HIGH PERFORMANCE ELLIPTIC CURVE GF(2^k) CRYPTOPROCESSOR ARCHITECTURE FOR MULTIMEDIA*

Adnan Abdul-Aziz Gutub and Mohammad K. Ibrahim

Department of Computer Engineering
King Fahd University of Petroleum and Minerals
Dhahran 31261, SAUDI ARABIA
Email: {gutub,ibrahimm}@ccse.kfupm.edu.sa

ABSTRACT

A high performance GF(2^k) Elliptic Curve Crypto processor architecture suitable for multimedia security is proposed. To meet the high data rates of multimedia, the new architecture exploits parallelism within Elliptic Curve point operations after using projective coordinates. In this paper, the decision on which projective coordinate to use is based on its efficiency with regard to its parallel implementation. Two different projective coordinates are compared here. This parallelism is exploited in the new architecture by using three separate bit-level pipelined digit serial-parallel multipliers that can operate in parallel. It is worth pointing that such multipliers are ideally suited for the repetitive multiplications inherent in Elliptic Curve cryptography. It is believed that such high performance architectures are needed for high end servers that need to support the security of many multimedia streams at the same time.

1. INTRODUCTION

Security of multimedia is becoming critical in many internet based applications including personal communications, e-commerce, entertainment, etc. Encryption is seen as an ideal way of providing data security in such applications. The long word length needed in cryptosystems, however, posses a significant challenge for real time multimedia encryption. For example, RSA one of the most popular public key methods known requires a key size of 1-2K bits [2].

Elliptic Curve Cryptosystem (ECC), which was originally proposed by Niel Koblitz and Victor Miller in 1985 [1-9], is seen as a serious alternative to RSA with a much shorter word length. ECC with a key size of 128-256 bits is shown to offer equal security to that of RSA with key size of 1-2Kbits [2]. To date, no significant breakthroughs have been made in determining weaknesses in the ECC algorithm, which is based on the discrete logarithm problem over points on an elliptic curve. The fact that the problem appears so difficult to crack means that key sizes can be reduced in size considerably, even exponentially [2,5,8]. This advantage of ECC is being recognized recently where it is being incorporated in many standards. In 1999, the Elliptic Curve Digital Signature Algorithm was adopted by ANSI, and it is now included in the ISO/IEC 15946 draft standards. Other standards that include Elliptic Curves as part of their specifications are the IEEE P1363 (www.stdsbbs.ieee.org), Internet Engineering Task Force (www.ietf.cnri.reston.va.us), and the ATM Forum.

Software implementations of ECC are too slow to meet the high data rate demands of multimedia. Hardware solutions offer a more realistic alternative and are in fact more secure.

Inversion operations, which are needed in point addition over Elliptic Curves are the most expensive operation over Finite Fields [9,19]. The approach adopted in the literature is to represent Elliptic Curve points in projective coordinates in order to replace the inversion operations with repetitive multiplications [1,4,7,9,15].

Recently, several ECC processors have been proposed in the literature [4,7,15] based on projective coordinate representation. There are many projective coordinates systems to choose from. In exiting architecture, the selection of the projective coordinate is based on the number of arithmetic operations, mainly multiplications. This is to be expected due to the sequential nature of these architectures where a single multiplier is used. For high performance multimedia servers such sequential architectures are too slow to meet the demand of increasing number of customers. For such servers, high-speed crypto processors are becoming crucial.

The approach adopted here is to increase performance by exploiting parallelism within the Elliptic curve point operations in projective coordinates. This advantage of projective coordinate computation has not been exploited before. In this paper, two projective coordinate systems are compared with regard to their parallel implementation.

This parallelism is exploited in the new architecture by incorporating three multipliers that work in parallel. It should be pointed out that area is not a limiting factor in current technology. We also propose to use bit level pipelined GF(2^k) digit serial multipliers reported in [16,17] since they have a better performance than both unpipelined and pipelined parallel multipliers for algorithms with repeated multiplications such as those found in ECC. It is worth noting that using pipelined parallel multipliers is not efficient for ECC where the multiplication of an iteration cannot commence before the multiplication operation of the previous iteration is finished. Comparisons illustrate the improved performance of the proposed architectures with respect to both time and cost (AT^2),

* ***Acknowledgment****: The authors would like to thank King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia, for its support of this research project.*

which makes the proposed design very attractive for multimedia security.

