IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 9, 2019, accepted January 21, 2019, date of publication January 30, 2019, date of current version February 27, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2896259

High-Speed and Adaptive FPGA-Based Privacy Amplification in Quantum Key Distribution

QIONG LI[1], BING-ZE YAN[1], HAO-KUN MAO[1], XIAO-FENG XUE[1], QI HAN[1], AND HONG GUO[2]

[1]Information Countermeasure Technique Institute, School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China
[2]State Key Laboratory of Advanced Optical Communication Systems and Networks, Center for Quantum Information Technology, Center for Computational Science and Engineering, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China

Corresponding author: Qiong Li (qiongli@hit.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61471141, Grant 61531003, and Grant 61771168, in part by the Key Technology Program of Shenzhen, China, under Grant JSGG20160 427185010977, and in part by the Space Science and Technology Advance Research Joint Funds under Grant 6141B06110105.

ABSTRACT Privacy amplification (PA) is a vital procedure in quantum key distribution (QKD) to shrink the eavesdropper's information about the final key almost to zero. With the increase of repeat frequency of discrete variable QKD (DV-QKD) system, PA processing speed has become the bottleneck in many high-speed DV-QKD systems. In this paper, a high-speed adaptive field-programmable gate array (FPGA)-based PA scheme using a fast Fourier transform (FFT) is presented. To decrease the computation complexity, a modified 2-D FFT-based Toeplitz PA scheme is designed. To increase the processing speed of the scheme on the constraint of limited resources, a real-value oriented FFT acceleration method and a fast read/write balanced matrix transposition method are designed and implemented in our scheme. The experimental results on a Xilinx Virtex-6 FPGA demonstrate that the throughput is nearly double of the latest FPGA based Toeplitz PA scheme according to the literature. Besides, this scheme owns not only the good adaptivity to compression ratio but also the compression ratio independent resource consumption. Therefore, this scheme can fit many high-speed QKD applications.

INDEX TERMS Quantum key distribution, privacy amplification, fast Fourier transform, field-programmable gate array.

I. INTRODUCTION

Quantum key distribution (QKD) is a notable technique which exploits the principles of quantum mechanics to accomplish the secure key distribution between two remote parties, called Alice and Bob. Since Bennet and Brassard proposed the first practicable protocol in 1984 [1], many protocols have been proposed successively. These protocols can be divided into discrete variable (DV) protocols and continuous variable (CV) protocols [2]–[10]. Mainly due to the thorough security analysis, DV-QKD has drawn more attentions, and many DV-QKD systems have been developed [7]–[10]. A DV-QKD system includes two parts: quantum subsystem and post-processing subsystem. The function of quantum subsystem includes quantum state preparation, transmission and measurement. The post-processing subsystem mainly consists of error reconciliation and privacy amplification (PA) [11]. The task of error reconciliation is to correct error bits between two parties and get the identical corrected bit string by means of exchanging information over a public classical channel [12], [13]. Since the attacker, called Eve, may not only eavesdrop the quantum channel but also have full access to the classical channel, he may obtain some information about the corrected bit string. Therefore, it is necessary for PA to shrink Eve's information about the final key to almost zero. Furthermore, PA is also an open issue in some technique associated with QKD, e.g. quantum private query [14]–[16]. PA eliminates the leaked information by mapping a long corrected bit string to a much shorter final key via universal$_2$ hash function families [17]–[21]. To reduce the finite size effect in a practical QKD system, the length of an input block for PA should be at least 10^6, which leads to the high computation complexity and large storage requirement [22]. Therefore, PA has

The associate editor coordinating the review of this manuscript and approving it for publication was Yinghui Zhang.

become the bottleneck in many QKD systems. To solve this problem, the researchers studied different kinds of hash functions, implementation algorithms and platforms. In the aspect of hash function selection, C. M. Zhang et al. chose a multiplicative-based universal$_2$ class of hash function to speed up PA process, and they constructed an optimal multiplication algorithm with four basic multiplications on the central processing unit (CPU) [23]. While the multiplication of large numbers is a complex calculation, which is difficult to transplant and further optimize. Nowadays, Toeplitz hashing is the most widely used in PA process because of its simple structure and parallel feature [24]. To speed up the implementation of Toeplitz hashing based PA, several implementation algorithms have been proposed. Hayashi et al. proposed a modified Toeplitz matrix to further decrease the computation [25]. Zhang *et al.* [26] proposed a block parallel algorithm. Yuan *et al.* [27] applied number theoretical transform (NTT) algorithm to Toeplitz matrix multiplication. For the first time, the use of fast Fourier transform (FFT) was proposed to accelerate Toeplitz hashing and improved the process speed significantly by Liu *et al.* [28]. Among these algorithms, the computation complexity of FFT-based algorithm is the lowest, i.e., $O(n \log n)$. As for the platform selection, CPU is the conventional option for Toeplitz based PA [27], [28]. While the performance improvement of Toeplitz based PA on CPU is limited by the weak parallel computation support of CPU. Graphic processing unit (GPU) draws many attentions due to its great advantage in parallel computing. Wang *et al.* [29] proposed a FFT-based PA algorithm in CV-QKD based on GPU and improved the processing speed of PA to over 1Gbps. However, the volume and power consumption of GPU are pretty high, making it not suitable for practical DV-QKD applications. Field-programmable gate array (FPGA) is a suitable platform for DV-QKD system with the feature of high-parallelism, compact volume and low power consumption. Zhang *et al.* [26] first proposed a block parallel algorithm to speed Toeplitz hashing on FPGA. Constantin *et al.* [30] and Yang *et al.* [31] proposed an improved block parallel algorithm for Toeplitz hashing on FPGA respectively [30], [31]. The scheme of S. S. Yang et al. achieves 64Mbps processing speed based on FPGA and reduces memory resources significantly [31].

As far as we know, all existing PA schemes on FPGA use parallel block method Toeplitz hashing with computation complexity of $O(n^2)$. It is natural to think of the FFT-based algorithm when a Toeplitz PA is designed on FPGA. However, it is a big challenge to implement the FFT-based Toeplitz PA on FPGA due to the requirements of input block length at least 10^6, and the limited resources. In this paper, a high speed FFT-based Toeplitz PA hardware scheme is proposed for the first time. The scheme is implemented on a Virtex-6 FPGA. The throughput of our scheme reaches 116Mbps with the input block length $n = 1M$. Compared with the latest FPGA based PA scheme, our scheme achieves nearly twice throughput on a lower level hardware platform. Except for the high throughput, our scheme owns the good adaptivity to compression ratio and the compression ratio independent resource consumption. These advantages helped it to fit more QKD applications.

The rest of this paper is organized as follows. Some related works are described in Section 2 as the basis. In Section 3, the proposed high speed FFT-based Toeplitz PA hardware scheme is introduced in details. In Section 4, the experiment results and analysis are given. In Section 5, some conclusions are drawn.

