Middle-East Journal of Scientific Research 25 (4): 684-689, 2017
ISSN 1990-9233

DOI: 10.5829/idosi.mejsr.2017.684.689

Design of High Speed Architecture of ECSM

[1]S. Jayashree, [1]E. Shanthini and [2]N. Suma

[1]Department of ECE, SNS College of Technology, Coimbatore - 35, India
[2]Department of ECE, SNS College of Engineering, Coimbatore -641107, India

Abstract: Multiplication is a basic operation. An elliptic curve is a smooth, projective algebraic curve on which there is a specified point. Scalar multiplication is performed by repeated point addition and doubling which consists of FF arithmetics.Elliptic curve scalar multiplication is the operation of successively adding a point along an elliptic curve. Binary fields and prime fields are used. Multiplication is performed by polynomial multiplication and reduction modulo. The architecture consists of FF MAC, FF SQUARER,FSM,Control Block and Register Bank. To find the optimal number of pipeline stages, scheduling schemes are implemented and placement registers is analyzed. Design of a four stage pipelined ECSM using Montgomery ladder algorithm using Virtex-4 and Virtex-5 is an existing system. The proposed system is to implement a three-stage pipelined ECSM with area reduction, high speed and to improve working frequency. Coding is implemented using Verilog language and the simulation is done using Xilinx ISE 14.5.

Key words: Elliptic curve cryptography(ECC) · Elliptic curve scalar multiplication (ECSM) · Karatsuba-Ofman multiplier (KOM) · Gallois field (GF) · Karatsuba Ofman algorithm (KOA)

INTRODUCTION

The Public-key Cryptographic algorithm is Elliptic Curve Cryptography.ECC can offer the similar secure with a much smaller key length [1]. The performance of ECC cryptosystem controls the key operation of Elliptic Curve Scalar Multiplication. Working Frequency and number of clock cycles are reduced using Pipelining and Parallelism. Many designs were used word serial or digit serial multipliers for considerations of tradeoff between area and speed to carry out ECSM. It usually require a large number of clock cycles for Scalar Multiplication. Elliptic curves are explained over FFs.Finite fields and Prime fields are mutually used. FF multiplication is complex than polynomial basis and Normal Basis has lower FF square. The worth of Public key cryptography is to have an advance exchange of a secret key over not secure channels of a secure communications. The most important operation is scalar multiplication in a curve-based primitive. Coding theory and cryptography had been used in Finite field arithmetic's. Public key cryptosystems are built over finite fields of higher order hence encryption and decryption running time is powered by multiplication and division. To optimize the speed, extensive parallelization and pipelining ECC designs are produced by Design generator. Wiring patterns and data widths are produced automatically. A parametric model was developed for appraisal of the number of clock cycles to make easier the performance characterization for generic ECC architecture [2]. The common feature of an intransitive is a design generator which produces an altered that fulfills the specific-user defined requirements for different uses. Optimal Normal Basis and Polynomial Basis are uniquely used for arranging binary fields.Additonal field multipliers reduce the speed gains but proceed in an area penalty. There is a tradeoff between efficiency and area. The time to perform multiplication is much more complex than shift operation inversion algorithm for the number of clock cycles can be approximated. Although point addition is not as point multiplication, some security protocols are required. The algorithm used to compute the operation is simple.ECC is performed over one of two underlying Galois fields: prime order fields or characteristic two fields. Both fields provide the same level of security. Addition and subtraction in the Galois field are performed by modulo-2 addition, a bit XOR operation.

Corresponding Author: S. Jayashree, Department of Ece, Sns College of Technology, Coimbatore-35, India.

Finite Field: Scalar multiplication is performed by repeated point addition and doubling, which essentially fully satisfied on a series of FF arithmetics, such as multiplication, square addition and inversion [3]. It involve inversion for point addition and doubling in affine coordinate each time. It is to represent the curve points in projective coordinate because inversion is the complex and time consuming operation in FF.In this way, only one inversion is required for the entire scalar multiplication at the cost of more other FF operations[4]. Elliptic curves are explained over FFs.Binary fields and Prime fields are commonly used. They both can provide the same level of security [4]. Polynomial basis and normal basis are most commonly used because elements have many basic representations in GF.FF square is cheaper in Normal basis and FF Multiplication is complex than polynomial basis.