2. ECC POINT OPERATION ALGORITHM

There are many ways to apply elliptic curves for cryptographic purposes. All the security relies on the difficulty of obtaining the integer n knowing the two elliptic curve points: nP and P. It will be assumed that the reader is familiar with the arithmetic over elliptic curve and how it is used in cryptography. For a good review the reader is referred to [9].

ECC is based on computing nP from P [1-9]. One ECC algorithm used for calculating nP from P is the binary method, since it is known to be efficient for hardware implementation [2,5,7,9,10]. This binary method algorithm is shown below:

Define k: number of bits in n and n_i: the i^{th} bit of n
Input: P (a point on the elliptic curve).
Output: $Q = nP$ (another point on the elliptic curve).

1. if $n_{k-1} = 1$, then $Q:=P$ else $Q:=0$;
2. for $i = k-2$ down to 0;
3. { $Q := Q+Q$;
4. if $n_i = 1$ then $Q:= Q+P$; }
5. return Q;

Basically, the binary method algorithm scans the binary bits of n and doubles the point Q k-times. Whenever, a particular bit of n is found to be one, an extra operation is needed. This extra operation is $Q+P$. Other algorithms to reduce the number of additions also exist, such as NAF. All these algorithms, including the binary algorithm, require point doubling and point additions as their basic operation in each iteration. As mentioned earlier, the point operations over elliptic curve requires inversion [9]. As in the crypto processor in [6,15], inversion is eliminated using projective coordinates as discussed in the next section.

3. PROJECTIVE COORDINATES IN GF(2^k)

The projective coordinates are to eliminate the need for performing inversion. For elliptic curve defined over $GF(2^k)$, two different forms of formulas are found [9,18] for point addition and doubling. One form projects $(x,y)=(X/Z^2,Y/Z^3)$ [9], while the second projects $(x,y)=(X/Z,Y/Z)$ [18].

The two forms procedures for projective point addition of $P+Q$ (two elliptic curve points) is shown below:

$P=(X_1,Y_1,Z_1);Q=(X_2,Y_2,Z_2);P+Q=(X_3,Y_3,Z_3);$ *where* $P \neq \pm Q$

$(x,y)=(X/Z^2,Y/Z^3) \rightarrow (X,Y,Z)$		$(x,y)=(X/Z,Y/Z) \rightarrow (X,Y,Z)$	
$A = X_1Z_2^2$	2M	$A = X_1Z_2$	1M
$B = X_2Z_1^2$	2M	$B = X_2Z_1$	1M
$C = A+B$		$C = A+B$	
$D = Y_1Z_2^3$	2M	$D = Y_1Z_2$	1M
$E = Y_2Z_1^3$	2M	$E = Y_2Z_1$	1M
$F = D+E$		$F = D+E$	
$G = Z_1C$	1M	$G= C+F$	
$H = FX_2+GY_2$	2M	$H= Z_1Z_2$	1M
$Z_3 = GZ_2$	1M	$I=C^3+aHC^2+HFG$	6M
$I =F+Z_3$		$X_3 = CI$	1M
$X_3= aZ_3^2+IF+C^3$	5M	$Z_3 = HC^3$	1M
$Y_3= IX_3+HG^2$	3M	$Y_3=GI+C^2[FX_1+CY_1]$	5M
	---		---
	20 M		***17M***

Similarly, the two forms of formulas for projective point doubling are shown below:

