DPA on n-bit sized Boolean and Arithmetic Operations and its Application to IDEA, RC6 and the HMAC-Construction

Kerstin Lemke, Kai Schramm, and Christof Paar

Communication Security Group (COSY)
Department of Electrical Engineering and Information Sciences
Ruhr-Universität Bochum, Germany
{lemke, schramm, cpaar}@crypto.rub.de

Abstract. Differential Power Analysis (DPA) has turned out to be an efficient method to attack the implementations of cryptographic algorithms and has been well studied for ciphers that incorporate a nonlinear substitution box as e.g. in DES. Other product ciphers and message authentication codes are based on the mixing of different algebraic groups and do not use look-up tables. Among these are IDEA, the AES finalist RC6 and HMAC-constructions such as HMAC-SHA-1 and HMAC-RIPEMD-160. These algorithms restrict the use of the selection function to the Hamming weight and Hamming distance of intermediate data as the addresses used do not depend on cryptographic keys. Because of the linearity of the primitive operations secondary DPA signals arise. This article gives a deeper analysis of the characteristics of DPA results obtained on the basic group operations XOR, addition modulo 2^n and modular multiplication using multi-bit selection functions. The results shown are based both on simulation and experimental data. Experimental results are included for an AVR ATM163 microcontroller which demonstrate the application of DPA to an IDEA implementation.

Keywords: DPA, Boolean and arithmetic operations, IDEA, RC6, HMAC-construction.

1 Introduction

Since 1998 ([2]) it is known that Simple Power Analysis (SPA) and Differential Power Analysis (DPA) can be applied to extract cryptographic keys by measuring the power dissipation of the cryptographic module during the processing. Early investigations on SPA/DPA against symmetric ciphers have been done on the DES Feistel scheme. For the AES candidates the key whitening process was studied using bitwise key hypotheses ([1]). Algorithms that are based on the mixing of different algebraic groups as IDEA and RC6 are theoretically treated in [1] and [6], but not deeply studied in practice, yet.

Software countermeasures to secure cryptographic algorithms with arithmetic and boolean operations turn out to be costly for the conversion algorithm from arithmetic to boolean masking ([9], [10]). In constrained environments these performance costs might not be acceptable for iterated product ciphers, so DPA remains an issue.

This paper aims to give a deeper analysis of DPA scenarios against product ciphers based on arithmetic and boolean operations. Therefore, the expected DPA signals are studied for primitive operations as XOR, addition modulo 2^n and modular multiplication using multi-bit selection functions. For these algorithms, multi-bit selection functions offer an improved trade-off between the number of key hypotheses and the number of DPA calculations regarding to single-bit selection functions. Moreover, the use of single-bit selection functions can require detailed information on the implementation ([1]), which is not as critical for multi-bit selection functions.

Experimental results are given using an implementation of IDEA on an ATM163 micro-controller which is based on the Harvard architecture. To the knowledge of the authors, detailed DPA methods against IDEA and related algorithms as well as concrete results have not previously been published, though it is generally agreed that DPA should be successful.

2 Differential Power Analysis

Differential Power Analysis (DPA) was first introduced by Kocher et al.([2]) and turned out to be a very efficient side channel attack that makes use of a statistical analysis of the power consumption during the execution of a cryptographic algorithm. DPA needs the knowledge of either the plaintext or the ciphertext as a pre-condition.

Power Analysis exploits the dependency of the power consumed by the hardware on the value of intermediate data and addresses used. The attacker knows or assumes a model for this dependency. Two types of leakage have been confirmed which are caused by the Hamming weight and by the Hamming distance of data ([4], [3]). The Hamming weight model is applied best if a pre-charged bus is used. The Hamming distance model considers the dynamic dissipation due to the number of gates that change the state during a transition.

The choice of the key hypotheses and the selection functions depends on the cryptographic algorithm and the implementation to be attacked. In case of DES and AES the preferred selection functions focus on S-box look-ups. Both the address of the S-box table look-up (which is the S-box input), if implemented in software, and the S-box output can leak information. In case of DES, a selection function targeting one S-box access makes use of 6-bit key hypotheses. In case of the AES there are 8-bit key hypotheses. The selection function can be set up on 1-bit or on multiple-bits of intermediate data.

DPA identifies the correct key value by statistical methods for hypothesis testing. An attacker does not need to know details of the implementation as DPA points itself to the relevant points in time. Suitable tests are the 'Distance-of -Means' test, the student's T-Test and the correlation method ([5]).

Algorithms that do not incorporate small S-box tables and use algebraic functions restrict the use of the selection function to the Hamming weight and Hamming distance of intermediate data, since the addresses used do not depend on cryptographic key material. Whereas S-box tables are sufficiently non-linear and have uniform distributions of output values for all key values, this is in general not the case for algebraic constructions. Due to their linearity, secondary DPA peaks occur at related but wrong key hypotheses.

Difficulties evolve if the Hamming distance corresponding to transition counts is the dominant source for power leakage. In this case the selection function should be adapted to the implementation and eventually even restricted to a certain time frame. In case of microcontrollers based on the von-Neumann architecture (shared data/address bus), the Hamming distance to an additional unknown reference variable might be incorporated for the optimisation of DPA results ([8]). In case of an Harvard architecture, the correlation signals depend on the concrete sequence of instructions and registers used by the implementation.

