FPGA IMPLEMENTATIONS OF THE DES AND TRIPLE-DES MASKED AGAINST POWER ANALYSIS ATTACKS

F.-X. Standaert, G. Rouvroy, J.-J. Quisquater

UCL Crypto Group, Place du Levant 3, B-1348 Louvain-la-Neuve, Belgium
e-mails: fstandae,rouvroy,quisquater@dice.ucl.ac.be

ABSTRACT

This paper presents FPGA implementations of the DES and Triple-DES with improved security against power analysis attacks. The proposed designs use Boolean masking, a previously introduced technique to protect smart card implementations from these attacks. We demonstrate that recent reconfigurable devices offer excellent opportunities to implement a masked DES. In particular, we use the large embedded memories available in the Xilinx Virtex-II pro® FPGAs to store precomputed and masked substitution tables. Compared to an unprotected DES design, our proposal only requires 45% more logic resources and 128 Kbit of memory and yields a throughput of about 1 Gbit/sec.

1. INTRODUCTION

Since their publication by Kocher *et al*. in 1998 [5], power analysis attacks have attracted significant attention within the cryptographic community. So far, they have been successfully applied to different kinds of (unprotected) implementations of symmetric and public-key encryption schemes. Although less general than classical cryptanalysis (because they target one specific implementation), power analysis attacks usually present a very serious threat for practical cryptosystems implemented on various platforms. Among the different countermeasures proposed in the literature to protect an implementation from such attacks, one of the most popular is the Boolean masking method. In this proposal, the cryptographic algorithm is modified in such a way that the intermediate data never appears as such, but is always "masked" with random boolean vectors. The masking has been successfully applied to smart card implementations of the DES and the AES Rijndael, *e.g.* in [1, 4]. However, recent works have shown that power analysis attacks are also practical against ASIC and FPGA implementations of cryptographic algorithms, *e.g.* in [12, 17]. A practical problem is therefore to protect these devices.

In this context, one important concern is the implementation cost of the countermeasure. In particular, the protected algorithms usually have much higher memory requirements than the unmasked ones. For this reason, it is often assumed that masking is not a practical solution for the protection of hardware implementations. On the opposite, we demonstrate in this paper that FPGA implementations of the DES offer very simple and interesting opportunities to implement the Boolean masking method. In practice, we propose a secure cryptographic design, based on the use of large embedded memories available inside certain recent FPGAs. As the efficiency of the proposal highly depends on the size of the substitution tables used in the encryption algorithm, it was particularly well-fitted to the DES (and, for example, could not be applied as such to the AES Rijndael). Therefore, our resulting protected DES implementation only requires a moderate additional hardware cost. We note that, as most of the present countermeasures against side-channel attacks, the masking does not provide any perfect security and only makes the attack more difficult. Although the aim of this work is mainly to evaluate the performances of such a protection, we conclude the paper with a brief discussion of security issues and provide references to security evaluations recently applied to FPGAs.

2. DATA ENCRYPTION STANDARD

In 1977, the DES algorithm [10] was adopted as a Federal Information Processing Standard (FIPS) for unclassified government communication. Although a new Advanced Encryption Standard was selected in October 2000 [11], the DES and Triple-DES are still widely used, particularly in the financial sector. DES encrypts 64-bit blocks with a 56-bit key and processes data with permutations, substitutions and XOR operations. Triple-DES simply applies three encryptions with three different keys to the plaintexts

Basically, the plaintext is first permuted by a fixed permutation *IP*. Next the result is split into two 32-bit halves, denoted with *L* (left) and *R* (right) to which a round function f is applied 16 times. The ciphertext is calculated by applying the inverse of the initial permutation *IP* to the result of the 16th round. The secret key is expanded by the key schedule algorithm to sixteen 48-bit round keys K_i and in each round, a 48-bit round key is XORed to the text. The key schedule consists of known bit permutations and shift operations. As a consequence, finding any round key bit directly involves that the secret key is corrupted. Finally, the round function is represented in Figure 1 (a) and is easily described by: $L_{i+1} = R_i,\ R_{i+1} = L_i \oplus f(R_i, K_i)$, where f is a nonlinear function detailed in Figure 1 (b): the R_i part is first expanded to 48 bits with the *E* box, by doubling some R_i bits. Then, it performs a bitwise modulo 2 sum of

