Preventing SPA/DPA in ECC Systems Using the Jacobi Form

P.-Y. Liardet[1] and N.P. Smart[2]

[1] STMicroelectronics, Dept. System Engineering,
Z.I. Rousset, 13 106 Rousset Cedex, France
Pierre-Yvan.Liardet@st.com
[2] Dept. Computer Science, University of Bristol,
Merchant Venturers Building, Woodland Road,
Bristol, BS8 1UB, United Kingdom
nigel@cs.bris.ac.uk

Abstract. In this paper we show how using a representation of an elliptic curve as the intersection of two quadrics in $\mathbb{P}^3$ can provide a defence against Simple and Differental Power Analysis (SPA/DPA) style attacks. We combine this with a 'random window' method of point multiplication and point blinding. The proposed method offers considerable advantages over standard algorithmic techniques of preventing SPA and DPA which usually require a significant increased computational cost, usually more than double. Our method requires roughly a seventy percent increase in computational cost of the basic cryptographic operation, although we give some indication as to how this can be reduced. In addition we show that the Jacobi form is also more efficient than the standard Weierstrass form for elliptic curves in the situation where SPA and DPA are not a concern.

1 Introduction

Elliptic curve based cryptosystems are particularly suited for cost-effective implementations of public key primitives on low powered computational devices such as Smart Cards, Mobile Phones and PDAs. Nevertheless, the use of side channel information, such as that provided by Simple and Differential Power Analysis (SPA/DPA) [7] on naive implementations can lead to the revelation of the secrets that the algorithm is working on.

Elliptic curve systems have the advantage of almost always using a new random ephemeral secret integer in the *double and add* algorithm for each run of a protocol, unlike RSA. Hence, a DPA attack on ECC is harder to mount for this reason than one against RSA. On the other hand smart card vendors require any implementation to be as immune as possible from SPA and DPA.

One problem with elliptic curve systems is that the doubling operation is significantly more efficient than the general addition operation. This needs to be compared to the RSA case, where squaring is only slightly more efficient than general multiplication. Hence, it may be possible to use SPA to recover some bits

Ç.K. Koç, D. Naccache, and C. Paar (Eds.): CHES 2001, LNCS 2162, pp. 391–401, 2001.

of each ephemeral exponent, since one may be able to distinguish an addition from a doubling. Recall [6] that for EC-DSA only a few bits of each ephemeral exponent need to be leaked in this way per message, for the underlying secret key to be revealed.

Hence, various proposals have been made to completely secure elliptic curve systems against SPA and DPA. To protect against DPA it has been proposed to use a randomised projective coordinate system. Here the base point on the curve $P = (x, y)$ on each protocol run is first randomised by replacing P with the (Jacobian) projective point

$$P' = (xz^2, yz^3, z),$$

or the (homogeneous) projective point

$$P'' = (xz, yz, z),$$

for some random non-zero field element z. This still allows some of the efficient techniques for point multiplication to be used, such as those described in [1] and [5]. The use of mixed coordinate (i.e. affine and projective coordinates used together) multiplication algorithms are, however, not used which causes some efficiency loss.

Moreover, the above defence will not protect against SPA, hence for SPA protection one of two defences are usually proposed. The first is as follows, instead of computing $[k]P$ one computes $[k + rq]P$, where q is the order of P and r is some random integer. This defence significantly increases the cost of a point multiplication. This does not provide any defence against SPA since if one can recover $k' = k + rq$ from a single run then one can recover $k = k' \pmod q$ for this run since q is known. A second technique is to take a random integer r and compute $k' = rk \pmod q$ and $r' = 1/r \pmod q$. One then computes $Q = [k']P$ and then $[r']Q = [k]P$, again a task which significantly increases computational cost.

Neither of these defences against SPA address the underlying cause, which is the disparity between the addition and doubling algorithms. A model for the elliptic curve in which addition and doubling are given by the same formulae will not suffer from such side channel analysis on the code dependent nature of the operation. In this paper we proposed such a model, based on the Jacobi form of an elliptic curve. Our model, for certain elliptic curves, will provide a defence against SPA and will only give a 70 percent increase in computational cost.

To understand our defence against SPA we first explain roughly how an SPA attack on a standard elliptic curve binary point multiplication method would proceed. Recall the binary method for point multiplication proceeds as in the following algorithm.