II. RELATED WORK

A. PRIVACY AMPLIFICATION

Privacy amplification is a process that allows two parties, Alice and Bob, to distill a secure final key from a partially secure bit string [17]. The definition of privacy amplification is given below from the standpoint of information theory. Before PA procedure in QKD, Alice and Bob share a random n-bit binary string $\mathbf{X}$, called the corrected key in QKD. Eve learns a correlated random string $\mathbf{W}$ providing $t(t < n)$ bits of information about $\mathbf{X}$, i.e., $\mathrm{H}(\mathbf{X}|\mathbf{W}) \geq n - t$. Alice and Bob wish to publicly choose a compression function $\mathrm{g} : \{0, 1\}^n \rightarrow \{0, 1\}^r$ such that Eve's partial information about $\mathbf{X}$ and her complete information of g can only give her little information about $\mathbf{Y} = \mathrm{g}(\mathbf{X})$. Such procedure is indicated as Fig. 1.

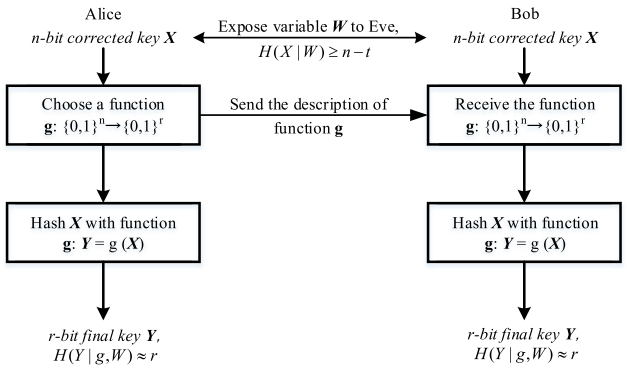


FIGURE 1. The Procedure of Privacy Amplification.

To choose a universal$_2$ hash function as the compression function g for PA, the mutual information between the distilled key compressed by universal$_2$ hash function and Eve's information follows:

$$I(\mathbf{Y}; \mathrm{g}, W) \leq 2^{-s} / \ln 2, \tag{1}$$

where $s = n - t - r$ denotes the security coefficient of PA [17], [21].

While the above definition based on the information theory ignores the setting where Eve holds quantum information. To address this problem, Renner et al. refined the definition of privacy amplification from the standpoint of composable security [32]. They further proved the upper bound of Eve's information asymptotically tight under both definitions, for n approaching infinity [32]. In practical QKD system, n is suggested to be larger than or equal to 10^6 to eliminate the difference between two definitions [33]. Therefore,

the implementation of PA is usually computation and storage expensive.

B. TOEPLITZ MATRIX BASED PA

Toeplitz matrix based PA is most popular in DV-QKD, and the essential processing is multiplying the input n-bit string X by a $n \times r$ random Toeplitz matrix [34]. For instance, (2) is a Toeplitz matrix,

$$\mathbf{T}_{r\times n} = \begin{bmatrix} V_{r-1} & V_r & \cdots & V_{n+r-3} & V_{n+r-2} \\ V_{r-2} & & & & V_{n+r-3} \\ \vdots & \ddots & & \ddots & \vdots \\ V_1 & & & V_{n-1} & V_n \\ V_0 & V_1 & \cdots & V_{n-2} & V_{n-1} \end{bmatrix}. \quad (2)$$

Because a Toeplitz matrix is a diagonal-constant matrix, it can be constructed by its first column and first row which means $n + r - 1$ random binary numbers are required. FFT algorithm is an efficient algorithm to calculate the Toeplitz matrix multiplication. The computation complexity of such multiplication operation can be reduced from $O(n^2)$ to $O(n \log n)$ via FFT. The main process of Toeplitz matrix multiplication accelerated by FFT is shown as (3),

$$\begin{aligned} &\mathbf{V}_{n+r-1} \otimes \mathbf{X}_{n+r-1} \\ &= [IFFT(FFT(\mathbf{V}_{n+r-1}) \bullet FFT(\mathbf{X}_{n+r-1})) \\ &= \begin{bmatrix} \mathbf{P}_{n\times r} & \mathbf{P}_{n\times n} \\ \mathbf{P}_{r\times r} & \mathbf{V}_{r\times n} \end{bmatrix} \times \begin{bmatrix} \mathbf{0} \\ \mathbf{X}_n \end{bmatrix} = \begin{bmatrix} \mathbf{P}_{n\times n} \times \mathbf{X}_n \\ \mathbf{V}_{r\times n} \times \mathbf{X}_n \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{P}_{n\times n} \times \mathbf{X}_n \\ \mathbf{Y}_r \end{bmatrix}, \end{aligned} \quad (3)$$

where $\mathbf{X_n} = [X_0, X_1, \cdots, X_{n-1}]'$ is the input sequence of PA, $\mathbf{Y_r} = [Y_0, Y_1, \cdots, Y_r]'$ is the final key sequence, $\mathbf{X}_{n+r-1} = [0, 0, \cdots, 0, X_0, X_1, \cdots, X_{n-1}]'$ and $\mathbf{V_{n+r-1}} = [V_0, V_1, \cdots, V_{n+r-2}]$ is the description of Toeplitz matrix. The function of matrix $\mathbf{P}$ is to transfer the Toeplitz matrix to a cyclic matrix. Using this method, the input length of FFT to calculate the Toeplitz matrix would be $n + r - 1$, which depends on the length of both input and output of PA.

Hayashi et al. proposed a modified Toeplitz matrix as the compression function and gave the security proof [25]. The modified Toeplitz matrix is constructed by the concatenation of Toeplitz matrix and the identity matrix ($\mathbf{I}$, $\mathbf{V}$). For instance, (4) is a modified Toeplitz matrix,

$$\mathbf{S_{r\times n}} = \begin{bmatrix} 1 & & V_{r-1} & V_r & \cdots & V_{n-2} \\ & 1 & V_{r-2} & V_{r-1} & & V_{n-3} \\ & \ddots & \vdots & & \ddots & \vdots \\ & & V_1 & & & V_{n-r} \\ & 1 & V_0 & V_1 & \cdots & V_{n-r-1} \end{bmatrix}. \quad (4)$$

Let $\mathbf{X_n} = [X_0, X_1, \cdots, X_{n-1}]'$ be the input sequence of PA, and $\mathbf{Y_r} = [Y_0, Y_1, \cdots, Y_r]'$ be the final key sequence. (5) is the calculative process of the final key with the modified Toeplitz matrix $\mathbf{S}_{r\times n}$,

$$\begin{aligned} \mathbf{Y_r} &= \mathbf{S_{r\times n}} \times \mathbf{X_n} = [\mathbf{I_{r\times r}}, \mathbf{V_{r\times(n-r)}}] \times \begin{bmatrix} \mathbf{X_r} \\ \mathbf{X_{n-r}} \end{bmatrix} \\ &= [\mathbf{I_{r\times r}} \times \mathbf{X_r}] + [\mathbf{V_{r\times(n-r)}} \times \mathbf{X_{n-r}}] = \mathbf{X_r} + \mathbf{Y_r'}. \end{aligned} \quad (5)$$