Modified Montgomery Ladder Scalar Multiplication: The Montgomery ladder algorithm consists of three stages:1)initialization: conversing from affine coordinate to LD projective coordinate;2)the main loop: performing point addition and doubling in LD projective coordinate; and 3) postprocess:recovering the y-coordinate and conversing from LD projective coordinate back to affine coordinate. One FF MAC and one FF squarer are used to achieve high performance ECSM.The FF MAC combines addition with the reduction in FF multiplication. This brings the merit that extra clock cycle is not needed for addition, it would not worse the critical path delay. The multiplier can be implemented in serial, parallel or digit-serial way. The serial and digit-serial multipliers consume smaller area, but need more clock cycles [5]. We use the parallel multiplier for high speed consideration, takes only one clock cycle to complete one multiplication.

Latency Reduction in Point Multiplication: The efficiency of point multiplication, Q ¼ k.P;attached on finding the minimum number of steps to reach kP from a given point multiplication on binary Edwards and infer Hessian curves is efficient when one uses w-coordinate differential addition and doubling formulas and Montgomery's ladder. Combine point addition and Doubling are needed to perform every bit of scalar k. The optimum number of multipliers is to achieve high speed calculations of point multiplication.

Data Dependence analysis of ECSM: The modified Montgomery ladder scalar multiplication entirely takes m (6M+5S+3A)+(11M+5A+I) operations, where M,S,A,I denote multiplication, square, addition and inversion in GF(2m) respectively and m is the dimension of the binary field GF(2m). The original Montgomery ladder scalar multiplication requires (m-1) (6M+5S+3A)+ (10M+ 7A+ 3S+I) operations. The increased operations are due to the merged initialization and an altered post process for better sharing the data path with the main loop [6]. As square and addition are much cheaper than multiplication and inversion occurs only once, we can see that optimizing operations in the main loop, the FF multiplication, is the key to accomplish high-performance ECSM.Each iteration in the main loop performs point addition and point doubling, which take 6M+5S+3A together. The critical path lies in calculating the X-coordinate of point addition, which takes 2M+1S+2A. This uses only one FF MAC and one FF squarer to achieve high performance ECSM. The FF MAC joins addition with the reduction in FF multiplication [7]. This brings merits that extra clock cycle is not needed for addition and it would not worse the critical- path delay. The multiplier can be implemented in serial, parallel, or digit-serial way. The serial and digit-serial multipliers consume less area, but more clock cycles are required. Parallel Multipliers are used for high speed consideration takes only one clock cycle to finish one multiplication.

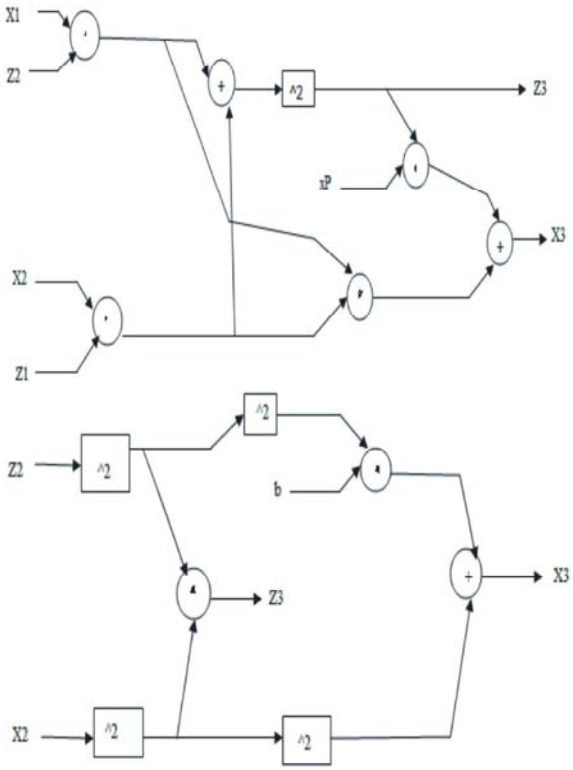

Fig. 1: Data dependence graph of a)point addition and b)point doubling in the Montgomery Ladder Algorithm

Existing System: Six clock cycles using one bit-parallel FF MAC needs six multiplication iteration [10]. By inserting the registers into the combinational logic working frequency of FF MAC can be improve using the pipelining technique [8]. The FF MAC is to compute MR=A.B+C.Addition had been performed with the final reduction which is provided at the last clock cycle. In the event of an n-stage pipelined FF MAC, we input data A and B at the first clock cycle and C at the nth cycle, the result MR can be obtained at the (n+1)th clock cycle. The FF squarer is to calculate $SR=S^2$, which has a small procrastination of only two or three XOR gates for a trinomial or pentanomial [9].The FF squarer is not pipelined. The result SR can be obtained after one clock is inputting data S.

