12th International Conference on Recent Innovations in Science, Engineering and Management
Sri Venkateswara College of Engineering & Technology,Etcherla, Srikakulam (ICRISEM'18)
17th February 2018 www.conferenceworld.in ISBN: 978-93-87793-01-9

Design of Low Power Versatile Bit-Serial Multiplier in Finite Fields GF (2^m)

M.Srinivasan[1] G M. Tamilselvan[2]

[1]*Department of Electronics and Communication Engineering,*
KGiSL Institute of Technology, Coimbatore, (India)
[2]*Department of Electronics and Communication Engineering,*
Bannariamman Institute of Technology, Sathyamangalam,(India)

ABSTRACT

Finite field arithmetic is the most important component in applications like cryptography, computer algebra and error correcting codes. In this paper we have proposed an efficient VLSI design for versatile bit-serial multiplier in finite fields GF (2^m).The versatile multiplier designed here modifications done by reducing the unwanted switching activity removed by clock gating scheme. Our design provides a solution to the power reduction. For functional verification, the design is simulated using I-sim simulator within the software. The power report for the architecture is obtained by using the X-power analyzer tool. Experimental results prove the efficiency of our design by comparing with the existing design.

Keywords: Finite Field Multiplication; Versatile; Verilog HDL; Xilinx; VLSI; Low power.

I. INTRODUCTION

Finite field arithmetic has became a backbone for most of the advanced technologies in the applications like cryptography, error correction and computer algebra [1]. Among this, cryptography plays an important role the field of network security, memory confidentiality and integrity, biometric template security etc. Even though a number of cryptographic techniques exist till date, techniques like Diffie-Hellman, Advanced Encryption Standard and Elliptic Curve Cryptography are preferred more because of their use of finite field arithmetic for computation. Finite field multiplication is the vital operation among the finite field operations [2].

The low level in the hierarchy of ECC computation is the point multiplication operation, which is completely based on the adapted finite field multiplication operation. Since the processing time of the complete ECC process relies on the incorporated finite field multiplication process, multiplier occupies the most remarkable position in the designing of the ECC processor also the selection of the multiplier is a critical task while designing the processor. Hence, by modifying the basic structure of the multiplier design, a large contribution for the development of the cryptosystem can be made [3]. This feature attracted the researchers to concentrate much in this area for the implementation of efficient finite field architecture. The main drawbacks of existing multipliers are the fact that operand conversions from integer to the Montgomery domain representation and

vice-versa are necessary before, respectively, after the multiplication. This, however, is a general drawback of Montgomery's method in opposition to other techniques. Also, this becomes negligible once a sufficient number of modular multiplications need to be performed in a row.

Three types of finite field $GF(2^m)$ multipliers are well known among the existing types. They are bit serial [4], bit parallel [5] and a hybrid of both [6], incorporating partial bit serial and partial bit parallel characteristics. Even though the computation speed of the hybrid multiplier is higher than that of the bit serial multiplier, the resource consumption is high compared to the bit parallel multiplier. Among the three, bit serial exhibit a complexity of $O(m)$, whereas the bit parallel has a complexity of $O(m^2)$. Apart from these three multipliers, there exists another type of classification based on the adopted basis representation. They are normal, digit and polynomial [7-9]. The multiplier's requirement is directly proportional to the irreducible polynomial. Hence this can be reduced by adopting an irreducible polynomial is of All-One Polynomial (AOP) [10] or a trinomial [11] and a redundant field representation [12, 13].

Most of the existing finite field multipliers operate over a fixed field, i.e., a new multiplier is needed if there is a change on the irreducible polynomial that defines the field elements. In this paper, we propose a new versatile bit-serial multiplier in finite fields $GF(2^m)$ and the canonical (standard, polynomial) basis representation where the field set of parameters can be changed according to the application environment.

S. S.Roy etal. [14] planned theoretical modeling of elliptic curve scalar multiplier on LUT-based FPGAs for area and speed. In that method two primitives employed in elliptic curve scalar multiplier architecture (ECSMA) executed on k input lookup table (LUT) based field-programmable gate arrays to approximate the delay of dissimilar feature. It was employed to find out the optimal number of pipeline stages and the ideal placement of each stage in the ECSMA. In order to carry out point addition and doubling in a pipelined data path appropriate scheduling was formed. The three stage pipelined architecture for double and add based scalar multiplication is carried out on Xilinx Virtex-V platforms over $GF(2^{163})$.