Binary Multiplication Method

`INPUT:` A point P and an integer k
`OUTPUT:` The point $Q = [k]P$.

```
1.  Q ← O.
2.  For i from t down to 0 do:
3.      Q ← [2]Q.
4.      If (k_i = 1) then
5.          Q ← Q + P.
6.  Return Q.
```

With a standard representation an attacker can attempt to determine the bits of k by seeing how the program behaves at the *if*-statement. The test is always carried out but the subroutine for point addition will only be called when the ith bit of k is set. The attacker can attempt to spot this jump to a subroutine, which will have a different power trace to point doubling, and hence determine k.

The most common idea to make point addition and doubling indistinguishable, is to unify the common code part for both operations, and add dummy code to balance the difference between point addition and point doubling. Ideally one needs to execute the same code at the same addresses but with different results, but this is unfortunatelly not possible if point addition and point doubling are not unified.

Now suppose exactly the same code was called for point addition and point doubling with the same power trace profile for both operations. The attacker would now need to determine whether one or two calls to this procedure were performed on each iteration. This is a much harder problem for SPA to solve, but if this is still a worry one can unroll the loop to make this task harder for the attacker. But for standard elliptic curve Weierstrass models one cannot use the point addition code in the case where the two points are equal, since the addition formulae contain a singularity when the inputs are the same.

Notice that the defence of simply adding spurious multiplication operations into the doubling code, as mentioned above, would not be a suitable defence since the point doubling and point addition code would still have seperate execution profiles, and would reside in different areas of memory or hardware.

Nevertheless, with the basic double and add algorithm a little bit of information can leak from the bit test, even if the same code is used for point addition and doubling. A carefull implementation can make this information not usable in practice from the point of view of an attacker. Moreover, we present in the last section a multiplication algorithm that reduce significantly the amount of information that can leak from point multiplication.

One is still left open to a DPA style attack whereby internal data bits are guessed (depending on whether the *if* statement produces a branch) and these are correlated over a number of runs. However, for ECC systems these are easily prevented by point blinding (essentially using the redundancy of a projective coordinate representation) or by the protocol using ephemeral point multiples on each run.

2 Intersection of Two Quadrics

Let K denote our ground field, which in applications will be a finite field $\mathbb{F}_p$ of characteristic greater than three. It is well known that an intersection of two quadric surfaces in $\mathbb{P}^3$

$$\mathcal{Q} : \{Q_1(x_0, x_1, x_2, x_3) = 0\} \cap \{Q_2(x_0, x_1, x_2, x_3) = 0\}$$

generically defines a curve of genus one. Hence, assuming $\mathcal{Q}$ has a point defined over K, the curve $\mathcal{Q}$ is birationally equivalent to an elliptic curve, also defined over K.

Just as the chord-tangent law defines a geometric group law on the elliptic curve we can also define a group law on $\mathcal{Q}$ in geometric terms, see [8]. We first let P_0 denote our given K-rational point on $\mathcal{Q}$, which we shall treat as the identity. Three points $P_1, P_2, P_3 \in \mathcal{Q}(K)$ will sum to zero if and only if the four points P_0, P_1, P_2 and P_3 are coplanar. The negation of a point $-P_1$ is given as the residual intersection of the plane through P_1 containing the tangent line to $\mathcal{Q}$ at P_0.

An algorithm to pass from a general intersection of two quadric surfaces with a K-rational point to an elliptic curve is given in [2, p 36]. In [3, pp 63–64] a method is given to pass in the other direction, from a general elliptic curve over K

$$E : Y^2 = X^3 + AX + B,$$

to the intersection of two quadrics given by

$$\mathcal{Q} : \begin{cases} z_1^2 - Bz_3^2 - Az_2z_3 - z_0z_2 = 0, \\ z_2^2 - z_0z_3 = 0. \end{cases}$$

The map from a point $(X, Y) \in E(K)$ to a point $(z_0, z_1, z_2, z_3) \in \mathcal{Q}(K)$ is given by $z_0 = X^2$, $z_1 = Y$, $z_2 = X$ and $z_3 = 1$.