Make up the Toeplitz matrix $\mathbf{V}_{r\times(n-r)}$ to a cyclic matrix, then the calculation of $\mathbf{Y}'_r$ can be accelerated by FFT as shown in (6).

$$\begin{aligned} &\begin{bmatrix} \mathbf{P_{(n-r)\times(n-r)}} \times \mathbf{X_{n-r}} \\ \mathbf{Y_r'} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{P_{(n-r)\times(n-r)}} \times \mathbf{X_{n-r}} \\ \mathbf{V_{r\times(n-r)}} \times \mathbf{X_{n-r}} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{P_{(n-r)\times r}} & \mathbf{P_{(n-r)\times(n-r)}} \\ \mathbf{P_{r\times r}} & \mathbf{V_{r\times(n-r)}} \end{bmatrix} \times \begin{bmatrix} \mathbf{0} \\ \mathbf{X_{n-r}} \end{bmatrix} = \mathbf{V_{n-1}} \otimes \mathbf{X'_{n-1}} \\ &= \mathrm{IFFT}(\mathrm{FFT}(\mathbf{V_{n-1}}) \bullet \mathrm{FFT}(\mathbf{X'_{n-1}})), \end{aligned} \quad (6)$$

where $\mathbf{V}_{n-1} = [V_0, V_1, \cdots, V_{n-2}]$ is the description of Toeplitz matrix, $\mathbf{X}'_{n-1} = [0, 0, \cdots, X_r, \cdots, X_{n-1}]'$ is a part of the input sequence. The function of matrix $\mathbf{P}$ is to transfer the modified Toeplitz matrix to a cyclic matrix.

Using the modified Toeplitz matrix, the required quantity of random bits can be reduced from $n + r - 1$ to $n - 1$ and the input length of FFT can also be reduced from $n + r - 1$ to $n - 1$. In this case, the length of FFT is only related to the input length, not to the output length anymore. This feature can notably reduce the design complexity of FFT-based Toeplitz PA.

III. HIGH SPEED FFT-BASED PRIVACY AMPLIFICATION HARDWARE SCHEME

A high-speed PA hardware scheme for FPGA implementation is proposed in this section. Firstly, the overall process of the scheme is given. Then a modified 2-dimensional (2-D) FFT algorithm is proposed, followed by a real-value oriented FFT acceleration method and a block-wise matrix transposition method.

A. OVERALL PROCESS OF FFT-BASED PA HARDWARE SCHEMES

The overall process of FFT-based PA Hardware Scheme is indicated as Fig. 2.

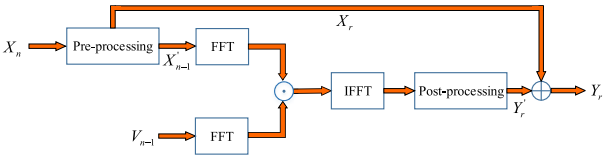


FIGURE 2. The Overall Process of FFT-based PA Hardware Scheme.

As a Toeplitz matrix based PA, our scheme is built upon the FFT algorithm and it only relies on the corrected key length without the final key length. Therefore, the scheme owns good adaptivity to the compression ratio. The compression ratio is defined as the ratio of the final key length r and the corrected key length n. The pre-processing phase is designed to adapt the varying compression ratio, in which the input sequence $\mathbf{X}_n = [X_0, X_1, \cdots, X_{n-1}]'$ is divided into $\mathbf{X}_r = [X_0, X_1, \cdots, X_{r-1}, 0, \cdots, 0]'$ and

$\mathbf{X}'_{n-1} = [0, 0, \cdots, 0, X_r, \cdots, X_{n-1}]'$. As the compression ratio changes, the pre-processing phase adjusts the length r in $\mathbf{X}_r$ and the length $n-r$ in $\mathbf{X}'_{n-1}$ as shown in Fig. 3.

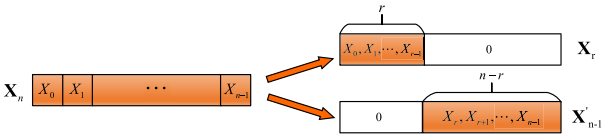


FIGURE 3. The Adaptive Design for the Compression Ratio.

The dot operational character means the dot product of the two FFT results. In the post-processing phase, the result of inverse FFT (IFFT) is rounded to a boolean sequence. The final key sequence of PA is the XOR of $\mathbf{X}_r$ and the result of the Toeplitz cyclic convolution $\mathbf{Y}'_r$.

During the whole processing, the main computation load comes from FFT/IFFT. Although some FFT/IFFT cores can be obtained for FPGA design, the input length of these cores cannot satisfy the requirement of PA. To overcome this problem, a 2-D large-point FFT is designed with small-point FFT hardware cores [35]. The procedure of 2-D large-point FFT algorithm is described as Algorithm 1.

Algorithm 1 2-D Large-Point FFT

Input: $X_n = x_0, x_1, \cdots, x_{n-1}$
Output: $Y_n = FFT(X_n)$

Convert a one-dimensional input sequence X_n into a two-dimensional matrix $A_{k\times k}$
$A' = Transposed(A)$
// $Transposed(A)$ is the transpose of the matrix A
for $i = 0$ to $k-1$ **do**
  $A_1[i][0:k-1] = FFT(A'[i][0:k-1])$
end for
for $i = 0$ to $k-1$ **do**
  for $j = 0$ to $k-1$ **do**
    $A_2[i][j] = A_1[i][j] \times W[i \times j]$
// W is the multiply rotation factor of FFT
// $W[i \times j] = e^{\frac{-i2\pi ij}{k}}$
  end for
end for
$A'_2 = Transposed(A_2)$
for $i = 0$ to $k-1$ **do**
  $A_3[i][0:k-1] = FFT(A'_2[i][0:k-1])$
end for
$A'_3 = Transposed(A_3)$
for $i = 0$ to $k-1$ **do**
  for $j = 0$ to $k-1$ **do**
    $Y[i \times k + j] = A'_3[i][j]$
  end for
end for

Although this method makes it possible to accomplish FFT/IFFT via multiple small-point FFT cores at high speed, many matrix transpositions and memory access are needed repeatedly. Thus, to speed up the PA scheme, the most important task is to optimize the number and speed of matrix transposition and memory access. Aiming at such a challenge, we design a modified 2-D FFT algorithm, a real-value oriented FFT acceleration method and a fast read/write balanced matrix transposition method.

