Software Engineering and Applications 软件工程与应用, 2023, 12(3), 402-409
Published Online June 2023 in Hans. https://www.hanspub.org/journal/sea
https://doi.org/10.12677/sea.2023.123040

Hans汉斯

基于Toom-Cook多项式乘法SNTRUP算法的FPGA快速实现

马　钰*，丁海洋，李子臣

北京印刷学院数字版权保护技术研究中心，北京

收稿日期：2023年4月4日；录用日期：2023年5月31日；发布日期：2023年6月9日

摘　要

本文设计了基于Toom-Cook多项式乘法Streamlined NTRU Prime算法的FPGA快速实现方法，Toom-Cook多项式乘法能对Streamlined NTRU Prime算法中封装以及解封装运算的多项式系数相乘进行优化，能明显减少乘法运算次数，增加Streamlined NTRU Prime算法的运算效率。在ModelSim仿真软件Intel Cyclone IV GX系列EP4CGX150DF31I7AD芯片上进行仿真实验。实验结果表明，本方案可以在封装以及解封装速度上可以提升26%。

关键词

Streamlined NTRU Prime，多项式乘法，现场可编程门阵列，后量子密码

FPGA Fast Implementation Based on Toom-Cook polynomial Multiplication SNTRUP Algorithm

Yu Ma*, Haiyang Ding, Zichen Li

Digital Rights Protection Technology Research Center, Beijing Institute of Graphic Communication, Beijing

Received: Apr. 4th, 2023; accepted: May 31st, 2023; published: Jun. 9th, 2023

Abstract

This paper designs a fast FPGA implementation method based on Toom-Book polynomial multiplication Streamlined NTRU Prime algorithm. Toom-Book polynomial multiplication can optimize

*通讯作者。

文章引用: 马钰, 丁海洋, 李子臣. 基于Toom-Cook多项式乘法SNTRUP算法的FPGA快速实现[J]. 软件工程与应用, 2023, 12(3): 402-409. DOI: 10.12677/sea.2023.123040

the multiplication of polynomial coefficients of encryption and decryption operations in Streamlined NTRU Prime algorithm, significantly reduce the number of multiplication operations, and increase the efficiency of Streamlined NTRU Prime algorithm. The simulation experiment is carried out on the ModelSim simulation software Intel Cyclone IV GX series EP4CGX150DF31I7AD chip. The experimental results show that the encryption and decryption speed of this scheme can be improved by 26%.

Keywords

Streamlined NTRU Prime, Polynomial Multiplication, FPGA, Post-Quantum Cryptography

1. 引言

近年来，随着密码技术的快速发展，传统的密码算法已经不能抵抗量子计算机[1]的攻击，许多曾被认为无法破解的密码学困难问题，在量子计算技术面前变得不再可靠。20 世纪 90 年代中期，Shor [2]提出大数分解的量子算法，能够在多项式时间内解决大整数分解问题以及可以对离散对数问题进行求解，比传统计算机计算速度有指数级的提升。2001 年，O'brien J L 提出光量子计算[3]，仅使用单光子源，线性光学原件就可以进行可扩展的量子计算，又基于错误编码方法大大减少资源开销，使全光学架构量子计算机大规模实现成为可能。2018 年，Preskill J 提出了一种新的量子计算技术[4]，具有 50~100 个量子位的量子计算机能够代替现在的传统计算机，并且其量子门中的噪声可以使量子电路变得很小。基于此美国国家标准与技术研究院(NIST)于 2016 年开始后量子密码算法的征集工作[5]，2022 年 7 月，NIST 公布了第三轮 15 个后量子密码方案，目前的后量子加密算法主要分为基于格的算法[6]、基于编码的算法[7]、基于哈希的算法[8]、基于多变量的算法[9]、基于超奇异椭圆曲线的算法[10]和基于零知识证明的算法[11]，其中基于格的 NTRU Prime 算法[12]成为备选公钥加密和密钥生成算法，它具有较高的可移植性及安全性，且更易于在硬件上实现[13]。文献[14]对如何在 FPGA 上快速实现认证密钥协商协议的问题进行深入研究，设计了一种双向认证密钥协商协议，并且对 SNTRUP 算法的密钥生成速度和封装解封装速度都有较大提升。文献[15]针对格密码算法计算开销大的问题，设计了一种多项式乘法器，实现快速模乘运算。

