高能效混合基多项式乘法算法及可重构硬件结构研究与设计

别梦妮,李 伟*,陈 韬,李慧琴,杜怡然,南龙梅
(信息工程大学,河南郑州 450001)

摘 要: 本文针对快速多项式乘法算法与可重构单元的高能效设计问题展开研究,首先对现有的格基后量子密码算法展开研究,提出了一种基于数论变换(Number Theoretic Transform,NTT)的快速多项式乘法算法,并针对其中的核心运算过程,提出了高能效混合基的NTT和INTT(Inverse Number Theoretic Transform)算法,该算法可以利用NTT变换高效实现所有基于有限域的格基后量子密码算法中的多项式乘法. 在此基础上,对快速多项式乘法算法运算结构进行研究,在不增加额外运算部件的前提下,通过优化网络连接关系,提出了一种高能效可重构的混合基多项式乘法加速网络,在可灵活实现基2、基3、基4的NTT/INTT算法的同时,将基3与基4的NTT运算效率提升了一倍. 本文针对混合基NTT运算过程中的访存冲突问题展开研究,从理论上分析了冲突产生的原因,在此基础上分析提出了一种高能效混合基的内存管理方案,设计了相应的地址生成逻辑. 本文提出的内存访问方案是原地内存访问的一种,硬件固化后仍可实现不同的多项式乘法算法的内存管理. 实验结果表明,在55 nm CMOS工艺下,完成维度为256,模数小于2^{16}的多项式乘法运算仅需0.785 μs,最高工作频率可达到476 MHz,功耗为83.6 mW,面积时间积(Area Time Product,ATP)为152.604 kGE·μs. 与当前现有研究相比,本文提出的结构的ATP值降低了40%以上.

关键词: 后量子密码算法;格;多项式乘法;数论变换
基金项目: 国防预研项目(No.2019-JCJQ-JJ-123)
中图分类号: TN402;TP309 **文献标识码:** A **文章编号:** 0372-2112(2024)12-3957-10
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20230945

Research and Design of High Energy Efficient Hybrid Base Polynomial Multiplication Algorithm and Reconfigurable Hardware Structure

BIE Meng-ni, LI Wei*, CHEN Tao, LI Hui-qin, DU Yi-ran, NAN Long-mei
(*University of Information Engineering*, *Zhengzhou*, *Henan* 450001, *China*)

Abstract: In this paper, we present a fast polynomial multiplication algorithm, the hybrid-basis number theoretic transform (NTT) and inverse NTT (INTT) algorithms. These algorithms can efficiently implement polynomial multiplication based on finite domain using NTT conversion. On this basis, the paper explores the computational structure of fast polynomial multiplication algorithms. Without adding extra computational components, it optimizes network connectivity and proposes an energy-efficient reconfigurable hybrid-basis polynomial multiplication acceleration network. This network can flexibly implement base-2, base-3, and base-4 NTT/INTT algorithms, while doubling the operational efficiency of base-3 and base-4 NTT. This paper studies the issue of memory access conflicts in the computation process of hybrid-basis NTT. It theoretically analyzes the causes of these conflicts and, based on this analysis, proposes an energy-efficient hybrid-basis memory management scheme, designing the corresponding address generation logic. The proposed memory access scheme is a form of in-place memory access, and once implemented in hardware, it can still manage memory for different polynomial multiplication algorithms. Experimental results show that, under the 55 nm CMOS process, completing polynomial multiplication with a dimension of 256 and modulus less than 2^{16} requires only 0.785 μs. The maximum operating frequency can reach 476 MHz, with a power consumption of 83.6 mW and an area time product (ATP) of 152.604 kGE·μs. Compared to the existing research, the ATP value of the proposed structure in this paper is reduced by more than 40%.

收稿日期:2023-10-09;修回日期:2024-04-22;责任编辑:李勇锋
*通讯作者:李伟

Key words: post-quantum cryptography algorithm; lattice-based; polynomial multiplication; NTT
Foundation Item(s): National Defense Pre-Research Project (No.2019-JCJQ-JJ-123)

1 引言

新的信息时代背景下,自主安全可控的密码芯片是网络与信息安全领域的核心与基础. 近年来量子计算技术取得了一系列突破性进展,对传统公钥密码带来不容忽视的威胁. NIST(National Institute of Standards and Technology)后量子密码标准化征集至今已提交的算法高达百余种,其中基于格的算法占比最大. 2022年NIST公布的首批4个后量子密码标准中,格基算法高达三种,2018年中国密码学会举办的密码竞赛中一等奖获得者也都是基于格的算法. 由此可见,基于格的算法是未来后量子密码应用的主流选择. 量子计算的突破与后量子密码标准的提出急需后量子密码硬件设施作为基础支撑. 而欧洲、中国等国家和组织的后量子密码标准仍未确立,后量子密码算法的实际部署还需要一段过渡期. 此阶段针对后量子密码算法设计的芯片一方面需要满足当前已知算法的需求,另一方面还需要具备一定的可扩展性,以适应不断演进的后量子密码算法的发展需求. 现阶段对后量子密码算法的研究和评估主要由密码学者从算法理论层次上进行分析和优化,重点关注其安全性及计算复杂度. 在硬件实现层次上,当前的大多数研究集中面向高性能的应用场景,过大的能耗对小型移动设备上的实时通信并不友好. 如何从硬件角度对后量子密码算法的实现过程进行优化,从而实现算法的高能效应用具有重要的研究价值. 综上,本文面向格基后量子密码算法,研究一种可重构可扩展的加速器,以提高格基后量子密码算法的能效.

格基后量子密码算法中占用大量计算资源的多项式乘法是限制算法实际能效的主要因素. 在当前已公开提出的格基算法中,绝大多数多项式乘法均在有限域上进行,故而多数引入了基于数论变换(Number Theoretic Transform,NTT)[1~4]的快速运算方案. 遗憾的是,若算法不是特别基于NTT设计的则无法使用常规NTT进行加速. 例如NTRU、Saber、LAC等. 近年来,为提升这些NTT不友好的算法的实现性能,提出了Tom-Cook、Karatsuba及其与NTT之间的组合算法等[5~7]. 2021年,Chung[8]等人针对以上算法展开研究,分别针对不同的后量子密码算法设计了基于NTT进行加速的方案,取得了较好的加速效果. 以上方案均在不同程度上应用了包括基2^k的NTT、基3的NTT、Karastuba、Tom-Cook乘法在内的多种算法,或将其中几种算法混合应用,以达到最佳加速效果. 但截至目前,没有出现一种通用的快速多项式乘法算法满足多种格基后量子密码算法的能效需求.