3 DPA Correlation Signals using n-bit sized Basic Operations

Each operation that is considered below is carried out between a known n-bit variable X and a secret n-bit variable K. As the assumption for DPA, X is known and follows the uniform distribution while K is a secret, constant value.

K and X can be represented as the concatenation of k-bit ($k \leq n$) blocks: $K = K_{n/k-1}|K_{n/k-2}|...|K_1|K_0$ and accordingly $X = X_{n/k-1}|X_{n/k-2}|...|X_1|X_0$. A common choice is $k = 8$. We define:

$$K_j = (K \bmod 2^{(j+1)*k}) \operatorname{div}(2^{j*k}) \tag{1}$$

$$X_j = (X \bmod 2^{(j+1)*k}) \operatorname{div}(2^{j*k}) \tag{2}$$

where $j \in \{0, ..., n/k - 1\}$. In the following, the index j is the block number of a n-bit sized variable.

The key hypotheses are set up on each value of K_j. There are 2^k key hypotheses H_{ji}, namely for each j

$$H_{ji} \text{ is } \{K_j = i\} \tag{3}$$

where $i \in \{0, ..., 2^k - 1\}$. From now on, the index i is the key hypothesis for a certain value K_j.

The selection function is defined by the Hamming weight W of an intermediate k-bit-wise result $f(X, j, i)$. $f(X, j, i)$ can be a primitive operation $X_j * i$, wherein $*$ marks the actual operation used.

$$d(X, j, i) = W(f(X, j, i)) - \overline{W(f(X, j, i))} \tag{4}$$

$\overline{W(f(X, j, i))}$ is the mean value of the Hamming weight for the function $f(X, j, i)$ using a summation of all possible input values X. Group operations that are bijective and show a uniform distribution of the resulting values lead to the mean $\overline{W(f(X,j,i))} = k/2$.

If the power leakage is dominated by the Hamming distance the selection function is modified to

$$d(X,j,i) = W(Z_j \oplus (f(X,j,i))) - \overline{W(f(X,j,i))} \tag{5}$$

where Z_j is an additional data item (which can be either constant or random, known or secret) that is in conjunction with the predecessor or successor, the secret intermediate value to be attacked. If Z_j is zero, the Hamming weight model is revealed as a special case of the Hamming distance model. The application of the Hamming weight model can lead to erroneous results, e.g. if $*$ is the XOR operation and Z_j is a constant nonzero value, DPA will point to $(Z_j \oplus K_j)$ as the correct key value. Note, that in the case where Z_j is a random secret value, (first order) DPA will fail. Generally, the Hamming distance model requires a more detailed analysis of the implementation than the Hamming weight model.

If the selection function $d(X,j,i)$ is zero for certain values of X, these single measurements are neglected for DPA; otherwise they are weighted according to the result of the selection function. This multi-bit approach is different to [3] who suggested to use only measurements with the extreme results of $d(X,j,i)$, namely $k/2$ and $-k/2$, which, however, results in highly increased measurement costs. Using our method, only $\binom{k}{k/2}$ single measurements are discarded.

DPA tests for significant statistical power differences between the distributions of single measurements with positive and negative values of $d(X,j,i)$. According to [3] we assume that the data dependent power difference $\Delta P(X,t) = P(X,t) - \overline{P(X,t)}$ is proportional to the Hamming weight of processed data. This power difference $\Delta P(X,t)$ is the signal to be detected.

The DPA results presented here were produced by using the correlation method as follows:

$$c(t,j,i) = \frac{\sum_m d(X_m,j,i)\ \Delta P(X_m,t)}{\sqrt{\sum_m d(X_m,j,i)^2}\sqrt{\sum_m \Delta P(X_m,t)^2}} \tag{6}$$

The number m runs through all single measurements. The correlation coefficient $c(t,j,i)$ will be near zero if the selection function $d(X,j,i)$ and $\Delta P(X,t)$ are uncorrelated. In case of a strong correlation $c(t,j,i)$ approaches 1 or -1 at some points in time.

The following subsections exclusively deal with the generally applicable assumption of the Hamming weight model. The selection functions are to be modified if the Hamming distance is the major source of correlation signals.

3.1 Boolean Operation XOR

XOR is the most important boolean operation that is used in cryptographic algorithms. The selection function used is

$$d(X,j,i) = W(X_j \oplus i) - k/2$$

The correlation coefficient between $d(X, j, i)$ and the power consumption $\Delta P(X, t)$ reaches the maximum if i equals K_j and the minimum if i is $\neg K_j$. The absolute value of the correlation coefficient for both cases is the same. If the power consumption increases with the Hamming weight (which is the normal case), the correct key hypothesis has a positive correlation coefficient; otherwise the correlation coefficient is negative. If the attacker does not know the sign of the linear dependency, a small brute-force analysis has to be applied. Besides the correct value and its bitwise inverted value, less significant correlation coefficients occur at other key hypotheses that differ only by 1-3 bits regarding the correct key hypothesis or the bitwise inverted key hypothesis. Key hypotheses that differ by 4 bits are uncorrelated. The number of key hypotheses that differ by m bits regarding a certain correct key hypothesis is given by $\binom{k}{m}$.