$P = (X_1,Y_1,Z_1); P+P = (X_3,Y_3,Z_3)$

$(x,y)=(X/Z^2, Y/Z^3) \rightarrow (X,Y,Z)$		$(x, y) = (X/Z, Y/Z) \rightarrow (X,Y,Z)$	
$Z_3=X_1Z_1^2$	2M	$A=X_1Z_1$	1M
$A = bZ_1^2$	1M	$B= bZ_1^4+X_1^4$	5M
$B = X_1+A$		$C= AX_1^4$	1M
$X_3 = B^4$	2M	$D=Y_1Z_1$	1M
$C = Z_1Y_1$	1M	$E=X_1^2+D+A$	
$D=Z_3+X_1^2+C$	1M	$Z_3=A^3$	2M
$E = DX_3$	1M	$X_3=AB$	1M
$Y_3 = X_1^4Z_3+E$	2M	$Y_3= C+BE$	1M
	---		---
	10M		***12M***

The squaring calculation over $GF(2^k)$ is assumed very similar to the multiplication computation. They are both denoted as M (multiplication) in the above. Since the number of additions is taken to be, on the average, half the number of bits, it can be clearly seen form the above tables that the projective coordinate $(x,y) = (X/Z^2,Y/Z^3)$ has on the average 20 multiplication iteration, while the projection $(x,y) = (X/Z,Y/Z)$ has on the average 20.5 multiplications. Clearly, the former would be the projection of choice for sequential implementation. However, as will be discussed in section 5, the projection $(x,y) = (X/Z,Y/Z)$ has an advantage for parallel implementation.

4. PROPOSED ARCHITECTURE AND PARALLEL IMPLEMENTATION OF PROJECTIONS

The architecture of the new ECC processor is shown in Figure 1. This architecture can be used to implement ECC based on either of the two projective coordinate forms discussed in section 5. Unlike existing designs, which use a single multiplier, the new architecture has three multipliers to meet the high data rate demands of applications such as multimedia. It can also perform multiply-add operations in one instruction.

To implement the above projective coordinates using the new architecture, the dataflow of the corresponding procedures are shown in Figures 2 and 3. It can be easily seen that these particular dataflow arrangements of these procedures require a maximum of three multipliers at each step. The reader can verify the correctness of these dataflow with the corresponding description in the previous section.

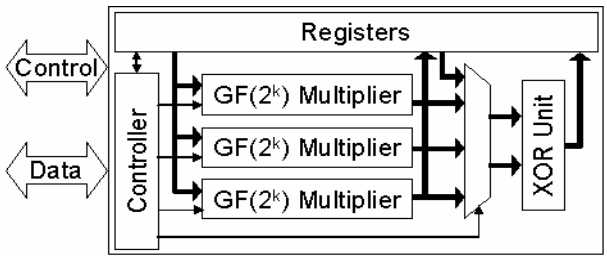
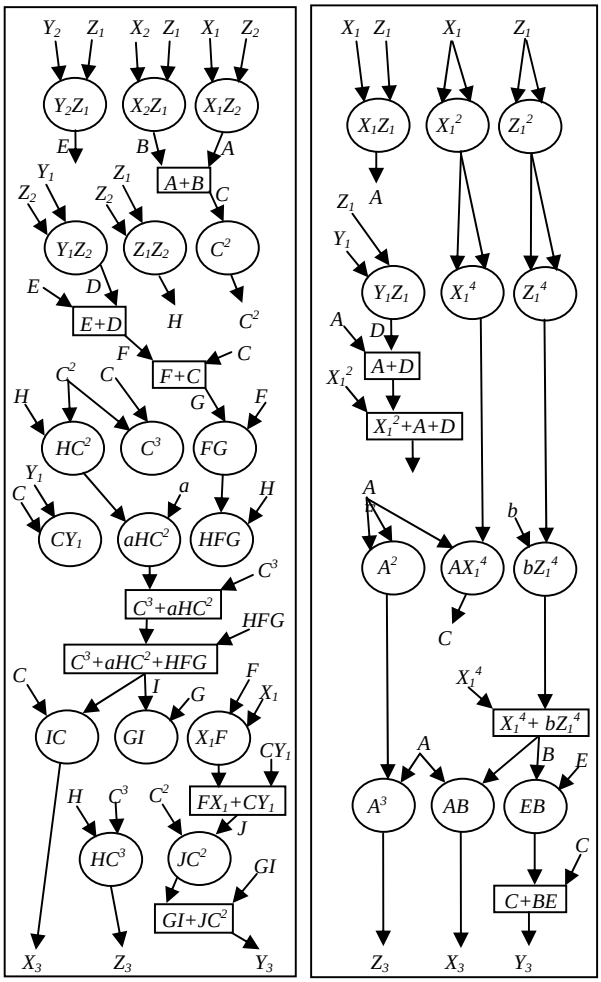