Also in [3] formulae are given to add points on $\mathcal{Q}(K)$. If we let $\mathbf{a}$=(a_0,a_1,a_2,a_3) and $\mathbf{b} = (b_0, b_1, b_2, b_3)$ denote two points on $\mathcal{Q}(K)$ then their sum is given by $\mathbf{c} = \mathbf{a} + \mathbf{b}$ with

$$\begin{aligned} c_0 &= R(\mathbf{a}, \mathbf{b})^2, \\ c_1 &= b_1 S(\mathbf{a}, \mathbf{b}) + a_1 S(\mathbf{b}, \mathbf{a}), \\ c_2 &= R(\mathbf{a}, \mathbf{b}) \cdot T(\mathbf{a}, \mathbf{b}), \\ c_3 &= T(\mathbf{a}, \mathbf{b})^2, \end{aligned}$$

where

$$\begin{aligned} R(\mathbf{a}, \mathbf{b}) &= a_0b_0 - 2Aa_2b_2 - 4Ba_3b_2 - 4Ba_2b_3 + A^2a_3b_3, \\ S(\mathbf{a}, \mathbf{b}) &= a_0^2b_0 + 2Aa_2a_0b_2 + 4Ba_2a_0b_3 + 3Aa_3a_0b_0 \\ &\quad +12Ba_3a_0b_2 - 3A^2a_3a_0b_3 + 4Ba_3a_2b_0 - 2A^2a_3a_2b_2 \\ &\quad -4ABa_3a_2b_3 - A^3a_3^2b_3 - 8B^2a_3^2b_3, \\ T(\mathbf{a}, \mathbf{b}) &= 2a_1b_1 + a_2b_0 + a_0b_2 + Aa_3b_2 + 2Ba_3b_3. \end{aligned}$$

What is remarkable about these equations is that they also hold when $\mathbf{a} = \mathbf{b}$, i.e. when a doubling operation is performed. Hence, the use of such a representation will remove the distinction between doubling and adding, and hence help to defeat SPA as argued above. However, the above formulae are overly complicated and therefore not particularly suited to a real life implementation, so in the next section we reduce to a special class of elliptic curves over K for which the above formulae can be made particularly simple leading to efficient implementation.

3 Jacobi Form

To make the formulae from the above section more amenable to machine calculation we require that our quadrics $\mathcal{Q}$ be simultaneously diagonalisable over K. This is equivalent to saying that our initial elliptic curve has three points of order two defined over K, or equivalently that the polynomial $X^3 + AX + B$ has all three roots defined over K.

Hence, from now on we shall assume we have chosen an elliptic curve

$$E : Y^2 = X^3 + AX + B$$

which has three points of order two defined over K. This means that the group order $N = \#E(F_p)$ is divisible by 4, hence we should choose such a curve with $N = 4q$ with q a prime.

By applying a standard Möbius transformation we can move the three points of order two to the positions $(0, 0), (-1, 0)$ and $(-\lambda, 0)$ where $\lambda \in K$. Our elliptic curve has then become

$$E' : y^2 = x(x + 1)(x + \lambda).$$

To obtain this transformation, first write the factorisation of $X^3 + AX + B$ over K as

$$X^3 + AX + B = (X - \theta_1)(X - \theta_2)(X - \theta_3).$$

Then we define the following Möbius transformation, where $\{i, j, k\} = \{1, 2, 3\}$,

$$\begin{aligned} x &= \frac{X - \theta_i Z}{(\theta_i - \theta_j)^2}, \\ z &= \frac{Z}{\theta_i - \theta_j}, \\ y &= Y(\theta_i - \theta_j)^{5/2}, \end{aligned}$$

where (X, Y, Z) is a homogeneous projective point on $E(K)$ and (x, y, z) is a homogeneous projective point on $E'(K)$. Then setting

$$\lambda = \frac{\theta_i - \theta_k}{\theta_i - \theta_j}$$

we see that the curve E is mapped to the curve E' since

$$\begin{aligned} x(x+z)(x+z\lambda) &= \frac{1}{(\theta_i-\theta_j)^6}(X-\theta_1 Z)(X-\theta_2 Z)(X-\theta_3 Z) \\ &= \frac{Y^2}{(\theta_i-\theta_j)^6} = y^2 z. \end{aligned}$$

This change of variable requires that for some $1 \leq i, j \leq 3$ with $i \neq j$ we have that $\theta_i - \theta_j$ is a square modulo p. If $p \equiv 3 \pmod 4$ then -1 will not be a square modulo p and so either

$$\theta_i - \theta_j \text{ or } \theta_j - \theta_i$$

will be a square modulo p, for all possible i and j. When $p \equiv 1 \pmod 4$ then there is a $1/8$ chance for given $\theta_1, \theta_2, \theta_3$ that we cannot find a pair of indices such that $\theta_i - \theta_j$ is a square modulo p.

