(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 119968610 A
(43)申请公布日 2025.05.09

(21)申请号 202380070569.2
(22)申请日 2023.08.23
(30)优先权数据
22192305.5 2022.08.26 EP
(85)PCT国际申请进入国家阶段日
2025.04.01
(86)PCT国际申请的申请数据
PCT/EP2023/073098 2023.08.23
(87)PCT国际申请的公布数据
WO2024/042113 EN 2024.02.29
(71)申请人 鲁汶天主教大学
地址 比利时鲁汶
(72)发明人 扬-彼得·德安弗斯
米希尔·范贝伦多克
(74)专利代理机构 成都超凡明远知识产权代理有限公司 51258
专利代理师 卫兰英
(51)Int.Cl.
G06F 7/499(2006.01)
G06F 7/523(2006.01)

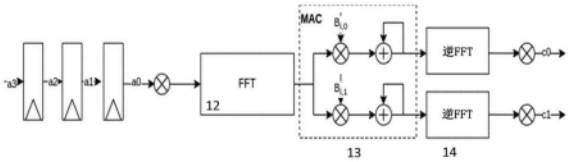
权利要求书2页 说明书12页 附图3页

(54)发明名称

设计用于加密应用的设备的方法

(57)摘要

本发明涉及一种设计用于在加密应用中执行多项式的乘法的设备的方法。该方法包括:-将对多项式的乘法的执行划分到设备的至少两个数据处理阶段上,其中,至少一个数据处理阶段被配置成接收输入操作数以执行乘法,并且至少一个数据处理阶段被配置成提供对乘法的输出信号,-针对每个数据处理阶段限定与该数据处理阶段要处理的数据的表示相关的一个或更多个参数,-针对输出信号限定一个或更多个约束,-针对每个数据处理阶段单独地确定针对一个或更多个参数的值,其中考虑了所述一个或更多个约束,-将在所述数据处理阶段中的每个数据处理阶段中所确定的针对一个或更多个参数的值应用于在加密应用中执行多项式的乘法的所述设备中。



CN 119968610 A

1.一种设计用于在加密应用中执行多项式的乘法的设备的方法,所述方法包括:

-将对所述多项式的乘法的执行划分到所述设备的至少两个数据处理阶段上,其中,至少一个数据处理阶段被配置成接收输入操作数以执行所述乘法,并且至少一个数据处理阶段被配置成提供所述乘法的输出信号,

-针对每个数据处理阶段限定与所述数据处理阶段中要处理的数据的表示相关的一个或更多个参数,

-针对所述输出信号限定一个或更多个约束,

-针对每个数据处理阶段单独地确定针对所述一个或更多个参数的值,其中考虑了所述一个或更多个约束,

-将在所述数据处理阶段中的每个数据处理阶段中所确定的针对所述一个或更多个参数的值应用于在所述加密应用中执行所述多项式的乘法的所述设备中。

2.根据权利要求1所述的设计设备的方法,其中,所述输入操作数是多项式的向量。

3.根据权利要求1或2所述的设计设备的方法,其中,所述一个或更多个参数包括{位宽,动态范围,整数部分的大小,小数部分的大小,小数点的位置}中的一个或更多个。

4.根据前述权利要求中任一项所述的设计设备的方法,其中,所述设备是能重新配置的。

5.根据前述权利要求中任一项所述的设计设备的方法,所述方法包括将所述乘法分解成多个子任务的步骤,其中,每个数据处理阶段执行至少一个子任务。

6.根据前述权利要求中任一项所述的设计设备的方法,其中,所述设备具有包括形成流水线的多个连续阶段的架构。

7.根据权利要求6所述的设计设备的方法,其中,所述设备被应用在现场可编程门阵列中。

8.根据前述权利要求中任一项所述的设计设备的方法,其中,将最大噪声水平视为所述约束中的一个约束。

9.根据前述权利要求中任一项所述的设计设备的方法,其中,所述多项式的乘法是全同态加密方案的部分。

10.根据前述权利要求中任一项所述的设计设备的方法,其中,借助于模拟来执行所述针对每个数据处理阶段单独地确定针对所述一个或更多个参数的值的步骤。

11.根据权利要求1至9中任一项所述的设计设备的方法,其中,通过确定所述要处理的数据的一个或更多个属性,来执行所述针对每个数据处理阶段单独地确定所述值的步骤,确定所述要处理的数据的一个或更多个属性是基于对所述一个或更多个属性的所述一个或更多个约束造成影响的噪声源的模型并且/或者基于所述数据处理阶段的子任务的输入的至少一个属性来进行的。

12.根据权利要求11所述的设计设备的方法,其中,一个噪声源源于移除位于所述数据处理阶段的子任务的输入的最低有效位侧的位。

13.根据权利要求11或12所述的设计设备的方法,其中,一个噪声源源自丢弃位于所述数据处理阶段的子任务的所述输入的最高有效位侧的位。

14.根据前述权利要求中任一项所述的设计设备的方法,其中,使用负循环卷积来执行所述多项式的乘法。

15.根据前述权利要求中任一项所述的设计设备的方法,其中,所述数据的表示是定点表示。

设计用于加密应用的设备的方法

技术领域

[0001] 本发明总体上涉及加密的领域。更具体地,本发明涉及一种设计用于加密应用中的设备的方法。

背景技术

[0002] 在例如云计算等的背景下,用户将原始数据上传至不受信任的云服务器时会面临一定的风险。因此,需要提供足够的安全来保护用户的数据。在数据安全的领域中出现了一种有前途的新技术,即全同态加密(FHE),它允许人们在不了解有关所加密的数据(密文)的另外的信息的情况下对该数据执行同态计算。换言之,不需要首先对数据进行解密。多年来,执行FHE的方法已经得到改进,直至已经可以进行实际应用。

[0003] FHE算法通常在云计算服务器上执行。然而,计算速度较慢。典型地,对FHE方案的执行瓶颈在于乘法运算。在FHE方案中执行计算的密文数据是来自某个与方案相关的多项式环的大型多项式(长度为N)。这些多项式的典型运算包括加法和乘法。虽然加法与多项式的长度成线性关系(O(N)运算),但当使用通用的直接技术(也称为教科书式乘法)时,乘法具有二次成本(O(N^2)运算)。

[0004] FHE在实际应用中面临的主要挑战之一是其计算开销。由于FHE方案中成本最高的运算之一是乘法,因此提高乘法运算速度可能会非常有助于降低计算开销。这可以通过利用多项式的特定属性来实现。各种经过充分研究的算法可以用于加速这种乘法,包括数论变换(NTT)、图姆-库克乘法(Toom-Cook multiplication)、或Karatsuba乘法。使用NTT的乘法一般是这些算法中最有效的,但它也对所使用的多项式环施加了最严格的条件。因此,它不能用于每个FHE方案。

