# Optimum Hardware-Architecture for Modular Divider in GF($2^m$) with Chaos as Arbitrary Source Based on ECC

Azar Hosseini and Abolfazl Falahati

Department of Electrical Engineering (DCCS Lab)
Iran University of Science and Technology, Tehran
ahosseini@elec.iust.ac.ir, afalahati@iust.ac.ir

**Abstract.** The large-scale proliferation of wireless communications both inside and outside the home-office environment has led to an increased demand for effective and cheap encryption schemes. Now a new chaos based signals as arbitrary source and digital signals as main source make digits for Elliptic curve algorithm by 2 parallel-in (with pipelining), 1parallel-out and 1 serial-out to produce encrypted signals. For computing this scheme in application on MC-DS-CDMA transmitter and receiver, new algorithm of division with the least time consumption, is presented.

This algorithm is implemented in modular division over GF ($2^m$) without any considerable increase in hardware gate count. By considering appropriate circuit configuration, the simulation results are also presented.

**Keywords:** ECC, Chaos source, Proposed Divider, FPGA, MC-DS-CDMA.

## 1 Introduction

Finite fields GF ($2^m$ ) have several applications in such areas of communications as error-correcting codes and cryptography. In these applications, computing inverses or divisions in GF ($2^m$ ) is usually required. Since it is not efficient to perform such computations in real time on a general-purpose computer, hardware efficient architectures for inversion or division in GF ($2^m$ ) are highly desirable [2]. Computing inversion or division over GF($2^m$) can be performed in one of the following forms:

1. Little Fermat's theorem, computed by a modular exponentiation: $XY^{2m-2}$ mod p(x) [3];
2. Solution of a system of linear equations over GF( $2^m$) using Gaussian elimination [4,5];
3. The use of the extended Euclid's algorithm over GF( $2^m$) to perform iterative transformations of the GCD(Greatest Common Divisor) [2,6,7].

H. Jahankhani et al. (Eds.): ICGS3/e-Democracy 2011, LNICST 99, pp. 245–252, 2012.

The first and second schemes have area-time product of O($m^3$), but the last scheme has O($m^2$) [7]. So the last scheme is more desired than the first and the second schemes. But in the literature this kind of algorithms is usually considered as very slow [8]. On the other hand, due to the size of the field-m is not being constant it is desired to reduce its effect on the hardware design. In this paper a new and fast algorithm for division over GF ($2^m$ ) based on extended Euclid's algorithm scheme is developed that can perform a finite field division in m-1 clock cycles and its proposed hardware implementation of m. The rest of the paper is devoted to implement this modular in new scheme with chaos circuit as arbitrary source to ECC (Elliptic Curve Cryptosystem).

## 2 New Variant of Euclid's Algorithm for Division in GF($2^m$)

Let $A(x)$ and $B(x)$ be two elements in GF ($2^m$ ), $G(x)$ the primitive polynomial used to generate the field and $P(x)$ the result of the division $A(x)/B(x)$ mod $G(x)$, where:

$$A(X) = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + ... + a_1x + a_0 \tag{1}$$

$$B(X) = b_{m-1}x^{m-1} + b_{m-2}x^{m-2} + ... + b_1x + b_0 \tag{2}$$

$$G(X) = x^m + g_{m-1}x^{m-1} + g_{m-2}x^{m-2} + ... + g_1x + g_0 \tag{3}$$

$$P(X) = p_{m-1}x^{m-1} + p_{m-2}x^{m-2} + ... + p_1x + p_0 \tag{4}$$

Each coefficient of the polynomials is in (0, 1). $P(x)$ is called the inverse of $B(x)$ when $A(x) = 1$ [2].