NTRU Prime 算法由于代数结构上的差异以及加密方式的不同，分为 NTRU LPRime 算法(NTRU LPR)以及 Streamlined NTRU Prime 算法(SNTRUP)。本文重点介绍了 Streamlined NTRU Prime 算法的密钥产生过程以及封装和解封装过程，在算法封装过程中会大量使用多项式乘法运算，而乘法运算是影响封装速度的主要因素，并且会占用大部分的时间和资源，影响运算效率。因此本文对封装过程中的多项式乘法部分进行重点优化，在多项式相乘时，大整数相乘运算时间直接影响两个多项式相乘的时间，因此使用 Toom-Cook 算法[16] [17]对多项式系数进行拆分、评估、逐点相乘、插值、重组五个步骤，以减少多项式系数相乘的次数以及时间复杂度，增加 FPGA 硬件实现效率。最后在仿真软件 ModelSim 上对 Cyclone IV GX 系列 EP4CGX150DF31I7AD 芯片进行仿真实验，测试该算法的资源占用情况以及实现效率。

2. Streamlined NTRU Prime 算法

Streamlined NTRU Prime 算法是基于格理论的后量子加密算法，该算法主要包括四个部分：参数选择、

密钥生成算法、封装算法和解封装算法。下面是对这四个部分的重点介绍。

1) SNTRUP 参数选择

Streamlined NTRU Prime 定义了如下多项式环，令 $\mathbb{Z}$ 表示整数，环 $\mathbb{Z}[x]/(x^n-x-1)$、环 $(\mathbb{Z}/3)[x]/(x^n-x-1)$ 和环 $(\mathbb{Z}/q)[x]/(x^n-x-1)$ 分别记为 $\mathcal{R}$ 、 $\mathcal{R}/3$ 和 $\mathcal{R}/q$ 。Streamlined NTRU Prime 的参数(p, q, w)满足以下条件：p，q 均为素数，w 是正整数，$2p \geq 3w$，$q \geq 16w+1$，并且 x^p-x-1 在 $\mathcal{R}/q$ 中是不可约的。如果 $\mathcal{R}$ 的所有系数都在$\{-1, 0, 1\}$范围内，则称 $\mathcal{R}$ 的多项式为 Small 多项式。如果一个 Small 的 $2t$ 个系数不为零，我们就称它为 t-Small。Weight w 表示的多项式恰好有 w 个非零系数。Round 表示将多项式的所有系数近似到最接近 3 的倍数。$Hash_a(x)$表示以 a 作为 x 的前缀取 SHA-512 哈希值的前 256 位。Encode 和 Decode 表示使用编码或解码算法将 $\mathcal{R}/3$ 和 $\mathcal{R}/q$ 中的多项式转换为字节字符串。

Streamlined NTRU Prime 算法的参数集设置有六种，分别为：sntrup653、sntrup761、sntrup857、sntrup953、sntrup1013、sntrup1277，本文将针对其中的 sntrup761 参数集做具体研究，其具体参数为 $p=761$，$q=4591$，$w=286$。