B. A MODIFIED 2-D FFT FOR PA

Since matrix transposition is very time consuming, the PA can be accelerated if fewer matrix transpositions are required. According to Algorithm 1, three matrix transpositions are needed in the 2-D large-point FFT. It is found that removing the first and the third matrix transposition operations would only affect the order of output final key, while the mutual information between the input and output does not change at all. Therefore, the 2-D large-point FFT/IFFT algorithm can be simplified as Algorithm 2, the Modified 2-D Large-Point FFT for PA.

Algorithm 2 Modified 2-D Large-Point FFT for PA

Input: $X_n = x_0, x_1, \cdots, x_{n-1}$
Output: $Y_n = FFT(X_n)$

Convert a one-dimensional input sequence X_n into a two-dimensional matrix $A_{k\times k}$
for $i = 0$ to $k-1$ **do**
  $A_1[i][0:k-1] = FFT(A[i][0:k-1])$
end for
for $i = 0$ to $k-1$ **do**
  for $j = 0$ to $k-1$ **do**
    $A_2[i][j] = A_1[i][j] \times W[i \times j]$
  end for
end for
$A'_2 = Transposed(A_2)$
for $i = 0$ to $k-1$ **do**
  $A_3[i][0:k-1] = FFT(A'_2[i][0:k-1])$
end for
for $i = 0$ to $k-1$ **do**
  for $j = 0$ to $k-1$ **do**
    $Y[i \times k + j] = A_3[i][j]$
  end for
end for

If $\mathbf{Y} = \mathrm{g}(\mathbf{X})$ indicates the process of Toeplitz-based PA with 2-D large-point FFT, the process of Toeplitz-based PA with modified 2-D large-point FFT can be described by

$$\mathbf{Y}' = \mathrm{T}(\mathbf{Y}_1) = \mathrm{T}(\mathrm{g}(\mathbf{X}_1)) = \mathrm{T}(\mathrm{g}(\mathrm{T}(\mathbf{X}))) = \mathrm{g}'(\mathbf{X}), \quad (7)$$

where the function T is a sequence transformation indicated as

$$\mathrm{T}(\mathbf{X}(i + j \times C)) = \mathbf{X}_1(j + i \times C), \quad (8)$$

where $i, j = 0, 1, \cdots, C-1$, C means the number of rows of the matrix in the 2-D FFT. Although the transformation makes $\mathbf{Y}'$ different from $\mathbf{Y}$, it can be proved that the security of PA using Algorithm 2 and Algorithm 1 are exactly equal, i.e., $I(\mathbf{Y}'; \mathrm{g}', W) = I(\mathbf{Y}; \mathrm{g}, W)$. The detailed proof is presented in the following Proposition 1.

Proposition 1: Let $\mathbf{X}$ be a random n-bit string with uniform distribution over $\{0,1\}^n$. Let $W = e(\mathbf{X})$ for an arbitrary eavesdropping function $e : \{0,1\}^n \to \{0,1\}^t$, where $t < n$, and let the length of $\mathbf{Y}$ and $\mathbf{Y}'$ is $r = n - t - s$, where s is a positive safety parameter and $s < n - t$. Let function T be the sequence transformation indicated as (8). If Alice and Bob choose $\mathbf{Y}' = \mathrm{g}'(\mathbf{X})$(7) or $\mathbf{Y} = \mathrm{g}(\mathbf{X})$ as their secret key, where g is chosen at random from a universal$_2$ class of hash functions from $\{0,1\}^n$ to $\{0,1\}^r$, then Eve's expected information about the secret key satisfies $I(\mathbf{Y}'; \mathrm{g}', W) = I(\mathbf{Y}; \mathrm{g}, W) \leq 2^{-s}/\ln 2$.

Proof: According to (7), the main differences between $\mathbf{Y}$ and $\mathbf{Y}'$ are $\mathbf{Y}' = \mathrm{T}(\mathbf{Y}_1)$ and $\mathbf{X}_1 = \mathrm{T}(\mathbf{X})$. So the proof starts with the equivalent of the information uncertainty of Eve about $\mathbf{X}$ and $\mathbf{X}_1$ indicated as (9) and (10), respectively.

$$H(\mathbf{X}|\mathbf{W}, \mathrm{T}) = H(\mathbf{X}, \mathbf{W}, \mathrm{T}) - H(\mathbf{W}, \mathrm{T}), \tag{9}$$
$$H(\mathbf{X}_1|\mathbf{W}, \mathrm{T}) = H(\mathbf{X}_1, \mathbf{W}, \mathrm{T}) - H(\mathbf{W}, \mathrm{T}). \tag{10}$$

Expand combination entropy $H(\mathbf{X}, \mathbf{W}, \mathrm{T}, \mathbf{X_1})$ with chain rule as:

$$H(\mathbf{X}, \mathbf{W}, \mathrm{T}, \mathbf{X_1}) = H(\mathbf{X}, \mathbf{W}, \mathrm{T}) + H(\mathbf{X_1}|\mathbf{X}, \mathbf{W}, \mathrm{T}), \tag{11}$$
$$H(\mathbf{X}, \mathbf{W}, \mathrm{T}, \mathbf{X_1}) = H(\mathbf{X_1}, \mathbf{W}, \mathrm{T}) + H(\mathbf{X}|\mathbf{X_1}, \mathbf{W}, \mathrm{T}). \tag{12}$$

Because $\mathbf{X_1} = \mathrm{T}(\mathbf{X})$ is an one-one mapping relationship, $H(\mathbf{X_1}|\mathbf{X}, \mathbf{W}, \mathrm{T}) = H(\mathbf{X}|\mathbf{X_1}, \mathbf{W}, \mathrm{T}) = 0$. Then (13), (14) can be established,

$$H(\mathbf{X}, \mathbf{W}, \mathrm{T}) = H(\mathbf{X_1}, \mathbf{W}, \mathrm{T}), \tag{13}$$
$$H(\mathbf{X}|\mathbf{W}, \mathrm{T}) = H(\mathbf{X_1}|\mathbf{W}, \mathrm{T}). \tag{14}$$

Apparently,

$$H(\mathrm{g}(\mathbf{X})|\mathbf{W}, \mathrm{T}, \mathrm{g}) = H(\mathrm{g}(\mathbf{X}_1)|\mathbf{W}, \mathrm{T}, \mathrm{g}), \tag{15}$$
$$H(\mathbf{Y}|\mathbf{W}, \mathrm{T}, \mathrm{g}) = H(\mathbf{Y}_1|\mathbf{W}, \mathrm{T}, \mathrm{g}). \tag{16}$$

Applying the same deduction as (9) - (14), (17) can be obtained,

$$H(\mathbf{Y}_1|\mathbf{W}, \mathrm{T}, \mathrm{g}) = H(\mathbf{Y}'|\mathbf{W}, \mathrm{T}, \mathrm{g}), \tag{17}$$

so,

$$H(\mathbf{Y}|\mathbf{W}, \mathrm{T}, \mathrm{g}) = H(\mathbf{Y}'|\mathbf{W}, \mathrm{T}, \mathrm{g}). \tag{18}$$

Therefore, $I(\mathbf{Y}; \mathrm{g}, W) = I(\mathbf{Y}'; \mathrm{g}', W)$ according to the definition of mutual information. On the basis of existing proof in [17], $I(\mathbf{Y}; \mathrm{g}, W) = I(\mathbf{Y}'; \mathrm{g}', W) \leq 2^{-s}/\ln 2$. ■

Let us take a one-million points PA as an example. Because our PA algorithm is based on the 2-D large-point FFT algorithm, the input sequence will be loaded into a 1024×1024 matrix. The input and output sequences of the PA algorithm with the 2-D large-point FFT are shown in Fig. 4 (a). The input and output sequences of the PA algorithm with the modified 2-D large-point FFT for PA are shown in Fig. 4 (b). We have proved that the security of two methods is equivalent.