The execution employs a new pipelined bit-parallel Karatsuba multiplier that has sub quadratic difficulty. In this plan competent choice of scalar multiplication algorithm, optimized field primitives, balanced pipeline stages, and improved scheduling of point arithmetic effected in a high-speed architecture with a considerably small area.

G. D. Sutter et al.[15] planned competent elliptic curve point multiplication by means of digit-serial binary field operations. In their plan a novel high-speed point multiplier for elliptic curve cryptography by means of either field programmable gate array or application specified integrated circuit technology. Their plan modified a digit-serial approach in GF multiplication and GF division in order to erect a competent elliptic curve multiplier by means of projective coordinates. This plan attained point multiplication over $GF(2^{163})$ in 19.38 μs in Virtex-E devices and in 5.48 μs in Virtex-5.

K. Sakiyama et al.[16] executed a tripartite modular multiplication. Systematic approach was executed for modular multiplication in that multiplication for maximizing a level of parallelism. In this plan a modular multiplier by means of these algorithms attains a higher speed comparing to the other algorithms for modular multiplication. [17] Offer a low latency systolic Montgomery multiplier over $GF(2^m)$ based on irreducible

pentanomials. A competent algorithm was offered to decompose the multiplication into a number of independent units to make easy parallel processing. Moreover, a new so-called pre-computed addition technique was brought in to further decrease the latency.

In [18] an area-time-efficient systolic structure for multiplication over $GF(2^m)$ based on irreducible all-one polynomial (AOP) was offered. The plan implemented a new cut-set retiming to decrease the period of the critical-path to one XOR gate delay. It is further revealed that the systolic structure can be decayed into two or more parallel systolic branches, where the pair of parallel systolic branches has the similar input operand, and they can distribute the similar input operand registers.

In [19] new multipliers for Montgomery multiplication described on binary fields $GF(2^m)$. Dissimilar to condition of the art Montgomery multipliers, a linear feedback shift register (LFSR) is employed as the main building block. The planned multipliers are for dissimilar classes of irreducible polynomials: general, all one polynomials, pentanomials and trinomials.

A. Zakerolhosseini and M. Nikooghadam executed an architecture for a versatile polynomial basis multiplier over $GF(2^m)$[20]. The plan and circuit execution of polynomial basis multiplier architecture over Galois Fields $GF(2^m)$ was examined. The architecture holds up field multiplication of two m-term polynomials where m is a positive integer. This architecture presents low latency, polynomial basis multiplication irreducible polynomial $P(x) = xm+pkt.x\ kt+...+p1.x+1$ with $m \geq kt + 4$ is dynamically reconfigurable.

Effects of the difficulty study illustrate that the suggested architecture necessary less logic resources compared to presented sequential polynomial basis multipliers. A finite field $GF(2^m)$ multiplier is said to be versatile, if a multiplier designed for a specific bit size m can also adapt itself for performing the operation for any inputs with bit $size<m$. Most of the existing architecture for finite field multiplier is designed with constant field sizes. Hence if there is a requirement for more than one multiplier or a multiplier with some other field size as in the case of applications like cryptography, then the complete architecture has to be redesigned [21].

In our work, an efficient versatile architecture for the Most Significant Bit (MSB)-first, bit-serial, polynomial basis multiplier over $GF(2^m)$ is designed and implemented, where m is the degree of the irreducible polynomial and can be varied easily according to the application requirements. The proposed solution is a modification to the architecture implemented in [20]. In the proposed modified architecture, all the AND gate logic are replaced with an array of tri-state buffer for reducing the hardware area occupied by the complete architecture. Also a clock gating scheme is used for removing the unwanted switching of the registers, thereby contributing the reduction in dynamic power consumption.

