Fast and Secure Elliptic Curve Scalar Multiplication Over Prime Fields Using Special Addition Chains

Nicolas Meloni[1,2]

[1] Institut de Mathmatiques et de Modlisation de Montpellier,
UMR 5149, Montpellier, France
[2] Laboratoire d'Informatique,
de Robotique et de Microélectronique de Montpellier,
CNRS UMR 5506, Montpellier, France
meloni@lirmm.fr

Abstract. In this paper, we propose a new fast and secure point multiplication algorithm. It is based on a particular kind of addition chains involving only additions (no doubling), providing a natural protection against side channel attacks. Moreover, we propose new addition formulae that take into account the specific structure of those chains making point multiplication very efficient.

1 Introduction

Since it has been introduced by Miller and Koblitz in [12, 9], elliptic curve cryptography (ECC) has been the subject of plenty of improvements and attacks. Various methods has been proposed to speed up and secure the computation of the scalar point multiplication (the computation of kP where k is an integer and P a point of a curve). See [3, 4] for a complete overview of methods.
In this paper, we study a very particular kind of addition chains (that we called Special Addition Chains) that leads to a natural side channel analysis (SCA) resistant exponentiation algorithm. Moreover we show that it is very well suited to general and Montgomery elliptic curves over prime fields, giving rise to a fast and secure point multiplication.
After some recall about ECC, we introduce special addition chains (SAC) and the way they can be adapted to ECC. Then we study more precisely the length of such chains and finally compare them to other SCA resistant algorithms.

2 Background

2.1 Elliptic Curve Cryptography

Definition 1. *An elliptic curve E over a field K denoted by E/K is given by the equation*

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

where $a_1, a_2, a_3, a_4, a_6 \in K$ are such that, for each point (x, y) on E, the partial derivatives do not vanish simultaneously.

In practice, the equation can be simplified into

$$y^2 = x^3 + ax + b$$

where $a, b \in K$ and $4a^3 + 27b^2 \neq 0$, over field of characteristic greater than 3.

The set of points of E/K is an abelian group. There exist explicit formulae to compute the sum of two points, and several coordinate systems have been proposed to speed up this computation. For a complete overview of those coordinates, one can refer to [3]. As an example, in jacobian coordinates, the curve E (over a prime field) is given by $Y^2 = X^3 + a_4XZ^4 + a_6Z^6$, the point (X, Y, Z) on E correspond to the affine point $(\frac{X}{Z^2}, \frac{Y}{Z^3})$ and the formulae are :

Addition:
$P = (X_1, Y_1, Z_1), Q = (X_2, Y_2, Z_2)$ and $P + Q = (X_3, Y_3, Z_3)$

$$A = X_1Z_1^2,\ B = X_2Z_1^2,\ C = Y_1Z_2^3,\ D = Y_2Z_1^3,\ E = B - A,\ F = D - C$$

and

$$X_3 = -E^3 - 2AE^2 + F,\ Y_3 = -CE^3 + F(AE^2 - X_3),\ Z_3 = Z_1Z_2E$$

Doubling:
$[2]P = (X_3, Y_3, Z_3)$

$$A = 4X_1Y1^2,\ B = 3X_1^2 + a_4Z1^4$$

and

$$X_3 = -2A + B^2,\ Y_3 = -8Y1^4 + B(A - X_3),\ Z_3 = 2Y_1Z_1.$$

The computation cost is 12 multiplications (M) and 4 squarings (S) (8M and 3S if one of the point is given in the form (X,Y,1)) for the addition and 4M and 6S for the doubling.