鉴于此,本文研究并提出一种高能效的混合基多项式乘法优化算法,并在此基础上设计一种高能效可重构的快速多项式乘法加速网络,并提出一种高能效的无冲突内存访问方案. 本文的贡献总结如下.

(1)针对格基后量子密码算法,提出了一种高能效的混合基快速多项式乘法算法,改进了NTT/INTT算法,缩短了关键路径延迟.

(2)提出了一种高能效的可重构快速多项式乘法运算网络结构,通过优化网络连接关系,将基3与基4NTT的运算效率提升了一倍.

(3)将混合基NTT访存冲突问题理论化,分析了内存访问冲突的必然因素,在此基础上提出了一种高能效的内存访问方案,并设计了相应的地址映射逻辑.

2 高能效混合基快速多项式乘法算法研究与设计

为了深入理解各类快速多项式乘法算法,找到其中的内在共同逻辑并为高能效设计奠定基础,首先需要对目标算法中涉及的多项式乘法进行分析和优化,进而确定可重构实现的具体算法方案.

根据现有研究,有限域上的多项式乘法均可使用NTT及基于NTT的变种方案实现. 这些方案归纳总结为以下四种.

方案1: 原始Cooley-Tukey和Gentleman-Sande算法. 这两种算法是最原始的NTT算法,通常情况下结合二者使用以避免复杂的重排序运算. 该方案可以加速两个维度为n的多项式的乘法,且多项式在有限域上$Z[x]/g(x)$时,最终的计算结果需要模多项式$g(x)$. 其中,n为2的幂次,且模数p满足$n\left|(p-1)\right.$,存在n次本原单位根ω使得$\omega^n=1 \bmod p$.

方案2: 采用正、负包卷积的NTT算法. 正包卷积(也称循环卷积)方案要求模多项式为X^n-1,负包卷积方案要求模多项式为X^n+1,其中,n为2的幂次,且模数p满足$n\left|(p-1)\right.$,存在n次本原单位根ω使得$\omega^n=1 \bmod p$,负包卷积方案还要求存在$2n$次本原单位根γ使得$\gamma^{2n}=1 \bmod p$. 若不存在$2n$次本原单位根γ时,可将n次多项式分解为多个n'次多项式,直到可以找到$2n'$次本原单位根. 此时,对多个n'次多项式分别采用负包卷积进行计算,然后采用Schoolbook方法将多个n'次多项式进行整合. 该方案在拆分层次较少的情况下可以取得较高的能效,但若拆分层次较多时,算法能效提升不明显.

方案3:扩域后的NTT算法. 方案3主要针对$n\nmid(p-1)$的情况,此时,需要找到一个$q>np^2/2$,使得$n\mid(p-1)$,则可采用前述2种方案进行模q域上的计算,结果直接取低位即可. 若扩域后模值q过大,也可选择多个小素数q_i的乘积为模值,即令$q=\prod q_i$,分别在模q_i域内计算多项式乘法,然后通过中国剩余定理整合所有q_i域内得到的结果,从而将多项式系数恢复到模q_i域[9].

方案4:混合基的NTT算法. 方案4主要针对n不为2的幂次的情况,此时,需要对n值进行分解,找到可分解到的最小幂次k. 然后依照前述三种方案情况进行计算,与前述方案不同的是,其核心变换应当为基k的NTT变换. 若n值不可分解,或分解后的k值过大,也可对n值进行扩域后再行分解,在新域上进行计算后进行常规多项式取模.

结合上述四种情况,本文对当前已提出的格基后量子密码算法进行分析,表1列出了其中部分算法适合的多项式乘法加速方案.

表1　后量子密码算法中多项式乘法加速方案对照表

后量子算法	模多项式	模数q	维度n	加速方案
Kyber	X^n-1	3 329	256	2+3
Dilithum	X^n-1	8 380 417	256	2
Saber	X^n+1	2^{13}	256	2+3
NTRU	X^n-1、$(X^n-1)/(X-1)$	2^{11}、2^{12}、2^{13}	509、677、821、701	1+2+3+4
LAC	X^n+1	251	512、1 024	2+3+4
NewHope	X^n+1	12 289	1 024	2
……				

综上所述,只要是基于有限域设计的后量子密码算法,均可通过NTT进行加速,但具体实现过程存在差异. 若需要以一个统一的结构实现上述所有算法的多项式乘法加速,需要满足以下条件:

(1)该结构能实现基k的NTT变换,其中k为n的最小次幂.

(2)该结构除NTT外还需支持简单的Schoolbook方法以实现不完整的NTT算法的整合过程以及扩域后的NTT算法的恢复过程.

(3)该结构应当支持模数、维度可自由配置.

根据以上分析,本节提出一种混合基的多项式乘法算法如算法1所示. 算法1中,根据模多项式的不同,进行NTT运算时采用不同的旋转因子. 模数q默认满足$n\mid(p-1)$,若出现方案3的情况,输入的q值应为扩域后的q值. 维度n默认可整除基k,若出现不能整除的情况即方案4的情况,输入的n值应为扩域后的n值.

算法1的核心与基础是$\mathrm{NTT}_k(*)$和$\mathrm{INTT}_k(*)$,也就是基为k的NTT正变换及其逆变换. 基本Schoolbook方法核心运算为模乘累加,该运算已包含在NTT变换中,换言之能实现NTT变换的结构一定能实现Schoolbook乘法. 故而算法1是否能实现高能效关键在于基k的NTT的实现.

由于当前的后量子密码算法中提出的多项式维度n基本可以拆解为2的幂次或3的幂次,如若不可拆解也可通过扩域的方法(即2.1节所述方案3与方案4)解决. 4的幂次也是2的幂次,可通过基2结构重复构造. 而5的幂次及以上情况在当前算法中出现较少,且其运

算法1　基于NTT的快速多项式乘法算法

输入: 多项式$a(x),b(x)\in Z_q[x]/f(x)$,其中$f(x)$为模多项式,维度n,模数q满足$n\mid(q-1)$,基数k满足$n=k^l\times m$且$\exists\gamma^{2k^l}=\omega^{k^l}=1\bmod q$

输出: $c(x)\in Z_q[x]/f(x)$

1. $a(x)=a_0(x^m)+x\cdot a_1(x^m)+\cdots+x^{m-1}\cdot a_{m-1}(x^m),b(x)=b_0(x^m)+x\cdot b_1(x^m)+\cdots+x^{m-1}\cdot b_{m-1}(x^m),a_i(y),b_i(y)\in Z_q[y]/f(y^{n/4})$
2. for i=0 to m−1 do
3. $\widehat{a_i(y)}=\mathrm{NTT}_k(a_i(y)),\widehat{b_i(y)}=\mathrm{NTT}_k(b_i(y))$
4. end for
5. for i=0 to m−1 do
6. $\widehat{c_i(y)}=0$
7. for j=0 to m−1 do
8. $\widehat{c_i(y)}=\widehat{c_i(y)}+\widehat{a_j(y)}\odot\widehat{b_{(i-j)\bmod m}(y)}\odot\mathrm{NTT}_k(y)$
9. end for
10. end for
11. for i=0 to m−1 do
12. $c_i(y)=\mathrm{INTT}_k(\widehat{c_i(y)})$
13. end for
14. $c(x)=c_0(x^m)+x\cdot c_1(x^m)+\cdots+x^{m-1}\cdot c_{m-1}(x^m)$

算式与基4、基3、基2的网络相差较大,不利于重构实现. 故本节立足于基2、基3、基4三种情况提出一种高能效的混合基NTT算法.