2) SNTRUP 密钥生成算法

Streamlined NTRU Prime 算法的密钥生成过程需要产生公钥和私钥，具体算法如下：首先在 Small 中随机生成一个多项式 $g(x)$，并且确保多项式 $g(x)$ 在 $\mathcal{R}/3$ 中可逆。再随机生成一个属于 t-Small 的多项式 $f(x)$，并且满足多项式 $f(x) \in \mathcal{R}$。接下来随机生成一个属于 t-Small 的多项式 $\rho(x)$，并且多项式 $\rho(x)$ 长度为(p + 3)/4。然后把多项式 $g(x)$ 的倒数 $1/g(x)$ 用 $v(x)$ 表示，$g(x)/3f(x)$ 的结果用 $K(x)$ 表示，$K(x)$ 经过 Encode 编码生成 $\overline{K}(x)$，多项式 $f(x)$，$v(x)$ 共同编码生成 $\overline{k}(x)$，最后把 $\overline{K}(x)$ 作为公钥进行输出，$\left(\overline{k}(x), \overline{K}(x), \rho(x), hash_4\left(\overline{K}(x)\right)\right)$ 作为私钥进行输出。

3) SNTRUP 封装算法

Streamlined NTRU Prime 算法的加密过程需要输入在密钥生成过程中产生的公钥，输出结果为加密后的密文以及会话密钥，具体算法如下：首先随机生成一个属于 t-Small 的多项式 $r(x)$，接着对 $\overline{K}(x)$ 进行 Decode 解码生成 $K(x)$，再将多项式 $K(x)$ 和 $r(x)$ 相乘的结果 Round 后赋值给 $c(x)$，并将 $c(x)$ 进行 Encode 编码生成 $\overline{c}(x)$，接下来将 $r(x)$ 进行 Encode 编码生成 $\overline{r}(x)$。接着把 $\overline{c}(x)$、$hash_2\left(hash_3\left(\overline{r}(x)\right)\right)$ 和 $hash_4\left(\overline{K}(x)\right)$ 组合赋值给 C，C 就是最终的密文，把 $hash_1\left(hash_3\left(\overline{r}(x)\right)\right)$ 和密文 C 共同组合生成会话密钥。最后把密文以及会话密钥共同输出。

4) SNTRUP 解封装算法

Streamlined NTRU Prime 算法的解密过程需要把密文 C 以及在密钥生成过程中产生的私钥 $\left(\overline{k}(x), \overline{K}(x), \rho(x), hash_4\left(\overline{K}(x)\right)\right)$ 共同作为算法的输入，解密过程具体算法如下：首先把 $\overline{c}(x)$ 进行 Decode 解密生成 $c(x)$，再对 $\overline{k}(x)$ 解密生成 $f(x)$，$v(x)$，接着把 $3f(x)c(x)$ 的结果做模 3 运算，结果赋值给 $e(x)$，再把 $e(x)$ 和 $v(x)$ 乘积的结果用 $r'(x)$ 表示。接下来判断如果 $r'(x)$ 中没有 w 个非零系数，就把 $(1,1,\cdots,0,0,\cdots,0)$ 赋值给 $r'(x)$，其中前 w 个元素为 1，其余为 0。接着用 $\overline{K}(x)$ 和 $r'(x)$ 重新做封装计算新的密文 C’，然后把 $r'(x)$ 进行 Encode 加密生成 $\overline{r}'(x)$，判断 C’是否等于 C，如果等于就输出 $hash_1\left(hash_3\left(\overline{r}'(x)\right), C\right)$，如果不等于就输出 $hash_0\left(hash_3\left(\rho(x)\right), C\right)$。

3. SNTRUP 算法 FPGA 优化算法

3.1. 算法实现

在 Streamlined NTRU Prime 算法的封装和解封装以及生成公私钥的过程中会大量使用多项式乘法，而多项式乘法是影响算法效率的主要因素，两个多项式直接相乘虽然可以实现但是效率较低资源占用率较大，因此采用 Toom-Cook 算法对多项式乘法进行优化，需要把多项式均匀分成 k 个部分，进行 $2k-1$

