JAGAN A[1], NAGARAJAN V[2]

Department of Computer Science and Engineering, Surya Group of Institutions, Villupuram, TamilNadu, India (1), Department of ECE, Adhiparasakthi Engineering College, Melmaruvathur, TamilNadu, India (2)

A comprehensive performance investigation on ingenious ECC co-processor architecture for Different Multipliers

Abstract. *This paper proposes an Elliptic Curve Cryptography (ECC) co-processor over GF(2^{256}), based on the Montgomery scalar multiplication algorithm and provides a comprehensive evaluation of the architecture when different multipliers are involved in the scalar multiplication. The multipliers, namely array multiplier, modified Booth multiplier and hybrid encoded low power (HELP) multiplier are considered for the study. The proposed architecture is designed using Spartan3E family device XC3S1600E and synthesized using Modelsim 5.7.*

Streszczenie. *W artykule zaproponowano algorytm mnożenia bazujący na mnożeniu skalarnym typu Montgomery. Rozważano różne architektury – matrycowe, mnożnik typu Booth i hybrydowy zakodowany mnożnik małej mocy HELP. (**Badania właściwości koprocesorów ECC w zastosowaniu do mnożników**)*

Keywords: Elliptic curve cryptography, public key cryptography, Galois fields, key length, Montgomery scalar multiplication algorithm, Array multiplier, Modified booth multiplier, HELP multiplier.
Słowa kluczowe: mnożniki, algorytm skalarnego mnożenia, koprocesopr ECC

Introduction

Secured transmission and storage of the information are the paramount requirements in the emerging information technology. Techniques for realizing secure information handling are provided by cryptography. The classical goals of cryptography are secrecy and authentication: to protect information against unauthorized disclosure and modification. In the recent years, research in cryptography has addressed a broad range of more advanced questions, ranging from the authorization of user access to computer systems over secure electronic voting schemes to the realization of untraceable electronic cash. The Elliptic Curve Cryptography (ECC) has provided a matured, computation efficient and secure data transaction in the family of public key cryptography (PKC), over the last 20 years [1]-[2]. The fundamental operations of elliptic curve (EC) systems are based on the intractability of integer factorization and discrete logarithm problems. The ECC can give equivalent level of security with a much smaller key length compared with another one public key cryptosystem RSA [3]. These smaller key sizes features are attractive for security applications where computational power and integrated circuit space is limited, such as smart cards, personal computer cards, and wireless devices [4]-[5].

Yinan Kong has presented a way to optimize the improved Barrett modular multipliers for PKC [6]. The modular multiplier has been optimized by evaluating different versions of the improved Barrett algorithm and implemented on field programmable gate array (FPGA). The resulting optimized multiplier is of twelve bit length and uses separate multiplication and reduction. Marisa W. Paryasto et al have suggested the use of composite field to implement finite field multiplication in ECC implementation [7]. The composite field multiplier has been implemented using Look-Up Table (LUT) in ground field and Karatsuba Offman algorithm has been used for the extension of field multiplication. This generic architecture has been modeled in very-high-speed integrated circuits hardware description language (VHDL) for the target device Altera DE-2. Mark Hamilton et al have provided a comparison of different modular multipliers in ECC, when working with a Mersenne prime modulus [8]. The different multipliers are compared for speed, area and power consumption when implemented on a Virtex 5 FPGA. Justin Hensley et al have introduced a counter flow organization, in which the data bits flow in one direction and the Booth commands piggyback on the acknowledgments flowing in the opposite direction [9]. Moreover, the arithmetic and shifter units are merged together to obtain significant improvement in area, energy as well as speed by using the overlapped execution of multiple iterations of the Booth algorithm in a 0.18µm TSMC process. Sandeep Kumar et al have proposed hardware implemented of elliptic and hyper-elliptic curve cryptography over GF(2^m) using the digit serial multipliers [10]. Al-Somani et al have presented an ECC on a Nano FPGA [11]. The ECC crypto-processor has been modeled using VHDL and synthesized on Actel IGLOO AGLN250V2-VQFP100 Nano FPGA. Since the limited number of resources available on Nano FPGAs, the synthesis results showed that the targeted Nano FPGA cannot exceed the values of $m \leq 11$ bits.