**A.** *Reclaimed Euclid's Algorithm*
This algorithm needs a total of 2m iterations. This fact within the following is obtained [2]:

```
R = B(x); S = G = G(x); U = A(x); V = T = 0; state = 0; count = 0;
For i = 1 to 2m  do  R = x .R;  T = x .T  mod G; //key operations
  If state = = 0   then   count = count + 1;
    If rm = = 1 then R  <->  S; // rm : the coefficient of x^m in R
       R = R + S;  T= T + U;  T= U;  state = 1;  End
  Else   count = count - 1;
  If  rm = = 1    then    R = R + S;  T = T + U;  End
    If count = = 0   then   V= V + T;  u <->  v;  state = 0;  End
  End
End // V has  P(x) = A(x)/B(x)  mod G(x);  count = 0.
```

R is a polynomial with a degree of at most m, S is a polynomial with degree m, and U, V and T are polynomials with the degrees at most $m - 1$.

**Table 1.** An example of the claimed algorithm for division in GF($2^4$ ) [2]

| i | counter | state | R | S | U | V | T |
|---|---|---|---|---|---|---|---|
| | 0 | 0 | $x^3+x+1$ | $x^4+x+1$ | $x^3+x^2+x$ | 0 | 0 |
| 1 | 1 | 1 | $x^2+1$ | $x^4+x^2+x$ | $x^3+x^2+x$ | 0 | $x^3+x^2+x$ |
| 2 | 0 | 0 | $x^3+x$ | $x^4+x^2+x$ | $x^3+x^2+x+1$ | $x^3+x^2+x$ | $x^3+x^2+x+1$ |
| 3 | 1 | 1 | x | $x^4+x^2$ | $x^3+x^2+x+1$ | $x^3+x^2+x$ | $x^3+x^2+x+1$ |
| 4 | 0 | 0 | $x^2$ | $x^4+x^2$ | $x+1$ | $x^3+x^2+x+1$ | $x^3+x^2+1$ |
| 5 | 1 | 0 | $x^3$ | $x^4+x^2$ | $x+1$ | $x^3+x^2+x+1$ | $x^3+1$ |
| 6 | 2 | 1 | $x^2$ | $x^4$ | $x+1$ | $x^3+x^2+x+1$ | $x+1$ |
| 7 | 1 | 1 | $x^3$ | $x^4$ | $x+1$ | $x^3+x^2+x+1$ | $x^2+x$ |
| 8 | 0 | 0 | 0 | $x^4$ | 0 | $x+1$ | $x^3+x^2+x+1$ |

**B.** Proposed Algorithm for Division

The main disadvantage of the above architecture is the required high number of clock cycles per a division over GF ($2^m$ ) i.e,. ($2m-1$ cycles). The proposed new algorithm merges two stages of the binary extended GCD algorithm over GF ($2^m$ ) to obtain the result of two stages simultaneously so that the whole stage can be done in one clock cycle, and in a way that the algorithm is directly proper to be implemented in hardware. It has been tried to obtain a relation between the registers (R, S, U and V) and states (count and state) of iteration with the corresponding registers and states of two previous stages.

```
$R=B(x)$; $U=A(x)$;$S=G(x)$;$V==0$; $(C1=C2=J)==0$;
For $i=1$ to $m-1$ do
    If $C1==0$ then $C2==0$;
    If $C2==0$ then $C1++$;    Else $C1--$;
    If $((C2\&C1)==0)$ then $J==1$;    Else $J==0$;
$R = R + [(!R[0]\&R[1])XOR(R[0]|R[1]))].S + [((R[0]\&!R[1])\&(R[0]|R[1])].$
        $[(C2|!R[0]).S+(!(C2|!R[0]).R].X$;
$U = U + [(!R[0]\&R[1])XOR(R[0]|R[1]))].V + [((R[0]\&!R[1])\&(R[0]|R[1])].$
        $[(C2|!R[0]).V+(!(C2|!R[0]).U].X$;
$S=(J|R[0]).R+[(!(C2|R[0]|R[1])XOR(C2\&!(J\&R[1]))))].S+[((C2|R[0]|!R[1])$
      XOR(C2 & R[1] & !J))].$\frac{R+R[0].S}{X}$;
$V=(J|R[0]).U+[(!(C2|R[0]|R[1])XOR(C2\&!(J\&R[1]))))].V+[((C2|R[0]|!R[1])$
      XOR(C2 & R[1] & !J))].$\frac{U+R[0].V}{X}$;
    If $((C2|!R[0])\&R[1]==1)$ then $C2==1$;
    If $(!(C2|!R[0])|C2==1)$ then $C1++$;    Else $C1--$;
$R=\frac{R}{X^2}$
$U=\frac{U+R[0].X+R[1]}{X^2}+(R[0]|R[1]).\frac{S}{X}+(R[0]\&R[1]).\frac{S}{X^2}$
End.
```