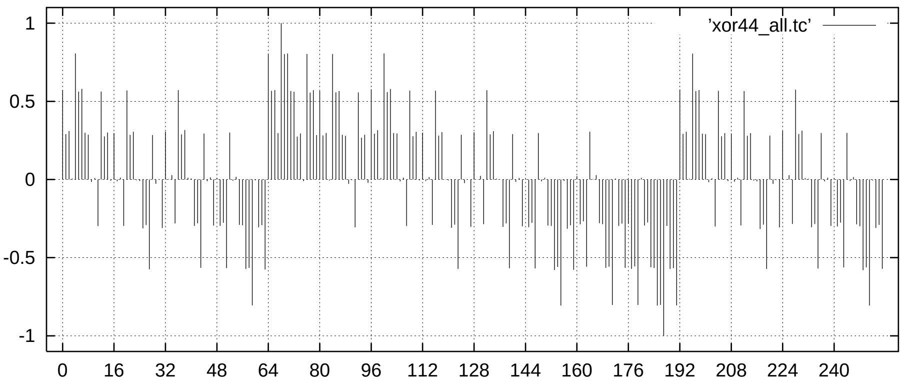

Fig. 1. Correlation coefficient (y-axis) versus all key hypotheses (x-axis) for a XOR operation in case the key hypothesis 68 (0x44) is correct. The results were obtained using simulation data.

Other binary operations such as OR, AND, NOR, NAND, do not form a group operation on the set $\mathbb{Z}_n$. A corresponding k-bit selection function leads to the fact that $\overline{W(f(X, j, i))}$ is dependent on the key hypothesis as the number of single measurements that yield a certain Hamming weight is determined by the number of bits set to 1.

3.2 Addition modulo 2^n

Addition and XOR operation are related primitive operations with the difference that the carry propagates between the bit positions.

The selection function uses the addition modulo 2^n which is denoted by the symbol $\boxplus$. For the case $j = 0$ the selection function is

$$d(X, 0, i) = W(X_0 \boxplus i) - k/2.$$

In case of $j > 0$ the carry of all previous additions has to be incorporated as $C(X_0, K_0, ..., X_{j-1}, K_{j-1}) \in \{0, 1\}$. This results in

$$d(X, j, i) = W(X_j \boxplus i \boxplus C(X_0, K_0, ..., X_{j-1}, K_{j-1})) - k/2$$

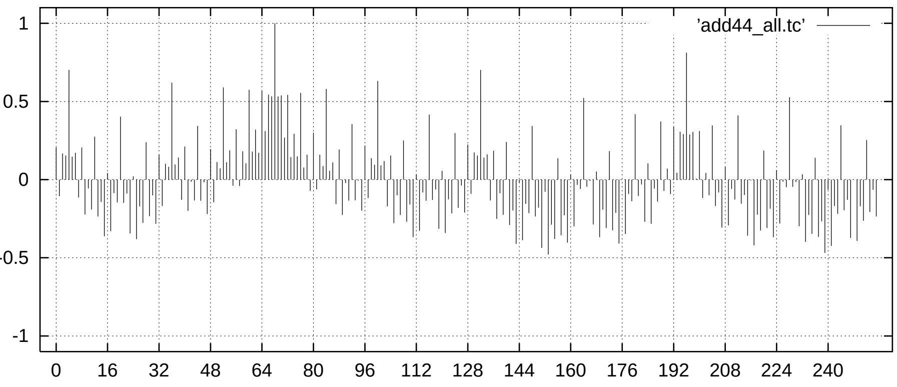

Fig. 2. Correlation coefficient versus all key hypotheses in case of an Addition modulo 2^n with the correct key hypothesis 68 (0x44). The results were obtained using simulation data.

In contrast to boolean operations there is a preferred direction for DPA starting from the least significant block $j = 0$ to incorporate the carry. The correlation coefficient between $d(X, j, i)$ and the power consumption $\Delta P(X, t)$ reaches the maximum if i equals K_j.

Besides the correct value less significant correlation coefficients occur at related hypotheses. The ranking of these hypotheses is $\{K_j, K_j \pm 2^{k-1}, K_j \pm 2^{k-2}, ...\}$ and can be explained by the carry propagation. The result of the selection function using $K_j \pm 2^{k-1}$ differs for all possible values of X_j only by 1-bit with respect to the correct Hamming weight assuming that not all more significant bits of K are set to 1. The two hypotheses $K_j \pm 2^{k-2}$ lead for $2^k/2$ values to a 1-bit difference, for $2^k/4$ values to a zero-bit difference, but for $2^k/4$ values the Hamming weight differs by 2-bits. If the hypotheses differ only at the least significant bit position with respect to the correct key value, the carry propagation leads to a maximal mismatch of the prediction at the transition values 0 and $2^k - 1$.

3.3 Modular Multiplication

The set $\mathbb{Z}_n^* = \{a \in \mathbb{Z}_n | gcd(a, n) = 1\}$ forms a group with the operation multiplication modulo n, whereas the set $\mathbb{Z}_n$ is not a group.

