Programmable architecture for quantum computing

Jialin Chen,[1,2] Lingli Wang,[1,*] Edoardo Charbon,[3] and Bin Wang[2]
[1]*State Key Laboratory of ASIC & System, Fudan University, People's Republic of China*
[2]*Department of Electrical Engineering, Fudan University, People's Republic of China*
[3]*Department of Circuits and Systems, TU Delft University, Netherlands*
(Received 27 January 2013; revised manuscript received 16 June 2013; published 9 August 2013)

A programmable architecture called "quantum FPGA (field-programmable gate array)" (QFPGA) is presented for quantum computing, which is a hybrid model combining the advantages of the qubus system and the measurement-based quantum computation. There are two kinds of buses in QFPGA, the local bus and the global bus, which generate the cluster states and general multiqubit rotations around the z axis, respectively. QFPGA consists of quantum logic blocks (QLBs) and quantum routing channels (QRCs). The QLB is used to generate a small quantum logic while the QRC is used to combine them properly for larger logic realization. Considering the error accumulating on the qubus, the small logic is the general two-qubit quantum gate. However, for the application such as n-qubit quantum Fourier transform, one QLB can be reconfigured for four-qubit quantum Fourier transform. Although this is an implementation-independent architecture, we still make a rough analysis of its performance based on the qubus system. In a word, QFPGA provides a general architecture to integrate different quantum computing models for efficient quantum logic construction.

DOI: 10.1103/PhysRevA.88.022311 PACS number(s): 03.67.Lx

I. INTRODUCTION

Quantum computers could outperform their classical counterparts when solving many fundamental and complex problems such as integer factorization [1], database search [2], global binary optimization [3], linear equation solving [4], and so on [5]. In a quantum computer, these problems are solved by using various quantum algorithms, which often require a large number of qubits to obtain results. Therefore, how to design programmable and scalable quantum computer architectures is at the core of quantum computation research.

Quantum computation is executed under the framework of quantum mechanics. There are several models for quantum computation. The most widely used one is the quantum circuit (QC) model [6]. The QC model is a quantum edition of "the reversible classical circuit model." Computation in the QC model is run by a sequence of quantum gates and represented by its circuit diagram, where the connecting wires stand for the logical qubits, which carry the information, and the information is processed by the sequence of quantum gates. In the end, the result of the computation is read out by the projective measurements on the qubits.

The measurement-based quantum computation (MBQC) is another well-recognized model [6–9]. In MBQC a highly entangled cluster state is generated, and then through single-qubit measurements alone, any desired unitary can be implemented only up to random but known Pauli transformations [10,11]. The MBQC methods enable one to simulate any quantum circuit on a sufficiently large two-dimensional graph state by arranging the spatial pattern of measurement bases for the graph qubits according to the temporal order of required quantum gates.

Recently, a number of hybrid quantum models have been proposed, such as the hybrid quantum computational model (HQCM) [12] and the ancilla-based computation (ABC) model [13]. The main idea behind these models is to use two or more different types of quantum models or systems to perform calculations simultaneously. In other words, HQCM combines elements of both QC and MBQC models while ABC uses a flying ancilla qubit that mediates between fixed qubits. In fact, the ancilla-based model is among the most promising for scalable chip-based quantum computer architectures as it allows the interaction between distant qubits without SWAP gates so as to offer individual addressability. For this reason, we will concentrate on the qubus system [14,15]—an important kind of ancilla-based model—which combines both matter and optical elements. In this hybrid system, the quantum gates among qubits of one type can be mediated by a shared bus of the other type [16,17]. The disadvantage of hybrid systems is that some specific types of error may be introduced, because of the use of the mediating ancilla. In cases where the ancilla is not destroyed after each gate, there is the additional possibility of errors propagating through ancilla reuse.

In this paper, our goal is to realize universal quantum computing by providing an array architecture that can be appropriately programmed. Programmability means flexibility and universality, while the size and the design of the architecture are fixed, but scalable. It was shown by Nielsen and Chuang [18] that in a pure quantum computer no such fixed quantum gate array architecture can be realized in a deterministic fashion. This is because the dimension of the program system has to be infinite as even for a single qubit the set of operations is infinite. However, a quantum computer can be aided by a classic control computer [19]. In essence, one can choose classic bits as the program system to decide which operations are carried out. Inspired by this concept, we present a programmable architecture called "quantum FPGA" (QFPGA) similar to classical field-programmable gate arrays (FPGAs) in the area of traditional digital circuits.

As shown in Fig. 1, all commercial FPGA devices consist of a large number of programmable logic blocks and programmable routing resources. Each block implements a small digital logic and programmable routing resources allow the

*llwang@fudan.edu.cn

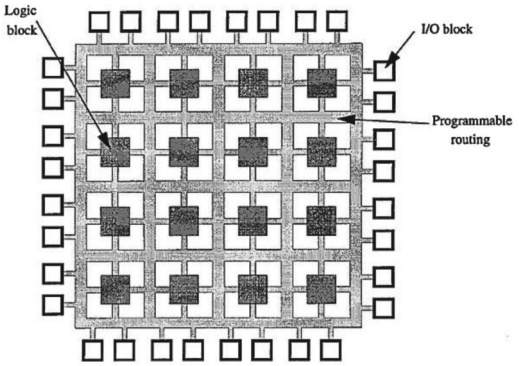

FIG. 1. A generic FPGA architecture [20].

logic block inputs and outputs to be connected to form larger circuits.

In the traditional FPGA, three factors determine performance: the quality of the computer-aided design (CAD) tools used to map circuits into the FPGA, the quality of the FPGA architecture, and the electrical (i.e., transistor-level) design of the FPGA [20]. Similarly, quantum logic synthesis CAD tools, programmable architecture, and quantum gates implementation are the three most important factors of the QFPGA.

Quantum logic synthesis has shown that any unitary transformation can be exactly realized if the set of single-qubit operation plus CNOT are allowed as elementary gates [21]. We note that quantum gate arrays in the literature [22,23] are generally based on some matrix decompositions such as QR decompositions [24], CS decomposition [25], QS decomposition, and some variety of these methods [26]. The purpose of these works is to optimize the number of CNOT gates. However, these methods are suitable to implement a quantum gate in a custom style, rather than in a fixed programmable architecture. Reference [27] presents a programmable quantum circuit scheme to simulate any operator by setting the angle values in the circuit; thus it can provide a fixed circuit but at the cost of many multiqubit controlled gates (i.e., a programmable two-qubit circuit needs sixteen five-qubit controlled-rotation-y gates). Considering the current experimental limits of quantum mechanics, it may not be feasible to realize a large quantum circuit based on that scheme. In this paper, we use a method based on CS and QS decompositions, which can decompose an arbitrary n-qubit gate into a circuit containing m-qubit gates ($m < n$) and multiqubit diagonal gates. Mathematically, it means that any n-bit quantum gate from the group $\mathrm{U}(2^n)$ equals the product of several $\mathrm{U}(2^m)$ matrices and diagonal unitary matrices.

As we know, programmability is the key feature of the MBQC. In a classical approach, implementing a circuit in a traditional FPGA requires that hundreds of thousands or even millions of programmable switches and configuration bits be set to the proper states, on or off. The similar situation exists in MBQC. Given a large enough two-dimensional cluster state (lattice in most cases), configuration bits are the measurement basis of every qubit. However, it would become much more difficult to implement a large quantum circuit using a lattice cluster state without a certain structure, as the placement of the measurement is very complex. Worse still, a large cluster state means a high probability of qubit errors, such as phase flip and decoherence. Therefore, one of the most important requirements for a QFPGA is that we develop a structural architecture to use the MBQC in a certain predesigned way.

Our proposed architecture consists of two parts: The first part is an array of static, long-lived quantum memories that can interact with moving, short-lived quantum registers—flying qubits—via a fixed interaction [28]. We call this part "quantum routing channels" (QRCs), which are used as quantum routing resources and to realize diagonal unitary operators. The second part is the "quantum logic block" (QLB) to implement a general two-qubit quantum gate. Theoretically, as any $\mathrm{U}(2^n)(n \geqslant 2)$ can be decomposed into U(4) and diagonal unitary operators, thus the architecture consisting of QLBs and QRCs is also universal. In contrast to the general MBQC, here quantum logic inputs and outputs are restricted in QRCs. Moreover, by using qubus to design an optimal scheme for dynamic cluster-state generation, we will show that our architecture is feasible as the error probability ϵ is below 10^{-2}, which is an accepted threshold of the MBQC.

This paper is organized as follows. In Sec. II, we review CS and QS decompositions, which are the most fundamental logic synthesis methods for QFPGA. In Secs. III and IV, we give a brief review of the MBQC and element operators of the qubus quantum computer, which are critical components in our architecture. In Sec. V, we give a detailed description of the QFPGA which consists of four parts. The first two parts are the structures of the QRC and QLB, the third part is the error analysis, and the fourth part describes the whole architecture of QFPGA. Then in Sec. VI, based on QFPGA, we provide two applications—the general quantum gates and quantum Fourier transform to show some important advantages of our architecture. Finally, we conclude our paper in Sec. VII with a summary and outlook. The Appendix deals with the decomposition of a CS gate.

