Robust Codes for Fault Attack Resistant Cryptographic Hardware

Konrad J. Kulikowski, Mark G. Karpovsky, Alexander Taubin

Reliable Computing Laboratory, Boston University
8 Saint Mary's Street, Boston, MA 02215
{konkul, markkar, taubin}@bu.edu

Abstract. Hardware implementations of cryptographic algorithms are vulnerable to fault analysis attacks. To detect these attacks we propose an architecture based on robust nonlinear systematic *(n,k)*-error-detecting codes. These nonlinear codes offer advantages over linear codes since they are capable of providing uniform error detecting coverage independently of the error distributions. They make no assumptions about what faults or errors will be injected by an attacker. Architectures based on these codes have fewer undetectable errors than linear codes with the same *(n,k)*. We also present several optimization approaches which provide for a tradeoff between the levels of robustness and required overhead for hardware implementations.

1 Introduction

Hardware implementations of cryptographic algorithms are vulnerable to malicious analyses that exploit the physical properties of the designs. These attacks which exploit the implementation specific weaknesses are known as Side-Channel Attacks (SCA). Power consumption, electro-magnetic radiation, execution time, and behavior in the presence of faults can all be used to drastically decrease the complexity of cryptanalysis. Mobile cryptographic devices such as smartcards and mobile computers are especially vulnerable since the physical hardware implementing the algorithms, and hence the side-channel information, is easily accessible.

The side-channel attacks of interest to this paper are Differential Fault Analysis (DFA) attacks. DFA attacks use the information obtained from an incorrectly functioning implementation of an algorithm to derive the secret information. DFA attacks were first proposed by Biham et al. [1] against hardware implementations of the Data Encryption Standard (DES). They have since been extended to other symmetric key algorithms, such as the Advanced Encryption Standard (AES) in [5-9].

Incorrect operation can result from faults within the circuit (permanent or transient) which may be due to natural effects or be maliciously induced. DFA attacks pose a serious security threat since faults can be injected into a circuit even in the presence of temper resistant packaging by introducing the device to elevated levels of radiation or temperature, atypical clock rate, or incorrect voltage [3].

Current DFA protection methods for symmetric ciphers based on error-detecting codes use linear codes such as parity or repetition codes (e.g. duplication). These linear methods provide for good overall coverage but their error detecting capabilities depend on error distributions. The protection they provide is not uniform against all errors. Linear codes have areas of poor error coverage which can be exploited by an attacker regardless of how good the overall average protection is. In this paper we demonstrate a method of transforming from protection based on linear codes to protection based on non-linear robust codes which provide for more uniform error-detection coverage. This results in a drastic reduction in a number of undetected faults which can be exploited by an attacker. We also present optimization methods for design of robust smart cards which provide for a tradeoff between levels of robustness (uniformity of error coverage) and hardware overheads.

2 Current Protection Methods for Symmetric Ciphers

Several methods and architectures have been proposed for protecting symmetric key ciphers like AES against DFA attacks. These methods range in their granularity, protection they provide, and the overhead they require. The first method, proposed by several groups, is based on linear error-detecting codes [11-12]. Hardware redundancy is added to the circuit to concurrently predict and verify a signature of the device. Usually very simple linear codes such as parity or duplication are used. The second method [10] exploits the symmetry and reversibility of private key algorithms. The method performs the encryption (or decryption) operations followed by their inverse decryption (or encryption). If no error was present in the operation, then performing the inverse operation should lead to the original data. The inverse and comparison can be performed on various granularities. This method usually requires large temporal overhead, since the inverse operation cannot be performed before the original computation is performed, or large hardware overhead to facilitate the verification on a finer granularity. The solutions based on linear codes can have smaller overheads but are efficient only if the errors are within the given distribution for which the codes were designed. We note that several recently published DFA attacks [5-9] require very few faults to be injected.