[0005] 当底层环结构允许NTT时,FHE方案典型地使用NTT进行快速多项式乘法。不应用NTT的两个值得注意的例外是在论文“FHEW:Bootstrapping homomorphic encryption in less than a second(FHEW:在不到一秒内引导同态加密)”(L.Ducas等人,Eurocrypt,第617-640页,2015年)中公开的FHEW方案和在“TFHE:Fast Fully Homomorphic Encryption Over the Torus(TFHE:环面上的快速全同态加密)”(I.Chillotti等人,J.Cryptol.33,34-91,2020年)中描述的TFHE方案。它们两者的环结构都禁止使用NTT。相反,这些方案可以使用Toom-Cook乘法、Karatsuba乘法、或快速傅里叶变换(FFT)进行快速多项式乘法,其中,典型的FFT是最快的选项。FFT变换类似于NTT变换,但对多项式环对FFT变换的条件要求没有那么严格。FHEW和TFHE两者都支持使用同态布尔代数,例如,除了同态加法和乘法之外,还有NAND逻辑门、XOR逻辑门、和XNOR逻辑门。

[0006] 为了安全目的,FHE密文将噪声包括在加密中。此外,每个FHE运算都会增加密文中存在的噪声。附加的FFT量化噪声在某种程度上是可以容忍的。量化噪声可能源于FFT使用实数工作并用浮点或定点表示近似的事实。只要不超过某个阈值噪声水平,FHE就可以容忍这种噪声。因此,FHE方案必须定期调用引导运算来降低密文中的噪声量,使得噪声量保持在阈值噪声水平以下。然而,过多的噪声会导致引导失败,并且因此,对FFT的使用需要非常

小心。

[0007] TFHE和FHEW等方案在每个加密门之后进行引导(bootstrap)。与前几代FHE方案相比,它们以更快的引导算法为特征,还附加地允许使用应用于密文的任意函数进行编程。加密的门计算(加密的NAND、XOR、XNOR等)主要由昂贵的(可编程)引导((P)BS)计算主导。因此,需要对引导应用硬件加速。

[0008] PBS涉及在所谓的"外积"中连续地乘以密文数据。该外积限定了General-LWE密文与General-GSW密文之间的乘法。带有误差的通用学习(LWE)和General-GSW(Gentry、Sahai、和Waters)都是完全同态加密方案。从概念上讲,General-LWE密文是k+1个多项式的向量;General-GSW密文是(k+1)|×(k+1)个多项式的矩阵。在TFHE中,这些多项式在实环面上被定义,即它们的系数位于区间(0,1]内。在FHEW中,这些多项式具有整数系数。

[0009] "外积"的第一步是将General-LWE密文中的每个多项式分解成I个多项式。然后,以分解格式的外积可以表示为多项式的向量矩阵乘法。输出也是General-LWE密文。

[0010] $$\underset{1\times(k+1)}{\left(c_0(X)\ c_1(X)\ \dots\ c_k(X)\right)} = \underset{1\times(k+1)l}{\left(a_0(X)\ a_1(X)\ \dots\ a_{(k+1)l-1}(X)\right)} \times \underset{(k+1)l\times(k+1)}{\begin{pmatrix} \mathcal{B}_{0,0}(X) & \dots & \mathcal{B}_{0,k}(X) \\ & \ddots & \\ \mathcal{B}_{kl,0}(X) & \dots & \mathcal{B}_{kl,k}(X) \end{pmatrix}} \quad (1)$$

[0011] 在这个向量矩阵乘法中,向量(分解后的GLWE密文)的维度为1×(k+1)|,矩阵(General-GSW密文)的维度为(k+1)|×(k+1),并且输出的维度为1×(k+1)。变量k和I是加密FHE参数。

[0012] 如已经提及的,FHE应用的重要特征是它们具有存在于密文数据中的固有加密数学噪声。当对密文数据解密时,噪声被舍入并检索到正确的结果。类似地,外积c(X)=(c_0(X(,c_1(X),...,c_k(X))的GLWE输出密文包含加密数学噪声。使用快速傅里叶变换(FFT)可以高效地计算多项式乘法。在这种情况下,由于这种固有的数学噪声,即使在加密设置中,也可以容忍c(X)中的附加的FFT量化噪声。可容忍的噪声量取决于加密参数集。例如,在上面所提及的I.Chillotti的论文中,推导出针对c(X)中存在的加密噪声的噪声公式。

[0013] 实数可以以多种方式用有限精度表示。在CPU上,典型的方法是使用单精度浮点数或双精度浮点数。精度由尾数的大小限定,并且动态范围由指数的大小限定。由于在CPU中对浮点单元(FPU)的集成,因此这种方法是有效的,并且因此它对于软件设计人员来说是典型的选定的表示方式。上面提及的TFHE和FHEW方案的实现已经仅限于双精度浮点FFT,因为发现单精度FFT会引入太多噪声。已经发现双精度浮点FFT将引入的噪声量保持在足够小的水平。定点表示由该表示中的位数和缩放因子决定。在定点表示中,尾数具有固定的位数。

[0014] 在硬件中,实现双精度算法的成本很高,并且因此如果不是绝对需要,优选地避免双精度算法。一种便宜得多的替代方案是定点算法。在最先进的解决方案中,选择单一表示并在整个架构中使用。此外,到目前为止,尚未利用加密设置中的输入数据的特定属性和针对输出处的噪声的特定要求。

[0015] 在论文"MATCHA:A Fast and Energy-Efficient Accelerator for Fully Homomorphic Encryption over the Torus(MATCHA:一种用于环面上全同态加密的快速节能的加速器)"(L.Jiang等人,第59届年度设计自动化会议,2022年7月,2022年2月17日预发布在互联网上)中,呈现了一种用于对TFHE门进行处理的硬件加速器,在效率方面,它优于

频繁调用成本昂贵的双精度浮点FFT和IFFT内核的加速器。为了充分利用对TFHE的容错能力,通过使用仅需要加法和二进制移位的近似无乘法整数FFT和IFFT来加速多项式乘法。虽然近似的FFT和IFFT会在每个密文中引入误差,但由于误差可以在解密期间与噪声一起四舍五入,因此仍然可以对密文进行正确地解密。整数表示可以看作是定点表示的经过缩放以移除小数点的版本。

[0016] 此外,在“MATCHA”方法中,为FFT和IFFT选择单一表示。通过对解密失败进行多次测试来完成验证噪声是否保持足够小。然而,在构成多项式乘法的步骤序列中,使用单一表示忽略了每个运算对输出噪声的贡献不同这一事实,这取决于所处理的数据。此外,将FFT噪声与数学外积噪声相匹配允许比仅对解密失败的测试更细粒度的噪声控制和优化。

[0017] 因此,设计用于加密应用中的设备的方法还有改进的空间。

发明内容

[0018] 本发明的实施方式的目的是提供设计用于在加密应用中执行多项式的乘法的设备的方法,其中,考虑到加密应用中数据的特殊性并且其中内置了一定程度的灵活性。

[0019] 上面的目的是通过根据本发明的解决方案来实现的。

[0020] 在第一方面,本发明涉及一种设计用于在加密应用中执行多项式的乘法的设备的方法。该方法首先将对多项式的乘法的执行划分到互连的至少两个数据处理阶段上,其中,至少一个数据处理阶段被配置成接收输入操作数以执行所述乘法,并且至少一个数据处理阶段被配置成提供所述乘法的输出信号。该方法还包括:

