# 面向全同态加密的有限域 FFT 算法 FPGA 设计

施 佺[①] 韩赛飞[①] 黄新明*[①②] 孙 玲[①②] 谢 星[①] 唐天泽[①]
[①](南通大学电子信息学院 南通 226019)
[②](江苏省专用集成电路设计重点实验室 南通 226019)

**摘 要**：大数乘法是全同态加密算法中一个不可或缺的单元模块，也是其中耗时最多的模块，设计一个性能优良的大数乘法器有助于推进全同态加密的实用化进程。针对 SSA 大数乘法器的实现需求，该文采用可综合 Verilog HDL 语言完成了一个 16×24 bit 有限域 FFT 算法的 FPGA 设计，通过构建树型大数求和单元和并行化处理方法有效提高了 FFT 算法的速度。与 VIM 编译环境下的系统级仿真结果比较，验证了有限域 FFT 算法 FPGA 设计的正确性。

**关键词**： 全同态加密；大数乘法；有限域快速傅里叶变换；现场可编程门阵列
**中图分类号**：TN918.91; TN79 **文献标识码**：A **文章编号**：1009-5896(2018)01-0057-06
**DOI**: 10.11999/JEIT170312

# Design of Finite Field FFT for Fully Homomorphic Encryption Based on FPGA

SHI Quan[①] HAN Saifei[①] HUANG Xinming[①②] SUN Ling[①②] XIE Xing[①] TANG Tianze[①]
[①](*School of Electronic Information*, *Nantong University*, *Nantong* 226019, *China*)
[②](*Jiangsu Key laboratory of ASIC Design*, *Nantong* 226019, *China*)

**Abstract:** Large multiplier is an indispensable module in fully homomorphic encryption, while is also the most time-consuming module. Therefore, design of a large multiplier with good performance is help to promote the practical process of fully homomorphic encryption. Aimed at the demand of SSA (Schönhage-Strassen Algorithm) large multiplier, a 16×24 bit finite field FFT based on FPGA is designed by using Verilog HDL language. By constructing the tree type large sum unit and using parallel processing method, the speed of FFT algorithm is improved effectively. And its correctness is proved by comparing with the system level simulation results in VIM compiler environment.

**Key words:** Fully homomorphic encryption; Large multiplier; Finite field FFT; Field Programmable Gate Array (FPGA)

## 1 引言

近年来，随着信息量的急剧增长，用户对于计算大量数据的需求也越来越大，导致个人计算机等终端设备已经无法满足大量数据的实时计算要求[1]。云服务作为一种新型的互联网技术应运而生，成为信息存储管理和运算处理的有效手段。但随着云服务的逐渐普及应用，其暴露出来的安全问题也越来越严重，许多用户的私密数据遭到了泄露。因此，云服务的安全问题成为制约其快速发展的一个瓶颈。

作为一种新型的加密算法体制，全同态加密算法被广泛认为可以有效地解决云服务数据安全性问题。全同态加密又称为隐私同态，允许在没有私钥的条件下，可以对加密后的数据进行各种运算操作，即对明文对应的密文数据进行操作等价于对明文进行相同的操作。这种良好的同态性可以让用户将加密后的数据在任意服务商的终端进行计算[2]。与许多传统的加密算法相比，全同态加密理论上可以完全解决云计算、物联网中存在的各种信息安全问题。但实际实现和应用过程中，现有的全同态加密算法的运算复杂度过高，其中的密文与秘钥的长度过大，导致全同态加密算法的运算效率一直处在比较低的水平，在云服务的环境中暂时还不具备良好的实用性。近年来，国内外研究者有的从算法角度，有的从算法物理实现角度等不同方面加紧研究，不断推进全同态算法的实用化进程[3,4]。

收稿日期：2017-04-10；改回日期：2017-07-19；网络出版：2017-08-28
*通信作者：黄新明 hxm@ntu.edu.cn
基金项目：国家自然科学基金(61571246)，南通大学杏林学院自然科学基金(13010538)
Foundation Items: The National Natural Science Foundation of China (61571246), The Natural Science Foundation of Xinglin College of Nantong University (13010538)