II. QUANTUM LOGIC SYNTHESIS

In this section we will review the main results derived in Ref. [26]. Two quantum circuits are equivalent if matrix products of their corresponding gates are identical. In order to synthesize a given unitary matrix, equivalent circuits may be applied to reduce the circuit cost. To do this, various quantum circuit identities have been proposed in recent years [26,29,30].

An arbitrary single-qubit gate U(2) can be decomposed into R_z and R_x rotation gates (ZXZ decomposition) as shown in Eq. (1). Hence, it can be implemented as a sequence of at most three elementary gates,

$$\mathrm{U}(2) = R_z(\alpha)R_x(\beta)R_z(\gamma), \tag{1}$$

where α, β, and γ represent the angles of rotation.

For the n-qubit unitary matrix $\mathrm{U}(2^n)$, cosine-sine decomposition (CSD) can be expressed by Eq. (2) where A_1, B_1, A_2, and B_2 are unitary $2^{n-1} \times 2^{n-1}$ matrices and C and S are unitary $2^{n-1} \times 2^{n-1}$ diagonal matrices with real elements such that $C^2 + S^2 = I_{n-1}$, where I is the identity operator.

$$\mathrm{U} = \begin{pmatrix} A_1 & 0 \\ 0 & B_1 \end{pmatrix} \begin{pmatrix} C & S \\ -S & C \end{pmatrix} \begin{pmatrix} A_2 & 0 \\ 0 & B_2 \end{pmatrix}. \tag{2}$$

In Eq. (2) above, the left and right factors $A_j \oplus B_j$ are quantum multiplexors controlled by the most significant qubit

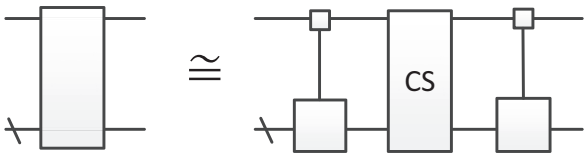

FIG. 2. CS decomposition of a unitary matrix.

which determines whether A_j or B_j is to be applied to the lower-order qubits. The central factor has the same structure as the rotation-y gate R_y [26]. A closer inspection reveals that it applies a different R_y gate to the most significant bit for each classical configuration of the low-order bits. Thus, the CSD of a unitary matrix can be seen in Fig. 2.

Since the left and the right matrices in Eq. (2) are block diagonals, they can be further decomposed into two generic unitary matrices and a specific block diagonal matrix, which is called quantum Shannon decomposition (QSD) [26].

$$\begin{pmatrix} A_j & 0 \\ 0 & B_j \end{pmatrix} = \begin{pmatrix} V & 0 \\ 0 & V \end{pmatrix} \begin{pmatrix} D & 0 \\ 0 & D^\dagger \end{pmatrix} \begin{pmatrix} W & 0 \\ 0 & W \end{pmatrix}. \quad (3)$$

From this expression, we obtain the relation $A_j B_j^\dagger = V D^2 V^\dagger$. So, both V and D can be easily calculated through the diagonalization of $A_j B_j^\dagger$. Furthermore, $W = DV^\dagger B_j$. It should be noted that matrix $D \oplus D^\dagger$ of dimension $2^n \times 2^n$ has the diagonal form of Eq. (4), written as $D_n(\theta_0, \theta_1, \ldots, \theta_{2^{n-1}})$ or D_n for short. The D_n gate is in fact a multiplexed R_z gate acting on the most significant bit in the circuit. Because the CS matrix is a controlled-R_y gate to the most significant qubit according to the lower-order qubits, we can also apply QSD as shown in Fig. 3.

Hence, any arbitrary n-qubit gate can be implemented by a circuit containing three D_n gates and six generic $(n-1)$-qubit gates, which can be viewed as cofactors of the original operator. For $n \geqslant 2$, by using CSD and QSD recursively, any U(2^n) can be decomposed into several U(4) and D_k gates $(2 \leqslant k \leqslant n)$.

$$\begin{pmatrix} e^{i\theta_0} & & & & & & & \\ & e^{i\theta_1} & & & & & & \\ & & \ddots & & & & & \\ & & & e^{i\theta_{2^n-1}} & & & & \\ & & & & e^{-i\theta_0} & & & \\ & & & & & e^{-i\theta_1} & & \\ & & & & & & \ddots & \\ & & & & & & & e^{-i\theta_{2^n-1}} \end{pmatrix} \quad (4)$$

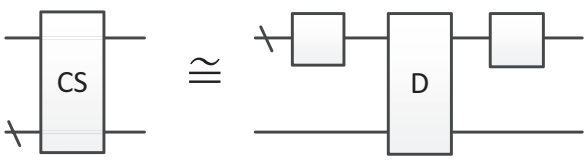

FIG. 3. Decomposition of a CS gate.

III. MEASUREMENT-BASED MODEL

We give a brief review of the MBQC, including the preparation of cluster states [8] and three examples which are very important in our architecture. For a fuller presentation we refer the reader to a tutorial [10] and Ref. [11] which use a high-level diagrammatic language to understand cluster states in a simple and fascinating way. Furthermore, we represent the Pauli vector operator $\bar{\sigma}$ by (X,Y,Z) and the identity operator by I.

Cluster states can be realized in many physical systems by first preparing all the qubits in the state $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$. Then, entanglement between each pair of nearest-neighbor qubits is established by the controlled-Z gate:

$$CZ(a,b) = |0\rangle_a\langle 0| \otimes I^b + |1\rangle_a\langle 1| \otimes Z^b. \quad (5)$$

Here, the indices a and b stand for the qubits at lattice site a and its nearest-neighbor lattice site b of graph G, respectively.

Once the cluster state resource is ready, then the logical input qubits are attached to the resource via the same entangling operations given by Eq. (5). Now the computation is carried out by a sequence of single-qubit (adaptive) projective measurements in a certain direction of the Bloch sphere and in a certain temporal order. The choice of measurement basis is characterized by the direction of measurement (θ,ϕ) in the Bloch sphere, where it represents two kets,

$$|\uparrow(\theta,\phi)\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle, \quad (6)$$

and

$$|\downarrow(\theta,\phi)\rangle = \cos\frac{\theta}{2}|0\rangle - e^{i\phi}\sin\frac{\theta}{2}|1\rangle, \quad (7)$$

respectively.

Example 1. $R_x(\alpha)R_z(\beta)$ can be realized by the three-qubit cluster states in Fig. 4 where the measurement direction is in the XY plane, so α and β in the circle represent $(\pi/2,\alpha)$ and $(\pi/2,\beta)$, respectively. The following measurement pattern implements the unitary transformation:

$$\begin{aligned} X^{m_2} H R_z(\beta) X^{m_1} H R_z(\alpha) &= X^{m_2} H R_z(\beta) H Z^{m_1} R_z(\alpha) \\ &= X^{m_2} R_x(\beta) Z^{m_1} R_z(\alpha) \\ &= X^{m_2} Z^{m_1} R_x[(-1)^{m_1}\beta] R_z(\alpha), \end{aligned} \quad (8)$$

where we introduce a binary digit $m_i \in \{0,1\}$ to represent a measurement outcome of $(-1)^{m_i}$ of qubit i and use the identities $HR_z(\theta)H = R_x(\theta)$ and $ZR_x(-\theta) = R_x(\theta)Z$. It should be noted that the two measurement outcomes $m = 0,1$ for every qubit of the cluster state are equally probable because the reduced density matrix for each qubit is the completely mixed state $I/2$. Due to this randomness introduced by the measurements, temporal ordering among the measurements is necessary as it can keep the computation deterministic. So in

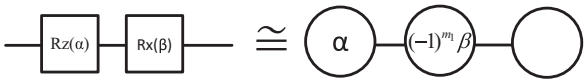

FIG. 4. The measurement pattern to realize $R_x(\alpha)R_z(\beta)$.

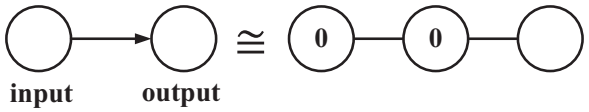

FIG. 5. The construction of an identity gate (quantum wire).

this case, to obtain the desired unitary, the measurement angle of the second qubit must depend on the measurement value m_1 of the first qubit, that is, $(-1)^{m_1}\beta$. In other words, two time steps are needed to accomplish this operator. By the way, $R_x(\alpha)$ is realized when $\beta = 0$ and it also needs three-qubit cluster states.

In principle we can correct by-product operators of each gate step by step, but it is more convenient to let them pass through the gates, and use the propagation relations to keep track of the accumulated measurement outcomes to avoid logic errors. For example, the propagation relation of the single-qubit gate is

$$R_z(\alpha)R_x(\beta)R_z(\gamma)(X)^x(Z)^z = (X)^x(Z)^z R_z((-1)^x\alpha)R_x((-1)^z\beta)R_z((-1)^x\gamma). \quad (9)$$

Then we can change the corresponding measurement basis of the current gate to obtain the correct results.

Example 2. Quantum wire (identity gate). The identity gate can be used as a quantum wire to transport a qubit to a different site of cluster state. For example, two cascaded gates might not lie in the adjacent sites of the cluster state; then we should use the identity gate to link them, so it is a very useful tool in gate construction considering the flexibility and scalability.