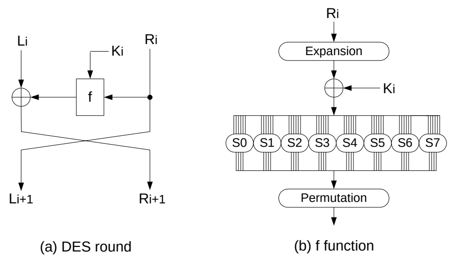


Fig. 1. Data Encryption Standard.

the expanded R_i part and the 48-bit round key K_i. The output of the XOR function is sent to eight non-linear S-boxes, with six input bits and four output bits. The resulting 32 bits are permuted by a bit permutation *P*. DES decryption is the same algorithm with the round keys in reversed order.

3. BOOLEAN MASKING

Boolean masking is a general method to thwart power analysis attacks in which all the intermediate data inside an implementation is "masked" (*i.e.* XORed with random Boolean values), so that the power consumption becomes unpredictable [1, 4]. For the method to be effective in practice, it is necessary that all the block cipher transformations can be applied to the masked data, without being applied to the original data. This is formalized in the following lemmas for bit permutations, XOR operations and S-boxes.

Lemma 1: Let $P : GF(2)^n \rightarrow GF(2)^n$ be a bit permutation and b, b_1, b_2 be three Boolean vectors $\in GF(2)^n$ such that $b = b_1 \oplus b_2$. Then, we have:$P(b) = P(b_1) \oplus P(b_2)$.

Lemma 2: Let b, b_1, b_2, a, a_1, a_2 be six Boolean vectors $\in GF(2)^n$ such that $b = b_1 \oplus b_2$ and $a = a_1 \oplus a_2$. Then, we have: $a \oplus b = (a_1 \oplus a_2) \oplus (b_1 \oplus b_2) = (a_1 \oplus b_1) \oplus (a_2 \oplus b_2)$.

Lemma 3: Let $S : GF(2)^n \rightarrow GF(2)^m$ be a S-box and b, b_1, b_2 be three bit vectors $\in GF(2)^n$ such that $b = b_1 \oplus b_2$. Then, $\exists$ a S-box $S' : GF(2)^{2n} \rightarrow GF(2)^m$ such that: $S(b) = S(b_1) \oplus S'(b_1, b_2)$.

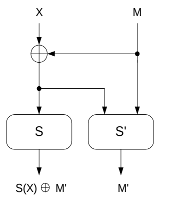
Using these lemmas, it is possible to rewrite the DES algorithm in the masked domain. Lemma 3 indicates that such an implementation requires the use of new S-boxes S'. In practice, the simplest solution is to precompute and store them in embedded memories. It is important to remember that the security of masking relies strongly on the fact that the mask is randomly updated for every new encryption [4].

4. IMPLEMENTATION

4.1. Masking scheme

From the previous section, it is clear that the most critical part of a masked implementation is the S-box. As the mask has to be updated for every encryption, we need one S' for any possible mask. A single masked DES S-box has memory requirements of $2^{12} \times 4 \simeq 16$ Kbit. Although expensive for most devices, these modified S-boxes fit to the recent Xilinx Virtex-II pro® FPGAs that provide embedded dual-port synchronous RAM blocks of 16 Kbit. It means that we can store two precomputed boxes in one RAM block.

