# Multiplication and Division over Extended Galois Field GF($p^q$): A new Approach to find Monic Irreducible Polynomials over any Galois Field GF($p^q$).

Sankhanil Dey[1] and Ranjan Ghosh[2].
sdrpe_rs@caluniv.ac.in[1], rghosh47@yahoo.co.in[2].
Institute of Radio physics and Electronics.
92 APC Road, Kolkata-700009.
University of Calcutta.

**Abstract.** Irreducible Polynomials (IPs) have been of utmost importance in generation of substitution boxes in modern cryptographic ciphers. In this paper an algorithm entitled Composite Algorithm using both multiplication and division over Galois fields have been demonstrated to generate all monic IPs over extended Galois Field GF($p^q$) for large value of both p and q. A little more efficient Algorithm entitled Multiplication Algorithm and more too Division Algorithm have been illustrated in this Paper with Algorithms to find all Monic IPs over extended Galois Field GF($p^q$) for large value of both p and q. Time Complexity Analysis of three algorithms with comparison to Rabin's Algorithms has also been exonerated in this Research Article.

**Keywords.** Galois Field, Finite field, Irreducible Polynomials (IPs), Monic IPs.

**1. Introduction.** Monic IPs have been of utmost importance in Applied Cryptography, The standard 8-bit S-Box of Advance Encryption Standard or AES has been usually generated by all Multiplicative Inverse (MI) Polynomials under a particular Monic IP {11B} over Galois Field GF($2^8$) with a particular additive constant {63} [1].

The Basic Polynomials or BPs over Galois field GF($p^q$) have been Polynomials with highest degree of terms q and so it must have (q+1) terms . Elemental Polynomials or EPs have been polynomials with highest degree of terms less than q and so it must have less than (q+1) terms from 1 through q. BPs with leading co-efficient unity have been termed as Monic BPs. Monic BPs that do not have two Monic EPs rather than Constant Polynomials have been termed as Monic IPs. The EPs with degree d = 0 has been termed as Constant Polynomials (CPs) and they are p in numbers and not in consideration in this paper. Generator Polynomials or GPs have been polynomials with number of terms less than or equal to (q+1) and the code word or generated polynomials from BPs have been divisible by GPs but that are also not in consideration in this paper.

The hands on computation to find Monic Irreducible Polynomials over Galois Field GF($p^q$) for p = 2, q = 2 through 11, p = 3, q = 2 through 7, p = 5, q = 2 through 5 and for p = 7, q = 2 through 4 has initiated by Church [1] in his contribution. The GF equivalents of each Monic BP for p = 2 through 7 had also been reported in his contribution [2]. Here each two Monic EPs are multiplied to obtain the reducible Monic BPs. The search for Monic IPs ended up with cancellation of all reducible Monic BPs leaving behind the Irreducible Monic BPs. In Rabin's Algorithm [3] all Monic BPs (F(x)) over Galois Field GF(p) of degree n has been tested for divisibility with ($x^n$-x) and the gcd of (F(x), $x^{nki}$-x) where the $k_i$ are all the prime divisors of n , to be unity. If Iany monic BP, F(x) satisfies both condition, the Monic BP is termed as Monic IPs.

According to Zaman and Ghosh [4] if the residue of each polynomial division of each Monic BP with all Monic EPs are unity or every Monic EP has a multiplicative inverse over Galois Field under a Monic BP then the Monic BP is termed as a Monic IP. The algorithm has also been implemented using Galois Field division and termed as composite algorithm.

Multiplication over Galois field of two Monic EPs with degree d and q-d generate Monic reducible Polynomials or RPs. Search of IPs through The list all Monic BPs gives Monic IPs as the left behind Monic BPs after generation of all Monic RPs through Multiplication Algorithm. If The Division over Galois field of a Monic BP with all Monic EPs with highest degree 1 through (q-1)/2 has been incomplete then the Monic BP has been termed as Monic IP.

The composite Algorithm has been demonstrated in sec.2. The Multiplication and division Algorithm has been described in sec.3. and sec.4. respectively. The Comparison of Time Complexity of three algorithms with Rabin's algorithm has been given in section 5. Conclusion and references of the paper have been given in Sec.6. and sec. 7. respectively. The Pseudo Code of algorithms of Galois field Multiplication and division have been given in Appendix.