3 Attack Model

Attackers inject faults and observe errors which are manifestations of the faults at the output. In general, a fault produces useful information for analysis only if an erroneous output can be observed by the attacker. The erroneous output is the expected output distorted by some error e ($e = x \oplus \tilde{x}$, where x is the expected output and $\tilde{x}$ is the observed distorted output, and $\oplus$ stands for componentwise XOR of binary vectors). In this model, detection and prevention of a fault attack is equivalent to determining if the output is distorted by an error (error detection) and suppressing the output to prevent analysis. Multiple faults have to be injected and observed for suc-

cessful cryptanalysis so it is important to have the highest protection possible to detect the attack before it is successful and disable the device.

We do not limit the analysis to any method of fault injection but assume that tamper proof packaging is used so that the attacker does not have direct access to the chip surface. We assume that the attacker cannot precisely control the location of the injected faults and so the locations of the actual faults are randomly distributed within some given area of the chip. We assume that it is realistic for the attacker to have control over the multiplicity (number) of faults introduced. The multiplicity of faults can be controlled by manipulating the fault injection mechanisms such as the level of radiation, temperature, etc.

4 Limitations of Methods Based on Linear Error-Detecting Codes

Protection methods based on linear error-detecting codes do not provide for uniform level of protection against all possible faults but rather concentrate on a certain subclass of the possible faults. One of the most important criteria for evaluating the effectiveness of a protection method is not to consider the overall average protection the method provides, but rather focus on the size and type of the security holes which exist.

The three main criteria that are important for evaluating the effectiveness and practicality of a protection scheme are:

1. The number of undetectable output distortions or output errors
2. The maximum probability of missing an error or class of errors
3. Spatial and temporal overhead

Methods based on linear error-detecting codes do not provide optimum solutions with respect to the above criteria. For example, consider protection based on duplication where the hardware is doubled and the outputs of both copies are compared. If the copies match, then no error was assumed to have occurred. If an error affects an odd number of bits per a ($n = k + r,\ k = r$)-bit codeword (where k=number of information bits, r=number of redundant bits), then this protection scheme can detect those errors, and hence prevent an attack, 100% percent of the time. However, when errors are of an even multiplicity it is possible that the error will distort both copies in the same manner thus making the error undetectable. As an example, Figure 1 shows the percent of detectable errors as a function of error multiplicity (number of distorted bits) for 7-bit duplication (k=r=7).

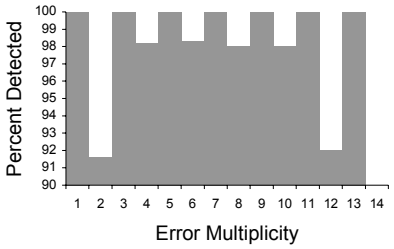

Fig 1. Percent of errors detected for 7 bit linear duplication (*k*=*r*=*7*).

Clearly, duplication is not robust; its detection capability depends largely on the multiplicity and type of the error. The scheme offers relatively poor protection for errors of even multiplicities. Although the overall probability of injecting an undetectable error (assuming all errors are equally likely) is 2^{-r}, which for the *k*=*r*=*7* linear duplication example is $2^{-7} = 0.78\%$, it is deceiving to imply that the method provides for this level of protection. In addition to the overall average protection it is important to note the class of errors with the weakest protection level. As shown in Figure 1, for errors with multiplicity 2, the probability of successfully injecting an undetectable error increases by an order of magnitude.

The above limitations shown for duplication are present in any protection scheme based on linear error-detecting codes. The classes of errors with lower probabilities of detection can serve as weak points for attack, regardless of how good the overall protection is.

The protection method proposed in this paper minimizes the size and weakness of the least protected areas under given limitations imposed by overheads. We propose a protection method based on a class of nonlinear systematic error-detecting codes called robust codes. Robust nonsystematic codes which provide equal probabilities of detection for all error distributions were presented in [15-16]. The methods in this paper are based on systematic robust nonlinear error detecting codes which are similar to those described in [4],[13]. We present a new construction of these codes which uses a multiplicative inverse as the nonlinear transformation. A method of reducing the hardware overhead while preserving much of the robustness is also presented.