In [4] Chudnovsky and Chudnovsky consider the following intersection of two quadrics

$$\mathcal{Q} : \begin{cases} x_0^2 + x_1^2 - x_3^2 = 0, \\ k^2 x_0^2 + x_2^2 - x_3^2 = 0. \end{cases}$$

From two points (a_0, a_1, a_2, a_3) and (b_0, b_1, b_2, b_3) on $\mathcal{Q}$ we can compute their sum (c_0, c_1, c_2, c_3) via the formulae

$$\begin{aligned} c_0 &= a_3 b_1 \cdot a_0 b_2 + a_2 b_0 \cdot a_1 b_3, \\ c_1 &= a_3 b_1 \cdot a_1 b_3 - a_2 b_0 \cdot a_0 b_2, \\ c_2 &= a_3 a_2 b_3 b_2 - k^2 a_0 a_1 b_0 b_1, \\ c_3 &= (a_3 b_1)^2 + (a_2 b_0)^2. \end{aligned}$$

The zero of this group law is given by the point $(0, 1, 1, 1)$. The above formulae for the group law on $\mathcal{Q}$ are also valid when $(a_0, a_1, a_2, a_3) = (b_0, b_1, b_2, b_3)$, and so the same formulae can be used both for doubling and general addition. Each addition or doubling can be efficiently implemented so that it requires a total of 16 field multiplications.

For use in signed window methods of point multiplication we require the formulae for point negation in the Jacobi model. Given the addition formulae above it is easy to see that

$$-(a_0, a_1, a_2, a_3) = (-a_0, a_1, a_2, a_3).$$

We now, for a moment, leave our main application of defences against SPA and DPA and turn to the use of Jacobi form as a way of speeding up algorithms for elliptic curve point multiplication in environments where SPA and DPA are not a concern.

By using the doubling formulae given in [4]

$$\begin{aligned} c_0 &= 2 a_1 a_3 \cdot a_2 a_0, \\ c_1 &= (a_1 a_3)^2 - (a_2 a_3)^2 + (a_1 a_2)^2, \\ c_2 &= (a_2 a_3)^2 - (a_1 a_3)^2 + (a_1 a_2)^2, \\ c_3 &= (a_2 a_3)^2 + (a_1 a_3)^2 - (a_1 a_2)^2, \end{aligned}$$

where $(c_0, c_1, c_2, c_3) = [2](a_0, a_1, a_2, a_3)$, we obtain doubling formulae which only requires eight field multiplications.

However, with a little care one can even achieve doubling in seven field multiplications, which is more efficient than doubling in projective coordinates on a standard Weierstrass equation in odd characteristic.

Lemma 1. *A point can be doubled in the Jacobi model using seven field multiplications.*

Proof. We first take the doubling formulae obtained from specialising the general point addition method to obtain

$$\begin{aligned} c_0 &= 2a_3a_1 \cdot a_2a_0, \\ c_1 &= (a_3a_1)^2 - (a_2a_0)^2, \\ c_2 &= (a_3a_2)^2 - k^2(a_0a_1)^2, \\ c_3 &= (a_3a_1)^2 + (a_2a_0)^2, \end{aligned}$$

which requires ten field multiplications to evaluate. Using the equations of the curve,

$$k^2a_0^2 = a_3^2 - a_2^2 \text{ and } a_0^2 = a_3^2 - a_1^2,$$

we see that we can, assuming $a_2 \neq 0$, rewrite c_2 as

$$c_2 = (a_0a_2)^2 - (a_1a_3)^2 + 2(a_1a_2)^2.$$

Then we can perform a doubling by evaluating

$$\begin{aligned} \ell_1 &= a_3a_1, \\ \ell_2 &= a_0a_2, \\ \ell_3 &= 2(a_1a_2)^2, \\ c_0 &= 2\ell_1\ell_2, \\ c_3 &= (\ell_1 + \ell_2)^2 - c_0, \\ c_1 &= c_3 - 2\ell_2^2, \\ c_2 &= -c_1 + \ell_3. \end{aligned}$$

It is easy to verify that the same equations hold when $a_2 = 0$.

It is interesting to note that this means we can triple a point in $16 + 7 = 23$ field multiplications. Note, in [4] triplication formulae for points in the Jacobi model are also given, which also require only 23 field multiplications.

