Design of FFT Processor Supporting Novel Multiplier for Fully Homomrphic Encryption

Susi.Mu
M.E IInd year, Dept of ECE,
K. Ramakrishnan College of Engineering,
Trichy, India
susimuthu20@gmail.com

Nagarajan.N.R.
Assistant Professor, Dept of ECE,
K.Ramakrishnan college of Engineering,
Trichy
naguube@gmail.com

***Abstract*— Security is a substantial requisite in day-to-day applications. Cryptography is one of the modes to provide security for network and communication. This paper proposes a multiplier design for fully homomorphic encryption. Fully Homomorphic Encryption (FHE) is an encryption technique which supports communication to be performed on a ciphertext. This multiplier design is used to perform multiplication operation of data in the encrypted form. The operation over cipher text increases the security level. The multiplication operation is carried out by the use of Fast Fourier Transform (FFT) processor. The design is mainly concerned with reduction in delay and area consumption and focuses mainly on increase in speed. Delay is reduced by use of carry save adder in a pipelined architecture.**

Keywords: Multiplier, Fully Homomorphic Encryption, Fast Fourier Transform

I. INTRODUCTION

Very Large Scale Integration (VLSI) is the process of forming an integrated circuit which involves a combination of a large number of transistors. It thus helps in combining processors and memory into a single unit. Cryptography involves provision of security to data transmitted form a source to a destination. Data from the source is encrypted by use of an algorithm employing keys and then they are transmitted. On reception of data at the destination it is decrypted by use of the same algorithm and same or different key. Processing of data in its encrypted form increases the security level and this phenomenon is called as Fully Homomorphic Encryption (FHE). The multiplier designed in the paper is capable of operation in FHE mode. The multiplication operation is performed with the use of FFT processor. The core operation of the processor is performed using radix 16 and modular multiplication operation which enhances the system speed. Radix 16 unit comprises of two operations namely summing and shifting performed by carry save adders and shifters respectively.

A. RADIX 16 UNIT

The term radix is defined as the FFT decomposition size. In general the transform size should be power of the radix. In the design of FFT processor the design of radix-16 unit is simple consisting of sum unit and shifting unit. The sum unit consists of a number of 14 carry-save adders that operate in a pipelined fashion. Carry save adder is chosen on comparison with the other adders that form complex adder circuits. Pipelined system is incorporated in order to perform faster computation by taking multiple inputs and providing instantaneous output. The shifter is used to perform left shift of one bit. The input as well as the output of the radix 16 unit consists of 16 samples each of 64 bits. The input provided at each stage of the summer is different only because of the shifting operation that takes place prior to the summing operation.

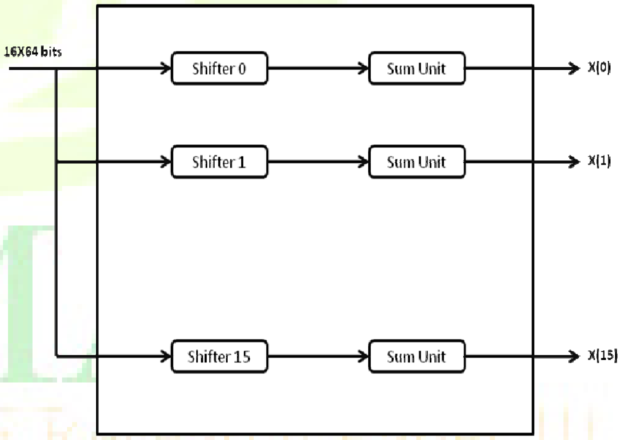


Figure: 1 Block diagram for Radix 16 unit

B. MODULAR MULTIPLICATION

The modular multiplication unit is an important sector to compute the Fast Fourier Transform in the FFT processor block. Modular multiplication is used to perform faster computation. This is done only with respect to integers. All operations involved in modular arithmetic are performed with the usage of reminders. In the modular multiplication block the two inputs comprise of the output from radix-16 block and the output of the multiplexer which is the twiddle factor. These factors are the coefficients used to combine results from previous stages to provide as input to the next stage. The registers used in modular multiplication block are 32 bit registers. Both the inputs to the modular multiplication block consist of 64 bits. The multiplier is a pipelined structure which can perform multiple modular operations simultaneously thereby reducing time consumption. The modular multiplication process consists of operations such as segmentation, addition, shifting and subtraction followed by combining.

