A Sound Method for Switching between Boolean and Arithmetic Masking

Louis Goubin

Schlumberger – CP8
68 route de Versailles, F-78431 Louveciennes, France
Louis.Goubin@louveciennes.tt.slb.com

Abstract. Since the announcement of the Differential Power Analysis (DPA) by Paul Kocher and al., several countermeasures were proposed in order to protect software implementations of cryptographic algorithms. In an attempt to reduce the resulting memory and execution time overhead, a general method was recently proposed, consisting in "masking" all the intermediate data.
This masking strategy is possible if all the fundamental operations used in a given algorithm can be rewritten with masked input data, giving masked output data. This is easily seen to be the case in classical algorithms such as DES or RSA.
However, for algorithms that combine boolean and arithmetic functions, such as IDEA or several of the AES candidates, two different kinds of masking have to be used. There is thus a need for a method to convert back and forth between boolean masking and arithmetic masking.
A first solution to this problem was proposed by Thomas Messerges in [15], but was unfortunately shown (see [6]) insufficient to prevent DPA.
In the present paper, we present two new practical algorithms for the conversion, that are proven secure against DPA.
The first one ("BooleanToArithmetic") uses a constant number of elementary operations, namely 7, on the registers of the processor. The number of elementary operations for the second one ("ArithmeticToBoolean"), namely $5K + 5$, is proportional to the size K (in bits) of the processor registers.

Key words: Physical attacks, Differential Power Analysis, Electric consumption, AES, IDEA, Smartcards, Masking Techniques.

1 Introduction

Paul Kocher and al. introduced in 1998 ([12]) and published in 1999 ([13]) the concept of *Differential Power Analysis* attack, also known as DPA. The initial focus was on symmetrical cryptosystems such as DES (see [12,16]) and the AES candidates (see [1,3,7]), but public key cryptosystems have since been shown to be also vulnerable to the DPA attacks (see [17,5,11]).

In [10,11], Goubin and Patarin proposed a generic countermeasure consisting in splitting all the intermediate variables. A similar "duplication" method

Ç.K. Koç, D. Naccache, and C. Paar (Eds.): CHES 2001, LNCS 2162, pp. 3–15, 2001.

was suggested shortly after by Chari and al. in [3] and [4]. Although the authors of [3] state that these general methods generally increase dramatically the amount of memory needed, or the computation time, Goubin and Patarin proved that realistic implementations could be reached with the "duplication" method. However, it has been shown in [9] that even inner rounds can be aimed by "Power-Analysis"-type attacks, so that the splitting should be performed on all rounds of the algorithm. This makes the issue of the memory and time computation overhead even more crucial, especially for embedded systems such as smartcards.

In [15], Thomas Messerges investigated on DPA attacks applied on the AES candidates. He developped a general countermeasure, consisting in masking all the inputs and outputs of each elementary operation used by the microprocessor. This generic technique allowed him to evaluate the impact of these countermeasures on the five AES algorithms.

However, for algorithms that combine boolean and arithmetic functions, two different kinds of masking have to be used. There is thus a need for a method to convert back and forth between boolean masking and arithmetic masking. This is typically the case for IDEA [14] and for three AES candidates: MARS [2], RC6 [18] and Twofish [19].

T. Messerges proposed in [15] an algorithm in order to perform this conversion between a "$\oplus$ mask" and a "$+$ mask". Unfortunately, Coron and Goubin described in [6] a specific attack, showing that the "BooleanToArithmetic" algorithm proposed by T. Messerges is not sufficient to prevent Differential Power Analysis. In a similar way, his "ArithmeticToBoolean" algorithm is not secure either.

In the present paper, we present two new "BooleanToArithmetic" and "ArithmeticToBoolean" algorithms, proven secure against DPA attacks. Each of these algorithms uses only very simple operations: "XOR", "AND", subtractions and "logical shift left". Our "BooleanToArithmetic" algorithm uses a constant number (namely 7) of such elementary operations, whereas the number of elementary operations involved in our "ArithmeticToBoolean" algorithm is proportional (namely equal to $5K + 5$) to the size (*i.e.* the number K of bits) of the processor registers.