经过数理等价变换,常规基2核心运算可以表述为$a=a+\omega b,b=a-\omega b$;基3核心运算可以表述为$a=a+\omega b+\omega^2 c$,　$b=(a-\omega^2 c)+\mu(\omega b-\omega^2 c)$,　$c=(a-\omega^2 c)-$

算法2 高能效混合基NTT算法(NTT_k)

输入：多项式$f=f_0+f_1\cdot x+\cdots+f_{N-1}\cdot x^{N-1}$

输出：NTT表示的多项式$F=F_0+F_1\cdot x+\cdots+F_{N-1}\cdot x^{N-1}$

1.$F_i=\mathrm{brv}(f_i)$,其中brv表示位反转操作
2. for $p=0$ to $\log_k N-1$ do
3. $J=k^p$
4. $\omega_m=\varphi_{2N}^{N/kJ}$
5. for $l=0$ to $N/(kJ)-1$ do
6. for $j=0$ to $J-1$ do
7. if $k=2$ then $A=F_{klJ+j}$; $B=0$; $C=F_{klJ+j+J}$; $D=0$; end if
8. if $k=3$ then$A=F_{klJ+j}$; $B=F_{klJ+j+J}$; $C=F_{klJ+j+2J}$; $D=F_{klJ+j+2J}$; end if
9. if $k=4$ then$A=F_{klJ+j}$; $B=F_{klJ+j+J}$; $C=F_{klJ+j+2J}$; $D=F_{klJ+j+3J}$; end if
10. $t_0=(A+\omega_2C)+(\omega_1B+\omega_3D)$
11. $t_1=(A-\omega_2C)+\mu_1(\omega_1B-\omega_3D)$
12. $t_2=(A+\omega_2C)-(\omega_1B+\omega_3D)$
13. $t_3=(A-\omega_2C)-\mu_2(\omega_1B-\omega_3D)$
14. $t_4=(A+\omega_2C)+\omega_1B$
15. if $k=2$ then $F_{klJ+j}=t_0$; $F_{klJ+j+J}=t_1$; end if
16. if $k=3$ then $F_{klJ+j}=t_4$; $F_{klJ+j+J}=t_1$; $F_{klJ+j+2J}=t_3$;end if
17. if $k=4$ then $F_{klJ+j}=t_0$; $F_{klJ+j+J}=t_1$; $F_{klJ+j+2J}=t_2$; $F_{klJ+j+3J}=t_3$; end if
18. end for
19. end for
20. end for

算法3 高能效混合基INTT算法(INTT_k)

输入：NTT表示的多项式$F=F_0+F_1\cdot x+\cdots+F_{N-1}\cdot x^{N-1}$

输出：多项式$f=f_0+f_1\cdot x+\cdots+f_{N-1}\cdot x^{N-1}$

1. for $p=\log_k N-1$ to 0 do
2. $J=k^p$
3. $\omega_m=\varphi_{2N}^{-N/kJ}$
4. for $l=0$ to $N/(kJ)-1$ do
5. for $j=0$ to $J-1$do
6. if $k=2$ then $A=F_{klJ+j}$; $B=0$; $C=F_{klJ+j+J}$; $D=0$; end if
7. if $k=3$ then $A=F_{klJ+j}$; $B=F_{klJ+j+J}$; $C=F_{klJ+j+2J}$; $D=F_{klJ+j+2J}$; end if
8. if $k=4$ then $A=F_{klJ+j}$; $B=F_{klJ+j+J}$; $C=F_{klJ+j+2J}$; $D=F_{klJ+j+3J}$; end if
9. $t_0=\frac{1}{k}((A+C)+(B+D))$
10. $t_1=\frac{1}{k}\omega_1^{-1}((A-C)+\mu_1{}^{-1}(B-D))$
11. $t_2=\frac{1}{k}\omega_2^{-1}((A+C)-(B+D))$
12. $t_3=\frac{1}{k}\omega_3^{-1}((A-C)-\mu_2{}^{-1}(B-D))$
13. $t_4=\frac{1}{k}((A+C)+B)$
14. if $k=2$ then $F_{klJ+j}=t_0$; $F_{klJ+j+J}=t_1$; end if
15. if $k=3$ then $F_{klJ+j}=t_4$; $F_{klJ+j+J}=t_1$; $F_{klJ+j+2J}=t_3$; end if
16. if $k=4$ then $F_{klJ+j}=t_0$; $F_{klJ+j+J}=t_1$; $F_{klJ+j+2J}=t_2$; $F_{klJ+j+3J}=t_3$; end if
17. end for
18. end for
19. end for
20.$f\leftarrow\mathrm{brv}(F)$,其中brv表示位反转操作

$(\mu+1)(\omega b-\omega^2c)$；基4核心运算可以表述为$a=(a+\omega^2c)+(\omega b+\omega^3d)$，$b=(a-\omega^2c)+\mu(\omega b-\omega^3d)$，$c=(a+\omega^2c)-(\omega b+\omega^2d)$，$d=(a-\omega^2c)-\mu(\omega b-\omega^3d)$；观察可知，基3和基4的核心运算均可由基2的核心运算组合而成，而基3与基4的核心运算具有极其类似的表达式形式，故而提出混合基的NTT算法如算法2所示．同理提出混合基的INTT算法如算法3所示．