As the result of the states are conditioned to the result of the registers after an iteration and also the registers are calculated respect to the status of the states, an extra state condition , J , is added to register the state in the middle stage. The following algorithm shows the result of the performed experience. In the above algorithm, the three parameters: C1, C2 and J, control the flow of the iterations and in each stage, the content of R, S, U and V are computed using these condition parameters along by the first and second bits of R in each stage (i.e. R[0], R[1]). At the end of each iteration a division by $X^2$ is performed that can be done just by a right shifting of the corresponding register. Although the proposed algorithm seems to be complex, its hardware implementation is very simple and just use logical gates and latches.

**Table 2.** An Example of Proposed Algorithm for Division in GF( $2^4$)

| i | C1 | C2 | J | R | S | U | V |
|---|---|---|---|---|---|---|---|
|  | 0 | 0 | 0 | $x^3+x+1$ | $x^4+x+1$ | $x^3+x^2+x$ | 0 |
| 1 | 1 | 0 | 0 | $x^4+x^3 \rightarrow x^2+x$ | $x^2+x+1$ | $x^3+x^2+x \rightarrow 1$ | $x^3+1$ |
| 2 | 2 | 1 | 1 | $x^2+x \rightarrow x$ | $x^2+x$ | $1 \rightarrow x+1$ | 1 |
| 3 | 1 | 1 | 0 | $x \rightarrow x$ | x+1 | $x+1 \rightarrow x$ | x+1 |
| 4 | 0 | 1 |  | x |  | x | x+1 |

In table 2, first line shows initial polynomials, at first cycle (i=1) by state conditions (C1=1, C2=0) and with the equations for R, S, U and V, we would have new polynomials. In this cycle after calculating R and U, at the end of the program new values should be replaced (i=1, columns 5,7). This computation continues to i=4 and we can see the same value for V that had been obtained by i=3, thus for m-1 cycle will catch.

## 3 Hardware Implementation

The algorithm is designed so that it could be implemented by simple basic gates and latches for synchronization for the iterations. The proposed hardware consists of four separated modules. These modules are designed to calculate R, S, U and V registers that are exposed in a block diagram as follows. The calculation manner of control parameters, C1, C2 and J have been shown in program of proposed algorithm (VHDL language-section 2.B). The statements of $/X$ and $/X^2$