2 Background

2.1 The "Differential Power Analysis" Attack

The "Differential Power Analysis" (DPA) is an attack that allows to obtain information about the secret key (contained in a smartcard for example), by performing a statistical analysis of the electric consumption records measured for a large number of computations with the same key.

This attack does not require any knowledge about the individual electric consumption of each instruction, nor about the position in time of each of these instructions. It applies exactly the same way as soon as the attacker knows the

outputs of the algorithm and the corresponding consumption curves. It only relies on the following fundamental hypothesis:

Fundamental hypothesis: *There exists an intermediate variable, that appears during the computation of the algorithm, such that knowing a few key bits (in practice less than 32 bits) allows us to decide whether two inputs (respectively two outputs) give or not the same value for this variable.*

2.2 The Masking Method

In the present paper, we focus on the "masking method", initially suggested by Goubin and Patarin in [10], and studied further in [11].

The basic principle consists in programming the algorithm so that the fundamental hypothesis above is not true any longer (*i.e.* an intermediate variable never depends on the knowledge of an easily accessible subset of the secret key). More precisely, using a secret sharing scheme, each intermediate variable that appears in the cryptographic algorithm is splitted. Therefore, an attacker has to analyze multiple point distributions, which makes his task grow exponentially in the number of elements in the splitting.

2.3 The Conversion Problem

For algorithms that combine boolean and arithmetic functions, two different kinds of masking have to be used:

$$\text{Boolean masking}: \quad x' = x \oplus r$$
$$\text{Arithmetic masking}: A = x - r \bmod 2^K$$

Here the variable x is masked with random r to give the masked value x' (or A). Our goal is to find an efficient algorithm for converting from boolean masking to arithmetic masking and conversely, in which all intermediate variables are decorrelated from the data to be masked, so that it is secure against DPA.

In all the present paper, we suppose that the processor has K-bit registers (in practice, K is most of the time equal to 8, 16, 32 or 64). All the arithmetic operations (such as the addition "+", the subtraction "−", or the doubling "$z \mapsto 2z$") are considered modulo 2^K. For simplicity, the "mod2^K" will often be omitted in the sequel.

3 From Boolean to Arithmetic Masking

3.1 A Useful Algebraic Property

Let $I = \{0, 1, 2, \ldots, 2^K - 1\}$, with $K \geq 1$ being an integer. Let $x' \in I$. We consider the function $\Phi_{x'} : I \to I$, defined by:

$$\Phi_{x'}(r) \equiv (x' \oplus r) - r \bmod 2^K.$$

We identify each element of I with the sequence of coefficients in its binary representation, so that I can be viewed as a vector space of dimension K over GF(2), isomorphic to GF(2)^K.

Theorem 1

$$\Phi_{x'}(r) = x' \oplus \bigoplus_{i=1}^{K-1} \Big[\Big(\bigwedge_{j=1}^{i-1} \big(2^j \overline{x'}\big) \Big) \wedge \big(2^i x'\big) \wedge \big(2^i r\big) \Big],$$

where $\overline{x'}$ *stands for the ones complement of* x'*, and* $\wedge$ *stands for the boolean "AND" operator.*

See Appendix 1 for a proof of Theorem 1.

Corollary 1.1 *The function* $\Phi_{x'}$ *is affine over GF(2).*

This result is an easy consequence of Theorem 1.

3.2 The "BooleanToArithmetic" Algorithm

Since $\Phi_{x'}$ is affine over GF(2), the function $\Psi_{x'} = \Phi_{x'} \oplus \Phi_{x'}(0)$ is *linear* over GF(2). Therefore, for any value γ,

$$\Psi_{x'}(r) = \Psi_{x'}(\gamma \oplus (r \oplus \gamma)) = \Psi_{x'}(\gamma) \oplus \Psi_{x'}(r \oplus \gamma).$$

Corollary 1.2 *For any value* γ*, if we denote* $A = (x' \oplus r) - r$*, we also have*

$$A = [(x' \oplus \gamma) - \gamma] \oplus x' \oplus [(x' \oplus (r \oplus \gamma)) - (r \oplus \gamma)].$$

$A = (x' \oplus r) - r$ can thus be obtained from the following algorithm:

Algorithm 1. BooleanToArithmetic

Require: (x', r) such that $x = x' \oplus r$
Ensure: (A, r) such that $x = A + r$
  Initialize Γ to a random value γ
  $T \Leftarrow x' \oplus \Gamma$
  $T \Leftarrow T - \Gamma$
  $T \Leftarrow T \oplus x'$
  $\Gamma \Leftarrow \Gamma \oplus r$
  $A \Leftarrow x' \oplus \Gamma$
  $A \Leftarrow A - \Gamma$
  $A \Leftarrow A \oplus T$

The "BooleanToArithmetic" algorithm uses 2 auxiliary variables (T and Γ), 1 random generation and 7 elementary operations (more precisely: 5 "XOR" and 2 subtractions).

3.3 Proof of Security against DPA

From the description of the "BooleanToArithmetic" algorithm, we easily obtain the list of all the intermediate values V_0, ..., V_6 that appear during the computation of $A = (x' \oplus r) - r$:

$$\begin{cases} V_0 = \gamma \\ V_1 = \gamma \oplus r \\ V_2 = x' \oplus \gamma \\ V_3 = (x' \oplus \gamma) - \gamma \\ V_4 = [(x' \oplus \gamma) - \gamma] \oplus x' \\ V_5 = x' \oplus \gamma \oplus r \\ V_6 = (x' \oplus \gamma \oplus r) - (\gamma \oplus r) \end{cases}$$

If we suppose that γ is randomly chosen with a uniform distribution on $I = \{0,1\}^K$, it is easy to see that:

- the values V_0, V_1, V_2 and V_5 are uniformly distributed on I.
- the distributions of V_3, V_4 and V_6 depend on x' but not on r.

4 From Arithmetic to Boolean Masking

4.1 A Useful Recursion Formula

Theorem 2 *If we denote $x' = (A + r) \oplus r$, we also have $x' = A \oplus u_{K-1}$, where u_{K-1} is obtained from the following recursion formula:*

$$\begin{cases} u_0 = 0 \\ \forall k \geq 0,\ u_{k+1} = 2[u_k \wedge (A \oplus r) \oplus (A \wedge r)]. \end{cases}$$

See Appendix 2 for a proof of Theorem 2.

4.2 The "ArithmeticToBoolean" Algorithm

Let γ be any value. The change of variable $t_k = 2\gamma \oplus u_k$ leads to the following consequence of Theorem 2.

Corollary 2.1 *For any value γ, if we denote $x' = (A + r) \oplus r$, we also have $x' = A \oplus 2\gamma \oplus t_{K-1}$, where t_{K-1} is obtained from the following recursion formula:*

$$\begin{cases} t_0 = 2\gamma \\ \forall k \geq 0,\ t_{k+1} = 2[t_k \wedge (A \oplus r) \oplus \omega], \end{cases}$$

in which $\omega = \gamma \oplus (2\gamma) \wedge (A \oplus r) \oplus A \wedge r$.

As a consequence, $x' = (A + r) \oplus r$ can be obtained from the "ArithmeticToBoolean" algorithm below.

This method requires 3 auxiliary variables (T, Ω and Γ), 1 random generation and $(5K + 5)$ elementary operations (more precisely: $(2K + 4)$ "XOR", $(2K + 1)$ "AND" and K "logical shift left").

Algorithm 2. ArithmeticToBoolean

Require: (A, r) such that $x = A + r$
Ensure: (x', r) such that $x = x' \oplus r$
Initialize Γ to a random value γ
$T \Leftarrow 2\Gamma$
$x' \Leftarrow \Gamma \oplus r$
$\Omega \Leftarrow \Gamma \wedge x'$
$x' \Leftarrow T \oplus A$
$\Gamma \Leftarrow \Gamma \oplus x'$
$\Gamma \Leftarrow \Gamma \wedge r$
$\Omega \Leftarrow \Omega \oplus \Gamma$
$\Gamma \Leftarrow T \wedge A$
$\Omega \Leftarrow \Omega \oplus \Gamma$
for $k = 1$ to $K - 1$ **do**
$\Gamma \Leftarrow T \wedge r$
$\Gamma \Leftarrow \Gamma \oplus \Omega$
$T \Leftarrow T \wedge A$
$\Gamma \Leftarrow \Gamma \oplus T$
$T \Leftarrow 2\Gamma$
end for
$x' \Leftarrow x' \oplus T$