**2. Computational Algorithm of Improved Composite Algorithm.**

Let us consider Monic BPs BP over Extended Galois Field GF($p^q$) with degree of basic polynomial BP, BPD €q and consider Monic EPs EP with degree EPD € {1,2,…..,(q-1)/2}. Since Monic EPs with degree d and q-d are MIs of each other so the division is restricted to aforesaid condition. Now let total Number of Monic BPs have been $p^q$ € n and Monic EPs ($p^q$-p) € n-p;

**Start.**
**Step 0A:** BP_Numbers: n; // Defining Total Numbers of Monic BP to be tested for being a Monic IP.
**Step 0B:** EP_Numbers: n-p; // Defining Total Numbers of Monic EPs.
**Step 01:** For BP_index::1: n. // Accessing Each Monic BP.
**Step 02:** For EP_index::1: n-p. // Accessing Each Monic EP.
**Step 03:** If( BP %(GF) EP == 1) Flag[EP_index]=1; // Testing of Boundary Condition for a Monic BP to be a Monic IP
End For EP_index. // End of For loop BP_index.
**Step 04:** If (Flag[EP_index]==1) ¥ EP_index. BP= IP. // Declaration of a Monic BP to a Monic IP.
Else BP= RP. // Declaration of a Monic BP to a Monic RP.
End For BP_index. // End of For loop EP_index.

**Stop.**

Time Complexity of the aforesaid Algorithm or Composite Algorithm have been defined as,
Time Complexity of Composite Algorithm = $O(n^2)$.

### 3. Computational Algorithm of Multiplication Algorithm.

Let us consider Monic BPs BP over Extended Galois Field $GF(p^q)$ with degree BPD €q and consider Monic EPs $EP^1$ and $EP^2$ with degree $EPD^1$ € {1,2,…..,(q-1)/2} and $EPD^2$ € {q-1,q-2,…..,q-(q-1)/2}. Since Monic EPs with degree d and q-d have been multiplied to obtain Monic RPs. List of RPs have been cancelled leaving behind list of Monic IPs. Number of monic BPs have been $p^q$ € n and Monic EPs ($p^q$-p) € n-p;

**Start.**
**Step 0A:** BP_Numbers: n; // Defining Total Numbers of Monic BP to be tested for being a Monic IP.
**Step 0B:** EP_Numbers: n-p; // Defining Total Numbers of Monic EPs.
**Step 01:** For $EP\_index^1$::1: (q-1)/1. And $EP\_index^2$::q-1: q-(q-1)/1. // For loop to access each Monic EP
**Step 02:** RP = $EP[EP\_index^1]$×(GF) $EP[EP\_index^2]$. // Testing Condition.
End For $EP\_index^1$ and $EP\_index^2$. // End of For loop $EP\_index^1$ and $EP\_index^2$.
**Step 03:** List of Monic RPs have been cancelled from the list of Monic
BPs leaving behind the list of Monic IPs. // Condintion to search for Monic IPs
**Stop.**

Time Complexity of the aforesaid Algorithm or Composite Algorithm have been defined as,
Time Complexity of Composite Algorithm = $O(n^2)$.

### 4. Computational Algorithm of Division Algorithm.

Let us consider Monic BPs BP over Extended Galois Field $GF(p^q)$ with degree BPD €q and consider Monic EPs EP with degree EPD € {1,2,…..,(q-1)/2}. Since Monic EPs with degree d and q-d are MIs of each other so the division is restricted to aforesaid condition. Now let total Number of Monic BPs have been $p^q$ € n and Monic EPs ($p^q$-p) € n-p;

**Step 0A:** BP_Numbers: n; // Defining Total Numbers of Monic BP to be tested for being a Monic IP.
**Step 0B:** EP_Numbers: n-p; // Defining Total Numbers of Monic EPs.
**Step 01:** For BP_index::1: n. // Accessing Each Monic BP.
**Step 02:** For EP_index::1: n-p. // Accessing Each Monic EP.
**Step 03:** If( BP %(GF) EP != 0) Flag[EP_index]=1; // Testing of Boundary Condition for a Monic BP to be a Monic IP
End For EP_index. // End of For loop BP_index.
**Step 04:** If (Flag[EP_index]==1) ¥ EP_index. BP= IP. // Declaration of a Monic BP to a Monic IP.
Else BP= RP. // Declaration of a Monic BP to a Monic RP.
End For BP_index. // End of For loop EP_index.

Time Complexity of the aforesaid Algorithm or Composite Algorithm have been defined as,
Time Complexity of Composite Algorithm = $O(n^2)$.

### 5. Time Complexity of Computational Algorithms..

The three new algorithms to find Monic IPs over Galois field $GF(p^q)$ have a time complexity of $O(n^2)$. Since Time complexity of Rabin's algorithm and its modification depends upon the value of prime modulus P so it becomes slower for large

value of P. Now in these three algorithms the complexity depends upon the value of extension q so they are faster and eligible to find Monic IPs for very large value of P as well as extension q. Since each Algorithm using two nested loops so the Time Complexity turns to be $O(n^2)$. The comparison of Time Complexity of three algorithms with Rabin's algorithm and its modification has been given in Table 1.