[0021] -针对每个数据处理阶段限定与该数据处理阶段要处理的数据的表示相关的一个或更多个参数,

[0022] -针对所述乘法的输出信号限定一个或更多个约束,

[0023] -针对每个数据处理阶段单独地确定针对所述一个或更多个参数的值,其中考虑了所述一个或更多个约束,

[0024] -将在所述数据处理阶段中的每个数据处理阶段中所确定的针对所述一个或更多个参数的值应用于在加密应用中执行多项式的乘法的所述设备中。

[0025] 所提出的解决方案确实允许设计用于执行多项式的乘法的设备的各个数据处理阶段,使得每个阶段都有其自己的、优选地优化的参数集。如此确定的参数集允许在给定表示类型(例如,定点或浮点)的情况下以最佳方式表示执行乘法时使用的变量,从而满足关于输出信号的一个或更多个约束。

[0026] 优选地,输入操作数是多项式的向量。第二输入操作数则是多项式的矩阵,其可以预先计算到FFT域中。

[0027] 在优选实施方式中,一个或更多个参数包括{位宽,动态范围,整数部分的大小,小数部分的大小,小数点的位置}中的一个或更多个。

[0028] 有利地,该设备是可重新配置的。

[0029] 在一个实施方式中,设计设备的方法包括将所述多项式乘法分解成多个子任务的步骤,其中,每个数据处理阶段执行至少一个子任务。

[0030] 在该方法的实施方式中,划分在至少两个数据处理阶段上取决于所述设备的硬件实现。在有利的实施方式中,该设备具有包括形成流水线的多个连续阶段的架构。然后,该

设备可以应用在现场可编程门阵列中。

[0031] 在优选实施方式中,将最大噪声水平视为所述约束中的一个约束。

[0032] 有利地,多项式的乘法是全同态加密方案的部分。

[0033] 在一些实施方式中,借助于模拟来执行所述针对每个数据处理阶段单独地确定参数值的步骤。

[0034] 在该方法的实施方式中,通过确定所述要处理的数据的一个或更多个属性,来执行针对每个数据处理阶段单独地确定所述值的步骤,确定所述要处理的数据的一个或更多个属性是基于对所述一个或更多个属性的所述一个或更多个约束造成影响的噪声源的模型并且/或者基于所述数据处理阶段的子任务的输入的至少一个属性来进行的。

[0035] 在一个实施方式中,一个噪声源可能源自移除位于上述数据处理阶段的子任务的输入的最低有效位侧的位。一个噪声源可能源自丢弃位于数据处理阶段的子任务的输入的最高有效位侧的位。

[0036] 在一些实施方式中,使用负循环卷积来执行多项式的乘法。

[0037] 在优选实施方式中,数据的表示是定点表示。

[0038] 为了概括本发明及其相对于现有技术的优势,上面对本发明的某些目的和优势进行了描述。当然,应理解的是,并非所有这样的目的或优势都可以根据本发明的任何特定实施方式实现。因此,例如,本领域技术人员将认识到,本发明可以以实现或优化如本文所教导的一个优势或一组优势的方式实施或执行,而不必实现本文所教导或建议的其他目的或优势。

[0039] 参照下文所描述的一个或更多个实施方式,本发明的上面和其他方面将变得明显并得以阐明。

附图说明

[0040] 现在将通过示例参照附图对本发明进行进一步描述,其中在各个图中相同的附图标记指代相同的元件。

[0041] 图1示出了可以在其上应用本发明的方法的加速器结构的框图。

[0042] 图2示出了图1的框图的展开版本。

[0043] 图3示出了FFT实现中常用的蝶形结构。

[0044] 图4示出了在第三示例中分解成子任务以及针对中间变量的附加的构建块。

[0045] 图5示出了执行乘法的三个步骤程序。

[0046] 图6示出了针对该设备的一种可能的替代架构。

具体实施方式

[0047] 本发明将针对特定实施方式并参照特定附图进行描述,但本发明并不限于此,而仅受权利要求的限制。

[0048] 此外,说明书和权利要求书中的术语“第一”、“第二”等用于区分类似元素,而不一定用于描述时间上、空间上、排序上、或任何其他方式的顺序。应理解的是,在适当情况下,如此使用的术语可以互换,并且本文中所描述的发明的实施方式能够以除本文中所描述的或所示的顺序以外的顺序进行运算。

[0049] 将注意的是,权利要求中使用的术语“包括”不应被解释为仅限于其后所列出的装置;它不排除其他元件或步骤。因此,应将其解释为指定所提及的所述特征、整体、步骤、或组件的存在,但并不排除一个或更多个其他特征、整体、步骤或组件或其组的存在或添加。因此,“一种包括装置A和装置B的设备”这一表述的范围不应限于仅包括组件A和组件B的设备。这意味着,就本发明而言,设备的相关组件仅是A和B。

[0050] 在整个说明书中,对“一个实施方式”或“实施方式”的引用意指结合该实施方式描述的特定特征、结构、或特性被包括在本发明的至少一个实施方式中。因此,在整个说明书中,短语“在一个实施方式中”或“在实施方式中”出现在各处并不一定都指代同一实施方式,但可以指同一实施方式。此外,如本领域的普通技术人员从本公开中明显理解的那样,特定特征、结构、或特性可以以任何合适的方式组合在一个或更多个实施方式中。

[0051] 类似地,应理解的是,在本发明的示例性实施方式的说明中,为了精简本公开内容并帮助理解一个或更多个不同的发明方面,本发明的各种特征有时会在单个实施方式、附图、或其描述中被组合在一起。然而,这种公开的方法不应被解释为反映所要求保护的发明需要比每个权利要求中明确叙述的更多的特征的意图。相反,如所附权利要求所反映的,发明方面在于比前述所公开的单个实施方式的所有特征少。因此,详细说明所附的权利要求在此明确地并入本详细说明中,其中每个权利要求其自身都作为本发明的单独实施方式。

[0052] 此外,尽管本文中所描述的一些实施方式包括了一些其他实施方式中未包括的其他特征,但不同的实施方式的特征的组合旨在本发明的范围内,并且形成不同的实施方式,如本领域技术人员所理解的。例如,在所附权利要求中,任何要求保护的实施方式可以以任何组合使用。

[0053] 应当注意的是,在描述本发明的某些特征或方面时使用特定术语不应被理解为暗示在本文中该术语被重新限定为仅限于包括与该术语相关联的本发明的特征或方面的任何特定特性。

[0054] 在本文中提供的说明中,阐述了许多具体细节。然而,可以理解的是,本发明的实施方式可以在没有这些具体细节的情况下实践。在其他情况下,为了不模糊对本说明的理解,未详细示出公知的方法、结构和技术。

[0055] 本发明公开了一种设计用于在加密应用中执行多项式乘法的设备的新颖的方法。

[0056] 为了实现执行FFT运算的多项式乘法的有效实现,可以利用上面提及的特征,即在FHE方案中,计算会将一定量的噪声引入到需要安全的密文中。因此,这些方案还可以容忍包括量化噪声的近似计算。还将注意的是,加密使用均匀分布的随机值工作,这限制了计算所基于的系数的动态范围。这使得对表示系数所需的必要位表示范围(即字长)进行预测变得更容易。加密应用中FFT所容忍的噪声取决于加密参数集,而不是取决于应用约束。可以将加密参数集选择成能够容忍更多或更少的噪声。发明人已经意识到,这些见解提供了以这样的方式执行乘法的机会,使得执行该运算时应用的中间变量具有优化的表示。

