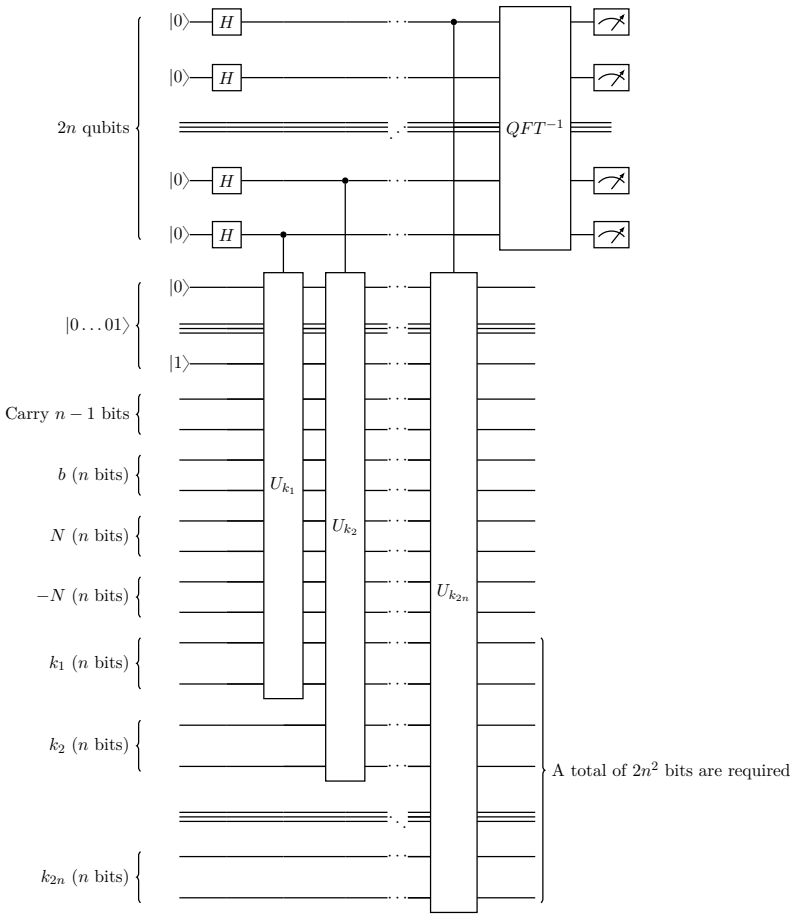


Figure 6: Semi-classical quantum computers

3 Circuit modules for Shors algorithm

This section introduces how to construct U and the circuit for Shors algorithm based on quantum Fourier transform (QFT).

3.1 Quantum Fourier Transformation

QFT[23][24] is a linear transformation for quantum states, widely used in quantum computers and various quantum algorithms. It maps one quantum state to another as

$$QFT\,|j\rangle = \frac{1}{\sqrt{N}} \sum_{0}^{N-1} \omega_N^{k_j}\,|k\rangle\ .$$

Let $R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{bmatrix}$. According to the above equation, we can draw the diagram of the QFT-based circuit (see Fig.7), as well as the inverse QFT-based circuit (see Fig.8).

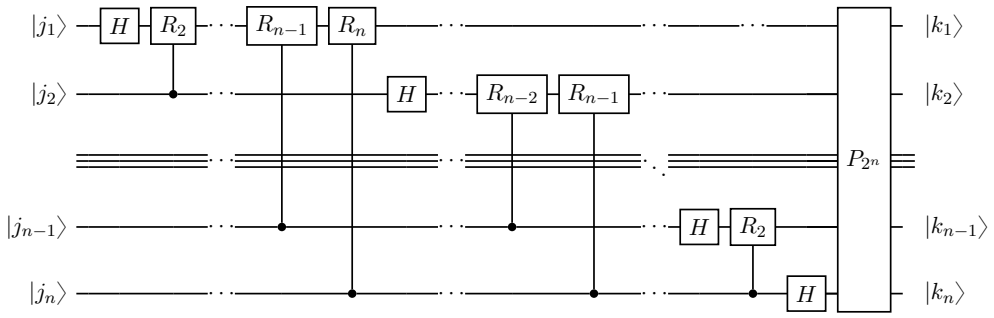


Figure 7: Circuit for n-qubit QFT

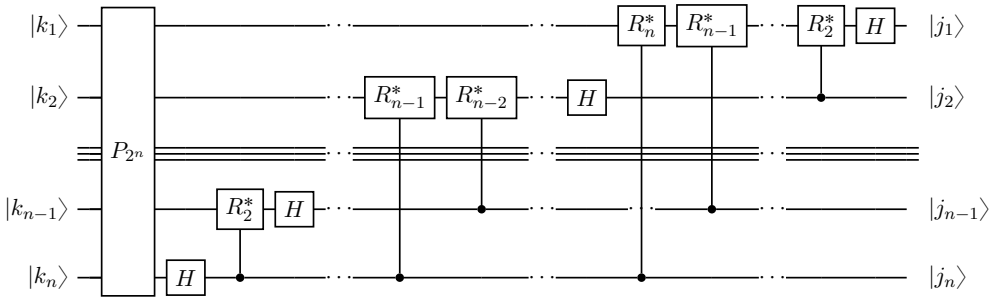


Figure 8: Circuit for n-qubit inverse QFT

Next, we construct a circuit for 5-qubit QFT and inverse QFT on the IBMs platform. Considering that CNT-R_n can be decomposed into universal logic gates[25], modules are constructed for subsequent use, as shown in Fig.9.

