Implementación de aritmética de torres de campos finitos binarios de extensión 2

Implementation of binary finite fields towers of extension 2

Fabián Velásquez[1]
Javier F. Castaño[2]

Fecha de envío: marzo 2013
Fecha de recepción: marzo 2013
Fecha de aceptación: mayo 2013

Resumen

En el presente trabajo se muestran los aspectos básicos de la aritmética de campos finitos binarios GF(2^m) extendidos, usando el concepto de torres de campos GF(2^{2m}), en este caso con extensión 2 o cuadrática. El uso de torres de campos agiliza el cómputo de operaciones en los campos finitos, lo cual es aplicado en el cálculo de emparejamientos bilineales, parte fundamental de la criptografía basada en identidad. Se presentan los conceptos básicos de aritmética en GF(2^m) y la construcción de las operaciones suma y multiplicación en campos binarios extendidos. De igual manera, se presentan los resultados de la implementación en un dispositivo FPGA XV5LX110T de Xilinx Inc., desarrollada usando lenguaje VHDL y la herramienta ISE Design Suite System Edition 13.4.

Palabras clave:
Torres de campos, aritmética de campos finitos, campos de Galois, aritmética computacional

Abstract

The present work shows the basics of arithmetic of binary finite fields GF (2^m), using the concept of extended towers of fields GF (2^{2m}), in this case with quadratic extension. Using field towers improve the computation of operations over finite fields, which is applied in the calculation of bilinear pairings, a main part of the identity-based cryptography; we present the basic concepts of arithmetic in GF (2^m) and construction of operations addition, multiplication and multiplicative inverse in extended binary fields. Similarly presents the results of the implementation in a Xilinx FPGA device XV5LX110T, developed using VHDL language and tool ISE Design Suite System Edition 13.4.

Keywords:
Field tower, finite field arithmetic, Galois field, computational arithmetic

1. Introducción

En la actualidad la aplicación de matemática discreta, en especial de aritmética de campos

1 Ingeniero electrónico, Universidad de los Llanos, Colombia. MSc. (c) en Matemáticas Aplicadas, Universidad Eafit, Colombia. Profesor investigador, Grupo de Investigación Macrypt - Universidad de los Llanos, Colombia. Correo electrónico: *fvelasquez@unillanos.edu.co*

2 Ingeniero electrónico, Universidad de los Llanos, Colombia. Especialista en Diseño y Construcción de Soluciones Telemáticas, Universidad Autónoma de Colombia, Colombia. Profesor investigador, Grupo de Investigación Macrypt - Universidad de los Llanos, Colombia. Correo electrónico:*jfcastano@unillanos.edu.co*

finitos, es bastante amplia, principalmente en criptografía y codificación, técnicas de gran importancia en la tecnología moderna [1].

La criptografía es un área que se basa en algoritmos que, a su vez, se encuentran soportados en la aplicación, por ejemplo, de la teoría de Galois, la teoría de curvas modulares o el álgebra abstracta en general. La teoría de campos finitos proporciona operaciones en estructuras discretas, las cuales encajan perfectamente en los objetivos y necesidades de la criptografía. Para profundizar sobre el tema base remitimos al lector, por ejemplo, a [2].

De manera permanente, se presentan avances en esta área en dos frentes: la base matemática y la aplicación computacional. Uno de estos avances es la criptografía basada en identidad, que proporciona soluciones a diferentes problemáticas que posee el paradigma más usado hoy en día: la criptografía de clave pública.

La criptografía basada en identidad se encuentra totalmente definida en campos finitos, curvas modulares y, sobre todo, en la aplicación de *torres de campos*, estructura esta que permite precisamente construir de manera eficiente los emparejamientos bilineales, de cuyas propiedades surge la posibilidad de construir el concepto de identidad y su aplicación en criptografía.