算法2给出了混合基NTT算法的形式化表示，观察算法2，第7~9行及第15~17行在具体硬件实现过程中可直接对应输入输出选择电路，由基数进行数选控制．核心运算结构对应算法2第10~14行．进一步分析可发现，ω_1、ω_2、ω_3、μ_1、μ_2均为旋转因子，可通过预计算得到，故运算加速网络的输入数据为A、B、C、D四项，输出为t_0~t_4五项．又因为，t_4与t_0计算形式类似，若在计算t_0时旁路与ω_3D的加法，则t_4与t_0表述一致．根据以上分析，混合基的NTT网络如图1所示．图1中，红色标识线路为基2与基4网络，基为2时输出t_0~t_1，基为4时输出t_0~t_3．蓝色标识线路为基3网络，此时输出t_4、t_1和t_3．

与算法2同理，算法3给出了混合基INTT算法的形式化表示．算法3中t_4与t_0计算形式类似，若在计算t_4时旁路与D的加法，则t_4与t_0表述一致．根据以上分析，本文提出混合基的INTT网络如图2所示．图2中，红色标识线路为基2与基4网络，基为2时输出t_0~t_1，基为4时输出t_0~t_3．蓝色标识线路为基3网络，此时输出t_4、t_1和t_3．

如图1与图2所示，本文提出的混合基NTT/INTT算法，其关键路径延迟与常规基3、基4的NTT变换延迟持平．其运算部件数量与常规基4的NTT变换持平，较基3的NTT变换仅增加了3个模加法器．虽然该算法实现基2的NTT/INTT变换时资源利用率较低，关键延迟是原始算法的两倍，但在具体实现过程中，路径延迟可通过旁路寄存等方式降低，资源利用率也可通过并行复用等方式提高．综合而言，本文提出的混合基NTT/INTT算法在不影响关键路径延迟的基础上提高了硬件资源利用率，从而提高了基2、基3与基4的NTT/INTT变换的综合能效．

3 高能效可重构网络拓扑结构研究与设计

观察图1与图2，两者网络的区别主要在于前后三个模乘法器的位置不同，且INTT网络在输出前需经一步除法操作．前者在硬件结构中可以通过输入输出选择对三个模乘法器进行复用，后者可通过旁路选择进行重构．故而，可重构的混合基多项式乘法网络结构如图3所示．图3中，虚线框中为INTT特有结构，实线框

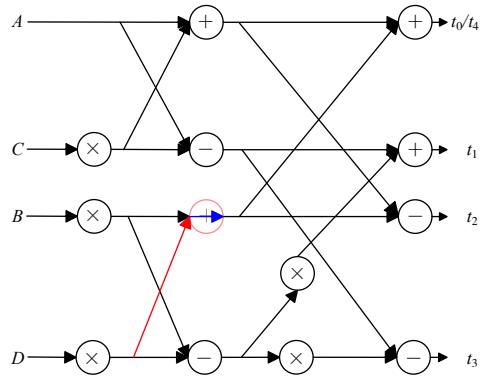


图1　混合基NTT网络

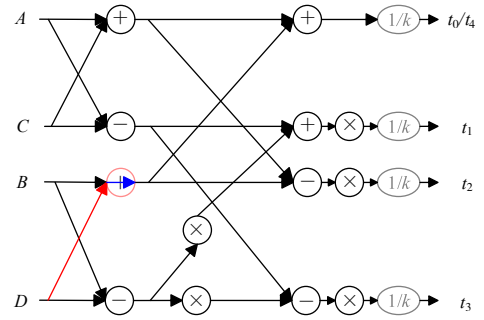


图2　混合基INTT网络

中为NTT特有结构,红色标识为基2、基4网络特有结构,蓝色标识为基3网络特有结构.

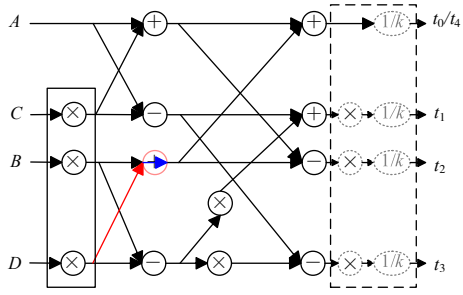


图3　可重构的混合基多项式乘法网络

图3所示网络按照算法2及算法3所示流程执行基2的NTT变换时,网络利用率不到整个结构的1/4,硬件结构利用率较低. 而实际上,若调整基2网络的输出位置提前到第二级模乘法器之前,则A、C输入与B、D输入可分别实现一个基2的蝶形变换,可将网络的利用率提高到1/2. 虽然利用率仍未达到基3、基4网络的较高水平,但考虑到当前的后量子密码算法中,大多数可以分解为4的幂次,少数只能分解为2的幂次的也可以采用不完整的NTT进行分解后按基4网络进行运算. 因此,本文不再特别对基2的网络进行优化.

如图3所示网络中使用了五个模乘法器构造相应的结构,最长关键路径为两个模乘法器和两个模加法器. 众所周知,在FFT中,基4蝶形变换的第二阶段乘法可以视作复数实部与虚部的简单位置和符号变换,故而省去了第二级乘法,实现了速度的成倍提升. NTT虽然与FFT具有类似的蝶形网络,但并不具有这种特性,基4的NTT算法实际上仅仅是4个基2蝶形的叠加,运算过程能效不变. 本文研究发现,基4的NTT算法无法实现性能成倍提升的关键在于蝶形网络中第二阶段的旋转因子乘(μ_1、μ_2),这一步操作带来了两个模乘法器的关键路径延迟. 由于旋转因子可预计算的特性,可将这一步乘法向前提取到第一阶段,前向提取后的旋转因子即可完美与第一阶段的旋转因子相融合,增加了一步乘法操作,实现了近乎两倍的计算速度的提升.

经上述方法优化后,基2的正向NTT核心运算可以表述为$a=a+\omega b, b=a-\omega b$;基3的正向NTT核心运算可以表述为$a=a+\omega b+\omega^2 c$, $b=a+\mu\omega b-\omega^2 c-\mu\omega^2 c$, $c=a-\mu\omega b-\omega b+\mu\omega^2 c$;基4的正向NTT核心运算可以表述为$a=a+\omega b+\omega^2 c+\omega^3 d$, $b=a+\mu\omega b-\omega^2 c-\mu\omega^3 d$, $c=a-\omega b+\omega^2 c-\omega^2 d, d=a-\mu\omega b-\omega^2 c+\mu\omega^3 d$. 同理,优化后基2的逆向NTT核心运算可以表述为$a=1/2(a+b), b=1/2\,\omega^{-1}(a-b)$;基3的逆向NTT核心运算可以表述为$a=1/3(a+b+c)$, $b=1/3\omega^{-1}\left(a-b-\mu^{-1}b+\mu^{-1}c\right)$, $c=1/3\omega^{-2}\left(a-c+\mu^{-1}b-\mu^{-1}c\right)$;基4的逆向NTT核心运算可以表述为$a=1/4(a+b+c+d)$, $b=1/4\,\omega^{-1}\left(a+\mu^{-1}b-c-\mu^{-1}d\right)$, $c=1/4\,\omega^{-2}(a-b+c-d)$, $d=1/4\,\omega^{-3}\left(a-\mu^{-1}b-c+\mu^{-1}d\right)$. 可以发现,基3的正向NTT核心运算中若预计算$\mu\omega$、$\mu\omega^2$则共需进行4次乘法,基4的正向NTT核心运算中若预计算$\mu\omega$、$\mu\omega^3$,则共需进行5次乘法,同理,基3的逆向NTT核心运算共需进行4次乘法,基4的逆向NTT核心运算共需进行5次乘法. 图3所示网络中恰好使用了5个模乘法器,故本文所提的优化方法并未增加额外的运算部件. 优化后的加速网络如图4所示. 图4中虚线框中为INTT特有结构,实线框中为NTT特有结构,红色标识为基2、基4网络特有结构,蓝色标识为基3网络特有结构.

