Journal of Advanced Research in Technology and Management Sciences (JARTMS)
Volume: 01 Issue: 03 ISSN: 2582-3078 November 2019
Available online at: http:// www.jartms.org

Design and Implementation of Montgomery Modular Using 8X8 Bit UT Multiplier

Vankudoth Venkanna[1], Dr. R.P.Singh[2], Dr. N Ashok Kumar[3]
[1] Research Scholar, Dept. of ECE, Enrolment ID: SSSEC1607S, SSSUTMS, Sehore, Bhopal (M.P.).
[2] Vice Chancellor, Professor of ECE Dept, Sri Satya Sai University of Technology & Medical Sciences, Sehore, Bhopal (M.P.)
[3] Assistant Professor, Head Of the Department ECE, Kasireddy Narayanreddy College of Engineering & Research, JNTUH, Telangana.

Abstract: - Large integer multiplication is the critical operation to design a modular multiplier. The final product can be obtained suitably by adding the outputs of 4X4 bit multipliers. Also two adders are required similarly in the final stage. Now 4x4 bits multiplier which is the basic building block of 8x8 bits UT multiplier is which implemented in its structural model. The fastest design of a circuit can be obtained by structural modeling. This design can run at the technology of 90nm with 1920 LUTs and No. of slices of 960 approximately. In addition, our design can obtain the result of Montgomery modular multiplier for every clock. Compared with other designs on FPGA, our design shows a better performance in term of area-time product.

Key Words: - Modular Multiplier, UT multiplier, FPGA, Montgomery Modular.

I. INTRODUCTION TO MODULAR MULTIPLICATION

With the advancement in communication systems, security is a prime concern which is offered by public key cryptosystems. These systems offer authentication, confidentiality and privacy. Many cryptosystems including RSA, DSA and ECC systems requires modular multiplication for private key generation. [1] P. Montgomery developed an efficient algorithm for the calculation of (A X B) mod M called Montgomery Multiplication algorithm. Montgomery multiplication has been used as a fundamental operation of arithmetic operations in RSA-Algorithm. This paper presents FPGA implementation of scalable architecture for radix-2 Montgomery multiplication algorithm for 1024-bit operand.

In this paper are implemented UT multiplier and comparative study is presented in terms of area and speed. Section I is the introduction about the Modular multiplication. Section II gives a conceptual explanation of the Montgomery Algorithm. Section III explains the Implementation Of 8x8 Bit UT Multiplier; Section IV gives the implementation details and simulation results. and finally Section V concludes with the conclusion.

II. MONTGOMERY MULTIPLICATION

Montgomery Multiplication in 1985 a method for modular multiplication using Residue Number System (RNS) representation of integers is proposed by Peter L. Montgomery. In this method, the costly division operation usually needed to perform modular reduction is replaced by simple shift operations by transforming the operands into the RNS domain before the operation and re-transforming the result after operation. A radix R is selected to be two to the power of a multiple of the word size and greater than the modulus, i.e. R = 2w> M. For the algorithm to work R and M need to be relatively prime i.e. must not have any common non-trivial divisors. With R a power of two, this requirement is easily satisfied by selecting an odd modulus. This also fits in nicely with the cryptographic algorithms that we are targeting, where the modulus is either a prime always odd with the exception of 2or the product of two primes and therefore odd as well. RNS representations of integers are called M residues and are usually denominated as the integer variable name with a bar above it. An integer **a** is transformed into its corresponding M-residue $\bar{a}$ by multiplying it by R and reducing modulo M. The back-transformation is done in an equally straight forward manner by dividing the residue by R modulo M.

Thus here are the following equations as transformation rules between the integer and the RNS Domain:

$$a = \bar{a}R^{-1}$$

Montgomery Multiplication can be defined simply as the product of two M residues divided by the radix modulo M:

$$\bar{c} = \bar{a}\bar{b}R^{-1}(mod M)$$

Division by the Radix is required to make the result again an M-residue. The key concepts of the Montgomery algorithm are the following: i) Adding a multiple of M to the intermediate result does not change the value of the final result; because the result is computed modulo M. M is an odd number. ii) After each addition in the inner loop the least significant bit of the intermediate result is inspected. If it is 1, i.e., the intermediate result is odd, we add M to make it even. This even number can be divided by 2 without remainder. This division by 2 reduces the intermediate result to n+1 bits again. iii) After n steps these divisions add up to one division by 2n.