Table 1: Scheduling Scheme Using a Two Stage Pipeined FF MAC

clk	FF MAC(A.B+C)	FF SQR(S^2)
1	$X_2.Z_1$	Z_2^2 (swap*)
2	$X_1.Z_2$	X_2^2 (swap*)
3	$Z2 \leftarrow X_2^2.Z_2^2$	Z_2^4
4	$X_1Z_2.\ X_2Z_1$	X_2^4
5	$X_2 \leftarrow b.Z_2^4+X_2^4$	$Z_1 \leftarrow (X_1Z_2+X_2Z_1)^2$
6	$X_1 \leftarrow x_P.Z_1+X_1Z_2X_2Z_1$	

If swap=1, perform $Z1^2$ instead of $Z2^2$ and $X1^2$ instead of $X2^2$.

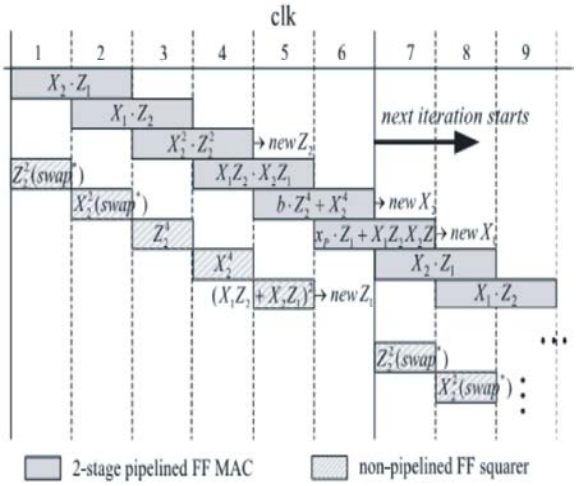

Fig. 2: Timing diagram of the scheduling scheme using a two-stage pipelined FF MAC

Table 2: Scheduling Scheme Using a Three Stage Pipeined FF MAC

clk	FF MAC(A.B+C)	FF SQR(S^2)
1	$X_2.Z_1$	Z_2^2(swap*)
2	$X_1.Z_2$	X_2^2(swap)
3	$Z_2\ X_2^2.Z_2^2$	Z_2^4
4	$X_2\ b.Z_2^4+X_2^4$	X_2^4
5	$X_1Z_2.X_2Z_1$	$Z_1\ (X_1Z_2+X_2Z_1)^2$
6	Idle	

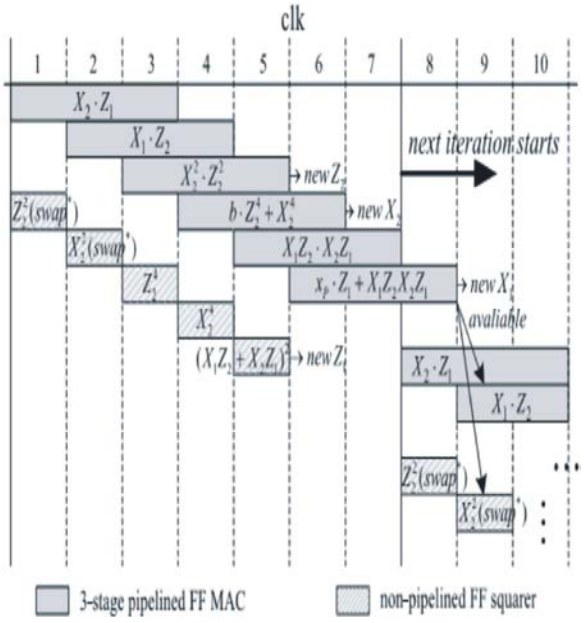

Fig. 3: Iming diagram of the scheduling scheme using a three-stage pipelined FF MAC

By using a three-stage pipelined FF MAC, the main loop takes seven clock cycles for each iteration [11]. The values of new Z2,Z1,X2 and X1 are available at the sixth, seventh and ninth clock cycles,respectively.The ninth clock cycle is the second clock cycle for the next iteration. In case of an n-stage (n>2) pipelined FF MAC, the main loop of ECSM needs (2n+1) clock cycles for each iteration.