考虑到不同模域的运算有效位宽差距较大,为了

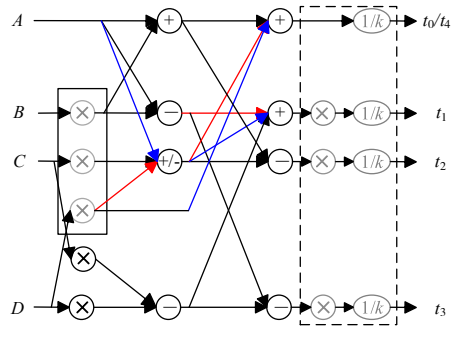


图4　高能效可重构多项式乘法加速网络

达到更高的资源利用率,对基本运算器进行优化. 采用16位的模乘法器和模加法器拼接实现32位的模乘法器和模加法器,从而使得一个运算器即可以实现一次32位运算,也可以并行实现2次16位运算. 本文提出的高能效可重构多项式乘法加速网络硬件结构如图5所示.

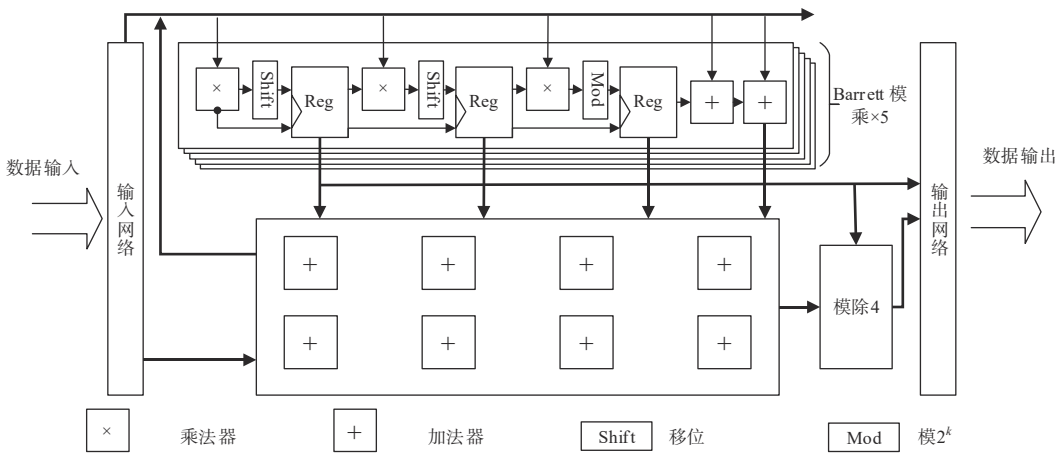


图5 高能效可重构多项式乘法加速网络硬件结构

4 高能效无冲突内存访问方案

NTT/INTT网络结构一次运算需要同时读入读出多组不同地址的数据,且一次性读出数据随着并行度的提高成倍增加. 分bank的存储结构可以保证一次性读写数据的需求,但受限于基于RAM的存储体进出数据大小的限制,在NTT/INTT执行过程同一时间需要的操作数不可存储于同一bank中. 一种最显而易见的方式是复制一套同样的存储结构进行前后运算转换,但这种方式直接带来两倍的存储消耗且地址生成逻辑同样复杂. 因此,研究NTT运算的无冲突原地内存访问方案是高能效多项式乘法硬件实现需要解决的一个关键问题.

本文提出的运算加速网络一次性读写数据量为4个,故而基础bank数量为4. 首先,考虑基3的内存管理方案. 基3的NTT变换一次性读写3个数据,数据地址跨度为3^n,假设一次NTT变化读取的第一个数据存储在地址a中,则需要同时读取的另外两个数据分别存储在$a+3^n$、$a+2\cdot 3^n$中. 若数据在4个bank中按顺序存储,则上述三个数据的bank索引分别为$a \bmod 4$、$\left(a+3^n\right)\bmod 4$、$\left(a+2\cdot 3^n\right)\bmod 4$. 由于$3^n \bmod 4\neq 0$与$2\cdot 3^n \bmod 4\neq 0$恒成立,故$a \bmod 4\neq \left(a+3^n\right)\bmod 4\neq \left(\left(a+3^n\right)+3^n\right)\bmod 4$恒成立. 所以,当基为3时,在4个bank内按顺序存储不会产生访存空间冲突. 此理论可进一步推广为,只要基数与bank块数互质,则数据按顺序存储不会产生访存空间冲突.

基为2或基为4时,基数是bank块数的因数,若不限制多项式的维度,则一定在某一NTT变换层级存在操作数存储于同一bank的情况,从而发生访存空间冲突. 在已有研究中,文献[10]考虑了蝶形单元的并行度,提出了任意基的地址映射方案,是当前能效最高的解决方法. 但是,文献[10]提出的算法仅针对一种基数的情况进行设计,当基数不同时,所采用的地址映射方法也不同. 如果直接应用在混合基的NTT中,需要对应多套地址生成逻辑,地址生成部分资源消耗明显且硬件利用率较低. 受文献[10]思想的启发,本文针对基2、基3、基4混合的NTT加速网络提出了一种高能效无冲突内存访问方案,描述如算法4所示.