The level of security offered by the ECC system depends on the complexity of the algorithm used in encryption. In other side, the computation complexity depends on the efficiency, speed of EC scalar multiplications, and finite field that is defined over. Expansion of any ECC architecture to a system that requires higher level of security mandates the increased key length. The additional resource binding with the increased key size will largely rest on the type of multiplier used as the most time/resource consuming operation in the ECC is scalar multiplication. Hence a careful selection multiplier can led to enhance performance of the ECC architecture irrespective of the algorithm. The objectives of this paper are in two fold, the first is to develop an efficient ECC processor over GF(2^{256}) based on the Montgomery scalar multiplication. Second aim is to provide a comprehensive comparison of performance and resource utilization the architecture when different multipliers (array, modified booth and hybrid multipliers) are used. The architectures are designed and implemented using spartan3E family device XC3S1600E using Modelsim 5.7 and Xilinx 9.2i.

Multipliers

The performance of the ECC processor is mainly based on the effectiveness of multipliers used in the Montgomery scalar multiplication algorithm. The proposed ECC architecture over GF(2^{256}) is designed and analyzed using three different multipliers namely array multiplier, modified Booth multiplier and hybrid multiplier.

Array Multiplier

An array multiplier is a digital combinational circuit that is used for the multiplication of two binary numbers by employing an array of full adders and half adders. This array is used for the nearly simultaneous addition of the various product terms involved. An array multiplier exhibits a vast improvement in speed over the traditional bit serial multipliers in which only one full adder along with a storage memory is used to carry out all the bit additions involved and also over the row serial multipliers in which product rows (also known as the partial products) are sequentially added one by one via the use of only one multi-bit adder. The tradeoff for this extra speed is the extra hardware required to lay down the adder array. But with the much decreased costs of these adders, this extra hardware has become quite affordable to a designer.

The rules for performing the binary multiplication can be stated as follows

a. Check each and every bit in the multiplier.
b. If the multiplier bit is 1, then multiplicand is simply copied down and represents the product line.
c. If the multiplier digit is a 0, then product line is represent as 0.

The well designed array multiplier circuit will have the following three capabilities as inherent properties. An example of typical array multiplication is represented in Fig.1.

a. Identifying a bit as either '0' or '1'.
b. Left shifting operation and
c. Add all the partial products

