Resource reduction in multiplexed high-dimensional quantum Reed-Solomon codes

Shin Nishio,[1,2,3,*] Nicolò Lo Piparo,[2,3] Michael Hanks,[4] William John Munro,[5,3] and Kae Nemoto[2,3,1,†]

[1] *Department of Informatics, School of Multidisciplinary Sciences, SOKENDAI (The Graduate University for Advanced Studies), 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan*

[2] *Quantum Information Science and Technology Unit, Okinawa Institute of Science and Technology Graduate University, Onna-son, Kunigami-gun, Okinawa 904-0495, Japan*

[3] *National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan*

[4] *QOLS, Blackett Laboratory, Imperial College London, London SW7 2AZ, United Kingdom*

[5] *NTT Basic Research Laboratories & NTT Research Center for Theoretical Quantum Physics, NTT Corporation, 3-1 Morinosato-Wakamiya, Atsugi, Kanagawa 243-0198, Japan*

(Received 15 June 2022; revised 18 October 2022; accepted 15 March 2023; published 29 March 2023)

Quantum communication technologies will play an important role in quantum information processing in the near future as we network devices together. However, their implementation is still a challenging task due to both loss and gate errors. Quantum error-correcting codes are one important technique to address this issue. In particular, the quantum Reed-Solomon codes well suited to communication tasks as photons can naturally carry more than one qubit of information. The high degree of physical resources required, however, makes such a code difficult to use in practice. A recent technique called quantum multiplexing has been shown to reduce resources by using multiple degrees of freedom of a photon. In this work, we propose a method to decompose multicontrolled gates using fewer controlled-x (CX) gates via this quantum multiplexing technique. We show that our method can significantly reduce the required number of CX gates needed in the encoding circuits for the quantum Reed-Solomon code. Our approach is also applicable to many other quantum error-correcting codes and quantum algorithms, including Grovers and quantum walks.

DOI: 10.1103/PhysRevA.107.032620

I. INTRODUCTION

Quantum communication systems utilizing the principles of superposition and entanglement will provide new capabilities that we cannot achieve in our current telecommunication counterparts [1–5]. Such communication is expected to provide the basis for distributed quantum information processing systems, including quantum key distribution [6,7], blind quantum computation [8,9], distributed quantum computation [10], quantum remote sensing [11], and the quantum internet [12]. However, the fragile nature of quantum states as they propagate through such channels will severely limit the performance of the systems. Quantum error-correcting codes (QECCs) are a fundamental tool to address these issues as they can correct both channel loss and gate/general errors.

Since the first QECC was found by Shor [13], quite a number of codes have been proposed, including stabilizer codes [14], quantum low-density parity-check codes [15], and Gottesman-Kitaev-Preskill (GKP) codes [16]. Further, the Calderbank-Shor-Steane (CSS) codes [17] are widely studied because they are quantum codes that can be constructed using a variety of classical codes and inherit the nice properties of the existing classical codes. For quantum computation, topological CSS codes, including surface codes [18], have known useful properties such as high thresholds and capability for performing practical transversal gates due to the locality of stabilizers. For quantum communication and quantum memories, the delay introduced in the decoding is not as critical as it is in computation [19], meaning complex decoding circuits are acceptable compared to those typically used in the computation. Therefore, for quantum communication and quantum memory-related tasks, code rate and minimum distance are more important than the locality of the stabilizers. It is known that the quantum Reed-Solomon codes [20] are efficient codes for quantum communication from that point of view. As research on QECCs becomes more active, so does research on efficient implementations of encoding [21,22] and decoding circuits [23,24] for QECCs.

The classical Reed-Solomon (RS) code [25] has been used in various communication systems [26,27] due to it being a maximum distance separable (MDS) code [28]. MDS codes are important because they can detect and correct the greatest number of errors for a fixed code-word length n and message length k. In the classical RS code, multiple consecutive bits correspond to an element of the Galois field (GF) [25]. This allows that code to be particularly resilient to burst errors (errors occurring on consecutive bits) [26], where multiple bits are used to represent a single element of the GF. Now the quantum variant of these codes known as the quantum Reed-Solomon (QRS) code [20] is an important CSS code due to its ability to correct qudit loss errors, which are the most critical issue in long-distance quantum communication [29]. However, quantum error-correcting codes require significant physical sources (in terms of the numbers of both qubits and photons in the communication) for their realization, making their implementation quite challenging. It is well known that

[*] parton@nii.ac.jp
[†] nemoto@nii.ac.jp

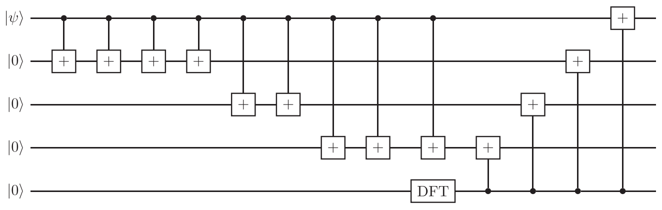


FIG. 1. The encoding circuit for the $[[5, 1, 3]]_5$ quantum Reed-Solomon code using five-dimensional qudits. The two qudit SUM gates are defined by the black dot representing the control and the target with a square box containing the "+" symbol. The discrete Fourier transform gate is represented by the DFT labeled box.

a photon can carry multiple qubits encoded in its degrees of freedom (DOF), such as polarization and time bin. Two-qubit gates can be applied to those DOFs in a deterministic fashion using simple optical elements, such as beam splitters and optical switches. Recently, a technique that exploits those DOFs, called quantum multiplexing, has been used to reduce the number of quantum memories in a purification protocol [30] and the number of physical resources [31,32] in the redundancy code and small-scale QRS codes. Here we show that by using such quantum multiplexing techniques, one can drastically reduce the number of controlled gates, in particular the controlled NOT (CX) gate required to implement the encoding circuit of QRS codes used in quantum communication tasks.

Our paper is organized as follows. In Sec. II, we will review the construction of QRS codes and their coding circuits. We then briefly overview in Sec. III the quantum multiplexing technique and propose a method to implement multiple controlled gates (C_kX gates) using fewer CX gates enabled by quantum multiplexing. Section IV then shows the reduction in the number of gates required that can be achieved when quantum multiplexing is applied to the QRS codes encoding circuit. The results and applications of our method to other quantum tasks are discussed in Sec. V.

II. THE QUANTUM REED-SOLOMON CODE

Let us begin with a brief review of the quantum Reed-Solomon code. The $[[d, 2K-d, d-K+1]]_d$ QRS code [20,33–35] is defined by the CSS construction [17] of the classical RS code in which d (a prime number) is the dimension of the qudits used to encode the logic states. Then, $2K-d$ is the number of logical qudits encoded into the code, while $d-K+1$ is the minimum distance of the code. In our work, we will consider the $2K-d=1$ case (meaning we are only encoding one qudit), which has the ability to retrieve the original encoded quantum information when $(d-1)/2$ or fewer physical qudits are lost. For qudit computation with d being a prime number greater than or equal to 5, the universal set of quantum gates is {DFT, Q[i], SUM} [36], which are analogous to the {H, T, CX} gates for qubit computation. The DFT gate can be considered the generalized Hadamard gate, while the SUM gate is the generalized CX gate on the qudit.