With the modified method, the number of matrix transformation and memory access that the large-point FFT/IFFT algorithm needs will significantly decrease. Since the PA algorithm needs one FFT and one IFFT. The number of matrix transposition in the whole PA algorithm will be decreased from six to two, and the time consumption of the PA algorithm will decrease significantly.

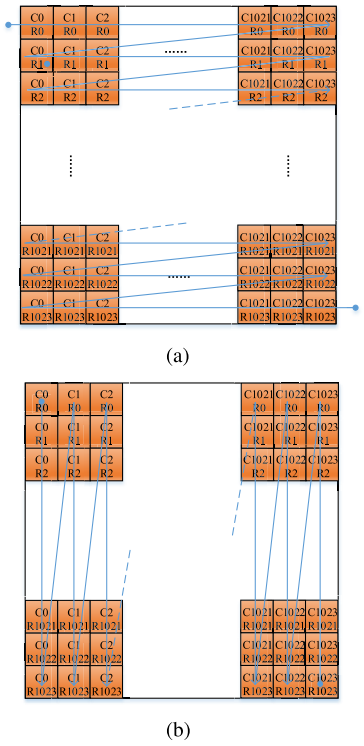


FIGURE 4. The Input/Output Sequence Order Diagram of FFT. (a) 2-D Large-Point FFT. (b) Modified 2-D Large-Point FFT.

C. REAL-VALUE ORIENTED FFT ACCELERATION

The commercially available FFT hardware cores are designed to compute the FFT of a complex sequence, while both the input sequence X_n and the description of Toeplitz V_n are real sequences. Most FFT-based PA schemes regard the input sequence as the real part and set the imaginary part to zero directly. This method leads to a waste of computing resource and storage resource. A real-value oriented FFT algorithm is introduced to solve this problem, to compute the FFT of the input sequence $x(n)$ and Toeplitz sequence $v(n)$. Their FFT results $X(k)$ and $V(k)$ can be obtained via one complex-valued FFT as described by (19)-(24) [36].

$$z(n) = x(n) + i \cdot v(n) \tag{19}$$
$$Z(k) = FFT(z(n)) \tag{20}$$
$$\mathrm{Re}[X(k)] = 1/2 \cdot (\mathrm{Re}[Z(k)] + \mathrm{Re}[Z(N-k)]) \tag{21}$$
$$\mathrm{Im}[X(k)] = 1/2 \cdot (\mathrm{Im}[Z(k)] - \mathrm{Im}[Z(N-k)]) \tag{22}$$
$$\mathrm{Re}[V(k)] = 1/2 \cdot (\mathrm{Im}[Z(k)] + \mathrm{Im}[Z(N-k)]) \tag{23}$$
$$\mathrm{Im}[V(k)] = 1/2 \cdot (\mathrm{Re}[Z(N-k)] - \mathrm{Re}[Z(k)]) \tag{24}$$

Such optimization method can save nearly half computing resource and storage resource of PA.

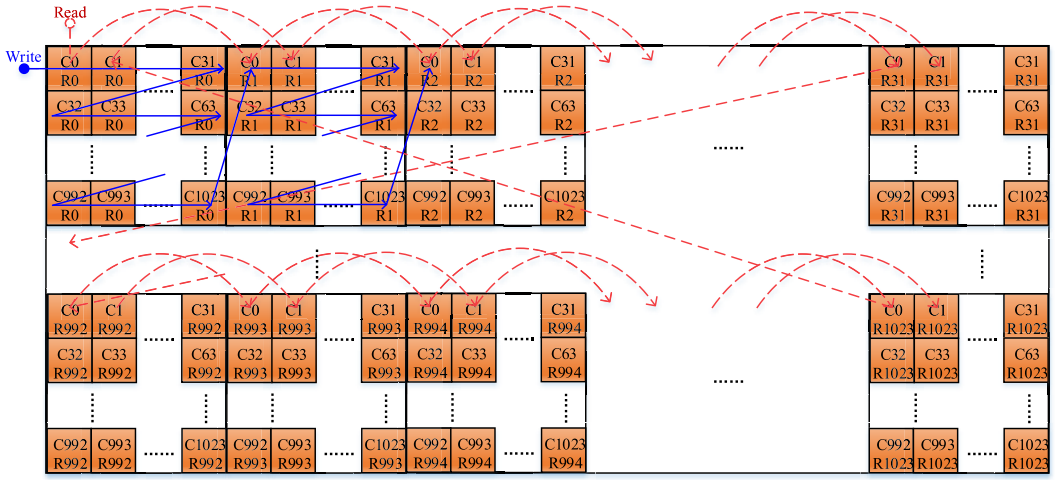


FIGURE 5. The High Effective Matrix Transposition Process.

D. BLOCK WISE MATRIX TRANSPOSITION

Although the modified 2-D FFT algorithm has decreased the number of the matrix transposition, the matrix transposition is still time-consuming. To improve the processing rate of PA further, an effective matrix transposition method, so-called block wise matrix transposition, is introduced in our scheme [37]. The access mechanism of double data rate synchronous dynamic random access memory (DDR-SDRAM, DDR for short) makes the row span access operation cost much more time than the inline access. Because the matrix transposition needs a large amount of the row span access operations, the block wise matrix transposition is introduced to reduce the number of the row span access operations, which is shown as (25) and (26).

$$\mathbf{M}^{write}_{l\times C+i,j+k\times C} = \mathbf{A}_{k+C\times l,j+C\times i}, \tag{25}$$

$$\mathbf{A}'_{k+C\times l,j+i\times C} = \mathbf{M}^{read}_{l+i\times C,k+j\times C}, \tag{26}$$

where $\mathbf{A}$ is the square matrix to be transposed, the size of the matrix is $N\times N$, N should be a perfect square, $C=\sqrt{N}$, and $i,j,k,l = 0,\cdots,C-1$. $\mathbf{M}$ indicates the DDR memory to access.