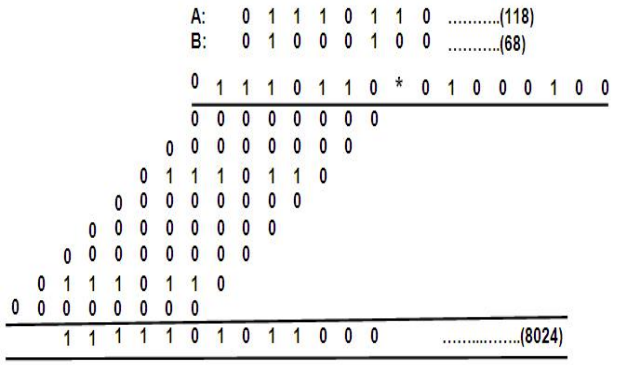


Fig. 1 Example of Array multiplier

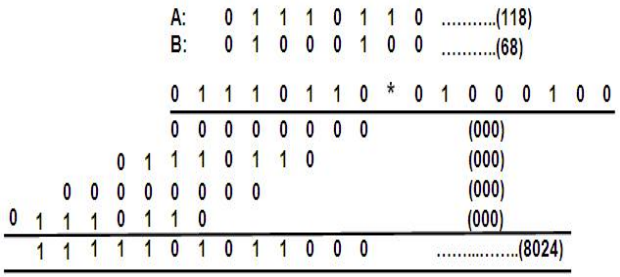


Fig. 2 Example of modified Booth multiplier

Modified Booth Multiplier

The modified Booth multiplier is another commonly used multiplier in digital systems, which can reduce the number of partial products by a factor of 2. The two important steps of Modified booth multiplication are shifting and addition. Generally this type of multiplication demands more number of shifting operations than additions. Shifting operations consume less power and comparatively faster than addition. A representative case of such multiplication is presented in Fig.2. The salient features of this multiplication are:

- Only 'n/2' clock cycles are needed for 'n-bit' multiplication as compared to 'n' clock cycles in Booth's algorithm.
- For even 'n', the two's complement multipliers are handled automatically, whereas for odd 'n' an extension of sign bit is required

HELP multiplier

High speed multipliers are fundamental elements in the arithmetic based systems such as ECC co-processors. The hybrid encoded low power (HELP) multiplier is designed using spurious switching suppression technique (SSST). Number of bits in the input data plays vital role in the multiplication process. The dynamic power and switching activity of multiplication is mainly depending upon the input pattern in the given input. The HELP multiplier depends on the number of 1's and their position in the input-multiplier data. The gross switching activity of HELP multiplier is reduced to 46% compared to modified booth multiplier [13]. As a result of reduced of switching activity, power consumption also reduced compared to modified booth multiplier. The detailed flow of HELP multiplier is given in [13], which is shown in the Fig.3.

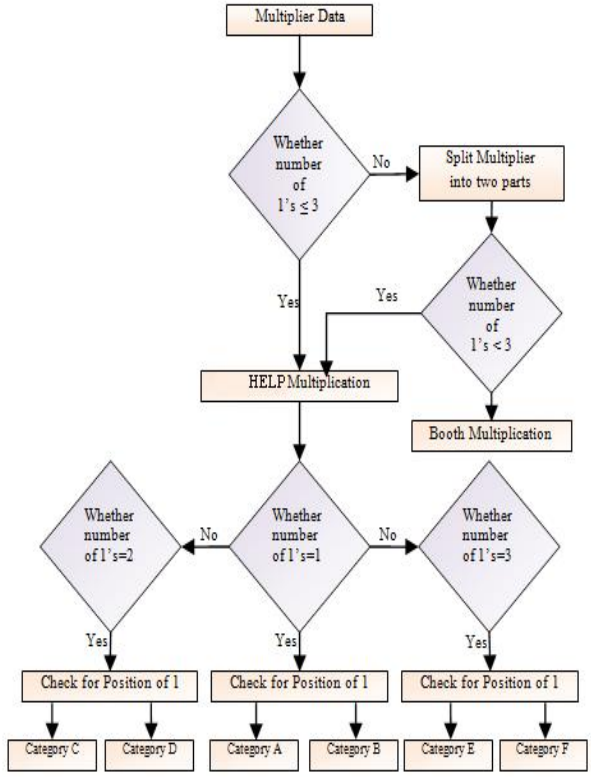


Fig.3 Flow chart of the hybrid encoded low power multiplier [13]

According to the conventional shift and add multiplications, the number of partial products are equal to the number of bits in the multiplier input. The number of partial products can be reduced by half using Booth recoding. Through the encoding techniques, the partial products can still be reduced which in turn reduces the switching activity and hence the power consumption. The operation can be defined according to the number of 1's and its position in the multiplier. When the number of 1's in the multiplier data is less than or equal to 3, the control directs to the HELP multiplication technique, else the control splits the multiplier in to two parts. Again the number of 1's in the part of the multiplier is verified.