次乘法来增加 Streamlined NTRU Prime 算法的运算效率。

Toom-Cook 算法是采用分治的思想，把给定的大整数分成 k 份，用 Toom-Cook-k 算法进行递归运算。本次设计综合考虑了 Streamlined NTRU Prime 算法的特性以及硬件资源占用比，采用 Toom-Cook-3 快速乘法对 Streamlined NTRU Prime 算法进行优化设计，图 1 是 SNTRUP 优化算法的流程图，在 $K(x)$多项式和 $r(x)$多项式相乘时使用 Toom-Cook-3 多项式乘法对 Streamlined NTRU Prime 算法进行优化，两个多项式的系数经过拆分、评估、逐点相乘、插值和重组五个步骤最终计算出结果 $c(x)$，其中 SNTRUP 算法的多项式相乘是影响该算法效率的主要步骤，逐点相乘是 Toom-Cook-3 多项式乘法的关键步骤，其流程图如图 2 所示。

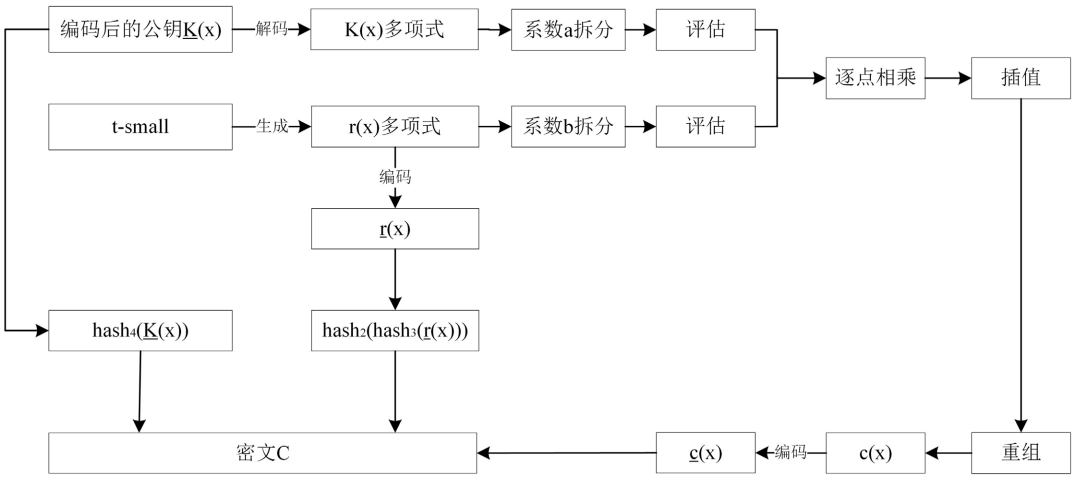

Figure 1. SNTRUP algorithm flow chart
图 1. SNTRUP 算法流程图

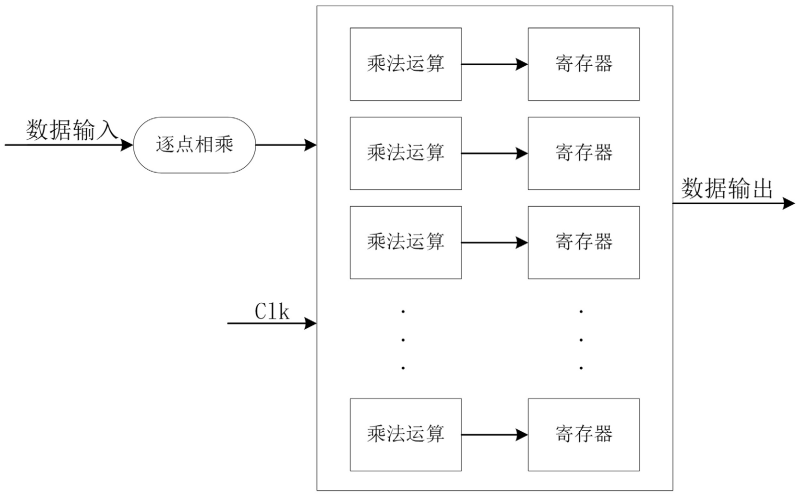

Figure 2. Point-by-point multiplication algorithm flowchart
图 2. 逐点相乘算法流程图

接下来具体介绍 Streamlined NTRU Prime 多项式乘法优化算法的五个步骤，首先该算法把 n 位数多项式系数 a 和 b 分成三段，每一段都有 $\left\lceil \frac{n-1}{3} \right\rceil$ 位数，把 a 和 b 可以表示为，如式(1)所示：