The quantum wire can be easily realized in Fig. 5. The unitary transformation is

$$X^{m_2}HX^{m_1}H = X^{m_2}Z^{m_1}, \quad (10)$$

so this is a quantum wire with by-product operators according to the measurement outcomes. In the following text, we will use arrows to represent quantum wires.

The propagation relation of the quantum wire is very easy,

$$I(X)^x(Z)^z = (X)^x(Z)^z I. \quad (11)$$

Example 3. The unitary operation for the n-qubit rotation around the z axis is

$$R_{zz\cdots z}^{12\cdots n}(\theta) = \exp(-i\theta Z^{\otimes n}). \quad (12)$$

Here, we define θ as the interaction value among n qubits. In MBQC, Eq. (12) can be easily implemented by performing a single measurement on the ancilla qubit a in the $(1+n)$-qubit star graph state as shown in Fig. 6 [12].

Measuring qubit a in basis $(-\phi, -\pi/2)$ is equivalent to performing $[R_x(\phi)]_a$ and then measuring Z_a; hence it implements the logical unitary $R_{zz\cdots z}^{12\cdots n}(\phi)$ on the desired n qubits with by-product $Z^{\otimes n}$ according to the measurement value m_a of qubit a,

$$|\psi_{\text{out}}\rangle = (Z^{\otimes n})^{m_a} R_{zz\cdots z}^{12\cdots n}(\theta)|\psi_{\text{in}}\rangle. \quad (13)$$

The propagation relation of this rotation gate is

$$R_{zz\cdots z}^{12\cdots n}(\theta)(X)^x(Z)^z = (X)^x(Z)^z R_{zz\cdots z}^{12\cdots n}((-1)^x\theta). \quad (14)$$

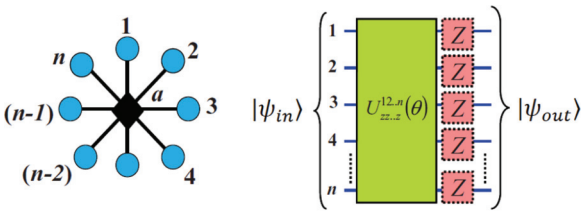

FIG. 6. (Color online) (a) $(1+n)$-star graph state $|\psi\rangle_{1+n}$ where the ancilla qubit a is represented by the black diamond. (b) The effect on the input state $|\Psi_{\text{in}}(n)\rangle$ when the qubit a is measured in an appropriately chosen basis [12].

IV. THE QUBUS QUANTUM COMPUTER

We give a brief introduction to the qubus system; a more detailed description is available in Refs. [14–16]. A qubus quantum computer is a hybrid system of a processing unit made of qubits and a continuous variable field "bus" that generates interactions and transfers information between qubits [15]. We focus on the qubus quantum computer because it can generate cluster states as error free as possible, which is extremely important for MBQC.

The element operators of interactions in the qubus system can be written in the form

$$D(\sigma_z\beta) = \exp(\sigma_z\beta a^\dagger - \sigma_z\beta^* a), \quad (15)$$

where $\beta = \chi t e^{i(\theta-\frac{\pi}{2})}$; thus

$$\beta = \begin{cases} i\chi t, & \text{if} \quad \theta = 0 \\ \chi t, & \text{if} \quad \theta = \pi/2 \end{cases}.$$

Here χ is nonlinearity strength, a ($a^\dagger$) are the field annihilation (creation) operators, and $\theta = 0$ ($\pi/2$) describes coupling of the qubit to the position (momentum) quadrature of the field. According to the following equation, if σ_1 and σ_2 commute, then

$$D(\sigma_1\beta_1)D(\sigma_2\beta_2) = \exp\left[\frac{(\beta_1\beta_2^* - \beta_1^*\beta_2)\sigma_1\sigma_2}{2}\right] D(\sigma_1\beta_1 + \sigma_2\beta_2). \quad (16)$$

This expression gives us a phase factor if one of β_1 and β_2 is a real number and the other is an imaginary number. This means that a phase factor is generated when we connect two qubits to opposing quadratures of the bus, but that no phase factor is generated when we connect two qubits to the same quadrature of the bus.

From Eq. (16), if we apply the following sequence displacement operator between qubits 1 and 2, then the sequence performs a geometric phase gate U between them,

$$\begin{aligned} \text{U} &= D(\beta_1\sigma_{z1})D(-i\beta_2\sigma_{z2})D(-\beta_1\sigma_{z1})D(i\beta_2\sigma_{z2}) \\ &= \exp(i\beta_1\beta_2\sigma_{z1}\sigma_{z2})D(\beta_1\sigma_{z1} - i\beta_2\sigma_{z2}), \\ &D(-\beta_1\sigma_{z1})D(i\beta_2\sigma_{z2}) \\ &= \exp(2i\beta_1\beta_2\sigma_{z1}\sigma_{z2})D(-i\beta_2\sigma_{z2})D(i\beta_2\sigma_{z2}) \\ &= \exp(2i\beta_1\beta_2\sigma_{z1}\sigma_{z2}). \end{aligned} \quad (17)$$

When $\beta_1\beta_2 = \pi/8$, this provides a gate which is a local equivalent to a CZ gate required for cluster-state construction:

$$[R_{z,1}(\pi/2)R_{z,2}(\pi/2)]\mathrm{U} = CZ. \tag{18}$$

As the local unitary is $R_z(\pi/2)$, so all measurement basis (θ,ϕ) obtained in Sec. III to drive the computation should be modified as $(\theta,\phi - d\pi/2)$, where d is the degree of a vertex in cluster states. For example, the ancilla qubit in the $(1+n)$ star graph states has a degree n (see Fig. 6).

V. THE QUANTUM PROGRAMMABLE ARCHITECTURE

As described in Sec. II, any quantum circuits can be decomposed recursively into a sequence of D gates and two-qubit quantum gates. Correspondingly, our architecture consists of two parts: quantum routing channels (QRCs) and quantum logic blocks (QLBs). The main purpose of QRCs is to realize D gates, and the QLBs are set to realize general two-qubit quantum gates, respectively.

As shown in Fig. 7, a QRC consists of the memory qubits (long-lived qubits) and channel qubits (ancilla qubits), while QLBs are short-lived qubits that possess short coherence time. The different choices of qubits are due to the fact that the running time of a QRC is often longer than a QLB; i.e., much more time is needed to realize a large D_n gate while a two-qubit gate is comparatively easy to implement.

In electronics and especially synchronous digital circuits, a clock signal is a particular type of signal that triggers and synchronizes actions of circuits. A clock cycle is the minimum time unit required to perform a basic operation. In some cases, more than one clock cycle is required to perform a predictable operation. Taking advantage of this concept, we suppose that in QFPGA a bus operation can be implemented within one clock cycle and so is a single-qubit measurement. The motivation of this idea is that the timing performance of QFPGA can be analyzed as the conventional digital circuits.

In next three parts, we will describe the details of QRCs, QLBs, and how they are combined to build a programmable architecture.

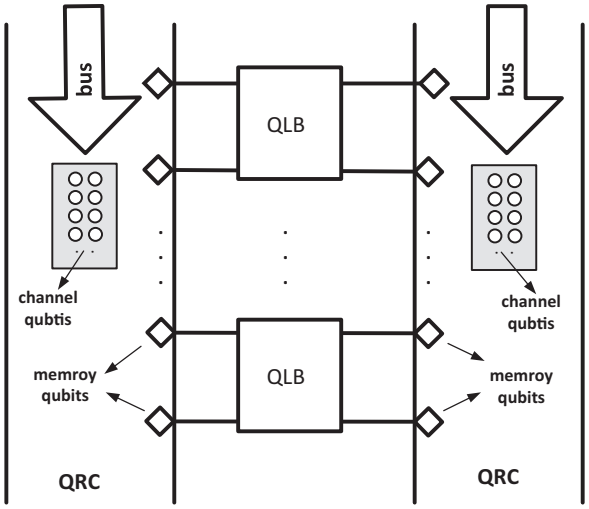

FIG. 7. Pictorial sketch of programmable quantum computing architecture.

A. Quantum routing channel (QRC)

One of the advantages of qubus is that we can use the bus for more efficient cluster-state construction, especially the star graph state. Consider the following sequence of unitary operators for n qubits provided that $\beta = \sqrt{\pi/8}$:

$$\prod_{l=1}^{n} D(\beta\sigma_{z,l})D(-i\beta\sigma_{z,a})\prod_{l=1}^{n} D(-\beta\sigma_{z,l})D(i\beta\sigma_{z,a}) = \exp\left[2i\beta^2\sigma_{z,a}\left(\sum_{l=1}^{n}\sigma_{z,l}\right)\right]. \tag{19}$$

Then, we use this method to construct D_n gates. Obviously, every D_n gate has 2^{n-1} independent variables from Eq. (4), it can be written into the expression

$$D_n = e^{i(Z\otimes I\cdots\otimes I\alpha_1 + Z\otimes I\cdots\otimes Z\alpha_2 + \cdots + Z\otimes Z\cdots\otimes Z\alpha_{2^{n-1}})}, \tag{20}$$

where Z and I represent Pauli-Z matrix and identity matrix, and $\alpha_1,\alpha_2,\ldots,\alpha_{2^{n-1}}$ are 2^{n-1} independent variables. In other words, each term in Eq. (20) is an n-qubit rotation operator that is defined in Eq. (12). Therefore, $2^{n-1}-1$ multiqubit rotation gates are enough to perform a D_n gate, as the first variable α_1 is just a single-qubit rotation about the z axis. For example, D_2 gate can be realized by two rotation gates $R^{12}_{zz}(\alpha_0)$ and $R^{12}_{zI}(\alpha_1)$. Since any rotation gate $R^{12\cdots n}_{zz\cdots z}(\theta)$ can be implemented by performing a single measurement in the $(1+n)$-qubit star graph state, D_n gates can be realized.