If the number of 1's is more than three, the control goes to Booth multiplication. For the failed case the control goes

to HELP multiplication technique. If the number of 1's in the multiplier is one and depends upon its position, the control goes to execute the operation in category A or B. If the number of 1's in the multiplier is two and depends upon its position, the control goes to execute the operation in category C or D. Otherwise the number of 1's in the multiplier is three and depends upon its position, the control goes to execute the operation in category E or F. The operation of hybrid encoding rule is stated in [13], [14], which are also shown in Table I.

Table 1 Hybrid Encoding Scheme [13]

No. of 1 (Multiplier)	Position of '1' bit	Category	Operation
1	1^{st} bit	A	Add 0 to multiplicand (M)
1	i^{th} bit	B	Shift M left by i-1 and add 0
2	1^{st} and i^{th} bit	C	Shift M left by i-1 and add M
2	1^{st} and $i+j^{th}$ bit	D	Shift M left by j, add M and shift the result left by i-1
3	$i^{th}=1^{st}$, j^{th} and k^{th} bit	E	Shift M by k-j, add M and shift the result left by j-1,add M.
3	i^{th}, j^{th} and k^{th} bit	F	Shift M by k-j, add M and shift the result left by j-1,add M and shift the result by i.

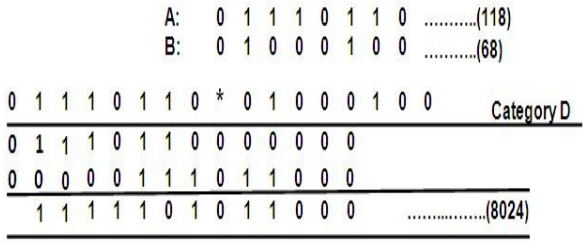


Fig. 4 Example of hybrid multiplier

According to category E, the proposed encoding rule needs one partial product P1 with two additions. The remaining partial products P2 to P8 are zero, so the addition operation in this area can be neglected, which reduces the switching activity and power consumption. This spurious switching activity can be reduced by freezing the adders which perform this unwanted addition. Typical example illustrated in Fig.4 explains such a multiplication for category D.

Proposed ECC Architecture

The proposed ECC processor architecture over GF (2^{256}) is shown in the Fig.5. The main modules of the design are point identifier, clock control, Mongo scalar multiplication and multipliers modules. The performance of the proposed processor is studied for three different multipliers. The point P(x,y) is selected on EC over GF(2^{256}) is done on Point identifier module. Clock control unit used to issue the clock signal to all the modules. Montgo scalar multiplication module performs the Montgomery scalar multiplication algorithm to generate the key for encryption by using point addition and point doubling operations and also performs the coordinate conversion. In multiplier module three different multipliers are employed namely, array, modified booth and hybrid multipliers to perform the point addition and point doubling.

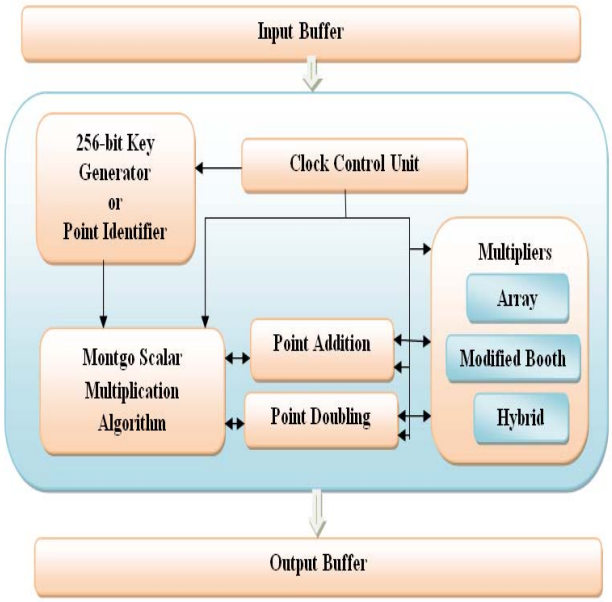


Fig. 5 The Proposed 256-bit ECC architecture

Result and Discussion

The proposed 256-bit binary field ECC architecture is synthesized in Xilinx 9.2i for Spartan3E family XC3S1600E device and simulated in Modelsim 5.7. In order to improve the scalar multiplication in key generation processor, three different multipliers are used and performance analysis shows that ECC architecture with HELP multiplier produces better performance than other two multipliers. The representative RTL view of the ECC design with HELP multiplier is shown in the Fig.6.