算法4 混合基NTT的高能效无冲突存访问方案

输入: 原始地址ADDR_orign=$\{a_{n-1},\cdots,a_1,a_0\}$,{ }表示按二进制位拼接,基数$R$

输出: 映射地址ADDR_new=$\{\text{Bank\_addr},\text{Bank\_index}\}$

1. Bank_addr=ADDR_orign$\gg 2$
2. Bank_index=ADDR_orign mod 4
3. $a_{\text{even}}=\sum_{1}^{n/2-1} a_{2i} \bmod 4$
4. $a_{odd}=\sum_{1}^{n/2-1} a_{2i+1} \bmod 4$
5. if $R\neq 3$ then Bank_index=$\left(2a_{\text{odd}}+a_{\text{even}}+2a_1+a_0\right)\bmod 4$
6. if $R\neq 4$ then
   Bank_index=$\left(\text{Bank\_index}+a_{\text{even}}\right)\bmod 4$
7. end if
8. end if

通过算法4可以看到,我们提出的无冲突内存访问方案只需要固定的加法与选择逻辑,即可实现基2、基3和基4三种情况下的访存,这种设计可以同时使用多种

基数的NTT来实现多项式乘法. 为方案3和4的应用提供了更高效的支撑. 图6以16点NTT变换为例,详细地介绍了内存映射过程.

	R=3					R=4			R=2	
Orign	Index	Addr	Odd	Even		Index	Addr		Index	Addr
0000	00	00	0	0	00+00=000	00	00	00+0=000	00	00
0001	01	00	0	0	00+01=001	01	00	01+0=001	01	00
0010	10	00	0	0	00+10=010	10	00	10+0=010	10	00
0011	11	00	0	0	00+11=011	11	00	11+0=011	11	00
0100	00	01	0	1	01+00=001	01	01	01+1=001	01	01
0101	01	01	0	1	01+01=010	10	01	10+1=010	10	01
0110	10	01	0	1	01+10=011	11	01	11+1=100	00	01
0111	11	01	0	1	01+11=100	00	01	00+1=001	01	01
1000	00	10	1	0	10+00=010	10	10	10+0=010	10	10
1001	01	10	1	0	10+01=011	11	10	11+0=011	11	10
1010	10	10	1	0	10+10=100	00	10	00+0=000	00	10
1011	11	10	1	0	10+11=101	01	10	01+0=001	01	10
1100	00	11	1	1	11+00=011	11	11	11+1=100	00	11
1101	01	11	1	1	11+01=100	00	11	00+1=001	01	11
1110	10	11	1	1	11+10=101	01	11	01+1=010	10	11
1111	11	11	1	1	11+11=110	10	11	10+1=011	11	11

图6　16点NTT变换无冲突内存访问方案

图6中,第一列为16点NTT的原始地址,由于我们将存储器分为4个bank,所以将原始地址的低位与高位拆分. 对于基为3的情况,不会在4个bank的存储其中产生冲突,所以低位即bank索引地址,高位为实际bank内地址. 对于基为2和4的情况,会产生存储冲突,应用算法4,将原始地址高位按奇偶拆分并计算汉明权重,即第3、4列所示. 第5列应用算法4第5行计算了新的索引值,bank内地址不变,如第6列所示. 第7列应用算法4第6行计算基2情况下的索引值,与bank内地址共同组成基2时的新地址,如第8列所示. 可以看到,基数分别为3、4、2时,按阴影部分划分情况,每个数据都存放在不同的bank中,不会发生访存冲突.

根据算法4,本文设计地址生成逻辑结构如图7虚框内所示,除开运算中的地址生成,将数据存入和读出也需要这套转换逻辑. 然而,当直接执行算法2和3时,需要在NTT的开始和INTT的结束分别进行位反转操作. 这是因为NTT的设计均以位反顺序输入,以自然顺序输出,INTT正好相反. 通过重排旋转因子的输入过程,可以避免位反转操作. 文献[10]将这种操作扩展到了基4的NTT/INTT中. 这种思路从算法层面简化了操作,但同时带来了旋转因子使用的不规律性,因此在这些研究中,旋转因子必须预先计算后存储在一个固定空间中. 这种方法预计算的量级过大,且可扩展性不强. 对于大点数的变换而言,受限于固定空间大小,极有可能在中途需要进行一次或多次旋转因子计算;对于小点数的变换而言,固定空间不能被充分利用,导致更大的冗余空间及功耗. 考虑到位反转操作只在算法初始或结尾时出现一次,从硬件角度上考虑,可以通过分bank的存储结构,将位反转操作直接并入输入输出的地址生成逻辑中. 由于地址生成逻辑无论是否进行位反转都必然存在,故合并位反操作的硬件和时间代价极低. 因此,本文选择将位反操作融入输入输出地址生成逻辑中,如图7虚线框外部分所示,与前文的地址转换结构相结合共同组成地址生成逻辑结构.

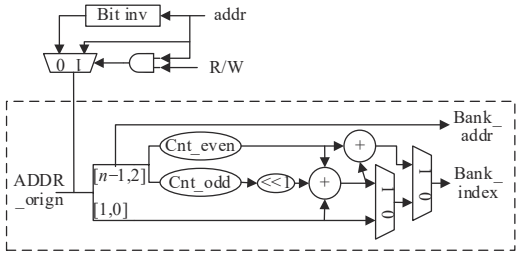


图7　地址转换逻辑结构图

5　综合验证与对比分析

为对本文提出的可重构多项式乘法加速网络进行评估,我们使用Verilog HDL对可重构多项式乘法加速网络进行描述,在CMOS 55纳米标准单元库工艺下进行逻辑综合,电路的最高工作频率可达到476 MHz. 利用PTPX工具进行功耗测试,平均功耗为83.6 mW. 此外,本文针对FPGA实现方式,选择Vivado 2019.1软件分别基于Artix-7、Virtex-7、UltraScale+及Zynq-7000四类FPGA芯片进行仿真综合,最高工作频率分别可达到196 MHz、303 MHz、286 MHz和256 MHz.

表2列出了近三年来较为典型的快速多项式乘法在FPGA上的实现研究. 在参数n为256,参数q小于16 bit的情况下(例如Kyber中的多项式乘法),本文提出的加速网络速度分别是文献[11]、文献[12]和文献[13]的2.82、5.17和2.47倍. ATP值较文献[11]和文献[12]具有明显优势. 由于本文的设计可支持的算法参数种类比文献[13]更多,故而面积开销较大,致使ATP值略高. 但同时也表明本文的设计仅牺牲约10%的ATP值即可换取更强大的灵活性.

在参数n为1 024,参数q小于16 bit的情况下,本文提出的加速网络速度是文献[10]的2.19倍. 文献[10]也是支持单一算法的设计,它仅支持q值在14 bit以内的多项式乘法,定制化程度更高,故而面积开销小,致使ATP值低于本文提出的网络结构.