4.3 Proof of Security against DPA

From the description of the "BooleanToArithmetic" algorithm, we easily obtain the list of all the intermediate values W_0, ..., W_{5K+4} that appear during the computation of $x' = (A + r) \oplus r$:

$$\begin{cases} W_0 = \gamma \\ W_1 = 2\gamma \\ W_2 = \gamma \oplus r \\ W_3 = \gamma \oplus \gamma \wedge r \\ W_4 = 2\gamma \oplus A \\ W_5 = \gamma \oplus 2\gamma \oplus A \\ W_6 = (\gamma \oplus 2\gamma \oplus A) \wedge r \\ W_7 = \gamma \oplus (2\gamma) \wedge r \oplus A \wedge r \\ W_8 = (2\gamma) \wedge A \\ W_9 = \gamma \oplus (2\gamma) \wedge (A \oplus r) \oplus A \wedge r = \omega \\ \text{for } k = 1 \text{ to } K-1 : \begin{cases} W_{5k+5} = (2\gamma \oplus u_{k-1}) \wedge r \\ W_{5k+6} = \gamma \oplus (2\gamma) \wedge A \oplus u_{k-1} \wedge r \oplus A \wedge r \\ W_{5k+7} = (2\gamma \oplus u_{k-1}) \wedge A \\ W_{5k+8} = \gamma \oplus u_{k-1} \wedge (A \oplus r) \oplus A \wedge r \\ W_{5k+9} = 2\gamma \oplus u_k \end{cases} \end{cases}$$

If we suppose that γ is randomly chosen with a uniform distribution on $I = \{0, 1\}^K$, it is easy to see that:

- the values W_0, W_2 and W_{5k+8} $(1 \leq k \leq K-1)$ are uniformly distributed on I.
- the values W_1 and W_{5k+9} are uniformly distributed on the subset $\{0,1\}^{K-1} \times \{0\}$ of I.
- the distributions of W_3 and W_{5k+5} $(1 \leq k \leq K-1)$ depend on r but not on A.
- the distributions of W_4, W_8 and W_{5k+7} $(1 \leq k \leq K-1)$ depend on A but not on r.

To study the distribution of the remaining values (W_5, W_6, W_7, W_9 and W_{5k+6}), we will make use of the following result:

Theorem 3 *For any $\delta \in I$, the following function is bijective:*

$$\Theta_\delta : \begin{cases} I \to I \\ \gamma \mapsto \gamma \oplus (2\gamma) \wedge \delta. \end{cases}$$

See Appendix 3 for a proof of Theorem 3. As a result:

- the values $W_5 = \Theta_{-1}(\gamma) \oplus A$, $W_7 = \Theta_r(\gamma) \oplus A \wedge r$, $W_9 = \Theta_{A \oplus r}(\gamma) \oplus A \wedge r$ and $W_{5k+6} = \Theta_r(\gamma) \oplus u_{k-1} \wedge r \oplus A \wedge r$ $(1 \leq k \leq K-1)$ are uniformly distributed on I.
- the distribution of $W_6 = (\Theta_{-1}(\gamma) \oplus A) \wedge r$ depends on r but not on A.

5 Conclusion

In this paper, we solved the following open problem (stated in [6]): "find an efficient algorithm for converting from boolean masking to arithmetic masking and conversely, in which all intermediate variables are decorrelated from the data to be masked, so that it is secure against DPA".

The construction of our "BooleanToArithmetic" and "ArithmeticToBoolean" algorithms also led us to prove some results of independent interest. In particular we proved that $r \mapsto (a \oplus r) - r \bmod 2^K$ is an affine function, which seems to be a new result.

Finally, a direction for further research would be to find an improved version of the "ArithmeticToBoolean" algorithm, in which the number of elementary operations is less than $5K + 5$, or (even better) a constant independent of the size K of the registers.