本文对同态加密算法中最耗时的大数乘法操作进行了分析，采用树形结构与并行化处理方案完成了大数算法中的有限域 FFT 关键模块设计与优化。本文首先简要分析了基于整数的全同态加密方案以及大数算法特点；然后详细给出了大数乘法中有限域 FFT 单元的算法原理，提出了该单元的硬件架构设计方案，完成了其 FPGA 设计与功能仿真，并对仿真结果的正确性进行了验证；最后总结全文。

## 2 全同态加密算法概述与大数乘法方案

### 2.1 全同态加密的定义

假设，$m_1,m_2,\cdots,m_k$ 为明文，$f$ 是数据处理函数，En 是加密函数，Dec 是解密函数[4]。如果 $f$ 既满足加法又满足乘法，则该算法称为全同态加密算法。用公式表达，即 $\mathrm{Dec}(f(\mathrm{En}(m_1)),f(\mathrm{En}(m_2)),\cdots,f(\mathrm{En}(m_k)))=f(m_1,m_2,\cdots,m_k)$。如图 1 给出了其示意图，图中 $m_1,m_2,\cdots,m_k$ 为输入的明文，经 En 加密函数加密之后，分别对密文进行乘法与加法操作。密文经过解密之后得到结果与直接对明文进行加法与乘法运算相一致。图 1 中，$f_1$ 为加法运算，$f_2$ 为乘法运算。

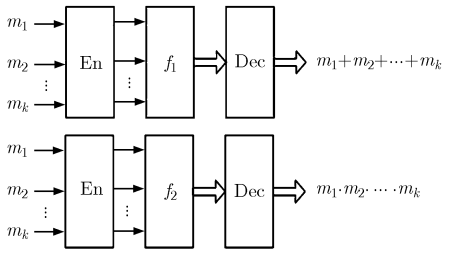



图 1 全同态加密示意图

### 2.2 基于整数的全同态加密方案

自 2009 年 Gentry[5]首次提出基于理想格的全同态加密算法后，其团队又提出了基于整数和基于带错学习(Learning With Errors, LWE)问题等方案。其中，基于整数方案的全新全同态加密体制把原始的理想格方案改进为只利用加法、乘法和模运算来代替基于多项式环的理想格，这样可以在不影响安全性的同时又降低了算法的复杂度，大大提高了加密系统的计算效率。因此，本设计就是针对 Gentry-Halevi 提出的基于整数的全同态加密方案而开展的工作[5−7]。该加密方案包括秘钥生成、加密、解密和重加密这 4 个过程，其中，加、解密这两个过程最为耗时。其加密过程就是用公共秘钥 $(d,r)$ 对明文进行多项式求值运算从而得出密文，加密公式如式(1)所示，解密公式如式(2)所示。

$$c=[u(r)]_d=\left[b+2\sum_{i=1}^{n-1}u_i r^i\right]_d \tag{1}$$

$$m=[c\times w]_d \bmod 2 \tag{2}$$

式中，$b$ 表示明文数据；$d$，$r$ 为公钥，其长度一般有 785000 bit；$u_i$ 为加密过程中随机生成的“噪声矢量”，$u=<u_0,u_1,\cdots,u_{n-1}>$，其中若噪声为 0 的概率是 $p$，则为 1 或−1 的概率就是 $(1-p)/2$；$w$ 为私钥，其长度与公钥的长度一致，$c$ 为密文。

### 2.3 基于 FPGA 的全同态加密的加速方案

从式(2)可以看出，解密算法中 $c\times w$ 是两个 785 kbit 数据的乘法运算，用基本的乘法运算实现必然要经过很长的运算时间。因此，大整数的乘法成了整个全同态加密系统中最耗时间的操作，也是密码学中的一个计算瓶颈。最近研究显示，GPU 和 FPGA 等硬件架构的特点和并行化处理技术具有大幅度提高同态加密方案效率的潜力[8]，尤其是实现许多 FHE 方案中的底层加密单元，比如模归约和大数乘法等。此外，如今的 FPGA 还包括了优化的嵌入式乘法与加法 IP 模块，使得基于 FPGA 的算术运算设计具有较高的灵活性，对本文实现大数乘法也是非常有益的。