[0057] 更具体地,本发明呈现了设计用于在加密应用中执行多项式的乘法的设备的方法。在该方法中,首先将乘法划分成至少两个数据处理阶段,以使用优化的数据表示来实现。接下来,建立一个或更多个参数,从而确定在加密应用中执行乘法时如何表示数据处理阶段中使用的变量。虽然对执行这样的运算的算法的数学描述典型地假设变量的无限精度,但硬件实现(就像软件实现一样)需要决定变量的具体表示类型,诸如定点、浮点、块浮

点,以及所选表示的精度。这种选择会对实现的成本和准确性产生影响。

[0058] 一旦做出了针对每个变量的表示类型的选择,仍然必须决定该表示的确切参数。例如,针对定点表示,需要决定所表示的最低有效位(LSB)和最高有效位(MSB)。浮点表示由尾数和指数两者的最小可能值和最大可能值(因此,位大小(字长))参数化。例如,MSB值以是否可能发生溢出为特征。

[0059] 通过示例,MSB和LSB的参数化如下所示。假设值a具有8个整数位和16个小数位,以及值b具有4个整数位和8个小数位。全精度结果c=a×b则有8+4=12个整数位和16+8=24个小数位。在这种情况下,可以说c的MSB位于位置12处并且LSB位于位置-24处。在实现方式中,针对变量c执行参数化:MSBc和LSBc。通过截断MSB处的位,即执行重新缩放,(例如,选择MSBc=11),允许一定的溢出概率$P_{overflow}$。该值取决于值a和值b的分布,并且实际上所有MSB都设置成使得,例如,$P_{overflow}<2^{-64}$。通过截断LSB侧处的位,会引入由于量化而产生的一些噪声,这会增加输出噪声。注意的是,可以考虑使用任何形式的舍入来代替截断。

[0060] 重要地,变量的表示会对实现的成本产生影响,因为可能的值的范围越大使得计算成本就越高,但同时也会致使实现具有越高的准确性。针对典型的实现,将对输出变量的准确性有设计约束。这些约束是预先已知的,并且例如,可以采用所引入的最大噪声方差或具有变量溢出的最大概率(例如2^{-20}或2^{-40})的形式,但不限于此。目标是确定允许有效实现的参数,同时满足针对输出信号的所述设计约束。

[0061] 现在提供所提出方法的一般概述,该方法应用于要在设备中实现的算法,以在加密应用中对输入操作数(operant)执行多项式的乘法。多项式乘法运算被分解成多个子任务。根据本发明的设备包括至少两个相互连接的数据处理阶段。每个数据处理阶段执行至少一个子任务。至少一个数据处理阶段被配置成接收输入操作数以执行乘法。至少一个数据处理阶段被配置成提供对表示多项式的乘法的结果的输出信号。任何不生成乘法运算的输出的子任务都会产生一个变量,在本说明书中该变量被称为中间变量。任何中间变量接下来被用作一个或更多个后续子任务的输入。输入操作数可以包含要乘以的变量中的一个变量或两个(所有)变量。例如,当执行两个变量的乘法时,这两个变量可以形成输入操作数。在有利的实施方式中,输入操作数是多项式的向量。对运算的所得的输出的某些属性施加一个或更多个约束(例如,所引入的噪声的方差的最大值)。目标是确定对输出变量和实现方式中的中间变量的表示的有效参数。

[0062] 在高水平上讲,本发明的设计方法包括以下主要步骤。对于设备的每个数据处理阶段,限定与表示在该阶段中要处理的数据(即,中间变量)的方式相关的一个或更多个参数。还限定了多项式乘法运算的输出必须满足的一个或更多个约束。在下一步骤中,针对每个数据处理阶段,单独地确定针对所述表示中的每个参数的特定值,使得满足所有约束。优选地,以最佳的方式确定针对这些参数的值,即,根据适当的优化函数来确定针对这些参数的值。然后在设备的每个数据处理阶段中使用这些参数值来实现设备中的多项式的乘法。

[0063] 用于确定针对每个数据处理阶段的参数值的优选方法如下。对所考虑的由数据处理阶段的子任务中的一个子任务输出的变量的一个或更多个属性进行确定。变量的这些属性可以是基于对由该一个子任务输出的信号的属性的一个或更多个约束造成影响的噪声源的模型和/或基于所述一个子任务的输入的至少一个属性的。该模型是根据要确定的表示的参数构建的。针对所有数据处理阶段的所有中间变量以及针对输出变量重复此操作。

对于每个数据处理阶段,确定针对表示中的每个参数的特定值,使得所述一个或更多个属性满足所有约束。然后将如在相应数据处理阶段中所确定的参数值用在执行多项式的乘法的设备中。

[0064] 构建噪声源的模型可以按如下方式执行。执行乘法的算法被分解成各种子任务,并且对中间变量进行识别。针对每个子任务,确定属性相对于约束的输入-输出行为(例如,所引入的噪声、输入至输出的缩放因子,……)。针对这些变量中的每一个变量,选择表示的类型(即,定点、浮点,……)。此表示的参数(例如,如最高可表示值、最低可表示值,……)仍未确定,并且最初保留为符号变量,即,尚无特定值的变量。

[0065] 为了对算法的执行进行分析,人们利用了额外的构建块,该构建块被引入到乘法的方案中每个中间变量的位置处。该构建块表示对变量的表示的有限精度的影响,但对算法本身没有影响。为了实现这一点,额外的构建块具有将符号参数与所施加的约束联系起来的输入-输出行为。

[0066] 逐步从输入到输出进行算法,并且针对每个约束构建模型。为此,确定子任务的输入处或输入操作数的输入处的一个或更多个相关属性(例如,噪声方差、输入方差、最大可能值,……),并且然后将这些属性传播至输出。针对每个子任务,使用输入-输出行为将输入属性转换成对应的输出属性。结果,针对每个约束,都将获得该变量的属性的模型,在该模型上根据符号参数施加约束。

[0067] 在该方法的下一步骤中,确定针对符号参数中的每个符号参数的值,使得满足约束。优选地,选择这些值使得实现成本尽可能低。实现这一点的一种方式是为每个符号参数选择一个成本函数,该函数为该参数的给定值建模实现成本。然后可以通过解决优化问题来查找解决方案,其中确定一值,该值在遵守所有约束的同时降低了总成本函数并且优选地使总成本函数最小化。

[0068] 最后,将针对如所确定的参数的所选择的值应用于对加密应用的设备的各个数据处理阶段的设计进行实例化。

[0069] 在其他实施方式中,可以借助于模拟来执行对数据表示的调整。在这些实施方式中,假设每个计算阶段对输出噪声的贡献都是独立的。首先,每个计算阶段都以非常高的精度实现,从而导致输出噪声可忽略不计。接下来,对单个阶段进行调整(例如,降低定点表示的位宽),并且对所得的输出噪声进行测量并将其与高精度情况进行比较。在某个阈值位宽下,由于调整后的阶段而产生的输出噪声接近可容忍的输出噪声(例如,在X位内)。然后,将该阈值位宽视为针对该特定阶段的“最佳”位宽。重复此操作,直到为每个特定阶段确定阈值位宽。最后,每个阶段都以其阈值位宽实现,并对所有阶段的总输出噪声进行集体地测量。如果该输出噪声超过可容忍的输出噪声,则针对更大的阈值位宽重复整个过程(即,X位的容差增加了)。