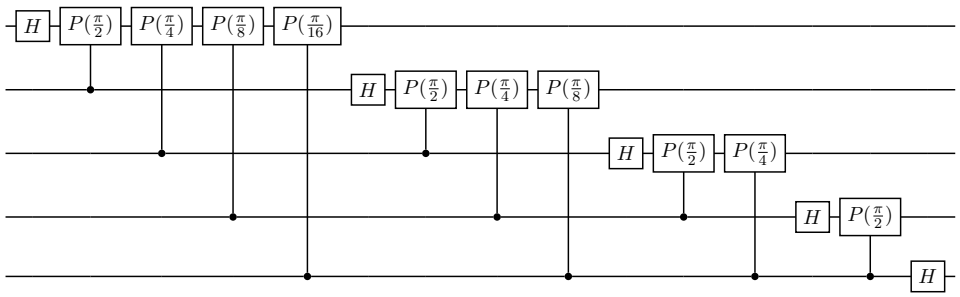


Figure 9: Circuit for 5-qubit QFT on IBMs platform

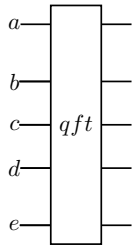


Figure 10: Circuit module for 5-qubit QFT on IBMs platform

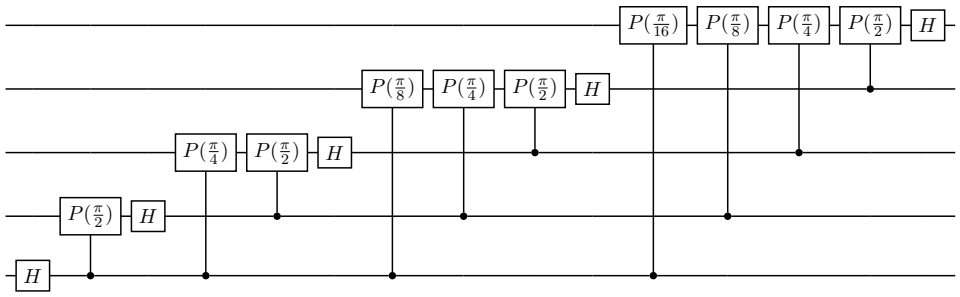


Figure 11: Circuit for 5-qubit inverse QFT on IBM's platform

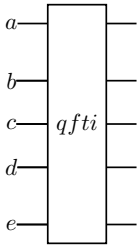


Figure 12: Circuit module for 5-qubit inverse QFT on IBM's platform

3.2 Adder

Following QFT, this subsection describes how to construct a quantum adder[26]. For a classical computer, the adder has three inputs (A, B, C_{in}) and two outputs (S, C_{out}), suggesting that the transformation is irreversible. In contrast, reversible adders are essential in quantum computing systems. To this end, a unitary transformation is designed to obtain both S and C_{out} simultaneously through one operation with the input quantum states, as Fig.13.

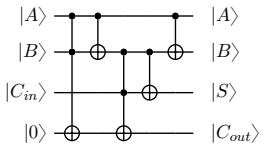


Figure 13: Quantum full adder

The relation between the inputs and outputs in the above figure is expressed as

$$S = A \oplus B \oplus C_{in}\,; \quad C_{out} = AB \oplus BC_{in} \oplus AC_{in}\,.$$

However, the above-mentioned quantum full adder is only the most basic quantum full adder and can be further optimized. Many people are interested in creating new adders or perfecting existing adder designs. Some studies minimize the necessary cost, and some focus on optimizing the time required for the overall calculation.

Our goal is to optimize the basic quantum full adder circuit, hoping to use fewer logic gates to achieve the same result. To achieve the same result with fewer logic gates, a new adder needs to be constructed using QFT. This QFT-based adder can convert the quantum state through operations and obtain the inverse QFT results. The QFT-based quantum addition requires fewer qubits than the full adder, though its logic gate design is more complicated.

To reduce the cost of Shors algorithm, IBM's platform is used in this subsection to test the validity of the quantum adder, QFT, and inverse QFT. Related modules are also constructed to be used for factorization in Section 4.

The diagram of the quantum adder based on QFT is shown in Fig.14.

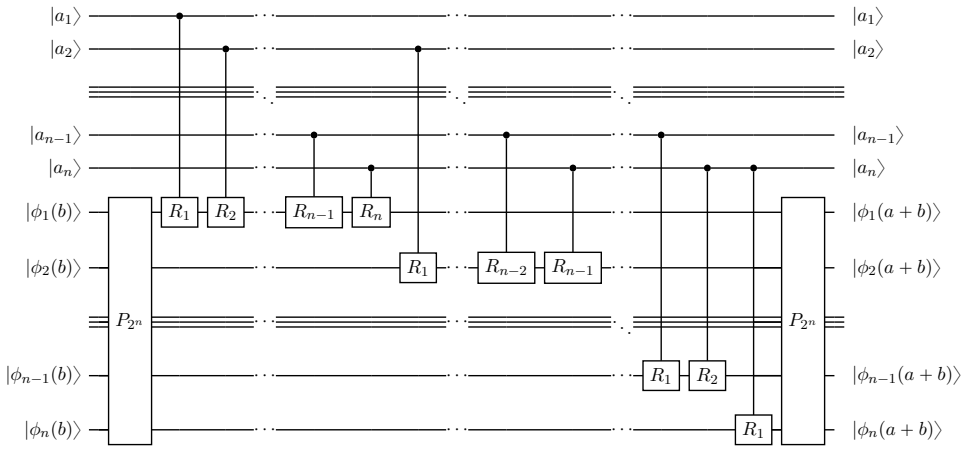


Figure 14: n-qubit quantum adder

Then, a 5-bit quantum adder is constructed on the IBM's platform, and a module is built for subsequent use, as shown in Fig.15 and Fig.16.

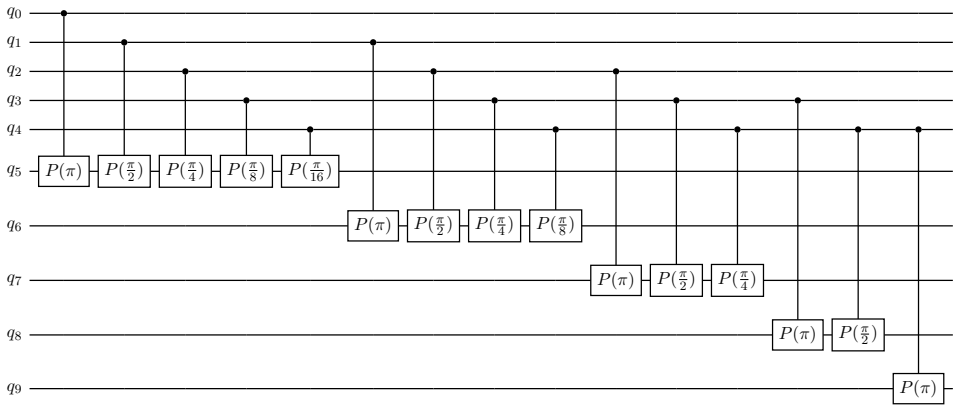


Figure 15: 5-bit quantum adder on IBM's platform

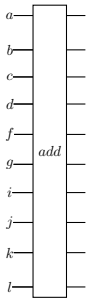


Figure 16: Circuit module for 5-qubit quantum adder on IBM's platform

3.3 Control-Adder

The next operation that needs to be included is controlled addition. In addition to the qubits for the operation of addition, another control qubit is used to determine whether the addition operation is applied. If the control qubit is $|0\rangle$, the addition operation is applied. If the control qubit is $|1\rangle$, the addition operation is not applied. We first add a control point to each logic gate when constructing the control adder, as shown in Fig.17.