通过查阅相关的文献资料，表 1 给出了小学算法、Karatsuba 算法、Toom-Cook 算法和 Schönhage-Strassen 算法(SSA)这 4 种主要大数乘法算法的时间-复杂度对比[9−13]，其中 $n$ 表示乘法的位数。图 2 给出了不同乘数位数下它们的时间-复杂度 Matlab 仿真曲线，由于小学算法比另 3 种算法大了两个数量级，因此将它分开在单独的图中显示，可见，在计算同等大小数据时，SSA 算法的时间复杂度最小。因此，本次设计选择 SSA 大数算法来实现全同态加密中的乘法运算。

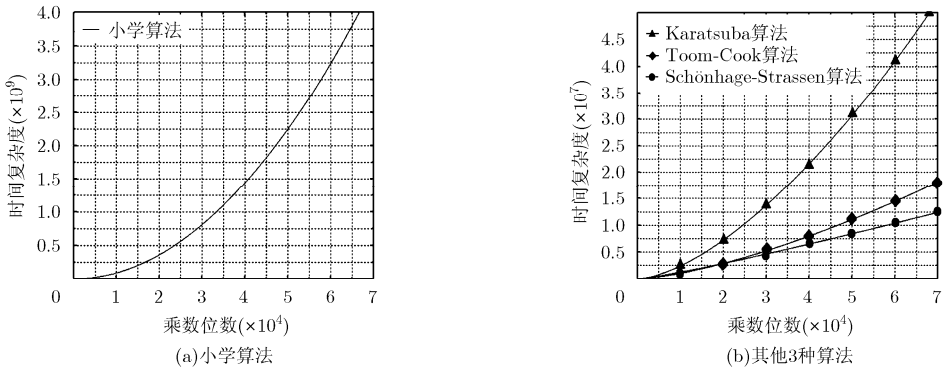
### 2.4 Schönhage-Strassen 算法(SSA)概述

根据 SSA 算法原理[14]，结合本文拟设计的 16×24 bit 有限域 FFT 算法需求，这里将 SSA 算法设计为如图 3 所示结构，其主要实现步骤如下：

步骤 1 将两个拟相乘的大数 $A$ 和 $B$ 分解成一系列的 $a(n)$ 和 $b(n)$，($a(n)$，$b(n)$ 都为 32 bit 的二进制数)；

**表 1 大数乘法主要算法时间-复杂度对比**

| 算法名称 | 时间复杂度 |
|---|---|
| 小学算法(普通算法) | $n^2$ |
| Karatsuba 算法 | $n^{1.58}$ |
| Toom-Cook 算法 | $n^{\lg(5)/\lg(3)}$ |
| Schönhage-Strassen 算法(SSA) | $n\lg n\lg(\lg n)$ |




(a)小学算法　　(b)其他3种算法

图 2 不同算法复杂度对比

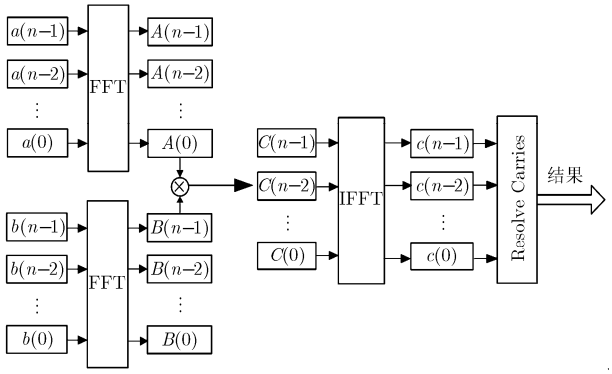



图 3 本设计中的 Schönhage-Strassen 算法结构

步骤 2　分别计算分解后 $A$ 和 $B$ 序列的 FFT，分别得到 $A(n)$ 和 $B(n)$；

步骤 3　将 $A$ 和 $B$ 序列的 FFT 计算结果按部分对应相乘得到，$C(n)$，即

$$C(i)=\mathrm{FFT}(A(i))\cdot\mathrm{FFT}(B(i))$$

步骤 4　对 $C(i)$ 作 FFT 逆变换 IFFT，得到 $c(n)$，即：$c(n)=\mathrm{IFFT}(C(i))$；

步骤 5　解决进位问题，具体为：若 $c(i)\geq b$，则 $c(i+1)=c(i+1)+(c(i)\ \mathrm{div}\ b)$，并且 $c(i)=c(i)\ \mathrm{mod}\ b$；$c(i)<b$，则 $c(i)=c(i)$。