The most initiating stage algorithm of scalar multiplication is the Montgomery Ladder algorithm over GF [12]. To implement an ECSM pipelined bit parallel FF MAC based on KOM along with one FF Squarer is used. There are only two extra primitives an addition and a 4:1 multiplexer (MUX) added to the critical path separately from the FF MAC. To find the best tradeoff between the working frequency and the required number of clock cycles more than one pipeline stages are applied to the data path in ECSM [13].

Proposed System: The post process stage of ECSM requires careful consideration. While this stage is not the crucial part of ECSM, to share the data path with the main loop is the optimization goal earlier than to reduce the required number of clock cycles. The high- performance architecture based on an improved Montgomery ladder scalar multiplication algorithm was proposed after proper scheduling of ECSM.The proposed ECSM architecture consists of one-bit parallel FF MAC, one FF square, a

register bank, a finite-state machine and a 6*18 control ROM [14]. The FF MAC is implemented using the Karatsuba-Ofman algorithm and is well pipelined. The n-stage pipelined FF MAC takes n clock cycles to finish one multiplication. The FF squarer is not pipelined and one clock cycle is needed to finish one square [15]. The inputs to FF MAC are A, B and C and the input to FF squarer is S are all registered. The four registers are T1, T2, T3 and T4 used in the data path for data caching. Each register has a MUX before it. To switch between different operations in ECSM, the control signals of MUXs are given at each clock cycle. The inputs are fed to MUXs of registers are allocated with that each MUX contains at most four branches [16]. The input delay for registers is only the delay of a 4:1 MUX.

The control signals are different for each iteration of the main loop at every clock cycle and the post process stage. A heavy state machine provides all the control signals in sequence. A 6*18 control ROM is to store the control signals for MUXs(16 bits) and the swapping selection (2 bits).A small state machine is used for conditional branching and jumping and it provides the 6-bit address to the control ROM.The control ROM can be realized using Block RAMs for the implementation of FPGA. The terms X1, X2, Z1and Z2 are the intermediate results of the FF MAC or FF Squarer. The critical path of the three-stage pipelined architecture consists of a pipelined FF MAC, an addition (XOR) and 4:1 MUX.Control signals are stored in the control ROM are different [17]. But the critical path delay remains unchanged.