En este trabajo se explican los conceptos básicos de aritmética en campos finitos y la construcción de torres de campos, se muestran los algoritmos requeridos y su enfoque computacional y, adicionalmente, se presentan los resultados de una implementación en FPGA de las operaciones suma, multiplicación e inverso multiplicativo en una torre de campo $GF(2^{2m})$. Estos resultados hacen parte de un proyecto global que busca la implementación funcional de criptografía basada en identidad en dispositivos FPGA, para su uso en diferentes entornos.

2. Estructuras algebraicas

Una estructura algebraica es un conjunto dotado de una o más operaciones binarias que cumple ciertas propiedades. Si el conjunto tiene finitos elementos se dice que la estructura algebraica tiene orden finito, de lo contrario tiene orden infinito. Entre las estructuras algebraicas más utilizadas en los algoritmos criptográficos están los grupos y los campos.

2.1. Grupos

Un conjunto G, dotado de una operación binaria*, $\langle G,* \rangle$, que cumple las siguientes propiedades, tiene estructura algebraica de grupo.

i) Clausurativa: $a*b \in G, \ \forall a,b \in G$

ii) Asociativa: $a*(b*c)=(a*b)*c, \forall a,b,c \in G$

iii) Modulativa: $\exists e \in G | a*e=e*a=a, \ \forall a \in G*$

iv) Invertiva: $\exists a\text{-}1 \in G | a\text{-}1*a=e, \ \forall a \in G$

Si además de las propiedades anteriores se cumple la propiedad conmutativa, en la cual $a*b=b*a, \forall a,b \in G$, entonces el grupo es abeliano.

En las aplicaciones criptográficas solo interesan los grupos de orden finito y en los cuales sus operaciones se realizan módulo un numero primo *n*, o módulo un polinomio irreducible *f(x)*. Un subconjunto H de G que cumple las mismas propiedades de grupo se llama subgrupo y el orden de cualquier subgrupo de G divide al orden de G.

2.2. Anillos

Un anillo es un conjunto G dotado de dos

operaciones binarias *, Δ. De esta forma se genera una estructura algebraica ⟨G,*,Δ⟩ que cumple las siguientes condiciones:

i) La estructura algebraica ⟨G,*⟩ es grupo abeliano.

ii) La estructura ⟨G,Δ⟩ cumple las propiedades asociativa y clausurativa.

iii) El conjunto *G* dotado de las dos operaciones ⟨G,*,Δ⟩ cumple la propiedad distributiva.

2.3. Cuerpos

Un conjunto G dotado de dos operaciones * y Δ que cumple las siguientes condiciones tiene estructura algebraica de campo:

i) La estructura algebraica ⟨G,*⟩ es grupo abeliano.

ii) El conjunto G con la operación Δ es grupo abeliano.

iii) La estructura ⟨G,*,Δ⟩ cumple la propiedad distributiva a derecha e izquierda.

Los campos aplicados a la criptografía son aquellos que tienen orden finito, también llamados campos de Galois o campos finitos. La aritmética en estos campos permite, por ejemplo, las operaciones de suma y multiplicación de puntos racionales en la criptografía de curvas elípticas, ya que estos tienen estructura de grupo finito aditivo abeliano y las curvas elípticas se definen sobre campos de Galois [3]. La eficiencia, la velocidad y el espacio ocupado en un procesador por la aritmética de campos finitos constituyen un factor imprescindible en el momento de elegir un algoritmo para la implementación de cualquier operación en el campo. La investigación en esta área se encamina a lograr mayor eficiencia, tanto en la representación de los campos finitos como en la implementación de aritmética en estos campos, en *software* y en *hardware.*

3. Aritmética en campos finitos

3.1. Aritmética en campos finitos binarios GF(2^m)

3.1.1. Conceptos de bases polinomiales

Si se considera una extensión finita $F = F_{q^m}$ del campo finito $K=F_q$ como un espacio vectorial sobre *K*, entonces *F* tiene dimensión *m* sobre *K*, y si $(\alpha_1,\alpha_2,\ldots,\alpha_m)$ es un conjunto de m elementos que forman una base de *F* sobre *K*, entonces cada elemento $\alpha \in F$ se puede representar de forma única como: $\alpha = c_1\alpha_1 + \ldots + c_m\alpha_m$, donde $c_j \in K$, $1 \leq j \leq m$, de acuerdo con la noción tradicional de espacio vectorial y base de espacio vectorial.