Figure 1. Proposed architecture

In the next section, the comparison of the parallel implementation using the proposed parallel architecture as well as the sequential implementation of both projective coordinates is presented.

5. COMPARISON WITH DIFFERENT DESIGN

Table 1 illustrates a summary of the comparisons of using the proposed parallel architecture, Figure 1, in implementing both projection of *(x,y)*, i.e. *(X/Z,Y/Z)* and *(X/Z^2,Y/Z^3)*, with the corresponding sequential implementations when using a single multiplier (traditional design as proposed in [9]).

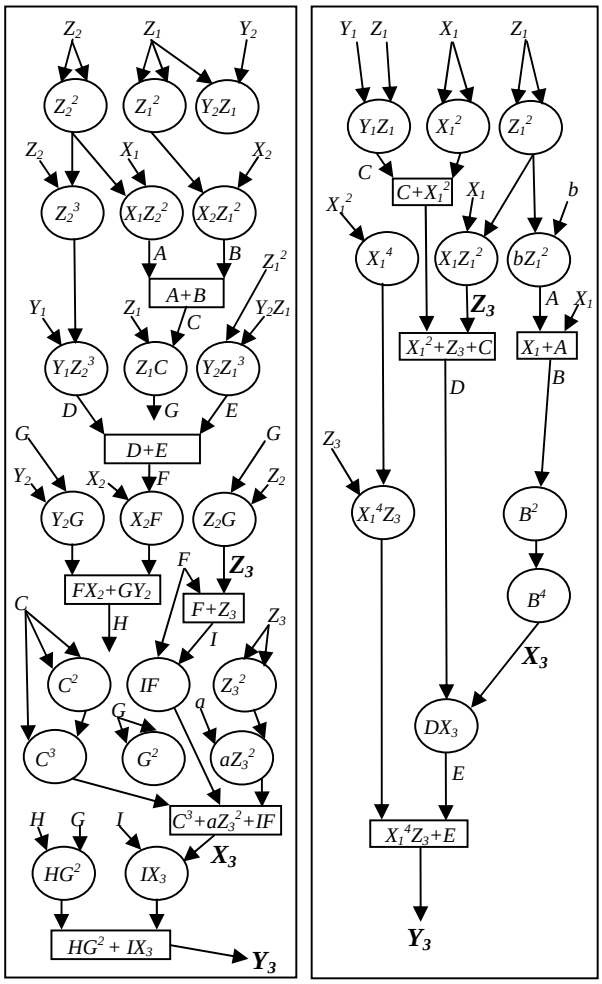


(a) adding two points (b) doubling a point

Figure 2. Data flow graphs for the elliptic curve point operations of projecting *(x,y)* to *(X/Z,Y/Z)*

Comparing the two projective forms, projecting *(x,y)* to *(X/Z^2,Y/Z^3)* requires, on the average, less number of multiplications than projecting into *(X/Z,Y/Z)* as shown in Table 1. Although, the later uses three less multiplication operations in adding two different elliptic points, it uses two more multiplication operations in doubling an elliptic point. It is worth remembering that the number of doubling is usually more than the number of additions. Since, the projection *(X/Z^2,Y/Z^3)* requires less number of multiplications, it is more suitable for sequential implementation, i.e. when using a single multiplier, as has been reported in the literature.

