Redundant Trinomials for Finite Fields of Characteristic 2

Christophe Doche

Division of ICS, Department of Computing
Macquarie University, NSW 2109 Australia
doche@ics.mq.edu.au

Abstract. In this article we introduce *redundant trinomials* to represent elements of finite fields of characteristic 2. This paper develops applications to cryptography, especially based on elliptic and hyperelliptic curves. After recalling well-known techniques to perform efficient arithmetic in extensions of $\mathbb{F}_2$, we describe redundant trinomial bases and discuss how to implement them efficiently. They are well suited to build $\mathbb{F}_{2^n}$ when no irreducible trinomial of degree n exists. Depending on $n \in [2, 10000]$ tests with `NTL` show that, in this case, improvements for squaring and exponentiation are respectively up to 45% and 25%. More attention is given to extension degrees relevant for curve-based cryptography. For this range, a scalar multiplication can be sped up by a factor up to 15%.

1 Introduction

Although this is the first time *redundant trinomials* are used in cryptography, Brent and Zimmermann introduced the similar concept of *almost irreducible trinomials* in the context of random number generators in [2,3]. In particular, they have shown that there exist almost irreducible trinomials of degree n for every $n \in [2, 10000]$ and explained how to compute efficiently with them.

We discovered the concept of *redundant trinomial*, designed an algorithm to find them, and searched for efficient arithmetic independently. Then Brent and Zimmermann pointed out that some of this work was already contained in [2]. Additionally, this paper provides a careful analysis of the best algorithm to use to compute an inverse in a redundant trinomial basis depending on the extension degree chosen. Also, we implemented our ideas and give a precise comparison of running times between redundant trinomials and irreducible pentanomials, focused on extension degrees of cryptographic interest. Tests reveal that a certain class of redundant trinomial, called *optimal redundant trinomials*, give even better results. This leads us to introduce *optimal redundant quadrinomials* that can outclass irreducible pentanomials and even trinomials.

At present, let us recall basic facts on fields of characteristic 2. There are mainly two types of bases to compute in finite fields of characteristic 2, namely polynomial and normal bases. It is well known that there is a normal basis of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$ for every extension degree n. However only a certain category of normal

C. Boyd and J.M. González Nieto (Eds.): ACISP 2005, LNCS 3574, pp. 122–133, 2005.

bases, namely optimal normal basis of type I or II can be used in practice. Those bases are quite rare. Considering extension fields of degree up to $10,000$, only 17.07% of them have an optimal normal basis.

For every extension degree, there is a polynomial basis as well. Sparse irreducible polynomials are commonly used to perform arithmetic in extension fields of $\mathbb{F}_2$ since they provide a fast modular reduction. As a polynomial with an even number of terms is always divisible by $x+1$, we turn our attention to so-called *trinomials*. When no such irreducible polynomial exists, one can always find an irreducible *pentanomial*, at least for extension degrees up to $10,000$. In this range this situation occurs quite often. In fact one has to choose an irreducible pentanomial in about 50% of the cases (precisely 4853 out of 9999 [11]).

The next section describes in more detail efficient algorithms to perform reduction, addition, multiplication, and inversion in $\mathbb{F}_{2^n}/\mathbb{F}_2$.

2 Finite Field Arithmetic

Let $\mu(x)$ be an irreducible polynomial of degree n over $\mathbb{F}_2$. An element of $\mathbb{F}_{2^n} \simeq \mathbb{F}_2[x]/\big(\mu(x)\big)$ is uniquely represented as a polynomial f of degree less than n with coefficients in $\mathbb{F}_2$. If f is a polynomial such that $\deg f \geqslant n$ one first reduces f modulo the irreducible polynomial μ. The usual way to get this reduction is to compute the remainder of the Euclidean division of f by μ. When μ is sparse there is a dedicated algorithm which is much faster [7].

Algorithm 1. Division by a sparse polynomial

INPUT: Two polynomials $\mu(x)$ and $f(x)$ with coefficients in a commutative ring, where $\mu(x)$ is the sparse polynomial $x^n + \sum_{i=1}^{t} a_i x^{b_i}$ with $b_i < b_{i+1}$.
OUTPUT: The polynomials u and v such that $f = u\mu + v$ with $\deg v < n$.

1. $v \leftarrow f$ and $u \leftarrow 0$
2. **while** $\deg(v) \geqslant n$ **do**
3. $\quad k \leftarrow \max(n, \deg v - n + b_t + 1)$
4. $\quad$ write $v(x) = u_1(x)x^k + w(x)$
5. $\quad v(x) \leftarrow w(x) - u_1(x)\big(\mu(x) - x^n\big)x^{k-n}$
6. $\quad u(x) \leftarrow u_1(x)x^{k-n} + u(x)$
7. **return** (u, v)

Remarks.