Se puede establecer la siguiente correspondencia lineal de *F* en *K* mediante la función traza $Tr_{F/K}(\alpha)$: sea *K* un campo finito y tómese una extensión finita de *K*. La base polinomial $(1,\alpha,\alpha^2,\ldots,\alpha^{m-1})$ es construida con las potencias de un elemento definitorio α de *F* sobre *K*, donde el elemento α es un elemento primitivo de *F* Con esta base se representan los elementos de un campo finito mediante polinomios con coeficientes $a_i \in F_q$ que pertenecen al campo *K* y las potencias de los elementos de la base. Cada polinomio es una clase residual módulo el polinomio irreducible, es decir, $F_{q^m} = \left\{ a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m / a_i \in F_q \right\}$

3.1.2. Multiplicación

Dos elementos $A,B \in GF(2^m)$ con polinomio irreducible $p(x)$ son representados en la forma de polinomios:

$$A = a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^{m-1}$$

$$B = b_0 + b_1 x + b_2 x^2 + \ldots + b_m x^{m-1}$$

El producto $C = A*B$ se representa como:

$$C = c_0 + c_1 x + c_2 x^2 + \ldots + c_m x^{m-1}$$

De esta expresión puede notarse que el producto C tiene el mismo grado que los operandos *A* y *B*, lo cual se debe a que son elementos de un campo finito.

Para realizar la multiplicación entre dos elementos de un campo finito con representación en base polinomial, se realiza la multiplicación normal de polinomios, lo que origina un polinomio de grado máximo *2m-2*. Como puede observarse, este polinomio no pertenece al campo finito, por lo cual se implementa una reducción módulo el polinomio irreducible. Dado que la multiplicación de polinomios se puede realizar paso a paso, este tipo de multiplicadores se denominan seriales [4].

El otro enfoque es tomando el polinomio irreducible para generar una base, calcular las expresiones que generan cada uno de los coeficientes del resultado de la multiplicación, la cual por lo tanto se puede implementar en paralelo [4].

3.1.3. Adición en GF (2^m)

Dados dos elementos:

$$A = a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^{m-1}$$

$$B = b_0 + b_1 x + b_2 x^2 + \ldots + b_m x^{m-1}$$

En el campo finito $GF(2^m)$, entonces $A+B=C=c_0+c_1x+c_2x^2+\ldots+c_mx^m$, donde $c_i=(a_i+b_i) mod 2$, lo que equivale a la operación XOR bit a bit.

3.1.4. Inverso multiplicativo GF(2^m)

Para el caso del inverso multiplicativo se implementa el algoritmo de Itoh-Tsujii [5], el cual en forma recursiva realiza la exponenciación del valor al cual se le va a calcular el inverso multiplicativo. La expresión general para calcular el inverso multiplicativo, se deriva de la siguiente forma:

Dado un elemento $A \epsilon GF(2^m)$, existe un elemento A^{-1} tal que se cumple $AA^{-1} = e$, siendo e el elemento neutro de la operación producto, en este caso $e = 1 = z^0$.

La operación inversión es importante en sí misma y también para la implementación de la división, ya que:

$$\frac{A}{B} = AB^{-1} \text{ siempre y cuando } B \neq 0$$

Para esto es necesario calcular el inverso de B y luego realizar una multiplicación. La inversión es la operación más difícil de implementar en aritmética de campos finitos.

Existen dos tipos de algoritmos de inversión: basados el teorema extendido de Euclides y sus variantes y otros basados en multiplicaciones en el campo. Si se opta por un método basado el algoritmo extendido de Euclides, la complejidad computacional es bastante alta, lo que impacta directamente en el diseño. Si se opta por un método basado en multiplicaciones, se utiliza la función de multiplicación reduciendo la complejidad computacional.