Acknowledgement

I would like to thank Jean-Sébastien Coron for interesting discussions and suggestions.

References

1. Eli Biham and Adi Shamir, "Power Analysis of the Key Scheduling of the AES Candidates", in *Proceedings of the Second Advanced Encryption Standard (AES) Candidate Conference*, http://csrc.nist.gov/encryption/aes/round1/Conf2/aes2conf.htm, March 1999.
2. Carolynn Burwick, Don Coppersmith, Edward D'Avignon, Rosario Gennaro, Shai Halevi, Charanjit Jutla, Stephen M. Matyas, Luke O'Connor, Mohammad Peyravian, David Safford and Nevenko Zunic, "MARS - A Candidate Cipher for AES", NIST AES Proposal, June 1998. Available at: http://www.research.ibm.com/security/mars.pdf
3. Suresh Chari, Charantjit S. Jutla, Josyula R. Rao and Pankaj Rohatgi, "A Cautionary Note Regarding Evaluation of AES Candidates on Smart-Cards", in *Proceedings of the Second Advanced Encryption Standard (AES) Candidate Conference*, http://csrc.nist.gov/encryption/aes/round1/Conf2/aes2conf.htm, March 1999.
4. Suresh Chari, Charantjit S. Jutla, Josyula R. Rao and Pankaj Rohatgi, "Towards Sound Approaches to Counteract Power-Analysis Attacks", in *Proceedings of Advances in Cryptology – CRYPTO'99*, Springer-Verlag, 1999, pp. 398-412.
5. Jean-Sébastien Coron, "Resistance Against Differential Power Analysis for Elliptic Curve Cryptosystems", in *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems*, Springer-Verlag, August 1999, pp. 292-302.
6. Jean-Sébastien Coron and Louis Goubin, "On Boolean and Arithmetic Masking against Differential Power Analysis", in *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems*, Springer-Verlag, August 2000.
7. John Daemen and Vincent Rijmen, "Resistance Against Implementation Attacks: A Comparative Study of the AES Proposals", in *Proceedings of the Second Advanced Encryption Standard (AES) Candidate Conference*, http://csrc.nist.gov/encryption/aes/round1/Conf2/aes2conf.htm, March 1999.
8. John Daemen, Michael Peters and Gilles Van Assche, "Bitslice Ciphers and Power Analysis Attacks", in *Proceedings of Fast Software Encryption Workshop 2000*, Springer-Verlag, April 2000.
9. Paul N. Fahn and Peter K. Pearson, "IPA: A New Class of Power Attacks", in *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems*, Springer-Verlag, August 1999, pp. 173-186.
10. Louis Goubin and J. Patarin, "Procédé de sécurisation d'un ensemble électronique de cryptographie à clé secrète contre les attaques par analyse physique", European Patent, Schlumberger, February 4th, 1999, Publication Number: 2789535.
11. Louis Goubin and Jacques Patarin, "DES and Differential Power Analysis – The Duplication Method", in *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems*, Springer-Verlag, August 1999, pp. 158-172.
12. Paul Kocher, Joshua Jaffe and Benjamin Jun, "Introduction to Differential Power Analysis and Related Attacks", http://www.cryptography.com/dpa/technical, 1998.
13. Paul Kocher, Joshua Jaffe and Benjamin Jun, "Differential Power Analysis", in *Proceedings of Advances in Cryptology – CRYPTO'99*, Springer-Verlag, 1999, pp. 388-397.
14. Xuejia Lai and James Massey, "A Proposal for a New Block Encryption Standard", in *Advances in Cryptology - EUROCRYPT '90 Proceedings*, Springer-Verlag, 1991, pp. 389-404.