- *If* $\deg f = m$ *then Algorithm 1 needs at most* $2(t-1)(m-n+1)$ *field additions to compute* u *and* v *such that* $f = u\mu + v$. *In this case the number of loops is at most* $\lceil (m-n+1)/(n-b_t-1) \rceil$. *If* $m \leqslant 2n-2$, *as is the case*

when performing arithmetic modulo μ, then the number of loops is at most equal to 2 as long as $1 \leqslant b_t \leqslant n/2$.
- *To avoid computing the quotient u when it is not required, simply discard Line 6 of Algorithm 1.*

Concerning operations, additions are performed at a word level and correspond to XOR. Computing a squaring only costs a reduction modulo f. Indeed if $f(x) = \sum a_i x^i$ then $f^2(x) = \sum a_i x^{2i}$. Multiplications are also performed at a word level, but processors do not provide single precision multiplication for polynomials. Nevertheless it is possible to emulate it doing XOR and shifts. One can also store all the possible single precision products and find the global result by table lookup. This method is fast but for 32-bit words the number of precomputed values is far too big. A tradeoff consists in precomputing a smaller number of values and obtaining the final result with Karatsuba's method. Typically two 32-bit polynomials can be multiplied with 9 precomputed multiplications of 8-bit block polynomials [6].

Once the single precision multiplication is defined, different multiplication methods can be applied depending on the degree of the polynomials. In [7] the crossover between the schoolbook multiplication and Karatsuba's method is reported to be equal to 576. Other more sophisticated techniques like the F.F.T. or Cantor's multiplication [6] based on evaluation/interpolation methods can be used for larger degrees. For example, the crossover between Karatsuba's method and Cantor's multiplication is equal to 35840 in [7].

There are usually two different ways to compute the inverse of an element of $\mathbb{F}_{2^n}$. The first one is to compute an extended Euclidean GCD. The second one takes advantage of the group structure of $\mathbb{F}_{2^n}^*$ and computes α^{-1} as α^{2^n-2}.

3 Redundant Trinomials

With Algorithm 1, the product of two elements in $\mathbb{F}_{2^n}$ can be reduced with at most $4(n-1)$ elementary operations using trinomials and at most $8(n-1)$ operations using pentanomials.

For some extension degree there is an even better choice, namely *all one polynomials*. They are of the form

$$\mu(x) = x^n + x^{n-1} + \cdots + x + 1. \tag{1}$$

Such a $\mu(x)$ is irreducible if and only if $n+1$ is prime and 2 is a primitive element of $\mathbb{F}_{n+1}$. This occurs for 470 values of n up to $10,000$, but n has to be even.

It is clear from the definition of $\mu(x)$ that $\mu(x)(x+1) = x^{n+1} + 1$. Thus an element of $\mathbb{F}_{2^n}$ can be represented on the *anomalous basis* $(\alpha, \alpha^2, \ldots, \alpha^n)$ where α is a root of $\mu(x)$. In other words an element of $\mathbb{F}_{2^n}$ is represented by a polynomial of degree at most n with no constant coefficient, the unity element 1 being replaced by $x + x^2 + \cdots + x^n$.

The reduction is made modulo $x^{n+1} + 1$ and a squaring is simply a permutation of the coordinates. In one sense computations in $\mathbb{F}_{2^n}$ are performed in the

ring $\mathbb{F}_2[x]/(x^{n+1}+1)$. Unfortunately this very particular and favorable choice does not apply very well to odd degrees. When n is odd, one can always embed $\mathbb{F}_{2^n}$ in a cyclotomic ring $\mathbb{F}_2[x]/(x^m+1)$. But $m \geqslant 2n+1$ so that the benefits obtained from a cheap reduction are partially obliterated by a more expensive multiplication [13]. Note that for elliptic and hyperelliptic curve cryptography only prime extension degrees are relevant [5,8,10].

We now adopt this idea and transfer it to the setting of polynomial bases. When there is no irreducible trinomial for some extension degree n one can try to find a trinomial $t(x) = x^m + x^k + 1$ with m slightly bigger than n such that $t(x)$ admits an irreducible factor $\mu(x)$ of degree n. Such a trinomial is called a *redundant trinomial*. The idea is then to embed $\mathbb{F}_{2^n} \simeq \mathbb{F}_2[x]/\big(\mu(x)\big)$ into $\mathbb{F}_2[x]/\big(t(x)\big)$. From a practical point of view an element of $\mathbb{F}_{2^n}$ is represented on the redundant basis $1, \alpha, \ldots, \alpha^{m-1}$ where α is a root of $\mu(x)$ and the computations are reduced modulo $t(x)$. As $\mu(x)$ divides $t(x)$, one can reduce modulo $\mu(x)$ at any time and obtain consistent results. If $m-n$ is sufficiently small then the multiplication of two polynomials of degree less than m has the same cost as the multiplication of two polynomials of degree less than n, since multiplications are performed at a word level.

To reduce the results one needs at most 2 iterations using Algorithm 1 since one can always choose $t(x) = x^m + x^k + 1$ such that $k \leqslant \lfloor m/2 \rfloor$. Indeed if $k > \lfloor m/2 \rfloor$ the reciprocal polynomial of $t(x)$ can be considered instead.

However with these settings, the expression of a field element is no longer unique, but the result can of course be reduced modulo $\mu(x)$, when it is required. Note that it is possible to perform a fast reduction modulo $\mu(x)$ knowing only $t(x)$ and $\delta(x) = t(x)/\mu(x)$. The same kind of idea provide a quick way to test if two polynomials represent the same field element. Finally, one examines how inversion algorithms behave with this representation.

These topics are discussed in the next section.