Montgomery proposed in [13] to work with the following kind of curves:

Definition 2. *Let K be a prime field, an elliptic curve E_M/K is said to be in Montgomery form if its equation is:*

$$E_M : By^2 = x^3 + Ax^2 + x$$

Note that curves in Montgomery form can always be converted into short classic form, however the converse is false.
On such curves the addition and doubling formulae are the following :

Addition: $n \neq m$

$$X_{m+n} = Z_{m-n}((X_m - Z_m)(X_n + Z_n) + (X_m + Z_m)(X_n - Z_n))^2,$$
$$Z_{m+n} = X_{m-n}((X_m - Z_m)(X_n + Z_n) - (X_m + Z_m)(X_n - Z_n))^2.$$

Doubling: $n = m$

$$4X_nZ_n = (X_n + Z_n)^2 - (X_n - Z_n)^2,$$
$$X_{2n} = (X_n + Z_n)^2(X_n - Z_n)^2,$$
$$Z_{2n} = 4X_nZ_n((X_n - Z_n)^2 + ((A + 2)/4)(4X_nZ_n)),$$

where (X_n, Y_n, Z_n) represent the point $[n]P$, for a given point P. Thus, an addition takes 4M and 2S whereas a doubling needs 3M and 2S.

Note that to compute the point $[m + n]P = [m]P + [n]P$, on need to know the x and z-coordinates of the points $[m]P, [n]P$ and $[m]P - [n]P$.

Finally, one should notice that there exist formulae to recover the y-coordinate at the end of a point multiplication [14].

2.2 Side Channel attacks

Side channel attacks have been discovered by Kocher in [10, 11]. They consist in deducing secret informations, as the bits of the exponent in a point multiplication, by analysing the amount of time required to perform secret operations, but also power consumption or electromagnetic radiations. This weakness mainly depends on the fact that during a point multiplication, additions are more expensive than doublings, thus a side-channel analysis allows to deduce what kind of operations are computed, and so to guess the bits of the exponent.
Several counter measures have been proposed against this threat. We can cite for example the use of dummy operations during the process in order to make the group operations look identical, side channel atomicity which consist in splitting the curve operations into identical atomic blocks, Montgomery ladder or elliptic curve in Hessian form [1, 3, 5].

In this paper, to avoid side channel attack, we propose to perform the point multiplication using only point additions. We will show that this can be done in an efficient way using Montgomery curves or with special addition formulae in jacobian coordinates.

3 Scalar point multiplication without doubling

In this section, we present our new exponentiation method and how it can be adapted to elliptic curve scalar point multiplication.

3.1 Special addition chains

We begin by some classic definition used in addition chain study :

Definition 3. *An addition chain computing an integer k is given by two sequences v and w such that*

$$\begin{aligned} &v = (v_0, \ldots, v_s),\ v_0 = 1,\ v_s = k \\ &v_i = v_{i_1} + v_{i_2}\ \textit{for}\ 1 \leq i \leq s\ \textit{with} \\ &w = (w_1, \ldots, w_s),\ w_i = (i_1, i_2)\ \textit{and}\ 0 \leq i_1, i_2 \leq i-1 \end{aligned}$$

The length of the addition chain is s.

Definition 4. *A star addition chain is an addition chain which satisfies:*

$$\forall i, w_i = (i-1, j),$$

for some j such that $0 \leq j \leq i-1$. That is to say that for all i we have $v_i = v_{i-1} + v_j$.
In this case we can omit $i-1$ an just write $w_i = j$.

One can find lot of literature about addition chains [7] and how they are used in exponentiation problems.

In the remainder of the paper we will study a particular kind of star addition chains define as follow :

Definition 5. *A special addition chain is a star addition chain with* $w_i = \begin{cases} i-2\ \textit{or} \\ w_{i-1} \end{cases}$

As w_i can take only two different values we rewrite w as follow:
$w = (w_3, \ldots, w_s) \in \{0,1\}^{s-2}$ satisfying :

$$\begin{aligned} &v_0 = 1,\ v_1 = 2,\ v_2 = 3, \\ &v_i = v_{i-1} + v_j \Rightarrow v_{i+1} = v_i + \begin{cases} v_{i-1}\ \ if\ \ w_{i+1} = 0 \\ v_j\ \ if\ \ w_{i+1} = 1 \end{cases} \end{aligned}$$

Finally, in order to lighten the notations, we will abusively note $k = (w_3, \ldots, w_s)$.