由上述分析可见，SSA 是基于 FFT 的乘法算法，需要计算每个操作数的 FFT 以及点积之后的 IFFT，所以 FFT 模块是整个大数乘法器中的最关键部分。下面给出适用于 SSA 算法的 FFT 硬件设计具体过程。

## 3　有限域 FFT 硬件设计与实现

### 3.1 有限域 FFT 运算分析

复频域的浮点 FFT 运算被广泛用于信号处理，通常情况下，硬件实现复频域 FFT 的乘法运算时需要舍入操作，这会使得最终结果产生误差。因此，常用的 FFT 计算方法无法在严格的加密系统中使用[15−17]。本设计为 SSA 算法选择了在有限域 $Z/pZ$ 的 FFT 运算，运算中只使用模加、模减和模乘操作。此外，与浮点 FFT 比较，有限域 FFT 的硬件结构将节省大量的资源，其计算公式如式(3)所示。

$$X_i=\sum_{j=0}^{k-1}x_j(r_k)^{ij}\ \mathrm{mod}\ p \tag{3}$$

与传统的FFT算法类似，有限域FFT中$(r_k)^{ij}$代替了传统算法中的$\mathrm{e}^{\mathrm{j}2\pi i/k}$。对于 FHE 算法，本设计选择素数 $p=2^{64}-2^{32}+1$。在这个有限域上，素数 $p$ 具有以下的特殊性质，$2^{96}\ \mathrm{mod}\ p=-1$，$2^{192}\ \mathrm{mod}\ p=1$，$2^{64}\ \mathrm{mod}\ p=2^{32}-1$。

在全同态加密系统中，大数乘法一般需要计算 64 k 点数的 FFT。如果直接设计一个 64 k 点的 FFT，其结构十分复杂，很难进行设计。最主要的原因是 64 k 点 FFT 的基数 $r_k$ 为 E9653C8DEFA860A9，直接使用这么大基数的电路十分庞大而且计算效率很低。幸运的是，有限域 $Z/pZ$ 中，有 $2^{192}\mathrm{mod}\ p=(2^{12})^{16}\mathrm{mod}\ p=1$，因此，本设计选择了 16 点 FFT 为基本单元，并且 16 点的 FFT 基数为 $2^{12}$，这远远比 0xE9653C8DEFA860A9 小的多。因此 64 k 点 FFT 可以分解为 4 级的 16 点 FFT，在每一级中完成 64 k 个点的 FFT 运算需要 4096 个时钟周期。16 点有限域 FFT 可以用公式(4)表示。

$$X(k)=\sum_{n=0}^{15}x(n)2^{12nk\%192}\ \mathrm{mod}\ p \tag{4}$$

其中，%为取模运算。

### 3.2 16 点 FFT 硬件结构

分析式(4)可知，有限域 16 点 FFT 运算过程只需要模加和移位操作，图 4 给出了其运算体系结构示意图。该模块包括了 16 个移位单元和一个大数求和单元。在每个时钟周期内，16 组样本被发送到移位单元，经过在若干个流水线延时的时钟周期内进行移位和累加后输出 16 点 FFT 结果。为了防止在移位的时候发生数据溢出，我们将输入数据进行位扩展，其中 64 bit 的输入数据中，有效数据只有 24 bit。

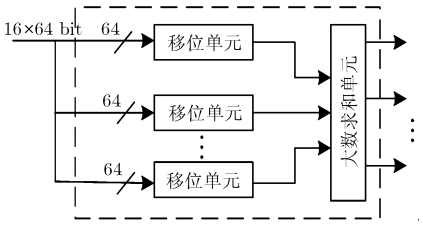



图 4 16 点有限域 FFT 运算结构

### 3.3 移位单元

利用 16 点有限域 FFT 运算具有的特性，本文采用了移位处理单元来代替公式中的乘法，并利用 FPGA 中查找表具有的移位寄存功能实现了移位操作。在进行移位操作之前，首先需要计算出所有的移位因子，即通过计算$12nk\%192$ (其中，$n$ 和 $k$ 的取值分别为 0 到 15 的整数)得出如表 2 所示的移位因子表。观察表 2 可以看出，该表中的数据可以构成一个对称矩阵。可见，通过预先计算移位因子可以节省不少的计算步骤，并且可以有效节约硬件实现时逻辑资源的使用。