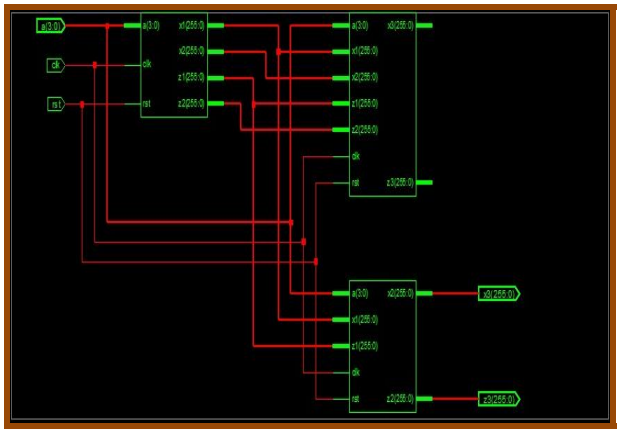
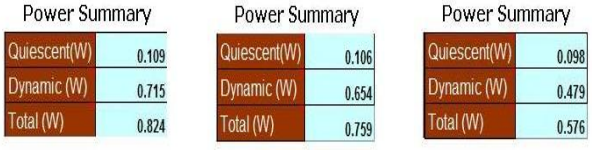


Fig. 6 RTL view of ECC architecture using HELP multiplier



(a) Array (b) Modified Booth (c) HELP

Fig.7 Power summary of ECC architecture with different multipliers

Fig.7 shows the power summary of ECC architecture using different multipliers, which includes both static and dynamic power consumed. The static power is power consumed by leakage current and the dynamic power is defined as amount of power consumed by switching activities of flip flops (FFs). The power consumption of the ECC processor with frequency of 100MHz is 109mW in static and 715mW in dynamic, hence the total summation of

824mW. For the same frequency, the power consumption of the processor is 759mW with the modified Booth multiplier while with the HELP multiplier is 576mW.

Device Utilization Summary		
Logic Utilization	**Used**	**Available**
Number of 4 input LUTs	27,113	29,504
Logic Distribution		
Number of occupied Slices	14,750	14,752
Number of Slices containing only related logic	14,638	14,750
Number of Slices containing unrelated logic	112	14,750
Total Number of 4 input LUTs	27,114	29,504
Number used as logic	27,113	
Number used as a route-thru	1	
Number of bonded IOBs	272	376
Total equivalent gate count for design	164,292	
Additional JTAG gate count for IOBs	13,056	

Fig.8 Device utilization summary -Array multipliers

Device Utilization Summary		
Logic Utilization	**Used**	**Available**
Number of Slice Flip Flops	578	29,504
Number of 4 input LUTs	23,421	29,504
Logic Distribution		
Number of occupied Slices	13,707	14,752
Number of Slices containing only related logic	13,707	13,707
Number of Slices containing unrelated logic	0	13,707
Total Number of 4 input LUTs	23,693	29,504
Number used as logic	23,421	
Number used as a route-thru	272	
Number of bonded IOBs	274	376
IOB Flip Flops	170	
Number of GCLKs	1	24
Total equivalent gate count for design	203,642	
Additional JTAG gate count for IOBs	13,152	

