Mathematical and Software Engineering, Vol. 1, No. 1 (2015), 18-24
Varεpsilon Ltd,   http://varepsilon.com

Algorithms for Extended Galois Field Generation and Calculation

Zhaneta N. Savova-Tasheva[1] and Antoniya T. Tasheva[2]

[1]Faculty of Artillery, AAD and CIS, National Military University;
Faculty of Technical Sciences, Konstantin Preslavski University of Shumen;
Bulgaria
[2]Faculty of Computer Systems and Control, Technical University of
Sofia, Bulgaria

Abstract

The paper aims to suggest algorithms for Extended Galois Field generation and calculation. The algorithm analysis shows that the proposed algorithm for finding primitive polynomial is faster than traditional polynomial search and when table operations in GF(p^m) are used the algorithms are faster than traditional polynomial addition and subtraction.

Subject Codes (ACM, MSC, or PACS): F.2.1, G.1.3, H.4.3, 11Y04.

Keywords: Extended Galois Field, GF(p), GF(p^m), Primitive Polynomials.

1 Introduction

The broadcasting channels, especially the wireless, are not ideal due to the presence of outside influences like noise, interference or echo effects, which superpose with the useful signal and lead to the occurrence of errors. It is needed to assure low error level in the channel (for example in the DVB, it is necessary to achieve BER in the order of $10^{-10} - 10^{-12}$ with data transmission speed of 30 *Mb/s*). A channel with that low error level is called a *quasi-error-free* channel. In order to achieve this low error level some prevention measures need to be taken. By them the reception side will be provided with a detecting and correcting mechanism for as many errors as possible. This can be achieved by error-correcting coding by introducing a recalculated piece of redundant information. One of the most commonly used codes in error correction codes are Reed-Solomon (RS) [11] and Bose-Chaudhuri-Hocquenghem (BCH) [3]. One of their key features is that they allow correction of the multiple-burst bit-errors in the transmission channel.

Since RS and BCH codes are linear, i.e. they are vector fields that exist only if the size of the alphabet q is a prime p or a degree of a prime number p^m, it is needed to synthesize accelerated algorithms for generation and operations in the Galois Field GF(p) and its extension $GF(\mathrm{p}^m)$ in order to generate the codes.

The algorithms for constructing an Extended Galois Field $GF(\mathrm{p}^m)$ include as their first step the task of finding a primitive polynomial of degree m over $GF(p)$. Typically, this task is quite laborious and is solved by the consistent check of all the polynomials $p(x) \in GF(p)[x]$ of degree m. The volume of calculations to solve the problem is growing tremendously with the increasing of the prime p and the degree m.

Considering the problems above the main purpose of this article is synthesis of algorithms for accelerated generation and work in Extended Galois Field $GF(p^m)$. The paper is organized as follows. First a brief introduction to the issue is made. Then an algorithm for generating an Extended Galois field $GF(q = p^m)$ is proposed and tables for accelerated implementation of the operations for addition and subtraction in the generated finite extension are synthesized. Finally, an assessment of the performance of proposed algorithms compared to the traditional is made.

2 Methods and Algorithms

2.1 Algorithm Synthesis for Accelerated Generating of Extended Galois Field *GF*(*p*^{m})

The task for building an extension of the Extended Galois Field $GF(p^m)$ is closely linked to the task of finding a primitive polynomial over $GF(p)$. Typically, this task is quite laborious and is solved by the consequent testing of all the polynomials $p(x) \in GF(p)[x]$ of degree m [4], [5], [6], [8], [9]. Large numbers of mathematicians and coding theoreticians have worked on this issue [1], [6], [7], but still no one has established an algorithm to exclude completely the search among polynomials. Some mathematical proofs were made to facilitate the resolution of the problem and reduce the number of required searches.