### 3.4 树形大数求和单元

由于计算 16 点 FFT 需要用到模加法运算，如果使用传统的加法器将会产生大量的进位传递延时，导致电路计算速度的降低。为了减少多个操作数相加时进位传递延迟的影响，本设计采用了以进位保留加法器(Carry Save Adder, CSA)为基本单元，设计了一个 16 输入的大数加法单元，能有效的减少中间的进位链，保证高速系统的性能要求。传统的进位保留加法器只接受 3 个 $n$ 位操作数，产生 2 个 $n$ 位的结果：一个是部分和输出，还有一个是进位输出。给出 $a$ , $b$ , $c$ 3 个操作数，其部分和为 $\mathrm{ps}=a\oplus b\oplus c$，进位为 $\mathrm{sc}=ab+ac+bc$。因为输入的 $a$ , $b$ , $c$ 是并行的，所以 1 个 CSA 的总延迟时间等于单个全加器的延迟时间。因此，3 个数相加产生两个数只需 1 个全加器延迟时间。因此通过加入更多的并行单元，很容易扩大规模。使用两个进位保留加法器进行串联实现 1 个 4 输入的 CSA，其中 4 个输入的权重是相同的，4 输入 CSA 的内部结构如图 5 所示。图 5 中 $a$ , $b$ , $c$ , $d$ 为 4 个操作数，ps 为部分和输出， sc 为进位输出。

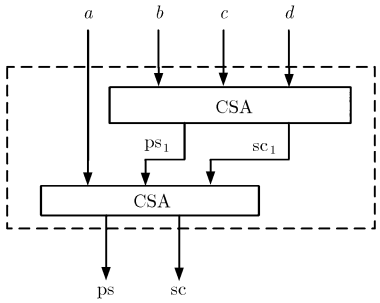



图 5 4 输入进位保留加法器

**表 2 移位因子表**

| $n$ | $k$ | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 12 | 24 | 36 | 48 | 60 | 72 | 84 | 96 | 108 | 120 | 132 | 144 | 156 | 168 | 180 |
| 2 | 0 | 24 | 48 | 72 | 96 | 120 | 144 | 168 | 0 | 24 | 48 | 72 | 96 | 120 | 144 | 168 |
| 3 | 0 | 36 | 72 | 108 | 144 | 180 | 24 | 60 | 96 | 132 | 168 | 12 | 48 | 84 | 120 | 156 |
| 4 | 0 | 48 | 96 | 144 | 0 | 48 | 96 | 144 | 0 | 48 | 96 | 144 | 0 | 48 | 96 | 144 |
| 5 | 0 | 60 | 120 | 180 | 48 | 108 | 168 | 36 | 96 | 156 | 24 | 84 | 144 | 12 | 72 | 132 |
| 6 | 0 | 72 | 144 | 24 | 96 | 168 | 48 | 120 | 0 | 72 | 144 | 24 | 96 | 168 | 48 | 120 |
| 7 | 0 | 84 | 168 | 60 | 144 | 36 | 120 | 12 | 96 | 180 | 72 | 156 | 48 | 132 | 24 | 108 |
| 8 | 0 | 96 | 0 | 96 | 0 | 96 | 0 | 96 | 0 | 96 | 0 | 96 | 0 | 96 | 0 | 96 |
| 9 | 0 | 108 | 24 | 132 | 48 | 156 | 72 | 180 | 96 | 12 | 120 | 36 | 144 | 60 | 168 | 84 |
| 10 | 0 | 120 | 48 | 168 | 96 | 24 | 144 | 72 | 0 | 120 | 48 | 168 | 96 | 24 | 144 | 72 |
| 11 | 0 | 132 | 72 | 12 | 144 | 84 | 24 | 156 | 96 | 36 | 168 | 108 | 48 | 180 | 120 | 60 |
| 12 | 0 | 144 | 96 | 48 | 0 | 144 | 96 | 48 | 0 | 144 | 96 | 48 | 0 | 144 | 96 | 48 |
| 13 | 0 | 156 | 120 | 84 | 48 | 12 | 168 | 132 | 96 | 60 | 24 | 180 | 144 | 108 | 72 | 36 |
| 14 | 0 | 168 | 144 | 120 | 96 | 72 | 48 | 24 | 0 | 168 | 144 | 120 | 96 | 72 | 48 | 24 |
| 15 | 0 | 180 | 168 | 156 | 144 | 132 | 120 | 108 | 96 | 84 | 72 | 60 | 48 | 36 | 24 | 12 |