Now the encoding quantum circuit for the QRS code (illustrated in Fig. 1 for the $[[5, 1, 3]]_5$ code) can be implemented using the discrete Fourier transform (DFT) gate [37,38] and

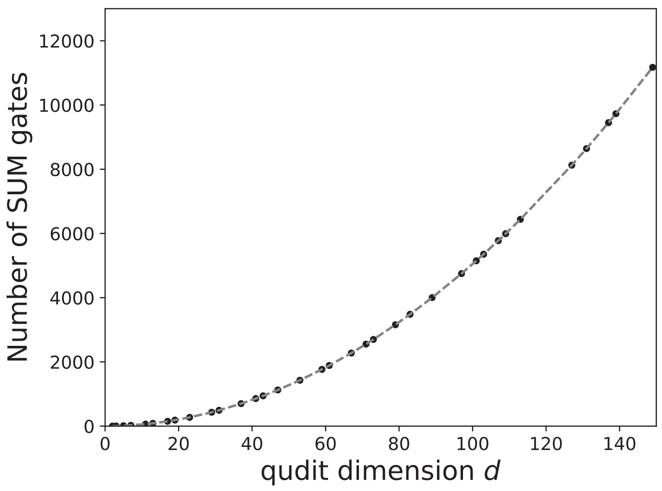


FIG. 2. Required number of SUM gates for encoding the $[[d, 1, (d+1)/2]]_d$ QRS codes versus qudit dimension d. The dots represent the prime values of d.

a series of quantum SUM gates [16]. Noting that the DFT gate is the generalization of the Hadamard gate, this gate, when applied to a single qudit, creates a superposition of states given by

$$\mathrm{DFT}(|j\rangle) = \frac{1}{\sqrt{d}} \sum_{k=0}^{d-1} e^{2\pi i(jk/d)} |k\rangle ,$$

where $|k\rangle$ is the kth Fock state while $|j\rangle$ is the jth phase state. Next, the SUM gate is the generalization of the CX gate for d-dimensional qudit and is given by

$$\mathrm{SUM}(|A\rangle\, |B\rangle) = |A\rangle\, |(A+B)\, [\mathrm{mod}\, d]\rangle ,$$

where A and B are integers less than or equal to $d-1$. This gate is essentially a modulo adder. To illustrate how these operations can be used to establish the encoding circuit, Fig. 1 shows an example of the encoding circuit for the $[[5, 1, 3]]_5$ code. This encoding circuit construction is a higher-dimensional generalization of the encoding circuit for $[[3, 1, 2]]_3$ QRS codes proposed in Ref. [39]. Further, in Appendix A we have described the details of the circuit configuration for the general $[[d, 1, (d+1)/2]]_d$ code. It is important for performance and efficiency comparisons to establish the number of gates required to implement these codes. It is straightforward to show (with details given in Appendix A) that the number of SUM gates required to create the $[[d, 1, (d+1)/2]]_d$ code using a d-dimensional qudit is $(d^2+d-4)/2$, which we plot in Fig. 2. This number increases quadratically with the qudit dimension, but it is important to note that the number of CX gates used in each SUM gate also increases with the qudit dimension (as we will show later), making the implementation of the higher-dimensional QRS codes more difficult. In order to show the advantage of applying the technique of quantum multiplexing, it is more convenient to encode each logical d-dimensional qudit with k qubits, where $2^{k-1} < d < 2^k$.

There are several ways to implement SUM gates, but with this insight that is effectively a modulo adder, we will decompose our SUM gate into two parts. The first part is an adder

TABLE I. The number of gates $N_{\mathrm{RCA}}(N_{\mathrm{M}})$ required for the RCA (modulo) operations in the SUM gate, respectively. Here $\mathrm{H_D}(a, b)$ is a function that returns the Hamming distance between the binary representations of the input vector a and b. See Appendix B for further details about the $\mathrm{H_D}(a, b)$ functions in the C_1X gates.

	$\mathrm{C}_{k+1}\mathrm{X}$	$\mathrm{C}_k\mathrm{X}$	C_2X	C_1X	Condition
N_{RCA}			$3k-2$	$2k-1$	
N_{M}		$d-1$		$\sum_{i=d}^{2(d-1)} \mathrm{H_D}(i, i\,[\mathrm{mod}\, d])$	$2(d-1) \leqslant 2^k$
	$2d-2^k$ -1	2^k-d		$\sum_{i=d}^{2(d-1)} \mathrm{H_D}(i, i\,[\mathrm{mod}\, d])$	$2(d-1) > 2^k$

circuit, of which the Ripple carry adder (RCA) [40] is a natural choice. That circuit performs the following transformation:

$$\mathrm{RCA}(|A\rangle\,, |B\rangle) = |A\rangle\, |(A+B)\,[\mathrm{mod}\, 2^k]\rangle\,,$$

while the second part is a modulo conversion that performs the following transformation:

$$\mathrm{Mod}(|A\rangle\,, |(A+B)\,[\mathrm{mod}\, 2^k]\rangle) = |A\rangle\, |(A+B)\,[\mathrm{mod}\, d]\rangle\,.$$

In this case we use k qubits to encode d-dimensional qudits, which means the RCA part adds two binary numbers having k bits. This part utilizes auxiliary qubits called “carry” to store the outcome of the addition of two qubits both having values of 1. Further, the modulo conversion can also be split into two distinct operations. In the first operation, a series of logic gates are used to check whether the outcome of the RCA exceeds d. In the case it exceeds d, the result will be stored in an auxiliary qubit we label “check if.” To illustrate this, a specific example of how these auxiliary qubits have been used is presented in Appendix B. Next, the second operation, which we label as a conversion element, transforms the output of the RCA part in $[\mathrm{mod}\, 2^k]$ representation to the desired output of the SUM gate with $[\mathrm{mod}\, d]$ representation. We need to consider this in a little more detail, so let us look at the various outcomes of the RCA, which correspond to our three different cases. In the first case, where the outcome of the RCA is less than d, the second operation will not change this outcome, meaning no further action is needed. In the second case, where the outcome is greater than $d-1$ but less than 2^k, check-if auxiliary qubits are used for storing each value. Those values are then used to convert the outcome to the desired state. In the third case, where the outcome is greater than or equal to 2^k, check-if auxiliary qubits will be used to store each value, and the biggest carry qubit is also used to distinguish the value and the value minus 2^k in the second operation. However, the check-if qubit is not required when the outcome is 2^k if and only if $2(d-1) = 2^k$. This is because the biggest can be used as the check-if value since 2^k is the only value that is bigger than 2^k. It is straightforward to observe that this SUM gate construction requires $k+d-2$ auxiliary qubits, where k qubits are for the carry and $d-2$ qubits are for the check if.

The total number of CX gates N_{SUM} for each SUM gate in a d-qudit system with $2^{k-1} < d \leqslant 2^k$ is given by $N_{\mathrm{SUM}} = N_{\mathrm{RCA}} + N_{\mathrm{M}}$, where $N_{\mathrm{RCA}}(N_{\mathrm{M}})$ are the number of gates for the RCA (modulo) parts, respectively. The number of gates required by those parts is presented in Table I, where $\mathrm{C}_k\mathrm{X}$ is the X gate controlled by k qubits ($\mathrm{C}_2\mathrm{X}$ is controlled by 2 and is the well-known Toffoli gate). Further, the $\mathrm{C}_{k+1}\mathrm{X}$ gate is the X gate controlled by k qubits + 1 carry auxiliary qubit.

In Fig. 3 we plot N_{RCA} versus d, and since the RCA part is a simple k bit ripple carry adder, N_{RCA} depends only on integer values of k for $2^{k-1} < d < 2^k$, which is shown as the gray line. When d is sufficiently large, N_{RCA} can be negligibly small compared to N_{M}, and so N_{SUM} is dominated by N_{M}. In Appendix B, details of the modulo adder implementation are described and we show how N_{RCA} is explicitly determined.

In the modulo conversion part, the C_kX gate can be decomposed into $4(k-2)$ $\mathrm{C_2X}$ gates and the $\mathrm{C}_{k+1}\mathrm{X}$ gate can be decomposed into $4(k-1)$ $\mathrm{C_2X}$ gates as shown in Ref. [41]. Further, the $\mathrm{C_2X}$ gate can be decomposed into 6 CX, 2 H, 3 $\mathrm{T}^{\dagger}$, and 5 T gates. We will refer to this decomposition as a “general decomposition” and compare it to our more efficient multiplexing decomposition later on. In the next section, we will show that applying quantum multiplexing can drastically reduce the number of CX gates required to implement these circuits. The details of the modulo adder implementation are described in Appendix B.

III. QUANTUM MULTIPLEXING

It is useful to begin by reviewing the concept of quantum multiplexing and how it can be used to save resources. The key initial insight is that a single photon is able to encode more than a single qubit of information onto it. The photon’s multiple degrees of freedom can be used for this, as well as higher-dimensional encoding available within many of those degrees of freedom.