As discussed in the previous section, the new architecture can implement the procedures of both projective coordinate forms. As can be seen from Figure 2, the longest data path of each dataflow diagram (2a and 2b) is effectively 6 GF(2^k) multiplications and of 4 GF(2^k) multiplications, respectively. Here the time of GF(2^k) addition is ignored since it is simple bit-wise XOR-operation. Therefore, for *(x,y)=(X/Z,Y/Z)*, the



(a) adding two points (b) doubling a point

Figure 3. Data flow graphs for the elliptic curve point operations of projecting *(x,y)* to *(X/Z^2,Y/Z^3)*

minimum computation time to perform one elliptic point operation in the calculation of *nP* is, on the average, 7 GF(2^k) multiplications when using three multipliers. When using the projection of (*X/Z^2,Y/Z^3*) as in Figure 3, this time is, on the average, 8.5 GF(2^k) multiplications. Furthermore the utilization of the three multipliers in Figure 2 is much higher than that in Figure 3. As can be seen in Figure 2, all the three multipliers will be used in all of the steps except one. While in Figure 3, there are idle multipliers in four multiplication steps. This clearly indicates that the parallel implementation of projective coordinate *(X/Z,Y/Z)* requires less number of cycles and hence it is faster than projecting *(x,y)* to *(X/Z^2,Y/Z^3)*, and should be the

projection of choice for our parallel architecture. This is contrary to sequential implementation.

As can be seen from Table 1, the time required by our design in projecting *(x,y)* to *(X/Z,Y/Z)* is almost one third the time of the regular sequential implementation in [9] and more than 17% faster than using three parallel multipliers similar to Figure 1 but projecting *(x,y)* to $(X/Z^2,Y/Z^3)$. What is more significant observation from Table 1 is that using the proposed architecture with projections *(x,y)* to *(X/Z,Y/Z)* is not only faster for parallel implementation but it also leads to a better AT^2 (cost) than other alternatives.

To increase the performance even further, we propose to use bit-level pipelined digit serial-parallel multipliers which are detailed in [16,17]. It is significant to point out that these multipliers are in fact faster and use less area than their *un-pipelined* bit-parallel counterparts [15,16]. Furthermore, sub-digit pipelining of digit serial computation leads to a much better performance than the conventional digit serial structures as shown in [16]. Bit-level digit serial computation is more suitable for the elliptic curve crypto algorithm discussed above since the computation of elliptic point doubling, addition and the algorithm of computing multiples of the base point is such that the multiplication of one stage must be completed before starting the multiplication of the subsequent stage. Therefore even if a pipelined bit-parallel multipliers is used, the throughput of such a multiplier can not be exploited since the next multiplication operation can not commence until the multiplication operations in the previous stage has completed. As with regard to the $GF(2^k)$ adder, it is to be implemented in bit parallel fashion since the area is not significant compared to the multiplier and minimizing the addition time will reduce the overall multiply-add cycle time.

Table 1. Comparison between the different designs

Procedure of Projecting *(x,y)* to	Hardware Design	Number of Multipliers (A)	Avg. Number of Cycles for Multiplication (T)	AT^2
$(X/Z^2,Y/Z^3)$	Conventional (Figure 2)	1	20	400
		3	8.5	217
(X/Z,Y/Z)	(Figure 1) Proposed	1	20.5	420
		3	7	147

6. CONCLUSION

A novel $GF(2^k)$ elliptic curve cryptographic processor is proposed in this paper suitable for the high data rates demand of multimedia streams. It has three bit-level pipelined digit serial-parallel multipliers. It circumvents the need for a $GF(2^k)$ inverse module by using projective coordinates to convert inverse operation into consecutive multiplication steps using projective coordinates. It is also shown that projection of *(x,y)* to *(X/Z,Y/Z)* leads to a better parallel implementation than the usually selected projection of *(x,y)* to $(X/Z^2,Y/Z^3)$.

7. REFERENCES

[1] Miyaji A., "Elliptic Curves over F_P Suitable for Cryptosystems", *Advances in cryptology - AUSCRUPT'92*, Australia, December 1992.

[2] Stallings, W. "*Cryptography and Network Security: Principles and Practice*", Second Edition, Prentice Hall Inc., New Jersey, 1999.