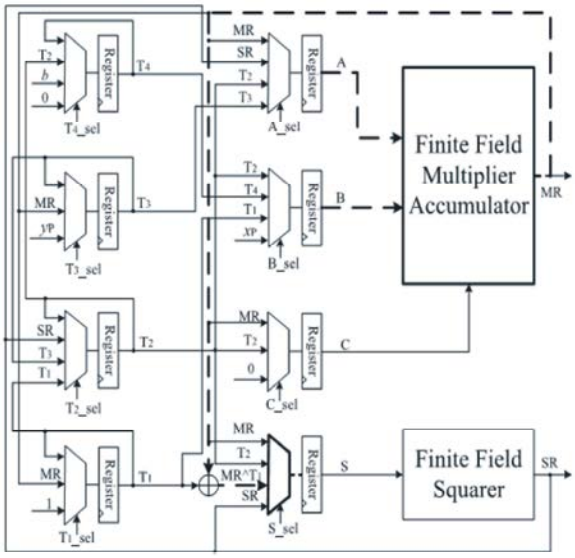

Fig. 4: Data path of ECSM using a three-stage pipelined FF MAC

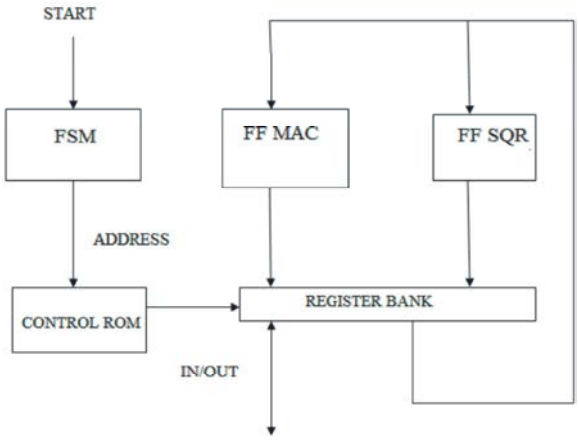

Fig. 5: Proposed architecture of ECSM

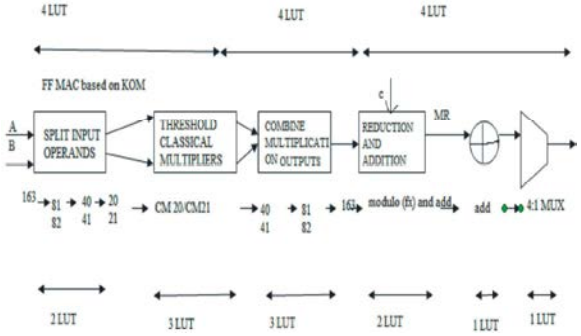

Fig. 6: Critical Path of a three-stage pipelined FF MAC

Delay Estimation and Pipeline Analysis: The FF MAC based on KOM consists of four stage:1) splitting the input operands to produce threshold operands;2) performing threshold Cms;3) recursively aligning the outputs to combine higher-level multiplications; and 4)doing the modular reduction and addition [18]. The critical path in the proposed architecture contains an addition(XOR) and a 4:1 MUX,apart from the FF MAC [17]. The overlapped operands (ah+al) and (bh+bl) requires addition and the r-step splitting generates r the longest chain of 2 input bits.Thus,the splitting stage has an LUT delay of Dsp=log kr2 for k-input LUT FPGAs.Classical multipliers of threshold size t have an LUT delay of DCM=log2t. The aligning stage has r steps and each step has one LUT delay for k=4 [18]. The delay of this stage is Dal k=r.If there are t variables in the longest chain,with regard to the merged reduction and addition, then the delay is Dmod=logt.For trinomials and pentanomials, this stage has a delay of one or two LUTs 4 for given k=6.Therefore,the entire delay of the FF MAC based on the KOM is given by k.For a 2s:1 MUX, there are selection lines s,then the output of a 2s:1 MUX is a

function of 2s+s variables [19]. The delay for a 2s input MUX is DMUX=logk (2s+s).Thus, a 4:1 MUX has a delay of one LUT for k=6 and two LUTs for k=4.The pipeline is achieved by inserting registers into the combinational logic. The obtained circuit will have a minimum delay,if the critical path is divided equally. In case of k=6, there are 12 LUTs in the critical path of ECSM over GF(2^m). For a two-stage pipelined, it should be divided in the ratio 6:6,which indicates that the registers should be put after the combined KOM40/KOM41.For a three-stage pipelined, the best division lies in the ratio 4:4:4.Regisers are put inside the threshold CM20/CM21 for the first stage and after the combined KOM for the second stage. The best division for a four-stage pipeline is of the ratio3:3:3:3.However, for pipeline stages more than three, the frequency improvement is not obvious compared with the linearly increasing clock cycles.

Result Comparison and Discussion: In order to compare with other ECSM designs on the same device, we also implement the proposed architecture on Xilinx XC4VLX200. The fastest reported results in GF(2^m) are 7.7 µs for Virtex-4 and 5.5 µs for Virtex-5.Our proposed three-stage pipelined ECSM achieves 6.1 µs(26.2% faster) on Virtex-4 and 4.6 µs (19.6% faster) on Virtex-5,while the occupied slices are much less (only 35.3% and 49.4%).

The KOM was three-stage pipelined, a quad circuit was used to speed up inversion. The required clock cycles were reduced to 1429(1404) due to data forwarding and the Powerblock.The two FF squares, two additions(XOR),three 4:1 MUXs and one 2:1 MUX into the critical path which greatly increased the critical path delay, apart from the FF Multiplier.