[0070] 在另一实施方式中,对数据表示的调整以类似的方式进行,不同之处在于为调整后的阶段中的每个阶段分配一个成本函数。首先将具有较高计算成本的阶段调整为其最小位宽。随后,对后续阶段进行调整,同时将成本高昂的阶段保持在调整后的位宽。这样,成本高昂的阶段以绝对最小精度实现,而其他成本的阶段则以更高精度实现。

[0071] 现在呈现一个示例,其中展示了所提出的方法,用于设计硬件架构以加速如上面所提及的公式(1)中给出的向量矩阵乘法。该架构利用FFT块来加速多项式乘法。

[0072] 图1中呈现了一种有利加速器结构的框图。该架构与流式DSP处理器架构有明显的相似之处,都具有大型全流水线、直接级联计算阶段、和高度简化的控制逻辑。该加速器被设想成实现最大吞吐量/面积,具有最大程度展开的算法单元,从而消除了控制逻辑和硬编码路由路径。

[0073] 对多项式乘法的加速可以通过卷积定理实现:

[0074] $c=a\times b=IFFT(FFT(a)\bullet FFT(b))$ (2)

[0075] 在此重复公式(1):

[0076] $$\underset{1\times(k+1)}{\left(c_0(X)\ c_1(X)\ \ldots\ c_k(X)\right)}=\underset{1\times(k+1)l}{\left(a_0(X)\ a_1(X)\ \ldots\ a_{(k+1)l-1}(X)\right)}\times\underset{(k+1)l\times(k+1)}{\begin{pmatrix}\mathcal{B}_{0,0}(X) & \ldots & \mathcal{B}_{0,k}(X)\\ & \ddots & \\ \mathcal{B}_{kl,0}(X) & \ldots & \mathcal{B}_{kl,k}(X)\end{pmatrix}}$$

[0077] 其中$c=[c_0\ c_1\ldots c_k]$和$a=[a_0\ a_1\ldots a_{(k+1)l+1}]$分别表示维度为$1\times(k+1)$和$1\times(k+1)$I的多项式的向量,并且B表示维度为$(k+1)l\times(k+1)$的多项式的矩阵。基于FFT的乘法根据公式(2)通过使用FFT将输入多项式$[a_0\ a_1\ldots a_{(k+1)l-1}]$转换成另一表示而进行。在该域中,可以逐点执行与多项式$B_{i,j}$(在FFT域中预先计算)的乘法运算(公式(2)中成对进行N次运算)。矢量矩阵乘积的累加步骤优选地在FFT域中执行。之后,需要使用逆FFT(IFFT)将结果转换回初始表示。FFT和IFFT转换运算典型地是基于FFT的乘法中最昂贵的运算,需要O(N.log(N))次运算,其中N是多项式中的系数的数目。系数的数目确定了FFT的深度和宽度。

[0078] 基于FFT的乘法对复数进行运算,其中实部和虚部两者都是实数,而其他乘法算法使用整数。当使用有限精度来表示实数时,乘法的计算并不总是精确的并且可能存在噪声,即,可能引入小的误差δ:

[0079] $FFT^{-1}(FFT(a).FFT(b))=c+\delta$

[0080] 如已经所述的,除了为保证安全而在公式中已经存在的(数学)噪声之外,在FHE还可以容忍由FFT引入的一定程度的噪声。这意味着需要非常仔细地考虑噪声δ的大小。FHE的实现对所引入的噪声施加了严格的限制。如果由FFT在多项式乘法中引入了过多的噪声,计算就会失败并返回不正确的结果。

[0081] 图2示出了图1中所描绘的具有流水线结构的示例性加速器硬件架构的展开版本。这种架构有利地应用在现场可编程门阵列(FPGA)中。公式1中的矩阵输入$B_{i,j}$可以在FFT域中预先计算。从图1和图2还可以看出,输入$B_{i,j}$在FFT域中是可用的。FFT组件(12)对应用于结构的输入的向量a的FFT进行计算。乘积-累加(MAC)组件(13)在FFT域中累加逐点运算的分量。提供逆FFT组件(14)以将输出向量从FFT域放回。在公式1中,有(k+1).I个正向FFT和(k+1)个逆FFT,因此这些块需要不同的吞吐量。所有块都连接为流水线,如已经所提及的。

[0082] 从图1和图2可以看出,该架构可以划分成多个阶段(功能块)。有FFT块、MAC块、和逆FFT块。在这些块中的每一个块内也可能有多个阶段。MAC具有乘法阶段和加法阶段。乘法本身是复数乘法器(out=(a+b.j)(c+d.j)),而这一运算同样有多个阶段。每个阶段都有特定的数据表示参数,诸如(定点、浮点、位数、舍入模式),这些参数会对输出c(c=c0,c1)中存在的噪声造成影响。FFT块有$\log_2(N)$个“蝴蝶”单元的阶段。这些蝴蝶单元执行蝶形运算,这是FFT变换的重要部分。图3描绘了蝶形运算的示意图。具有其两个输入和两个输出的蝶形运算在FFT算法的实现中是众所周知的,并且将复合大小N=rm的离散傅里叶变换递归地

分解成大小为m的r个较小变换,其中r是变换的基数。然后经由大小为r的蝴蝶将这些较小的DFT组合起来,蝴蝶本身是大小为r的DFT(在子变换的对应输出上执行m次),预乘以称为旋转因子的单位根。

[0083] 由于FHE应用具有存在于多项式c中的固有加密数学噪声,并且考虑到可以容忍额外的基于FFT的舍入噪声,因此针对图2的架构中的每个阶段都可以采用不同的数据表示。针对一些额外的噪声的可用空间允许使用不同的位宽、舍入模式、定点格式等。这适用于架构中的每个阶段。

[0084] 存在各种方式可用于确定针对每个阶段的数据表示。这些方法中的每一种方法都开始于输出信号的属性(向量c(X)=(c0(X),c1(X)),例如,从所考虑的FHE方案中存在的噪声的数学公式得出的最大可容忍噪声水平。除了最大噪声水平之外,还可以施加其他约束,例如,与溢出相关的约束。在实践中,最大可容忍噪声水平始终是约束中的一个约束。

[0085] 快速傅里叶变换中的变量是复数,并且因此蝶形运算中的变量也是复数。然而,典型地可以假设实部和虚部的分布和属性相同。在这种情况下,人们在分析中可以只关注实部的属性。

[0086] 在此处所考虑的示例中,仅对每个蝶形结构施加最大噪声方差约束。例如,目标是为每个中间变量的最低有效位(LSB)的位置选择一值。噪声和信号的LSB和方差被视为相关属性。将算法划分成子任务之后,将针对中间变量v_{at}、v_c、v_d的附加的构建块添加至方案,如图4中所描绘的。这些块对由于表示的有限范围而导致的不准确性进行建模。例如,通过假设被截断的LSB位是独立均匀分布的,建立了由于截断最低有效位而引入的噪声的模型。