For IDEA a modified multiplication modulo $n = 2^{16} + 1$ is relevant which is denoted by $\odot$. The number of key hypotheses (2^{16}) for DPA is computationally costly, but still feasible using standard equipment. This algebraic operation can be interpreted as a large S-box though it is not implemented as a look-up table. The selection function is

$$d(X, j, i) = W((X \odot i)_j) - k/2.$$

The simulation results in Figure 3 show that some signals occur at related hypotheses. We observed related hypotheses which are given by four sequences $K_{1,m}$,$K_{2,m}$, $K_{3,m}$ and $K_{4,m}$ ($m \in \{0, 1, 2, 3, ...\}$), namely

1. $K_{1,m} = 2^m K \bmod n$,
2. $K_{2,m} = 2^m (n - K) \bmod n$,
3. the following recursive sequence of divisors starting with $K_{3,0} = K$:
   if $K_{3,m}$ is even, then $K_{3,m+1} = \frac{K_{3,m}}{2}$;
   otherwise, $K_{3,m+1} = \frac{(n-K_{3,m})}{2}$.
4. $K_{4,m} = (n - K_{3,m})$

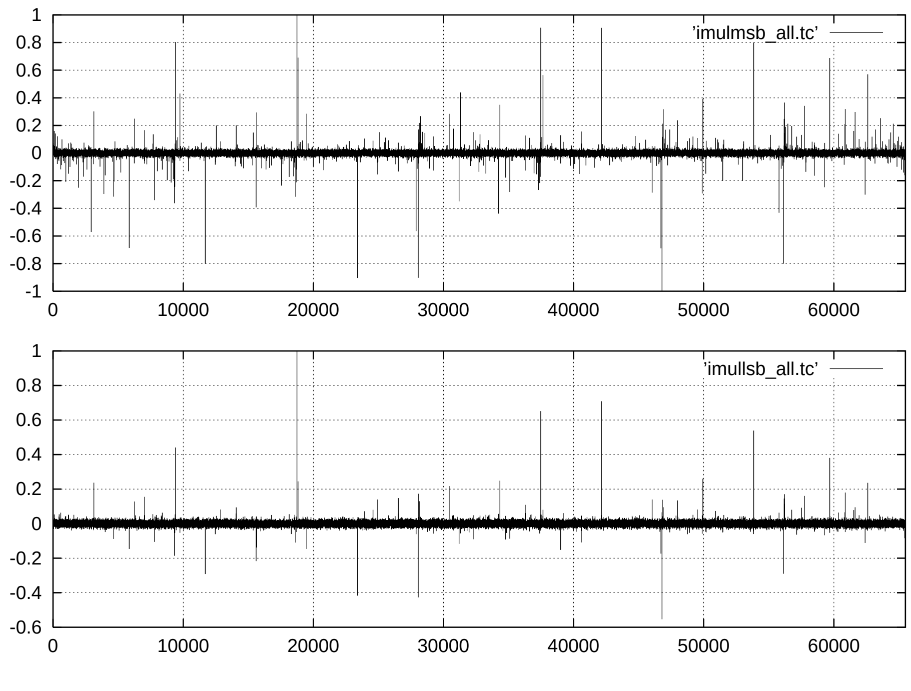

Fig. 3. Correlation coefficient versus all key hypotheses in case the key hypothesis 18737 (0x4931) is correct. The results were obtained using simulation data. The selection function used was at the most significant byte (upper picture) and at the least significant byte (lower picture).

To give an example regarding Figure 3: let the correct key value be $0x4931$. Then the related key hypotheses are

1. $K_1 = \{$`0x4931, 0x9262, 0x24C3, 0x4986, 0x930C,...`$\}$
2. $K_2 = \{$`0xB6D0, 0x6D9F, 0xDB3E, 0xB67B, 0x6CF5,...`$\}$
3. $K_3 = \{$`0x4931, 0x5B68, 0x2DB4, 0x16DA, 0x0B6D,...`$\}$
4. $K_4 = \{$`0xB6D0, 0xA499, 0xD24D, 0xE927, 0xF494,...`$\}$

In case of an 8-bit hardware architecture, both the least significant byte and the most significant byte of the intermediate result can be used for DPA. The correct key value is more highlighted at the least significant byte of the selection function.

As the number of key hypotheses is increased to 2^{16}, DPA against an unknown implementation of IDEA's modified multiplication is much more time-consuming compared to DES or AES. A two-stage approach is desirable that first localizes the relevant points in time and afterwards applies DPA using all key hypotheses. The selection function at the most significant byte can be used for this purpose. For instance, a test based on all hypotheses showed that more than 99,9 % of them are detected at the first stage DPA step using 2^{14} key hypotheses. For this test we assumed that secondary correlation signals can be detected for $m < 5$ for all four sequences. Further improvements are likely to succeed.