A masked S-box is represented in Figure 3 and it is extended to the complete non-linear function f in Figure 4. As the expansion E and permutation P only constitute routing information, it is clear that the additional cost needed to mask the function f lies in the S-box S' only. From these schemes,



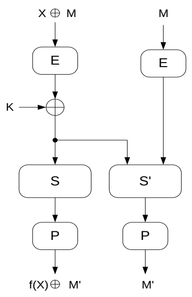


Fig. 3. Masked S-box. **Fig. 4.** Masked non-linear function f.

we can easily protect a complete round of the DES algorithm. Assuming that the round inputs L_i, R_i are already masked with M_{Li}, M_{Ri} and if the non-linear function output mask at round i is denoted as M_i', it leads to the scheme of Figure 5. The round outputs are consequently masked with the following values: $M_{Li+1} = M_{Ri}$, $M_{Ri+1} = M_{Li} \oplus M_i'$. These equations exhibit that, additionally to the round function of Figure 5, the final implementation requires a small Feistel structure to compute the values M_{Li+1}, M_{Ri+1}.

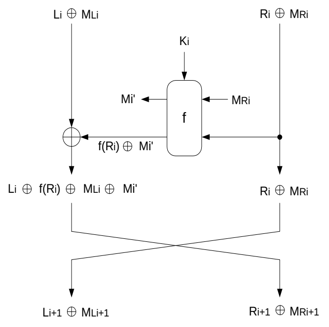


Fig. 5. Masked round function.

4.2. Practical designs

In practice, we decided to implement a loop architecture with a one-cycle-per-round structure. Loop architectures are a relevant choice for investigation because they satisfy the usual area and throughput requirements for block cipher applications. This structure also allows considering encryption modes with feedback and limits the memory requirements for the S' boxes. The round structure and the additional logic needed to update the masks during the 16 encryption rounds are in Figures 6, 7, in which the same

Device	**# LUTs**	**# Regs**	**# Slices**	**# RAMBs**	**Frequency** (MHz)	**Throughput** (Mbit/sec)
XC2vp20	550	350	347	4	207	828

Table 1: Performances of our masked DES implementation.

masked function f is actually used. Finally, we provide the implementation results on the Xilinx Virtex-II pro® technology in Table 1. Synthesis and implementation were performed with Xilinx ISE 6.1®. The frequency is estimated after implementation and the hardware cost is evaluated by the number of LUTs, registers and slices.

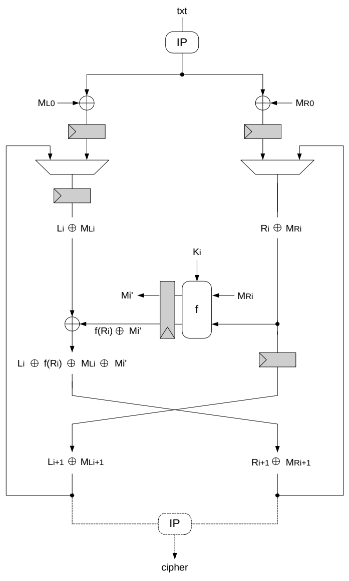


Fig. 6. Masked DES.

5. COMPARISONS

In order to evaluate the additional cost of the masking countermeasure, we implemented exactly the same architecture as the one of Figure 6, without masking. We also compared our results to the best-reported loop architecture of the DES, found in reference [15]. However, this latter result involves a modified mathematical description of the algorithm that was not used in our designs. Therefore, comparisons between similar architectures are probably more relevant. From these results, we observe that a masked implementation of the DES requires additional resources of about 100 FPGA slices and 4 RAM blocks (*i.e.* 128 Kbit of memory). However, the final design remains efficient for most applications and underlines that the DES offers excellent opportunities to implement the Boolean masking countermeasure in Virtex-II pro® FPGAs. Finally, due to the sequential nature of the presented DES implementation, a Triple-DES design is straightforwardly derived. Regarding performances, the resulting designs will involve either a reduced throughput (*i.e.* divided by 3) or larger area requirements (*i.e.* three times more resources). For readability and comparison purposes, we only mentioned the results of our single DES.