Fig.9 Device utilization summary -Modified Booth multiplier

Device Utilization Summary		
Logic Utilization	**Used**	**Available**
Total Number Slice Registers	159	29,504
Number used as Flip Flops	118	
Number used as Latches	41	
Number of 4 input LUTs	13,737	29,504
Logic Distribution		
Number of occupied Slices	7,753	14,752
Number of Slices containing only related logic	7,753	7,753
Number of Slices containing unrelated logic	0	7,753
Total Number of 4 input LUTs	13,938	29,504
Number used as logic	13,737	
Number used as a route-thru	201	
Number of bonded IOBs	274	376
IOB Flip Flops	170	
Number of GCLKs	2	24
Total equivalent gate count for design	117,213	
Additional JTAG gate count for IOBs	13,152	

Fig.10 Device utilization summary- HELP multiplier

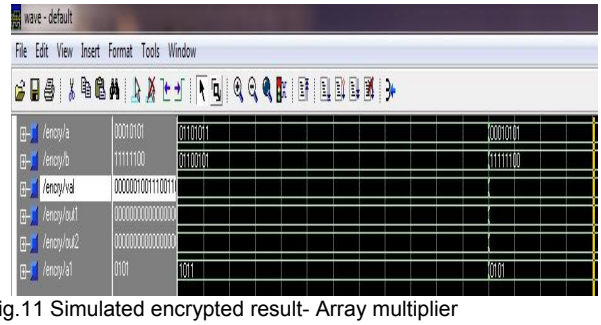


Fig.11 Simulated encrypted result- Array multiplier

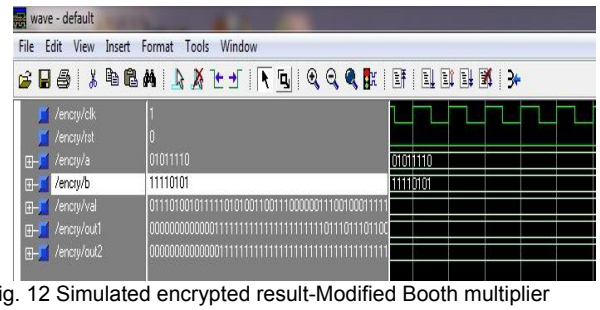


Fig. 12 Simulated encrypted result-Modified Booth multiplier

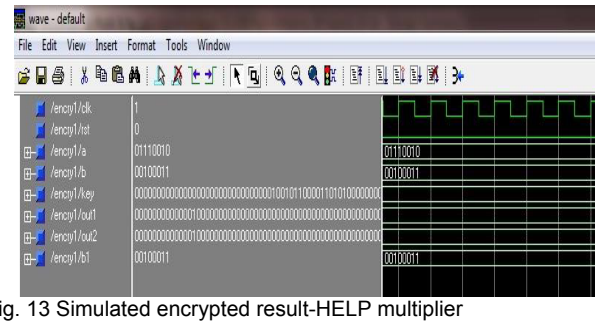


Fig. 13 Simulated encrypted result-HELP multiplier

Synthesis results show that ECC architecture with array multiplier occupies 27114 LUT's, with modified Booth multiplier occupies 23693 LUT's and with hybrid multiplier occupies 13938 LUT's. Among the three multipliers, the hybrid multiplier occupies less area. Fig.8 shows the device utilization summary of ECC architecture with array multiplier. Fig.9 and 10 depict the device utilization summary of ECC architecture with modified Booth and HELP multipliers respectively. The consolidate device utilization summary of 256-bit ECC architecture with three different multipliers are shown in the Fig.11. Fig.12, 13 and 14 demonstrates the simulation result of encrypted cipher text of ECC processor with array, modified Booth and HELP multipliers respectively.