4 Efficient Implementation of Redundant Trinomials

To reduce a polynomial $f(x)$ modulo $\mu(x)$ one could perform the Euclidean division of $f(x)$ by $\mu(x)$, but this method has a major drawback. It obliges to determine or to know $\mu(x)$ which is not sparse in general. A better idea is to write $f(x) = q(x)\mu(x) + r(x)$. Then $f(x)\delta(x) = q(x)t(x) + r(x)\delta(x)$ so that

$$f(x) \bmod \mu(x) = \frac{f(x)\delta(x) \bmod t(x)}{\delta(x)}. \tag{2}$$

The last division is exact and can be obtained by an Algorithm easily derived from Jebelean's one for integers [9]. Now two elements $f_1(x)$ and $f_2(x)$ correspond to the same element in $\mathbb{F}_{2^n}$ if and only if $\mu(x) \mid \big(f_1(x) + f_2(x)\big)$. This implies that $t(x) \mid \delta(x)\big(f_1(x) + f_2(x)\big)$. We could compute an exact division but there is a more efficient way to proceed. First note that if $f_1(x)$ and $f_2(x)$ are both of degree at most $m-1$ then

$$\deg\big(\delta(x)\big(f_1(x) + f_2(x)\big)\big) \leqslant 2m - n - 1. \tag{3}$$

So the quotient $q(x)$ of the division of $\delta(x)\big(f_1(x)+f_2(x)\big)$ by $t(x)=x^m+x^k+1$ is of degree at most $m-n-1$. Writing the division explicitly we see that if

$$m-k>m-n-1 \tag{4}$$

then $q(x)$ is equal to the quotient of the division of $\delta(x)\big(f_1(x)+f_2(x)\big)$ by x^m. This is just a shift and it is a simple matter to determine if $\delta(x)\big(f_1(x)+f_2(x)\big)$ is equal to $q(x)(x^m+x^k+1)$ or not.

Now one can check, *cf.* [4], that all the redundant trinomials found for n up to $10,000$ satisfy $m-k>m-n-1$.

Concerning inversion, it is clear that the algorithm based on Lagrange's theorem works without any problem with redundant polynomials. One must be careful with the extended GCD algorithm. Let $\alpha\in\mathbb{F}_{2^n}$ be represented by $f(x)$. When the algorithm returns u and v such that

$$f(x)u(x)+t(x)v(x)=1 \tag{5}$$

then the inverse of α is given by $u(x)$. But one could have

$$f(x)u(x)+t(x)v(x)=d(x) \tag{6}$$

with $\deg d(x)>0$. In this case two possibilities arise. If $\mu(x)\mid d(x)$, which can be checked by looking at the degree of $d(x)$, then $\alpha=0$. Otherwise $d(x)\mid\delta(x)$ and the inverse of α is given by $u(x)e(x)$ where $e(x)$ is the inverse of $d(x)$ modulo $\mu(x)$. Nevertheless there is a more simple technique. Indeed, as we will see, $t(x)$ is squarefree. So the gcd of $f(x)\delta(x)$ and $t(x)$ is equal to $\delta(x)$ and

$$f(x)\delta(x)u_1(x)+t(x)v_1(x)=\delta(x) \tag{7}$$

so that

$$f(x)u_1(x)+\mu(x)v_1(x)=1 \tag{8}$$

and the inverse of $f(x)$ is directly given by $u_1(x)$. The degree of $\delta(x)$ is usually much smaller than the degree of $e(x)$. So the multiplication is faster. No reduction modulo $t(x)$ is required at the end. It is not necessary to compute or precompute anything new. Even when $\gcd\big(f(x),t(x)\big)=1$ this last technique works. So one can either compute the extended $\gcd\big(f(x),t(x)\big)$, test its value and compute the extended $\gcd\big(f(x)\delta(x),t(x)\big)$ if necessary, or always perform only this last computation. The tradeoff in time depends on the number of irreducible factors of δ and the cost of a modular multiplication. Indeed the degree and the number of factors of $\delta(x)$ determine the probability that a random polynomial is prime to $t(x)$. If $\delta(x)$ is irreducible of degree r then this probability is clearly equal to $1-1/2^r$. If $\delta(x)$ has two factors of degree r_1 and r_2, necessarily distinct since $t(x)$ is squarefree, the probability becomes $1-1/2^{r_1}-1/2^{r_2}+1/2^{r_1+r_2}$. By induction, if $\delta(x)$ has ℓ distinct factors of degree $r_1,r_2,\ldots,r_\ell$ then the probability that $t(x)=x^m+x^k+1$ is prime to a random polynomial of degree less than m is

$$1\quad-\sum_{\substack{n=1\\ 1\leqslant i_1<\cdots<i_n\leqslant\ell}}^{\ell}\frac{(-1)^n}{2^{r_{i_1}+\cdots+r_{i_n}}}. \tag{9}$$

Note that $\delta(x)$ is irreducible in about 95% of the cases, *cf.* Section 6.

5 Example

Let us consider $\mathbb{F}_{2^8}$. There is no trinomial of degree 8 irreducible over $\mathbb{F}_2$. Instead one usually chooses the irreducible pentanomial $p(x) = x^8 + x^4 + x^3 + x + 1$. Nevertheless it is easily seen that $t(x) = x^{11} + x^5 + 1$ splits as $\mu(x)$ times $\delta(x)$ where $\mu(x) = x^8 + x^6 + x^5 + x^4 + x^2 + x + 1$ and $\delta(x) = x^3 + x + 1$ are both irreducible. The explicit expression of $\mu(x)$ is not important. In fact $t(x)$ and $\delta(x) = x^3 + x + 1$ are enough to compute in $\mathbb{F}_{2^8}$.