III.PROPOSED METHOD

Versatility is an important property the hardware industry lacks and trying to establish as much as possible. To survive in this booming technological word, the new designs should be of an adjustable one, which processes

the versatile property. In our research, the problem for research we have considered is the conversion of a conventional MSB bit serial finite field multiplier $GF(2^m)$ into a finite field multiplier with a maximum bit length m that can reconfigure itself for performing any finite field multiplication with bit length $l<m$, where l is the bit length for the required multiplication. For clear interpretation of the work, we start with the problem of transforming a conventional multiplier into a versatile multiplier before discussing the proposed versatile architecture. Fig.1, depicts the problem in using a $GF(2^9)$ multiplier as a $GF(2^4)$multiplier. Algorithm 1 shows the conventional MSB-first algorithm we have considered in this work. In the algorithm, we have used the notation C^i to represent the value of C after i iterations. The main difference between MSB-first multiplication and the LSB-first multiplication is that the former use the bits in the B register from MSB to LSB, whereas the latter use the bits from LSB to MSB. As the iteration steps forwardC^{i+1},c_mP and b_iA are added to form the C^i value for the corresponding iteration. Where, $c_m$is the MSB bit ofC^{i+1}.

The final output is stored in the R register. Comparing MSB-first multiplier with the LSB-first multiplier, LSB first include shorter critical path delay, thereby contributing in the total processing speed of the design, whereas it requires an extra register for temporarily storing a data.

Algorithm: 1

Input: Polynomial A, B and P

Output: R=ABmod P

1. Cm←0;
2. for i=m-1 to 0 do
3. Ci← x(Ci+1+cmP+biA);
4. end for
5. Return : R←Co/x.

As highlighted in the Fig. 1, for using a $GF(2^9)$ multiplier as a $GF(2^4)$multiplier, the feedback should be from the $R(3)$and not from the $R(8)$ as in the case of the $GF(2^9)$.By designing a technique to overcome this problem of selecting the suitable feedback path based on the m value the conventional multiplier can be resigned to a multiplier incorporating a versatile property.

The P bit registers with oriented on OR operation is done initially so only for first clock cycle it is computed then it always available for other clock cycle for given operand length if operand length changes then it will compute for that particular cycle after that it will available for all clock cycle. In this way by using serial structure of OR gates critical path is reduced considerably. But number of control signal has increased compared to previously designed structures.

The power consumption is reduced because of gated clock technique is used. Control signals are generated such a way that they should on only those flip-flops which are required to be on during our operation it will turn off flip-flops which are not used in operation. In this way power consumption is reduced.

A vast number of solutions for this problem have been reported in the literatures so far. Among the solutions that exist, we have considered the one that is proposed in [20]. Fig. 2, illustrates the solution for the problem

provided in [20]. In this work, for automatically reconfiguring the feedback based on the irreducible polynomial value in the *P* register control logic by adopting the basic gates and an array of tri-state buffer is designed and a clock gating logic is designed for reducing the unnecessary transitions in the registers which are not included in the current multiplication operation.

Although the solution seems to be efficient, a few drawbacks remain in the architecture such as overuse of logic gates count and the unnecessary switching of output lines in the R registers, which lies within the feedback loop in the architecture. By redesigning the architecture in Fig. 2 based on the modifications for eradicating the problems discussed above. In our proposed technique, we have redesigned the architecture in Fig. 2 and the proposed architecture is as shown in Fig. 3. In Galois field multiplication operation is done as A×B mode P. so according to that first multiplication of A and B happens so by multiplying 4 bit by 4 bit we get 8 bit that are 00111010 but output should not be more than m-1 degree therefore irreducible polynomial helps us to reduce that degree by XOR operation.

If the bits are more even that then of irreducible polynomial then zero padding is done in that polynomial then XOR operation is performed so after that degree will be reduced, still degree is higher than m-1 then previous operation is performed and try to compress it in m-1 degree. So according to that 111010 XOR with 101010 is done and the answer is 10000 further XOR operation is done with 10101 and the answer is 0101.