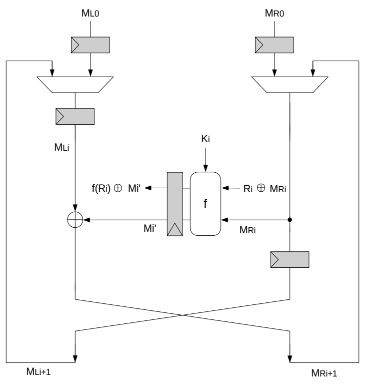


Fig. 7. Mask generation.

6. ADDITIONAL CONCERNS

6.1. Mask generation

In the previous descriptions, we completely neglected the generation process of the random mask values. However, in practice, this strongly influences the security of a masked implementation. The two important constraints regarding the random number generation process are:

• The random masks should be generated within the FPGA.
• The random numbers should be generated from a true random generator in order to avoid attacks targeting a deterministic mask generation process.

Although generating random numbers is a difficult task to perform using digital hardware, there exist several proposals allowing to deal with this problem efficiently, *e.g.* in [2, 3, 16]. Reference [3] is particularly convenient for our application because is does not require the use of an analog Phase-Locked Loop (PLL) as in [2] and is therefore applicable to a wide range of FPGAs, including Xilinx® ones. For this reason, we do recommend it for the generation of the mask values. For efficiency purposes, the true random number generator can also be used to product the initial seeds of a PRNG which will consequently generate the masks.

6.2. More efficient schemes

As a matter of fact, the masking countermeasure implemented in this paper was applied naively, using large substitution tables. It is clear that more efficient solutions could be considered. For example, the masking could be implemented at the gate level, *e.g.* in [20] or use any particular structure in the S-boxes, *e.g.* in [13], and yield smaller memory requirements. The reason why we chose this strategy was because it was perfectly adapted to the size of the memory blocks available inside the Virtex-II pro® devices.

Algorithm	**Device**	**# LUTs**	**# Regs**	**# Slices**	**# RAMBs**	**Frequency** (MHz)	**Throughput** (Mbit/sec)
Masked DES	Virtex-II pro®	550	347	350	4	207	828
Unmasked DES	Virtex-II pro®	390	221	250	0	259	1036
Unmasked DES [15]	Virtex-II ®	365	202	189	0	274	974

Table 2: Performance comparisons on the Xilinx® FPGAs.

6.3. Higher-order and glitch attacks

While masked implementations were showed to be secure against the first order power analysis attacks discussed in this paper, they still can be defeated by higher-order techniques [9]. Basically, a higher-order side-channel attack is based on the use of multiple leakage points in an implementation and/or probabilistic (rather than deterministic) predictions of the leakages. Nevertheless, such attacks require the use of more measurements than a first order attack against an unmasked implementation. As already mentioned, masking an implementation does not perfectly prevent side-channel attacks but makes them more difficult to apply. Examples of higher-order attacks applied to FPGAs can be found in [14]. In addition to the higher-order concern, recent results have shown that the glitching activity within microelectronic circuits potentially provides the side-channel adversary with another leakage allowing to defeat certain masked implementations [7]. To the best of our knowledge, such effects have not yet been observed in the context of FPGA implementations. The use of large memory tables rather than gate-level masks probably has some positive effect with this respect as well. Anyway, this constitutes a scope for further research.