$$\begin{cases} a = a_2 \times 10^{2\times\left\lceil\frac{n-1}{3}\right\rceil} + a_1 \times 10^{\left\lceil\frac{n-1}{3}\right\rceil} + a_0 \\ b = b_2 \times 10^{2\times\left\lceil\frac{n-1}{3}\right\rceil} + b_1 \times 10^{\left\lceil\frac{n-1}{3}\right\rceil} + b_0 \end{cases} \tag{1}$$

例如：当整数为 12345678 时，就可以把此数写作 $12\times10^6+345\times10^3+678$。

由于 a 和 b 中 x 的最高次项均是 2 次项，所以乘积 c 中最高次项是 4 次项，乘积 c 的多项式表达式如式(2)所示。

$$c = a\times b = c_4 \times 10^{4\times\left\lceil\frac{n-1}{3}\right\rceil} + c_3 \times 10^{3\times\left\lceil\frac{n-1}{3}\right\rceil} + c_2 \times 10^{2\times\left\lceil\frac{n-1}{3}\right\rceil} + c_1 \times 10^{\left\lceil\frac{n-1}{3}\right\rceil} + c_0 \tag{2}$$

由于 10 的幂次是 $\left\lceil\frac{n-1}{3}\right\rceil$ 的整数倍，因此只需要计算 c_0，c_1，c_2，c_3，c_4 这 5 个系数，再根据各个系数与 $10^{\left\lceil\frac{n-1}{3}\right\rceil}$ 所相乘的值再进行左移操作，最后通过简单的加法操作得到乘积 c。由于 c 是一个关于 $10^{\left\lceil\frac{n-1}{3}\right\rceil}$ 的四次多项式，而 c_0，c_1，c_2，c_3，c_4 是这个多项式的五个未知数，因此就把问题转换为求解五个未知数的问题。因为求解四次多项式至少需要五个已知点才能求解，所以需要选取五组点构造五个方程，在点的选取上，可以选取任意的点来求解方程，但是在实际的操作中，会尽量选取容易计算的点，以简化计算，所以一般选取 x 的值为 0，1，−1，2，∞。将上述 x 的取值带入式(1)中得到 a 和 b 的结果，结果如下：

$$\begin{cases} a(0) = a_0 \\ a(1) = a_2 + a_1 + a_0 \\ a(-1) = a_2 - a_1 + a_0 \\ a(2) = 4a_2 + 2a_1 + a_0 \\ a(\infty) = a_2 \end{cases} \tag{3}$$

$$\begin{cases} b(0) = b_0 \\ b(1) = b_2 + b_1 + b_0 \\ b(-1) = b_2 - b_1 + b_0 \\ b(2) = 4b_2 + 2b_1 + b_0 \\ b(\infty) = b_2 \end{cases} \tag{4}$$

将上述 x 的取值带入式(2)中得到 c 的结果，结果如下：

$$\begin{cases} c(0) = c_0 \\ c(1) = c_4 + c_3 + c_2 + c_1 + c_0 \\ c(-1) = c_4 - c_3 + c_2 - c_1 + c_0 \\ c(2) = 16c_4 + 8c_3 + 4c_2 + 2c_1 + c_0 \\ c(\infty) = c_4 \end{cases} \tag{5}$$

以上式子中，$a(\infty)$、$b(\infty)$ 均是在趋近于 ∞ 时，$a(\infty)$ 或 $b(\infty)$ 与 ∞^2 的比值，$c(\infty)$ 是在 x 趋近于 ∞ 时与 ∞^4 的比值。因此，该算法就转换为求解式(5)的线性方程组。可以把方程组(5)表示为矩阵形式，如式(6)所示：

$$\begin{bmatrix} c(0) \\ c(1) \\ c(-1) \\ c(2) \\ c(\infty) \end{bmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 \\ 16 & 8 & 4 & 2 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \times \begin{bmatrix} c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{bmatrix} \tag{6}$$