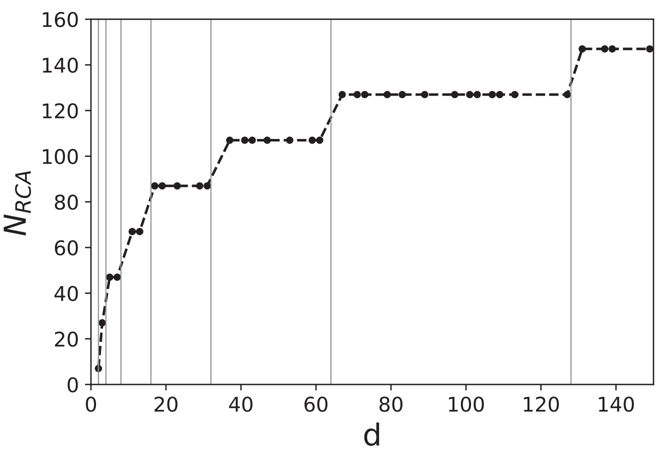


FIG. 3. The number of CX gates N_{RCA} required to perform the RCA part of the SUM gate for d-dimensional qudits. The vertical gray lines correspond to the 2^m integer values, while the dotted lines between data points are used as a visual guide for the eye.

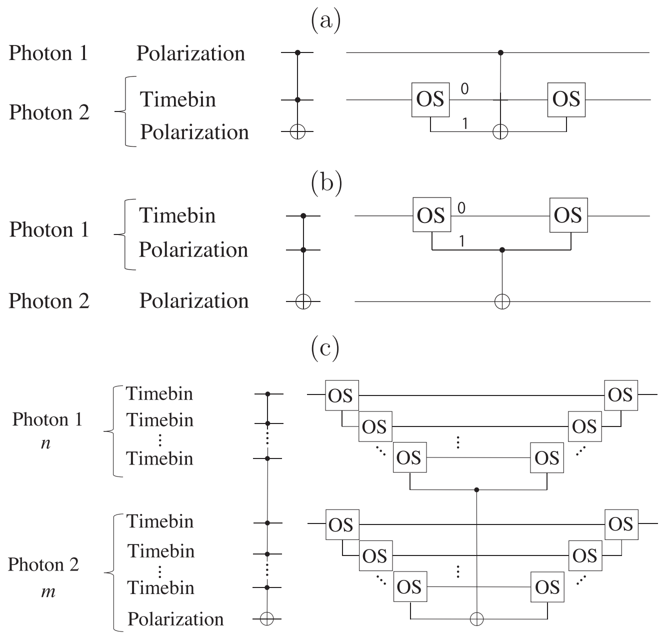


FIG. 4. (a), (b) Quantum circuit showing the C_2X gate between two photons (left) and a multiplexing circuit in which the C_2X gate is realized by an optical switch (OS) and a CX gate (right). (c) The $C_{n+m-1}X$ gate between two photons (left) and its optical implementation (right).

Within one photon where two qubits are encoded in one time-bin and one polarization degree of freedom, a controlled unitary between the time-bin qubit (as control) and polarization qubit (as target) can be simply implemented using an optical switch to convert the time-bin degree of freedom to spatial-mode degrees of freedom. A unitary operation on the polarization degree of freedom in the $|1\rangle$ basis state spatial mode (that corresponded to the $|1\rangle$ basis state of the time bin) followed by the use of a second optical switch (to convert the spatial encoding back to time-bin encoding) finishes the implementation of the controlled unitary gate. While no direct two-qubit gate needed to be implemented, two optical switches were required instead. This approach can be easily extended to multiple control gates within that single photon as well. If the multiple control qubits are all encoded in time bins, multiple optical switches can be used to convert all those time-bin qubits into spatial modes. Our polarization unitary is then applied on the spatial mode corresponding to all control qubits in their $|1\rangle$, followed by further optical switches to convert the spatial modes back to time bin. Overall, two optical switches are needed per time-bin qubit.

So far our discussion has only involved a single photon, but this procedure can be applied to multiple photons as well [41]. Here a direct two-qubit gate would be required. It has been shown that the Toffoli gate can be decomposed into a single CX gate and optical switches by splitting one DOF of a photon into two separate spatial modes and applying a CX gate directly on one mode [39], as shown in Fig. 4(a). Consider another situation of a Toffoli gate (controlled NOT, C_2X), where two of the qubits are present on one photon (one time-bin and one polarization qubit) and the remaining qubit (the target polarization qubit, say) on the second photon [as shown in Fig. 4(b)]. With the conventional set of quantum gates, six CX gates would have been required. However, with multiplexing these changes, we can use an optical switch to convert the time-bin qubit into a spatial mode qubit, with one mode each corresponding to the $|0\rangle$ ($|1\rangle$) time-bin basis states. A single CX gate between the polarization qubit in the $|1\rangle$ spatial mode of photon 1 and the polarization qubit on photon 2 then implements our desired Toffoli gate. An optical switch is needed on the spatial modes after that gate to convert it back to its time-bin encoding. What we immediately notice is that our Toffoli gate now requires one CX gate and two optical switches—a significant improvement, assuming that the optical switches are more efficient than control gates.

This result can obviously be generalized in a natural way to the situation when the first photon has n control qubits while the second photon has m control qubits and one target qubit and we want to implement an $C_{n+m-1}X$ gate [as shown in Fig. 4(c)]. For convenience, we will assume that the control qubits are time-bin encoded and the target qubit polarization (more on this later). In this case, $n-1$ time-bin qubits for the first photon and m time-bin qubits for the second photon are converted to a spatial encoding (using $n-1$ OS on the first photon and m OS on the second). A CX gate is then applied between the last time-bin qubit on the first photon and the polarization qubit in the second on the spatial modes associated with the $n+m-1$ control qubits all being in the $|1\rangle$ state. After the CX gate, OS is then used to convert the spatial encoding back to a time-bin one (using $n+m-1$ of them). Overall, one observes that the $C_{n+m-1}X$ gate requires $2(n+m-1)$ OS and one CX gate. We have proved that we can substitute C_kX gates with a single CX gate alongside a number of optical switches by using the induction principle shown in Appendix D. Of course, the key issue becomes how good the optical switches are, and this is where we need to be careful. These switches need to be fast and low loss. Currently, this is an issue, but there is a significant effort going into the development of such devices, as they have many classical applications. Further current OS are still much more efficient than optical control gates.

In our examples so far, we have focused on the controlled operations occurring for time-bin encoded qubits. This does not have to be the case. Other degrees of freedom, including frequency or angular orbital momentum (AOM), are alternatives. In that case, the function of the optical switch is to convert that encoding to spatial modes.

IV. PERFORMANCE

Our focus now will be to compare the number of gates required to encode a QRS code versus its dimension when single-mode photons and multiplexed photons are used. The blue (red) curve of Fig. 5(a) shows the total number of CX gates required to implement a single SUM gate for a d-dimensional QRS code when single-mode (multiplexed) photons are used. The blue curve increases rapidly with the dimension of the code reaching 21 182 CX gates required for $d=139$, whereas the red curve increases modestly with the code dimension (using only 1049 at $d=139$). Thus, applying quantum multiplexing has resulted in a significant reduction in