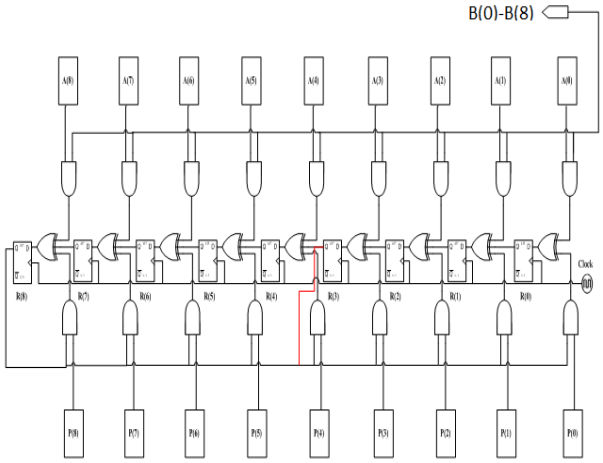

Figure.1 Problem in Redesigning a $GF(2^9)$ to $GF(2^4)$

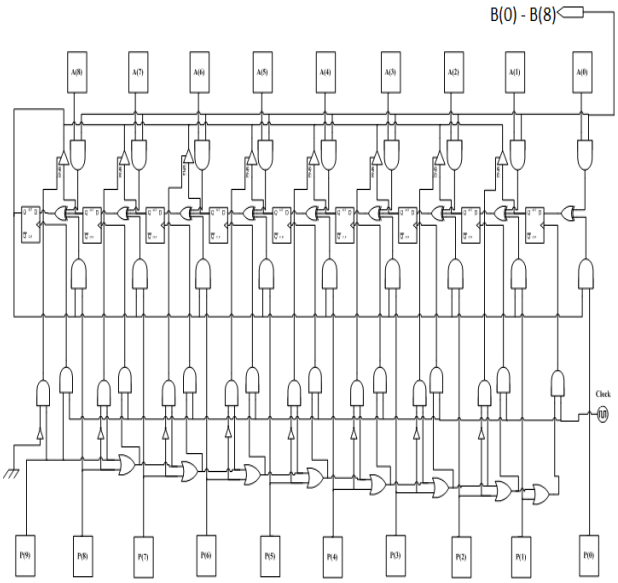

Figure.2 Proposed Versatile Multiplier

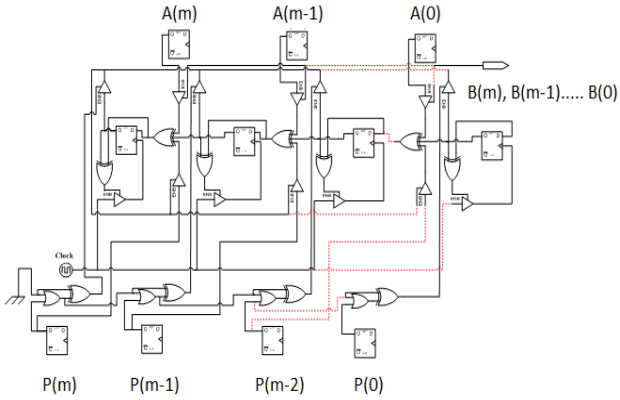

Figure.3 Proposed Versatile Bit-Serial Multiplier in Finite Fields *GF* (2^m)

IV. RESULT AND DISCUSSION

The complete architectures including the work proposed in [20] and our proposed architecture are coded using verilog HDL using the Xilinx software version 14.5. Vertex-4 is set as a target device with power optimization. The XST tools in the Xilinx synthesize the designs and map to the target device. Experiments are carried out in a PC with windows 7 operating system with 4 GB ram and a core i3 Intel processor. For illustrating the efficiency in terms of functioning, for the proposed let us consider an example of finite field multiplication with *A=15*, *B=15* and *P=19*, for which in pen and paper mathematical calculation we obtain the result as *R=10*. This in binary can be represented as *A=1111*, *B=1111*, *P=100011* and *R=10101*. The step by step partial product, their bit transition and clock for the work proposed in [20] and our work is tabulated in Table 1 and 2 below.

Table 1 Partial product for each global clock and their corresponding clock signal for [20]

R3	R2	R1	R0	Clk3	Clk2	Clk1	Clk0
0	0	0	0	1	1	1	1
1	1	1	1	1	1	1	1
0	0	1	0	1	1	1	1
1	0	1	1	1	1	1	1
1	0	1	0	1	1	1	1

Table 2 Partial product for each global clock and their corresponding clock signal for our work

R3	R2	R1	R0	Clk3	Clk2	Clk1	Clk0
0	0	0	0	1	1	1	1
1	1	1	1	1	1	1	1
0	0	1	0	1	1	0	1
1	0	1	1	1	0	0	1
1	0	1	0	0	0	0	1

Experimental Result of Power

The power reduction is achieved in our architecture by regulating the unwanted clock signal flow, thereby reducing the unwanted switching activity in the flip-flops.