将上述矩阵等式记作 $C = M \times N$，若要求解 N 矩阵，就要根据矩阵求逆法则，对矩阵 M 求逆，可得式(7)：

$$\begin{bmatrix} c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{bmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{6} & \frac{1}{6} & \frac{1}{2} \\ -1 & \frac{1}{2} & \frac{1}{2} & 0 & -1 \\ -\frac{1}{2} & 1 & -\frac{1}{3} & -\frac{1}{6} & -\frac{1}{2} \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \times \begin{bmatrix} c(0) \\ c(1) \\ c(-1) \\ c(2) \\ c(\infty) \end{bmatrix} \tag{7}$$

根据式(2)，可以快速求得 $c(0)$、$c(1)$、$c(-1)$、$c(2)$ 和 $c(\infty)$ 的值，再根据上述式(7)矩阵，求得 c_4、c_3、c_2、c_1 和 c_0 的值。再利用插值法对 c_4 左移 $4\times\left\lceil\frac{n-1}{3}\right\rceil$ 位，c_3 左移 $3\times\left\lceil\frac{n-1}{3}\right\rceil$ 位，c_2 左移 $2\times\left\lceil\frac{n-1}{3}\right\rceil$ 位，c_1 左移 $\left\lceil\frac{n-1}{3}\right\rceil$ 位，c_0 进行累加，最后得出运算结果 c。

Toom-Cook-3 算法把两个大整数分成三部分进行相乘，把优化前的 9 次乘法化简成 5 次乘法，它的时间复杂度是 $O(n^{1.465})$，对比不使用 Toom-Cook-3 算法两个大整数直接相乘的时间复杂度是 $O(n^2)$，因此 Toom-Cook-3 算法能提高多项式乘法的运算速度。

3.2. SNTRUP 算法框架

经过上述多项式乘法算法的优化，表 1 展示了 SNTRUP 算法的框架。

Table 1. Algorithmic framework
表 1. 算法框架

SNTRUP 优化封装算法
输入：编码后的公钥 $\overline{K}(x)$
输出：密文 C
1、对编码后的公钥 $\overline{K}(x)$ 解码，生成 $K(x)$ 多项式
2、生成一个属于 t-small 的多项式 $r(x)$
3、使用 Toom-Cook 算法对 $K(x)$ 和 $r(x)$ 相乘进行优化生成 $c(x)$
4、对相乘结果 $c(x)$ 进行编码得到 $\overline{c}(x)$
5、对 $\overline{K}(x)$ 进行 $hash_4\left(\overline{K}(x)\right)$ 运算
6、对 $r(x)$ 进行编码生成 $\overline{r}(x)$ 再进行 $hash_1\left(hash_3\left(\overline{r}(x)\right)\right)$ 运算
7、密文 C 等于 $\overline{c}(x)$、$hash_2\left(hash_3\left(\overline{r}(x)\right)\right)$ 和 $hash_4\left(\overline{K}(x)\right)$ 的组合

4. SNTRUP 的 FPGA 实验结果

4.1. 实验环境

本次实验的 FPGA 硬件环境平台是 Intel Cyclone IV GX 系列 EP4CGX150DF31I7AD 芯片，其查找表(LUT, Look Up Table)资源是 1497600，触发器(FF, flip flop)资源是 299520，数字信号处理器(DSP, Digital Signal Processing)资源是 1200，块 RAM (BRAM, Block RAM)存储器资源是 500，输入输出端口(I/O, Input/Output Part)资源是 508。使用硬件描述语言 Verilog-HDL 在 Quartus II 13.0 软件上实现了 Streamlined NTRU Prime 算法的整体硬件设计，在此环境中，使用 ModelSim SE-64 10.5 仿真软件仿真实现了后量子密码算法 Streamlined NTRU Prime 的密钥产生以及加解密部分，并针对 Streamlined NTRU Prime 算法的多项式乘法部分进行了具体优化。