[0087] 蝶形运算中的乘法子任务可以通过利用旋转输入属性的知识来简化。任何具有实部t_{real}和虚部t_{imag}的旋转因子t的一个感兴趣属性是$t_{real}^2+t_{imag}^2=1$。给定一个实部和虚部的方差相同的数x,将x与旋转因子t相乘不会改变方差,即var(x.t)=var(x)。这可以很容易地推导出如下:

[0088]
$$\begin{aligned}&\operatorname{var}(\operatorname{real}((x_{real}+i\cdot x_{imag})\cdot(t_{real}+i\cdot t_{imag})))\\&=\operatorname{var}(x_{real}\cdot t_{real}-x_{imag}\cdot t_{imag})\\&=t_{real}^2\cdot\operatorname{var}(x_{real})+t_{imag}^2\cdot\operatorname{var}(x_{imag})\\&=\operatorname{var}(x_{real}),\end{aligned}$$

[0089] 对于乘法,输入-输出行为因此可以被描述为$LSB_{xt}=LSB_x+LSB_t$和$\sigma^2_{noise,out}=\sigma^2_{noise,x}+\sigma^2_x\sigma^2_{noise,t}$(假设输入具有零均值的情况下)。变量in1和in2相加的加法块的输入-输出行为由以下公式给出:

[0090] $LSB_{out}=\min(LSB_{in1},LSB_{in2})$和$\sigma^2_{noise,out}=\sigma^2_{noise,in1}+\sigma^2_{noise,in2}$

[0091] 然后可以从旋转输入开始对模型进行计算,由于有限的表示,其准确性降低(旋转1和-1以及i和-i除外)。符号参数以粗体显示,以将其与其他参数或属性区分开。

[0092] $LSB=LSB_t$

[0093] $\sigma^2=1$

[0094] $\sigma^2_{noise}=2^{2LSBt}/12$

[0095] 在针对中间变量v_{at}的附加的构建块之后,从而获得乘积的实部:

[0096] $LSB=LLSB_{at}$

[0097] $\sigma^2=\sigma^2_a$

[0098] $\sigma^2_{noise}=\sigma^2_{nois,a}+2^{2LSBt}/12*\sigma^2_a+ramp(2^{2LSBt}-2^{2(LSBa)}/12$

[0099] 对于虚部,表达式结果类似。ramp()函数针对负的输入值返回0,而针对正的输入值返回输入值。之所以有此函数,是因为只有当相关位被丢弃时才会出现附加的噪声,如果$LSB_v<LSB_{in}$就会发生这种情况。

[0100] 在针对中间变量v_c的附加的构建块之后,可以写出:

[0101] $LSB_{out}=LSB_c$

[0102] $\sigma^2_c=\sigma^2_a+\sigma^2_b$

[0103] $\sigma^2_{noise,out}=\sigma^2_{noise,c}=\sigma^2_{noise,a}+2^{2LSBt}/12*\sigma^2_a+\sigma^2_{noise,b}$

[0104] $+ramp(2^{2LSBt}-2^{2(LSBa)}/12+ramp(2^{2LSBc}-2^{2min(LSBat,LSBb)})/12$

[0105] 接下来确定符号参数的值。现在应将每个参数固定为一值,使得满足约束$\sigma^2_{noise,out}\leq\sigma^2_{maxnoise}$。实现此目的的一种方式是构建成本函数,例如,其中所有参数都根据其位宽被同等地计算成本的函数。然后可以使用在给定约束下对成本函数进行优化的优化器来找到有效的参数值。在一些实施方式中,当然可以将成本函数更改为更接近表示实现成本的函数。

[0106] 注意的是,对于简单的计算,例如在MAC阶段执行的加法,也可以应用使用如上面概述的模型的方法。同样,首先对输入-输出行为进行描述。一旦完成这一步,可以从开始到结束对模型进行检查,以根据符号参数确定与约束相关的属性。然后可以写下各个节点处的中间计算。接下来确定符号参数的值。导出约束函数$\sigma^2_{noise,out}\leq\sigma^2_{maxnoise}$。每个参数都固定为一值,使得满足此约束。

[0107] 如上面示例中所讨论的使用FFT的多项式乘法也可以被看作输入(多项式的向量)与引导密钥(也是多项式的向量)之间的内积的计算。为了对这种乘法进行有效地处理,可以使用三个步骤的程序:FFT、系数乘积和累加、以及逆FFT。图5中描绘了这种程序。注意的是,与典型的基于FFT的乘法相反,图中的第二乘法项(即,引导密钥)没有经过显式FFT运算。这是因为输入是预先已知的(如上面的公式(1)中关于矩阵b已经指出的),并且因此可以以非常高的准确性预先计算FFT,这暗示着不需要考虑这个特定的FFT。

[0108] 首先考虑乘积-累加运算。该运算以系数方式执行,并且因此可以使用与上面示例中讨论的类似的乘法运算和加法运算来建模。如已经所提及的,FFT运算和逆FFT运算主要包括多层蝶形运算。因此,将对上面示例中的分析应用在这些蝶形运算上产生FFT运算和IFFT运算的模型。在多项式乘法的一些实施方式中,实现可以使用不同类型的蝶形运算(基数2、基数4、……),但对这些蝶形运算的分析可以以与上面示例中的分析类似的方式进行。通过组合先前所讨论的构建块,可以构建基于全FFT的多项式乘法的噪声模型。

[0109] 一个挑战是需要确定大量的运算,并且因此需要确定大量的参数。为了减少模型中的参数的数目,可以将相似的参数组合在一起。在这个示例中,存在高度的并行性和结构,可以用来实现这一点。例如,可以将FFT的同一“层”中的变量(即,经历了相同数目的蝶形运算的变量)的相似的参数或乘积-累加中的乘法运算之后的变量组合在一起。这将使参数的数目从大约O((V+1)N/2log2(N/2))(其中V是向量长度,N是多项式中的系数的数目)减少至约O(2log2(N/2)),因为这大约是所提出的算法中的层的数目。

[0110] 此外,将注意的是,像FHE这样的典型的加密应用需要执行负循环卷积,而不是传

统的循环卷积。在循环卷积(具有N个系数)中,超出界限的系数(在位置i>N处)被循环回(cycled around)至第一系数(在位置i-N处)。相反,在负循环卷积中,这些系数不仅被循环回,而且还被取反。为了实现这一点,许多加密算法的实现在算法的开始和结束时执行所谓的扭曲(twist)和折叠步骤,这解释了负循环行为。这些步骤可以在图1和图2中看到:它们分别由FFT之前和IFFT之后的乘法表示。这个扭曲和折叠步骤包括对输入的额外打包和与复数的乘法(旋转因子)。打包采用两个整数a、b并将它们组合成复数a+bi。此操作典型地不会产生任何噪声。可以使用前面所描述的方法对附加的乘法运算进行建模。IFFT之后的后处理是逆运算。负循环卷积可以在单独的数据处理阶段执行。

[0111] 如已经提及的,必须选择针对变量的表示的类型。在如本发明考虑的执行多项式乘法的FFT方案中,有利地选择定点表示。然后可以应用上面所描述的方法来确定当设备执行运算时出现的变量的定点表示的参数,同时满足所施加的约束。