4 Application to IDEA, RC6 and the HMAC-Construction

4.1 IDEA

IDEA uses three 16-bit group operations as there are XOR ($\oplus$), addition modulo 2^{16} ($\boxplus$) and the multiplication modulo $2^{16}+1$ ($\odot$) which treats the all-zero subblock as 2^{16} ([11], [16]). The IDEA encryption is reprinted below.

```
for r:=0 to 8 {
        $X_1 = X_1 \odot K_1^{(r)}; X_4 = X_4 \odot K_4^{(r)}; X_2 = X_2 \boxplus K_2^{(r)}; X_3 = X_3 \boxplus K_3^{(r)};$
        $t_0 = K_5^{(r)} \odot (X_1 \oplus X_3);$
        $t_1 = K_6^{(r)} \odot (t_0 \boxplus (X_2 \oplus X_4));$
        $t_2 = t_0 \boxplus t_1;$
        $X_1 = X_1 \oplus t_1; X_4 = X_4 \oplus t_2; a = X_2 \oplus t_2; X_2 = X_3 \oplus t_1; X_3 = a;$
}
$Y_1 = X_1 \odot K_1^{(9)}; Y_4 = X_4 \odot K_4^{(9)}; Y_2 = X_3 \boxplus K_2^{(9)}; Y_3 = X_2 \boxplus K_3^{(9)};$
```

X_1, X_2, X_3 and X_4 are the four 16-bit input words, and Y_1, Y_2, Y_3 and Y_4 are the four 16-bit output words. $K_1^{(r)}$ to $K_6^{(r)}$ are the 16-bit subkey values entering round r of IDEA. Due to the key schedule, the first eight 16-bit subkeys directly give the original IDEA key.

The selection functions used are set up on the operations $\boxplus$ and $\odot$. DPA against the subkey values $K_1^{(r)}$, $K_4^{(r)}$, $K_5^{(r)}$ and $K_6^{(r)}$ uses the operation $\odot$ for the selection function; DPA against the subkey values $K_2^{(r)}$ and $K_3^{(r)}$ uses the operation $\boxplus$.

The operation $\oplus$ can also serve as an additional selection function that reduces remaining candidates of previous results.

4.2 RC6

The design of RC6 makes use of simple primitive operations (integer addition modulo 2^w, integer multiplication modulo 2^w , bitwise exclusive-or and key-dependent bit rotations).

RC6-w/r/b ([12]) works on four w-bit registers A, B, C and D which contain the plaintext and the corresponding ciphertext. The number of rounds is given by r and b denotes the number of key bytes. The key schedule of RC6 yields $2r + 4$ w-bit subkeys $S[i]$, with $i \in \{0, 1, ..., 2r + 3\}$. The RC6-w/r/b encryption is reprinted below.

```
B = B + S[0];
D = D + S[1];
for i:=1 to r {
      t = (B × (2B + 1) ⋘ lg w;
      u = (D × (2D + 1) ⋘ lg w;
      A = ((A ⊕ t) ⋘ u) + S[2i];
      C = ((C ⊕ u) ⋘ t) + S[2i+1];
      (A, B, C, D) = (B, C, D, A);
}
A = A + S[2r + 2];
C = C + S[2r + 3];
```

Key addition is carried out using the addition modulo 2^w. The first keys to be attacked are $S[0]$ and $S[1]$ using the known values B and D. During each iteration, A, B, C, D, t and u are known if all previous subkeys are already compromised by DPA. The key hypotheses are set up on $S[2i]$ and $S[2i + 1]$. The selection function is always the addition modulo 2^w. Signals are expected in the subsequent iteration wherein this intermediate value acts as partial multiplicator. Due to the complex key schedule all r rounds of RC6-w/r/b have to be attacked by DPA iteratively.

4.3 HMAC-Construction

The specification of the HMAC construction can be found in [13] and [14]. The HMAC makes use of a secure hash function H, as e.g. RIPEMD-160 and SHA-1. Let $Text$ be the input message to be secured for message authentication and let K be the secret key used. Further, two fixed strings *ipad* and *opad* are defined. The HMAC is a nested construction that uses two calls to the hash function H.

$$HMAC(Text, K) = H(K \oplus opad, H(K \oplus ipad, Text)) \tag{7}$$

As the first block for each call to H is a constant value that depends only on K in efficient implementations these two values can be precalculated and stored instead of K. Let the two secret values for the inner and outer hash function be defined as follows:

$$K_i = H(K \oplus ipad) \tag{8}$$

and

$$K_o = H(K \oplus opad) \tag{9}$$

In the HMAC-Construction these initialisation vectors IV are the secret values K_i and K_o to be attacked by DPA.

DPA is applied against the first iterations of the inner hash function of the HMAC and after the disclosure of K_i afterwards against the first iterations of the outer hash function. The preferred choice is to start DPA at the first iteration of the hash function which is assumed in the following considerations.

The selection functions depend on the hash function used. The concrete procedure is given only for RIPEMD-160; the approach for SHA-1 is similar.

HMAC-RIPEMD-160 In [15] RIPEMD-160 is specified including a pseudo code in Annex A that is reprinted below for the main loop of iterations.

```
for j:=0 to 79{
      T = rol$_{s(j)}$ (A + f(j,B,C,D) + X[r(j)] + K(j)) + E;
      A:= E; E:=D; D:= rol$_{10}$(C); C:= B; B:=T;
      T = rol$_{s'(j)}$ (A' + f(79-j,B',C',D') + X[r'(j)] + K'(j)) + E';
      A':= E'; E':=D'; D':= rol$_{10}$(C'); C':= B'; B':=T';
}
```

The secret IV is splitted into the five 32-bit words A, B, C, D and E as well as A', B', C', D' and E' before this main loop. X is the message. For each iteration j, $s(j)$ and $s'(j)$ give the number of left shifts and $r(j)$ and $r(j')$ are the number of the message block. The addition + is modulo 2^{32}.