15. Thomas S. Messerges, "Securing the AES Finalists Against Power Analysis Attacks", in *Proceedings of Fast Software Encryption Workshop 2000*, Springer-Verlag, April 2000.
16. Thomas S. Messerges, Ezzy A. Dabbish and Robert H. Sloan, "Investigations of Power Analysis Attacks on Smartcards", in *Proceedings of USENIX Workshop on Smartcard Technology*, May 1999, pp. 151-161.
17. Thomas S. Messerges, Ezzy A. Dabbish and Robert H. Sloan, "Power Analysis Attacks of Modular Exponentiation in Smartcards", in *Proceedings of Workshop on Cryptographic Hardware and Embedded Systems*, Springer-Verlag, August 1999, pp. 144-157.
18. Ronald L. Rivest, Matthew J.B. Robshaw, Ray Sidney and Yiqun L. Yin, "The RC6 Block Cipher", v1.1, August 20, 1998. Available at: ftp://ftp.rsasecurity.com/pub/rsalabs/aes/rc6v11.pdf
19. Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall and Niels Ferguson, "Twofish: A 128-Bit Block Cipher", June 15, 1998, AES submission available at: http://www.counterpane.com/twofish.pdf

Annex 1: Proof of Theorem 1

To prove theorem 1, we prove the following more precise result:

Lemma 1 *For any integer $k \geq 1$:*

$$\Phi_a(r) \equiv \Big\{ a \oplus \bigoplus_{i=1}^{k-1} \Big[\Big(\bigwedge_{j=1}^{i-1} \big(2^j \bar{a}\big) \Big) \wedge \big(2^i a\big) \wedge \big(2^i r\big) \Big] \Big\}$$

$$- \Big[\Big(\bigwedge_{j=1}^{k-1} \big(2^j \bar{a}\big) \Big) \wedge \big(2^k a\big) \wedge \big(2^k r\big) \Big] \bmod 2^K,$$

where $\bar{a}$ stands for the ones complement of a, and $\wedge$ stands for the boolean "AND" operator.

Theorem 1 easily follows from Lemma 1, by considering the particular value $k = K$ (and taking $a = x'$).

To prove Lemma 1, we will use the following elementary result.

Lemma 2 *For any integers u and v:*

$$u - v \equiv (u \oplus v) - 2(\bar{u} \wedge v) \bmod 2^K.$$

Proof of Lemma 2 (sketch): $u \oplus v$ gives almost the same result as $u - v$, except that carries have been forgotten. For a given index, a carry appears if and only if a '1' bit (from v) is subtracted from a '0' bit (from u), which corresponds to a '1' bit in $\bar{u} \wedge v = 1$. Since the carry is then subtracted in the *next* index, $\bar{u} \wedge v$ has to be shifted left, which is the same as to be doubled, before being subtracted from $u \oplus v$.

Proof of Lemma 1: We proceed by induction on k.

- We first apply Lemma 2 with $u = a \oplus r$ and $v = r$:

$$\Phi_a(r) \equiv (a \oplus r) - r \equiv a - 2(\overline{a \oplus r} \wedge r) \bmod 2^K.$$

Since $\overline{a \oplus r} = a \oplus \bar{r}$, we have:

$$\Phi_a(r) \equiv a - 2((a \oplus \bar{r}) \wedge r) \equiv a - 2(a \wedge r) \bmod 2^K,$$

which proves the case $k = 1$ of Lemma 1 (conventionally, the empty product $\bigwedge_{j=1}^{0}$ equals the identity element of the $\wedge$ operator).

- Let us suppose that the result of Lemma 1 is true for k:

$$\Phi_a(r) \equiv \Big\{a \oplus \bigoplus_{i=1}^{k-1} \Big[\Big(\bigwedge_{j=1}^{i-1} (2^j \bar{a})\Big) \wedge (2^i a) \wedge (2^i r)\Big]\Big\}$$

$$-\Big[\Big(\bigwedge_{j=1}^{k-1} (2^j \bar{a})\Big) \wedge (2^k a) \wedge (2^k r)\Big] \bmod 2^K$$

and let us show that it is also true for $k+1$.