在参数n为256,参数q小于32 bit的情况下(例如Dilithium中的多项式乘法),本文提出的网络结构速度分别是文献[15]和文献[17]的3.79倍和1.63倍,且LUT的ATP值均低于文献[15]和文献[17]. 文献[16]在速度方面是本文的近2倍,但资源消耗大,ATP值是本文的近4倍. 文献[14]同时实现了Kyber和Dilithium中的多项式乘法,时钟周期数较少但资源消耗大,由于文献内未体现在FPGA上实现的主频,ATP的比较不具备实际意义.

在参数n为1 024,参数q小于32 bit的情况下(例如Dilithium中安全级别5的多项式乘法),与文献[18]相

表2 多项式乘法在FPGA上的能效对比

文献	年份	工艺	参数 n	参数 q	频率/MHz	时钟周期数	时延/μs	LUT/FF/DSP/BRAMs	ATP		
									LUT/K	FF/K	DSP
文献[11]	2022	UltraScale+	256	$<2^{13}$	250	932	3.728	3 541/2 467/—/—	13.20	9.20	—
文献[12]	2021	Artix-7	256	3 329	161	1 600	9.937	1 737/1 167/2/3	17.26	11.59	19.87
文献[13]	2021	Artix-7	256	3 329	182	864	4.75	2 543/792/4/9	12.08	3.76	19
本文		UltraScale+	256	$<2^{16}$	286	377	1.32	5 975/3 350/45/6	7.89	4.42	59.4
		Artix-7			196	377	1.922	7 060/4 670/45/6	13.57	8.98	86.49
文献[10]	2022	Virtex-7	1 024	$<2^{14}$	270	3 924*	14.53	1 196/969/12/3	17.38	14.08	174.36
本文		Virtex-7	1 024	$<2^{16}$	303	2 010	6.63	6 520/3 830/45/6	43.23	25.39	298.35
文献[14]	2022	Artix-7	256	8 380 417	540**	128	0.237**	25 674/3 137/64/6	6.08	0.74	15.17
文献[15]	2021	Artix-7	256	8 380 417	216	3 510*	16.25	2 044/—/16/6(18k)	33.22	—	260
文献[16]	2023	Artix-7	256	$<2^{24}$	135	355	2.63	44 800/34 500/128/16	117.82	90.74	336.64
文献[17]	2021	Artix-7	256	$<2^{32}$	232	1 627	7.00	4 780/—/16/24	33.46	—	112
本文		Artix-7	256	$<2^{32}$	196	841	4.29	7 060/4 670/45/6	30.29	20.03	193.05
文献[18]	2022	Virtex-7	1 024	$<2^{32}$	250	2 075	8.3	10 272/6 704/80/87	85.26	55.64	664
本文		Virtex-7	1 024	$<2^{32}$	303	4 066	13.42	6 520/3 830/45/6	87.5	51.4	603.9
文献[19]	2022	Zynq-7000	701	8 192	130	5 507	42.362	3 963/3 389/45/—	167.88	143.56	1 906.29
本文		Zynq	701	8 192	256	7 240	28.236	7 060/4 510/45/6	199.346	127.344	1 270.62

注：由于部分文献只给出了NTT相关数据，为便于比较，以3倍时钟周期数近似一次多项式乘法的时钟周期；文献没有给出明确的FPGA实现的频率数据，540MHz为TSMC 28nm工艺下的综合频率.

比，本文提出的网络结构速度降低了38%，资源消耗较低，ATP值与文献[18]持平，但文献[18]只针对q值为32 bit的NTT/INTT进行加速，即使在q值不足32位的情况下，仍然按32位执行，对小模数情况并不友好，所以当执行Kyber中的多项式乘法时，本文结构仅需6.63 μs，较文献[18]的8.3 μs提升了20%，ATP值仅约为文献[18]的50%.

在参数n为701，参数q为8 192的情况下（即NTRU701），这种情况是应用扩域方法及混合基方法进行加速的代表算法. 与文献[19]相比，本文LUT的ATP值高于文献[19]约16%，FF与DSP的ATP值分别低于文献[19]约13%和50%.

综合上述分析结果，当前针对NTT的FPGA实现大多数只针对一种参数情况进行优化，通常可以以较高的能效实现其中一种或一类后量子密码算法的加速，实现其他算法时效率将大大降低或者不可实现. 本文提出的设计在多种参数配置情况下均可达到较高的能效，结果表明，本文的设计在多算法可重构前提下实现了高能效.

在现有后量子密码算法硬件实现的论文中，单独针对多项式乘法单元进行ASIC实现的工作较少，表3列出了三篇典型的快速多项式乘法在ASIC上的实现研究. 与文献[20]相比，本文提出的结构速度提升高达60余倍，一方面文献[20]是低功耗的设计，面积和主频均处于较低水平，另一方面文献[20]是针对完整后量子密码算法的设计，主频也受到其他功能部件的影响，但表3中的资源数据仅来源于多项式乘法部件. 从呈现的数据来看，本文的ATP值远低于文献[20]，能效值亦远高于文献[20]，即使将文献[20]的主频提升到与本文相同的476 MHz，其ATP值为154.36，能效值为0.013，本文的ATP值和能效值依然略有优势.

表3 不同多项式乘法架构在ASIC下的能效对比

文献	年份	支持算法	频率/MHz	时钟周期数	时延/μs	工艺/nm	n	q	资源	功耗/mW	ATP	能效/(μs·mW)$^{-1}$
Sapphire[20]	2019	NewHope、Kyber、Dilithium、Frodo	72	3 867*	53.71	40	256	7 681	19 kGE	9.27	1 020.49	0.002
VPQC[21]	2020	NewHope、Kyber、LAC	300	123*	0.41	28	256	3 329	521 kGE	164	213.61	0.015
MENTT[22]	2022	Kyber、Dilithium	151	3 171	23	65	256	2^{14}	0.36 mm^2	6.26	8.28	0.007
本文	—	Kyber、Dilithium、Saber、NTRU、LAC、NewHope…	476	374	0.785	55	256	2^{16}	0.35 mm^2	83.6	0.275	0.015

注：由于文献只给出了NTT相关数据，为便于比较，以3倍时钟周期数近似一次多项式乘法的时钟周期.

文献[21]针对NTT/INTT进行并行化设计,得到了近2倍于本文的速度,但同时并行度高带来的资源消耗大,其ATP值反而高出本文约40%. 本文在保持能效值与之持平的情况下,增加了可重构实现的算法种类.

对比文献[22],其消耗的资源与本文接近,但完成一次多项式乘法运算耗时是本文的29倍,ATP值也高于本文约30倍,虽然本文的功耗比之高出约13倍,但由于性能高,本文的能效反而比之高约2倍.