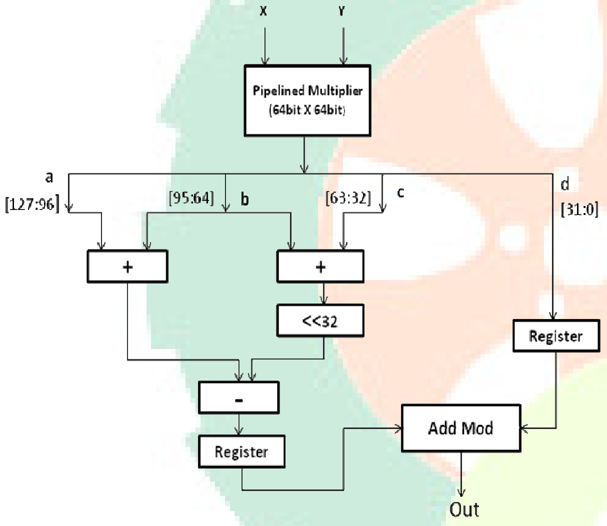


Figure: 2 Block diagram for modular multiplication unit

C. FFT PROCESSOR

The multiplier design can be achieved by first constructing the FFT processor that helps to obtain the result of multiplication. The multiplication process is based on convolution which is performed by the FFT processor. On resolving carries between the digits of the convolution result the final product of multiplication can be obtained.

The FFT multiplication involves an algorithm which has five distinct steps. In order to compute a product of X times Y, The numbers X and Y are expressed as sequence of digits followed by convolution and resolving caries.

The algorithm can be declared on basis of five stages namely

1. Decomposition-To represent A and B as a(n) and b(n).
2. FFT-Computation of FFT of a(n) and b(n).
3. Component-wise product-C[i]=FFT(A)[i]*FFT(B)[i].
4. IFFT-c(n)=IFFT(C).
5. Resolve carries

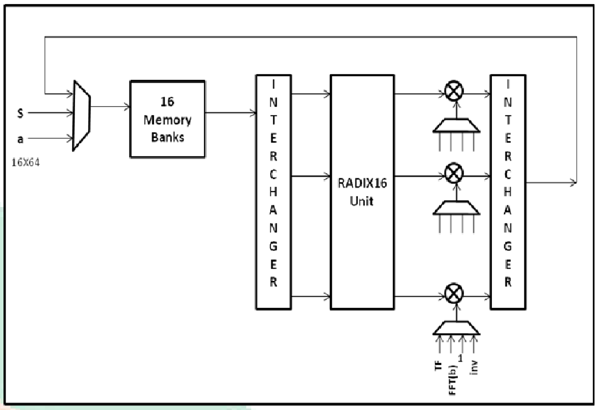


Figure: 3 Block diagram for FFT processor

D. *MULTIPLIER*

The multiplier design involves the forward and reverse operation of FFT processor. Output of a processor is provided as one of the inputs to the other processor to compute the inverse transform output. The output from the FFT processor is then processed for bandwidth resolution. This processing is handled by the resolve carries unit. The final output provides the product of the input ciphertext.

II. RELATED WORK

In[1] the concept of Software as a Service in cloud took place with the non existence of Fully Homomorphic Encryption scheme. In [2] the evolution of Fully Homomorphic Encryption took place. But this scheme did not suite actual deployment due to complexity of lattices. In [3] optimizations are made to develop a working implementation. In [4] the implementation was made on x3500 sever but remained impractical due to high latency. In [5] techniques are provided for hardware implementation using Advanced Encryption Standard (AES) algorithm. In [6] implementation of FFT processor with pipelined stages is exhibited. In [7] concept of shared memory architecture is explained which is used to reduce the memory usage. [8] and [9] describe the hardware implementation using different platforms.