For example if $r = (p^m - 1)/(p - 1)$ the necessary and sufficient conditions for $p(x)$ to be primitive polynomial in $GF(p)$ [10] are:

1. $((-1)^{m+1}a_0)^{(p-1)/q} \neq 1$ for every prime multiplier q of $p - 1$;
2. $x^r \bmod p(x) = (-1)^{m+1}a_0$;
3. $x^{r/q} \bmod p(x)$ has degree more than 0 for every prime multiplier q of r, $1 < q < r$.

In this synthesized algorithm for accelerated generating of Extended Galois Field another approach is used:

First, the value a_0 is found such that satisfies the shown above first necessary and sufficient condition of Knuth [10]. Since when generating the Extended Galois Field $GF(p^m)$ it is needed to generate the Galois Field $GF(p)$ in advance, the decomposition of the even number p–1 into prime multipliers has already been fulfilled.

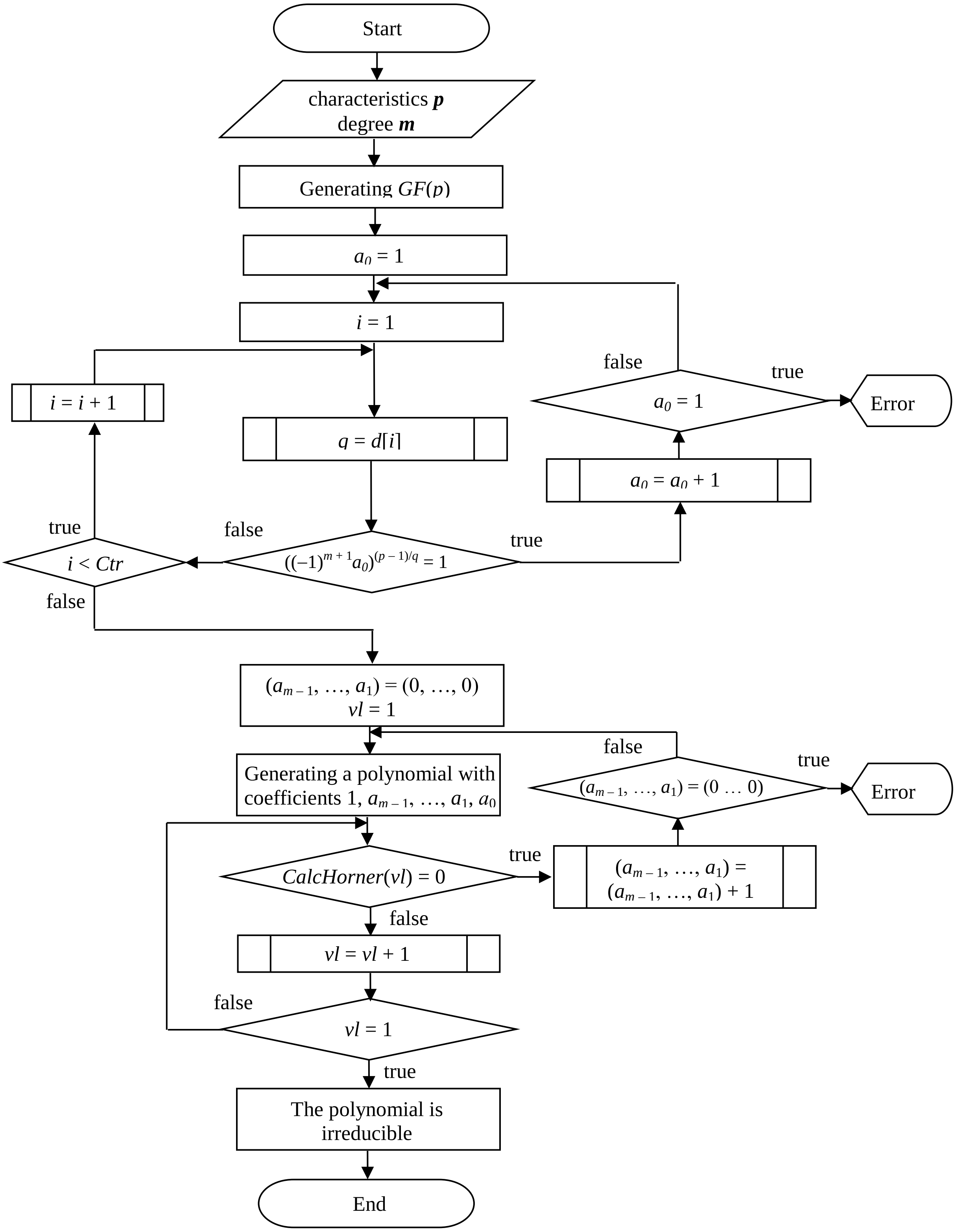

Start
characteristics p
degree m
Generating GF(p)
a0 = 1
i = 1
i = i + 1
q = d[i]
false
true
a0 = 1
Error
a0 = a0 + 1
true
false
true
i < Ctr
((−1)^(m + 1)a0)^((p − 1)/q) = 1
false
(am − 1, …, a1) = (0, …, 0)
vl = 1
false
true
Generating a polynomial with coefficients 1, am − 1, …, a1, a0
(am − 1, …, a1) = (0 … 0)
Error
true
CalcHorner(vl) = 0
(am − 1, …, a1) = (am − 1, …, a1) + 1
false
vl = vl + 1
false
vl = 1
true
The polynomial is irreducible
End

Figure 1. *Algorithm for finding the first irreducible polynomial in GF*(*p*)

Second, a random search among the other coefficients (a_{m-1}, …, a_1) is done.

This implementation lowers the volume of the search *p* times. If the search is done for all the possible coefficient combinations, all irreducible polynomials over Galois Field *GF*(*p*) will be found. In order to generate the field it is sufficient to find only a single polynomial. Therefore the search is suspended when the first primitive polynomial is found, and this accelerates the generation of the extension even more. The other irreducible polynomials are obtained after generating the field. A simplified block diagram of the algorithm for finding a primitive polynomial in the *GF*(*p*) is presented in Fig. 1.