[0112] 给定如所描述的获得的最佳参数,可以使用针对这些参数的定点算术构建设备,例如,硬件电路。实际上,具有一个参数化硬件电路实现的库可能是有利的,其中定点位宽是通用参数。可以选择与输入类型相匹配的电路,并且在“电路合成时间”处设置这些参数以与所需的输出噪声δ相匹配。

[0113] 如上面已经提及的,用于确定参数值的替代方式是借助于执行模拟而进行的。具体而言,这可能如下所示。与上面的示例一样,首先将计算以及确定数据表示的参数划分成一组处理阶段。例如,可以选择数据处理阶段作为更高水平的运算,例如,正向FFT、MAC、和逆FFT。参数根据阶段的选择而定,例如,a=正向FFT位宽,b=MAC位宽,c=逆FFT位宽。首先,使用针对a、b、和c的非常大的位宽来对架构进行模拟,从而导致输出噪声方差$\sigma^2_{noise,out}$ $<<\sigma^2_{maxnoise}$可忽略不计。接下来,将成本函数分配至参数'a'、'b'、和'c'。此处简单的成本函数是适用的:由于正向FFT多于逆向FFT(图1),因此分配至'a'的成本高于'b'。此外,由于逆FFT运算比MAC运算成本更高,因此分配至'b'的成本高于'c'。接下来,通过首先扫描成本最高的参数'a'来对输出噪声进行模拟。减小'a',直到$\sigma^2_{noise,out}\approx\sigma^2_{maxnoise}$。这被认为是'a'的最小值。考虑到小的裕度,例如,选择'a'比最小值大1位,并以类似的方式对下一参数'b'进行处理。重复此过程,直到找到针对所有参数的值。

[0114] 给定最佳参数,可以使用获得的参数集来对设备(例如,硬件电路)进行模拟。对输出噪声δ进行测量,并且可以将输出噪声与浮点参考实现进行比较。验证输出噪声是否满足上面确定的噪声界限(例如,标准偏差2)。可以为具有上面提出的方法中确定的最佳定点参数集的电路创建FPGA比特流。FPGA比特流允许加速FHE引导程序,该FHE引导程序涉及多项式向量乘法的许多(数千)次迭代。

[0115] 在第一优选示例中,该架构如图1中所描绘的方式来实现。该架构类似于流式DSP处理器,具有直接级联的全流水线单元。数据从一个处理阶段流向下一处理阶段。本发明的目的是用不同的优化数据表示实现每个数据处理阶段。有利地,在流式架构中,数据表示之间的转换可以在流式处理阶段之间的专用连接中硬连线。此外,该架构具有简化的控制逻辑。它被设想为具有最大吞吐量/面积,具有最大程度展开的算法单元,从而消除控制逻辑和硬编码路由路径。

[0116] 图6中描绘了可以考虑的架构的另一示例(基于Matcha)。在该实施方式中,该架构类似于更经典的CPU,具有中央存储器文件和连接的算术处理单元。在图6的结构中,计算组

件(FFT、MAC、逆FFT)可以借助于交叉开关连接至存储器文件。与前面的示例一样,将为该结构的每个阶段(即FFT块、各种MAC块、和逆FFT)确定合适的数据表示。此外,中央存储器可以被视为单独的写回阶段,并且其实现方式和数据表示可以类似地进行优化。

[0117] 虽然本发明已在附图和前述说明中进行了详细地图示和描述,但这种图示和说明应被视为说明性或示例性的,而非限制性的。前述说明详述了本发明的某些实施方式。然而,将理解的是,无论前述文本中显示的多么详细,本发明都可以通过多种方式得以实践。本发明不限于所公开的实施方式。

[0118] 本领域技术人员通过根据对附图、公开内容、和所附权利要求的研究,在实践所要求保护的发明时,可以理解并实现对所公开的实施方式的其他变型。在权利要求书中,“包括”一词并不排除其他元件或步骤,且不定冠词“一”或“一个”不排除多个。单个处理器或其他单元可以实现权利要求中记载的多个项的功能。在相互不同的从属权利要求中记载特定措施的事实本身并不表明这些措施的组合不可被有利地利用。计算机程序可以存储/分布在合适的介质上,诸如与其他硬件一起提供的或作为其他硬件的部分提供的光学存储介质或固态介质,但也可以以其他形式分布,诸如经由互联网或其他有线或无线电信系统。权利要求书中的任何附图标记不应被解释为限制该范围。