4.2. 实验结果

文中方案使用多项式优化算法是 Toom-Cook-3 算法，对 Streamlined NTRU Prime 算法加解密部分的多项式乘法进行了优化，表 2 展示了 Toom-Cook-3 算法以及 Streamlined NTRU Prime 算法总体的资源占用情况。

Table 2. Resource usage of the Toom-Cook-3 algorithm and the Streamlined NTRU Prime algorithm
表 2. Toom-Cook-3 算法以及 Streamlined NTRU Prime 算法的资源占用情况

资源情况	LUT	FF	DSP	BRAM	I/O
Toom-Cook-3	205	128	1	0	5
SNTRUP	9082	4376	25	17	197
总体资源	149,760	299,520	1200	500	508

主要的性能指标为 LUT 查找表，FF 触发器，I/O 输入输出端口，DSP 数字信号处理器，BRAM 块 RAM。Toom-Cook-3 算法使用 LUT 资源占 Streamlined NTRU Prime 算法总消耗的 2.26%，FF 资源消耗占 2.93%，DSP 资源消耗占 4%，I/O 资源消耗占 2.54%。Streamlined NTRU Prime 算法消耗的 LUT 资源占总体资源量的 6.06%，FF 资源消耗占 1.46%，DSP 资源消耗占 2.08%，BRAM 资源消耗占 3.4%，I/O 资源消耗占 38.78%。由此可见，在此环境下实现 Streamlined NTRU Prime 算法消耗 I/O 资源占比较高，其余资源消耗均处在较低水平，而 Toom-Cook-3 算法模块各资源消耗占比均处在较低水平。本次设计只占用较少的资源就可以在 FPGA 上实现 SNTRUP 算法的密钥生成、封装以及解封装过程，并且 Toom-Cook-3 算法可以对 SNTRUP 算法进行加速操作，可以更加快速的实现 SNTRUP 算法。

5. 结束语

本文对后量子密码 Streamlined NTRU Prime 算法的密钥生成过程以及加解密过程进行了详细的介绍，针对在加解密阶段的多项式乘法部分进行了具体优化。本文通过使用 Toom-Cook-3 算法把多项式的大整数系数分成三部分，利用拆分、评估、逐点相乘、插值、重组五个具体步骤，把九次乘法优化成五次乘法，大大减少了乘法次数，相比于多项式系数直接相乘，本方案速度更快且资源占用更少。

为了验证 Toom-Cook-3 算法的优化效率，在 Intel Cyclone IV E 系列 EP4CE40F29C8 芯片平台上进行仿真实验，结果表明，采用 Toom-Cook-3 算法，比多项式直接相乘提升了效率，可以解决大整数乘法速度较慢的问题，因此本文提出的多项式优化算法具有一定的研究价值。

基金项目

国家自然科学基金(61370188)；北京市教委科研计划(KM202010015009)；北京市教委科研计划资助(No. KM202110015004)；北京印刷学院博士启动金项目(27170120003/020)；北京印刷学院科研创新团队项目(Eb202101)；北京印刷学院校内学科建设项目(21090121021)；北京印刷学院重点教改项目(22150121033/009)；北京印刷学院科研基础研究一般项目(Ec202201)；北京印刷学院博士启动金项目(27170122006)；北京印刷学院基础研究一般项目(Ec202201)；北京市高等教育学会 2022 年立项面上课题(MS2022093)；北京市教育委员会科学研究计划项目资助(KM202310015002)；北京印刷学院网络空间安全培育学科建设项目(21090123010)。

参考文献

[1] 张威. 拥抱量子科技时代: 量子计算的现状与前景[J]. 人民论坛. 学术前沿, 2021(7): 64-75. https://doi.org/10.16619/j.cnki.rmltxsqy.2021.07.007

[2] Shor, P. (1994) Algorithms for Quantum Computation: Discrete Logarithms and Factoring. *Proceedings* 35*th Annual Symposium on Foundations of Computer Science*, Santa Fe, 20-22 November 1994, 124-134.