Another advantage of the qubus system is that we can reuse the bus [17], i.e., we can construct several star graph states in one bus, because in each step we could couple a new channel qubit and couple or decouple some memory qubits to the opposition quadrature of the bus. The reader may easily verify that if this method is used to implement an n-qubit D gate where $n>3$, some memory qubits must be visited more than once. For example, if $n=4$ (see Fig. 8), star graph states only from (i) to (v) can be generated in one bus if each qubit

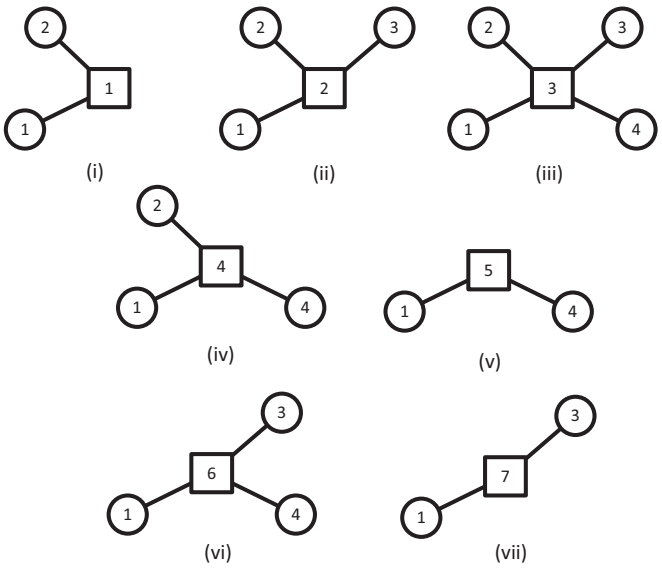

FIG. 8. Construction of a general D_4 gate in one bus. Square and circle qubits represent channel qubits and memory qubits, respectively.

can be visited once only. To generate (vi) and (vii), qubit 3 must be coupled to the bus again.

The detailed steps of implementing a general D_n gate are as follows:

(1) Prepare all required ancilla qubits in the state $|+\rangle$ as channel qubits.

(2) Couple all n memory qubits to position (momentum) quadrature of the bus and a channel qubit to momentum (position) quadrature of the bus.

(3) Measure the channel qubit in certain basis $[\theta, -\frac{(1+m)\pi}{2}]$, where m is the degree of this channel qubit.

(4) Decouple or recouple a certain memory qubit and simultaneously couple a new channel qubit to the bus.

(5) Repeat steps 3 and 4 until all desired star graph states are generated and measured.

An n-qubit D_n gate requires $2^{n-1}-1$ channel qubits at most and each channel qubit connects m memory qubits where $m \leqslant n$. In fact, only one channel qubit will connect n memory qubits. In conclusion, as long as the required channel qubits can be obtained, QRCs are efficient in implementing any D_n gates.

B. Quantum logic block (QLB)

The main function of QLBs is to realize the simple quantum logic: a general two-qubit quantum gate. To this end, consider $n = 2$ in Sec. II; the decomposition of a general two-qubit quantum gate is shown in Fig. 9(a), where we have used the commutative of R_z and D gates and the QSD of a two-qubit CS gate (see Appendix). Furthermore, from Eq. (20) any D_2 gate can be decomposed into a two-qubit R_{zz} gate and a single-qubit R_z gate, thus leading to Fig. 9(b).

From the discussion in Sec. III, both $R_x(\alpha)R_z(\beta)$ and D_2 gates can be realized by the certain measurement on three-qubit cluster states and as R_z commutes with all D gates, the last two R_z gates within the dotted square in Fig. 9(b) can be absorbed in the next gate; then we could ignore it at the moment. In principle, any two-qubit quantum gate can be realized in certain cluster states via the traditional MBQC method. However, as we use the qubus method to generate entanglements between two qubits, a slightly modified cluster state could be adopted. Recall Eq. (16), we can see that any $R_{zz}(\theta)$ gate can be efficiently realized as long as the interaction value between two qubits, 1 and 2, is $\theta = 2\beta_1\beta_2$. In other words, using qubus to generate three R_{zz} gates directly, three ancilla qubits can be saved which should be used to form corresponding star graph states (see Fig. 6).

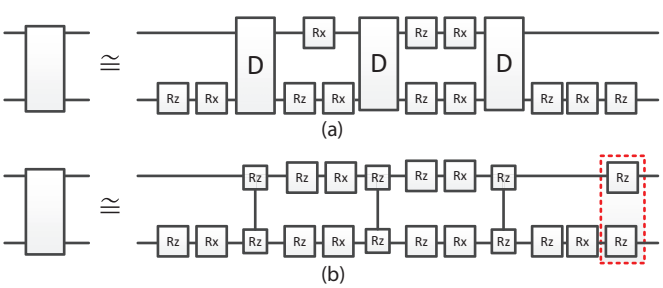

FIG. 9. (Color online) The general decomposition of a two-qubit quantum gate.

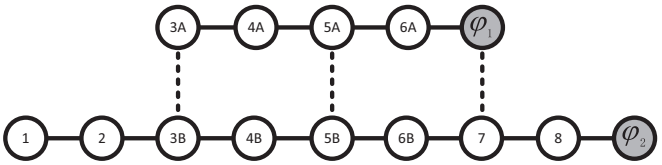

FIG. 10. The cluster states needed to realize a general two-qubit quantum gate, where the number in circles represent the order of measurement. (Since some qubits will be measured at the same cycle, a capital letter is appended after the numbers for discrimination.) The dotted line means certain $R_{zz}(\theta)$ gate between two qubits. Two gray circles represent two output qubits.

Therefore, the final cluster states needed are shown in Fig. 10, where three dotted lines mean three $R_{zz}(\theta)$ gates between two qubits and the numbers in circles represent the order of measurements. In total, 14 qubits are needed to realize a general two-qubit quantum gate. Because of the adaptive measurements, theoretically it will take at least eight clock cycles of measurements and three $R_{zz}(\theta)$ gates to complete the computation.

We now consider modifying the above cluster states to constitute a more compact QLB. The idea is to ignore two output qubits, as they can be memory qubits inside a QRC. Then we focus on how to realize the rest of the 12-qubit cluster states within a QLB. Suppose that these 12 qubits form a 4×3 lattice and a local bus within it; we can use it to generate the desired cluster states. This is because the displacement operators on the qubus allow a qubit on one quadrature to become entangled with all qubits on the other. For example to generate the cluster states of Fig. 10, if qubits 1 and 3B couple to the momentum quadrature of the bus, then qubit 2 should be coupled to the position quadrature and so on. To avoid the unwanted entanglement in the QLB, four qubits at most can remain on the bus: one position-quadrature qubit and three momentum-quadrature qubits or vice versa. By using this method properly, the desired cluster states of QLB can be realized.

Of course, to obtain the results of the QLB, we should link two memory qubits as outputs (see "φ_1" and "φ_2" in Fig. 10). This step is made easy by extending the QLB bus in the QRC, and no channel qubit is needed. Furthermore, we can also directly link two or more QLBs without using any memory qubits. This advantage will save a great number of memory qubits if some QLBs are cascaded without any D_n gates between them.

We should note that except for three qubits, operations between other qubits and the bus are just $D(\beta\sigma_z)$ or $D(i\beta\sigma_z)$, $\beta = \chi t = \sqrt{\pi/8}$; i.e., the interaction time between these qubits and the bus are the same. Thus in order to simplify the analysis, provided that the QLB bus can interact with multiple qubits for different displacement operators in one clock cycle, then we will find that it should take at most 19 clock cycles to complete the function of a QLB. For example, Fig. 11 shows the cluster states of the QLB when the system is at clock cycles 4, 9, and 19 respectively, where "M" means the qubit has been measured and "ON" means this qubit still stay on the bus. The details of the procedure can be seen in Table I; for example, the first row of the table means the qubit 1 is coupled to the

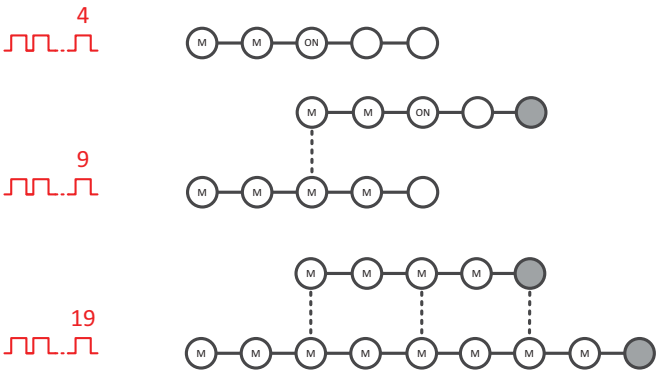

FIG. 11. (Color online) The cluster states of the QLB when the system is at clock 4, 9, and 19, here M means the qubit has been measured and ON means this qubit still stay on the bus.

bus at the first clock cycle, then decoupled at the second clock cycle, and measured at the third clock cycle in the end.