综上所述,本文的ATP值与三篇文献相比都具有明显优势,表明本文的设计具有较高的面积效率,本文的能效值也远高于文献[20,22],与文献[21]持平. 综合来看,本文提出的可重构多项式乘法加速网络在保持各类多项式乘法算法效率的前提下,增加了可重构实现算法种类,提高了运算加速单元的灵活性,取得了较高的综合能效.

6　结论

本文以高能效为目标,对现有的格基后量子密码算法中的多项式乘法运算展开研究. 提出了一种普适性的基于NTT的快速多项式乘法算法,可实现所有基于有限域上的多项式乘法运算加速. 进而,本文针对其核心运算过程,提出了高能效混合基的NTT算法和INTT算法. 在此基础上,对混合基的多项式乘法算法的运算结构展开研究,从硬件实现的角度进行能效优化,提出了一种高度可重构的统一化多项式乘法加速网络,在不增加额外运算部件的前提下,将基3与基4NTT运算效率提升了一倍. 对NTT运算过程中的存储冲突问题展开研究,以数学方法分析了冲突的根本原因,并提出了高能效混合基的内存管理方案. 本文提出的结构在保持各类多项式乘法算法效率的前提下,增加了可重构实现算法种类,提高了运算加速单元的灵活性,取得了较高的综合能效.

参考文献

[1] LYUBASHEVSKY V, MICCIANCIO D, PEIKERT C, et al. SWIFFT: A modest proposal for FFT hashing[C]//International Workshop on Fast Software Encryption. Berlin: Springer, 2008: 54-72.

[2] GÖTTERT N, FELLER T, SCHNEIDER M, et al. On the design of hardware building blocks for modern lattice-based encryption schemes[C]//International Workshop on Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2012: 512-529.

[3] FRITZMANN T, SEPULVEDA J. Efficient and flexible low-power NTT for lattice-based cryptography[C]//IEEE International Symposium on Hardware Oriented Security and Trust. Piscataway: IEEE, 2019: 141-150.

[4] DU C H, BAI G Q. Towards efficient polynomial multiplication for lattice-based cryptography[C]//IEEE International Symposium on Circuits and Systems. Piscataway: IEEE, 2016: 1178-1181.

[5] SHOR P W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer[J]. SIAM Journal on Computing, 1997, 26(5): 1484-1509.

[6] GÜNEYSU T, LYUBASHEVSKY V, PÖPPELMANN T. Practical lattice-based cryptography: A signature scheme for embedded systems[C]//International Workshop on Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2012: 530-547.

[7] LIU W Q, FAN S L, KHALID A, et al. Optimized schoolbook polynomial multiplication for compact lattice-based cryptography on FPGA[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2019, 27(10): 2459-2463.

[8] CHUNG C M M, HWANG V, KANNWISCHER M J, et al. NTT multiplication for NTT-unfriendly rings[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2021, 2021: 159-188.

[9] BERNSTEIN D J, SORENSON J P. Modular exponentiation via the explicit Chinese remainder theorem[J]. Mathematics of Computation, 2007, 76(257): 443-455.

[10] CHEN X R, YANG B H, YIN S Y, et al. CFNTT: Scalable radix-2/4 NTT multiplication architecture with an efficient conflict-free memory mapping scheme[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2021(1): 94-126.

[11] LI A B, LIU D S, LI X, et al. A flexible instruction-based post-quantum cryptographic processor with modulus reconfigurable arithmetic unit for module LWR&E[C]// 2022 IEEE Asian Solid-State Circuits Conference (A-SSCC). Piscataway: IEEE, 2022: 1-3.

[12] XING Y, LI S. A compact hardware implementation of CCA-secure key exchange mechanism CRYSTALS-KYBER on FPGA[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2021(2): 328-356.

[13] YAMAN F, MERT A C, ÖZTÜRK E, et al. A hardware accelerator for polynomial multiplication operation of CRYSTALS-KYBER PQC scheme[C]//2021 Design, Automation & Test in Europe Conference & Exhibition. Piscataway: IEEE, 2021: 1020-1025.

[14] ZHAO Y, XIE R, XIN G, et al. A high-performance domain-specific processor with matrix extension of RISC-V

for module-LWE applications[J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2022, 69(7): 2871-2884.

[15] ZHOU Z, HE D, LIU Z, et al. A software/hardware co-design of crystals-dilithium signature scheme[J]. ACM Transactions on Reconfigurable Technology and Systems, 2021, 14(2): 1-21.

[16] HU X, TIAN J, LI M H, et al. AC-PM: An area-efficient and configurable polynomial multiplier for lattice based cryptography[J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2023, 70(2): 719-732.

[17] 刘冬生, 赵文定, 刘子龙, 等. 应用于格密码的可重构多通道数论变换硬件设计[J]. 电子与信息学报, 2022, 44(2): 566-572.
LIU D S, ZHAO W D, LIU Z L, et al. Reconfigurable hardware design of multi-lanes number theoretic transform for lattice-based cryptography[J]. Journal of Electronics & Information Technology, 2022, 44(2): 566-572. (in Chinese)

[18] SU Y, YANG B L, YANG C, et al. A highly unified reconfigurable multicore architecture to speed up NTT/INTT for homomorphic polynomial multiplication[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2022, 30(8): 993-1006.

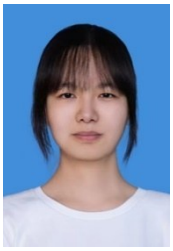
[19] DANG V B, MOHAJERANI K, GAJ K. High-speed hardware architectures and FPGA benchmarking of CRYSTALS-kyber, NTRU, and saber[J]. IEEE Transactions on Computers, 2023, 72(2): 306-320.

[20] BANERJEE U, UKYAB T S, CHANDRAKASAN A P. Sapphire: A configurable crypto-processor for post-quantum lattice-based protocols[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2019 2019: 17-61.

[21] XIN G Z, HAN J, YIN T Y, et al. VPQC: A domain-specific vector processor for post-quantum cryptography based on RISC-V architecture[J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2020, 67(8): 2672-2684.

[22] LI D, PAKALA A, YANG K Y. MeNTT: A compact and efficient processing-in-memory number theoretic transform (NTT) accelerator[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2022, 30(5): 579-588.

作者简介



别梦妮 女, 1997年2月出生于湖北省荆州市. 现为信息工程大学计算机科学与技术专业博士研究生. 主要研究方向为后量子密码处理器设计.
E-mail: raspberry0213@126.com

李 伟 男, 1983年11月出生于天津市. 现为信息工程大学教授. 主要研究方向为体系结构、安全芯片设计、集成电路技术.
E-mail: try_1118@163.com