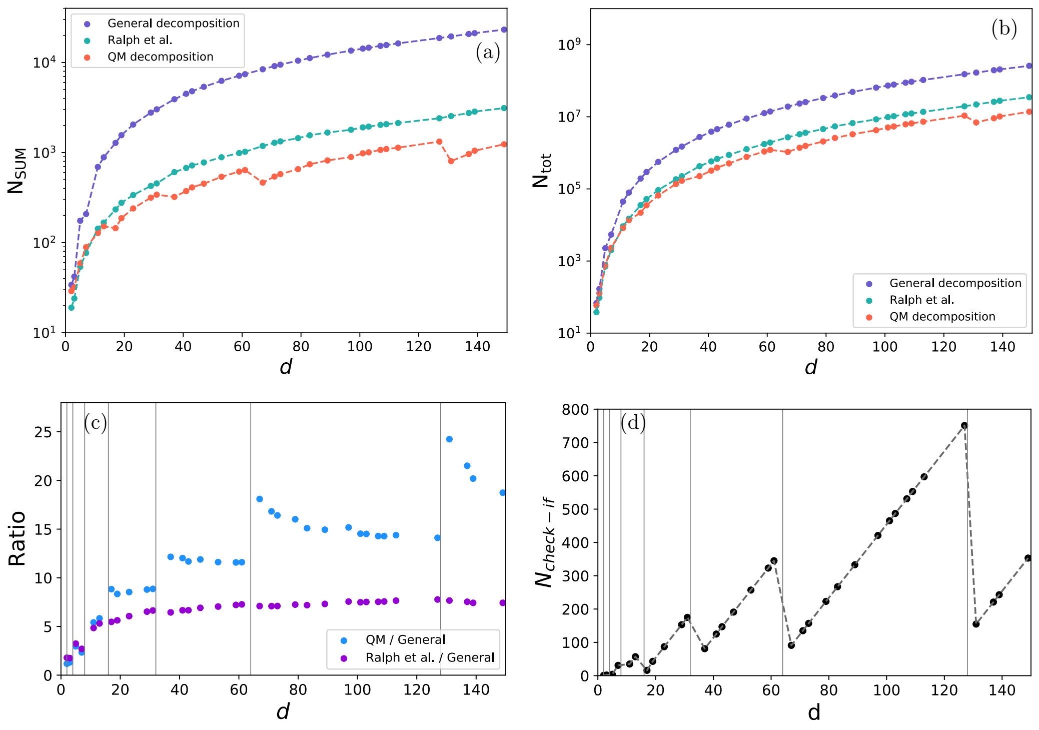


FIG. 5. (a) Plot of the required number of CX gates N_{SUM} for the construction of a single SUM gate versus prime number d, the size of the qudit. The blue curve is based on the general decomposition [41] while the magenta curve is based on Ralph *et al.* [42] and the red curve uses the multiplexed decomposition shown in Fig. 4. (b) Plot of the total number of CX gates N_{tot} for constructing the whole encoder of the $[[d, 1, (d+1)/2]]_d$ QRS code versus d. (c) The ratio R between the blue curve and the magenta (red) curve plotted in (a), respectively. The vertical gray lines correspond to 2^m for integer values of m. (d) Plot of the required number of CX gates $N_{\text{check-if}}$ for the check-if part versus d using quantum multiplexing.

CX gates. Moreover, for the multiplexing case, at $d = 67$ and $d = 137$ the number of CX gates is slightly less than $d = 61$ and $d = 131$, respectively. Although it seems an inconsistent result, it can be explained by looking at the second case of Table I. The number of C_kX gates and the number of $C_{k+1}X$ gates are present in the check-if part of the modulo conversion scheme. They reduce to CX and C_2X gates when multiplexing is applied. Therefore, the total number of CX gates will be much more affected by $N_{C_{k+1}X}$ than by N_{C_kX}. These numbers depend on the number of qubits k used to encode the states as well as on the dimension of the code, d. When $d \lesssim 2^k$, we require several check-if auxiliary qubits to implement the modulo addition, meaning $N_{C_{k+1}X}$ will be high. On the other hand, when $d \ll 2^k$, N_{C_kX} will be the preponderant term, as we can now use more qubits for the sum and fewer check-if qubits. Although it is not clear from Fig. 5(a), we also have that $N_{C_{k+1}X}(d = 31) > N_{C_{k+1}X}(d = 37)$. This behavior is more evident in Fig. 5(b), which shows the total number of CX gates required to construct the whole encoder. The gate reduction by quantum multiplexing is also significant in the overall cost of the encoding circuit. Regardless of the value of d, we always require a number of conversion gates proportional to the code dimension. Thus, the number of conversion gates represented by the Hamming distance in Table I is not a relevant term for determining the behavior.

Next, Fig. 5(c) shows the ratio of N_{SUM} of the nonmultiplexed case (blue) over the multiplexed case (red). It allows us to quantify the improvement we have. Here we can see that the curve jumps to a much higher ratio value when it crosses the gray lines, which corresponds to 2^k, with integer $k = 2 \ldots 7$. We label the prime numbers in which this happens by d_{cross}. At $d > d_{\text{cross}}$ the ratio decreases as shown in regions $d = 32$–63, 64–127, and 128–255) of Fig. 5(c). This is due to the higher number of C_2X required as d increases, as already explained in the previous paragraph. However, at $d > 19$ the ratio increases slightly until $d = 31$ due to the fact that for this range of d the number of check if increases only slightly, as shown in Fig. 5(d). We also compare our results with the ones obtained by Ref. [42], shown by the magenta curves in Fig. 5. In Ref. [42], the authors realize C_kX gates by introducing one k-dimensional qudit, $2k - 1$ two-qubit gates, and single-qudit gates they refer to as X_a and X_b. Our results still show an advantage compared to the ones proposed in Ref. [42] in terms of gate reductions. Furthermore, the system proposed in Ref. [42] requires special gates X_a and X_b, which might require more two-qubit gates when qubits representation is in use. We also determined that the encoding circuit for $[[d-1, d-2t, 2t+1]]_d$ QRS codes where $d = 2^m$ for integers m does not require Toffoli or C_kX gates, hence quantum multiplexing does not give any advantage in terms of gates reduction for these special cases. We show in Appendix C an example of an implementation of a QRS encoding circuit over GF(2^m).

There is a well-known method [21] for constructing efficient encoding circuits for stabilizer codes, but this method requires multiple-target gates but not multiple-controlled gates. Therefore, the proposed method cannot be directly applied to such encoding circuits. The proposed method can be applied to the encoding circuit of quantum error-correcting codes using multiple controlled gates.

V. CONCLUSION AND DISCUSSION

In this work we applied the quantum multiplexing technique to the encoding of a d-dimensional QRS code ($[[d, 1, (d+1)/2]]_d$) to reduce the number of physical resources required for its implementation. We first evaluate the number of qudit gates that allows us to create an arbitrary code word of the QRS code. We observe that this number increases quadratically with the code dimension. We then encode the d qudits of a conventional QRS using qubits. This allows decomposing the main logic gate for the encoding, which is a SUM gate, into a number of C_kX gates having k controls. These gates can be further decomposed into more elementary CX gates using a general decomposition or into a lower number of CX gates using the Ralph decomposition. We then apply our quantum multiplexing method to the C_kX gates by splitting multiple qubits of the time-bin DOF of a photon into an equivalent number of spatial modes. Thus, we can apply a single CX gate between the relevant split mode of a multiplexed photon and the DOF of another photon. Finally, we compare the total number of CX gates required for the general decomposition [41], for the Ralph method [42], and when quantum multiplexed is in use versus the dimension of the QRS code. We observe that using multiplexing allows reducing drastically the number of CX gates compared to the other two methods. When $d = 131$ the improvement is expected to be more than 24 times better compared to the general decomposition, and approximately three times better than the Ralph method. Also, our method is more effective for the case when the qudit dimension d is close to 2^k, where d is a prime number and k is greater than 6. Applying quantum multiplexing requires optical switches that might affect the performance of the system due to imperfection. However, optical switches are a fundamental tool for quantum computation, and we are confident that the efficiency of such systems will reach fault tolerance levels (99%+). The biggest challenge for large-scale quantum information processing in the near future will be errors, and the use of quantum error-correcting codes may partially solve this problem. QRS codes are expected to be an effective countermeasure against photon loss in quantum communication channels. Therefore, our work proposed an implementation method of the encoding circuit for QRS codes.

The quantum multiplexing approach allows us to substitute operations having multiple-control qubits with multiple spatial modes and optical switches by encoding multiple controlled qubits into DOFs of a single photon. In this case, the number of gates required for the circuit will vary significantly depending on which photon the DOFs are encoded to. Therefore, this decomposition provides a guideline for efficiently encoding DOFs. This has an impact on the compilation process for quantum computers. Thus, when mapping the DOFs to a particular physical system, it is practical to use a strategy matching the physical system's nature to utilize the computation time and computational space efficiently. Another method of realizing C_kX gates is to decompose the C_kX gates directly into optical elements instead of CX or C_2X gates, which has been proposed in Ref. [43]. This method requires beam splitters with transmittance, mirrors, and phase shift for realizing multiple controlled-phase gates. For realizing C_kX gates in real physical systems, it is necessary to quantitatively evaluate various resources and compare multiple implementation methods.