To use these formulae all that remains is to produce the link between $\mathcal{Q}$ and E'. The two parameters k and λ defining $\mathcal{Q}$ and E' are linked by the equation

$$\lambda = 1 - k^2.$$

To describe the map from E' to $\mathcal{Q}$, let (x, y, z) denote a projective point on E', i.e.

$$y^2z = x(x + z)(x + z\lambda),$$

such a point is obtained from (X, Y) by generating a random $z \in K^*$ and putting $(x, y, z) = (Xz, Yz, z)$, note this homogeneous projective representation, as remarked on both above and below, is needed to prevent DPA attacks. The equivalent point on $\mathcal{Q}$ is then given by the equations

$$\begin{aligned}
x_0 &= -2(x+z)y, \\
x_1 &= -z^2 + z^2k^2 + 2\,z\,k^2x + k^2x^2 + y^2 - x^2 - 2\,z\,x, \\
&= \lambda(-x^2 - z^2 - 2xz) + y^2, \\
x_2 &= -2\,z\,k^2x - k^2x^2 - z^2k^2 + z^2 + 2\,z\,x + y^2 + x^2, \\
&= \lambda(x^2 + z^2 + 2xz) + y^2, \\
x_3 &= -z^2k^2 + k^2x^2 + z^2 + 2\,z\,x + y^2 + x^2, \\
&= \lambda z^2 + y^2 + 2xz + (2-\lambda)x^2.
\end{aligned}$$

The reverse operation is obtained by computing

$$\begin{aligned}
x &= (x_2 - x_3)\lambda, \\
y &= x_0\lambda k^2, \\
z &= x_1k^2 - x_2 + x_3\lambda.
\end{aligned}$$

Suppose we implemented a standard point multiplication algorithm using a signed window method with $r = 5$, see [1, Algorithm IV.7], on the elliptic curve E over $\mathbb{F}_p$, where p is a 192-bit prime number. This would, on average, require 191 point doublings and 38 general point additions. The standard projective coordinate methods on the curve E require 16 field multiplications to perform a general addition and 8 field multiplications to perform a doubling. Hence, the average number of field operations required would be 2136.

Using our Jacobi representation and the same multiplication algorithm we would require on average 3664 field multiplications since both doubling and general addition requires 16 field operations. Hence, we obtain about 70 percent performance penalty as compared to the standard method. However, since doubling and addition is performed by the same code we hopefully obtain a better defence against SPA attacks.

If we were not concerned with a defence against SPA/DPA then using the Jacobi model we can perform a point multiplication in, on average, 1945 field multiplications. This is because we can perform a double in seven field multiplications. Therefore, the Jacobi model gives roughly a ten percent performance improvement over the standard Weierstrass model.

Returning to our main interest of defending against SPA/DPA we can obtain a better performance in the following way. We can flip a coin before doubling to decide whether we use the 7 or the 16 field operations formulae for doubling. The average number of field multiplications then becomes 2040, which is more efficient than the standard algorithm using a Weierstrass model. Hence, we obtain greater efficiency and a defence against SPA/DPA at the same time.

Chudnovsky and Chudnovsky [4] give a number of possible other improvements to multiplication algorithms in Jacobi models. However, they address this problem from the point of view of efficiency and not from the point of view of minimising the effect of DPA. We leave it as an open research problem to reconcile these two approaches for elliptic curves in Jacobi form.

To protect even further against DPA type attacks we stress we need to perform a method of point blinding, whilst transforming from the standard form to the Jacobi form, as above. Assume the affine point $P = (X, Y) \in E(K)$ is given, on every protocol run one then randomises the representation of P by taking a homogeneous representation. This is achieved by generating a random element $Z' \in K^*$ and replacing P by the equivalent point $P' = (X', Y', Z')$ where $X' = XZ'$ and $Y' = YZ'$.

3.1 Example Curve

The prime field $\mathbb{F}_p$ defined by

$$p = 2^{192} - 2^{64} - 1$$

is a popular choice for elliptic curve systems, since it offers a number of efficiency advantages. For this field one could choose the curve defined by

$$\lambda = 421$$

which has group order

$$6277101735386680763835789423320997497001573836313910896964$$

which is four times a 190 bit prime.

4 Randomised Signed Windows Method

To add even further defence against side channel analysis we propose the use of a signed window multiplication algorithm, which uses a random window width. This defence can also be used for standard elliptic curve systems, and not just those in the Jacobi model considered above.