From this point of view, we can say that every qubit of the QLB has a unique clock cycle as its characteristic value, such as "1(2)". As qubit 5B and φ_1 in Fig. 10 must be visited twice, thus at most 32 displacement operations are needed for each QLB bus. Moreover, assuming that the measurement of a qubit needs one clock cycle, we can roughly analyze the performance of the QLB. From Table I we can see that to output the result through adaptive measurements, 19 clock cycles are necessary.

To make the QLB more physical senses, we consider a special case of superconducting charge qubits [31,32]. We first review the method of Ref. [31]: $D(i\beta\sigma_z)$ operation (coupling to the position quadrature) can be realized if a qubit is placed in a microwave field at a position where there is a nonzero electric field across it. On the other hand, $D(\beta\sigma_z)$ operation (coupling to the momentum quadrature) can be realized if such a qubit is at a position where there is a nonzero magnetic field normal to the plane of this qubit. Now consider red (dark gray) qubits of the QLB in Fig. 12, which are positioned to couple to the electric field antinode of a microwave mode and gray qubits positioned to couple to the magnetic field antinode of the same mode; then the desired displacement operator can be applied through a single microwave bus.

TABLE I. Performance of the QLB.

Measurement order	Operation cycle index	Measurement cycle index
1	1(2)	3
2	1(2)	4
3A	5(6)	7
3B	1(6)	7
4A	5(8)	9
4B	3(4)	8
5A	7(12)	13
5B	3(4), 11(12)	13
6A	7(10)	14
6B	11(14)	15
7	13(16)	17
8	13(18)	19
φ_1	9(10),15(16)	
φ_2	17(18)	

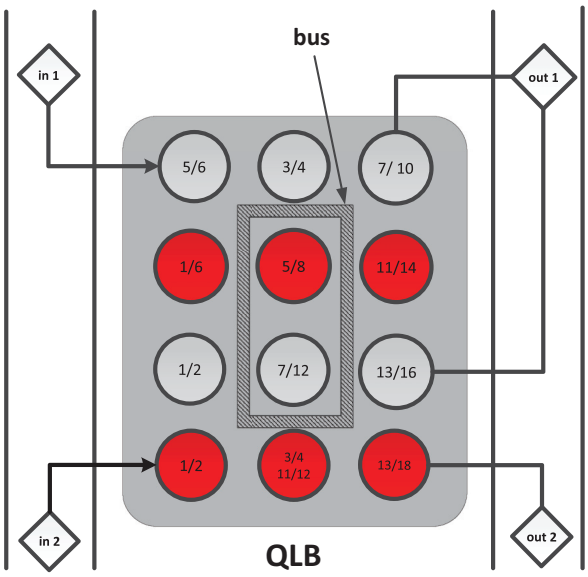

FIG. 12. (Color online) Schematic of the QLB (shadow). The QLB consists of 12 qubits and one bus. Gray qubits are coupled to position quadrature of the bus and red (dark gray) qubits are coupled to momentum quadrature of the same bus. Arrows mean that the corresponding CZ operations are taken by the QRC bus.

However, Fig. 12 is just one QLB configuration only, as it is the clock cycles of each qubit that decide the shape of cluster states rather than its position of the lattice. So the I/O (inputs/outputs) ports of a QLB are not necessarily fixed, since they can be moved to any other positions by changing the sequence of bus operations; i.e., the bus should be programmable to any sequence of interactions with the qubits. The only constraint is that the quadrature of each qubit cannot be changed dynamically as it is biased by the external field. In fact, the only requirement is that there are six qubits coupled to position quadrature and six qubits coupled to momentum quadrature of the bus. Thus any pair of qubits coupled to both quadrature of the bus can be two inputs or outputs.

In summary, a complete QLB should contain 12 qubits and one integrated local bus which can be programmable to interact with these qubits in the suitable sequence.

C. Error analysis

In this part, we use the method reported in Ref. [17] to calculate the dephasing error of our architecture. For N bus operations, the probability of dephasing is

$$\epsilon = \tfrac{1}{2}[1 - \exp(-N\gamma\tau - 4C\eta\beta^2)], \tag{21}$$

where τ is the time to perform for one bus operation and γ is the dephasing rate for qubits, C is the number of CZ gates constructed per bus, and η is the loss parameter for the bus. Consider the construction of the QLB in Fig. 12, then $N = 32$ and $C = 15$. If we take $\gamma\tau = 5 \times 10^{-4}$ and $\eta = 10^{-4}$ for practice [17], then the error of dephasing is $\epsilon = 0.0091$ which is below the error threshold 10^{-2}. On the other hand, coupling 17 qubits ($N = 34$) and creation of 20 CZ gate per bus are the

limitation under the error threshold—thus we can efficiently generate a 17-qubit star graph state, i.e., for a rotation gate larger than 16 qubits, we might use two or more buses in the QRC to generate a certain star graph state.

From Eq. (A2), we can see another potential reason for the choice of our QLB architecture: If we use pure MBQC method to implement the general two-qubit quantum gate and take only one bus to generate whole cluster states ($N = 36$, $C = 18$), it is likely to induce some uncorrectable errors ($\epsilon = 0.0103$) because it exceeds the threshold.

D. Quantum FPGA architecture

So far, we have proposed a simple architecture consisting of QLBs and QRCs (see Fig. 7), and show how they are implemented by the qubus system. Obviously, the most important advantage is the scalability and flexibility: QLBs and QRCs can be arranged in different kinds of array architectures to implement the universal quantum computation. In other words, the size and shape of the array can be made in various types to meet the application requirements. For an extreme example, in principle we can use many QLBs, which connect with only one QRC to construct a universal gate array. However, this will inevitably increase the length of the QRC and thus become impractical.

Based on the above discussion, we extend to the QFPGA architecture as in Fig. 13. Generally speaking, the FPGA-like architecture provides more flexibility for layout. Since QLBs can be placed in two dimensions, we can view this kind of architecture as having a two-dimensional topology.

There are two kinds of buses in the QFPGA: the local bus and the global bus. We have described how the local bus is mainly used to generate the desired cluster states within a QLB while the global bus is used to implement a general D_n gate or to establish links between different parts. In a sense we may say that the local bus generates the local information while the global bus entangles it.

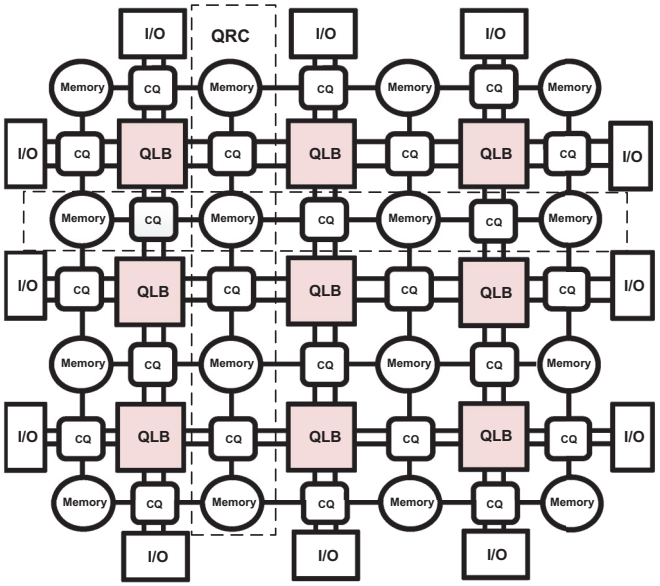

FIG. 13. (Color online) Quantum FPGA architecture. CQ is short for "channel qubits". I/Os represent the inputs and outputs of the QFPGA.

Ideally, a global bus should be able to link QLBs in any position. However, considering the physical limitation and the fault-tolerant requirement, the length of a global bus should be limited and only one bus can exist in a QRC. That means if we want to use two global buses in a QRC at the same time, one must be suspended until the other is over. Thus in order to realize a large quantum circuit, one object of the QLB placement is try to balance the lengths of global buses used and their spatial distribution to avoid the congestion of the QRCs.

Inside a QRC, each memory contains one qubit as the input or output of the corresponding QLB, and we can combine two or more QLBs directly into a large cluster state to save the number of memory qubits. In fact, this strategy depends on the coherence time of a QLB, which means that the output qubits exceeding the coherence time have to be registered in the memory. On the other hand, we notice that in the QFPGA all memory qubits are measured in the X basis; i.e., these qubits are in the state $|+\rangle$ or $|-\rangle$ after measurements. Hence they can be reused to form new cluster states in principle. This is because if the memory qubit is in the state $|-\rangle$, then nothing is changed but only the measurement value m_i of the previous qubit is set to $m_i \oplus 1$.