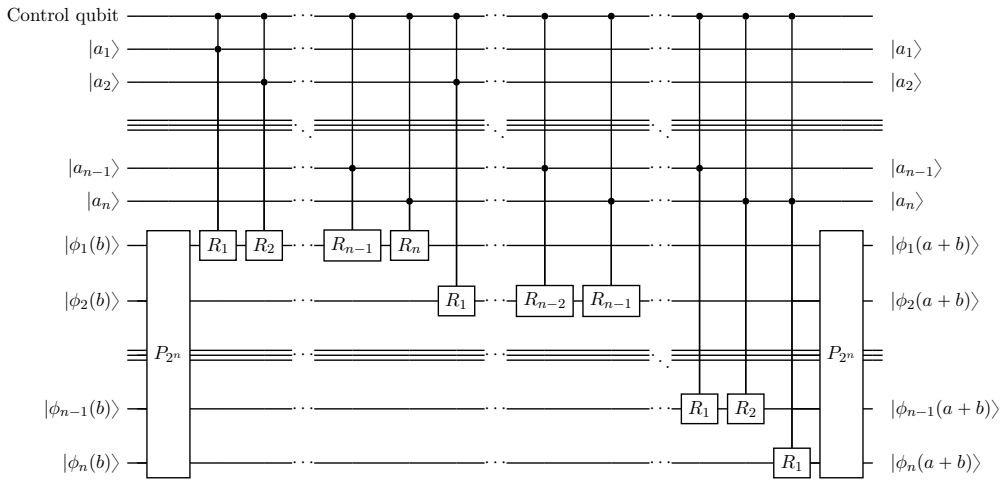


Figure 17: Diagram of an n-qubit control-adder

The question then arises as to how we could simplify CNT-CNT-R so that it can be input into IBM's platform. To solve this question, we adopted the following method for simplification[27].

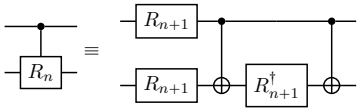


Figure 18: Decomposition of CNT-R

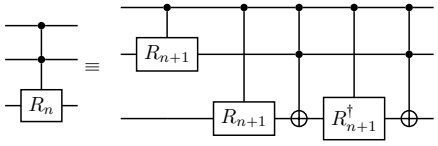


Figure 19: Decomposition of CNT-CNT-R

The construction of a 5-qubit control-adder requires CNT-CNT-R_n, where n is 15. The Toffoli gate can be used for decomposition[28], and the following modules can be pre-built

on IBM's platform.

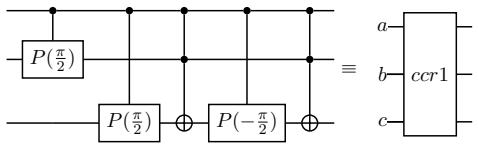


Figure 20: CNT-CNT-R_1 module on IBM's platform

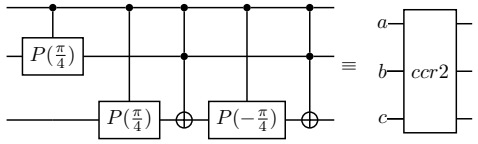


Figure 21: CNT-CNT-R_2 module on IBM's platform

We can then use the above modules to construct the control-adder, as shown in the figure below.

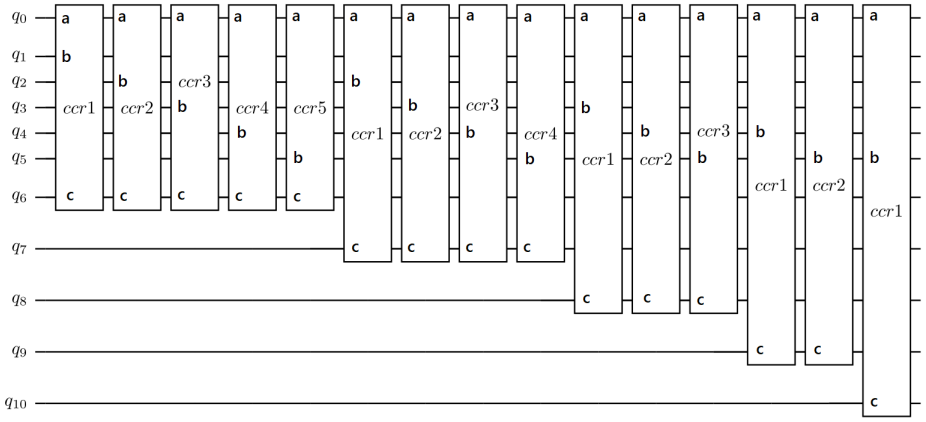


Figure 22: Control-adder module on IBM's platform

3.4 Control-Control-Adder

As described in the previous subsection, an additional control point is added to each logic gate for the addition operation. Similarly, the control-control-adder adds additional control points to each logic gate in the controlled addition operation. In this sense, we then require CNT-CNT-CNT-R_n rather than CNT-CNT-R_n, as shown in Fig.23.

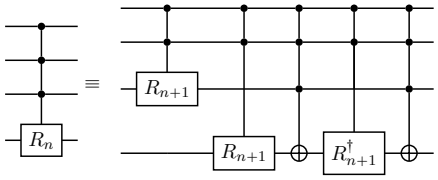


Figure 23: Decomposition of CNT-CNT-CNT-R_n

Specifically, CNT-CNT-R_n is decomposed by using the same method as for controlled addition. The controlled NOT (CNOT) gates at multiple control points are referred to as the multi-qubit CNOT gate, which can be decomposed in the following manner[29][30].

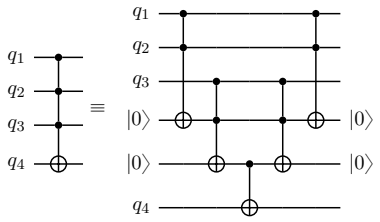


Figure 24: Decomposition of CNT-CNT-CNT-NOT

The above simplification requires an extra two-qubit register. Therefore, a total of 14 qubits are used to construct the control-control-adder, including 10 qubits for the addition operation, 2 control qubits, and 2 qubits for the register. The multi-qubit CNOT module is established as Fig.25.

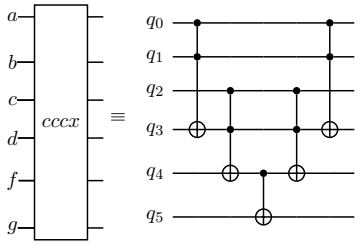


Figure 25: CNT-CNT-CNT-NOT module on IBM's platform

CNT-CNT-CNT-R_1 to CNT-CNT-CNT-R_5 are required for the 5-qubit control-control-adder, so we should first construct CNT-CNT-R_2 to CNT-CNT-R_6, as well as CNT-CNT-$R_2^\dagger$ to CNT-CNT-$R_6^\dagger$. The method for constructing CNT-CNT-R_n has been mentioned in the previous subsection, and the method for CNT-CNT-$R_n^\dagger$ is shown as Fig.26.