Radix-2 Montgomery Multiplication Algorithm Let X and Y be two n-bit operands and M be any odd integer which is greater than zero for satisfying radix-2 operation. Montgomery multiplication involves first transformation of operands into Montgomery domain and then after result is re-transformed into Montgomery domain. This conversion process replaces division by several shift operations then Montgomery multiplication process for inputs X, Y, M and output Z is described as follows: Output to be obtained: Z= (X,Y) mod M Where X’=X.2n mod M Y’=Y.2n mod M Z’ = MP(X’, Y’, M) = X Y 2n mod M Hence Z’ = Z 2n mod M Hardware reduction of this algorithm is possible by pre-computation. The values to be added to the intermediate result within the loop can be pre-computed. Delay due to carry propagation must be avoided.

Faster Montgomery Multiplier The motivation behind this optimized algorithm is that of reducing the chip area for practical hardware implementation. This is possible if we can pre-compute four possible values to be added to the intermediate result. These are the four possible scenarios: i) if the sum of the old values of S and C is an even number, and if the actual bit xi of X is 0, then we add 0 before we perform the reduction of S and C by division by 2. ii) if the sum of the old values of S and C is an odd number, and if the actual bit xi of X is 0, then we must add M to make the intermediate result even. Afterwards, we divide S and C by 2. iii) if the sum of the old values of S and C is an even number, and if the actual bit xi of X is 1, but the increment xi*Y is even, too, then we do not need to add M to make the intermediate result even. Thus, in the loop we add Y before we perform the reduction of S and C by division by 2. The same action is necessary if the sum of S and C is odd, and if the actual bit xi of X is 1 and Y is odd as well. In this case, S+C+Y is an even number, too. iv) if the sum of the old values of S and C is odd, the actual bit xi of X is 1, but the increment xi *Y is even, then we must add Y and M to make the intermediate result even. Thus, in the loop we add Y+M before we perform the reduction of S and C by division by 2. The same action is necessary if the sum of S and C is even, and the actual bit xi of X is 1, and Y is odd. In this case, S+C+Y+M is an even number, too. v) The computation of Y+M can be done prior to the loop. This saves one of the two additions which are replaced by the choice of the right operand to be added to the old values of S and C. Algorithm in 5.1 is a modification of Montgomery's method which takes advantage of this idea.

III. IMPLEMENTATION OF 8X8 BIT UT MULTIPLIER

The figure- 1 shows the construction of 8x 8 bit UT multiplier using 4X4 bit blocks using reversible logic gates. In this figure AH-AL are the two 4-bit pairs of the 8 bit multiplicand A. Similarly BH-BL is the two pairs of multiplicand B. The result of 16 bit product can be written as:

P= A x B= (AH-AL) x (BH-BL)

= AH x BH+AH x BL + AL x BH+ AL x BL

The final product can be obtained suitably by adding the outputs of 4X4 bit multipliers. Also two adders are required similarly in the final stage.

Now 4x4 bits multiplier which is the basic building block of 8x8 bits UT multiplier is which implemented in its structural model.

The fastest design of a circuit can be obtained by structural modeling.