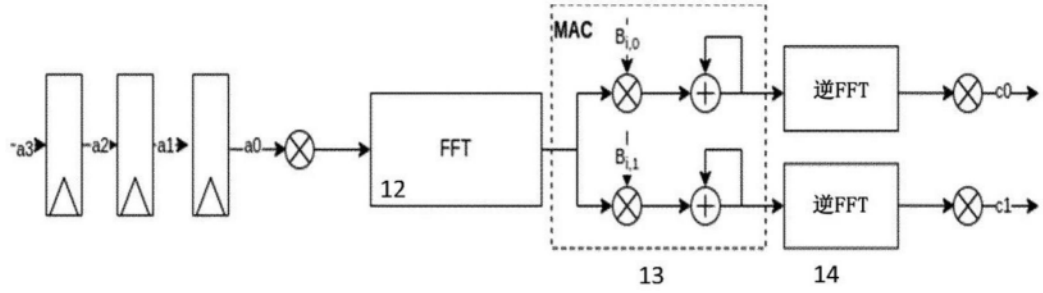

MAC
$B'_{I,0}$
$B'_{I,1}$
a3
a2
a1
a0
FFT
12
逆FFT
逆FFT
c0
c1
13
14

图1

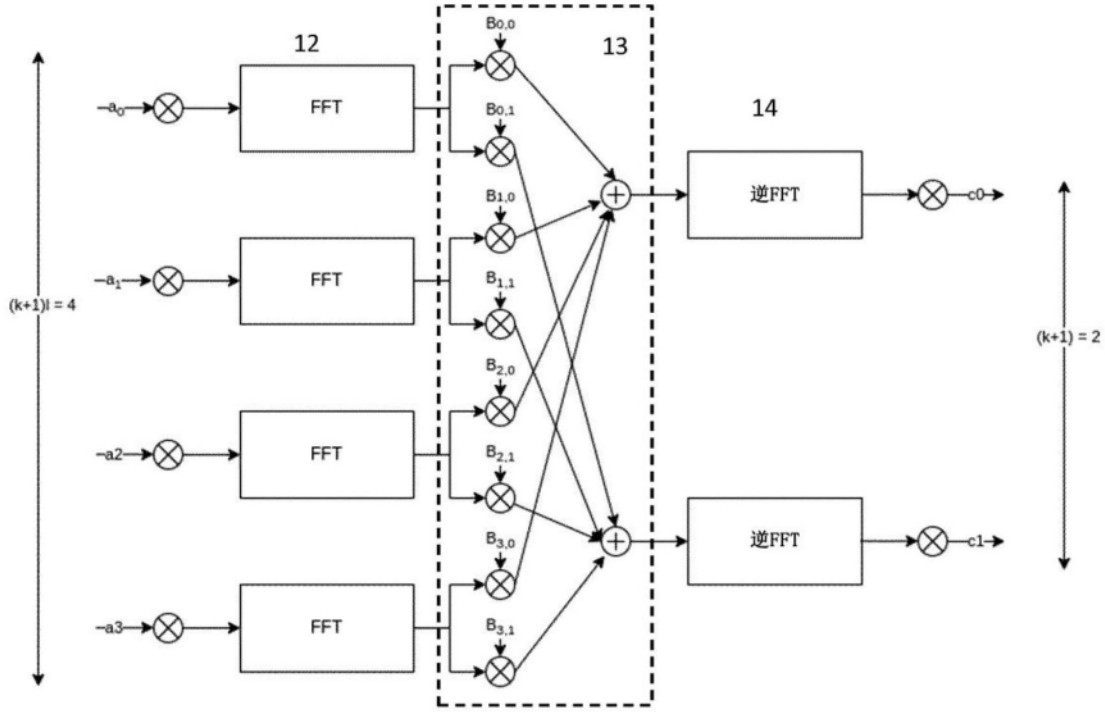

12
13
14
$B_{0,0}$
$B_{0,1}$
$B_{1,0}$
$B_{1,1}$
$B_{2,0}$
$B_{2,1}$
$B_{3,0}$
$B_{3,1}$
a_0
a_1
a2
a3
FFT
FFT
FFT
FFT
逆FFT
逆FFT
c0
c1
(k+1)I = 4
(k+1) = 2

图2

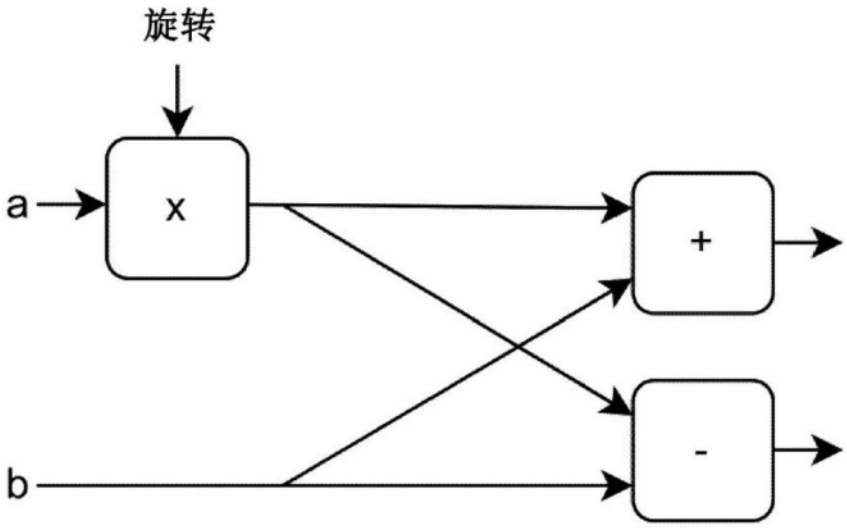


图3

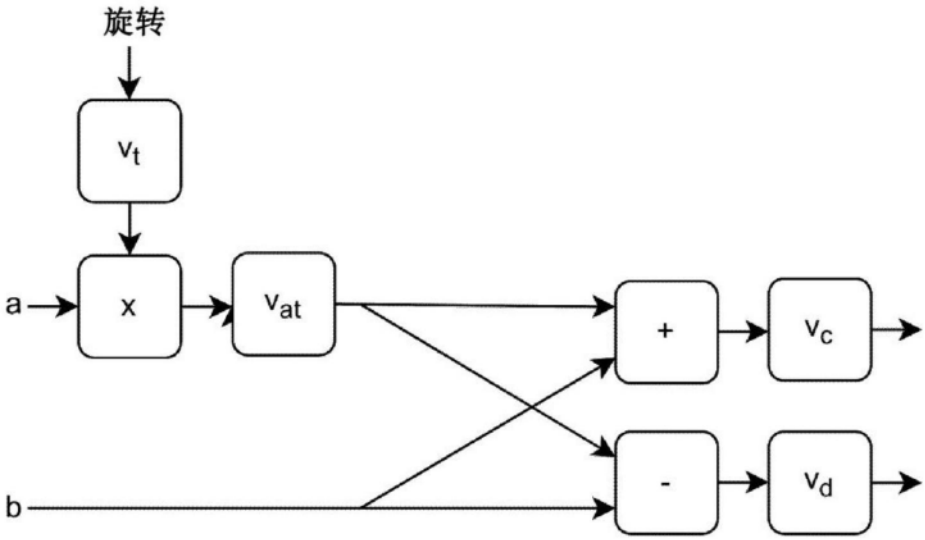


图4

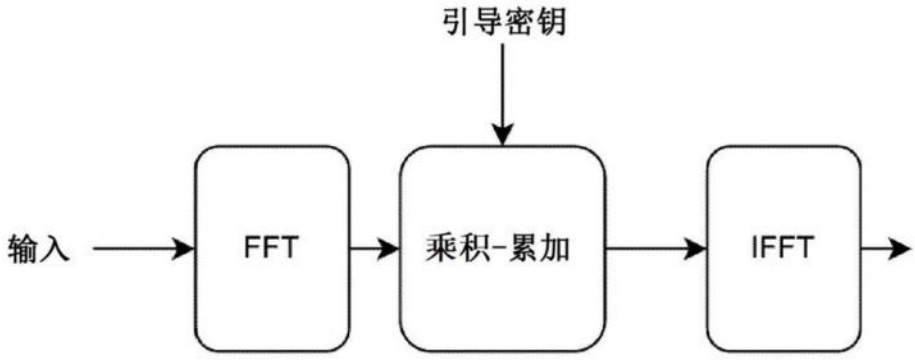


图5

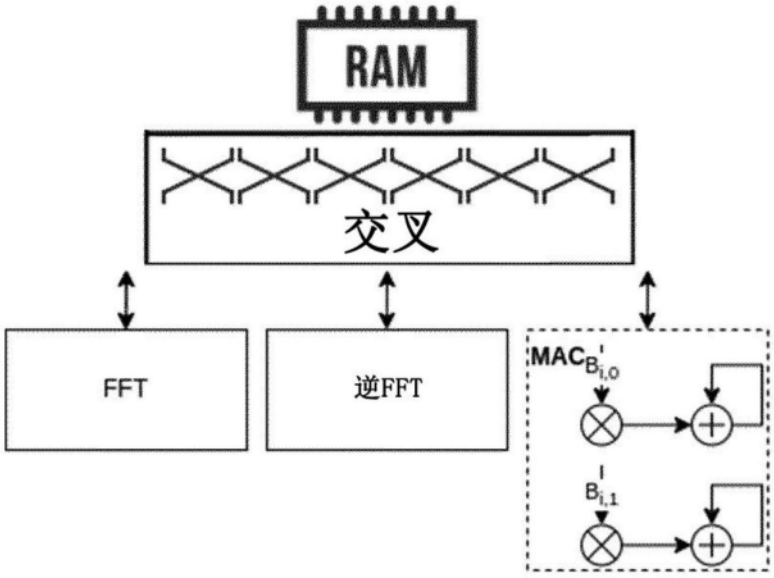


图6