Let $f(x)$ and $g(x)$ be two polynomials of degree 7, namely

$$f(x) = x^7 + x^6 + x^2 + x + 1$$

and

$$g(x) = x^7 + x^6 + x^3 + x^2 + x + 1. \tag{10}$$

The product of $f(x)$ and $g(x)$ reduced modulo $t(x)$ is $h(x) = x^{10} + x^9 + x^8 + x^6 + x^5 + x^2 + x + 1$, whereas it is equal to $x^6 + x^4 + x^2 + 1$ modulo $\mu(x)$. Of course $h(x) \equiv x^6 + x^4 + x^2 + 1 \pmod{\mu(x)}$ but there is no need to reduce $h(x)$ at this stage.

Now let us compute the inverse of $f(x)$ and $g(x)$. Using an extended GCD algorithm. One obtains

$$f(x)(x^9 + x^8 + x^7 + x^4 + x^2 + x + 1) + t(x)(x^5 + x^2) = 1 \tag{11}$$

and

$$g(x)(x^4 + x^3 + x^2 + x) + t(x) = x^3 + x + 1. \tag{12}$$

We conclude immediately that the inverse of $f(x)$ is

$$f(x)^{-1} \equiv x^9 + x^8 + x^7 + x^4 + x^2 + x + 1 \pmod{t(x)}. \tag{13}$$

For the inverse of $g(x)$ one can first multiply $g(x)$ with $\delta(x)$ and compute an extended Euclidean GCD again. We get

$$g(x)\delta(x)(x^6 + x^5 + x^2 + 1) + t(x)(x^5 + x^2 + x) = x^3 + x + 1 \tag{14}$$

so that

$$g(x)^{-1} \equiv x^6 + x^5 + x^2 + 1 \pmod{t(x)}. \tag{15}$$

Using Lagrange's theorem, one gets directly

$$f(x)^{-1} \equiv f(x)^{2^8-2} \equiv x^3 + x^2 + x \pmod{t(x)} \tag{16}$$

and

$$g(x)^{-1} \equiv g(x)^{2^8-2} \equiv x^{10} + x^9 + x^5 \pmod{t(x)}. \tag{17}$$

The results are different representations of the same elements. If one wants to check it out, for example for the inverse of $f(x)$, it is enough to compute

$$(x^3 + x + 1)\big((x^9 + x^8 + x^7 + x^4 + x^2 + x + 1 + x^3 + x^2 + x) + (x^3 + x^2 + x)\big) \tag{18}$$

which is equal to $x^{12} + x^{11} + x^6 + x^5 + x + 1$ and test if this polynomial is a multiple of $t(x)$. If so the quotient must be $x + 1$ and indeed

$$(x + 1)(x^{11} + x^5 + 1) = x^{12} + x^{11} + x^6 + x^5 + x + 1 \tag{19}$$

so that

$$x^9 + x^8 + x^7 + x^4 + x^2 + x + 1 + x^3 + x^2 + x \equiv x^3 + x^2 + x \pmod{\mu(x)}. \tag{20}$$

6 Search of Redundant Trinomials

An exhaustive search of redundant trinomials has been conducted using `NTL` [12] for extension degrees $n \leqslant 10,000$ when no irreducible trinomial exists. More precisely, given n we try to find a trinomial $t(x) = x^m + x^k + 1$ such that

- $t(x)$ has an irreducible factor of degree n
- m is as small as possible
- k is as small as possible.

It turns out that such a polynomial always exists for the investigated range of degree. To simplify the search one notes that such a trinomial is necessarily squarefree. Indeed $\gcd\big(t(x), t'(x)\big)$ is equal to 1 when m or k is odd. Both m and k cannot be even otherwise $x^m + x^k + 1 = (x^{m/2} + x^{k/2} + 1)^2$ and one should have chosen $x^{m/2} + x^{k/2} + 1$ instead.

Then the idea is to test all the trinomials $x^m + x^k + 1$ with $n + 1 \leqslant m$ and $1 \leqslant k \leqslant \lfloor m/2 \rfloor$ until a good candidate is found, that is a trinomial with a factor of degree $m - n$.

It is well known that $x^{2^k} + x$ is equal to the product of all irreducible polynomials of degree d such that $d \mid k$. Since $t(x)$ is squarefree it is easy to determine if it has a factor $\delta(x)$ of degree $m - n$, computing $\gcd\big(x^{2^i} + x, x^m + x^k + 1\big)$ for successive $i \leqslant m - n$. Note that such a gcd computation can be very costly when $m - n$ is large. It is much faster to compute $g(x) \equiv x^{2^i} \pmod{t(x)}$ by successive squarings and reductions first and then $\gcd\big(g(x) + x, t(x)\big)$. If $t(x)$ has a factor $\delta(x)$ of degree $m - n$ the irreducibility of $t(x)/\delta(x)$ is finally checked.

For all the extensions up to the degree $10,000$ which do not have an irreducible trinomial, our proposal provides a redundant trinomial. There are 4748 such extensions. Note that when an all one polynomial is available it is given even if an irreducible trinomial exists for that extension degree.