Last, but not least, provided that QLB can be regenerated periodically after measurements, the size of the arrays can be reduced to a great extent. A realization of the regenerable cluster states can be seen in Ref. [33] and in general, the solid-state systems such as charge qubits or flux qubits are regenerable qubits, so we could reuse them in a fixed interval. Hence this QLB feature may be not difficult to satisfy in the future.

In summary, the general features of the QFPGA are as follows:

(1) Two kinds of components: QLBs and QRCs which have the local bus and the global bus, respectively.

(2) The memory qubits can be reused immediately.

(3) The QLB qubits can be regenerated periodically.

Based on the above features, we will present two important applications of the QFPGA in the next section.

VI. APPLICATIONS OF THE QFPGA

In this section, we will focus on two applications: construction of the general quantum gates and stimulation of the quantum Fourier transform (QFT). The purpose of the first one is to give a direct way to realize any quantum gate in the QFPGA, while the second one will show one of the important advantages of our hybrid architecture.

A. The general n-qubit quantum gate

We first realize a general three-qubit quantum gate, then extend to the n-qubit version. Using the synthesis method of Sec. II, we can decompose a general three-qubit gate into six two-qubit gates and three D_4 gates (see Fig. 14).

Thus, six QLBs and three QRCs are needed. However, if the QLB can be regenerated and memory can be reused, the three-qubit gate can be realized in a smaller gate array, such as 2×2 QLBs array (see Fig. 15). Here we roughly assume that a QLB can be regenerated after t, where t represents the

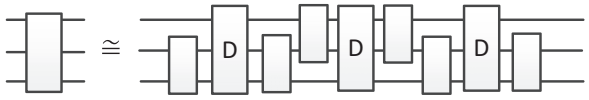

FIG. 14. The decomposition of a general three-qubit quantum gate.

running time of a QRC. As we have shown that any D_3 gate can be decomposed into three multiqubit rotation gates, so t equals eight clock cycles at least (see Sec. V A). Moreover, the local buses have been programmed in a different initial sequence of cluster states. Hence the position of inputs and outputs can be put properly to shorten bus paths, e.g., the two red lines (the two short lines across the QRC in the middle) in Fig. 15(a). As a result, we can see that the horizontal QRC is only used to transport the qubits while the vertical QRC is used to specialize in realizing D gates.

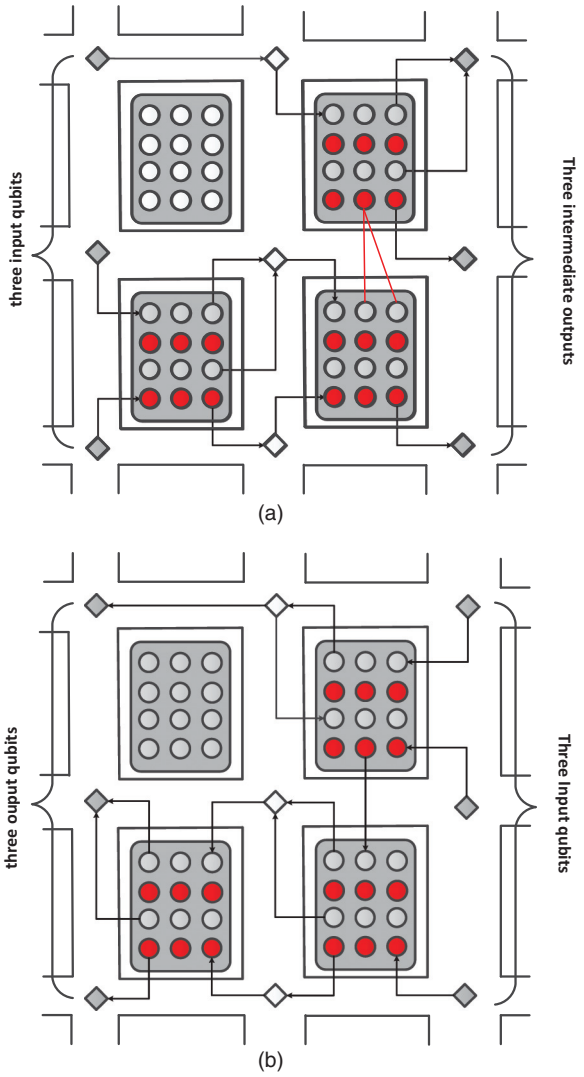

FIG. 15. (Color online) The instance of realizing a general three-qubit gate in 2×2 QLB. White circles represent channel qubits. Gray memory qubits are used as input or output qubits. Arrows denote not only the corresponding CZ operations but also the directions of the information flow. (a) The first three QLBs are used to get the intermediate result. (b) Reusing three QLBs and memory qubits to output the final result.

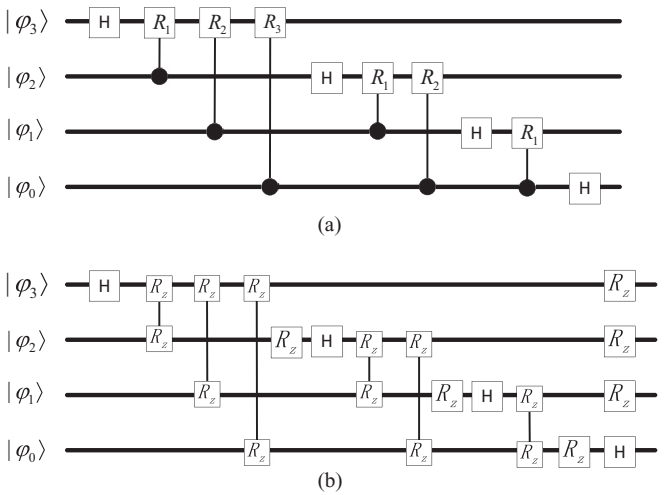

FIG. 16. (a) Quantum circuit of QFT on four qubits, and (b) the decomposition circuit of QFT-4. We exclude the necessary SWAP operations for clarity, which reverse the order of the output qubits.

Evidently, the method of this instance can be extended to realizing any n-qubit gate in a relative small-scale QFPGA architecture. However, to implement a large quantum gate, the QFPGA limitation is the requirement of a huge number of channel qubits, because a D_n gate alone needs at most $2^{n-1} - 1$ channel qubits. In a worst case of n-qubit gate ($n \geqslant 3$), the number of channel qubits is

$$\sum_{m=0}^{n-3} 3 \times 6^m \times D_{n-m} = \frac{4}{5} \times 6^{n-2} - 2^{n-2} + \frac{1}{5} \approx \theta(6^{n-2}). \tag{22}$$

For example, if $n = 10$, then we need 1 343 437 channel qubits under the worst-case condition. Some possible solutions to solve this problem in the future are to use regenerable qubits or integrate a more powerful system in the QRC which is good for the construction of multiqubit rotations.

B. Quantum Fourier transform

Quantum Fourier transform [29] is the key of many quantum algorithms such as Shor's factoring algorithm [1]. To realize QFT, firstly we simulate the quantum circuit of QFT-4 (QFT on four qubits) given in Fig. 16(a), where necessary SWAP operations are excluded for clarity. Secondly we will describe how to perform the quantum Fourier transform on any number of qubits.

As can be seen from Fig. 16(a), the QFT-4 consists of H gates and controlled-phase gates $^\wedge R_m$, where $R_m = |0\rangle\langle 0| + e^{i2\pi/2^m}|1\rangle\langle 1|$, so, each $^\wedge R_m$ is given by

$$^\wedge R_m = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i2\pi/2^m} \end{pmatrix}. \tag{23}$$

Using the result of Eq. (20), we can decompose these controlled-phase gates $^\wedge R_m$ as follows:

$$^\wedge R_m = R_{zz}^{12}\left(\frac{\pi}{2^{m+1}}\right)R_z^1\left(-\frac{\pi}{2^{m+1}}\right)R_z^2\left(-\frac{\pi}{2^{m+1}}\right). \quad (24)$$

The above equation means that one R_{zz} gate and two local rotation-z gates are needed to realize a $^\wedge R_m$ gate. As a result, the decomposition of the QFT-4 is given in Fig. 16(b). It is easy to find its MBQC version, as an H gate following an R_z gate can be realized on the two-qubit cluster states while an R_z gate alone can be realized on the three-qubit cluster states. Thus the required cluster states are shown in Fig. 17(a), where inputs and outputs should be memory qubits in the QRC. Thus 11 qubits are needed but at the cost of six additional $R_{zz}(\theta)$ gates, shown as dotted lines in Fig. 17(a). However, these two-qubit rotation gates are not expected to be realized by measurements because the qubus system is a better way to implement them; hence the QLB which combines a local bus and cluster states is an ideal candidate module for the QFT realization.