These nonlinear codes are robust in terms of having the capability of providing equal protection against all errors. That is, for a completely robust code *C* the probability $Q(e)$ of missing an error *e* should be constant independently of an error, i.e.

$$Q(e) = \frac{|\{w \mid w \in C, w \oplus e \in C\}|}{|C|} = \text{Constant}, \quad e \neq 0. \tag{1}$$

where $w, e \in GF(2^n)$, $C \subseteq GF(2^n)$, $|C|$ is the number of codewords of the code. Additionally, these codes for the same *n* and *k* have fewer undetectable errors than their linear counterparts. As we will see in the next section, for a systematic nonlinear *(n,k)* robust code the number of undetectable errors is 2^{k-r} versus 2^k for a linear code with the same length *n* and same number of redundant bits *r*. The construction and details of these robust codes are discussed in the next section.

5 Systematic Robust Codes

The binary robust codes presented in [4] were constructed using a cubic signature. In this paper we use inversion (multiplicative inverse in the field) as the nonlinear transformations for the signature. The same robust properties observed with the cubic are also observed with the robust codes which use inversion: data dependent error detection, reduction of a number of undetectable errors, and uniform distribution of the error-detecting power.

We will present now a formal description of these codes.

Let *V* be a binary linear *(n,k)* – code with $n \le 2k$ and check matrix $H = [P \mid I]$ with $rank(P) = n - k = r$ over $GF(2)$. Code V can be made into a nonlinear systematic robust code C_V by taking the multiplicative inverse in $GF(2^r)$ of the *r* redundant bits:

$$C_V = \{(x, v) \mid x \in GF(2^k), v = (Px)^{-1} \in GF(2^r)\} \quad \textbf{(2)}$$

where 0^{-1} is defined to be 0.

For the code C_V, error $e = (e_x \in GF(2^k), e_v \in GF(2^r))$ is not detected for data $(x, (Px)^{-1})$ iff

$$(P(x \oplus e_x))^{-1} = (Px)^{-1} \oplus e_v \quad \textbf{(3)}$$

For linear codes an error is either always missed or never missed $(Q(e) \in \{0,1\})$, regardless of data to which the error occurred, and error detection depends only on the error pattern. For these nonlinear codes detection of errors depends not only on the error, as shown in **(3)**, but also on the data to which the error occurred. For these robust codes there are additional classes of errors which are conditionally detected. There is also a redistribution of errors among the new classes of errors.

Table 1 summarizes this redistribution for nonlinear robust codes, C_V, when the data is assumed to be uniformly distributed. The redistribution differs depending on the number of redundant bits *r*. If the code has a signature where the multiplicative inverse is over $GF(2^r)$ where *r* is odd and $r > 2$, then there are 3 different classes

of errors, identical to the nonlinear robust codes based on a cubic signature presented in [4]. When r is even and $r > 2$, there is an additional class of errors which are detected with probability $1 - 2^{-r+2}$.

As Table 1 shows, one very desirable consequence of the addition of inversion to create a robust code is the reduction in the number of undetectable errors. The number of undetectable errors is reduced from 2^k to 2^{k-r}. When *k=r,* all nonzero errors are detectable.

Table 1. Redistribution of errors among the three classes for a linear and a robust code

	Number of errors		
detected with probability of	Linear	Robust with inversion (r is odd)	Robust with inversion (r is even)
0	2^k	2^{k-r}	2^{k-r}
1	$2^n - 2^k$	$2^{n-1} + 2^{k-1} - 2^{k-r}$	$2^{n-1} + 2^{k-1} - 2^{k-r} + 2^k - 2^{k-r}$
$1 - 2^{-r+1}$	0	$2^{n-1} - 2^{k-1}$	$2^{n-1} - 2^{k-1} - 2(2^k - 2^{k-r})$
$1 - 2^{-r+2}$	0	0	$2^k - 2^{k-r}$

The codes described above are capable of providing almost uniform error detection coverage for all errors. For example, if instead of performing simple duplication, the redundant bits are the multiplicative inverse (in $GF(2^r)$) of the *k*-information bits, the detection profile is much more uniform. In contrast to Figure 1, Figure 2.a shows the *k=r=7* robust duplication (codewords are in the form $\left(x, x^{-1}\right)$, $x \in GF(2^7)$). The error detection is much more uniform independently of the type of error that is injected. This kind of error profile is more desirable for security applications, since it provides equal protection regardless of what type of errors are injected.