Tables containing the redundant trinomials discovered, or all one polynomials when they exist, can be found in [4]. In this paper, we only give results for extensions of degree less than or equal to 1002, see Table 1.

The redundant trinomials $x^m + x^k + 1$ where $m = n + \deg \delta$ and the all one polynomial $(x^{n+1} + 1)/(x + 1)$ are respectively represented by $n, \deg \delta, k$ and $n, 1$. The degree of δ is rather small in general. In about 95% of the cases it is less than or equal to 10. It is maximum for $n = 5373$ and equals 40.

In about 87% of the cases δ is irreducible. With 32-bit processors, redundant trinomials require the same number of words as an irreducible polynomial of

2,1	4,1	8,3,5	10,1	12,1	13,3,3	16,3,4	18,1	19,3,3	24,3,4
26,3,12	27,2,1	28,1	32,5,16	36,1	37,6,4	38,2,17	40,3,3	43,10,2	45,7,9
48,3,20	50,3,5	51,2,4	52,1	53,8,28	56,2,5	58,1	59,2,26	60,1	61,5,17
64,7,12	66,1	67,9,29	69,3,13	70,7,11	72,3,8	75,2,4	77,3,9	78,2,31	80,3,11
82,1	83,2,14	85,8,28	88,8,19	91,8,1	96,2,1	99,2,13	100,1	101,2,2	104,5,9
106,1	107,2,8	109,9,21	112,3,22	114,3,4	115,10,6	116,5,17	117,6,31	120,2,25	122,3,9
125,3,3	128,2,17	130,1	131,7,61	133,3,43	136,3,30	138,1	139,3,3	141,3,13	143,3,53
144,7,19	148,1	149,2,2	152,2,65	157,7,25	158,5,19	160,3,27	162,1	163,8,70	164,5,59
165,5,9	168,3,1	171,2,10	172,1	173,3,5	176,2,53	178,1	179,2,14	180,1	181,7,51
184,3,60	187,7,45	188,4,61	189,3,37	190,4,33	192,3,53	195,2,25	196,1	197,3,69	200,5,42
203,7,73	205,5,29	206,2,17	208,3,45	210,1	211,3,103	213,3,37	216,3,101	219,2,1	221,15,77
222,2,37	224,3,86	226,1	227,2,77	229,3,61	230,2,35	232,3,69	235,14,7	237,3,41	240,2,37
243,2,52	245,2,2	246,2,109	248,2,41	251,2,74	254,2,71	256,16,45	259,5,103	261,3,109	262,4,89
264,3,68	267,2,88	268,1	269,5,47	272,3,87	275,3,99	277,6,12	280,5,103	283,3,51	285,6,122
288,2,133	290,3,114	291,2,31	292,1	293,2,2	296,5,15	298,7,91	299,2,5	301,6,78	304,3,46
306,8,55	307,5,119	309,7,87	311,8,139	312,9,143	315,2,127	316,1	317,3,113	320,3,26	323,2,41
325,3,151	326,2,5	328,3,52	331,13,69	334,5,115	335,6,20	336,3,1	338,3,32	339,2,13	341,10,124
344,2,125	346,1	347,2,173	348,1	349,6,177	352,3,78	355,11,173	356,15,38	357,3,79	360,3,53
361,8,64	363,2,169	365,3,89	368,8,55	371,2,56	372,1	373,3,85	374,2,5	376,3,159	378,1
379,3,187	381,8,99	384,3,94	387,2,67	388,1	389,5,193	392,3,71	395,11,187	397,5,13	398,9,203
400,3,159	403,5,127	405,3,13	408,9,90	410,4,107	411,5,105	413,3,9	416,6,15	418,1	419,2,176
420,1	421,8,14	424,9,112	427,5,5	429,3,137	430,8,91	432,3,38	434,3,170	435,2,61	437,6,12
440,3,146	442,1	443,10,68	445,5,193	448,3,78	451,7,139	452,7,211	453,3,227	454,5,10	456,2,25
459,2,202	460,1	461,3,27	464,2,101	466,1	467,2,29	469,8,109	472,3,214	475,5,133	477,3,89
480,7,224	482,3,108	483,2,16	485,7,181	488,3,180	490,1	491,2,224	493,3,37	496,3,66	499,16,137
501,10,101	502,5,70	504,5,167	507,2,49	508,1	509,12,204	512,9,252	515,2,5	517,5,65	520,3,18
522,1	523,3,3	525,10,89	528,3,121	530,3,24	531,2,226	533,3,195	535,7,25	536,2,113	539,2,92
540,1	541,6,37	542,2,209	544,5,215	546,1	547,7,131	548,2,107	549,5,261	552,2,133	554,3,27
555,2,58	556,1	557,8,12	560,3,99	562,1	563,2,86	565,3,3	568,3,40	571,5,187	572,5,281
573,5,249	576,2,169	578,9,153	579,2,148	581,11,241	584,3,72	586,1	587,2,104	589,3,97	591,8,67
592,7,37	595,3,135	597,7,257	598,5,13	600,2,145	603,2,4	605,3,219	608,3,48	611,2,11	612,1
613,10,76	616,3,64	618,1	619,5,265	621,3,283	624,2,193	627,5,261	629,3,269	630,2,37	632,2,281
635,2,290	637,3,127	638,2,89	640,7,23	643,5,191	644,2,287	645,3,103	648,5,26	652,1	653,3,155
656,2,125	658,1	659,2,80	660,1	661,3,81	664,3,297	666,3,173	667,3,211	669,5,139	672,3,8
674,3,186	675,8,219	676,1	677,3,59	678,2,169	680,2,269	681,2,193	683,2,47	685,3,255	688,3,204
691,14,298	693,8,258	696,3,95	699,2,160	700,1	701,3,167	703,3,25	704,5,169	706,3,34	707,2,74
708,1	709,3,123	710,2,251	712,3,136	715,7,165	717,6,110	720,3,251	723,3,295	725,8,168	728,2,53
731,2,146	733,3,45	734,4,329	736,3,174	739,7,27	741,7,83	744,2,49	747,2,241	749,8,205	752,2,353
755,2,98	756,1	757,3,97	760,5,46	763,3,247	764,4,299	765,3,127	766,5,130	768,3,19	770,3,44
771,2,103	772,1	773,3,11	776,3,132	779,2,161	781,6,375	784,9,86	786,1	787,3,67	788,9,266
789,10,276	790,5,136	792,2,325	795,2,169	796,1	797,7,347	800,2,77	802,7,341	803,2,89	805,3,219
808,6,403	811,5,161	813,3,181	816,5,288	819,2,313	820,1	821,6,363	824,2,149	826,1	827,2,68
828,1	829,3,291	830,2,323	832,3,94	835,5,101	836,2,275	837,5,223	840,3,155	843,2,187	848,2,341
851,2,119	852,1	853,3,307	854,2,161	856,3,235	858,1	859,9,197	863,6,300	864,5,144	867,2,25
869,3,75	872,9,27	874,6,111	875,2,392	876,1	877,10,69	878,7,341	880,3,61	882,1	883,5,395
885,3,137	886,9,314	888,3,241	891,2,442	893,5,59	896,3,65	899,2,329	901,6,1	904,3,6	906,1
907,7,105	909,3,55	910,7,131	912,2,337	914,9,369	915,2,349	917,3,9	920,3,375	922,3,229	923,2,389
925,12,18	928,7,64	929,2,302	931,10,415	933,3,1	934,5,428	936,2,25	939,2,67	940,1	941,3,317
944,2,125	946,1	947,2,50	949,8,38	950,2,383	952,3,324	955,7,321	957,3,367	958,7,174	960,2,241
962,3,464	963,2,7	965,3,293	968,2,29	970,7,226	971,2,179	973,12,233	974,7,65	976,3,394	978,3,425
980,5,11	981,5,235	984,2,313	987,2,28	989,3,11	992,5,472	995,2,89	997,3,319	1000,9,140	1002,3,41