Next, our approach can be applied to other error correction codes and quantum algorithms that require a large number of gates. For example, the implementation of a discrete-time quantum walk algorithm [44] requires a large number of shift operators [45], i.e., a unitary operator that walks a "quantum walker" in a certain space. The circuit construction of the shift operator depends on the space in which the quantum walk is performed (for instance, in a graph [46], one-dimensional space [47], or a hypercube [48]), but in many cases, they consist of multiple controlled quantum gates [49] similar to the ones considered in this work. This is because the shift operator depends on the state of the quantum walker at a certain time and transitions to the quantum state of the quantum walker at the next time step, and such dependence on the previous time is realized by control gates. Therefore, applying quantum multiplexing to this system can also reduce the total number of CX gates. Our approach can also be applied to Grover's algorithm [50], in which $C_{N-1}X$ gates are commonly used in the diffusion operator (inversion about the average) when the dimension of the search space of N qubits is 2^N [51]. Our gate decomposition to create an encoded logic state finds a natural way of being implemented through the quantum multiplexing technique. Quantum multiplexing allows for reducing the number of elementary CX gates, thus having a profound impact on the compilation process for quantum computers by optimizing efficiently the computational time and space.

ACKNOWLEDGMENTS

S.N. acknowledges Min-Hsiu Hsieh from Foxconn Quantum Computing Center for useful discussions throughout this project. This work was supported by JSPS KAKENHI Grants No. JP21H04880 and No. JP22J20882, the MEXT Quantum Leap Flagship Program (MEXT Q-LEAP) Grant No. JPMXS0118069605, and the JST Moonshot R&D Grant No. JPMJMS2061.

APPENDIX A: THE ENCODING CIRCUIT FOR THE $[[d, 1, (d+1)/2]]_d$ QRS CODES

Here we consider the encoding circuit of the $[[d, 1, (d+1)/2]]_d$ QRS code. The generator matrix for this code is given by

$$G = \begin{pmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 0 & \alpha & \alpha^2 & \dots & \alpha^{d-2} & 1 \\ 0 & \alpha^2 & \alpha^4 & \dots & \alpha^{2(d-2)} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \alpha^{d-k-1} & \alpha^{2(d-k-1)} & \dots & \alpha^{(d-2)(d-k-1)} & 1 \end{pmatrix}, \tag{A1}$$

where α is the primitive element of the Galois field GF(d). The second row of this matrix G can be used for deriving a logical operator for the code word. The logical X gate $\mathrm{X_L}$ can be realized by applying the X gate the number of times corresponding to an element to the corresponding qubit. As such, the logical operator $\mathrm{X_L}$ is given by

$$\begin{aligned}\mathrm{X_L} &= \mathrm{X}^0\mathrm{X}^{\alpha}\mathrm{X}^{\alpha^2}\ldots\mathrm{X}^{\alpha^{d-2}}\mathrm{X}^1\\ &= \mathrm{I}\mathrm{X}^{\alpha}\mathrm{X}^{\alpha^2}\ldots\mathrm{X}^{\alpha^{d-2}}\mathrm{X}.\end{aligned} \tag{A2}$$

Then, the code words for the code can be expressed as

$$|i\rangle_{\mathrm{L}} = \mathrm{X}_L^i\sum_{j=0}^{d-1}|jjj\ldots j\rangle, \tag{A3}$$

where $|jjj\ldots j\rangle$ are the basis states of the d qudits.

In order to evaluate the number of SUM gates required to create the code-word state $|i\rangle_{\mathrm{L}}$, let us consider the initial state given by the tensor products of $|\psi\rangle \equiv \sum_{k=0}^{d-1}\alpha_k|i\rangle$ with $d-1$ qudits initialized in the $|0\rangle$ state. First, we apply the DFT gate on the dth $|0\rangle$ state, giving

$$|\psi\rangle\otimes|00\ldots0\rangle \xrightarrow{\mathrm{DFT}_d} |\psi\rangle\otimes|00\ldots0\rangle\otimes|+\rangle. \tag{A4}$$

Then we apply α^i times the SUM gate on the target qubits, which are the ones initialized to $|0\rangle$ and the control qubit $|\psi\rangle$. Therefore, the total number of SUM gates will be given by

$$\sum_{l=0}^{d-1}\alpha^i = \frac{d^2-d-2}{2}. \tag{A5}$$

If we denote the gate used to perform such an operation as C-$\mathrm{X_L}$, using the same notation as in (6), we have

$$|\psi\rangle\otimes|00\ldots0\rangle\otimes|+\rangle \xrightarrow{\text{C-X}_\text{L}} \sum_{k=0}^{d-1}a_kX_{\mathrm{L}}^k|00\ldots0\rangle\otimes|+\rangle. \tag{A6}$$

We then apply a SUM gate between the state $|+\rangle$ and each of the other states, giving in total $d-1$ SUM gates. As before, by denoting D the gate used to perform the above operation, we have

$$\sum_{k=0}^{d-1}a_k\mathrm{X}_{\mathrm{L}}^k|00\ldots0\rangle\otimes|+\rangle \xrightarrow{\mathrm{D}} \sum_{k=0}^{d-1}a_k\mathrm{X}_{\mathrm{L}}^k\sum_{j=0}^{d-1}|jj\ldots j\rangle, \tag{A7}$$

which gives the encoded initial logic state $|\psi\rangle_{\mathrm{L}}$. In summary, this encoding circuit requires one DFT gate and $(d^2+d-4)/2$ SUM gates.

As an example, let us consider the $d=5$ case, where we start with the initial state

$$|\psi\rangle\otimes|0000\rangle = (a|0\rangle + b|1\rangle + c|2\rangle + d|3\rangle + e|4\rangle)\otimes|0000\rangle. \tag{A8}$$

Following the steps described above, we first apply the DFT gate on the last qudit $|0\rangle$ state. This will give

$$|\psi\rangle\otimes|0\rangle\otimes|0\rangle\otimes|0\rangle\otimes|0\rangle \xrightarrow{\mathrm{DFT}} |\psi\rangle\otimes|0\rangle\otimes|0\rangle\otimes|0\rangle\otimes|+\rangle, \tag{A9}$$

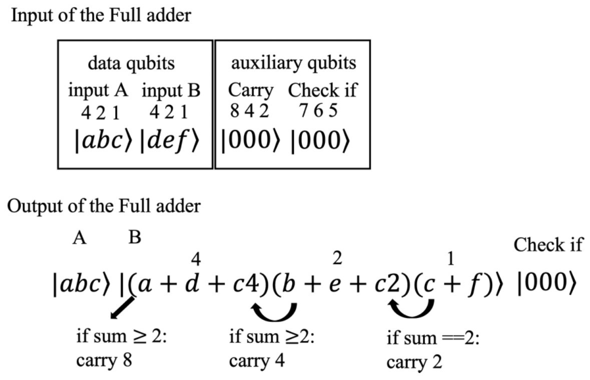


FIG. 6. Input and output of the RCA operation. Carry qubits are used to store the overflow of each place for 4, 2, and 1. Check-if qubits are not used here.

where we define $|+\rangle \equiv (|0\rangle + |1\rangle + |2\rangle + |3\rangle + |4\rangle)/\sqrt{5}$ as the equally weighted superposition. We then apply the C-$\mathrm{X_L}$ gate on four qudits (with the first qudit as the control), giving

$$\begin{aligned}|\psi\rangle\otimes|0\rangle\otimes|0\rangle\otimes|0\rangle\otimes|+\rangle &\xrightarrow{\text{C-X}_\text{L}} a|0000+\rangle + b|1423+\rangle\\ &+ c|2341+\rangle + d|3214+\rangle + e|4132+\rangle.\end{aligned} \tag{A10}$$

Finally, we apply the D gate (four SUM gates on first through fourth qudits as targets, with the fifth qudit being used as the control). That will give the initial logic state for the $d=5$ case:

$$\begin{aligned}&a|0000+\rangle\, b|1423+\rangle + c|2341+\rangle + d|3214+\rangle + e|4132+\rangle\\ &\xrightarrow{\mathrm{D}} a(|00000\rangle + |11111\rangle + |22222\rangle + |33333\rangle + |44444\rangle)\\ &\quad + b(|14230\rangle + |20341\rangle + |31402\rangle + |42013\rangle + |03124\rangle)\\ &\quad + c(|23410\rangle + |34021\rangle + |40132\rangle + |01243\rangle + |12304\rangle)\\ &\quad + d(|32140\rangle + |43201\rangle + |04312\rangle + |10423\rangle + |21034\rangle)\\ &\quad + e(|41320\rangle + |02431\rangle + |13042\rangle + |24103\rangle + |30214\rangle).\end{aligned}$$

The quantum circuit for this encoding is shown in Fig. 1.