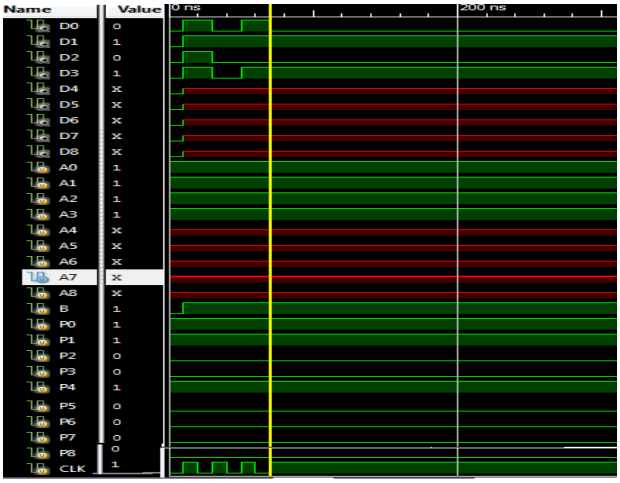

Figure.5 Simulation for our Proposed Versatile Multiplier

When the reduce bits are computed by cope with a dependency among these bits, e.g., $y_d{-}2$ depends on $y_d{-}1$,

$y_d{-}3$ depends on $y_d{-}2$. The critical path of the reduce bit computation can be decreased by eliminating this dependency. We can achieve this by substitution of all reduce bits y_j with $j > i$ into the equation for y_i. Thus, we get an equation for yi depending on s(t) and p(t) only, but not on other reduce bits for a representative example where m = 8 and d = 4.

Table 3 Power comparison between [20] and our proposed work with respect to increase in clock frequency for m = 9

Frequency (MHz)	[20]			Our work		
	Power (mW)			Power (mW)		
	Dynamic	Static	Total	Dynamic	Static	Total
100	21	219	241	24	167	191
200	46	220	266	50	167	217
300	74	220	294	50	167	217
400	101	221	322	91	168	259
500	129	221	351	113	169	282

The power consumption by our proposed versatile multiplier for m=9 increases with respect to the increase in frequency than existing work. The comparison results reported in table 3 justify the low power consumption of our proposed versatile finite field multiplier.

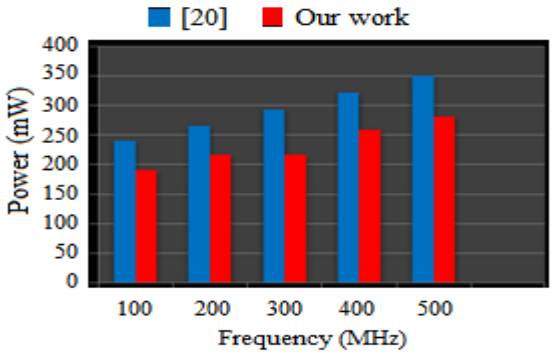

Figure.6 Power Comparison for m = 9

V.CONCLUSION

In this paper, we have studied the drawbacks in the existing works for finite field versatile multiplier design. A modified architecture with a solution to the problem identified in the similar existing work was proposed. The clock gating scheme in the existing design lacks the reduction of unwanted switching within the feedback loop. The modifications done in this work contribute to power reduction. From the experimental results, the maximum clock frequency of proposed multiplier achieves 49.751 MHz for m=9. In future, the proposed versatile bit serial multiplier will be analyzed for area and performance efficiency applied for an Elliptic curve cryptography processor.

REFERENCES

[1] A. P. Fournaris and O. Koufopavlou, "Versatile multiplier architectures in GF (2^k) fields using the Montgomery Multiplication Algorithm", *INTEGRATION, the VLSI Journal*, Vol. 41, No. 3, pp. 371-384, 2008.

[2] A. J. Menezes, P. C. Van Oorschotand S. A. Vanstone, "Handbook of Applied Cryptography", *CRC Press*, 1997.

[3] I.Shparlinski, "Finite Fields: Theory and Computation", *Kluwer Academic Publishers,*Vol. 477, 1999.

[4] H. Li and C.N. Zhang, "Efficient cellular automata based versatile multiplier for GF(2m)", *J. Inform. Sci. Engg,*Vol. 18, No. 4, pp. 479-488, 2002.