For the final generation of Extended Galois Field it is necessary to find a primitive element α [2], [4], [6] originating the field. To speed up the generation process, it is first checked whether *x* is a primitive element for the previously found irreducible polynomial (the polynomial *x* is a primitive element in about 60% of the fields $GF(p^m)$). Only when *x* is not a primitive element another primitive element is sought. Summarizing the foregoing, the proposed algorithm for generating the Extended Galois Field contains four basic steps shown in Fig. 2:

1. ***Finding an irreducible polynomial*** $p(x) \in GF(p)[x]$ in the Galois Field ***GF(p), such that p(x) ≠ 0, x = 1, …, p−1.***
2. ***Check,*** **whether *x* is a primitive element α for the irreducible polynomial found in step 1** $x^{p^m-1} \equiv 1 \bmod p(x)$**. If true, go to step 4.**
3. ***Search for a primitive element α ≠ x.***
4. ***Calculation of the powers*** **of the primitive element α.**

Figure 2. *Basic steps of the algorithm for generating an Extended Galois Field* $GF(p^m)$

2.2 Algorithm synthesis for operations in the Extended Galois Field $GF(p^m)$

The main operations that are carried out in Extended Galois Field $GF(p^m)$ are addition, subtraction, multiplication and division. Having the degrees of the primitive element α that were obtained by step 4 of the algorithm shown on Fig. 2 one can easily perform multiplication and division operations [2], [4], [6]. Instead of the traditional way to perform addition and subtraction operations in which conversion between power and polynomial representation, conducting the operations and again the opposite conversion is performed, creation of tables for accelerated addition is proposed. For their generation the conventional algorithm [2], [4], [6], [8] is used. In general, the tables have the size with q^2 elements, but since they are symmetrical against the main diagonal only

(1) $$q + (q - 1) + \ldots + 1 = q.(q + 1)/2$$

values from the table are stored. Example for a table for accelerated addition in $GF(2^3)$ is Table 1.