Example 1. $34 = (1,0,0,1,1,0)$

$$\begin{aligned} &v_2 = v_1 + v_0 = 2 + 1\ \textit{and}\ w_3 = 1 \Rightarrow v_3 = v_2 + v_0 = 4 \\ &w_4 = 0 \Rightarrow v_4 = 4 + 3 \\ &w_5 = 0 \Rightarrow v_5 = 7 + 4 \\ &w_6 = 1 \Rightarrow v_6 = 11 + 4 \\ &w_7 = 1 \Rightarrow v_7 = 15 + 4 \\ &w_8 = 0 \Rightarrow v_8 = 19 + 15 = 34 \end{aligned}$$

Given a point P on an elliptic curve E, an integer k and $w = (w_3, \ldots, w_s)$ an special addition chain computing k, it is easy to deduce the following exponentiation algorithm :

Algorithm 1: AddExp(k, P)

Data: $P \in E$ and $k = (w_3, \ldots, w_s)$;
Result: $[k]P \in E$;
$(U_1, U_2, U_3) \leftarrow (P, [2]P, [3]P)$;
for $i = 3 \ldots s$ **do**
  if $w_i = 0$ **then**
    $U_1 \leftarrow U_2$;
  end
  $U_2 \leftarrow U_3$;
  $U_3 \leftarrow U_1 + U_2$;
end
return U_3

This algorithm is particularly well suited to elliptic curves in Montgomery form as at each step we have the points $U_1 = [k_1]P$, $U_2 = [k_2]P$ and $U_3 = U1 + U2 = [k_1 + k_2]P = [k']P$ that is we have exactly what we need to compute $U_3 + U_i = [k']P + [k_i]P, i \in \{1, 2\}$.
Eventually, the cost of this algorithm is one initial doubling and $s - 1$ addition, that is $(4s - 1)$M and $(2s + 1)$S.

We show next that this approach can be generalized to non Montgomery curves.

3.2 New elliptic curve point addition formulae over prime field

Let $p > 3$ be a prime number and $E/\mathbb{F}_p$ an elliptic curve, if $P = (X_1, Y_1, Z)$, $Q = (X_2, Y_2, Z)$ and $P + Q = (X_3, Y_3, Z_3)$ are three points of E given in jacobian coordinates then we have :

$$\begin{aligned} X_3 &= (Y_2Z^3 - Y_1Z^3)^2 - (X_2Z^2 - X_1Z^2)^3 - 2X_1Z^2(X_2Z^2 - X_1Z^2)^2 \\ &= ((Y_2 - Y_1)^2 - (X_2 - X_1)^3 - 2X_1(X_2 - X_1)^2)Z^6 \\ &= ((Y_2 - Y_1)^2 - (X_1 + X_2)(X_2 - X_1)^2)Z^6 \\ &= X_3'Z^6 \end{aligned}$$

$$\begin{aligned} Y_3 &= -Y_1Z^3(X_2Z^2 - X_1Z^2)^3 + (Y_2Z^3 - Y_1Z^3)(X_1Z^2(X_2Z^2 - X_1Z^2)^2 - X_3) \\ &= (-Y_1(X_2 - X_1)^3 + (Y_2 - Y_1)(X_1(X_2 - X_1)^2 - X_3'))Z^9 \\ &= Y_3'Z^9 \end{aligned}$$

$$\begin{aligned} Z_3 &= Z^2(X_2Z^2 - X_1Z^2) \\ &= Z(X_2 - X_1)Z^3 \\ &= Z_3'Z^3 \end{aligned}$$

Thus we have $(X_3, Y_3, Z_3) = (X_3' Z^6, Y_3' Z^9, Z_3' Z^3) \sim (X_3', Y_3', Z_3')$.
So when P and Q have the same z-coordinate, $P + Q$ can be obtained using the following formulae:

Addition:
$P = (X_1, Y_1, Z), Q = (X_2, Y_2, Z)$ and $P + Q = (X_3', Y_3', Z_3')$

$$A = (X_2 - X_1)^2,\ B = X_1 A,\ C = X_2 A,\ D = (Y_2 - Y_1)^2$$

and

$$X_3' = D - B - C,\ Y_3' = (Y_2 - Y_1)(B - X_3) - Y_1(C - B),\ Z_3' = Z(X_2 - X_1)$$

This addition requires 5M and 2S.