在 4 输入 CSA 的基础上，进一步设计得到三级串联结构的 16 输入树形大数求和单元，具体结构如图 6 所示。

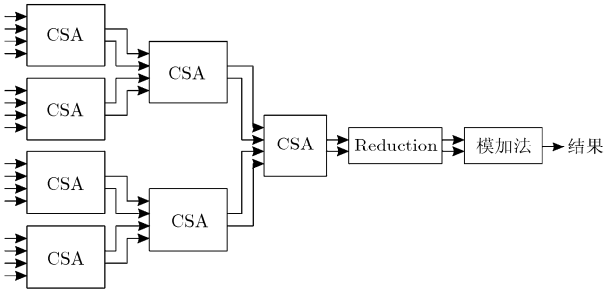



图 6　16 输入树型大数求和单元结构图

利用素数 $p$ 的特殊性质，经过图 6 中 Reduction 模块的处理可以把前一级输出结果从 192 bit 转换为 64 bit 。因为每一个 192 bit 数据 $Z$ 可以表示为

$$\begin{aligned} Z &= 2^{160}a + 2^{128}b + 2^{96}c + 2^{64}d + 2^{32}e + f \\ &= -\left(2^{32}-1\right)a - 2^{32}b - c + \left(2^{32}-1\right)d + 2^{32}e + f \\ &= \left(2^{32}e + f\right) + \left(2^{32}d + a\right) - \left(2^{32}b + c\right) - \left(2^{32}a + d\right) \end{aligned}$$

其中，$a,b,c,d,e$ 和 $f$ 都是 32 bit 数据，经过 Reduction 模块处理后的输出结果进行模加法运算，即可得出最终的求和结果。

### 3.5 基于有限域 16 点 FFT 的 FPGA 实现

本文工作之前，课题组团队已经在 Linux 环境下通过使用 NTL 库函数完成了 16 点有限域 FFT 的系统级仿真。根据有限域 16 点 FFT 硬件结构，在 Quartus Ⅱ 13.0 和 Modelsim 10.4 软件平台下，本文完成了 16 点 FFT 的可综合 Verilog HDL 语言设计，并在与系统级仿真相同的输入条件下，得到 Modelsim 功能仿真结果如图 7。在两个不同的仿真环境下输入 16 个整数序列[100,100,100,100,100,100,100,100,100,100,200,200,200,200,200,200,] ， Linux 下程序执行结果与 modelsim 的仿真结果如图 8 所示。图 7 和图 8 中分别标记出了 X[12], X[13]和 X[14]3 个点的仿真结果。对比可见，Modelsim 输出的运算结果与 Linux 下仿真结果完全一致。由此可以验证本文设计的 16 点有限域 FFT 模块功能正确。在时钟频率为 200 MHz 的情况下，完成一次 16×24 bit 的 FFT 计算需要 100 ns 时间，能够满足实时加密的要求。