Table 1.

degree n in more than 86% of the cases to represent field elements. Otherwise one more word is necessary, except for the extension of degree 5373 which needs two more words.

For each degree, the factor δ is not explicitly given in Table 1, but it is easy to retrieve since

$$\delta(x) = \gcd\left(x^m + x^k + 1, \prod_{i=1}^{m-n} \left(x^{2^i} + x\right)\right). \tag{21}$$

Also $\delta(x)$ can be found by trial divisions when its degree is small.

The complete data, including the expression of $\delta(x)$, are available on the Internet [4].

7 Tests

Computations have been done on a PC with a Pentium IV processor at 2.6 Ghz running Linux. The test program was written in `C++`, compiled with `gcc-2.96` using `NTL 5.3.1` [12] and compares the efficiency of irreducible pentanomials against redundant trinomials for some basic operations within extension fields of $\mathbb{F}_2$ of prime degree between 50 and 400. For both systems of representation, namely $\mathbb{F}_2[x]/\big(p(x)\big)$ and $\mathbb{F}_2[x]/\big(t(x)\big)$, we give in Table 2 the running times and the respective speedup (in percent) for

- the reduction of a polynomial of degree $2n-2$ (resp. $2m-2$) modulo $p(x)$ (resp. $t(x)$).
- the squaring of an element of $\mathbb{F}_{2^n}$
- the multiplication of two elements of $\mathbb{F}_{2^n}$
- the exponentiation of an element of $\mathbb{F}_{2^n}$ to an exponent less than 2^n.

The unit used is 10^{-7}s for reduction, squaring and multiplication. It is 10^{-5}s for exponentiation.

Redundant trinomials are not well suited for inversions, at least when computed with an extended GCD computation. Results show that inversions are about 15% slower with redundant trinomials.