La idea principal del inversor implementado radica en el "pequeño Teorema de Fermat", el cual indica que para todo elemento no cero en un campo F_2^m

$$a^{-1} = a^{2^m - 2}$$

Esto se demuestra ya que por las propiedades de los campos F_2^m

$$a^{2^m} = a$$

y haciendo una sustitución se llega a la expre-

sión anterior.

La expresión puede rescribirse como

$$a^{-1} = a^{2^{m-2}} = (a^{2^{m-1}-1})^2$$

De esta manera, se puede reducir la complejidad de la inversión a un número menor de elevaciones al cuadrado.

Itoh y Tsujii [5] propusieron una ingeniosa manera de reducir aún más la complejidad de la operación, haciendo una nueva sustitución de tal forma que se obtienen las fórmulas:

$$a^{2^{m-1}-1} = \begin{cases} (a^{2^{\frac{m-1}{2}}-1})^{2^{\frac{m-1}{2}}} * a^{2^{\frac{m-1}{2}}-1} \\ a * (a^{2^{m-2}-1})^2 \end{cases}$$

El primer caso es para *m* impar y el otro para *m* par.

3.2. Aritmética en torres de campos

Las torres de campos surgen a partir del trabajo de Baktir y Sunar [6], quienes aportaron una forma eficiente de abordar la representación y las operaciones en campos finitos.

Una torre de campo es la extensión a su vez de un campo de extensión $F_p / f(x)$ con característica p, es decir, un campo primo F_p que ha sido extendido módulo un polinomio irreducible $f(x)$ de grado n. El campo de extensión $F_p / f(x)$ se extiende módulo otro polinomio irreducible $g(x)$ de grado n/m, generando un campo con un número mayor de elementos. La representación en torres de campos permite la ejecución de las operaciones suma y multiplicación en forma más eficiente, las cuales son ideales para la implementación de emparejamientos bilineales que presentan una alta complejidad computacional por causa del número elevado de operaciones en el campo de extensión donde se representan los polinomios en forma binaria.

La extensión cuadrática (de orden 2) del campo finito $GF(2^m)$ está representada como:

$$GF(2^{2m}) \approx GF(2^m) / x^2 + x + 1 \quad (3.1)$$

En este caso se realiza una extensión 2 del campo $GF(2^m)$, que da como resultado la torre de campo $GF((2^m)^2)$ y, finalmente, $GF(2^{2m})$. De esta manera, un campo finito de característica 2, con m par, puede ser expresado en forma de torre de campo. Por ejemplo, el campo finito $GF(2^8)$ puede ser expresado como la torre de campo $GF((2^4)^2)$, reduciendo la complejidad de la aritmética de $GF(2^8)$ a $GF(2^4)$.

3.2.1. Suma en $GF(2^{2m})$

Dados dos elementos *a* y *b* que pertenecen a $GF(2^{2m})$, se calcula $a + b$ aplicando el algoritmo siguiente, el cual requiere dos sumas en $GF(2^m)$.

Entrada: $a = (a_0 + a_1 u), b = (b_0 + b_1 u) \in GF(2^{2m})$

Salida: $c = a + b \in GF(2^{2m})$

1: $c_0 \leftarrow a_0 \oplus b_0$;

2: $c_1 \leftarrow a_1 \oplus b_1$;

3: return $c = c_0 + c_1 u$;

Algoritmo 3.1 Suma en $GF(2^{2m})$ *[7]*

3.2.2. Multiplicación en torres de campos

La multiplicación de dos elementos *a,b* que pertenecen a $GF(2^{2m})$, definida como $(a_0+a_1u)(b_0+b_1u)=(a_0b_0+a_1b_1\beta)+(a_0b_1+a_1b_0)u$ es implementada eficientemente, usando el método de Karatsuba-Ofman, en donde $a_0b_1+a_1b_0=(a_0+a_1)(b_1+b_0)-a_0b_0-a_1b_1$

Esto se realiza mediante el algoritmo 3.2.