## 4　结束语

随着大数据时代的到来，全同态加密算有效地

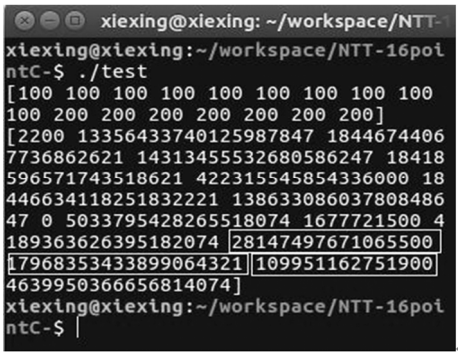



图 7　16 点有限域 FFT 的 Linux 行为级仿真结果

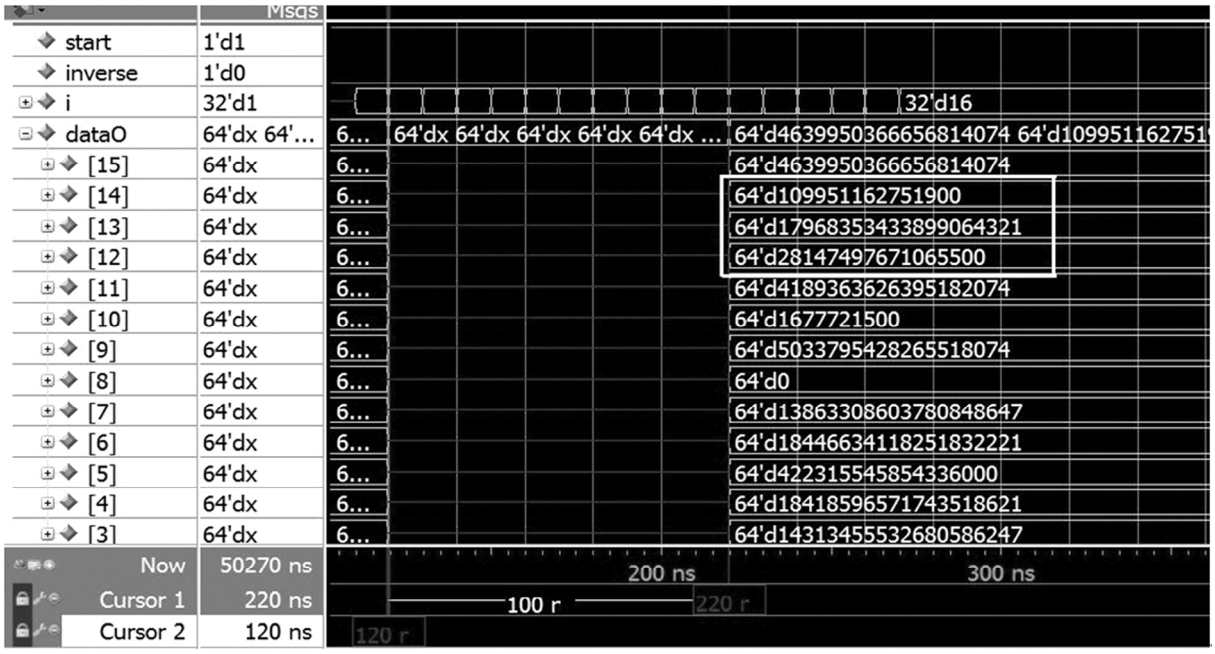



图 8　16 点有限域 FFT 的 Modelsim 寄存器传输级仿真结果

保障了云计算过程中数据安全。本设计提出了基于FPGA 有限域 FFT 乘法算法的研究，根据算法原理完成了 16 点有限域 FFT 的硬件电路设计，并且进行了 FPGA 仿真和功能仿真结果验证。作为大数乘法器中最关键的模块，16 点 FFT 模块的性能直接影响着整个大数乘法器的工作效率。本设计完成的 16 点有限域 FFT 模块具有计算精度高，运算速度快等特点，为实现全同态方案中大数乘法工作奠定了基础。

## 参 考 文 献


[1] 光焱, 祝跃飞, 顾纯祥, 等. 一种针对全同态加密体制的密钥恢复攻击[J]. 电子与信息学报, 2013, 35(12): 2999–3004. doi: 10.3724/SP.J.1146.2013.00300.
GUANG Yan, ZHU Yuefei, GU Chunxiang, *et al*. A key recovery attack on fully homomorphic encryption scheme[J]. *Jounal of Electronics & Information Technology*, 2013, 35(12): 2999–3004. doi: 10.3724/SP.J.1146.2013.00300.

[2] CAO Xiaolin and MOORE C. Optimised multiplication architectures for accelerating fully homomorphic encryption [J]. *IEEE Transactions on Computers*, 2016, 65(9): 2794–2806. doi: 10.1109/TC.2015.2498606.

[3] 刘明洁, 王安. 全同态加密研究动态及其应用概述[J]. 计算机研究与发展, 2014, 51(12): 2593–2603. doi; 10.7544/issn100-1239.2014.20131168.
LIU Mingjie and WANG An. The homomorphic encryption research dynamic overview and its application[J]. *Computer Research and Developmen*t, 2014, 51(12): 2593–2603. doi: 10.7544/issn100-1239.2014.20131168.