We remark that prime extension degrees 59, 197, 211, 277, 311, 317, 331, 347, 389, and 397 are quite particular. Indeed for these n, there exists a trinomial of degree $m = \lceil n/32 \rceil \times 32$ with an irreducible factor of degree n. We call such a polynomial an *optimal redundant trinomial*. For all these degrees, except for $n = 317$, another redundant trinomial of smaller degree exists. However tests show that the results are much better with optimal trinomials. Thus when it is possible, these polynomials are used instead. With the same conventions as previously they are

$59, 5, 9$	$197, 27, 103$	$211, 13, 67$	$277, 11, 83$	$293, 27, 91$
$311, 9, 33$	$331, 21, 81$	$347, 5, 127$	$389, 27, 205$	$397, 19, 175$

		Red.			Sqr.			Mul.			Exp.		
n	$\deg \delta$	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*
53	8	1.63	1.37	15.95	2.17	1.77	18.43	3.51	3.04	13.39	1.82	1.53	15.93
59	5	1.64	0.89	45.73	2.17	1.37	36.87	3.51	2.63	25.07	2.01	1.39	30.85
61	5	1.63	1.33	18.40	2.20	1.70	22.73	3.49	4.87	−39.54	2.07	2.05	0.97
67	9	1.57	1.31	16.56	2.18	1.80	17.43	5.37	4.99	7.08	2.67	2.28	14.61
83	2	2.01	1.48	26.37	2.46	1.89	23.17	5.70	5.40	5.26	3.51	3.00	14.53
101	2	1.88	1.50	20.21	2.42	2.01	16.94	6.60	6.08	7.88	4.44	3.88	12.61
107	2	1.91	1.50	21.47	2.49	2.02	18.88	6.53	6.02	7.81	4.64	4.05	12.72
109	9	1.93	1.64	15.03	2.47	2.16	12.55	6.53	6.26	4.13	4.76	4.33	9.03
131	7	2.04	1.50	26.47	2.62	2.14	18.32	10.28	10.07	2.04	7.26	6.28	13.50
139	3	2.37	1.87	21.10	3.00	2.16	28.00	10.77	10.24	4.92	8.19	6.95	15.14
149	2	2.69	1.86	30.86	3.24	2.39	26.23	11.02	10.55	4.26	9.15	7.68	16.07
157	7	2.73	1.82	33.33	3.31	2.39	27.79	11.01	12.46	−13.17	9.73	8.92	8.32
163	8	2.50	1.72	31.20	3.02	2.28	24.50	13.31	12.08	9.24	10.65	8.90	16.43
173	3	2.73	1.90	30.40	3.38	2.47	26.92	13.00	12.39	4.69	11.61	9.87	14.99
179	2	3.01	2.15	28.57	3.61	2.67	26.04	13.09	12.66	3.28	12.90	10.66	17.36
197	27	3.03	1.51	50.17	3.78	2.14	43.39	15.16	13.50	10.95	14.50	10.74	25.93
211	13	3.43	1.55	54.81	4.14	2.14	48.31	15.35	13.50	12.05	16.49	11.50	30.26
227	2	3.17	2.27	28.39	4.01	2.98	25.69	17.08	15.51	9.19	18.29	15.53	15.09
229	3	3.25	2.35	27.69	4.18	3.03	27.51	16.70	15.28	8.50	18.24	15.75	13.65
251	2	3.70	2.52	31.89	4.72	3.07	34.96	16.79	15.27	9.05	21.14	17.70	16.27
269	5	3.71	3.04	18.06	4.62	3.77	18.40	27.05	26.49	2.07	28.65	26.51	7.47
277	11	4.12	1.97	52.18	4.80	2.70	43.75	27.43	25.37	7.51	30.44	23.42	23.06
283	3	4.08	3.16	22.55	4.86	3.89	19.96	27.43	26.47	3.50	31.22	28.30	9.35
293	27	3.81	2.12	44.36	4.69	2.88	38.59	31.09	29.12	6.34	34.15	28.03	17.92
307	5	4.50	2.96	34.22	5.32	3.67	31.02	31.70	30.11	5.02	38.10	32.48	14.75
311	9	4.52	2.09	53.76	5.33	2.90	45.59	31.74	29.11	8.29	38.58	29.63	23.20
317	3	4.52	2.12	53.10	5.36	2.87	46.46	31.74	29.12	8.25	39.18	30.01	23.40
331	21	4.57	2.26	50.55	5.58	3.18	43.01	35.95	33.54	6.70	44.07	35.56	19.31
347	5	4.98	2.20	55.82	5.83	3.12	46.48	36.18	33.53	7.32	47.41	37.04	21.87
349	6	4.99	3.16	36.67	5.83	4.06	30.36	36.17	37.58	−3.90	47.77	43.24	9.48
373	3	5.18	3.51	32.24	6.23	4.33	30.50	38.44	36.55	4.92	53.66	45.72	14.80
379	3	5.20	3.34	35.77	6.25	4.26	31.84	38.44	36.67	4.60	54.44	46.21	15.12
389	5	4.50	3.29	26.89	5.50	4.15	24.55	41.67	40.44	2.95	56.47	50.41	10.73
389	27	4.56	2.41	47.15	5.52	3.35	39.31	41.67	39.40	5.45	56.13	46.48	17.19
397	19	5.24	2.39	54.39	6.20	3.36	45.81	42.14	39.41	6.48	60.50	47.41	21.64

Table 2.

		Red.			Sqr.			Mul.			Exp.		
n	$\deg \delta$	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*
1019	2	1.22	0.75	38.52	1.41	0.96	31.91	1.36	1.32	2.94	39.84	33.97	14.73
2499	2	2.57	1.80	29.96	2.94	2.05	30.27	7.60	7.50	1.32	365.91	340.75	6.88
5013	9	4.68	3.31	29.27	5.45	4.00	26.61	22.68	22.54	0.62	1840.55	1757.94	4.49
7597	17	7.87	5.05	35.83	8.65	5.97	30.98	35.34	35.09	0.71	4133.90	3896.40	5.75
9995	2	9.92	6.59	33.57	11.22	7.78	30.66	67.96	67.62	0.50	9561.80	9180.50	3.99

Table 3.