Entrada: $a = (a_0 + a_1 u), b = (b_0 + b_1 u) \in GF(2^{2m})$

Salida: $c = a.b \in GF(2^{2m})$

1: $v_0 \leftarrow a_0 \oplus b_0$;

2: $v_1 \leftarrow a_1 \oplus b_1$;

3: $c_0 \leftarrow v_0 \oplus \beta v_1$;

4: $c_1 \leftarrow (a_0 \oplus a_1) \otimes (b_0 \oplus b_1) - v_0 - v_1$;

3: return $c = c_0 + c_1 u$;

Algoritmo 3.2. Multiplicación en *GF(2²ᵐ)* [7: 18]

El algoritmo 3.2 requiere en total tres multiplicaciones y cinco sumas en *GF(2ᵐ)*, así como un producto entre un elemento $a_0 \in GF(2^m)$ por la constante β de la ecuación (3.1).

3.2.3. Inverso multiplicativo en GF(2²ᵐ)

El cálculo del inverso se realiza mediante el algoritmo 3.3, el cual requiere una inversión, tres multiplicaciones, dos sumas y una elevación al cuadrado en GF(2^m). Sea un elemento

$$U = (u_0 + u_1 s) \in GF(2^{2m}), U \neq 0$$

$$u_1, u_0 \in GF(2^m)$$

Su inverso multiplicativo se define como

$$V = (v_0 + v_1 s) \in GF(2^{2m})$$

$$v_1, v_0 \in GF(2^m)$$

Dado que UV = 1, se tiene que

$$u_0 v_0 + u_1 v_1 = 1$$

$$u_0 v_1 + u_1 v_0 + u_1 v_1 = 0$$

La solución de este sistema de ecuaciones, es

$$v_0 = w^{-1}(u_0 + u_1)$$

$$v_1 = w^{-1} u_1,$$

Donde

$$w = u_0^2 + (u_0 u_1) \in GF(2^m)$$

Entrada: $U = (u_0 + u_1 s) \in GF(2^{2m}), U \neq 0$

Salida: $V = U^{-1} = v_0 + v_1 s \in GF(2^{2m})$

1: $a_0 \leftarrow u_0 + u_1$;

2: $m_0 \leftarrow u_0^2; m_1 \leftarrow a_0 u_1$;

3: $a_1 \leftarrow m_0 + m_1$;

4: $i_0 \leftarrow a_1^{-1}$;

5: $v_0 \leftarrow a_0 i_0$;

6: $v_1 \leftarrow u_1 i_0$;

5: return $V = v_0 + v_1 s$;

Algoritmo 3.3. Inverso multiplicativo en *GF(2²ᵐ)*

4. Resultados

La implementación se realizó sobre un dispositivo XV5LX110T de Xilinx, usando una tarjeta de desarrollo XUPV5 de Digilent. Para el desarrollo y simulación se usó el entorno ISE Design Suite 10.1 de Xilinx y descripción en VDHL.

En la tabla 1 se presentan los resultados de la implementación, especificando la ocupación de área para cada módulo desarrollado para aritmética en *GF((2ᵐ)²)*, con m = 239. Se especifican los resultados de la simulación *post-place and route*, realizada con el software ISE Design Suite 13.4 System Edition de la empresa Xilinx. Los resultados se expresan en términos

de área ocupada en slices y frecuencia máxima de operación. Para cada caso la síntesis se realizó eligiendo la optimización respectiva, por área o por velocidad, como parámetro definido en el *software* de desarrollo. Por lo tanto, se presentan los mejores resultados posibles con dicha herramienta en ambos aspectos.

Para esta implementación se eligió un campo finito GF(2^{239}), extendido en forma de torre de campo GF($2^{2(239)}$). Las operaciones se realizan empleando los algoritmos 3.1, 3.2 y 3.3 en la torre de campo, usando operaciones suma, producto e inverso multiplicativo en el campo base GF(2^{239}). La operación suma en el campo base se implementó eficientemente mediante la operación XOR bit a bit, mientras que la multiplicación en el campo se implementó mediante el algoritmo bit-serial, que requiere m ciclos de reloj para obtener el resultado final, [4]. El cálculo del inverso multiplicativo se implementó mediante el algoritmo de Itoh-Tsujii. Se tomaron como base los módulos de aritmética de campos finitos desarrollados por el grupo Macrypt de la Universidad de los Llanos.