[4] 陈智罡, 石亚峰, 宋新霞. 全同态加密具体安全参数分析[J]. 密码学报, 2016, 3(5): 480–491.
CHEN Zhigang, SHI Yafeng, and SONG Xinxia. Estimating concert security parameters of fully homomorphic encryption [J]. *Journal of Cryptologic Research*, 2016, 3(5): 480–491.

[5] GENTRY C. Fully homomorphic encryption using ideal lattices[C]. The 41st ACM Symposium on Theory of Computing Proceedings, Bethesda, Maryland, USA, 2009: 169–178.

[6] 吕海峰, 丁勇, 代洪艳, 等, LWE 上的全同态加密方案研究[J]. 信息网络安全, 2015, (1): 32–38. doi: 10.3969/j.issn.1671-1122.2015.01.006.
LÜ Haifeng, DING Yong, DAI Hongyan, *et al*. Survey on LWE-based fully homomorphic encryption scheme[J]. *Net Inforamtion Security*, 2015, (1): 32–38. doi: 10.3969/j.issn.1671-1122.2015.01.006.

[7] GENTRY C and HALEVI S. Implementing Gentry's fully homomorphic encryption scheme[C]. Annual International Conference on the Theory and Applications of Cryptographic, Tallinn, Estonia, 2011: 129–148. doi: 10.1007/978-3-642-20465-4_9.

[8] GENTRY C. A fully homomorphic encryption scheme[D]. [Ph.D. dissertation], Stanford University, 2009.

[9] 吕金萍. 基于 LWE 的全同态加密的设计与研究[D]. [硕士论文], 杭州电子科技大学, 2014.
LÜ Jinping. Design and research of FHE based on LWE[D]. [Master dissertation], Hanzhou Electronic Science and Technology University. 2014.

[10] 吴晓园. 基于格的全同态加密方案的研究与设计[D]. [硕士论文], 西安电子科技大学, 2012.
WU Xiaoyuan. Study and design of fully homomorphic encryption scheme based on case[D]. [Master dissertation], Xidian University, 2012.

[11] WANG W, HU Y, CHEN L, *et al*. Accelerating fully homomorphic encryption using GPU[C]. IEEE Conference on High Performance Extreme Computing, Waltham, MA, USA, 2012: 1–5. doi: 10.1109/HPEC.2012.6408660.

[12] EMMART N and WEEMS C. High precision integer addition, subtraction and multiplication with a graphics processing unit[J]. *Parallel Processing. Letters*, 2010, 20(4): 293–306.

[13] WANG Wei, HUANG Xinming, and EMMART N. VLSI desgn of a large-number multiplier for FHE[J]. *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, 2014, 22(9): 1879–1887. doi: 10.1109/TVLSI.2013.2281786.

[14] SCHÖNHAGE A and STRASSEN V. Schnelle multiplikation grosser zahlen[J]. *Computing*, 1971, 7(3): 281–292. doi: 10.1007/BF02242355.

[15] 占席春, 蔡费杨, 王伟. 多路并行 FFT 算法的 FPGA 实现技术[J]. 现代电子技术, 2015, 38(19): 35–39.
ZHAN Xichun, CAI Feiyang, and WANG Wei. FPGA-based implementation technologies of multi-channel parallel FFT algorithm[J]. *Modern Electronics Tchnique*, 2015, 38(19): 35–39.

[16] SAID Boussakta. Generalized new mersenne number transforms[J]. *IEEE Transactions on Signal Processing*, 2012, 60(5): 2640–2647. doi: 10.1109/TSP.2012.2186131.

[17] EMMART N and WEEMS C. High precision integer multiplication with a GPU using Strassen's algorithm with multiple FFT sizes[J]. *Parallel Processing Letters*, 2011, 21(3): 293–306. doi: 10.1109/IPDPS.2011.336.



施 佺： 男，1973 年生，教授，研究方向为智能信息处理.

韩赛飞： 男，1992 年生，硕士生，研究方向为数字集成电路设计.

黄新明： 男，1974 年生，教授，研究方向为 VLSI 设计等.