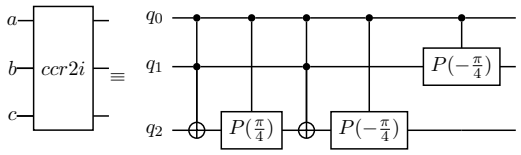


Figure 26: Construction of CNT-CNT-$R_2^\dagger$ module on IBM's platform

The constructed multi-qubit CNOT module, CNT-CNT-R, and its conjugate transpose are used to form the CNT-CNT-CNT-R module, which is used to build the control-control-adder.

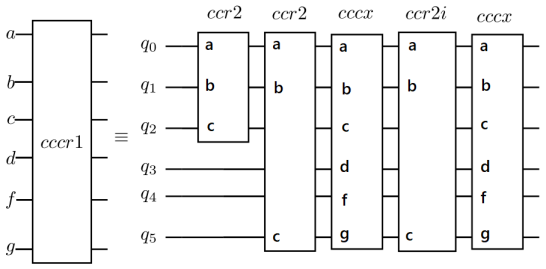


Figure 27: Construction of CNT-CNT-CNT-R_1 module

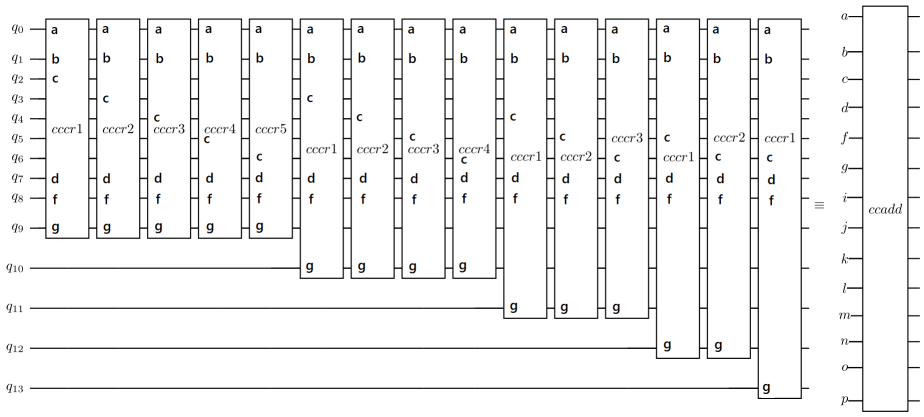


Figure 28: Construction of CNT-CNT-ADD on IBM's platform

3.5 CNT-CNT-$(b+a)$ mod N

The next step is to establish CNT-CNT-$(b+a)$ mod N. To be more specific, when two control qubits are $|1\rangle$, the operation of $(b+a)$ mod N is applied on the input a, b and N

must satisfy: the following condition

$$0 \leqslant a, b < N\,.$$

This operation will be realized by using the aforementioned modules, and its structure is shown as Fig.29.

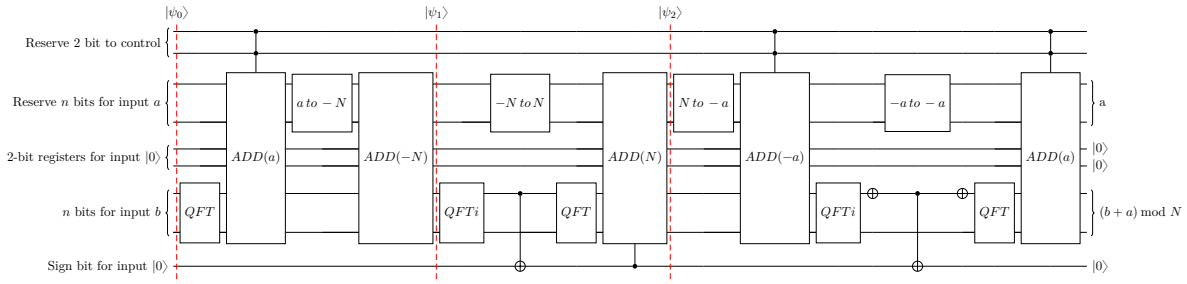


Figure 29: Schematic of the n-qubit CNT-CNT-$(b + a)$ mod N

Specifically, there are many conversion modules in the phase of the input qubits. As the value of Register A does not change after the addition operation is completed, we can change the value in the phase of the input qubits to reuse a when performing multiple additions continuously. To explain the operation of the entire circuit, let us assume that the two control qubits are both $|1\rangle$. The initial state of an input is

$$|\psi_0\rangle = |11\rangle \otimes |a\rangle \otimes |00\rangle \otimes |b\rangle \otimes |0\rangle\,.$$

The next step is to apply the operation of $(-a + N)$ to the area of operations, i.e., the location of $|b\rangle$, with the following equation:

$$|\psi_1\rangle = |11\rangle \otimes |-N\rangle \otimes |00\rangle \otimes |b + a - N\rangle \otimes |0\rangle\,.$$

We would then have either $b + a - N < 0$ or $b + a - N \geqslant 0$. If $b + a - N < 0$, when the result is obtained from inverse QFT, the highest qubit in the phase of operations will be $|1\rangle$, and then the label qubit will be changed to $|1\rangle$ through CNT-NOT. The label qubit can be then used to control whether to apply the operation of $(+N)$ to make the value in the phase of operations return to a positive value. On the contrary, if $b + a - N \geqslant 0$, no action will be taken. Its state is expressed as follows:

$$\begin{aligned}
|\psi_2\rangle &= |11\rangle \otimes |N\rangle \otimes |00\rangle \otimes |b + a - N\rangle \otimes |0\rangle\,, \quad \text{if } b + a - N < 0\,;\\
|\psi_2\rangle &= |11\rangle \otimes |N\rangle \otimes |00\rangle \otimes |b + a - N + N\rangle \otimes |1\rangle\,, \quad \text{if } b + a - N \geqslant 0\,.
\end{aligned}$$