III. PROPOSED SYSTEM

In this paper, the design of a multiplier for Fully Homomorphic Encryption is proposed. Here the main aim is to improve the security of data transmitted for processing to overcome the drawbacks of earlier stages of cloud computing [1]. To satisfy security needs Gentry put forth [2] FHE scheme. FHE first implementation was performed [4] on a GPU processor with high memory and device capabilities. This FHE scheme is further implemented to develop a means which could solve the urge for security. The core unit of the system is the FFT processor. This processor involves two main operations of summing and shifting. The main concern remains in reduction of time which is achieved using the pipeline structure. The sum unit makes use of Carry Save Adder to reduce the delay by employing pipelined stages. The shift unit performs one bit left shift operation at every stage.

The multiplier incorporates two Fast Fourier Transform processors. One processor is responsible in performing forward operation while the other provides input to the former processor in order to provide the inverse transform output.

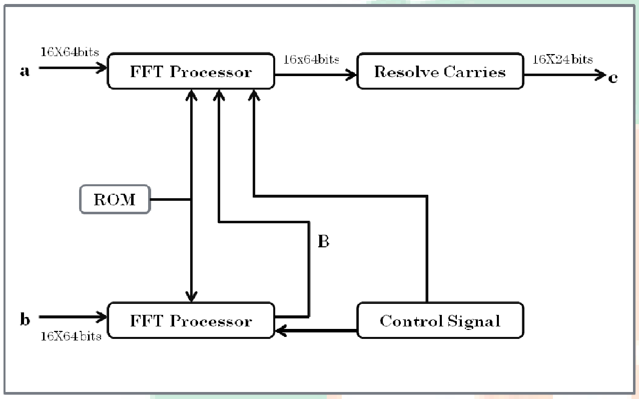


Figure: 4 Block diagram of multiplier for FHE

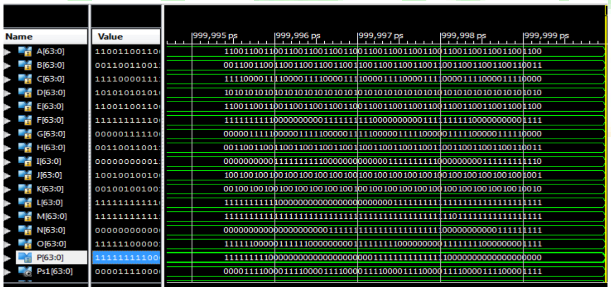
The output of the FFT processor after inverse operation is 16x64 bits. To reduce the bandwidth consumption of the output the size of the output is reduced to 16x24 bits by a resolve carry block. The purpose of resolve carry block is is bandwidth resolution but taking into consideration only the sum units and part of the carry bits.

IV. RESULTS AND DISCUSSION

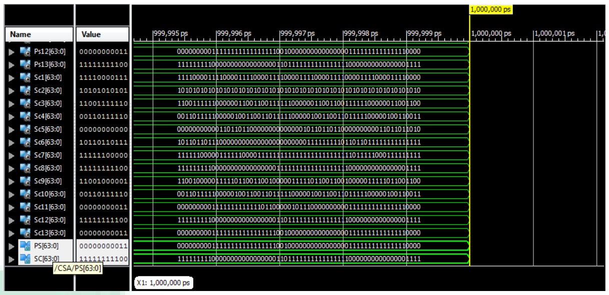
The multiplier design is simulated using Xilinx 14.2 software. The core of multiplier design is the Fast Fourier Transform processor. The processor utilizes 63% of device specifications of Spartan 3E.

A.RADIX 16 UNIT

The radix 16 unit designed using two modules namely the sum unit and shifting unit. The sum unit is the high resolution unit consisting of 16 inputs in pipelined stages producing a single output of partial sum and shift carry.