Related Works and Comparison

The Table 2 describes the performance comparison of proposed 256-bit ECC processor with other similar design. Yaxun Gong and Shuguo Li [15] have proposed a high-throughput FPGA implementation of 256 –bit Montgomery Modular multiplier with 30.38MHz. The embedded 18x18 multiplier employed dedicates three cycles to computes a modular multiplication. The algorithm using Karatusba-Ofman multiplication reduced the number of multiplier while the five stage pipeline structure increased the throughput. Zhang et al [16] have designed a 256-bit high speed ECC architecture using the Xilinx XC2V250 (Virtex-II) with 50MHz frequency. The architecture requires a high number of LUTs. The hardware architecture introduced by McIvor et. al. [17] can perform the main prime field arithmetic functions including modular inversion and multiplication. The processor described uses a full-word multiplier which requires much fewer clock cycles than previous methods,

while still maintaining a competitive critical path delay. A scalable unified dual-radix architecture for Montgomery multiplication in GF(P) and GF(2n) has been proposed using CMOS technology [18]. K. Ananyi et. al. has coined a flexible hardware processor for performing computationally expensive modular addition, subtraction, multiplication, and inversion over prime finite fields GF(p) [19]. The performance of the proposed architure is evaluated for three different multipliers at 100MHz. The area utlization with hybrid multipier is most economical.

Table 2 Comparison of ECC Processor architectures

Design	Key Size	Technology	f (MHz)	Area	Arithmetic Unit
[15]	256	Altera Cyclone 3 EP3C 40	30.38	----	81 embedded multipliers
[16]	256	Xilinx XC2V 250(Virtex-II)	50	2594 LUT's	8 modular multiplication and 13 modular addition
[17]	256	XC2V P125-7-ff1696	45.68	11,992 slices	cascading numerous 16 × 16-bit unsigned multipliers
[18]	256	90 nm CMOS	714	156k gates[2]	one radix-2^{64} multiplier
[19]	256	Xilinx Virtex-4 XCV4 FX100	60	20,793 slices	embedded 18 × 18-bit multipliers
Proposed	256	XC3S1600E	100	27,114 LUT's	Array Multiplier
				23,693 LUT's	Modified Booth Multiplier
				13,938 LUT's	Hybrid Multiplier

Conclusion

Nowadays, cryptography plays a major role in protecting the information of technology applications where information security is an important issue. This paper presented an architecture for performing ECC over GF(2^{256}). The proposed 256-bit binary field ECC architecture is analyzed for three different Multipliers. The architecture is implemented using spartan3E family device XC3S1600E using Modelsim 5.7 and Xilinx 9.2i. The synthesis result shows that the power consumption of 256-bit point multiplication and coordinates conversion can be done in 824mW for array multiplier, 759mW for modified Booth multiplier and 576mW for hybrid multiplier with frequency of 100 MHz. The HELP cuddled architecture utilizes 49 percent and 41 percent lesser area respectively than array and modified Booth multipliers. As the major computational task is on multilication, the performnace of any acrhitecure can be ehnaced by triumph multipliers.

REFERENCES

[1]. N. Koblitz, "Elliptic Curve Cryptosystems", *Mathematical on Computer*, vol.48, no.177, pp.203-209, 1987.

[2]. V.S. Miller, "Use of Elliptic Curves in Cryptography", *Proceeding of the International Conference on Advances in Cryptology (CRYPTO '85)*, Springer-Verlag, LNCS 218, pp.417–426, 1986.

[3]. R. Rivest, A. Shamir and L. Adleman, "A Method for Obtaining Digital Signatures and Public Key Cryptosystems", *Communication on ACM,* vol.21, no.2, pp.120-126, 1978.

[4]. A. Menezes, "Elliptic Curve Public Key Cryptosystems", 1st Edition, Kluwer Academic Publishers, ISBN-13: 978-0792393689, 1993.

[5]. D. Hankerson, A. Menezes, and S. Vanstone, "Guide to Elliptic Curve Cryptography", 1st Edition, Springer, 2004.

[6]. Yinan Kong, "Optimizing the Improved Barrett Modular Multipliers for Public-Key Cryptography", *Proceeding of IEEE Conference on Computational Intelligence and Software Engineering (CISE'10)*, pp.1-4, 2010.