3.6 CNT-$(b + ax)$ mod N

The next module to construct is CNT-$(b + ax)$ mod N, and the same requirements are predetermined as for CNT-CNT-$(b + a)$ mod N:

$$0 \leqslant a, b < N\,.$$

Furthermore, x should meet the following condition as a and b:

$$0 \leqslant x < N\,.$$

First, $(b + ax) \bmod N$ can be organized as

$$x = x_1 2^{n-1} + x_2 2^{n-2} \cdots + x_{n-1} 2^1 + x_n 2^0\,,$$

where $x_1, x_2, \ldots, x_{n-1}, x_n \in 0$ or 1. Substituting the above into $(b + ax) \bmod N$, we have

$$\begin{aligned}
&(b + ax_1 2^{n-1} + ax_2 2^{n-2} \cdots + ax_{n-1} 2^1 + ax_n 2^0) \bmod N \\
=&((\cdots((b + ax_n 2^0) \bmod N + ax_{n-1} 2^1) \bmod N + \cdots ax_2 2^{n-2}) \bmod N + ax_1 2^{n-1}) \bmod N \\
=&[\cdots((((b + ax_n 2^0) \bmod N) \bmod N)^{x_n} + a2^1 \bmod N)^{x_{n-1}} + \cdots a2^{n-1} \bmod N]^{x_1}\,.
\end{aligned}$$

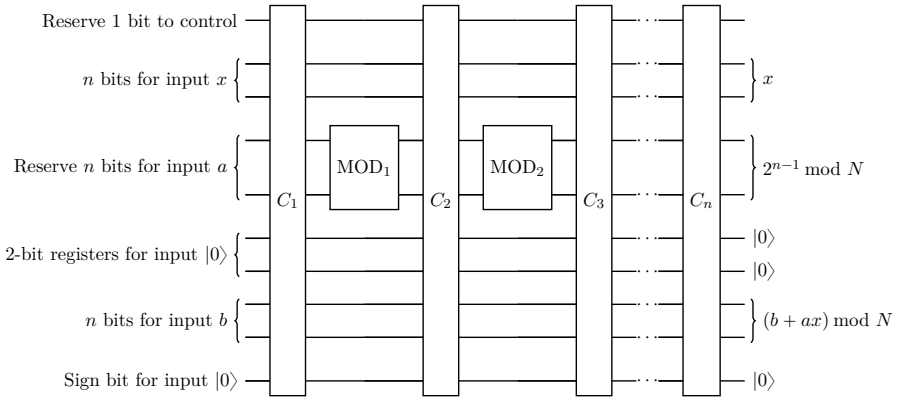


Figure 30: Diagram of n-qubit circuit

The circuit design related to the multiplication operation here uses the semi-classical quantum computer model mentioned in section 2.4. After comparing the time complexity of the semi-classical quantum model and the fully quantum computer model, we can get the results in the figure below. Thus, traditional computers for multiplication operations can significantly reduce the number of qubits required compared to the fully quantum computer model. To crack the widely used 1024-bit RSA encryption takes about 30 seconds to use the ideal Shor's algorithm, about 300 years to use the best classical (GNFS) algorithm, and about 50 minutes to use the semi-classical algorithm proposed in this paper.

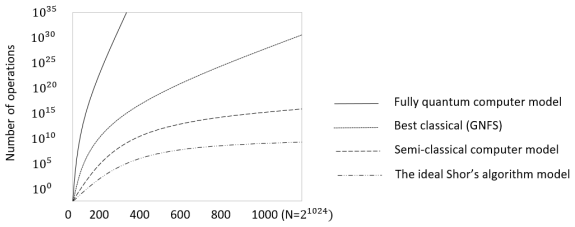


Figure 31: Comparison of different models

4 Prime factorization in Qiskit

In this section, we use the circuit modules in section 3 to compose the circuit of Shor's algorithm and actually decompose the composite number 15 on the IBM's platform.

4.1 Construction of the circuit

As previously mentioned, the modules required for Shors algorithm must have the input x and the output $(kx) \bmod N$. While this is very similar to $(b + ax) \bmod N$ (if $b = 0$) that we constructed in previous section, the biggest difference between the two is the location of the output. Here, the input x and the output $(kx) \bmod N$ must be in the same register. Therefore, we introduced CNT-SWAP after CNT-$(b + ax) \bmod N$ to have the input and output in the same place as Fig.32.

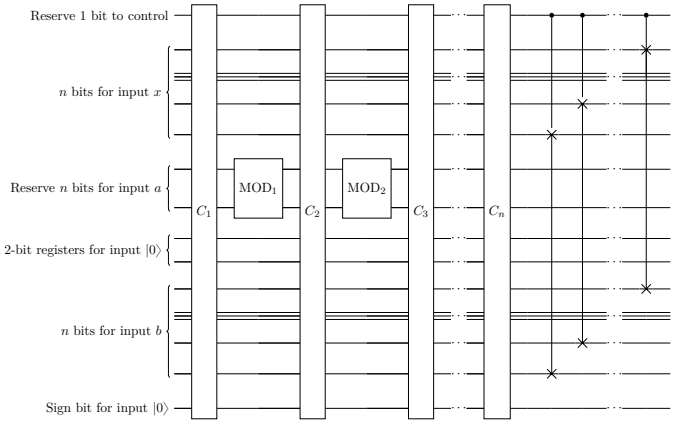


Figure 32: CNT-$(b + ax) \bmod N$ with CNT-SWAP

Another issue that needs to be addressed is that the value in the phase of operations is

no longer $|0\rangle$ when the next operation is executed. In that case, the register must be zero so that there will be no error in the next calculation. We also reserved n-qubit $|0\rangle$ that was initialized to the zero-position by using CNT-SWAP, as shown in Fig.33.