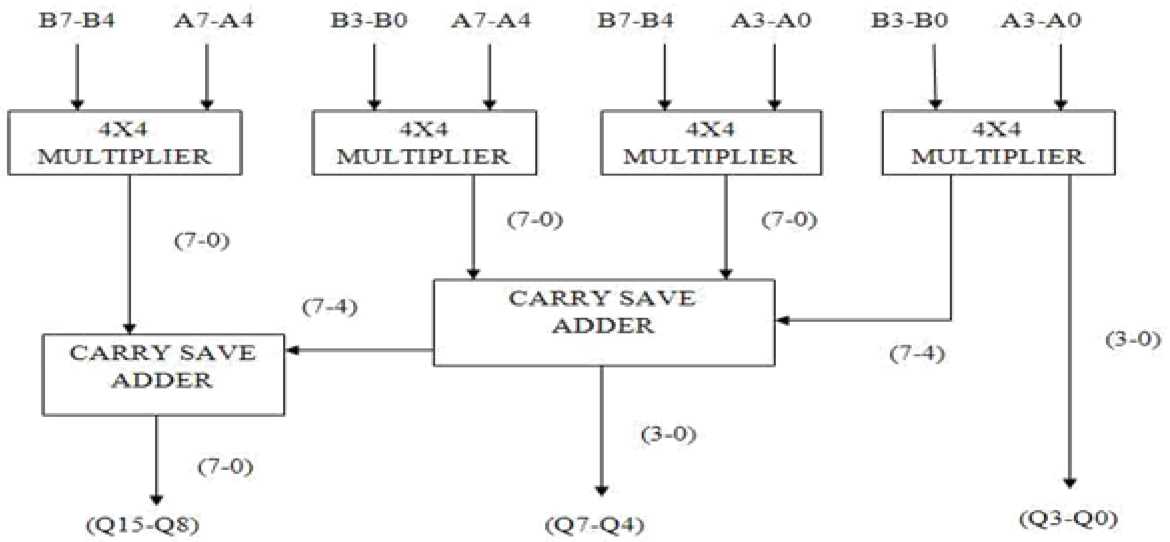

Figure: 1.Showing 8X8 Bits decomposed Vedic Multiplier.

RESULT = (Q15- Q8) & (Q7- Q4) & (Q3-Q0)

IV. IMPLEMENTATION AND TESTING OF UT MULTIPLIER

FPGA IMPLEMENTATION

The FPGA that is used for the implementation of the circuit is the Xilinx Spartan 3E (Family), XC3S100e (Device), vq100 (Package), -5 (Speed Grade). The working environment/tool for the design is Xilinx ISE14 .3i.

Device: Spartan 3E((Family))	**XC3S100e-5vq100**
Technology	90nm
I/O standards	232
No:of slices	960
4 input LUT"s	1920
Bonded IOB"s	66

Device family	**Sparta n 3E**
Target device	XC3S1 00e
Package	Vq100
Speed Grade	-5

Table: 1. Showing Summary of FPGA features

V. CONCLUSION

In this paper, we introduce the 8X8 Bit UT Multiplier algorithm based on UT Multiplier algorithm. Then we propose a design of 8x8-bit multiplier using UT Multiplier algorithm. Finally, a 8x8 four-stage pipelined modular multiplier is constructed by combining the features of Montgomery modular multiplication algorithm with that of UT Multiplier algorithm appropriately. Compared with other designs, the synthesized result of our design shows a significant improvement in latency and tradeoff between area and performance.

REFERENCES

1) P. L. Montgomery, “Modular Multiplication Without Trial Divisions,” Mathematics of Computation, vol. 44, no.170,pp. 519–521, 1985.
2) P. Barrett, ”Implemntation The Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor,” Advances in cryptology CRYPTO’86, pp. 311-323,1986.
3) X. Yan, G. Wu, D. Wu, F. Zheng, and X. Xie, ”An Implemntation of Montgomery Modular Multiplication on FPGAs,” in 2013 International Conference on Information Science and Cloud Computing, Dec2013,pp. 32–38.
4) V.Venkanna1 D.Spoorthy, “ High Speed Low Power Veterbi Decoder Design for TCM Decoders” vol. 02, no.08,pp. 175–180, 2014.
5) Kavyashree S , Dr. Uma B V , “Design and Implementation of different architectures of montgomery modular multiplication” 2017 2nd IEEE International Conference On Recent Trends in Electronics Information & Communication Technology (RTEICT), May 19-20, 2017, India, pp. 1101-1105.
6) Alan Daly, William Marnane, “Efficient Architectures for implementing Montgomery Modular Multiplication and RSA Modular Exponentiation on Reconfigurable Logic”
7) Nitha Thampi, Meenu Elizabath Jose, “Montgomery Multiplier for Faster Cryptosystems”, Global Colloquium in Recent Advancement and Effectual Researches in Engineering, Science and Technology (RAEREST 2016), Procedia Technology 25 (2016) pp. 392 – 398.
8) K. Pratibha, Muthaiah Rajappa, “Survey on Hardware Implementation of Montgomery Modular exponentiation”, International Journal of Pure and Applied Mathematics, Volume 119, No. 12, 2018, pp. 13437-13452.
9) Harmeet Kaur, Mrs.Charu Madhu, “Montgomery Multiplication Methods - A Review”, International Journal of Application or Innovation in Engineering & Management, Volume 2, Issue 2, February 2013.