APPENDIX B: CIRCUIT IMPLEMENTATION OF THE SUM GATES (THE MODULO ADDER)

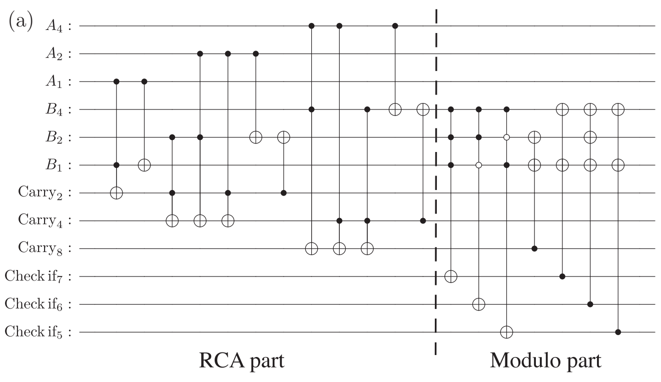
This Appendix describes the construction of a quantum circuit for a d-dimensional SUM gate. The inputs of this gate are the pair of k qubits A and B corresponding to the pair of d-dimensional qudits, and the output is the modulo with respect to d of the sum of the two inputs.

As described in Sec. II, the circuit consists of an RCA part and a modulo part. The RCA part adds the numbers of the same place in inputs A and B and stores the overflow of the place in the carry qubits for use in calculating the next digit. Figure 6 shows what information is stored in which qubit during the calculation of the RCA part for $d=5$.

It is important for us to provide the details of how the RCA circuit is constructed and the number of CX gates it uses to achieve that. We will consider the addition of two binary numbers, A and B, each represented by k qubits. We denote with A_i (B_i) the ith qubit of the number A(B), respectively. Now the RCA module is given by the sequential adder of two digits, A_i and B_i, with an ancilla qubit (C_{2^i}, which we will refer to as the carry qubit), that stores the possible overflow of the sum of A_{i-1} and B_{i-1}. This sequential adder unit will then be made by a sequence of smaller adders subunits. The entire RCA circuit is then made by k ancillae, one for each digit of the numbers A and B. Next, let us discuss the gates used in these subunits. First, one C_2X gate is used to calculate whether $(A_i + B_i)$, $(A_i + C_{2^i})$, or $(B_i + C_{2^i})$ equals 2, respectively, and write this information into the carry qubit $C_{2^{i+1}}$. This ancilla will be an input in the computation of the next digit. Then, A_i and C_{2^i} add to B_i using one CX gate each. Thus, three C_2X gates and two CX gates are needed for each subunit. One can think of the C_2X gate as a gate needed to propagate the information of overflow to the next subunit, whereas the CX gate is used to calculate the sum. We note that the number of gates required for the first subunit can be optimized and reduced to only two CX gates since the smallest digit has no carry. In such a case, one CX gate is used to calculate the possible overflow, and the other CX gate adds A_1 and B_1. Summarizing, a k-digit RCA quantum circuit requires one C_2X, one CX, $3(k-1)$ C_2X, and $2(k-1)$ CX, giving N_{RCA} as the result of the general decomposition of $3k-2$ C_2X gates and $2k-1$ CX gates.

Next the modulo circuit converts the $|(a+b)\,[\mathrm{mod}\,2^k]\rangle$ state stored in B to the $|(a+b)\,[\mathrm{mod}\,d]\rangle$ state. Such a conversion is necessary for the case in which the sum of the inputs is between d and $2(d-1)$. To achieve this, we prepare a check-if qubit for each of the numbers from d to $2(d-1)$, and then the check-if qubit is flagged (flipped from $|0\rangle$ to $|1\rangle$) when the sum of the inputs in each corresponding values is between d and $2(d-1)$. In order to perform such an operation, we perform a C_kX gate with the data qubits B as the control qubits, with the corresponding checkif qubit as the target qubit. It is important to mention that we have chosen our control and 0-control based on the binary representation of each number. For normal controls, the gate is applied to the target when the control qubit is $|1\rangle$, but for 0-controlled operations, the gate is applied when the control is $|0\rangle$. If one wants to invert the check-if qubit for 5 (101), use the control, 0-control, and control for C_kX. Next, when the sum cannot be a number greater than 2^k, the most significant carry can be used as the check-if qubit of 2^k. In this case, the C_kX gate for check-if qubit inversion is also unnecessary.

It is now useful to consider an explicit example. If $d = 5$, then a conversion is required when the sum of the inputs is between 5 and 8. Since $0\,[\mathrm{mod}\,8] = 0\,[\mathrm{mod}\,5],\ldots, 4\,[\mathrm{mod}\,8] = 4\,[\mathrm{mod}\,5]$, no conversion is required for $d = 0$–4. Further, one can also substitute $Carry_8$ as a check-if qubit of 8 in this case. After the flip of the check-if qubits, multiple-target controlled gates convert the data qubits B based on the check-if qubit. For instance, for $d = 5$, we apply controlled XIX, XXX, XIX, IXX gates with a check-if qubit as the control qubit. This works as the conversion from 5 (101 in binary), 6 (110), 7 (111), 8 (000 and 1 in carry) to 0 (000), 1 (001), 2 (010), 3 (011). The number of target qubits of the controlled gates for each conversion ($5 \rightarrow 0$, $6 \rightarrow 1$, $7 \rightarrow 2$, and $8 \rightarrow 3$) is the hamming distance between binary representation of in/out numbers. [As an example, $H_D(6, 1) = 3$ since the binary representation of 6 is 110 and 1 is 001.] Therefore, N_M requires $\sum_{i=d}^{2(d-1)} H_D(i, i\,[\mathrm{mod}\,d])$ CX gates.



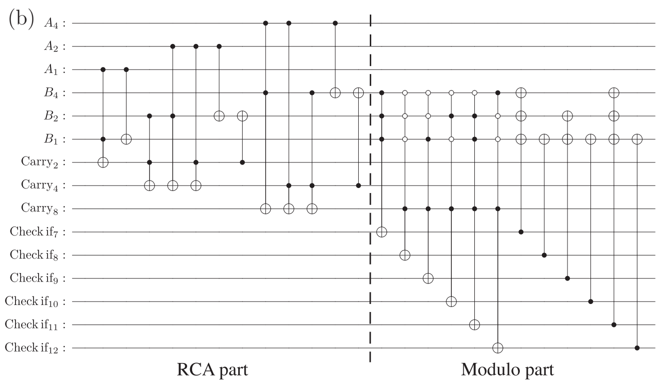


FIG. 7. Circuit layout for the SUM gate for the five-dimensional (a) and seven-dimensional (b) qudit in the qubit system. It consists of a RCA and a modulo part, where the black dots represent control qubits and the white dots represent 0-control qubits. Circuit (b) also includes the $C_{K+1}X = C_4X$ gates which have a control qubit in $Carry_8$.

It is important to note that when the maximum value that the sum can take $2(d-1)$ is greater than 2^k, the largest carry qubit must be used as a control qubit at the same time. For example, in the $d = 7$ case, the mod $2^k \rightarrow$ mod d transformation is needed when the sum is between 7 and 14. However, to determine that the sum is 8,9,10,11,12, not only must the data qubit be 000,001,010,011,100, but $Carry_8$ must be 1. This makes it necessary to use the $C_{k+1}X$ gate instead of C_kX. Such $C_{k+1}X$ gates can only be decomposed up to C_2X gates when multiplexing decomposition is used since the control qubits are in two qudits (data and carry). For the decomposition of C_2X gates to CX gates, we need to use the general method.

The case of constructing a modulo adder for the five- and seven-dimensional qudit using 2^3 multiplexed photon is shown in Fig. 7.