(a)

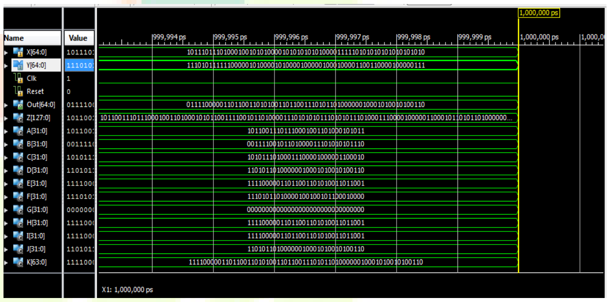


(b)

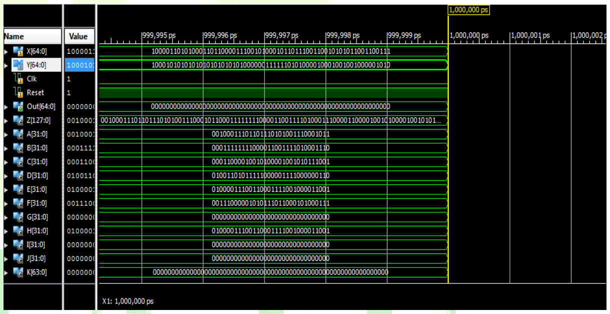
Fig: 5 Simulation results for sum unit of radix 16 module
(a) Input for sum unit (b) Sum and carry outputs of sum unit.

B. Modular multiplication

This module is used to enhance the speed of FFT. The process involves a series of arithmetic and logical operations. One of the inputs to this unit is the twiddle factor.



(a)

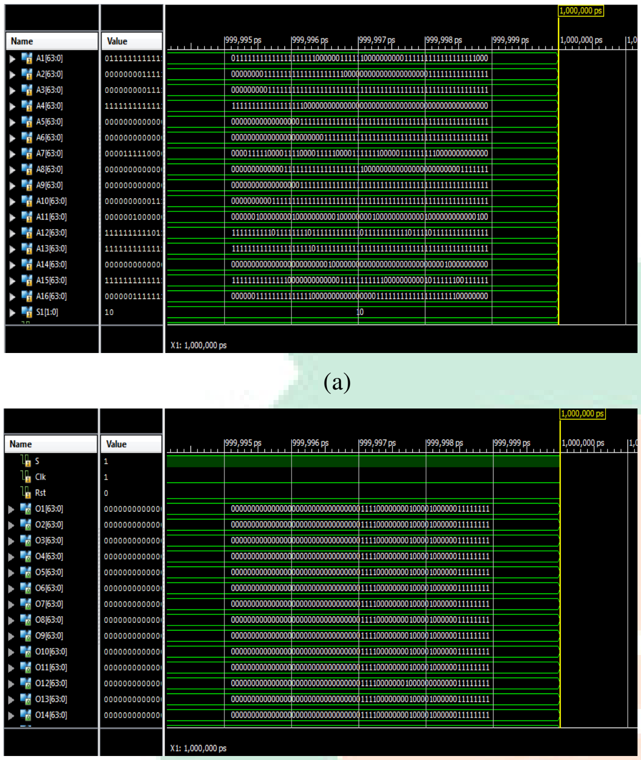


(b)

Fig: 6 Simulation results of modular multiplication unit
(a) Status with reset 0 (b) Status with reset 1

C. FFT PROCESSOR

The FFT processor provides a faster means of processing to provide product in the ciphertext form. This processor makes use of 63% use of Spartan 3E and can be made more efficient using Graphics Processing Unit (GPU) processor.