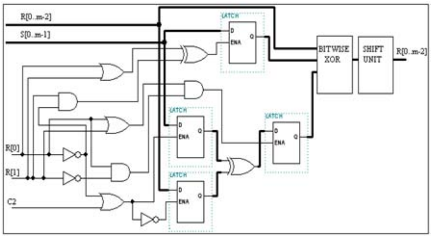


**Fig. 1.** Random Inputs for Security Block

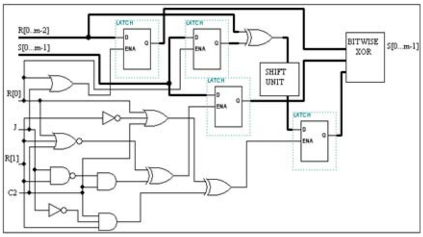


**Fig. 2.** Outputs of Security Block with $\lambda$ Digit

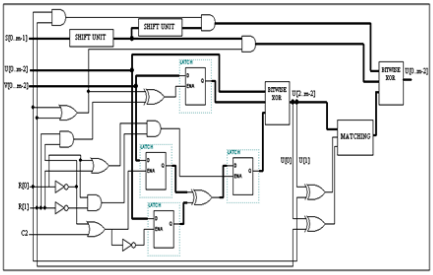


**Fig. 3.** Random Inputs for Security Block

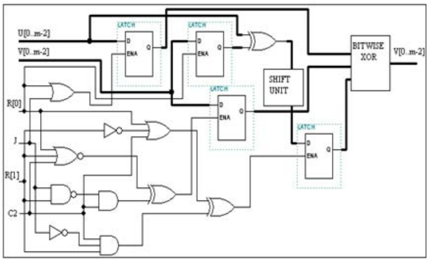
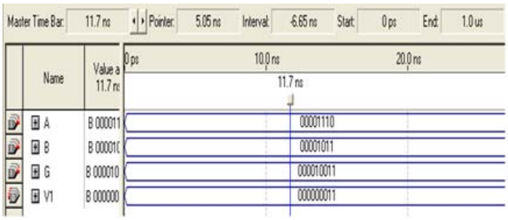


**Fig. 4.** Outputs of Security Block with $\lambda$ Digit


Master Time Bar: 11.7 ns Pointer: 5.05 ns Interval: -6.65 ns Start: 0 ps End: 1.0 us

| Name | Value a 11.7 n | 11.7 ns |
|---|---|---|
| A | B 000011 | 00001110 |
| B | B 00001( | 00001011 |
| G | B 000010 | 000010011 |
| V1 | B 000000 | 000000011 |

**Fig. 5.** The Result of new Algorithm for division

is implemented by shifted wiring of the corresponding data buses connecting to the latches. The module of R, S, U and V block in the new algorithm implementation are represented in Fig.1, Fig.2, Fig.3 and Fig.4 respectively. Also, the result of new algorithm for division is represented in Fig.5.

## 4 Encryption Scheme in Application on MC-DS-CDMA Transmitter and Receiver

Chaos was defined as process like random existing in the confirmed system. This process was non periodical, and it was stable on the whole and extensive on spot[1]. The chaos signal especially suited for secret communication because it has the characters of conceal, inscrutability, high complexity which are liable to achieve [9]. In this paper we offer simple chaos circuit that is combined with ECC to produce cypher. For this production, new algorithm for division helps that time consumption becomes the least. The basic character of the chaos motion [9] could be defined as follows for $n = 0, 1, 2, ...$ where:

$$x_{n+1} = k.x_n.(1 - x_n) \tag{5}$$

$$k \in (3.4688596, 4) \tag{6}$$

$$x_n \in (0, 1) \tag{7}$$

Assume that every symbol of one user is passed from serial to parallel block and we have $d_i^{(0)}(n)$ , $i \in \{0, 1, ..., v-1\}$ with n bit-symbols in each line [10]. Suppose that bit-symbols of one line go to security block (Fig.6), at first the digits would change with Chaos algorithm, afterwards these are varied with based digits to make Elliptic curve algorithm input data (Fig.7).

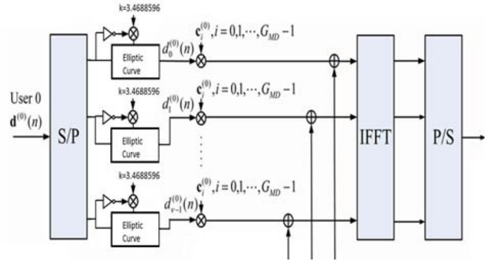


**Fig. 6.** Random Inputs for Security Block

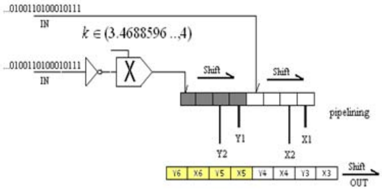


**Fig. 7.** Outputs of Security Block with $\lambda$ Digit

$a_2$, $a_6$ are calculated with each pair of $\{(x1, y1), (x2, y2)\}$ in Elliptic curve equation i.e., $(y^2 + x.y = x^3 + a_2.x^2 + a_6)$, and then:

If $a_6 \neq 0$ then
$x_3 = \lambda^2 + \lambda - x_1 - x_2 - a_2$ ; // $a_2 = \frac{y_1^2+x_1.y_1+x_2^3-y_2^2-x_2.y_2-x_1^3}{x_1^2-x_2^2}$
$y_3 = (x_1 - x_3).\lambda - x_3 - y_1$ ;
If $x_1 \neq x_2$ then $\lambda = \frac{y_1-y_2}{x_1-x_2}$ ; else $\lambda = \frac{3x_1^2+2a_2x_1-y_1}{2y_1+x_1}$
If $x_1 = x_2 = 1$ then $a_2 = -0.5$ ;
If $x_1 = x_2 = 0$ then $a_2 = 0$.

The rest of the computation for output digits is adherence for equations above. Conversly at the receiver with cypher digits and $\lambda$, we could gain our initial values again. Decryption is calculated in the following:

$\lambda = \frac{(\sim x_1 - \sim x_2).k}{x_1-x_2} = \sim k$ ;
if $(\lambda = \sim k)$ $x_1 \neq x_2$ ; else $x_1 = x_2$ ;
$y_1 = \sim x_1.k$ ; $x_1 = \frac{in(1,2)+(\lambda+1).in(1,1)}{\lambda - \sim k}$; $x_1 \propto x_2$.

In encryption, parallel symbols in each line are separated into 2 digits $(x_1, x_2)$ so in decryption, at receiver we have in(1, 1) and in(1, 2) for each stage.

## 5 Simulation Results by MATLAB

In $d^{(k)}(n) = [d_0^{(k)}(n), d_1^{(k)}(n), ..., d_{v-1}^{(k)}(n)]$[10] if $v = 100$ we have 100 lines after S/P (Fig.6) and we assume 4-bit symbols for each input symbol (Matrix [100,4]). Every 2-bit symbols input to the security block have 2-bit outputs (Fig.7) that calculate with fourth section equations.

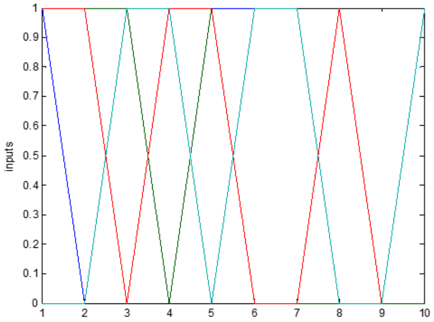


**Fig. 8.** Random Inputs for Security Block

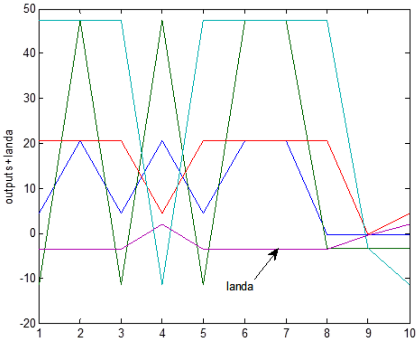


**Fig. 9.** Outputs of Security Block with $\lambda$ Digit

## 6 Conclusion and Comparison

In this paper the algorithm of calculating modular division over GF ($2^m$ ) based on extended binary GCD is modified to improve the system implementation speed. The important differences between Reclaimed Euclid's algorithm[2] and proposed algorithm are throughput and latency with 1/2(m-1) and (5m-4) values respectively for first algorithm and 1/(m-1) and m-1 values respectively for second algorithm. Additionally an algorithm is designed and provided directly for the hardware so that as shown in Table 3, almost affiliation of the hardware are obliterated with the size of the field selected (i.e. the amount of m in GF($2^m$)). Table 3 shows the compression of the proposed algorithm with the most commonly used algorithms of Kim [11] and Bruner [6] considering both in speed and area consumption. These comparisons show that proposed algorithm has the smallest latency.Of course, if the algorithm is implemented by the systolic architecture, it will result in better specifications in speed and cell delays as well as improvement in clock frequency. That is the future work of the author.