[5] C. K. Kocand B. Sunar, "Low-complexity bit-parallel canonical and normal basis multipliers for a class of finite fields", *IEEE Transactions on computers*, Vol. 47, No. 3, pp. 353-356, 1998.

[6] C. Paar, P. Fleischmannand P. Soria-Rodriguez, "Fast arithmetic for public-key algorithms in Galois field with composite exponents", *IEEE Transactions on computers*, Vol. 48, No. 10, pp. 1025- 1034, 1999.

[7] C. Paarand N. Lange, "A comparative VLSI synthesis of finite field multipliers", *Proceedings of the Third International Symposium of Communication Theory and Its Applications, Lake District, UK*, pp. 10-14 , 1995.

[8] L. Songand K.K. Parhi, "Low-energy digit-serial/parallel finite field multipliers", *Journal of VLSI signal processing systems for signal, image and video technology*, Vol. 19, No. 2, pp. 149-166, 1998.

[9] G. Orlandoand C. Paar, "A scalable GF(p) elliptic curve processor architecture for programmable hardware", *Proceedings of the Cryptographic Hardware and Embedded Systems-CHES, LNCS*, Springer, Paris, Vol. 2162, pp. 348-363, 2001.

[10] M.A. Hasan, M. Wangand V.K. Bhargava, "Modular construction of low complexity parallel multipliers for a class of finite field GF(2m)", *IEEE Transactions on Computers*, Vol. 41, No. 8, pp. 962-971, 1992.

[11] H. Wu, "Low complexity bit-parallel finite field arithmetic using polynomial basis", *Proceedings of the cryptographic hardware and embedded systems-CHES, LNCS*, Springer, *Worcester, MA*, Vol. 1717, pp. 280-291, 1999.

[12] W. Geiselmann and H. Lukhaub, "Redundant representation of finite fields", *In International Workshop on Public Key Cryptography*, Springer, Berlin Heidelberg, pp. 339-352, 2001.

[13] H. Ho, "Design and implementation of a polynomial basis multiplier architecture over GF (2 m)",*Journal of Signal Processing Systems*, Vol. 75, No. 3, pp. 203–208, 2014.

[14] S.S. Roy, C. Rebeiro and D. Mukhopadhyay, "Theoretical modeling of elliptic curve scalar multiplier on LUT-based FPGAs for area and speed", *IEEE Transactions on Very Large Scale Integration (VLSI) systems*, Vol. 21, No. 5, pp. 901-909, 2013.

[15] G.D. Sutter, J. P. Deschamps and J.L. Imana, "Efficient elliptic curve point multiplication using digit-serial binary field operations", *IEEE Transactions on industrial electronics*, Vol.60, No.1, pp. 217-225, 2013.

[16] K. Sakiyama, M. Knezevic, J. Fan, B. Preneel and I. Verbauwhede, "Tripartite modular multiplication", *Integration, the VLSI Journa*l, Vol. 44, No.4, pp. 259-269, 2011.

[17] J. Xie, J. jun He, and P. K. Meher, "Low Latency Systolic Montgomery Multiplier for Finite Field GF(2m) Based on Pentanomials",*IEEE Transactions On Very Large Scale Integration (Vlsi) Systems*, Vol. 21, No. 2, pp. 385-389,2013.

[18] J. Xie, P. K. Meher and J. He, "Low-Complexity Multiplier for GF (2m) Based on All-One Polynomials", *IEEE Transactions On Very Large Scale Integration (Vlsi) Systems*, Vol. 21, No. 1, pp. 168-173, 2013.

[19] M. Morales-Sandoval,C. Feregrino-Uribe and P. Kitsos, "Bit-serial and digit-serial GF(2m) Montgomery multipliers using linear feedback shift registers", *IET Computers & Digital Techniques*, Vol. 5, No. 2, pp. 86-94, 2011.

[20] A. Zakerolhosseiniand M. Nikooghadam, "Low- power and high-speed design of a versatile bit-serial multiplier in finite fields GF(2'")",*Integration. the VLSI Journal*, Vol. 46, No. 2, pp. 211-217, 2013.

[21] C. Paar, P. Fleischmann, P. Soria-Rodriguez, "Fast arithmetic for public-key algorithms in Galois field with composite exponents", IEEE *Transactions on Computers*, Vol. 48, No. 10, pp. 1025-1034, 1999.