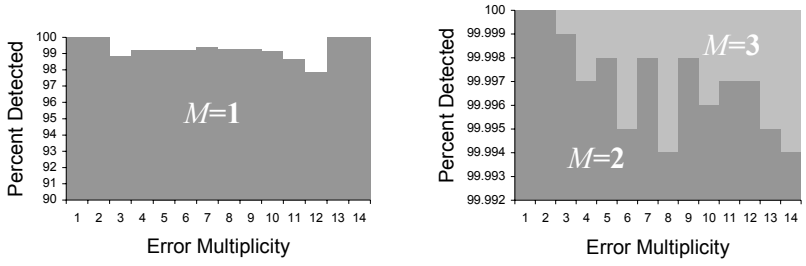

Fig 2. Robust Duplication where *k=r=7* a. detection for *M*=1, b. *M*=2 and *M*=3.

Additionally, as Table 1 shows, for the robust codes there is a class of errors which are conditionally detected. That is, for these errors their detection depends on the data to which the error occurred, and each error in this class is missed for 2^{k-r+1} or 2^{k-r+2} messages. Unlike in the linear case where all errors are either always detected or always missed regardless of the message, the detection of these errors for robust codes is data dependent. If an error of this class is missed for one message, there is a very high probability that it will be detected by the next message. For example, for the robust duplication for any $k=r$, where r is odd, there are at most two messages for which an error is missed. So if the same error is present for three different messages, the error is guaranteed to be detected, regardless of what the error is. More precisely, if $k=r$ and all messages are different, then:

$$\max Q(e) = 2^{-r+1} \quad \text{after } M\text{=1 message}$$

$$\max Q(e) = 2^{-r} \quad \text{after } M\text{=2 messages}$$

$$\max Q(e) = 0 \quad \text{after } M\text{=3 messages}$$

Figure 2.b shows the increased probability of detecting an error after M=2 and M=3 messages for the robust duplication where $k=r=7$.

For the case $k=r$ these systematic robust codes are optimum in terms of providing uniform level of protection against all errors [4]. We note that for any linear code there are always undetectable errors, so $\max Q(e) = 1$ regardless for how many messages the error is present.

6 General Architecture

The method of transforming protection based on a linear code to a more robust protection based on the systematic robust codes involves slight modification of the general linear error-detection architecture.

The general architecture used for protection with linear codes is presented in Figure 3. The architecture is composed of three major hardware components: original hardware, redundant hardware for predicting the r-bit signature v (which is a linear combination of components of the output x of the original device), and an error-detecting network (EDN).

The signature predictor contains the majority of the redundant hardware. The k bits of output of the original hardware and the r redundant output bits of the signature predictor form the $n=k+r$ extended output of the device. The extended output forms a codeword of the systematic *(n,k)* error-detecting code which can be used to detect errors in the original hardware or in the Predictor. It is the EDN which verifies, that the extended output of the device belongs to the corresponding code V, if it does not then the EDN raises an error signal. In a linear protection scheme the predicted r-bit signature v of the Predictor is a linear combination of the k-bit output of the original device. ($v = Px$, where P is a $(r \times n)$- check matrix for the linear *(n,k)* code V used for protection)

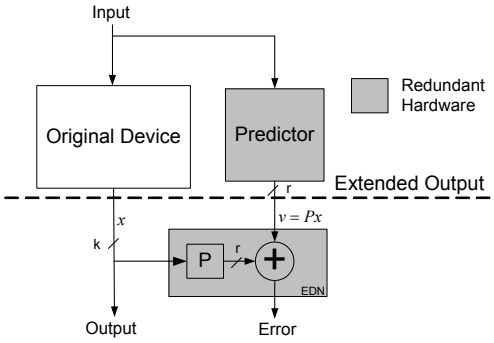

Fig 3. General architecture for protection of hardware with error-detecting codes.