Herein, we focus on the calculation of the five 32-bit words A, B, C, D, and E and skip the similar parallel processing part. For the first 16 iterations, $r(j)$ equals j and the compression function f is $f(x,y,z) = x \oplus y \oplus z$.

The selection functions are applied at successive intermediate results d_1 to d_5 that occur during the processing of the first three iterations. The intermediate results at d_1, d_2, and d_4 are revealed by an addition modulo 2^{32}, d_3 and d_5 are obtained by a XOR operation. The key value to be attacked at each selection function is included in brackets '[' and ']'. An additional subindex is used which denotes the current iteration number.

Intermediate results at the first iteration:

$d_1 = [A_0 + (B_0 \oplus C_0 \oplus D_0)] + X_0$

$d_2 = T_0 = rol_{11}(A_0 + (B_0 \oplus C_0 \oplus D_0) + X_0) + [E_0]$

Intermediate results at the second iteration:

$d_3 = B_1 \oplus [C_1 \oplus D_1] = T_0 \oplus [(B_0 \oplus rol_{10}(C_0))]$

$d_4 = T_1 = rol_{14}(A_1 + (B_1 \oplus C_1 \oplus D_1) + X_1) + [E_1] =$

$$rol_{14}(E_0 + (T_0 \oplus (B_0 \oplus rol_{10}(C_0)) + X_1) + [D_0]$$

Intermediate results at the third iteration:

$$d_5 = B_2 \oplus C_2 \oplus [D_2] = T_1 \oplus (B_1 \oplus [rol_{10}(C_1)]) = T_1 \oplus (T_0 \oplus [rol_{10}(B_0)]$$

If DPA is successful, the results of all selection functions can be combined to reveal A_0, B_0, C_0, D_0 and E_0 which is the IV.

5 Experimental Results of an IDEA Implementation

For the experimental testing IDEA was chosen as it uses three algebraic groups. The IDEA implementation was carried out in Assembly on an 8051 microcontroller (CISC, von-Neumann architecture) and on an 8-bit ATM163 AVR microcontroller (RISC, Harvard architecture). It was assured that both implementations have a constant execution time to exclude broad correlation signals based on timing displacements. The implementations did not include software countermeasures to counteract DPA.

In both tests the DPA characteristics of the simulation results were confirmed using the Hamming weight model. For the 8051 microcontroller we obtained nearly perfect DPA signals. The experimental results are presented in more detail for the ATM163 AVR microcontroller which turned out to be the more difficult analysis.

At a previous characterisation step of the ATM163 the following properties were determined:

- The outstanding DPA signals are caused by Hamming distances of data that is subsequently transferred on the internal bus.
- Correlation signals on the input data to an operation can be revealed with sufficient clarity using the Hamming weight model whereas correlation signals on the output data of an operation are difficult to prove.

Consequently, the Hamming weight model is expected to be successful at the points in time that process the output data of previous operations as the input values.
An additional result is that care has to be taken at the load sequence when alternating key data with known input/output data at subsequent instructions at an AVR core. If known data and secret data are moved one after the other from the SRAM to the working registers using the *ldd* instruction, nearly perfect correlation signals are revealed using the 8-bit XOR selection function. Note, that this observation was also made if two *ldd* instructions are seperated by some arithmetic instructions. An appropriate countermeasure would be to encapsulate the transfer of secret key data by the load of internal, random data.

For the experimental analysis, 5000 single measurements were accumulated at a sampling rate of 200 MHz using a standard DPA measurement set-up. The IDEA key used was in hexadecimal notation:

‘7E 24 95 E1 E5 0C 86 CE 8C C3 1B 80 C0 65 B2 AF’

Addition modulo 2^{16}: Generally, if not superposed by strong correlation signals on the input data, the correct key values are revealed by DPA using 8-bit selection functions for the modular addition. The particular points in time that show correlation signals on the input data can be independently identified by correlation signals on the input data of IDEA.

The experimental DPA characteristics do not always correspond to the expected ones (see Fig. 4). The deviations can be explained by the superposing of signals, especially by leakage of the input data. The analysis on the primary origin of each signal obtained turns out to be a difficult task on the ATM163.

The following is the actual code sequence of the modular addition:

```
ldd r0,Z+2 ; 1st addition: load subkey bytes from SRAM
ldd r1,Z+3
add r5,r1 ; addition with input bytes
adc r4,r0
ldd r0,Z+4 ; 2nd addition: load subkey bytes from SRAM
ldd r1,Z+5
add r21,r1 ; addition with input bytes
adc r20,r0
```

The *add* instruction turns out to be extremely vulnerable against the 8-bit XOR selection function if certain registers are used. In the current example, the instruction *add* $r5, r1$ yields significant DPA signals using the 8-bit XOR selection function at the least significant key byte (see Fig. 5). However, this strong dependency was not confirmed at the instruction *add* $r21, r1$.

Multiplication modulo $2^{16}+1$: The points in time that yield high signals are identified using the advantage, that the key is known. DPA yielded clear correlation signals for the least and most significant byte of the selection function at all relevant positions in time (see Fig. 6). The experimental DPA characteristics are consistent with the expected ones.

As result, the Hamming weight selection function was sucessfully applied, even in presence of a hardware platform that leaks for the most part differential signals.