We keep the main signed window algorithm as standard, see for example [1, Algorithm IV.7]. However, we alter the preprocessing of the 'exponent', as in [1, Algorithm IV.6], so as to produce a random window width. We assume that the system will multiply a fixed point P by a random number k, using a lookup table of the point multiples

$$P_i = [2i + 1]P,$$

for $0 \leq i \leq 2^{R-2} - 1$. The preprocessing in the signed window algorithm is used to express k as

$$k = \sum_{i=0}^{d-1} b_i 2^{e_i}$$

where $e_i \in \mathbb{Z}_{\geq 0}$ and

$$b_i \in \{-2^{R-1}+1, -2^{R-1}+3, \ldots, 2^{R-1}-3, 2^{R-1}-1\}.$$

Usually one uses fixed window lengths so that $e_{i+1} - e_i \geq R$ for $0 \leq i \leq d-2$. The following algorithm produces a randomised signed window representation of k which will provide a more difficult target for side channel analysis.

Signed m-ary Window Decomposition

INPUT: An integer $k = \sum_{j=0}^{\ell} k_j 2^j$, $k_j \in \{0,1\}$, $k_\ell = 0$.
OUTPUT: A sequence of pairs $\{(b_i, e_i)\}_{i=0}^{d-1}$.

1. $d \leftarrow 0$, $j \leftarrow 0$.
2. While $j \leq \ell$ do:
3. If $k_j = 0$ then $j \leftarrow j+1$.
4. Else do:
5. $r \leftarrow_{\mathcal{R}} \{1, \ldots, R\}$.
6. $t \leftarrow \min\{\ell, j+r-1\}$, $h_d \leftarrow (k_t k_{t-1} \cdots k_j)_2$.
7. If $h_d > 2^{r-1}$ then do:
8. $b_d \leftarrow h_d - 2^r$,
9. increment the number $(k_\ell k_{\ell-1} \cdots k_{t+1})_2$ by 1.
10. Else $b_d \leftarrow h_d$.
11. $e_d \leftarrow j$, $d \leftarrow d+1$, $j \leftarrow t+1$.
12. Return the sequence $(b_0, e_0), (b_1, e_1), \ldots, (b_{d-1}, e_{d-1})$.

The only change from the standard algorithm is the addition of line 5, where $\leftarrow_{\mathcal{R}}$ denotes a random assignment to the variable on the left from the set on the right.

5 Conclusion

In this paper we have proposed two new defences against side channel analysis for elliptic curve based cryptosystems. Firstly, the use of the Jacobi form for an elliptic curve means that the time/power required to perform a point addition will be almost identical to that of a point doubling. Such a balanced approach is a well known design technique for defeating side channel analysis, and this is the first time a truly balanced technique has been proposed for use in elliptic curve systems. Secondly, the use of a randomised window method creates another level of defence.

In addition our Jacobi form representation can be made more efficient than the standard Weierstrass representation for implementations where SPA and DPA are not a concern.

References

1. I.F. Blake, G. Seroussi and N.P. Smart. *Elliptic curves in cryptography.* Cambridge University Press, 1999.
2. J.W.S. Cassels. *Lectures on Elliptic Curves.* LMS Student Texts, Cambridge University Press, 1991.
3. J.W.S. Cassels and E.V. Flynn. *Prolegomena to a Middlebrow Arithmetic of Curves of Genus 2.* Cambridge University Press, 1996.
4. D.V. Chudnovsky and G.V. Chudnovsky. Sequences of numbers generated by addition in formal groups and new primality and factorisation tests. *Adv. in Appl. Math.*, **7**, 385–434, 1987.
5. H. Cohen, A. Miyaji and T. Ono. Efficient elliptic curve exponentiation using mixed coordinates. In *Advances in Cryptology, ASIACRYPT 98.* Springer-Verlag, LNCS 1514, 51–65, 1998.
6. N.A. Howgrave-Graham and N.P. Smart. Lattice attacks on digital signature schemes. To appear *Designs, Codes and Cryptography.*
7. P. Kocher, J. Jaffe and B. Jun. Differential power analysis. In *Advances in Cryptology, CRYPTO '99*, Springer LNCS 1666, pp 388–397, 1999.
8. J.R. Merriman, S. Siksek, and N.P. Smart. Explicit 4–descents on an elliptic curve. *Acta. Arith.*, **77**, 385–404, 1996.