**Table 3.** Compression of the proposed algorithm with the most commonly used algorithms of Kim and Bruner, in speed and area consumption

| | Bruner et.al | Kim et.al | Proposed algorithm |
|---|---|---|---|
| Throughput (1/cycles) | 1/2m | 1/2m | 1/(m-1) |
| Latency (cycles) | 2m | 2m | m-1 |
| Basic Components and their numbers | AND2:3M+log(m+1)<br>XOR:3m+log(m+1)<br>FF:4m+log(m+1)<br>MUX2:8m | AND2 :3m+5<br>XOR:3m+1<br>FF:5m+2<br>MUX2:4m+4<br>OR2:1 | AND2:12<br>XOR:5m+12<br>FF:16<br>OR2:8<br>AND3:2<br>OR3:2 |

The reclaimed Euclid's algorithm based on the COMPASS 0.6um CMOS is performed with maximum propagation delay 3ns and clock rate up to 167MHZ[2], while our proposed algorithm is implemented by MAX7000S series-FPGA is performed with delay time 1ns and maximum frequency 10000MHZ. In resumption, the algorithm of cryptography is security voucher with combination of two encryption algorithms to complete MC-DS-CDMA block. The results expose that finding relation between output-columns is impossible, because encryption equations are based on arbitrary environment such as Chaos algorithm with its random characters and Elliptic curve in cycloids too. The Fig.8 and Fig.9 show in-out and out-in encryption and decryption unit sequentially.

## References

1. Murali, K., Yu, H., Varadan, V., Leung, H.: Secure Communication Using a Chaos Based Signal Encryption Scheme. IEEE Transaction on Consumer Electronics 47, 709–714 (2001)
2. Guo, J.-H., Wang, C.-L.: Hardware-efficient systolic architecture for inversion and division in GF($2^m$). In: IEE Proc.-Comput. Digit. Tech., vol. 145, pp. 272–278 (July 1998)
3. Jain, S.K., Song, L., Parhi, K.K.: Efficient Semi-Systolic Architecture for Finit Field Arithmetic. IEEE Transaction on VLSI System 6, 101–113 (1998)
4. Wang, C.L., Lin, J.L.: A Systolic Archtecture for Computing Inverses and Divisions in Finit Field GF($2^m$). IEEE Transaction on Computer 42, 1141–1146 (1993)
5. Hasan, M.A., Bhargava, V.K.: Bit-Serial Systolic Divider and Multiplier for Finit Fields GF($2^m$). IEEE Transaction on Computer 41, 972–980 (1992)
6. Brunner, H., Curiger, A., Hofstetter, M.: On Computing Multiplicative Inverses in GF($2^m$). IEEE Trans. Comput. 42, 1010–1015 (1993)
7. Wu, C.-H., Wu, C.-M., Shieh, M.-D., Hwang, Y.-T.: High-Spesd, Low-Complexity Systolic Designs of Novel Iteration Division Algorithms in GF($2^m$). IEEE Transaction on Computer 53, 375–380 (2004)
8. Fong, K., Hankerson, D., Lopez, J., Menzes, A.: Field Inversion and Point Halving Revisited. IEEE Transaction on Computers 53, 1047–1059 (2004)
9. Deng-Hong, Z.: Encryption design for the database under the VFP environment based on Chaos algorithm. In: IEEE, ICACTE (2010)
10. Chen, J.-D., Ueng, F.-B., Chang, J.-C., Hsien, S.: Performance Analyses of OFDM-CDMA Receivers in Multipath Fading Channels. IEEE Transactions on Vehicular Technology 58, 4805–4818 (2009)
11. Kim, C.H., Kwon, S., Kim, J.J., Hong, C.P.: A Compact and Fast Division Architecture for a Finite Field GF($2^m$). In: Kumar, V., Gavrilova, M.L., Tan, C.J.K., L'Ecuyer, P. (eds.) ICCSA 2003. LNCS, vol. 2667, pp. 855–864. Springer, Heidelberg (2003)