5. Conclusiones

El concepto de torres de campos ha cobrado enorme importancia recientemente, debido a su gran utilidad, principalmente en el cálculo de emparejamientos bilineales, operación fundamental para la criptografía basada en identidad. Este nuevo paradigma, con fuerza avanza como la solución real a los múltiples problemas de la criptografía de clave pública.

Por esta razón es muy importante entender e implementar la aritmética en torres de campos en forma eficiente, en entornos como sistemas embebidos, microcontroladores y dispositivos FPGA.

Se inició el trabajo en el entendimiento e implementación de este conjunto de técnicas, en este caso en dispositivos FPGA. Se logró evidenciar que, efectivamente, la aplicación de torres de campos reduce la complejidad de la aritmética en campos finitos, siempre y cuando se pueda escribir el campo finito requerido como una extensión cuadrática de un campo más pequeño.

6. Trabajo futuro

Como trabajo a corto plazo se tiene definida la implementación de otras operaciones en campos finitos, que se requieren para la implementación de emparejamientos bilineales, en este caso el cálculo de exponenciaciones, raíces cuadradas y cúbicas, así como la solución de ecuaciones cuadráticas. Esto enmarcado en el proyecto general que tiene como objetivo la implementación completa de criptografía basada en identidad en dispositivos FPGA para su aplicación en diferentes entornos como herramienta de seguridad informática. En el conocimiento de los autores de este trabajo, revisado el estado del arte en Colombia, puede afirmarse que este es el primer referente en el área de aritmética de torres de campos y de implementación de criptografía basada en identidad.

Tabla 1. Resultados obtenidos

Operación GF(22(239))	Área [slices]	Frecuencia máx. [MHz]
Suma	240	180
Multiplicación	850	150
Elevación al cuadrado	420	120
Inverso multiplicativo	1450	90

Reconocimientos

Los autores expresan sus reconocimientos a la Dirección General de Investigaciones de la Universidad de los Llanos, la cual financia el proyecto de investigación "Implementación en FPGA de criptografía basada en identidad", en el marco del cual se obtienen los resultados presentados en este trabajo.

Referencias

[1] M. Merino, *Una introducción a la criptografía. El Criptosistema R.S.A*, I.E.S Cardenal López de Mendoza, 2004.

[2] C. Ivorra C., *Teoría de cuerpos de clases* [en línea], Universidad de Valencia, 2008, disponible: *www.uv.es*/~ivorra/Libros/*Cuerpos.pdf*

[3] L. Fuentes, *Estudio y análisis de emparejamientos bilineales definidos sobre curvas ordinarias con alto nivel de seguridad*, tesis de Maestría en Ciencias de la Computación, Centro de Investigaciones y Estudios Avanzados Cinvestav, Instituto Politécnico Nacional, Unidad Zacatengo, México D.F., 2011.

[4] D. Hankerson, A. Menezes y S. Vanstone, *Guide to elliptic curve cryptography*, Springer-Verlag, 2004.

[5] T. Itoh y S. Tsujii, "A fast algorithm for computing multiplicative inverses in GF(2^m) using nomal bases"*, Information and Computation,* no. 78, pp. 171-17, 1988.

[6] S. Baktir y B. Sunar, "Optimal tower fields", IEEE trans, *Comput. 53*, no. 10, pp. 1231-43, 2004.

[7] N. Cortez, *Multiplicadores de arquitectura segmentada y su aplicación al cómputo de emparejamientos bilineales,* tesis de Maestría en Ciencias de la Computación, Centro de Investigaciones y Estudios Avanzados Cinvestav, Instituto Politécnico Nacional, Unidad Zacatengo, México D. F., 2009.