[3] O'brien, J.L. (2007) Optical Quantum Computing. *Science*, **318**, 1567-1570. https://doi.org/10.1126/science.1142892

[4] Preskill, J. (2018) Quantum Computing in the NISQ Era and Beyond. *Quantum*, **2**, 79. https://doi.org/10.22331/q-2018-08-06-79

[5] NIST (2019) Post-Quantum Cryptography. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography

[6] IEEE (2009) IEEE Standard Specification for Public-Key Cryptographic Techniques Based on Hard Problems over Lattices, IEEE Standard p1363.1.

[7] Misoczki, R., Tillich, J., Sendrier, N. and Barreto, P.S.L.M. (2013) MDPC-McEliece: New McEliece Variants from Moderate Density Parity-Check Codes. 2013 *IEEE International Symposium on Information Theory*, Istanbul, 7-12 July 2013, 2069-2073. https://doi.org/10.1109/ISIT.2013.6620590

[8] Bernstein, D.J., Hopwood, D., Hulsing, A., Lange, T., Niederhagen, R., Papachristodoulou, L., Schneider, M., Schwabe, P. and Wilcox-O'Hearn, Z. (2015) SPHINCS: Practical Stateless Hash-Based Signatures. *EUROCRYPT* 2015: *Advances in Cryptology—EUROCRYPT* 2015, Sofia, 26-30 April 2015, 368-397. https://doi.org/10.1007/978-3-662-46800-5_15

[9] Ding, J. and Schmidt, D. (2005) Rainbow, a New Multivariable Polynomial Signature Scheme. In: Ioannidis, J., Keromytis, A. and Yung, M., Eds., *Applied Cryptography and Network Security*, Springer, Berlin, 164-175. https://doi.org/10.1007/11496137_12

[10] Costello, C., Longa, P. and Naehrig, M. (2016) Efficient Algorithms for Supersingular Isogeny Diffie-Hellman. *Advances in Cryptology—CRYPTO* 2016, Santa Barbara, 14-18 August 2016, 572-601. https://doi.org/10.1007/978-3-662-53018-4_21

[11] Koblitz, N. and Menezes, A. (2005) Pairing-Based Cryptography at High Security Levels. *Cryptography and Coding* 2005, Cirencester, 19-21 December 2005, 13-36. https://doi.org/10.1007/11586821_2

[12] Bernstein, D.J., Chuengsatiansup, C., Lange, T., *et al*. (2016) NTRU Prime. IACR Cryptology ePrint Archive, 2016/461. https://eprint.iacr.org/2016/461.pdf

[13] Peng, B.Y., Marotzke, A., Tsai, M.H., *et al*. (2022) Streamlined NTRU Prime on FPGA. *Journal of Cryptographic Engineering*, **13**, 167-186. https://doi.org/10.1007/s13389-022-00303-z

[14] 杨亚涛, 王在舟, 曾萍, 等. SNPAKA: 基于 SNTRUP 的双向认证密钥协商协议 FPGA 实现[J]. 密码学报, 2022, 9(4): 709-724. https://doi.org/10.13868/j.cnki.jcr.000544

[15] 余宏洲. 格密码算法多项式乘法器研究与设计[D]: [硕士学位论文]. 杭州: 浙江大学, 2022. https://doi.org/10.27461/d.cnki.gzjdx.2022.001002

[16] Bodrato, M. and Zanoni, A. (2007) Integer and Polynomial Multiplication: Towards Optimal Toom-Cook Matrices. *Proceedings of the* 2007 *International Symposium on Symbolic and Algebraic Computation*, Waterloo, 28 July-1 August 2007, 17-24. https://doi.org/10.1145/1277548.1277552

[17] Toom, A.L. (1963) The Complexity of a Scheme of Functional Elements Realizing the Multiplication of Integers. *Soviet Mathematics Doklady*, **3**, 714-716.