It may seem infrequent to have both P and Q sharing the same z-coordinate. However if we look at the quantities $X_1 A = X_1(X_2 - X_1)^2$ and $Y_1(C - B) = Y_1(X_2 - X_1)^3$ computed during the addition, they can be seen as the x and y-coordinates of the point $(X_1(X_2 - X_1)^2, Y_1(X_2 - X_1)^3, Z(X_2 - X_1)) \sim (X_1, Y_1, Z)$. Thus it is possible to add P and $P + Q$ with our new formulae.
The same remark can be done from the doubling formulae, indeed the quantities $A = X_1(2Y_1)^2$ and $8Y_1^4 = Y_1(2Y_1)^3$ are the x and y coordinates of the point $(X_1(2Y_1)^2, Y1(2Y_1)^3, 2Y_1 Z_1) \sim (X_1, Y_1, Z_1)$ allowing us to compute $P + [2]P$ without additional computation.

Using these formulae, the computational cost of algorithm 1 becomes $(5s - 1)$M and $(2s + 4)$S.

Some cryptographic protocols only require the x-coordinate of the point $[k]P$. In this case it is possible to save one multiplication by step of the algorithm 1 by noticing that Z does not appear during the computation of X_3' and Y_3', thus it is not necessary to compute Z_3' during the process. You will find in appendix A how to recover the x-coordinate in the end.

Thus we have proposed new addition point addition formulae taking advantage of the specificity of special addition chains. In the following section we make a theoretical study of those chains in order to find suitable ones for a cryptographic use.

4 Study of special addition chains

4.1 How to find special addition chains

Two questions rise from the previous sections: can any integer be obtained using a special addition chain and how to find such a chain ?
The following example answer both questions :

Example 2. Let $k = 34$ and $k' = 19$ and let apply them the subtractive form of Euclid's algorithm:

$$\begin{aligned} 34 - 19 &= 15 \\ 19 - 15 &= 4 \\ 15 - 4 &= 11 \\ 11 - 4 &= 7 \\ 7 - 4 &= 3 \\ 4 - 3 &= 1 \\ 3 - 1 &= 2 \\ 2 - 1 &= 1 \\ 1 - 1 &= 0 \end{aligned}$$

If we look at the first number of each line we obtain a special addition chain computing 34 : $v = (1, 2, 3, 4, 7, 11, 15, 19, 34)$. One should remark that, on the one hand, k' and k have to be coprime in order to be sure that the chain ends by 1, and on the other hand that k' can be taken greater than $\frac{k}{2}$ as $(k, k - k')$ gives the same addition chain as (k, k'). So in order to find a special addition chain computing an integer k it suffices to apply the following algorithm :

Algorithm 2: SpecialChain(k, k')

Data: $k > k' > \frac{k}{2}$ two coprime integers;
Result: $k = w \in \{0, 1\}^n$;
$v = ()$;
$(U_1, U_2) \leftarrow (k, k')$;
while $U_1 > 3$ **do**
  if $U_1 - U_2 > \frac{U_2}{2}$ **then**
    $(U_1, U_2) \leftarrow (U_2, U_1 - U_2)$;
    $concat(0, w)$;
  end
  $(U_1, U_2) \leftarrow (U_2, 2U_2 - U_1)$;
  $concat(1, w)$;
end
return V

As an example, algorithm 2 applied to 34 and 19 returns $34 = (1, 0, 0, 1, 1, 0)$.

4.2 About special addition chains length

At this point we know how to easily find special addition chains computing any integer, however we are going to see that the study of the length of such chains is a lot more complicated.
We begin by a theorem proved by D. Knuth and A. Yao in 1975 [6].

Theorem 1. *Let $S(k)$ denote the average number of steps to compute $gcd(k, k')$ using the subtractive Euclid's algorithm when k' is uniformly distributed in the range $1 \leq k' \leq k$. Then*

$$S(k) = 6\pi^{-2}(\ln k)^2 + O(\log k(\log\log k)^2)$$

This theorem show that if, in order to find a special addition chain for an integer k, we choose an integer k' at random, it will return a chain of length about $(\ln k)^2$, which is too much long to be used with ECC. Indeed, for a 160-bit exponent, we will see in the last section that to be efficient, our method requires chains of length at most 300, whereas the previous theorem tells us that, theoretically, special chains have a length of 7000 on average (it is rather 2500 in practice).