The main process of the common matrix transposition based on the DDR memory model is indicated as (27),

$$\mathbf{A}'_{j,i} = \mathbf{M}_{i,j} = \mathbf{A}_{i,j}. \tag{27}$$

Taking the one-million points PA algorithm as an example, the number of row span access operation in the common matrix transposition is calculated as (28),

$$T_{span} = T_{write} + T_{read} = 1024 + 1024\times 1024 = 1049600. \tag{28}$$

The block wise matrix transposition method uses the matrix partitioning technology to balance the number of row span access of read/write operations. This method can reduce the total row span access number and increase the processing rates of matrix transposition significantly. The main process of the block wise matrix transposition is indicated as Fig. 5.

In this case, each row of the matrix is transformed to a 32×32 matrix. The number of row span access operation of the block wise matrix transposition method is calculated as (29).

$$T_{span} = T_{write} + T_{read} = 32\times 1024 + 32\times 1024 = 65536. \tag{29}$$

To verify its improvement on the processing rates of matrix transposition, the experiment is carried out with the DDR3-SDRAM. The comparison experiment results of the two methods are indicated in TABLE 1.

According to the experiment results, the block wise matrix transposition method can bring a boost of the processing rate of matrix transposition by a factor of 2.76.

IV. IMPLEMENTATIONS AND RESULTS

The proposed PA scheme is implemented on the Xilinx ML605 Evaluation Kit. The kit includes a Virtex-6 XC6VLX240T-1FFG1156 FPGA with 241,152 logic cells and a 512MB DDR3 SDRAM. The overall structure of our PA scheme is shown in Fig. 6.

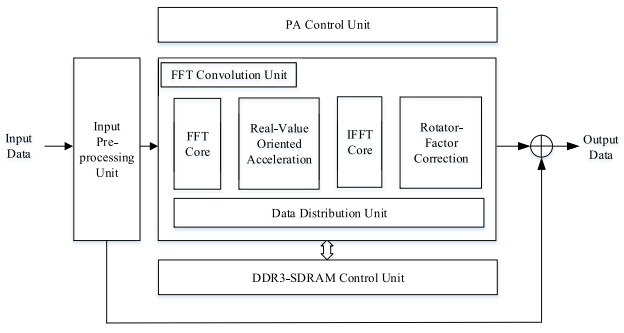


FIGURE 6. The Overall Structure of the PA Scheme.

The input preprocessing unit is designed to store the corrected keys and the Toeplitz random numbers. It also converts the data to the floating-points for the FFT convolution.

TABLE 1. The Effective Matrix Transposition Experiment Result.

	Data Size	Matrix Size	Operation time	Processing rate
The common method	64Mb	1024×1024	8.959 us	7.14Gbps
The block wise method	64Mb	1024×1024	3.242 us	19.74Gbps

TABLE 2. The Resource Utilization of the PA Scheme.

Resource	Used	Available	Rate
LUTs	37203	150720	24%
36K BRAMs	125	416	30%
18K BRAMs	64	832	7%
DSP48E1s	360	768	46%

[a] Used = the Scheme Used, Available = XC6VLX240T Available, Rate = Utilization Rate

In addition, because the Toeplitz matrix should be constructed randomly for each trail, the buffer size of the corrected keys and the Toeplitz random numbers should be same to guarantee the correctness and performance of the scheme. The function of PA control unit is to control the process sequence and data interaction of the other units. The FFT convolution unit is the key unit of the PA module, which contains five major parts. The FFT core is designed to calculate the FFT on each row of the matrix. The length of the FFT IP-core is set as 1024 in the one-million points PA algorithm. The processing rate of single FFT core is 12.8Gbps, and the maximum rate of the matrix transposition is 19.74Gbps. Hence, two FFT IP cores are used in our scheme to match the rates of the FFT cores and transposition. Similarly, the IFFT core is designed to calculate the IFFT on each row of the matrix. Two IFFT IP-cores are used and their lengths are also set as 1024. The real-value oriented acceleration unit completes the computation task of the real-value FFT efficiently. The rotator factory correction unit multiplies rotation factors point-wisely by the result of FFT/IFFT core to accomplish the 2-D large-point FFT algorithm. The data distribution unit distributes the data for the calculation units and exchanges data with DDR3-SDRAM controller. The scheme is simulated with Modelsim v10.4, and the function of the scheme is verified by comparing with the reference program on Matlab 2017a. Then the scheme is implemented on a ML605 Evaluation Kit and the results are accord with the simulation. The resource utilization of the PA scheme in hardware is shown in TABLE 2. According to the resource utilization, there is enough spare resource for other modules to constitute the complete post-processing system on the XC6VLX240T FPGA.

The comparison of several FPGA-based implementations of PA schemes is demonstrated in TABLE 3.

According to the implementation results, the processing speed of our PA scheme can reach 116Mbps, and it is nearly

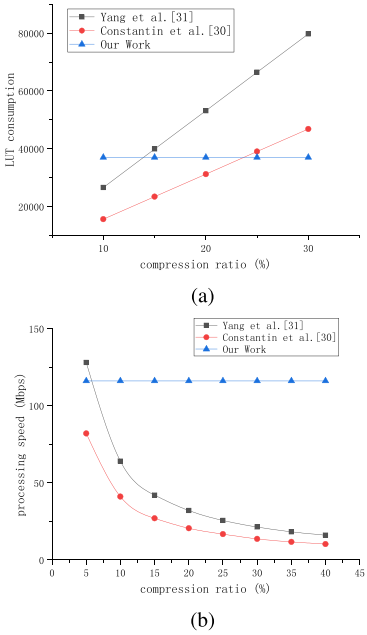


FIGURE 7. The PA Scheme Comparison as the Compression Ratio Changes. (a) the Resource Consumption vs the Compression Ratio. (b) the Processing-Speed vs the Compression Ratio.

double of the latest FPGA based Toeplitz PA scheme [31]. This processing speed enhancement mainly benefits for two reasons. Firstly, the computation complexity of the FFT algorithm is lower than that of the linear feedback shift register (LFSR) based algorithm. Secondly, the modified 2-D FFT algorithm with a real-value oriented FFT acceleration method and a block wise matrix transposition is employed in this scheme. Besides, the processing speed of the proposed scheme is mainly limited by the memory transfer rate, it can be improved greatly by simply replacing the DDR3 used in our scheme by a faster memory chip, e.g. DDR4-DRAM.

Except for the high processing speed, another advantage of our scheme is the good adaptivity to the compression rate. Both schemes in [30] and [31] are based on the LFSR, which suffer from the compression rate dependent resource consumption. To be more specific, the resource consumptions of those schemes increase sharply with the rising of compression rate to maintain the high processing speed. The compression ratio roughly varies in the range of 10% through 30% in existing QKD systems. For example, the compression ratio in the high speed QKD system of [27] is 29%. The scheme proposed in [31] is resource-saving when the compression ratio of PA is a fixed value 10%, but the resource requirement will double if the compression rate becomes 20% to maintain the same processing speed. Unlike the LFSR-based

TABLE 3. The comparison of Several FPGA-based PA schemes.