(a)

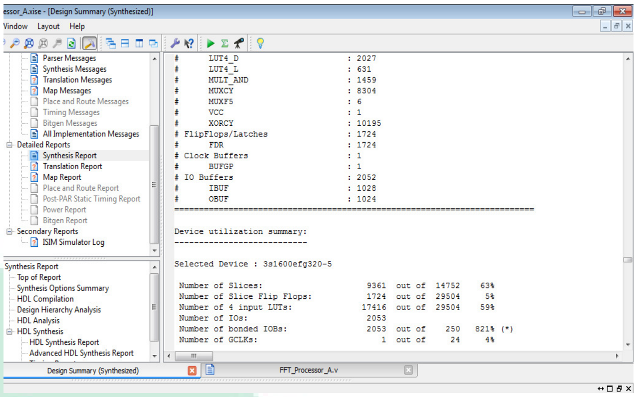
(b)

Fig: 7 Simulation results of Fast Fourier Transform processor

(a) 16x64 bits input (b) 16x64 bits output

D. DEVICE UTILIZATION

The FFT processor is implemented and simulated using Spartan 3E device which is a part of Xilinx 14.2 software. The delay is very much reduced and suitable for faster processing. Among this utilization the major portion is occupied by radix 16 unit which is about 32% of the device capabilities of Spartan 3E.



```
#      LUT4_D                      : 2027
#      LUT4_L                      : 631
#      MULT_AND                    : 1459
#      MUXCY                       : 8304
#      MUXF5                       : 6
#      VCC                         : 1
#      XORCY                       : 10195
# FlipFlops/Latches                : 1724
#      FDR                         : 1724
# Clock Buffers                    : 1
#      BUFGP                       : 1
# IO Buffers                       : 2052
#      IBUF                        : 1028
#      OBUF                        : 1024
=========================================================================

Device utilization summary:
---------------------------

Selected Device : 3s1600efg320-5

 Number of Slices:                     9361  out of  14752    63%
 Number of Slice Flip Flops:           1724  out of  29504     5%
 Number of 4 input LUTs:              17416  out of  29504    59%
 Number of IOs:                        2053
 Number of bonded IOBs:                2053  out of    250   821% (*)
 Number of GCLKs:                         1  out of     24     4%
```

Fig: 8 Device utilization of FFT processor

References

[1] K Bidet E. and Joanblanq (1995), "A fast single chip implementation of complex point FFT," IEEE J. Solid-State Circuits, vol. 30, no. 3, pp. 300-305.

[2] Black J., Rogaway P. and Shrimpton T. (2002), "Encryption-scheme security in the presence of key-dependent messages," SAC., pp. 62-75.

[3] Burnikel C., Fleischer R., Mehlhorn K. and Schirra S. (2000), "Efficient exact geometric computation made easy," Proc. 15th Annu. Symp. Comput. Geometry, pp. 341-350.

[4] Canetti R., Krawczyk H. and Nielsen J.B. (2003), "Relaxing chosen ciphertext security," Proc. Of Cryptography, pp.565-582.

[5] Cohen A. and Parthi K. (2010), "GPU accelerated elliptic curve cryptography in GF(2^m)," Proc. 53rd IEEE Int. MWCSAS, pp. 57-60.

[6] Cui X., Chen Y. and Mei H. (2009), "Improving performance of matrix multiplication and FFT on GPU," in Parallel and Distributed Systems(ICPADS), 15th International Conference on IEEE.

[7] Emmart N. and Weems C.C. (2011), "High precision integer multiplication with a GPU using Strassen's algorithm with multiple FFT sizes," Parallel Process. Lett., vol. 21,no. 3, pp. 359-375.

[8] Gentry C. (2009), "A Fully Homomorphic Encryption Scheme," Ph.D. Dept. Comp. Sci., Stanford Univ.

[9] Jia L., Gao Y., and Tenhunen H. (2000), "Efficient VLSI implementation of radix-8 FFT algorithm," Proc. IEEE Int. Symp. On Circuits and System, ISCAS.

[10] Kalalch K. and David J. P. (2005), "Hardware implementation of large number multiplications by FFT with modular arithmetic," Proc. 3rd Int. IEEE-NEWCAS Conf., pp. 267-270.