Therefore, the QLB-based realization of the QFT-4 is given in Fig. 17(b). Compared to the standard configuration of a QLB (see Fig. 12), one of the differences between them is that here we have four input memory qubits and they also serve as output qubits simultaneously, as the memory qubits can be reused again. Another difference is that now eight qubits are programmed to couple to the momentum quadrature of the bus and four qubits couple to the position quadrature of the bus. In order to realize additional R_{zz} gates efficiently, for example, the first three R_{zz} gates in Fig. 16(b), we can simultaneously couple the qubit $|\varphi_3\rangle$ to the position quadrature of the bus by operator $D(i\sigma_z\frac{\pi}{4})$ and $|\varphi_2\rangle$, $|\varphi_1\rangle$, and $|\varphi_0\rangle$ to the momentum quadrature of the bus by operators $D(-4\sigma_z)$, $D(-2\sigma_z)$, and $D(-\sigma_z)$ respectively.

Based on the above analysis, we again roughly analyzed the performance of the QFT-4 in Table II and shows that 21 clock cycles are needed to output the result.

We now show how to perform the necessary SWAP gates to get the final outputs. One advantage of the QFPGA is that we can freely choose the location of outputs, since the output qubits of a QLB are only decided by the sequence of bus operations, e.g., their characteristic values. As a result, no explicit SWAP gates are needed. As shown in Fig. 17(a), the output qubits of the QLB are qubits 3A, 4A, 5A, and

TABLE II. Performance of the QFT-4.

Measurement order	Operation cycle index	Measurement cycle index
1	1(2)	3
2A	1(15)	14
3A	14(15)	15
2B	4(5)	6
3B	4(11),16(17)	18
4A	16(17)	19
3C	7(8)	9
4B	7(11),18(19)	20
5A	18(19)	21
4C	10(13)	14
5B	12(13)	15

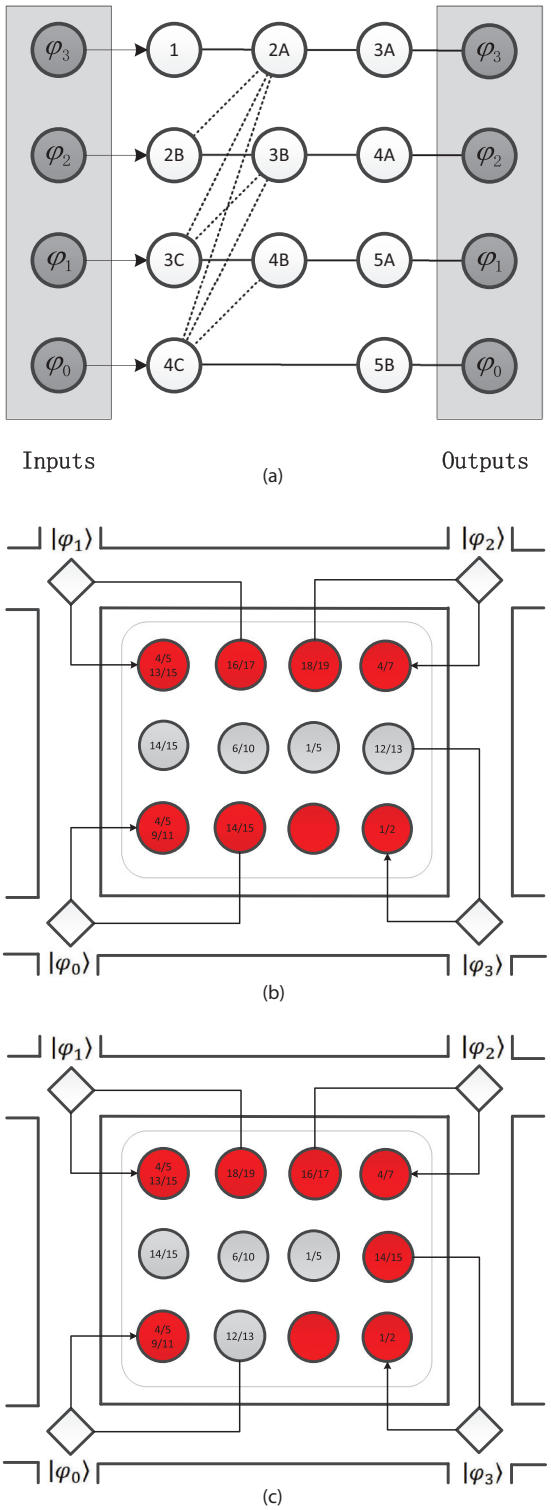

FIG. 17. (Color online) (a) The required cluster states of the QFT-4 where the number in circles represent the order of measurement (a capital letter appended after the number is to discriminate qubits which can be measured in the same time) and dotted lines represent certain controlled-phase gates between two qubits. (b) Schematic of the QLB-based realization of the QFT-4. Arrows toward the QLB represent quantum wires at the stage of inputs. (c) Schematic of the QLB-based realization of the QFT-4 where the order of outputs is not reversed.

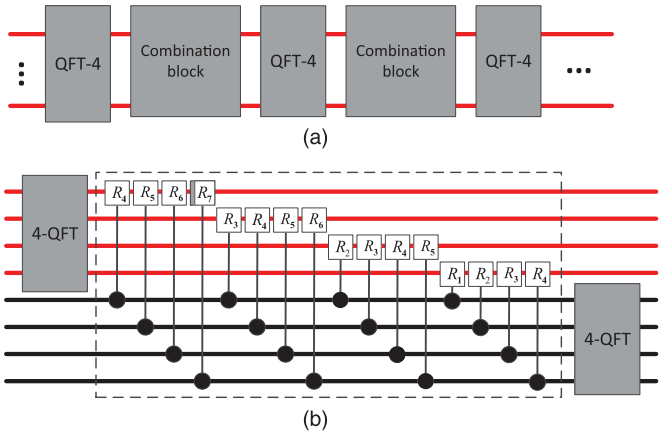

FIG. 18. (Color online) (a) Block circuit diagram simulating a general QFT-n. (b) The schematic of QFT-8 which is decomposed into two QFT-4 blocks.

5B whose corresponding characteristic values are "14/15," "16/17," "18/19," and "12/13" (see Table II). Thus, as the physical position of the output qubits of the QLB is fixed, different patterns of four characteristic values mean a different order of the outputs, such as Figs. 17(b) and 17(c) which represent the right order and origin order of the outputs, respectively. Next, we will see that the origin outputs order will be useful to build a large QFT.

The above QLB-based method can be extended to the QFT on more qubits, because it can be decomposed into a sequence of QFT-4 and combination blocks which contains sixteen $\hat{}R_m$ gates (see Fig. 18). Thus, the QFT-n will need $\lceil \frac{n}{4} \rceil$ QFT-4 and $\lceil \frac{n}{4} \rceil - 1$ combination blocks. For example, the QFT-8 can be decomposed into two cascaded QFT-4 blocks combined with sixteen $\hat{}R_m$ gates as shown in Fig. 18(b). In general if we want to realize a large QFT circuit, we should generate sixteen $\hat{}R_m$ gates in each QRC between any two QLBs which are configured as QFT-4.

Now we show how to realize these combination blocks, e.g., 16 controlled-phase gates, effectively. Recall that the interaction value we have defined in Eq. (12), and from Eq. (17) the interaction value between two qubits, 1 and 2, is $\theta_{12} = 2\beta_1\beta_2$. The method introduced in Ref. [14] tells us that if θ_{ml} can be expressed as $\beta_m\beta_l$, where there is a single value of β_m/β_l for every m/l, then each qubit is only required to couple to each quadrature of bus once to generate the whole interaction among n qubits. Consider the construction of QFT-n; if we want to link two QFT-4 blocks which are in levels j and k, respectively $(j,k = 0,1,\ldots,\lceil n/4 \rceil)$, then we can simply set values of β in the displacement operator of each qubit as follows:

$$\beta_n = \begin{cases} 2^n, & n = 0,1,2,3 \\ \frac{2\pi}{2^{4(k-j-1)+n}}, & n = 4,5,6,7 \end{cases}, \tag{25}$$

where $k > j$.

Taking these values of β, we start by coupling qubits $|\varphi_0\rangle$ through to $|\varphi_3\rangle$ to the momentum qudarature of the bus and qubits $|\varphi_4\rangle$ through to $|\varphi_7\rangle$ to the position quadrature of the bus, and then decouple all of them (see Fig. 19). This will generate all desired interacation values in Fig. 18 and require 16 bus operations only.

We can see some advantages of QFPGA. Firstly, compared to the pure MBQC method, the construction of R_{zz} gates between two QFT-4 blocks needs 16 star graph states and 16 ancilla qubits as well, while the qubus method has the significant advantage of saving the number of bus operations and no ancilla qubit is required. Secondly, compared to the pure qubus method to stimulate the QFT [14,15], our QLB-based method has the significant advantage of implementing the local unitaries. This is due to the limitations of the qubus architecture: Local unitaries cannot be created without performing local corrections [14], so all H gates and single-qubit rotation gates in Eq. (24) are difficult to create by the qubus alone. Moreover, provided one can perform local corrections, then another disadvantage of qubus is that local unitaries are expensive, with each operations requiring 14 bus operations. As a result, the authors in Ref. [15] assumed they have the ability to perform arbitrary local unitaries, and in Ref. [14] they resorted to a classical computer to perform a single correction for each set of qubits. However, there are no such problems in our architecture, as we use a "local" bus to generate the desired QLB which is good at local unitaries but at the cost of many ancilla qubits.