APPENDIX C: THE ENCODING CIRCUIT FOR THE GF(2^m) QRS CODES

Let us consider the $[[d-1, 1, \mathrm{D} \geqq \frac{N+1}{2}]]_d$ QRS code on GF(2^m) with $d = 2^m$. This code construction is based on $[[kN, k(N-2K), d \geqq K+1]]$ (written in binary form) for $N - 2K = 1$. To illustrate how this works, let us take the GF(2^2) code as a simple example. The Galois field for the code is represented as

$$\mathrm{GF}(2^2) = \{0, 1, \alpha, \alpha^2\}, \tag{C1}$$

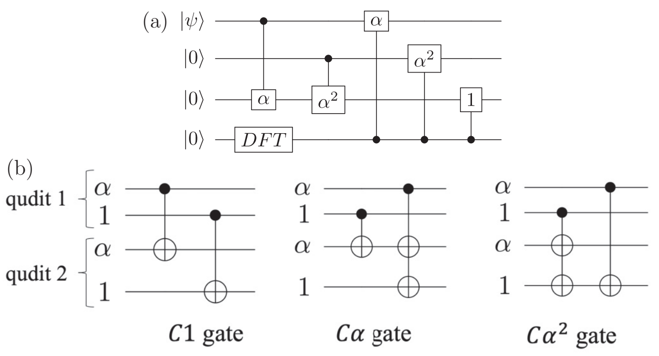


FIG. 8. (a) The encoding circuit of $[[3, 1, D \geqq 2]]_4$ quantum Reed-Solomon code. Each horizontal line represents a four-dimensional qudit (ququart). The two qudit gates defined by the black dot representing the control and the target with a square box containing the "1" (α, α^2) symbol represent the C1 (Cα, Cα^2) gates defined in (C5)–(C7). (b) Qubit implementation of the C1, Cα, and Cα^2 gates.

where we have chosen $x^2 + x + 1 = 0$ as the primitive polynomial. Then, the elements of the field can be written in polynomial and vector representations as

$$\begin{aligned} \mathrm{GF}(4) &= \{0, 1, \alpha, \alpha^2\} \text{ exponential representation} \\ &= \{0, 1, \alpha, \alpha + 1\} \text{ polynomial representation} \\ &= \{00, 01, 10, 11\} \text{ vector representation} \end{aligned}$$

We use the classical code $\mathcal{C} = [3, 2, 2]_4$ and its dual $\mathcal{C}^\perp$ for our CSS construction in this specific instance. The generator polynomial for the code is given as

$$g^\perp(x) = (x - 1)(x - \alpha) = x^2 + (1 + \alpha)x + \alpha, \tag{C2}$$

from which we get the generator and parity check matrices:

$$\mathrm{G} = \mathrm{H}^\perp = \begin{pmatrix} 1 & 0 & \alpha \\ 0 & 1 & \alpha^2 \end{pmatrix}, \tag{C3}$$

$$\mathrm{H} = \mathrm{G}^\perp = (\alpha \quad \alpha^2 \quad 1). \tag{C4}$$

We also show in Fig. 8(a) the quantum circuit for the encoder.

Now we denote the qudit gates required for our implementation as C1, Cα, and Cα^2, defining them as follows:

$$\mathrm{C1}(|a\rangle\,|b\rangle) = |a\rangle\,|a + b\rangle\,, \tag{C5}$$

$$\mathrm{C}\alpha(|a\rangle\,|b\rangle) = |a\rangle\,|\alpha a + b\rangle\,, \tag{C6}$$

$$\mathrm{C}\alpha^2(|a\rangle\,|b\rangle) = |a\rangle\,|\alpha^2 a + b\rangle\,. \tag{C7}$$

These gates correspond to the addition and multiplication of Galois elements for the calculation of coefficients of polynomials in the quantum Reed-Solomon codes. They can be implemented in a multiplexing system as shown in Fig. 8(b). As such, the gates of $\mathrm{GF}(2^{\mathrm{m}})$ can be constructed using only CX gates.

By generalizing the above method, we can calculate the cost of the gates we will need for implementing the $[[d - 1, 1, \mathrm{D} \geqq (N + 1)/2]]_d$ QRS code. First, the C1 gate can be achieved with k CX gates, while the Cα^n gates can be implemented with $\sum_{p=0}^{k-1} \mathrm{H_w}(\alpha^{\mathrm{n+p}})$ CX gates for the $\mathrm{GF}(2^k)$ system,

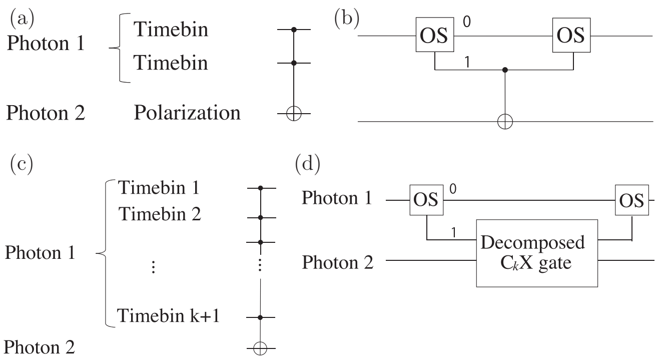


FIG. 9. (a) The $\mathrm{C_2X}$ gate with a photon having as control two time bins and another photon having as the target with the polarization DOF. (b) Optical implementation of (a). (c) $\mathrm{C}_{k+1}\mathrm{X}$ between two photons. (d) The circuit implementation for (a). Optical switches divide the $(k + 1)$th time-bin qubit.

where $\mathrm{H_w}()$ is a function whose input is an exponential representation of an element of the Galois field and whose output is a Hamming weight of the vector representation of the element. In this case, we do not need the $\mathrm{C}_k\mathrm{X}$ gate to implement the encoding circuit of the code. Therefore, the application of quantum multiplexing does not lead to any advantage in terms of the reduction of the number of gates.

APPENDIX D: PROOF FOR THE MULTIPLEXED DECOMPOSITION

Theorem. A $\mathrm{C}_k\mathrm{X}$ gate, which has $k \in \mathbb{Z}^+$ control time-bin qubits in a photon and a target qubit in another photon, can be decomposed into a single CX gate alongside a number of optical switches.

Proof. Let $S(k)$ be the statement that the $\mathrm{C}_k\mathrm{X}$ gate in the theorem can be decomposed into one CX gate and a number of optical switches, which we will now prove by induction, beginning with $k = 1$. Since $\mathrm{C_1X}$ gate is the CX gate from a control photon to a target photon, it holds by definition. Next, as shown explicitly by construction in Figs. 9(a) and 9(b), $\mathrm{C_2X}$ can be implemented with two optical switches and one CX gate, therefore $S(2)$ holds. Now assume the induction hypothesis $S(k)$ is true. Since the $\mathrm{C}_{k+1}\mathrm{X}$ gate can be realized by controlling the $\mathrm{C}_k\mathrm{X}$ gate with the $(k + 1)$th time-bin qubit, it can be decomposed as shown in Figs. 9(c) and 9(d). From $S(k)$, the $\mathrm{C}_k\mathrm{X}$ gate can be implemented with one CX and several OSs, therefore $S(k + 1)$ holds. ■

Note that, using the implementation of CX gate in Appendix C of Ref. [39], the target qubit can be either polarization or time bin. We now determine the number of

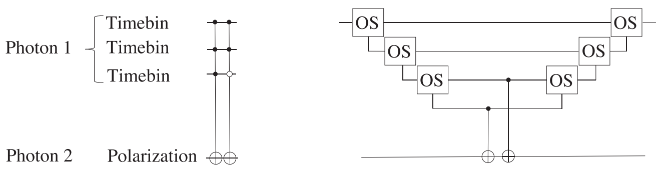


FIG. 10. A quantum circuit involving two $\mathrm{C_3X}$ gates (left) and an optimal way to implement that circuit with the multiplexed system (right).

OSs that would be required for the multiplexed decomposition. In the simplest case (OS has a one-input/two-output mode or two-input/one-output mode), $C_{k+1}X$ can be realized recursively with a C_kX gate and two OSs, hence a C_kX gate can be realized with a CX gate and $2(k-1)$ OSs. Some OSs can be used to split a multiplexed photon into multiple spatial modes. If the OS has more than two inputs/outputs then the number of OSs can be significantly reduced. This gives us an upper bound for the number of OSs required to implement a single C_kX gate. In general, $k-1$ OSs are used for splitting the components of the DOFs of the photon into multiple spatial modes, and other $k-1$ OSs are used for combining those into a single spatial mode. We want to highlight here that if a series of C_kX is executed, the number of OSs per C_kX can be further reduced. For example, we can realize the circuit of Fig. 10 using only six OSs. Here three OSs are used to decompose the time-bin modes of photon 1 into three corresponding spatial modes. Then, the relevant CX gates are successively performed before the other OSs recombine the time-bin modes into a single spatial mode. This procedure allows realizing the two C_3X gates shown on the left side of Fig. 10.