		Dbl.			Add.			Mul.		
n	$\deg \delta$	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*	*pent.*	*tri.*	*gain*
163	8	1.35	1.24	8.15	3.60	3.33	7.50	1.79	1.61	10.06
197	27	1.52	1.09	28.29	4.07	3.49	14.25	2.42	2.10	13.22
277	6	1.81	1.57	13.26	6.72	6.45	4.02	5.69	5.41	4.92
317	3	1.91	1.30	31.94	7.61	6.74	11.43	7.41	6.65	10.26

Table 4.

Unfortunately the extension degrees which allow the use of optimal redundant trinomials are quite rare. However an *optimal redundant quadrinomial* whose degree is a multiple of 32 and having an irreducible factor of degree n are much easier to find for a given n. Tests with `NTL` showed that in some cases optimal redundant quadrinomials give better result than nonoptimal redundant trinomials and even than irreducible trinomials.

In Table 3 we perform the same computation for bigger degrees. The units are in μs for reduction and squaring, 10^{-5}s for multiplication and 10^{-4}s for exponentiation.

Finally, we have done some computations on elliptic curves defined over finite fields represented with pentanomials and redundant trinomials. Table 4 contains the running times of an addition and a doubling in μs with Montgomery's method. The times for scalar multiplications, also with Montgomery's method, are in ms.

8 Conclusion

In this paper we propose to use reducible trinomials, called redundant trinomial, instead of irreducible pentanomials to represent finite fields of characteristic 2. This allows a faster reduction and more generally a faster arithmetic. The improvement is about 20% for reductions and squarings. For multiplications it is usually less than 5%. We also propose to use sparse reducible polynomials of degree a multiple of the word length (usually 32 bits) having an irreducible factor of degree n to represent $\mathbb{F}_{2^n}$. This idea seems promising but has to be investigated further. Testing the equality of two elements is a costly operation, and should be avoided if possible.

This work naturally extends to other fields, in particular extension fields of characteristic 3. It can be applied to larger characteristic as well. Indeed Mersenne prime numbers or primes of the form $2^n \pm c$ with c small are used to define prime fields of large characteristic and Optimal Extension Fields [1] because of the fast integer reduction they provide. However these primes are quite rare, but when $N = 2^n \pm c$ is not prime but has a large prime factor p the same kind of idea applies, namely working in $\mathbb{F}_p$ by actually computing in $\mathbb{Z}/N\mathbb{Z}$.

Acknowledgment

The author would like to thank Richard Brent and Paul Zimmermann who made him aware of their work [2,3].

References

1. D. V. Bailey and C. Paar. Efficient arithmetic in finite field extensions with application in elliptic curve cryptography. *Journal of Cryptology*, 14(3):153–176, 2001.

2. R. Brent and P. Zimmermann. Algorithms for finding almost irreducible and almost primitive trinomials. Primes and Misdemeanours: Lectures in Honour of the Sixtieth Birthday of Hugh Cowie Williams, The Fields Institute, Toronto, to be published by the American Mathematical Society.
   See `http://web.comlab.ox.ac.uk/oucl/work/richard.brent/pd/rpb212.pdf`.
3. R. Brent and P. Zimmermann. Random number generators with period divisible by a mersenne prime. In *Computational Science and its Applications - ICCSA 2003*, volume 2667, pages 1–10. Springer-Verlag, Berlin, 2003.
   See `http://web.comlab.ox.ac.uk/oucl/work/richard.brent/pd/rpb211.pdf`.
4. C. Doche. A table of redundant trinomials in characteristic 2 up to the degree 10000.
   See `http://www.math.u-bordeaux.fr/~cdoche/documents/redundant.gp.gz`.
5. G. Frey. Applications of arithmetical geometry to cryptographic constructions. In D. Jungnickel and H. Niederreiter, editors, *Fifth International Conference on Finite Fields and Applications*, pages 128–161. Springer-Verlag, Berlin, 2001.
6. J. von zur Gathen and J. Gerhard. Arithmetic and factorization of polynomials over $\mathbb{F}_2$, 1996.
7. J. von zur Gathen and M. Nöcker. Polynomial and normal bases for finite fields. To appear.
8. P. Gaudry, F. Hess, and N. P. Smart. Constructive and destructive facets of Weil descent on elliptic curves. *Journal of Cryptology*, 15(1):19–46, 2002. Online publication: 29 August 2001.
9. T. Jebelean. An algorithm for exact division. *J. Symbolic Computation*, 15(2):169–180, 1993.
10. A. Menezes and M. Qu. Analysis of the Weil descent attack of Gaudry, Hess and Smart. In *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Comput. Sci.*, pages 308–318. Springer-Verlag, Berlin, 2001.
11. G. Seroussi. Table of low–weight binary irreducible polynomials. Technical Report HPL–98–135, Hewlett–Packard, August 1998.
12. V. Shoup. NTL: A Library for doing Number Theory, ver. 5.3.1.
13. H. Wu, M. A. Hasan, I. F. Blake, and S. Gao. Finite field multiplier using redundant representation. *IEEE Trans. Computers*, 51(11):1306–1316, 2002.