For fast execution of the subtraction operation additional *q* in count elements (–*a*) are stored for conducting simple addition. Such table is not necessary for $GF(2^m)$ as (–*a*) = *a*.

Table 1. *Addition in GF*$(8 = 2^3)$

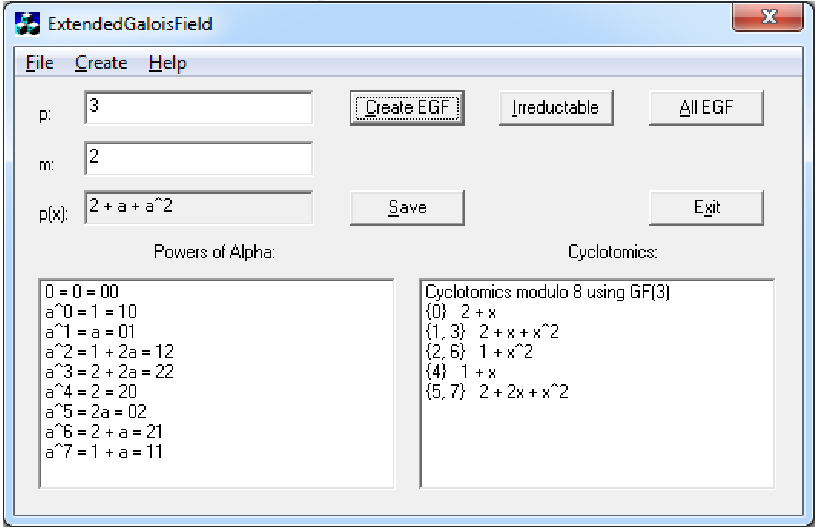
+	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	1	2	3	4	5	6	7
1	1	0	4	7	2	6	5	3
2	2	3	0	5	1	3	7	6
3	3	7	5	0	6	2	4	1
4	4	2	1	6	0	7	3	5
5	5	6	3	2	7	0	1	4
6	6	5	7	4	3	1	0	2
7	7	3	6	1	5	4	2	0

For applications that require even greater speed, such as the transmission of video in real time, similar tables for accelerated execution of the operation multiplication and division can be created.

3 Results and Discussion

To perform the functions for generation and operations in finite Extended Galois Field $GF(p^m)$ a class *ExtendedGaloisField* was created, which inherits the basic class *GaloisField*, and has the following additional features: Base Galois Field – *Base*, *Power*, primitive polynomial *ModuloPoly*, degrees of the primitive element *AlphaPowers*.

A screenshot of the program implemented in Visual Studio programming environment using the language C# is shown on Fig. 3. Example for results for generation of the field $GF(3^2)$ are presented in Table 2.



p: 3

m: 2

p(x): 2 + a + a^2

```
0 = 0 = 00
a^0 = 1 = 10
a^1 = a = 01
a^2 = 1 + 2a = 12
a^3 = 2 + 2a = 22
a^4 = 2 = 20
a^5 = 2a = 02
a^6 = 2 + a = 21
a^7 = 1 + a = 11
```

```
Cyclotomics modulo 8 using GF(3)
{0}   2 + x
{1, 3}   2 + x + x^2
{2, 6}   1 + x^2
{4}   1 + x
{5, 7}   2 + 2x + x^2
```

Figure 3. *Screenshot of the program Extended Galois Field GF*(p^m)

Performance evaluation of the suggested algorithm for $GF(p^m)$ generation is done by a comparison between the number of the operations for testing, multiplication and addition in Galois Field *GF*(*p*) when conducting the traditional and the accelerated search algorithms. The results of the comparative evaluation are presented in Table 3.

The following conclusion can be made from the analysis of the results. The suggested implementation lowers the volume of the random search, as well as the count of the multiplication and addition operations in *GF*(*p*) ***p*** times, i.e. the proposed algorithm is ***p*** times faster than the traditional.