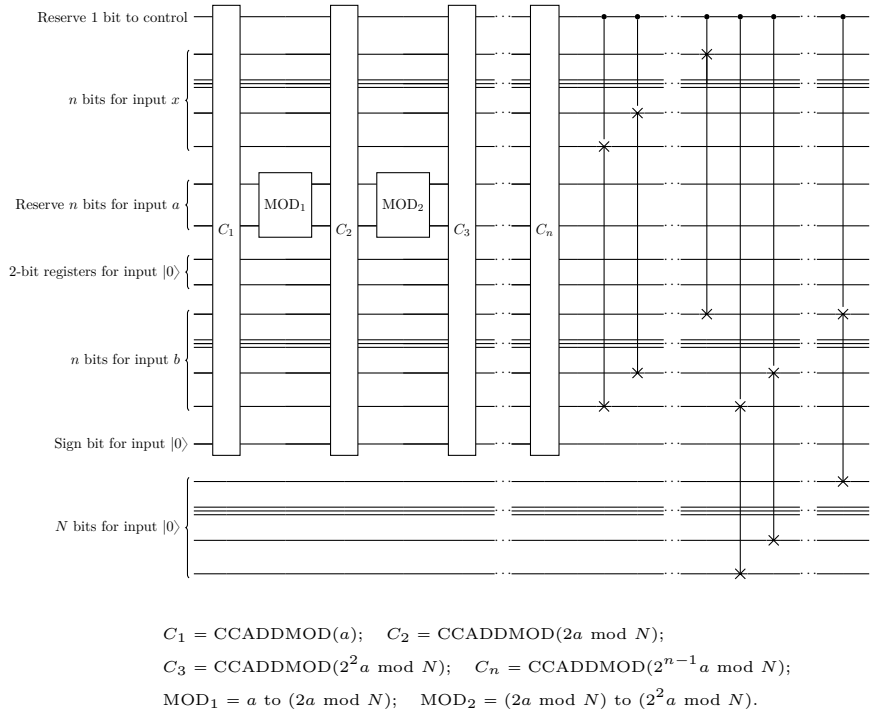


Figure 33: Diagram of the CNT-U module

The U module mentioned in Section 3 has now been constructed. The following subsection will implement the designed circuit for factoring the composite number 15 on the IBM's platform.

4.2 Quantum circuit ($N = 15$, $a = 4$)

A circuit with $N = 15$ and $a = 4$ was constructed on IBM's platform as Fig.34.

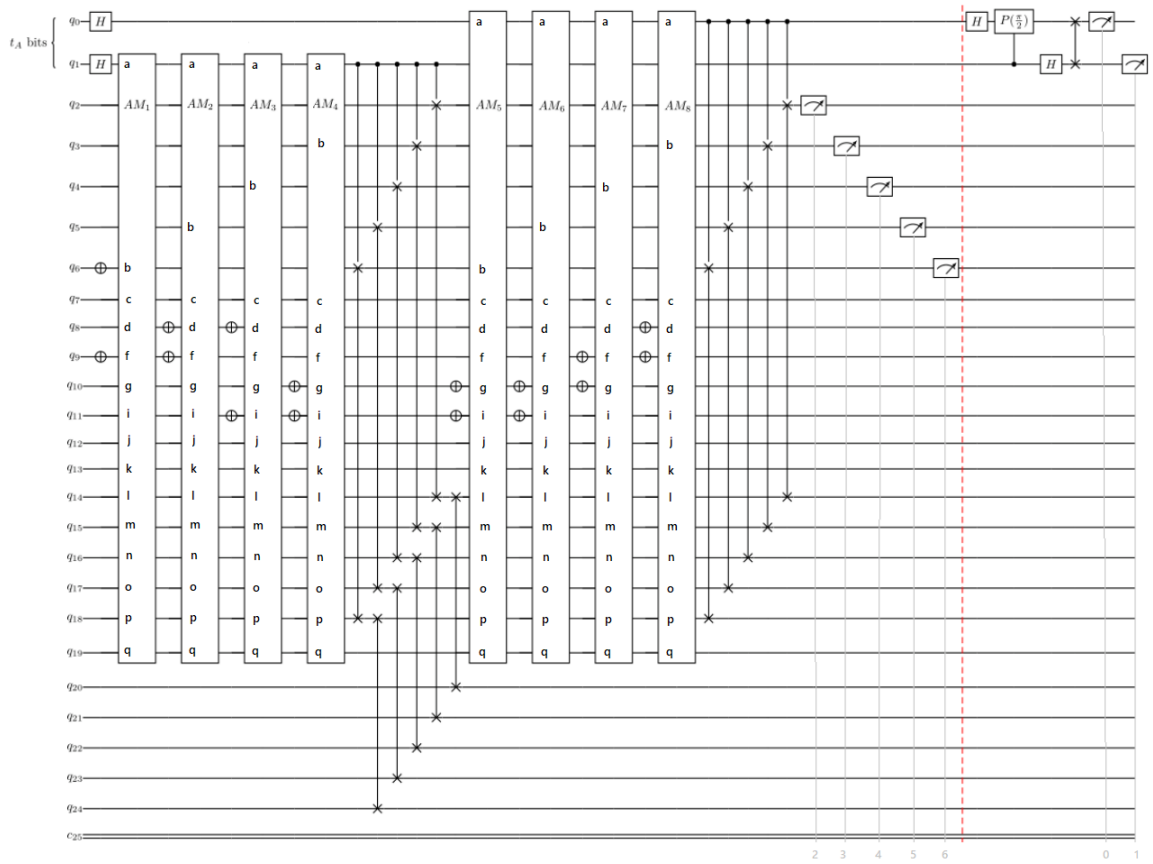
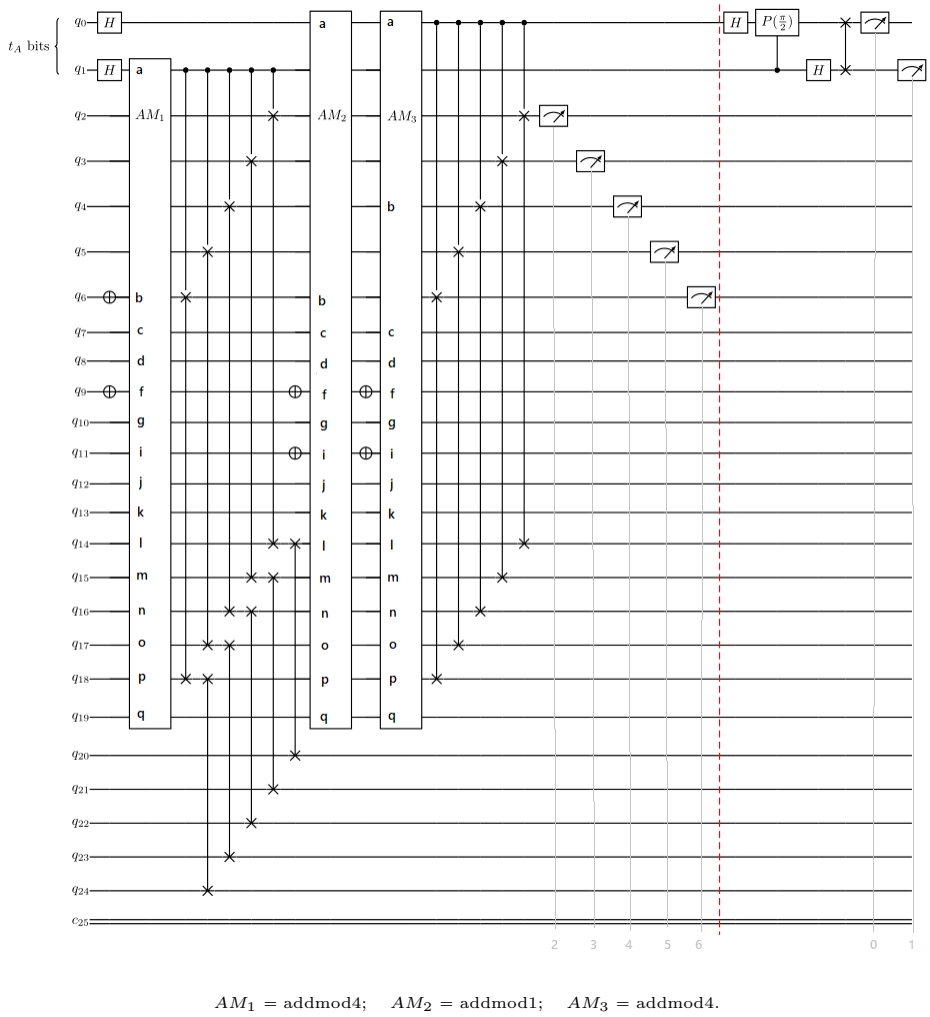