VII. CONCLUSION AND OUTLOOK

We have proposed a programmable architecture called QFPGA for universal quantum computing, which consists of QLBs and QRCs. As any large unitary gate can be decomposed into a sequence of two-qubit gates and D gates, they will be realized in QLBs and QRCs, respectively. The QLB consists of short-lived qubits while the QRC has ancilla channel qubits and memory qubits with long enough coherence time. The reason for this choice is that for a D_n gate, it often requires $\theta(2^{n-1})$ multiqubit rotation z gates which takes much more time to realize for a large D_n gate. Compared with previously published work related to MBQC, the QFPGA architecture combines small blocks of cluster states by using global buses,

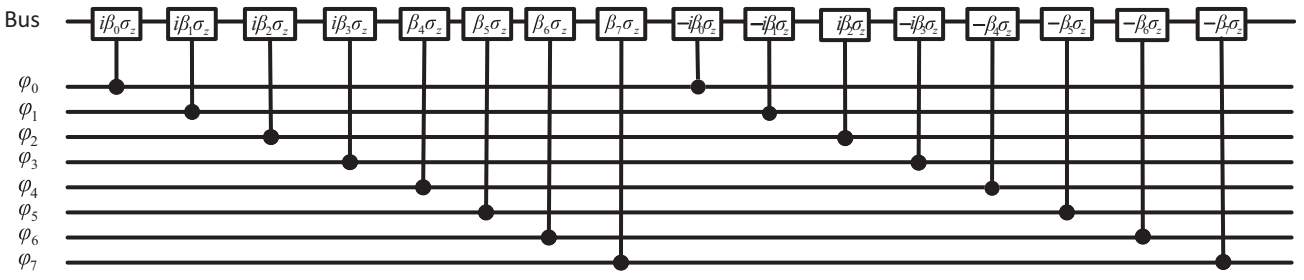

FIG. 19. Only 16 displacement operators are needed to realize the 16 $\hat{}R_m$ gates in the QFT.

so as to make the overall structure more scalable, flexible, and also clear. On the other hand, we not only take advantage of the concept of quantum memory to drive the sequential MBQC, but we also integrate small modules to simplify the operation which is complex in the original paper [28]. Hence our model is more effective but at a cost of some level of decoherence error. To guarantee an acceptable error, we have described an efficient method for generating cluster states in the QLB, each of which is generated with a single, reused bus of the qubus system. As every QLB has 12 qubits, our analysis shows that the error probability of both QLB and QRC is below an acceptable threshold under certain conditions. For multibus dynamic schemes, this means that the QFPGA approach is feasible, as fully scalable operation can be achieved. Furthermore, by realizing two scalable applications, the general n-qubit quantum gate and the QFT-n, we have shown the advantages of this architecture.

The main obstacle to wide adoption of QFPGAs is the need for a large number of qubits to drive the computation. However, in a solid-state quantum system, qubits should be able to regenerate effectively in a fixed interval so as to overcome this problem. It should be noted that the proposed QFPGAs do not force one to use MBQC or qubus only; rather they provide an architecture to integrate a number of quantum computing models for efficient quantum logic construction. A QFPGA only requires certain fault-tolerant quantum models to operate as QLBs and QRCs. Thus, it promises to be extremely useful for the many-qubit quantum computation in the future.

ACKNOWLEDGMENTS

This research is supported by National Natural Science Foundation of China (Grants No. 61131001, No. 61171011) and National 863 Program of China (Grant No. 2009AA012201).

APPENDIX: QSD OF TWO-QUBIT CS GATE

The two-qubit CS gate has a general form,

$$\begin{pmatrix} \cos\frac{\theta_0}{2} & 0 & \sin\frac{\theta_0}{2} & 0 \\ 0 & \cos\frac{\theta_1}{2} & 0 & \sin\frac{\theta_1}{2} \\ -\sin\frac{\theta_0}{2} & 0 & \cos\frac{\theta_0}{2} & 0 \\ 0 & -\sin\frac{\theta_1}{2} & 0 & \cos\frac{\theta_1}{2} \end{pmatrix}. \tag{A1}$$

Then we can exchange the rule of two qubits, i.e., change the matrix to the form $R_y(\theta_0) \oplus R_y(\theta_1)$,

$$\begin{bmatrix} R_y(\theta_0) & 0 \\ 0 & R_y(\theta_1) \end{bmatrix}. \tag{A2}$$

By the rule of Shannon decomposition (see Sec. II) $R_y(\theta_0 + \theta_1) = R_y(\theta) = VD^2V^\dagger$, then comparing this equation to the identity $R_x(\pi/2)R_z(-\theta)R_x(-\pi/2) = R_y(\theta)$, we can easily get the relations $V = R_x(\pi/2)$ and $D = R_z(-\theta/2)$.

The "W" operator can also be extracted by the following equation:

$$\begin{aligned} W &= DV^\dagger R_y(\theta_1), \\ &= R_z\left(-\frac{\theta}{2}\right) R_x\left(-\frac{\pi}{2}\right) R_x\left(\frac{\pi}{2}\right) R_z(-\theta_1) R_x\left(-\frac{\pi}{2}\right), \\ &= R_z\left(-\frac{\theta_0 + 3\theta_1}{2}\right) R_x\left(-\frac{\pi}{2}\right). \end{aligned} \tag{A3}$$

Commute R_z and D gate and we obtain the desired decomposition of Fig. 3.

[1] P. Shor, SIAM J. Comput. **26**, 1484 (1997).
[2] A. M. Childs, A. J. Landahl, and P. A. Parrilo, Phys. Rev. A **75**, 032335 (2007).
[3] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda, Science **292**, 472 (2001).
[4] A. W. Harrow, A. Hassidim, and S. Lloyd, Phys. Rev. Lett. **103**, 150502 (2009).
[5] D. Bacon and Wim v. Dam, Commun. ACM **53**, 84 (2010).
[6] S. Salek, F. Seifan, and E. Kashefi, Electron. Notes Theor. Comput. Sci. **270**, 155 (2011).
[7] R. Raussendorf, D. E. Browne, and H. J. Briegel, Phys. Rev. A **68**, 022312 (2003).
[8] R. Raussendorf and H. J. Briegel, Phys. Rev. Lett. **86**, 5188 (2001).
[9] A. Broadbent and E. Kashefi, Theor. Comput. Sci. **410**, 2489 (2009).
[10] D. E. Browne and H. J. Briegel, arXiv:quant-ph/0603226.
[11] B. Coecke, Contemp. Phys. **51**, 59 (2010).
[12] A. Sehrawat, D. Zemann, and B. G. Englert, Phys. Rev. A **83**, 022317 (2011).
[13] J. Anders, D. K. L. Oi, E. Kashefi, D. E. Browne, and E. Andersson, Phys. Rev. A **82**, 020301 (2010).
[14] K. L. Brown, Ph.D. Thesis, University of Leeds, 2011.
[15] K. L. Brown, S. De, V. M. Kendon, and W. J. Munro, New J. Phys. **13**, 095007 (2011).
[16] P. van Loock, W. J. Munro, K. Nemoto, T. P. Spiller, T. D. Ladd, S. L. Braunstein, and G. J. Milburn, Phys. Rev. A **78**, 022303 (2008).
[17] C. Horsman, K. L. Brown, W. J. Munro, and V. M. Kendon, Phys. Rev. A **83**, 042327 (2011).
[18] M. A. Nielsen and I. L. Chuang, Phys. Rev. Lett. **79**, 321 (1997).
[19] J. Anders and D. E. Browne, Phys. Rev. Lett. **102**, 050502 (2009).
[20] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAs* (Kluwer Academic Publishers, Dordrecht, 1999).
[21] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Phys. Rev. A **52**, 3457 (1995).
[22] M. Mottonen, J. J. Vartiainen,V. Bergholm, and M. M. Salomaa, Phys. Rev. Lett. **93**, 130502 (2004).
[23] G. Vidal and C. M. Dawson, Phys. Rev. A **69**, 010301 (2004).
[24] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. (Johns Hopkins University Press, Baltimore, MD, 1996).
[25] C. C. Paige and M. Wei, Linear Algebra Appl. **208**, 303 (1994).
[26] V. V. Shende, S. S. Bullock, and I. L. Markov, IEEE Trans. CAD **25**, 1000 (2006).

[27] A. Daskin, A. Grama, G. Kollias, and S. Kais, J. Chem. Phys. **137**, 234112 (2012).
[28] A. J. Roncaglia, L. Aolita, A. Ferraro, and A. Acin, Phys. Rev. A **83**, 062332 (2011).
[29] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2000).
[30] C. Lomont, arXiv:quant-ph/0307111.
[31] T. P. Spiller, K. Nemoto, S. L. Braunstein, W. J. Munro, P. van Loock, and G. J. Milburn, New J. Phys. **8**, 30 (2006).
[32] Y. Nakamura, Y. A. Pashkin, and J. S. Tsai, Nature (London) **398**, 786 (1999).
[33] S. J. Devitt, A. G. Fowler, A. M. Stephens, A. D. Greentree, L. C. L. Hollenberg, W. J. Munro, and K. Nemoto, New J. Phys. **11**, 083032 (2009).