	This Work	Yang et al. [31]	Constantin et al. [30]
Devices	Xilinx Virtex-6	Xilinx Virtex-7	Xilinx Virtex-6
Length of the final key	0-1000,000	100,000	100,000
LUTs	37,203	26,571	15,604
External-BRAM	DDR3-SDRAM	–	DDR2-SDRAM
Max. processing speed	116Mbps	64Mbps	41Mbps

PA scheme, the resource consumption of proposed FFT-based PA is independent of the compression rate.

Keeping the throughput constant, the resource consumptions of our scheme and the comparative two schemes are shown in Fig. 7(a). Keeping the resource consumption stable, the throughputs of the three schemes are shown in Fig. 7(b). From the comparison, our FFT-based scheme can meet the requirements of more QKD systems.

Although the proposed FFT-based scheme costs about 4MB BRAM, which is higher than the comparative LFSR-based schemes, 4MB is acceptable consumption considering the total BRAM resource of a typical FPGA. For example, the XC6VLX240T FPGA used in our implementation contains 30MB BRAM.

V. CONCLUSIONS AND OUTLOOK

This paper provides a high-speed PA hardware scheme and its implementation in FPGA based on the FFT. The experimental results on a Xilinx Virtex-6 FPGA demonstrate that the throughput is nearly double of the latest FPGA based Toeplitz PA scheme according to the literature. Compared with other representative works, the proposed PA scheme can support wide-range and variable compression ratio. It can reach faster processing speed with faster memory. The optimization schemes proposed in this paper also fits the FFT-based PA algorithm on other platforms, such as CPU and GPU. In the future, we will try to further improve the processing speed and reduce the resource consumption of FFT-based PA scheme on FPGA.

ACKNOWLEDGEMENTS

The authors would like to thank Mr. S. S. Yang for the helpful discussion.

REFERENCES

[1] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," *Theor. Comput. Sci.*, vol. 560, pp. 7–11, 2014.

[2] D. Stucki, N. Brunner, N. Gisin, V. Scarani, and H. Zbinden, "Fast and simple one-way quantum key distribution," *Appl. Phys. Lett.*, vol. 87, no. 19, p. 194108, 2005.

[3] V. Scarani, A. Acín, G. Ribordy, and N. Gisin, "Quantum cryptography protocols robust against photon number splitting attacks for weak laser pulse implementations," *Phys. Rev. Lett.*, vol. 92, no. 5, p. 057901, 2004.

[4] K. Inoue, E. Waks, and Y. Yamamoto, "Differential phase shift quantum key distribution," *Phys. Rev. Lett.*, vol. 89, no. 3, p. 037902, 2002.

[5] D. Bruß, "Optimal eavesdropping in quantum cryptography with six states," *Phys. Rev. Lett.*, vol. 81, no. 14, pp. 3018–3021, 1998.

[6] C. H. Bennett, "Quantum cryptography using any two nonorthogonal states," *Phys. Rev. Lett.*, vol. 68, no. 21, pp. 3121–3124, May 1992.

[7] S. Wang *et al.*, "2 GHz clock quantum key distribution over 260 km of standard telecom fiber," *Opt. Lett.*, vol. 37, no. 6, pp. 1008–1010, 2012.

[8] A. Muller, J. Breguet, and N. Gisin, "Experimental demonstration of quantum cryptography using polarized photons in optical fibre over more than 1 km," *Europhys. Lett.*, vol. 23, no. 6, pp. 383–388, 2007.

[9] P. D. Townsend and I. Thompson, "A quantum key distribution channel based on optical fibre," *J. Mod. Opt.*, vol. 41, no. 12, pp. 2425–2433, 1994.

[10] C. Z. Peng *et al.*, "Experimental long-distance decoy-state quantum key distribution based on polarization encoding," *Phys. Rev. Lett.*, vol. 98, no. 1, p. 010505, 2007.

[11] X. Wang, Y.-C. Zhang, S. Yu, and H. Guo. (2018). "High efficiency postprocessing for continuous-variable quantum key distribution: Using all raw keys for parameter estimation and key extraction." [Online]. Available: https://arxiv.org/abs/1806.00173

[12] X. Wang, Y.-C. Zhang, S. Yu, and H. Guo, "High speed error correction for continuous-variable quantum key distribution with multi-edge type LDPC code," *Sci. Rep.*, vol. 8, no. 1, 2018, Art. no. 10543.

[13] X. Wang, Y.-C. Zhang, Z. Li, B. Xu, S. Yu, and H. Guo, "Efficient rate-adaptive reconciliation for continuous-variable quantum key distribution," *Quantum Inf. Comput.*, vol. 17, nos. 13–14, pp. 1123–1134, 2017.

[14] M. Jakobi *et al.*, "Practical private database queries based on a quantum-key-distribution protocol," *Phys. Rev. A, Gen. Phys.*, vol. 83, no. 2, p. 022301, 2011.

[15] X. Gao, Y. Chang, S.-B. Zhang, F. Yang, and Y. Zhang, "Quantum private query based on bell state and single photons," *Int. J. Theor. Phys.*, vol. 57, no. 7, pp. 1983–1989, 2018.

[16] F. Gao, B. Liu, W. Huang, and Q.-Y. Wen, "Postprocessing of the oblivious key in quantum private query," *IEEE J. Sel. Topics Quantum Electron.*, vol. 21, no. 3, pp. 98–108, May/Jun. 2014.

[17] C. H. Bennett, G. Brassard, C. Crépeau, and U. M. Maurer, "Generalized privacy amplification," *IEEE Trans. Inf. Theory*, vol. 41, no. 6, pp. 1915–1923, Nov. 1995.

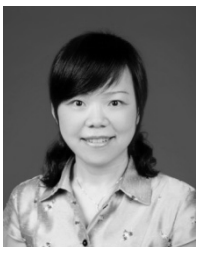
[18] C. H. Bennett, G. Brassard, and J.-M. Robert, "Privacy amplification by public discussion," *SIAM J. Comput.*, vol. 17, no. 2, pp. 210–229, Apr. 1988.

[19] R. Impagliazzo, L. A. Levin, and M. Luby, "Pseudo-random generation from one-way functions," in *Proc. 21st Annu. ACM Symp. Theory Comput. (STOC)*, 1989, pp. 12–24.

[20] M. N. Wegman and J. L. Carter, "New hash functions and their use in authentication and set equality," *J. Comput. Syst. Sci.*, vol. 22, no. 3, pp. 265–279, Jun. 1981.