With only a slight modification, the same architecture used for protection with linear error-detecting codes, can be used to provide protection based on the robust systematic nonlinear error-detecting codes presented earlier. The transformation only requires an addition of two copies of one extra component for multiplicative inverse in $GF(2^r)$.

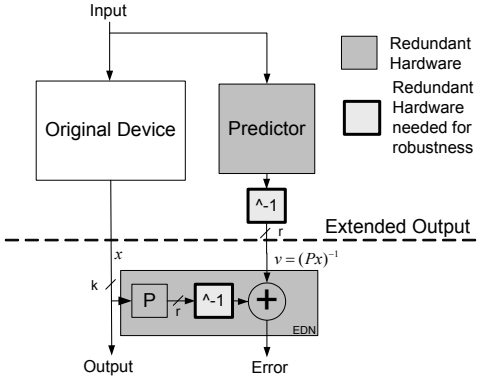

Fig 4. Architecture for protection of hardware with robust error-detecting codes.

The modified architecture is shown in Figure 4. The Extended Output of the device is now protected with the robust nonlinear code with the properties outlined above. An additional (and identical) multiplicative inverse is also needed in the EDN to verify the nonlinear signature. This transformation can be applied to any linear protection method regardless of what algorithm it is protecting.

7 Architectural Optimizations

The method for modifying an architecture based on linear codes into a robust architecture codes requires an overhead for computation of inverses in $GF(2^r)$, which is of the order $O(r^2)$.

Since large *r* may be necessary to provide for a sufficiently high error-detecting probability the use of one large device which takes the multiplicative inverse of all of the *r*-redundant bits might not be practical. Transforming an implementation protected by a linear code with *r*=32 into a robust systematic code would require several thousands additional 2-input gates.

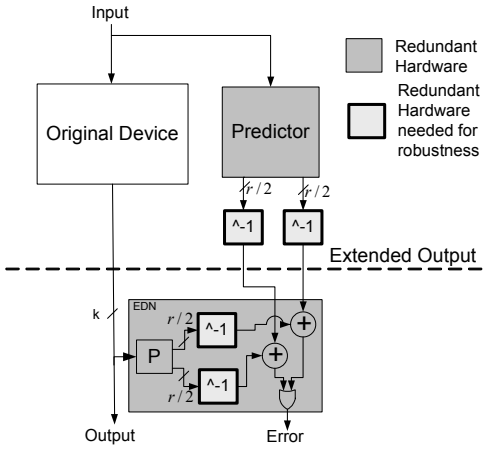

Fig 5. Optimized architecture, the multiplicative inverse in split into *t=2* separate modules.

It is possible to tradeoff the level of robustness for the amount of hardware overhead required to transform linear protection to protection based on systematic robust codes. Instead of taking one multiplicative inverse for all *r*-bit vectors, it is possible to divide the one large inversion into disjoint smaller inversions while retaining many of the robust properties outlined earlier. That is, we can replace multiplicative inverse in $GF(2^r)$ by t s-bit disjoint inverses in $GF(2^{\frac{r}{t}})$ to produce the nonlinear *r* bit output ($r = ts$). Thus, instead of having two *r*-bit multiplicative inverses in $GF(2^r)$ for the whole design, there could be $2t$ inverses in $GF(2^{\frac{r}{t}})$ as it is presented in Figure 5 for *t=2*. Since the number of two input gates to implement the inverse is proportional to the square of the number of bits at its input, a modification where *t=2* would result in roughly 50% decrease of an overhead associated with the architecture based on robust codes. As a consequence this also results in a slight

decrease in the level of robustness and an in introduction of errors which are detected with different probabilities.