Let

$$u = a \oplus \bigoplus_{i=1}^{k-1} \Big[\Big(\bigwedge_{j=1}^{i-1} (2^j \bar{a})\Big) \wedge (2^i a) \wedge (2^i r)\Big]$$

and

$$v = \Big(\bigwedge_{j=1}^{k-1} (2^j \bar{a})\Big) \wedge (2^k a) \wedge (2^k r).$$

We first obtain:

$$u \oplus v = a \oplus \bigoplus_{i=1}^{k} \Big[\Big(\bigwedge_{j=1}^{i-1} (2^j \bar{a})\Big) \wedge (2^i a) \wedge (2^i r)\Big].$$

Moreover,

$$\bar{u} = \overline{a \oplus \bigoplus_{i=1}^{k-1} \Big[\Big(\bigwedge_{j=1}^{i-1} (2^j \bar{a})\Big) \wedge (2^i a) \wedge (2^i r)\Big]}$$

$$= \bar{a} \oplus \bigoplus_{i=1}^{k-1} \Big[\Big(\bigwedge_{j=1}^{i-1} (2^j \bar{a})\Big) \wedge (2^i a) \wedge (2^i r)\Big],$$

so that:

$$\bar{u} \wedge v = \Big\{\bar{a} \oplus \bigoplus_{i=1}^{k-1} \Big[\Big(\bigwedge_{j=1}^{i-1} (2^j \bar{a})\Big) \wedge (2^i a) \wedge (2^i r)\Big]\Big\}$$

$$\wedge\Big(\bigwedge_{j=1}^{k-1}\big(2^j\bar{a}\big)\Big)\wedge\big(2^k a\big)\wedge\big(2^k r\big).$$

Therefore

$$\bar{u}\wedge v=\Big(\bigwedge_{j=0}^{k-1}\big(2^j\bar{a}\big)\Big)\wedge\big(2^k a\big)\wedge\big(2^k r\big)$$

because to each index i, $1\le i\le k-1$ in u corresponds an index j, $1\le j\le k-1$ in v (namely $j=i$), such that:

$$\big(2^i a\big)\wedge\big(2^j\bar{a}\big)=0.$$

Therefore, applying Lemma 2:

$$\Phi_a(r)\equiv\Big\{a\oplus\bigoplus_{i=1}^{k}\Big[\Big(\bigwedge_{j=1}^{i-1}\big(2^j\bar{a}\big)\Big)\wedge\big(2^i a\big)\wedge\big(2^i r\big)\Big]\Big\}$$

$$-\Big[\Big(\bigwedge_{j=1}^{k}\big(2^j\bar{a}\big)\Big)\wedge\big(2^{k+1}a\big)\wedge\big(2^{k+1}r\big)\Big]\bmod 2^K.$$

Annex 2: Proof of Theorem 2

We begin by the following elementary result:

Lemma 3 *For any z and δ, the following identity holds:*

$$z+\delta\equiv\overline{\bar{z}-\delta}\bmod 2^K.$$

Proof of Lemma 3: It is easy to see that, for any λ,

$$\lambda+\bar{\lambda}+1\equiv 0\bmod 2^K.$$

Applying this identity successively with $\lambda=\bar{z}-\delta$ and $\lambda=z$, we obtain:

$$\overline{\bar{z}-\delta}\equiv-(\bar{z}-\delta)-1\equiv-((-z-1)-\delta)-1=z+\delta\bmod 2^K.$$

Proof of Theorem 2: We first apply Lemma 3 with $z=A$ and $\delta=r$:

$$A+r=\overline{\bar{A}-r}$$

Moreover,

$$\bar{A}=A\oplus(-1)=((A\oplus r)\oplus(-1))\oplus r=\overline{A\oplus r}\oplus r.$$

Hence

$$A+r=\overline{(\overline{A\oplus r}\oplus r)-r}=\overline{\Phi_{\overline{A\oplus r}}(r)}.$$

From Theorem 1 (with $\overline{A \oplus r}$ instead of x'), we know that:

$$\Phi_{\overline{A\oplus r}}(r) = \overline{A \oplus r} \oplus \bigoplus_{i=1}^{K-1} \Big[\Big(\bigwedge_{j=1}^{i-1} \big(2^j(A \oplus r)\big) \Big) \wedge \big(2^i(\overline{A \oplus r})\big) \wedge \big(2^i r\big) \Big],$$

so that

$$A + r = A \oplus r \oplus \bigoplus_{i=1}^{K-1} \Big[\Big(\bigwedge_{j=1}^{i-1} \big(2^j(A \oplus r)\big) \Big) \wedge \big(2^i A\big) \big) \wedge \big(2^i r\big) \Big].$$