In our architecture, the FF MAC is used instead of FF multiplier. Most additions are absorbed into the FF MAC with no extra logic and clock cycle. The data path of our architecture is well designed, so that only one addition XOR and one 4:1 MUX are added to the critical path of the FF MAC. The total critical path delay of our architecture is 14 LUTs for k=4 and 12 LUTs for k=6. Our designs cost relatively less clock cycles and achieve higher clock frequency, by means of well scheduling and fair pipelining. The control signals are stored in Block RAMs on FPGA, a bulky state machine is saved, which also contributes to higher frequency and smaller area. Area reduction debt much to the use of the KOM. ECSM is 47.5%faster than and 59% faster than on Virtex-4 and 87.0% faster than on Virtex-5 and double the speed on Virtex-5 with even less resources.

Simulation Result:

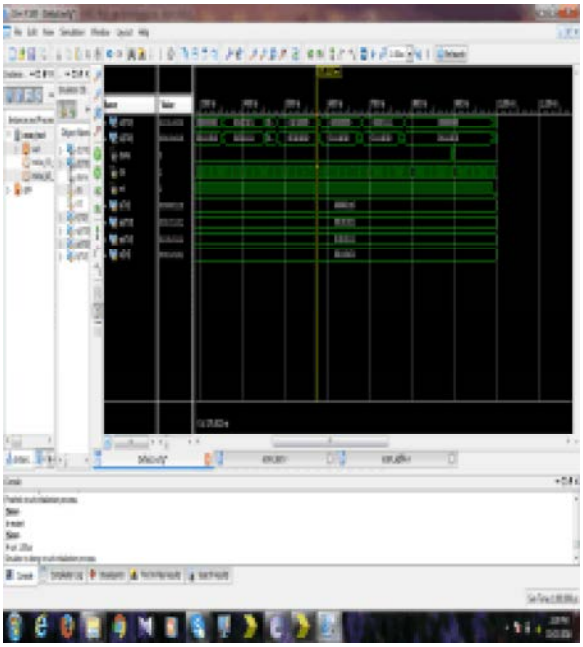

Fig. 7: Simulation Result of Elliptic Curve Scalar Multiplication

In order to compare with other ECSM designs on the same device, we had implemented the proposed architecture on Xilinx XC4VLX200.The fastest reported results in GF (2^m) are 7.7 µs for Virtex-4 and 5.5 µs for Virtex-5.It had used three FF cores to achieve ILP.The three digit-serial field multipliers and one field divider are used to obtain the best tradeoff between area and speed. The four-stage pipelined ECSM achieves 6.1 µs on Virtex-4 and 4.6 µs on Virtex-5, while the occupied slices are much less. The KOM was three-stage pipelined and a quad circuit was used to speed-up inversion.

Table 1: Comparison table of 4-stage pipelined ECSM

Existing System	GF(2^m)	Devices	Frequency(Mhz)	Occupied Slices	Slices as Luts	Clock Cycles	TIME (Nanosecond)
4-stage pipeline ECSM	2	XC4VLX200	222	7354	13237	1363	2.897ns

Table 2: Comparison table of 3-stage pipelined ECSM

Proposed System	GF(2^m)	Devices	Frequency(mhz)	Occupied Slices	Slices as Luts	Clock Cycles	TIME(Nanosecond)
3-stage pipeline ECSM	3	XC5VLX110-3	294	3041	9470	1363	2.051ns

CONCLUSION

This paper focuses on speeding up ECSM over GF(2^m) on FPGA with the premise of high area utilization efficiency. The proposed architecture mainly consists of a pipelined bit-parallel FF MAC and a non-pipelined FF squarer. Area reduction is achieved using the Karatsuba-Ofman algorithm. Compact scheduling schemes are suggested to reduce the required number of clock cycles for ECSM.The data path in the architecture is designed to achieve a short critical path. Pipeline techniques are applied to improve the working frequency. Thorough analyzes are supported with detailed experimental results, provided to find the architecture with the optimal number of pipeline stages. Compared with other existing designs, the proposed architecture outperforms their results in terms of both speed and area. Thus delay is reduced. Area reduction is also achieved.The future work is to implement an ECSM for 32,64,128 bits in order to reduce the delay, reduction of area, to improve the working frequency, reduction of area and to achieve the short critical path.