| **Algorithms** | **Composite Algorithm** | **Multiplication Algorithm** | **Division Algorithm** | **Rabin's Algorithm** | **Rabin's Algorithm(mod)** |
|---|---|---|---|---|---|
| **Time Complexity** | $O(n^2)$ | $O(n^2)$ | $O(n^2)$ | $O(n^4(\log P)^3)$ | $0(n^4(\log p)^2 + n^3(\log P)^3)$ |

**Table.1. Comparison of Time Complexity of Three new algorithms with Rabin's and Modified Rabin's Algorithms.**

## 6. Results and Discussion.

In the present work the three algorithms find Monic IPs over Galois fields, $GF(3^2)$ with Number of Monic IPs 3, $GF(3^3)$ with Number of Monic IPs 8, $GF(3^5)$ with Number of Monic IPs 50, $GF(3^7)$ with Number of Monic IPs 312, $GF(7^2)$ with Number of Monic IPs 21, $GF(7^3)$ with Number of Monic IPs 112, $GF(7^5)$ with Number of Monic IPs 2157, $GF(7^7)$ with Number of Monic IPs 117648, $GF(11^3)$ with Number of Monic IPs 440 and $GF(101^3)$ with Number of Monic IPs 343400. The Number of Monic IPs and Particular Monic IPs are same with three algorithms. It can also be predicted that the algorithms are able to find monic IPs over large values of Galois field prime modulus p and large values of extension q. All values of IPs have been checked with available derived monic IPs by Church[2] in his contribution. All IPs are concluded to be proper and right.

## 7. Conclusion.

Monic IPs over Galois Field $GF(p^q)$ have been of utmost importance in generation of large S-Boxes. Finding Monic IPs o over Galois Field $GF(p^q)$ for large value of p and q have been solved due to these three algorithms. The list of Monic IPs over Galois Field $GF(p^q)$ for 8 large values of p and q have been enclosed as supplementary material with this research article. These Algorithms have been fast and have less programming complexity. So it helps Crypto Community to Find Monic IPs over Galois Field $GF(p^q)$ for large value of prime moduli P and extension q.

## 8. References.

**[1]** Das, S, Zaman, J.K.M.S. Uz, Ghosh, R, "Generation of AES S-Boxes with various modulus and additive constant polynomials and testing their randomization", Proceedings (CIMTA) 2013, Procedia Technology vol. 10.( 2013 ), pp: 957 – 962.
**[2]** Church R., "Tables of Irreducible Polynomials for the first four Prime Moduli", Annals of Mathematics, Vol. 36(1), pp. 198 – 209, January, 1935.
**[3]** Jacques C'almet And Riidiger Loos, "An Improvement of Rabin's Probabilistic Algorithm For Generating Irreducible Polynomials Over Gf(P)", Information Processing Letters, 20 October 1980, Volume 11, No. 2.
**[4]** Zaman , J K M Sadique Uz, Dey sankhanil, Ghosh, R, "An Algorithm to find the Irreducible Polynomials over Galois Field $GF(p^m)$", International Journal of Computer Applications (0975 – 8887) Volume 109 – No. 15, January 2015.

## Appendix

**1. Pseudo Code of Algorithm of Multiplication over Galois Field GF($p^q$).**

```
Start.
Step 0A: Number of coeff_prod = q;                    // Defining Number of Product Terms.
Step 0B: Number of coeff_multiplicand = d;            // Defining Number of Multiplicand Terms.
         Number of coeff_multiplier = q-d;            // Defining Number of Multiplier Terms.
Step 01: For Prod_indx::1::q-1.                       // For loop to calculate all product Coefficients.
Step 02: For Mul_indx::1::q-1.                        // For loop to calculate all Multiplicand Coefficients
Step 03: Prod[Prod_indx]= [Prod[Prod_indx]+[Multiplicand[Mul_indx]×Multiplier[q-Mul_indx]]]%p; //Multiplication
End For Mul_indx.                                     // End of For loop Mul_indx
End For Prod_indx.                                    // End of For loop Prod_indx
Stop.
```

**2. Pseudo Code of Algorithm of Division over Galois Field GF($p^q$).**

```
Start.
Step 0A: Number of coeff_divident = q;    // Defining Number of Product Terms.
Step 0B: Number of coeff_divisor = d;     // Defining Number of Multiplicand Terms.
Step 0C: int quotient [q-d+1];
         int remainder[q+1];
         int MI_of_ep_coeff[d+1];
Step 01: For i::0::q-d+1.                 // For loop to calculate all quotient and remainder Coefficients.
Step 02:  quotient [q-d+i] = remainder[q-i]* MI_of_ep_coeff[d]; // Calculation of quotient Coefficients.
Step 03: For j::0::d+1.                   // For loop to calculate all remainder Coefficients.
Step 04: remainder [q-i-j] = remainder [q-i-j]-((quotient[q-d-i] ×divisor[d-j]))%p; // Calculating remainder Coefficients
End For j;                                // End of For loop j.
End For i;                                // End of For loop i.
Stop.
```