The division of the nonlinear signature results in the creation of additional classes of errors which detected with different probabilities depending on the number of divided signatures they affect. To account for this division Table 1 has to be extended. Table 2 shows the redistribution of errors among the additional classes for t s-bit signatures if r is odd, a very similar table can be constructed for the case when r is even.

Table 2. Redistribution of errors as function of the number of blocks t of the signature when r is odd

#of blocks	Number of errors missed with probability p				
	p=1 (undetectable)	p=0 (always detected)	$p=2^{-s+1}$	$p=2^{-2(s+1)}$	$p=2^{-i(s+1)}$
Linear	2^k	$2^{k+r}-2^k$	0	0	0
$t=1$ (robust)	N_1	N_2	N_3	0	0
$t<\frac{r}{2}$ (robust)	$(N_1)^t$	$\sum_{i=1}^{t}\binom{t}{i}N_2{}^i(2^s-N_2)^{t-i}$ $=(2^s)^t-(2^s-N_2)^t$	$tN_3(N_1)^{t-1}$	$\binom{t}{2}(N_3)^2(N_1)^{t-2}$	$\binom{t}{i}(N_3)^i(N_1)^{t-i}$
$t\geq\frac{r}{2}$	2^k	$2^{k+r}-2^k$	0	0	0

where $N_1=2^{\frac{k}{t}-s}$, $N_2=2^{\frac{k+r}{t}-1}+2^{\frac{k}{t}-1}-2^{\frac{k}{t}-s}$, $N_3=2^{\frac{k+r}{t}-1}-2^{\frac{k}{t}-1}$

The splitting of the signature has several effects. Depending on the number of blocks, t, there is a redistribution of errors and a difference in the level of robustness. The maximum robustness is achieved with no divisions when $t=1$. With an increasing number of blocks the robustness of the resulting code is reduced. As the number of blocks, t, increases, the number of undetectable errors increases exponentially. Likewise, the number of classes of errors increases linearly as t increases.

Figure 6 demonstrates the increase of robustness, or uniformity or error coverage, as the number of blocks in a r-bit signature decreases for duplication where $k=r=8$.

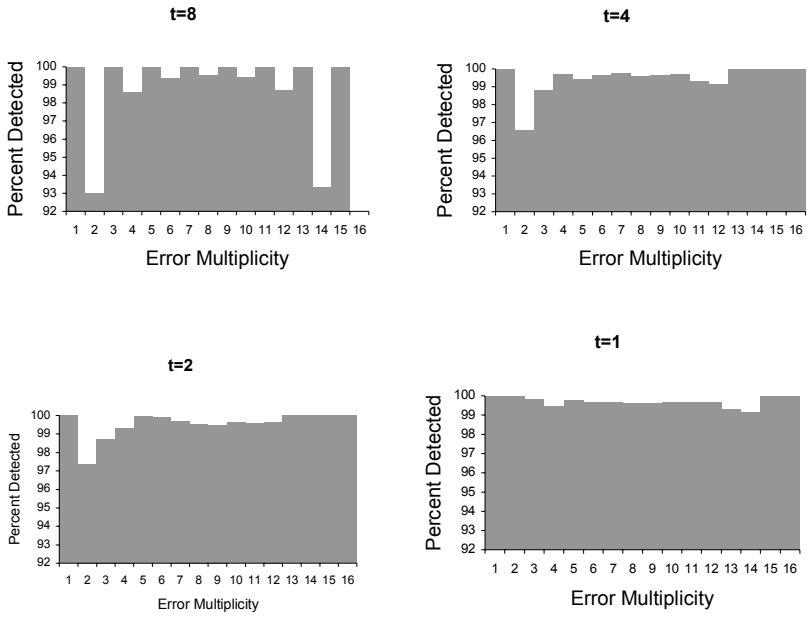

Fig 6. The effect of splitting the *r*-bit signature into *t* disjoint inverses on robustness where $k=r=8$.

As Figure 6 shows the level of robustness, or uniformity of error detection, increases as the number of signature divisions *t* decreases providing a tradeoff between overhead and robustness.