REFERENCES

1. Lijuan Li and Shuguo Li, 2016. High-performance pipelined architecture of Elliptic curve scalar multiplication over GF(2^m) on IEEE transactions on very large scale integration (vlsi) systems, 24(4).
2. Wener, E. and M. Hutter, 2011. Exploring the design space of prime field vs. binary field ECC-hardware implementations, in Proc.16th Nordic Conf.Secure IT Syst. Inf. Security Technol. Appl. (NordSec), Tallinn, Estonia, pp: 256-271.
3. Leong P.H.W. and I.K.H. Leung, 2002. A microcoded elliptic curve processor using FPGA technology, IEEE Trans.Very Large Scale Integr. (VLSI) Syst., 10(5): 550-559.
4. Gura N., *et al*., 2003. An end-to-end systems approach to elliptic curve cryptography, in Proc.4th Int.Workshop CHES London, U.K., pp: 349-365.
5. Satoh, A. and K. Takano, 2003. A scalable dual-field elliptic curve cryptographic processor,"IEEE Trans Comput, 52(4): 449-460.
6. Cheung, R.C.C., N.J. Telle, W. Luk and P.Y.K. Cheung, 2005. Customizable elliptic curve cryptosystems, IEEE Trans.Very Large Scale Integr (VLSI) Syst, 13(9): 1048-1059.
7. Sakiyama, K., L. Batina, B. Preneel and I. Verbauwhede, 2006. Superscalar coprocessor for high-speed curve-based cryptography,"in Proc.8th Int.Workshop Cryptography Hardware, Embedded Syst (CHES), pp: 415-429.
8. Schinianakis, D., A. Kakarountas, T. Stouraitis and A. Skayantzos, 2009. Elliptic curve point multiplication in GF(2n) using polynomial residue arithmetic, in Proc.16th IEEE Int.Conf. Electron Circuits. Syst. (ICECS), pp: 980-983.
9. Chelton, W.N. and M. Benaissa, 2008. Fast elliptic curve cryptography on FPGA, IEEE Trans.Very Large Scale Integr (VLSI) Syst., 16(2): 198-205.
10. Jarvinen, K. and J. Skytta, 2008. On parallelization of high-speed processors for elliptic curve cryptography, IEEE Trans.Very Large Scale Integr. (VLSI) Syst., 16(9): 1162-1175.
11. Azarderaksh, R. and A. Reyhani-Masoleh, 2013. High-performance implementation of point multiplication on Koblitz curves, IEEE Trans.Circuits Syst.II, Exp. Briefs, 60(1): 41-45.
12. Jarvinen, K., 2011. Optimized FPGA-based elliptic curve cryptography processor for high-speed applications, Integr VLSI., 44(4): 270-279.
13. Azarderaksh, R. and A. Reyhani-Masoleh, 2015. Parallel and high-speed computations of elliptic curve cryptography using hybrid-double multipliers,"IEEE Trans Parallel Distrib, Syst., 26(6): 1668-1677.
14. Kim, C.H., S. Kwon and C.P. Hong, 2008. FPGA implementation of high performance elliptic curve cryptographic processor over GF (2m), J. Syst. Archit., 54(10): 893-900.
15. Zhang, Y., D. Chen, Y. Choi, L. Chen and S. Ko, 2010. A high performance ECC hardware implementation with instruction-level parallelism over GF(2^m), Microprocess, Microsyst., 34(6): 228-236.
16. Sutter, G.D., J. Deschamps and J.L .Imaria, 2013. Efficient elliptic curve point multiplication using digit-serial binary field operations,"IEEE Trans. Ind. Electron, 60(1): 217-225.
17. Kumar, S., T. Wollinger and C. Paar, 2008. Optimum digit serial GF (2^m) multipliers for curve-based cryptography, IEEE Trans. Comput., 57(11): 144-1453.
18. Shu, C., K. Gai and T. El-Ghazawi, 2005. Low latency elliptic curve cryptography accelerators for NIST curves over binary fields, in Proc.IEEE Int.Conf.Field-Program. Technol., pp: 309-310.
19. Ansari, B. and M.A. Hasan, 2008. High-performance architecture of elliptic curve scalar multiplication, IEEE Trans. Comput., 57(11): 1443-1453.
20. Suma, N. and T. Purusothaman, 2014. Secure Authentication Based Multipath Routing Protocol for WSNs', Journal of theoretical and Applied Information technology, Vol.59 (1) January 2014, pp.222-231.