[3] Chung, J., Sim, S., and Lee, P., "Fast Implementation of Elliptic Curve Defined over $GF(p^m)$ on CalmRISC with MAC2424 Coprocessor", *Workshop on Cryptographic Hardware and Embedded Systems - CHES 2000*, Massachusetts, August 2000.

[4] Okada,S., Torii,N., Itoh,K., and Takenaka,M., "Implementation of Elliptic Curve Cryptographic Coprocessor over $GF(2^m)$ on an FPGA", *Workshop on Cryptographic Hardware and Embedded Systems - CHES 2000*, Massachusetts, August 2000.

[5] Crutchley, D. A., "*Cryptography And Elliptic Curves*", Master Thesis under Supervision of Prof. Gareth Jones, submitted to the Faculty of Mathematics at University of Southampton, England, May 1999.

[6] Orlando,G., and Paar,C., "A High-Performance Reconfigurable Elliptic Curve Processor for $GF(2^m)$", *Workshop on Cryptographic Hardware and Embedded Systems - CHES 2000*, Massachusetts, August 2000.

[7] Stinson, D. R., "*Cryptography: Theory and Practice*", CRC Press, Boca Raton, Florida, 1995.

[8] Paar, C., Fleischmann, P. and Soria-Rodriguez, P., "Fast Arithmetic for Public-Key Algorithms in Galois Fields with Composite Exponents", *IEEE Transactions on Computers*, Vol. 48, No. 10, October 1999.

[9] Blake, I., Seroussi, G., and Smart, N., "*Elliptic Curves in Cryptography* ", Cambridge University Press: New York, 1999.

[10] Hankerson, D., Hernandez, J., and Menezes, A., "Software Implementation of Elliptic Curve Cryptography Over Binary Fields", *Workshop on Cryptographic Hardware and Embedded Systems -CHES 2000*, Massachusetts, August 2000.

[11] Orton, G.A., Roy, M.P., Scott, P.A., Peppard, L.E., and Tavares, S.E., "VLSI implementation of public-key encryption algorithms", *Advances in Cryptology -- CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 277-301, 11-15 August 1986. Springer-Verlag, 1987.

[12] Scott, Norman R., "*Computer Number Systems and Arithmetic*", Prentice-Hall Inc., New Jersey, 1985.

[13] Tocci, R.J. and Widmer, N.S., "*Digital Systems: Principles and Applications*", 8th Edition, Prentice-Hall Inc., NJ, 2001.

[14] Ercegovac, M. D., Lang, T., and Moreno, J. H., "*Introduction to Digital System*", John Wiley & Sons, Inc., New York, 1999.

[15] Orlando,G., and Paar,C., "A scalable GF(p) elliptic curve processor architecture for programmable hardware", *Cryptographic Hardware and Embedded Systems - CHES 2001*, Paris, France, May 14-15, 2001.

[16] Ibrahim, M.K., Almulhem, A., "Bit-Level Pipelined Digit Serial $GF(2^m)$ Multiplier", *2001 IEEE International Symposium on Circuits and Systems*, Sidney Australia, 2001.

[17] Ibrahim, M.K., Junaid, A.K., Al-Abaji, R. H., Almulhem, A., "Trade-off analysis of a new sign digit serial GF multiplier", *Fifth World Multi-conference on Systemics, Cybernetics and Informatics*. XIV(II):52-56. July 2001, Orlando, 2001.

[18] Ernst M., Klupsch, Hauck, and Huss,"Rapid Prototyping for Hardware Accelerated Elliptic Curve Public-Key Cryptosystems", *The IEEE 12th International Workshop on Rapid System Prototyping*, Monterey, CL, June 25-27, 2001.

[19] Adnan Abdul-Aziz Gutub, A.F. Tenca, E. Savas, and C.K. Koc, "Scalable and unified hardware to compute Montgomery

inverse in GF(p) and GF(2^n)". *Cryptographic Hardware and Embedded Systems - CHES 2002*, August 13-15, 2002.