Table 2. *Galois Field GF(3^2) with Primitive Polynomial $a^2 + a + 2$*

Representation by powers of the primitive element α	Representation by ternary polynomials	Cyclotomics	Cyclotomic polynomials	Primitive element
0	00 = 0			
α^0	10 = 1	{0}	2 + x	
α^1	01 = a	{1, 3}	$2 + x + x^2$	α
α^2	12 = 1 + 2a	{2, 6}	$1 + x^2$	1 + α
α^3	22 = 2 + 2a	{1, 3}	$2 + x + x^2$	α
α^4	20 = 2	{4}	1 + x	
α^5	02 = 2a	{5, 7}	$2 + 2x + x^2$	α
α^6	21 = 2 + a	{2, 6}	$1 + x^2$	1 + α
α^7	11 = 1 + a	{5, 7}	$2 + 2x + x^2$	α

Table 3. *Operations count for generation of Extended Galois Field GF(p^m)*

Field *GF*(p^m)	**Method**	**Checks**	**Multiplications in *GF*(*p*)**	**Additions in *GF*(*p*)**
GF(2^3)	traditional	8	8.3 = 24	8.3 = 24
	accelerated	4	4.3 = 12	4.3 = 12
GF(2^4)	traditional	16	16.4 = 64	16.4 = 64
	accelerated	8	8.4 = 32	8.4 = 32
GF(3^3)	traditional	27	27.3.2 = 162	27.3.2 = 162
	accelerated	9	9.3.2 = 54	9.3.2 = 54
GF(3^4)	traditional	81	81.4.2 = 648	81.4.2 = 648
	accelerated	27	27.4.2 = 216	27.4.2 = 216
GF(5^2)	traditional	25	25.2.4 = 200	25.2.4 = 200
	accelerated	5	5.2.4 = 40	5.2.4 = 40
GF(5^3)	traditional	125	125.3.4 = 1500	125.3.4 = 1500
	accelerated	25	25.3.4 = 300	25.3.4 = 300

4 Conclusion

The proposed algorithms for generation and operations in the Extended Galois Field *GF(p^m)* are with higher performance than the traditional ones. They can be successfully implemented to ensure a low error level in communication channels when linear error correcting codes such as *BCH* or *RS* are generated. They can be used for generation of long pseudo-random sequences for encryption of the information transmitted in communication channels.

References

[1] E.R. Berlekamp, *Algebraic coding theory*. McGraw-Hill, 1968.

[2] G. Birkhoff, S. Mac Lane, *A survey of modern algebra*. Universities Press, 1965.

[3] R.C. Bose, D.K. Ray-Chaudhuri, On a class of error correcting binary group codes, *Information control,* 3, (1960) 279-290.

[4] C.G. Clark Jr., J.B. Cain, *Error-correction coding for digital communications*, Springer Science & Business Media, 2013.

[5] Y. Li, H. Wang, J. Zhao, On the Primitivity of some Trinomials over Finite Fields. *IACR Cryptology ePrint Archive* 2013 (2013): 252.

[6] R. Lidl, H. Niederreiter, *Finite fields,* Vol. 20, Cambridge university press, 1997.

[7] J.L. Massey, *Coding theory*, In W. Lederman and S. Vajda, editors, Handbook of Applicable Mathematics, Vol. 5, Combinatorics and Geometry, Chichester and New York: Wiley, (1985) 623-676.

[8] J.L. Massey, O. N. Garcia, Error-correcting codes in computer arithmetic, In J. Tou, editor, Advances in Information Systems Science, Vol. 4, New York: Plenum Press, (1972) 273-326.

[9] J.L. Massey, Some applications of coding theory in cryptography. In P. G. Farrell, editor, Codes and Cyphers: Cryptography and Coding IV, Essex, England, Formara Ltd. (1995) 33-47.

[10] W.W. Peterson, E.J. Weldon, *Error-correcting codes*. MIT press, 1972.

[11] I.S. Reed, G. Solomon, Polynomial codes over certain finite fields, J Soc. Ind. Appl. Match, 8, (1960) 300-304.