[1] W. J. Munro, K. Azuma, K. Tamaki, and K. Nemoto, Inside quantum repeaters, IEEE J. Sel. Top. Quantum Electron. **21**, 78 (2015).

[2] L.-M. Duan, M. D. Lukin, J. I. Cirac, and P. Zoller, Long-distance quantum communication with atomic ensembles and linear optics, Nature (London) **414**, 413 (2001).

[3] L. Jiang, J. M. Taylor, K. Nemoto, W. J. Munro, R. Van Meter, and M. D. Lukin, Quantum repeater with encoding, Phys. Rev. A **79**, 032325 (2009).

[4] H.-K. Lo and H. F. Chau, Unconditional security of quantum key distribution over arbitrarily long distances, Science **283**, 2050 (1999).

[5] W.-Y. Hwang, Quantum Key Distribution with High Loss: Toward Global Secure Communication, Phys. Rev. Lett. **91**, 057901 (2003).

[6] C. H. Bennett and G. Brassard, Quantum cryptography: Public key distribution and coin tossing, Theor. Comput. Sci. **560**, 7 (2014).

[7] A. K. Ekert, Quantum Cryptography Based on Bell's Theorem, Phys. Rev. Lett. **67**, 661 (1991).

[8] P. Arrighi and L. Salvail, Blind quantum computation, Int. J. Quantum. Inform. **04**, 883 (2006).

[9] A. Broadbent, J. Fitzsimons, and E. Kashefi, Universal blind quantum computation, in *2009 50th Annual IEEE Symposium on Foundations of Computer Science* (IEEE, 2009), pp. 517–526.

[10] J. I. Cirac, A. K. Ekert, S. F. Huelga, and C. Macchiavello, Distributed quantum computation over noisy channels, Phys. Rev. A **59**, 4249 (1999).

[11] E. Schanda, *Physical Fundamentals of Remote Sensing* (Springer, New York, 2012).

[12] H. J. Kimble, The quantum internet, Nature (London) **453**, 1023 (2008).

[13] P. W. Shor, Scheme for reducing decoherence in quantum computer memory, Phys. Rev. A **52**, R2493 (1995).

[14] D. Gottesman, Stabilizer codes and quantum error correction, arXiv:quant-ph/9705052.

[15] D. J. MacKay, G. Mitchison, and P. L. McFadden, Sparse-graph codes for quantum error correction, IEEE Trans. Inf. Theory **50**, 2315 (2004).

[16] D. Gottesman, A. Kitaev, and J. Preskill, Encoding a qubit in an oscillator, Phys. Rev. A **64**, 012310 (2001).

[17] A. R. Calderbank and P. W. Shor, Good quantum error-correcting codes exist, Phys. Rev. A **54**, 1098 (1996).

[18] A. Y. Kitaev, Quantum error correction with imperfect gates, in *Quantum Communication, Computing, and Measurement* (Springer, New York, 1997), pp. 181–188.

[19] W. H. Pierce, *Failure-Tolerant Computer Design* (Academic Press, New York, 1965).

[20] M. Grassl, W. Geiselmann, and T. Beth, Quantum Reed-Solomon codes, in *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes* (Springer, New York, 1999), pp. 231–244.

[21] R. Cleve and D. Gottesman, Efficient computations of encodings for quantum error correction, Phys. Rev. A **56**, 76 (1997).

[22] P. J. Nadkarni and S. S. Garani, Encoding of quantum stabilizer codes over qudits with $d = p^k$, in *2018 IEEE Globecom Workshops (GC Wkshps)* (IEEE, New York, 2018), pp. 1–6.

[23] G. Duclos-Cianci and D. Poulin, Fast Decoders for Topological Quantum Codes, Phys. Rev. Lett. **104**, 050504 (2010).

[24] A. Rigby, J. C. Olivier, and P. Jarvis, Modified belief propagation decoders for quantum low-density parity-check codes, Phys. Rev. A **100**, 012330 (2019).

[25] I. S. Reed and G. Solomon, Polynomial codes over certain finite fields, J. Soc. Ind. Appl. Math. **8**, 300 (1960).

[26] S. B. Wicker and V. K. Bhargava, *Reed-Solomon Codes and Their Applications* (John Wiley & Sons, New York, 1999).

[27] Y. C. Feng and P. C. Yuen, Protecting face biometric data on smartcard with Reed-Solomon code, in *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)* (IEEE, New York, 2006), pp. 29–29.

[28] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error Correcting Codes* (Elsevier, New York, 1977), Vol. 16.

[29] S. Muralidharan, J. Kim, N. Lütkenhaus, M. D. Lukin, and L. Jiang, Ultrafast and Fault-Tolerant Quantum Communication across Long Distances, Phys. Rev. Lett. **112**, 250501 (2014).

[30] N. Lo Piparo, W. J. Munro, and K. Nemoto, Quantum multiplexing, Phys. Rev. A **99**, 022337 (2019).

[31] T. C. Ralph, A. J. F. Hayes, and A. Gilchrist, Loss-Tolerant Optical Qubits, Phys. Rev. Lett. **95**, 100501 (2005).

[32] N. Lo Piparo, M. Hanks, C. Gravel, K. Nemoto, and W. J. Munro, Resource Reduction for Distributed Quantum Information Processing Using Quantum Multiplexed Photons, Phys. Rev. Lett. **124**, 210503 (2020).

[33] A. Ketkar, A. Klappenecker, S. Kumar, and P. K. Sarvepalli, Nonbinary stabilizer codes over finite fields, IEEE Trans. Inf. Theory **52**, 4892 (2006).

[34] G. G. La Guardia, Constructions of new families of nonbinary quantum codes, Phys. Rev. A **80**, 042331 (2009).

[35] S. Muralidharan, C.-L. Zou, L. Li, and L. Jiang, One-way quantum repeaters with quantum Reed-Solomon codes, Phys. Rev. A **97**, 052316 (2018).

[36] S. X. Cui and Z. Wang, Universal quantum computation with metaplectic anyons, J. Math. Phys. **56**, 032202 (2015).

[37] M. Grassl, M. Rötteler, and T. Beth, Efficient quantum circuits for non-qubit quantum error-correcting codes, Int. J. Found. Comput. Sci. **14**, 757 (2003).

[38] H.-H. Lu, Z. Hu, M. S. Alshaykh, A. J. Moore, Y. Wang, P. Imany, A. M. Weiner, and S. Kais, Quantum phase estimation with time-frequency qudits in a single photon, Adv. Quantum Technol. **3**, 1900074 (2020).

[39] N. Lo Piparo, M. Hanks, K. Nemoto, and W. J. Munro, Aggregating quantum networks, Phys. Rev. A **102**, 052613 (2020).

[40] V. Vedral, A. Barenco, and A. Ekert, Quantum networks for elementary arithmetic operations, Phys. Rev. A **54**, 147 (1996).

[41] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Elementary gates for quantum computation, Phys. Rev. A **52**, 3457 (1995).

[42] T. C. Ralph, K. Resch, and A. Gilchrist, Efficient Toffoli gates using qudits, Phys. Rev. A **75**, 022313 (2007).

[43] J. Fiurášek, Linear-optics quantum Toffoli and Fredkin gates, Phys. Rev. A **73**, 062313 (2006).

[44] Y. Aharonov, L. Davidovich, and N. Zagury, Quantum random walks, Phys. Rev. A **48**, 1687 (1993).

[45] J. Kempe, Quantum random walks: An introductory overview, Contemp. Phys. **44**, 307 (2003).

[46] D. Aharonov, A. Ambainis, J. Kempe, and U. Vazirani, Quantum walks on graphs, in *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing* (2001), pp. 50–59.

[47] A. Ambainis, E. Bach, A. Nayak, A. Vishwanath, and J. Watrous, One-dimensional quantum walks, in *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing* (ACM, New York, 2001), pp. 37–49.

[48] C. Moore and A. Russell, Quantum walks on the hypercube, in *International Workshop on Randomization and Approximation Techniques in Computer Science* (Springer, New York, 2002), pp. 164–178.

[49] B. L. Douglas and J. B. Wang, Efficient quantum circuit implementation of quantum walks, Phys. Rev. A **79**, 052335 (2009).

[50] L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing* (ACM, 1996), pp. 212–219.

[51] C. Lavor, L. Manssur, and R. Portugal, Grover's algorithm: Quantum database search, arXiv:quant-ph/0301079.