Before tackling this problem, we need to make some basic recall about the Fibonacci sequence.

Definition 6. *The Fibonacci sequence is defined as follow :*

$$F_n = \begin{cases} 0 \text{ if } n = 0 \\ 1 \text{ if } n = 1 \\ F_{n-1} + F_{n-2} \text{ if } n \geq 2 \end{cases}$$

The Fibonacci sequence has hundreds of properties, one can refers to [7] or [15] to find (almost) them all. We just recall here Binet's Formulae :

Theorem 2. *Binet's Formula :*

$$\forall n \in \mathbb{N},\ F_n = \frac{\phi^n - (1-\phi)^n}{\sqrt{5}}$$

where $\phi = \frac{1+\sqrt{5}}{2}$ *is the positive root of the real polynomial* $X^2 - X - 1$.

From this formula it is easy to deduce the classical results

$$\lim_{n\to\infty} \frac{F_n}{F_{n-1}} = \phi,$$

and

$$F_n = \lceil \frac{\phi^n}{\sqrt{5}} \rfloor$$

where $x \to \lceil x \rfloor$ is the nearest integer function.

Fibonacci numbers are easy to compute using special addition chains, indeed one can check that :

$$F_n = \underbrace{(0, \ldots, 0)}_{n-4 \text{ times}},$$

moreover, F_n is the greatest integer that can be compute by a special addition chain of length $n-2$ ($n-4$ additions plus the initial doubling and addition).
We define now $l_{inf}(k)$ as the minimal length of a special addition chain computing k one can expect, and $l_{min}(k)$ as the length of (one of) the shortest special addition chains computing k. So if n is such that $F_{n-1} < k \leq F_n$ then $l_{min}(k) \geq n-2 = l_{inf}(k)$.
As we see previously $l_{min}(F_n) = l_{inf}(F_n) = n-2$, however a random integer k does not always satisfy $l_{min}(k) = l_{inf}(k)$. As an example :

$$l_{min}(6) = 5 \quad = l_{inf}(6) + 1,$$
$$l_{min}(54) = 10 = l_{inf}(54) + 2,$$
$$l_{min}(43800) = 25 = l_{inf}(43800) + 3$$

An exhaustive search showed that there are no integer lower or equal than 2^{24} for which $l_{min} \geq l_{inf} + 4$. It is tempting to conjecture that $l_{min}(k) \leq l_{inf}(k) + 3$ for any k but up to now, we have not found any result tending to confirm such a conjecture and an exhaustive search from 2^{24} to at least 2^{160} is completely unrealistic.
At this point, we must admit that we have no method to find minimal chains, however efficient point multiplication can be made with just small but not optimal chains, so in the next section we propose a method to find some of them.

4.3 Finding small special addition chains

The first method we proposed to find a special addition chain computing k was to choose an smaller integer k' coprime to k and then apply Euclid's algorithm. Thus, a naive way to find small chains is to test several value for k' and keep the best one. Theorem 1 showed that this method will return chains of length about $6\pi^{-2}(\ln k)^2$on average, and in practice one need billions of attempts to find suitable chains.
So in order to find small chains more rapidly, our idea is to make an "clever" exhaustive search. More precisely, it is to reduce the choice of k' to an area of the range $[\frac{k}{2}, k]$ where the chains are smaller.
To find such an area, first remark that if $k = F_n$ for some n, then the best choice for k' is F_{n-1}, that is to say more or less $\frac{F_n}{\phi}$ (see 2). Intuitively it comes to mind that, for an ordinary k, searching k' around $\frac{k}{\phi}$ could be a clever choice. We are going to see that, indeed, it is the case.

Let k be a positive integers and k' in the range $[\lfloor \frac{k}{\phi} \rfloor - t, \lfloor \frac{k}{\phi} \rfloor + t]$. We have $k' = \frac{k}{\phi} + \epsilon$ for some real number ϵ.
We define also the sequence :

$$k_n = \begin{cases} k \text{ if } n = 0 \\ k' \text{ if } n = 1 \\ k_{n-2} - k_{n-1} \text{ if } n \geq 2 \end{cases}$$

In fact, as long as $k_n > 0$, this sequence correspond to the successive steps of Euclid's algorithm applied to k and k'. We show that choosing k' around $\frac{k}{\phi}$ implies that algorithm 2 will return a chain with many 0 in the end.