[7]. Marisa W. Paryasto, Budi Rahardjo, Fajar Yuliawan, Intan Muchtadi-Alamsyah, Kuspriyanto, " Composite Field Multiplier based on Look-Up Table for Elliptic Curve Cryptography Implementation", *Proceeding of the IEEE International Conference on Electrical Engineering and Informatics*, Bandung, Indonesia , pp.1-4, July 2011,

[8]. Mark Hamilton, William P. Marnane , Arnaud Tisserand, "A Comparison on FPGA of Modular Multipliers Suitable for Elliptic Curve Cryptography over GF(p) for Specific p Values", *Proceeding of the IEEE 21st International Conference on Field Programmable Logic and Applications*, pp.273-276, 2011.

[9]. Justin Hensley, Anselmo Lastra and Montek Singh, " An Area- and Energy-Efficient Asynchronous Booth Multiplier for Mobile Devices", *Proceedings of the IEEE International Conference on Computer Design*, pp.18-25, 2004.

[10]. Sandeep Kumar, Thomas Wollinger, and Christof Paar, " Optimum Digit Serial GF(2^m) Multipliers for Curve-Based Cryptography", *IEEE Transactions on Computers*, vol.55, no.10, pp.1306-1311, October 2006.

[11]. Al-Somani T.F. and Houssain H. , "Implementation of GF(2^m) Elliptic Curve Crypto- processor on a Nano FPGA", *Proceeding of the IEEE International Conference on Internet Technology and Secured Transactions (ICITST)*, Abu Dhabi, United Arab Emirates, pp. 7-12, 2011.

[12]. IEEE 1363, Standard Specifications for Public key Cryptography, 2000.

[13]. S.Saravanan and M.Madheswaran, "Design and Analysis of a Spurious Switching Suppression Technique Equipped Low Power Multiplier with Hybrid Encoding Scheme", *International Journal of Computer Science and Information Security*, vol.6, no.3, pp.73-78, 2009.

[14]. S.Saravanan and M.Madheswaran, "Design of Hybrid Encoded Booth Multiplier with Reduced Switching Activity Technique and Low Power 0.13µm Adder for DSP Block in Wireless Sensor Node", *Proceeding on IEEE Conference on Wireless Communication and Sensor Computing*, pp.1-6, 2010.

[15]. Yaxun Gong and Shuguo Li, "High-Throughput FPGA Implementation of 256-bit Montgomery Modular Multiplier", *Proceeding of the IEEE International Conference on Education Technology and Computer Science (ETCS)*, vol.3, p. 173-176, 2010.

[16]. Zhang Jiahong, Xiong Tinggang, and Fang Xiangyan, "A Fast Hardware Implementation of Elliptic Curve Cryptography", *Proceeding of the IEEE International Conference on Information Science and Engineering (ICISE)*, pp. 1519–1522, 2009.

[17]. McIvor, McLoone, and McCanny, " Hardware Elliptic Curve Cryptographic Processor Over GF(p)", *IEEE Transaction on Circuits and Systems I: Regular Papers*, vol.53, no.9, pp.1946–1957, 2006.

[18]. K. Tanimura, R. Nara, and S. Kohara, " Scalable unified dual-radix architecture for Montgomery multiplication in GF(P) and GF(2n)", *Proceeding of the IEEE International Conference on Design Automation Conference*, pp. 697–702, 2008.

[19]. K. Ananyi, H. Alrimeih and D. Rakhmatov, " Flexible Hardware Processor for Elliptic Curve Cryptography Over NIST Prime Fields", *IEEE Transaction on Very Large Scale Integration (VLSI) Systems,* vol.17, no.8, pp. 1099 - 1112, 2009.

***Authors**: A. Jagan, AP(Sr.G), Department of Computer Science and Engineering, Surya Group of Institutions, Villupuram, TamilNadu, India, E-mail: jagan.aa@gmail.com; Dr. V. Nagarajan, Professor, Department of ECE, Adhiparasakthi Engineering College, Melmaruvathur, TamilNadu, India. E-mail: nagarajanece31@rediffmail.com.*