Figure 34: Quantum circuit ($N = 15$, $a = 4$)

As the initial input value of the first module was $|00001\rangle$, the modules whose control points passed through the first to fourth qubits can be omitted. For the second module, the only possible input value was 1 or 4, $|00001\rangle$ or $|00100\rangle$, so modules with control points in the first, second, and fourth modules can be omitted. After omitting the above modules, the circuit diagram can be simplified as Fig.35.



AM_1 = addmod4; AM_2 = addmod1; AM_3 = addmod4.

Figure 35: Diagram of the simplified circuit ($N = 15$, $a = 4$)

The corresponding probabilities of register A in the $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$ states were calculated to be 26.66%, 22.56%, 26.46%, and 24.32%, respectively. Because of the noise on the hardware, it is necessary to square the amplitude before normalizing. The wavefunction was normalized to observe the peaks better. Subsequently, the obtained frequencies for the four states were 33.01%, 13.32%, 31.79%, and 20.85%, respectively, where the peaks were observed at $|00\rangle$ and $|10\rangle$. The use of regularization can make the distribution of probability more obvious. However, the error generated here is related to the hardware. In the case of many logic gates and a deep circuit depth, the hardware error will increase. In addition, many teams study noise reduction[31][32][33], so this is an important research direction in the future.

The obtained peak values $|00\rangle$ and $|10\rangle$ were converted into binary (i.e., 0 and 2). Sub-

stituting the binary values into the following equation, we have:

$$\frac{c_i}{2^{t_A}} = \frac{0}{4}, \frac{2}{4} = \frac{0}{2}, \frac{1}{2},$$

where c_i is the measured value; t_A is the number of qubits in Register A. The above equation shows that the period r is 2. Substituting it into the following equation, we obtain

$$\begin{aligned}\gcd(a^{\frac{r}{2}} - 1, 15) &= \gcd(3, 15) = 3\,;\\ \gcd(a^{\frac{r}{2}} + 1, 15) &= \gcd(5, 15) = 5\,.\end{aligned}$$

The above results show that we have successfully implemented the complete circuit for Shors algorithm and factored the composite number 15 ($15 = 3 \times 5$) on IBM's platform. The following equation can be used to estimate the complexity of decomposition of the composite number with n qubits, where k denotes the number of qubits in Register A.

$$\begin{aligned}&C^p\,(\text{circuit for n-qubit Shors algorithm})\\ =&k \cdot C^p(\text{CNT} - (b + ax) \mod N) + k \cdot n \cdot C^p(\text{CNT-SWAP}\\ &+ (k-1) \cdot n \cdot (\text{SWAP}) + (\text{QFT}))\\ =&k \cdot (425.5n^3 + 406.5n^2 + 3n) + k \cdot n \cdot 17 + (k-1) \cdot n \cdot 3 + 5 \cdot \frac{n^2 - n}{2} + n\\ =&k \cdot (425.5n^3 + 406.5n^2 + 23n) + 2.5n^2 - 4.5n\,.\end{aligned}$$

The total number of qubits required by the whole circuit is $2n + k(n+1)$, and the value of k is normally between $0.5n$ and $2n$. The commonly used RSA encryption usually uses 1024-bit keys, i.e., $n = 1024$. In this case, with $k = 0.5n$ and $n = 1024$, the space and time complexity of the algorithm are both approximately 10^{14}, and it would require $520,000$ qubits for factorization.

4.3 Comparison of cost between classical and quantum computing

For factoring an n-qubit number, we need to calculate

$$a^{2^0} \bmod N,\ a^{2^1} \bmod N,\ \cdots,\ a^{2^{\frac{n}{2}-1}} \bmod N,$$

a total of $\frac{n}{2}$ calculations. When using a classical computer, we theoretically need to calculate $2^{\frac{n}{2}}$ times. This shows that the computation time to crack RSA encryption can be significantly reduced with a quantum computer. The difference between classical and quantum computing is noticeable if $n = 1024$, which is the length of an RSA key.

5 Conclusion

Our study successfully constructs a complete quantum circuit for Shors algorithm that can actually be implemented. This research combined both classical multiplication operation

and quantum computers for reducing the circuit depth and the number of qubits as much as possible. In addition, Our quantum circuit greatly improves the efficiency of Shors algorithm by using fewer qubits. Furthermore, universal logic gates were adopted to realize operations required by Shors algorithm, including QFT, inverse QFT, the QFT-based adder, CNT-ADD, CNT-CNT-ADD, CNTCNTCNT-R_n, CNT-CNT-$(b+a) \mod N$, and CNT-$(b+ax) \mod N$.

According to the design of all circuit modules we mentioned before, we successfully factored 15 by using the constructed modules on IBM's platform. Although the noise is high with the actual operation of the quantum computer running on IBM's hardware, which often produces certain errors with a large number of logic gates and long runtime (or circuit depth). Future research will explore how to reduce the noise generated by the quantum circuit due to the hardware.

In addition, we herein calculated the complexity and the number of qubits required by the circuit to estimate the cost of cracking the n-bit RSA encryption. The related modules established can be used for the future development of other quantum algorithms. These modules can be applied directly for integer operations.

Finally, we summarize the main conclusions of this article as following:

1. The circuit we design to implement Shor's algorithm's first complete quantum circuit rather than a simplified circuit. Moreover, we decompose the composite number 15 on IBM's platform to validate our design of modules. Although the time and space complexity are still greater than traditional computers, it still proves that Shor's algorithm can be implemented in quantum computers without considering the total number of qubits.