We have

$$k_2 = k - k' = k - (\frac{k}{\phi} + \epsilon) = k(\frac{\phi^2 - \phi}{\phi^2}) - \epsilon = \frac{k}{\phi^2} - \epsilon$$

$$k_3 = k_1 - k_0 = \frac{k}{\phi} + \epsilon - (\frac{k}{\phi^2} - \epsilon) = \frac{k}{\phi}(\frac{\phi^2 - \phi}{\phi^2}) + 2\epsilon = \frac{k}{\phi^3} + 2\epsilon$$

By induction, it comes that :

$$k_n = \frac{k}{\phi^n} + (-1)^{n+1} F_n \times \epsilon$$

It is clear that $(F_n \times |\epsilon|)_n$ is an increasing sequence and $(\frac{k}{\phi^n})_n$ is a decreasing sequence, so if for some n we have $F_n \times |\epsilon| < \frac{k}{\phi^n}$, then $\forall\ m \leq n,\ 0 < k_m < k_{m-1}$. Moreover,$\forall m \leq n-2,\ k_m > \frac{k_{m-1}}{2}$, indeed let suppose that $k_m \leq \frac{k_{m-1}}{2}$ then :

$$\begin{aligned} k_{m+1} &= k_{m-1} - k_m \geq \frac{k_{m-1}}{2}, \\ k_{m+2} &= k_m - k_{m+1} \leq 0 \end{aligned}$$

whereas k_{m+2} should be greater than 0.

The previous property show that if n satisfy $F_n \times |\epsilon| < \frac{k}{\phi^n}$, then algorithm 2 will return a chain with at least $n-2$ zeros in the end. Now if $l(k_{n-1}, k_n)$ is the length of the chain returned by algorithm 2 applied to k_{n-1} and k_n, then, then the length of the total chain is $(n-2) + l(k_{n-1}, k_n)$.
Now remember that $k_{n-1} = \frac{k}{\phi^{n-1}} + (-1)^n F_{n-1} \times \epsilon$ so if $F_n \times |\epsilon| < \frac{k}{\phi^n}$ then $\ln k_{n-1} \simeq \ln \frac{k}{\phi^{n-1}}$. If we estimate $l(k_{n-1}, k_n)$ with theorem 1 we get $l(k_{n-1}, k_n) \simeq 6\pi^{-2}(\ln \frac{k}{\phi^{n-1}})^2$.

As an example, fix $n = 100$, (that is force $|\epsilon| < \frac{k}{\phi^{100} F_{100}}$), then we get a theoretical average length of 2500 (1100 in practice).

Of course it is not a rigorous proof, but this gives a good reason why, on average, special addition chains are smaller around $\frac{k}{\phi}$.
So in order to find small addition chains, we begin by testing the value $k' = \lceil \frac{k}{\phi} \rfloor$ and then we test consecutive integers until we find a sufficiently small chain. In table 1 we give practical results on the number of iterations one have to make to find chains computing a 160 bit integer smaller than a fixed length.
The experiments have been made on a 3 GHz Pentium 4, over 10000 random 160-bits integers for chains of length 320 to 270 and 100 random integers for chains of length 260. On average it takes on average 2.5 ms to find a 320 length chain, to 3.24 seconds for a 270 length chain. Finding 260 length can take a few minutes to hours.

Note that for a 160-bit integer k, $l_{min}(k) = 234$, but it is difficult to look for chains of length around 240 (we are not even sure that such chains always exist). However the longer the chains are, the easier it is to find them, so that we can find small chains relatively easily, even if, the computation time of the chain itself being greater than the one of the point multiplication, our method has to be restricted to protocols where the exponent k is part of the secret key (allowing to look for very small chains off-line).

chain length	on average	worst case
320	29	521
300	121	3 454
280	2 353	44 254
270	46 454	1 554 011
260	7 795 840	79 402 210

Table 1. Number of iterations needed to find a chain computing a 160 bit integer, using a "clever" exhaustive approach

Despite this limitation, we are going to see in the next section that special addition chains allow efficient point multiplication and may be taken into consideration in the future.