[21] J. L. Carter and M. N. Wegman, "Universal classes of hash functions," *J. Comput. Syst. Sci.*, vol. 18, pp. 143–154, Apr. 1979.

[22] R. Y. Q. Cai and V. Scarani, "Finite-key analysis for practical implementations of quantum key distribution," *New J. Phys.*, vol. 11, pp. 1–17, Apr. 2009.
[23] C.-M. Zhang *et al.*, "Fast implementation of length-adaptive privacy amplification in quantum key distribution," *Chin. Phys. B*, vol. 23, no. 9, pp. 1–6, 2014.
[24] H. Krawczyk, "LFSR-based hashing and authentication," in *Advances in Cryptology—CRYPTO*, vol. 839. Berlin, Germany: Springer, 1994, pp. 129–139.
[25] M. Hayashi, "Exponential decreasing rate of leaked information in universal random privacy amplification," *IEEE Trans. Inf. Theory*, vol. 57, no. 6, pp. 3989–4001, Jun. 2011.
[26] H.-F. Zhang *et al.*, "A real-time QKD system based on FPGA," *J. Lightw. Technol.*, vol. 30, no. 20, pp. 3226–3234, Oct. 2012.
[27] Z. Yuan *et al.*, "10-Mb/s quantum key distribution," *J. Lightw. Technol.*, vol. 36, no. 16, pp. 3427–3433, Aug. 15, 2018.
[28] B. Liu, B.-K. Zhao, W. Yu, and C. Wu, "FiT-PA: Fixed scale FFT based privacy amplification algorithm for quantum key distribution," *J. Internet Technol.*, vol. 17, no. 2, pp. 309–320, 2016.
[29] X. Wang, Y. Zhang, S. Yu, and H. Guo, "High-speed implementation of length-compatible privacy amplification in continuous-variable quantum key distribution," *IEEE Photon. J.*, vol. 10, no. 3, Jun. 2018, Art. no. 7600309.
[30] J. Constantin *et al.*, "An FPGA-based 4 Mbps secret key distillation engine for quantum key distribution systems," *J. Signal Process. Syst.*, vol. 86, no. 1, pp. 1–15, 2017.
[31] S.-S. Yang, Z.-L. Bai, X.-Y. Wang, and Y.-M. Li, "FPGA-based implementation of size-adaptive privacy amplification in quantum key distribution," *IEEE Photon. J.*, vol. 9, no. 6, Dec. 2017, Art. no. 7600308.
[32] R. Renner and R. König, "Universally composable privacy amplification against quantum adversaries," in *Proc. Theory Cryptogr. Conf.*, 2005, pp. 407–425.
[33] V. Scarani and R. Renner, "Quantum cryptography with finite resources: Unconditional security bound for discrete-variable protocols with one-way postprocessing," *Phys. Rev. Lett.*, vol. 100, no. 20, p. 200501, 2008.
[34] T. Tsurumaru and M. Hayashi, "Dual universality of hash functions and its applications to quantum cryptography," *IEEE Trans. Inf. Theory*, vol. 59, no. 7, pp. 4700–4717, Jul. 2013.
[35] B. Lerner, "Parallel implementation of fixed-point FFTs on TigerSHARC processors," in *Technical Notes on Using Analog Devices DSPs, Processors and Development Tools*, vol. 263. Norwood, MA, USA: Analog Devices, 2005, pp. 1–12.
[36] H. Sorensen, D. Jones, M. Heideman, and C. Burrus, "Real-valued fast Fourier transform algorithms," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 35, no. 6, pp. 849–863, Jun. 1987.
[37] Q. W. Wu, "High efficiency matrix transposition method based on FPGA and DDR," *Mod. Radar*, vol. 39, no. 4, pp.15–44, 2017.



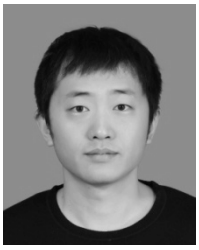
QIONG LI was born in Jishou, Hunan, China, in 1976. She received the B.S., M.S., and Ph.D. degrees in instrument science from the Harbin Institute of Technology, Harbin, Heilongjiang, China, in 1997, 1999, and 2005, respectively.

From 1997 to 2003 and from 2003 to 2006, she was a Teaching Assistant and a Lecturer with the School of Electrical Engineering, Harbin Institute of Technology. From 2006 to 2007 and from 2007 to 2015, she was a Lecturer and an Associate Professor with the School of Computer Science and Technology, Harbin Institute of Technology. In 2015, she was a Visiting Scholar with the Electrical and Computer Engineering Department, Toronto University. Since 2015, she has been a Professor with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin. She has authored or co-authored two books, more than 50 articles, and holds more than 20 authorized patents. Her research interests include post processing of quantum cryptography, multimedia security, and biometrics.

Dr. Li received the third prize of army scientific and technical progress award once, and the second prize of ministerial scientific and technical progress award once.

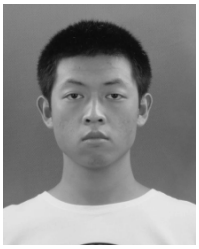
BING-ZE YAN was born in Heihe, Heilongjiang, China, in 1993. He received the B.S. and M.S. degrees in instrument science from the Harbin Institute of Technology, in 2015 and 2017, respectively.

He received the Ph.D. degree in computer science with the Harbin Institute of Technology, in 2018. His research interests include post processing of quantum cryptography and hardware security.



HAO-KUN MAO received the B.S. and M.S. degrees in computer science and technology from the Harbin Institute of Technology, in 2010 and 2013, respectively.

He received the Ph.D. degree in computer science and technology with the Harbin Institute of Technology, in 2018. His research interests include post processing of quantum cryptography and multimedia security.



XIAO-FENG XUE was born in Taiyuan, Shanxi, China, in 1993. He received the B.S. degree in computer science and technology from Xidian University, in 2017.

He received the Ph.D. degree in computer science and technology with the Harbin Institute of Technology, in 2018. His research interests include post processing of quantum key distribution and hardware security.

QI HAN was born in Pingdingshan, Henan, China, in 1981. He received the B.S. degree in fluid power transmission and control, the M.S. degree in instrumentation science and technology, and the Ph.D. degree in information security degree from the Harbin Institute of Technology, Harbin, Heilongjiang, China, in 2002, 2004, and 2009, respectively.

From 2009 to 2013, he was a Lecturer with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin, where he has been an Associate Professor since 2014. His research interests include multimedia security and signal processing.

HONG GUO received the B.S. degree from the National University of Defense Technology, in 1991, and the M.S. and Ph.D. degrees from the Shanghai Institute of Optics and Fine Mechanics, Chinese Academy of Sciences, in 1993 and 1995, respectively.

He is currently a Professor with the School of Electronics Engineering and Computer Science, Peking University, China.

He is involved in the research of quantum optics and quantum information.