2. To reduce the overall operation cost of Shor's algorithm, we have made the following efforts:

   (a) In the part of the adder, we use a quantum adder designed by quantum Fourier transform and design modules for it. The advantage is that our hardware needs fewer qubits, although the design of quantum gates is more complex than ordinary.

   (b) We abandon quantum computers in multiplication steps. Instead, we use traditional computers to process multiplications and then we use the results of the operations as inputs to the quantum circuit to process subsequent calculations. Actually, the cost of a semi-classical computer model designed in this way is much lower than a fully quantum computer model.

3. We decompose $15 = 5 \times 3$ on the IBM's platform with 63 qubits. However, to decompose a composite number requires a large number of quantum gates. The deeper circuit depth, the higher noise generated. The processing of noise must start with the improvement of quantum computer hardware. However, we can still see the peak probability from the quantum state of register A $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$, and finally complete the decomposition of the composite number 15.

The development of quantum algorithms is the key to the success of the development of quantum computers. Through the circuit realization and improvement of Shor's algorithm, we hope this research is helpful to future quantum algorithms.

References

[1] Dugic, Miroljub; Cirkovic, Milan M. Quantum parallelism in quantum information processing. International Journal of Theoretical Physics, 2002, 41.9: 1641-1649. 1

[2] Horodecki, R., et al., Quantum Entanglement. Rev. Mod. Phys., 2007. 81. 1

[3] Hanggi, Esther; Renner, Renato; Wolf, Stefan. Quantum cryptography based solely on Bell's theorem. arXiv preprint arXiv:0911.4171, 2009. 1

[4] Shor, Peter W., Preskill, John. Simple proof of security of the BB84 quantum key distribution protocol. Physical review letters, 2000, 85.2: 441. 1

[5] Holloway, Catherine, et al. Optimal pair generation rate for Entanglement-based QKD. arXiv preprint arXiv:1210.0209, 2012. 1

[6] Hemmer, P. and J. Wrachtrup, Where Is My Quantum Computer? Science, 2009. 324(5926): p. 473. 1

[7] Benioff, Paul (1980). "The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines". Journal of Statistical Physics. 22 (5): 563591. 1

[8] Toffoli, Tommaso. J. W. de Bakker and J. van Leeuwen, Automata, Languages and Programming, Seventh Colloquium. Noordwijkerhout, Netherlands: Springer Verlag. pp. 632644 1

[9] Show, Peter W. Algorithms for quantum computation: discrete logarithms and factoring. In:Proceedings 35th annual symposium on foundations of computer science. Ieee, 1994. p. 124-134. 1

[10] Show, Peter W. "Scheme for reducing decoherence in quantum computer memory". Physical Review A.,1996 52 (4): R2493R2496. 1

[11] DiVincenzo, David P. "Topics in quantum computers." Mesoscopic electron transport. Springer, Dordrecht, 1997. 657-677. 1

[12] Raussendorf, R; Briegel, H. J, "A One-Way Quantum Computer". Physical Review Letters.,2001, 86 (22): 518891 1

[13] Damgard, Ivan B., et al. Cryptography in the bounded-quantum-storage model. SIAM Journal on Computing, 2008, 37.6: 1865-1890. 1

[14] Martin-Lopez, Enrique, et al. Experimental realization of Shor's quantum factoring algorithm using qubit recycling. Nature photonics, 2012, 6.11: 773-776 1

[15] Gamel, Omar, and Daniel FV James. "Simplified Factoring Algorithms for Validating Small-Scale Quantum Information Processing Technologies." arXiv:1310.6446 (2013). 1

[16] Geller, Michael R., and Zhongyuan Zhou. "Factoring 51 and 85 with 8 qubits." Scientific reports 3.1 (2013): 1-5. 1

[17] Beauregard, Stephane. Circuit for Shor's algorithm using 2n+ 3 qubits.arXiv preprint quant-ph/0205095, 2002. 1

[18] Wiener, Michael J. Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information theory, 1990, 36.3: 553-558. 2

[19] Boneh, Dan, et al. Twenty years of attacks on the RSA cryptosystem. Notices of the AMS, 1999, 46.2: 203-213. 2

[20] Shand, Mark; Vuillemin, Jean. Fast implementations of RSA cryptography. In:Proceedings of IEEE 11th Symposium on Computer Arithmetic. IEEE, 1993. p.252-259. 2

[21] Vandersypen, Lieven Mk, et al. Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance.Nature, 2001, 414.6866: 883-887. 2.2

[22] Stephane Beauregard. Circuit for Shor's algorithm using 2n+3 qubits, Quantum Information and Computation. Volume 3, Issue 2, March 2003, 175185.

[23] Ruiz-Perez, Lidia; Garcia-Escartin, Juan Carlos. Quantum arithmetic with the quantum Fourier transform.Quantum Information Processing, 2017, 16.6: 152. 3.1

[24] Browne, Daniel E. Efficient classical simulation of the quantum Fourier transform.New Journal of Physics, 2007, 9.5: 146. 3.1

[25] Kim, Taewan; Choi, Byung-Soo. Efficient decomposition methods for controlled-R n using a single ancillary qubit. Scientific reports, 2018, 8.1: 1-7. 3.1

[26] Draper, Thomas G. Addition on a quantum computer. arXiv preprint quant-ph/0008033, 2000. 3.2

[27] Kim, Taewan; Choi, Byung-Soo. Efficient decomposition methods for controlled-R n using a single ancillary qubit.Scientific reports, 2018, 8.1: 1-7. 3.3

[28] Shende, Vivek V., Markov, Igor L. On the CNOT-cost of TOFFOLI gates. arXiv preprint arXiv:0803.2316, 2008. 3.3

[29] Lavor, Carlile, MANSSUR, L. R. U., PORTUGAL, Renato. Grover's Algorithm: quantum database search. arXiv preprint quant-ph/0301079, 2003. 3.4

[30] Williams, C.P., Explorations in quantum computing. [electronic resource]. 2nd ed. Texts in computer science. 2011: Springer-Verlag London Limited. 3.4

[31] Daniel Volya, Prabhat Mishra. Special Session: Impact of Noise on Quantum Algorithms in Noisy Intermediate-Scale Quantum Systems. 2020 IEEE 38th International Conference on Computer Design (ICCD). October 2020, Complete, 1-4. 4.2

[32] Robin Harper, Steven T. Flammia, Joel J. Wallman. Efficient learning of quantum noise. Nature Physics, VOL 16, December 2020, 11841188. 4.2

[33] Cheng Xue, Zhao-Yun Chen, Yu-Chun Wu, Guo-Ping Guo. Effects of Quantum Noise on Quantum Approximate Optimization Algorithm. CHIN. PHYS. LETT. Vol. 38, No. 3 (2021). 4.2