5 Comparisons to others SCA protected algorithms

In this section we compare our algorithm to the Montgomery ladder when it is used on Montgomery curves, and to the classic double-and-add, NAF and 4-NAF methods, plus the recent Double-base chain proposed in [2] when used on general curves.

5.1 Montgomery curves

The Montgomery ladder is a classical exponentiation algorithm naturally SCA resistant. Indeed, for each bit (except the last) of the exponent k one addition and one doubling are computed, which gives a complexity of (7M+4S)$(|k|_2 - 1)$ over prime fields (where $|k|_2$ is the bit length of k). So if we consider that the ratio S/M is about 0.8 in $\mathbb{F}_p$ then, for 160-bit integers, we obtain the following table:

Algorithm	#M
Montgomery ladder	1622
SAC 300	1680
SAC 280	1568
SAC 260	1456

Table 2. Comparison between Montgomery ladder and SAC in $\mathbb{F}_p$ for a 160-bit exponent

With chains of length 280 and 260 we obtain a gain of, respectively, 3 and 10 %.

5.2 General curves over $\mathbb{F}_p$

In the case of general curves, protecting the classic algorithms against SCA implies the use of side channel atomicity, which implies that the ratio S/M is 1 (the same multiplier is used for multiplication and square), whereas the very structure of special addition chains allows not to have resort to side channel atomicity (so that we keep the ratio S/M=0.8). We refer to [2] for a precise study of double-and-add, NAF, 4-NAF and Double-base chain complexities. For 160-bit integers we obtain:

Algorithm	#M
double-and-add	2511
NAF	2214
4-NAF	1983
double-base chain	1863
SAC 300	1983
SAC 280	1851
SAC 260	1719

Table 3. Comparison of different elliptic curve exponentiation algorithms over $\mathbb{F}_p$ for a 160-bit exponent

We remark that the use of special addition chains of length 300 already have a gain of 21% over double-and-add and 10% over NAF. From chains of length 280 to 260, we outperform all the previous methods, with a gain of 26 to 31% over double-and-add, 16 to 22% over NAF, 7 to 13% over 4-NAF and 1 to 8% over double base chain.

6 Conclusions

In this paper, we have proposed a new exponentiation method, based on special addition chains, that suits very well to Montgomery elliptic curves and general curves over prime fields. We also have presented new formulae in the case of general curves that allow to take advantage of the particular structure of special addition chains. Finally, even if we did not solve the problem of finding minimal chains, we have shown a way to find small chains by looking for them in a "clever" range. All of this leading to a very simple, efficient and naturally SCA resistant scalar multiplication algorithms. However it still implies either off-line computation (if k is part of the secret key) or, if k has to be chosen randomly, to generate directly the exponent as a special addition chain. In this latest case, a lot of study will have to be made in order to know how to generate a "random" chain. Yet we hope that the reader has been seduced by the originality of our approach and the interesting theoretical questions it raises.