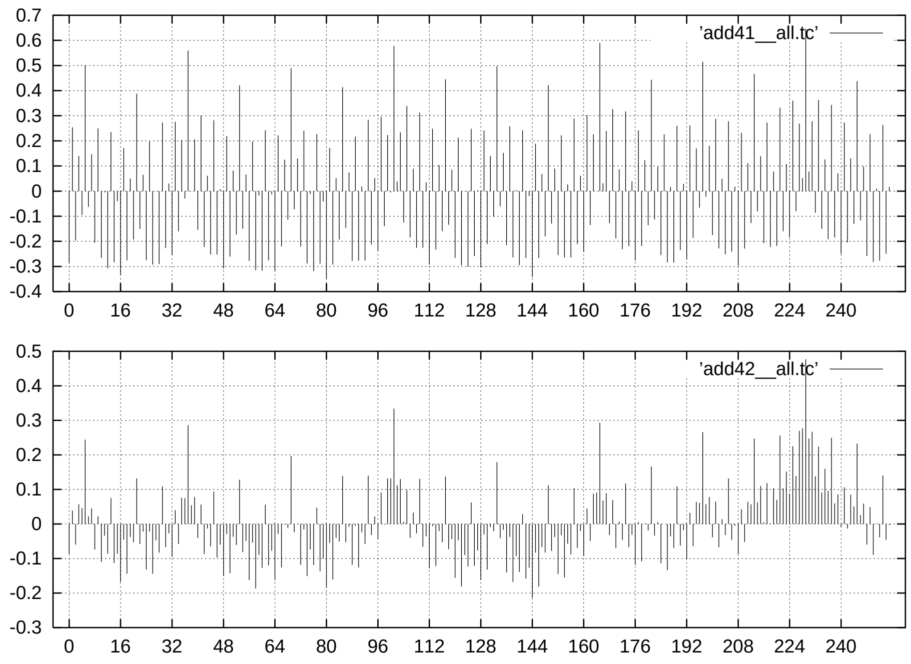

Fig. 4. Correlation coefficient versus all key hypotheses using the ADD selection function at two different points in time. The correct key value 229 (0xE5) for the most significant byte of K_3 is revealed, but only the characteristic in the lower plot points to a pure signal. During the time of the upper plot (negative) correlation signals on the input data are also proven.

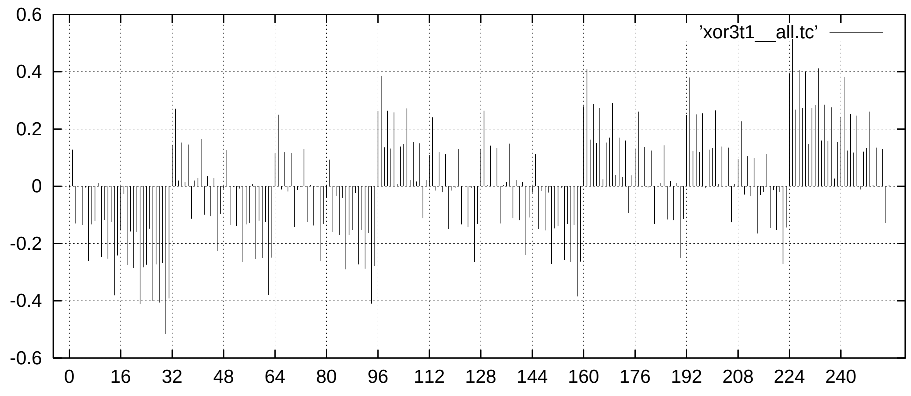

Fig. 5. Correlation coefficient versus all key hypotheses using the XOR selection function at the ldd instruction $ldd\, r1, Z + 3$. The correct key value 225 (0xE1) for the least significant byte of K_2 is revealed.

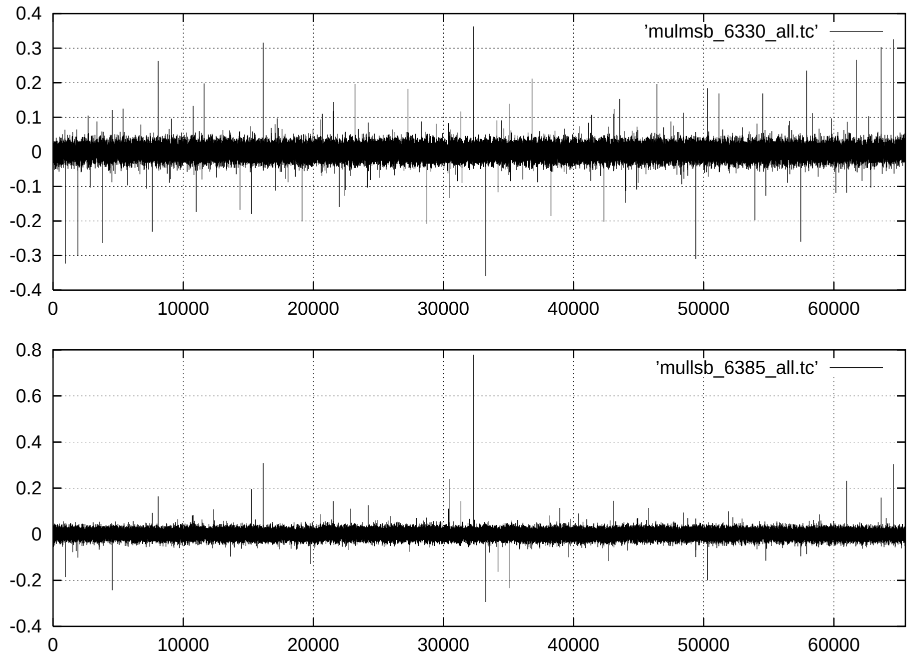

Fig. 6. Correlation coefficient versus all key hypotheses at two points in time. The key value K_1=32292 (0x7E24) is confirmed. The selection function used was at the most significant byte (upper plot) and at the least significant byte (lower plot).