Let us denote, for any integer $k \geq 0$,

$$u_k = \bigoplus_{i=1}^{k} \Big[\Big(\bigwedge_{j=1}^{i-1} \big(2^j(A \oplus r)\big) \Big) \wedge \big(2^i A\big) \big) \wedge \big(2^i r\big) \Big].$$

From the definition of u_k, we have $u_0 = 0$ and $A + r = A \oplus r \oplus u_{K-1}$. Moreover for all $k \geq 0$,

$$u_{k+1} = \bigoplus_{i=1}^{k+1} \Big[\Big(\bigwedge_{j=1}^{i-1} \big(2^j(A \oplus r)\big) \Big) \wedge \big(2^i A\big) \big) \wedge \big(2^i r\big) \Big]$$

$$= 2(A \wedge r) \oplus \bigoplus_{i=2}^{k+1} \Big[\Big(\bigwedge_{j=1}^{i-1} \big(2^j(A \oplus r)\big) \Big) \wedge \big(2^i A\big) \big) \wedge \big(2^i r\big) \Big],$$

so that, if we denote $i' = i - 1$ and $j' = j - 1$:

$$u_{k+1} = 2(A \wedge r) \oplus \bigoplus_{i'=1}^{k} \Big[\Big(\bigwedge_{j'=0}^{i'-1} \big(2^{j'+1}(A \oplus r)\big) \Big) \wedge \big(2^{i'+1} A\big) \big) \wedge \big(2^{i'+1} r\big) \Big]$$

$$= 2\Big\{ (A \wedge r) \oplus \bigoplus_{i'=1}^{k} \Big[\Big(\bigwedge_{j'=0}^{i'-1} \big(2^{j'}(A \oplus r)\big) \Big) \wedge \big(2^{i'} A\big) \big) \wedge \big(2^{i'} r\big) \Big] \Big\}$$

$$= 2\Big\{ (A \wedge r) \oplus (A \oplus r) \wedge \bigoplus_{i'=1}^{k} \Big[\Big(\bigwedge_{j'=1}^{i'-1} \big(2^{j'}(A \oplus r)\big) \Big) \wedge \big(2^{i'} A\big) \big) \wedge \big(2^{i'} r\big) \Big] \Big\}$$

$$= 2[(A \wedge r) \oplus (A \oplus r) \wedge u_k].$$

Annex 3: Proof of Theorem 3

Let δ be any value in I. We begin by proving that Θ_δ is surjective.

Let $y \in I$. If we denote:

$$\gamma = \bigoplus_{i=0}^{K-1} \Big[\Big(\bigwedge_{j=1}^{i} \big(2^{j-1}\delta\big) \Big) \wedge \big(2^i y\big) \Big]$$

(conventionally, the empty product $\bigwedge_{j=1}^{0}$ equals the identity element of the $\wedge$ operator), we have:

$$\gamma \oplus (2\gamma) \wedge \delta = \gamma \oplus \bigoplus_{i=0}^{K-1} \Big[\Big(\bigwedge_{j=0}^{i} \big(2^j \delta\big) \Big) \wedge \big(2^{i+1} y\big) \Big],$$

so that, if we denote $i' = i + 1$ and $j' = j + 1$:

$$\gamma \oplus (2\gamma) \wedge \delta = \gamma \oplus \bigoplus_{i'=1}^{K} \Big[\Big(\bigwedge_{j'=1}^{i'} \big(2^{j'-1} \delta\big) \Big) \wedge \big(2^{i'} y\big) \Big].$$

From the definition of γ, it is easy to see that:

$$\bigoplus_{i'=1}^{K} \Big[\Big(\bigwedge_{j'=1}^{i'} \big(2^{j'-1} \delta\big) \Big) \wedge \big(2^{i'} y\big) \Big] = \gamma \oplus y$$

Therefore:

$$\gamma \oplus (2\gamma) \wedge \delta = y.$$

We have proven that, for any $y \in I$, a value $\gamma \in I$ exists such that $\Theta_\delta(\gamma) = y$. As a consequence, Θ_δ is surjective. Since it maps I onto itself, we deduce that Θ_δ is bijective.