References

1. B. Chevalier-Mames, M. Ciet, and M. Joye. Low-cost solutions for preventing simple side-channel analysis: Side-channel atomicity. *IEEE Transactions on Computers*, 53(6):760-768, June 2004.
2. V. Dimitrov, L. Imbert, and P. K. Mishra. Efficient and secure Elliptic Curve Point Multiplication Using Double-Base Chains. *ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 59-78, 2005.
3. R. M. Avanzi, H. Cohen, C. Doche, G. Frey, T. Lange, K. Nguyen, and F. Vercauteren. *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. CRC Press, 2005.
4. D. Hankerson, A. Menezes, and S. Vanstone. *Guide to Elliptic Curve Cryptography*. Springer-Verlag, 2004.
5. M. Joye and J.-J. Quisquater, Hessian elliptic curves and side-channel attacks. *Cryptographic Hardware and Embedded Systems-CHESS 2001*, volume 2162 of *Lecture Notes in Computer Science*. Springer-Verlag, 2001.
6. D. Knuth, and A. Yao. Analysis of the subtractive algorithm for greatest common divisors. *Proc. Nat. Acad. Sct. USA*, volume 72, No 1, pages 4720-472, Dec. 1975.
7. D. Knuth. Fundamental Algorithms. *The Art of Computer Programming*, vol. 1. Addison-Wesley (1981)
8. D. Knuth. Seminumerical Algorithms. *The Art of Computer Programming*, vol. 2. Addison-Wesley (1981)
9. N. Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):203-209,Jan. 1987
10. P. C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In N. Koblitz, editor, *Advances in Cryptology - CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 104-113. Springer-Verlag, Aug. 1996.
11. P. C. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In M. Wiener, editor, *Advances in Cryptology - CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388-397. Springer-Verlag, Aug. 1999.
12. V.S. Miller. Uses of elliptic curves in cryptography. In H. C. Williams, editor, *Advaces in Cryptology-CRYPTO'85*, volume 218 of *Lecture Notes in Computer Science*, pages 417-428. Springer-Verlag, 1986.
13. P. L. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Math. Com.* 48:177 (1987) 143-264.
14. K. Okeya, K. Sakurai. Efficient Elliptic curve cryptosystems from a scalar multiplication algorithm with recovery of the y-coordinate on a Montgomery-form elliptic curve. CHESS 2001, volume 2162 of *Lecture Notes in Computer Science*, pages 126-141. Springer-Verlag, 2001.
15. N. Vorobiev. *Fibonacci Numbers*. Birkhuser Verlag, 2002.

A Recovery of x-coordinate

As said in section 3 the x-coordinate of the sum of two points P and Q can be recovered without computing the z coordinate. Or in other word the value $\frac{X_{P+Q}}{Z_{P+Q}^2}$ (where $P+Q=(X_{P+Q},Y_{P+Q},Z_{P+Q})$ can be recovered thanks to the the following property :

Proposition 1. *Let $P=(X_1,Y_1,Z)$, $Q=(X_2,Y_2,Z)$ and $P+Q=(X_3,Y_3,Z_3)$ be points of an elliptic curve E given in jacobian coordinates, then*

$$Z^2 = a4\frac{(X_1-X_2)(X_3+2Y_2Y_1-X_1X_2(X_1+X_2))-(X_1+X_2)(Y_1^2-Y_2^2+X_2^3-X_1^3)}{2a6(Y_1^2-Y_2^2+X_2^3-X_1^3)}$$

and so

$$x_3 = \frac{X_3}{Z_3^2} = \frac{X_3}{Z^2(X_1-X_2)^2}$$

Proof : P and Q satisfy $Y^2=X^3+a4XZ^4+a6Z^6$ so

$$Y_1^2-Y_2^2 = X_1^3-X_2^3+a4X_1Z^4-a4X_2Z^4+a6Z^6-a6Z^6$$

which gives

$$Z^4 = \frac{Y_1^2-Y_2^2+X_2^3-X_1^3}{a4(X_1-X_2)}$$

moreover

$$\begin{aligned}
X_3 &= (Y_2-Y_1)^2-(X_1+X_2)(X_2-X_1)^2\\
&= Y_2^2-2Y_2Y_1+Y_1^2-X_2^3+X_2^2X_1+X_1^2X_2-X_1^3\\
&= Y_2^2-X_2^3+Y_1^2-X_1^3-2Y_2Y_1+X_1X_2(X_1+X_2)\\
&= a4X_1Z^4+a6Z^6+a4X_2Z^4+a6Z^6-2Y_2Y_1+X_1X_2(X_1+X_2)\\
&= Z^4(a4(X_1+X_2)+2a6Z^2)-2Y_2Y_1+X_1X_2(X_1+X_2)
\end{aligned}$$

and so

$$Z^2 = a4\frac{(X_1-X_2)(X_3+2Y_2Y_1-X_1X_2(X_1+X_2))-(X_1+X_2)(Y_1^2-Y_2^2+X_2^3-X_1^3)}{2a6(Y_1^2-Y_2^2+X_2^3-X_1^3)}$$

Recovering the final x-coordinate can be done in 11M, 4S an one inversion.