6 Conclusions

This contribution provides an analysis of DPA signals that are revealed in n-bit sized primitive operations such as XOR, addition modulo 2^n and modular multiplication. The characteristics of the DPA results differ for these basic operations and can support the analysis of an unknown implementation.

The theoretical approach to apply DPA in ciphers and message authentication based on primitive operations is included, as are the specific examples of IDEA, RC6 and the HMAC-Construction.

Experimentally, both an IDEA implementation on an 8051 microcontroller and on an AVR ATM163 microcontroller were evaluated. The Hamming weight model was successfully applied at the primitive operations for both architectures and the expected DPA characteristics were confirmed.

Acknowledgements The authors thank Gregor Leander and Jonathan Hammell for the valuable comments which improved this paper.

References

1. S. Chari, C. Jutla, J. R. Rao, P. Rohatgi, *A Cautionary Note Regarding Evaluation of AES Candidates on Smart-Cards*, Proceedings of the second AES conference, pp. 135-150, 1999
2. P. Kocher, J. Jaffe, B. Jun, *Differential Power Analysis*, Advances in Cryptology — Crypto '99 Proceedings, LNCS 1666, pages 388-397, Springer, 1999
3. T. Messerges, E. Dabbish, R. Sloan, *Investigation of Power Analysis Attacks on Smartcards*, USENIX Workshop on Smartcard Techonolgy, USENIX Association, 1999, pp. 151-161
4. R. Mayer-Sommer, *Smartly Analyzing the Simplicity and the Power of Simple Power Analysis on Smartcards*, Cryptographic Hardware and Embedded Systems — CHES 2000, LNCS 1965, pages 78-92, Springer, 2000
5. M. Aigner, E. Oswald, *Power Analysis Tutorial*, available at http://www.iaik.tu-graz.ac.at/aboutus/people/oswald/papers/dpa_tutorial.pdf
6. E. Oswald, B. Preneel, *A Theoretical Evaluation of some NESSIE Candidates regarding their Susceptibility towards Power Analysis Attacks*, October 4, 2002, available at http://www.cosic.esat.kuleuven.ac.be/nessie/reports/phase2/kulwp5-022-1.pdf
7. J. Kelsey, B. Schneier, D. Wagner, C. Hall, *Side Channel Cryptanalysis of Product Ciphers*, Journal of Computer Security, v. 8, n. 2-3, 2000, pp. 141-158.
8. E. Brier, C. Clavier, F. Olivier, *Optimal Statistical Power Analysis*, IACR Cryptology ePrint Archive, Report 2003/152, available at: http://eprint.iacr.org/2003/152.pdf
9. L. Goubin, *A Sound Method for Switching between Boolean and Arithmetic Masking*, Cryptographic Hardware and Embedded Systems — CHES 2001, LNCS 2162, pages 3-15, Springer, 2001
10. J.-S. Coron, A. Tchulkine, *A New Algorithm for Switching from Arithmetic to Boolean Masking*, Cryptographic Hardware and Embedded Systems — CHES 2003, LNCS 2779, pages 89-97, Springer, 2003
11. X. Lai, J. L. Massey, *Markov Ciphers and Differential Cryptanalysis*, Advances in Cryptology — Eurocrypt '91, LNCS 547, pages 17-38, Springer, 1991
12. R. L. Rivest, M. J. B. Robshaw, R. Sidney, X. L. Yin, *The RC6^{TM} Block Cipher*, Version 1.1, August 20, 1998
13. M. Bellare, R. Canetti, H. Krawczyk, *Message Authentication using Hash Functions — The HMAC Construction*, RSA Laboratories' CryptoBytes, Vol. 2, No. 1, 1996
14. M. Bellare, R. Canetti, H. Krawczyk, *Keying Hash Functions for Message Authentication*, Advances in Cryptology — Crypto '96 Proceedings, LNCS 1109, N. Koblitz ed, Springer, 1996
15. H. Dobbertin, A. Bosselaers, B. Preneel, *RIPEMD-160: A Strengthened Version of RIPEMD*, Fast Software Encryption, Cambridge Workshop, LNCS 1039, pages 71-82, Springer, 1996, corrected version available at http://www.esat.kuleuven.ac.be/~cosicart/pdf/AB-9601/AB-9601.pdf
16. A. Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996
17. *ATmega163 ATmega163L, 8-bit AVR Microcontroller with 16K Bytes In-System Programmable Flash*, Rev. 1142E-AVR-02/03, Atmel, available at www.atmel.com