8 Conclusions

The protection provided by linear error detecting codes is not uniform and is not suitable for cryptographic hardware which is susceptible to fault attacks. The level of protection they provide depends largely on the type of error that is considered. We presented a method of protection based on nonlinear systematic robust codes which can provide for uniform protection against all errors thus drastically reducing the probability that an attacker will be able to inject an undetected error. We also presented an optimization which allows for a tradeoff between the level of robustness and area overhead.

The construction of the presented robust codes was based on the use of a multiplicative inverse as the nonlinear transformation. The multiplicative inverse is a building block of the Sbox of the Advanced Encryption Standard. This inverse based

construction of the codes might be useful in further reduction of overhead if the inversion hardware in AES can be used to produce the nonlinear signature.

References

1. E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems", CRYPTO 97, LNCS 1294, pp.513-525
2. FIPS PUB 197: "Advanced Encryption Standard", http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
3. Hagai Bar-El, Hamid Choukri, David Naccache, Michael Tunstall and Claire Whelan. "The Sorcerer's Apprentice Guide to Fault Attacks". Cryptology ePrint Archive, Report 2004/100. Available: http://eprint.iacr.org/2004/100.pdf
4. M.G.Karpovsky and A. Taubin, "A New Class of Nonlinear Systematic Error Detecting Codes", *IEEE Trans Info Theory*, Vol 50, No.8, 2004, pp.1818-1820
5. C.N. Chen and S.M. Yen. "Differential Fault Analysis on AES Key Schedule and Some Countermeasures". ACISP 2003, LNCS 2727, pp18-129, 2003.
6. P. Dusart, G. Letourneux, O. Vivolo, "Differential Fault Analysis on AES". Cryptology ePrint Archive, Report 2003/010. Available: http://eprint.iacr.org/2003/010.pdf
7. C. Giraud. "DFA on AES". Cryptology ePrint Archive, Report 2003/008. Available: http://eprint.iacr.org.
8. J. Blömer and J.P. Seifert. "Fault Based Cryptanalysis of the Advanced Encryption Standard (AES)". Financial Cryptography 2003: pp. 162-181.
9. J.J. Quisquater and G. Piret. "A Differential Fault Attack Technique against SPN Structures, with Application to the AES and KHAZAD". CHES 2003, LNCS 2779, pp 77-88, 2003.
10. R. Karri, K. Wu, P. Mishra, and Y. Kim. "Concurrent Error Detection of Fault Based Side-Channel Cryptanalysis of 128-Bit Symmetric Block Ciphers". *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, Vol.21, No.12, pp. 1509-1517, 2002
11. R. Karri, G. Kuznetsov, M. Gössel. "Parity-Based Concurrent Error Detection of Substitution-Permutation Network Block Ciphers". *In Proc. of CHES 2003*. pp.113-124.
12. G. Bertoni, L. Breveglieri, I. Koren, P. Maistri and V. Piuri. "Error Analysis and Detection Procedures for a Hardware Implementation of the Advanced Encryption Standard". *IEEE Transactions on Computers*, vol. 52, no. 4, 2003
13. M.G Karpovsky, K. Kulikowski, and A. Taubin, "Robust Protection against Fault-Injection Attacks of Smart Cards Implementing the Advanced Encryption Standard". Proc. Int. Conference on Dependable Systems and Networks (DNS 2004), July, 2004
14. M.G. Karpovsky, K. Kulikowski, and A. Taubin, "Differential Fault Analysis Attack Resistant Architectures for the Advanced Encryption Standard". Proc. World Computing Congress, Cardis, Aug., 2004
15. M. G. Karpovsky, P. Nagvajara, "Optimal Robust Compression of Test Responses," IEEE Trans. on Computers, Vol. 39, No. 1, pp. 138-141, January 1990.
16. M. G. Karpovsky, P. Nagvajara, "Optimal Codes for the Minimax Criterion on Error Detection," IEEE Trans. on Information Theory, November 1989.