7. CONCLUSION

We presented FPGA implementations of the DES and Triple-DES with improved security against power analysis attacks. Our designs are based on the Boolean masking technique, a previously introduced countermeasure against these attacks. While it is usually assumed that such a solution involves critical performance penalties, we demonstrated that a masked DES can be implemented very efficiently on recent FPGAs. This is mainly due to the availability of large embedded memories within these devices. From a security point of view, it is clear that masking an implementation is not sufficient to prevent side-channel attacks. However, if properly combined with other security elements, it may allow to improve the overall security level of an FPGA implementation. Examples of features to improve the resistance of a circuit against side-channel attacks include:

- The generation of noise within the circuit [6],
- The use of random pre-charges on the data buses [18],
- The randomization of a computation's clock cycles [8],
- The use (or emulation) of logic styles with (close to) constant power consumption [19]

We note finally that the main purpose that this work was to evaluate the performances of a masked design. We refer to previous publications for the precise security evaluations of these different countermeasures.

8. REFERENCES

[1] M.L. Akkar, C. Giraud, *An Implementation of DES and AES Secure againts Some Attacks*, in the proceedings of CHES 2001, Lecture Notes in Computer Sciences, vol 2162, pp 309-318, Paris, France, May 2001.

[2] V. Fischer, M. Drutarovsky, *True Random Number Generator Embedded in Reconfigurable Hardware*, in the proceedings of CHES 2002, Lecture Notes in Computer Science, vol 2523, pp 415-430, Redwood Shores, CA, USA.

[3] P. Kohlbrenner, K. Gaj, *An embedded true random number generator for FPGAs*, in the proceedings of FPGA 2004, pp 71-78, Monterey, USA, Feb 2004.

[4] L. Goubin, J. Patarin, *DES and Differential Power Analysis*, in the proceedings of CHES 1999, Lecture Notes in Computer Science, vol 1717, pp 158-172, Worcester, Massachussets, USA, August 1999, Springer.

[5] P. Kocher, J. Jaffe, B. Jun, *Differential Power Analysis*, in the proceedings of Cypto 1999, Lecture Notes in Computer Science, vol 1666, pp 398-412, Santa-Barbara, USA, August 1999, Springer-Verlag.

[6] S. Mangard, *Hardware Countermeasures against DPA - A Statistical Analysis of Their Effectiveness*, in the proceedings of CT-RSA 2004, Lecture Notes in Computer Science, vol 2964, pp 222-235, San Francisco, USA, 2004.

[7] S.Mangard, *Side-Channel Leakage of Masked CMOS Gates*, in the proceedings of CT-RSA 05, Lecture Notes in Computer Science, vol 3376, pp 351-365, San Fransisco, CA, USA, February 2005.

[8] D. May, H. Muller, N. Smart, *Randomized Register Renaming to Foil DPA*, in the proceedings of CHES 2001, Lecture Notes in Computer Sciences, vol 2162, pp 28-38, Paris, France, May 2001, Springer-Verlag.

[9] T.S. Messerges, *Using Second-Order Power Analysis to Attack DPA Resistant Software*, in the proceedings of CHES 2000, Lecture Notes in Computer Sciences, vol 1965, pp 71-77, Worcester, Massachusetts, USA, August 2000.

[10] National Bureau of Standards, *FIPS PUB 46, The Data Encryption Standard*, Federal Information Processing Standard, NIST, U.S. Dept. of Commerce.

[11] National Bureau of Standards, *FIPS 197, Advanced Encryption Standard*, Federal Information Processing Standard, NIST, U.S. Dept. of Commerce.

[12] S.B. Ors, F. Gurkaynak, E. Oswald, B. Preneel *Power-Analysis Attack on an ASIC AES implementation*, in the proceedings of ITCC 2004, Las Vegas.

[13] E. Oswald, S. Mangard, N. Pramstaller, V. Rijmen, *A Side-Channel Analysis Resistant Description of the AES S-Box*, in the proceedings of FSE 2005, LNCS, vol 3557, pp 413-423, Paris, France, February 2005.

[14] E. Peeters, F.-X. Standaert, N. Donckers, J.-J. Quisquater, *Improved Higher-Order Side-Channel Attacks With FPGA Experiments*, in the proceedings of CHES 2005, LNCS, vol 3659, pp 309-323, Edinburgh, Scotland.

[15] G. Rouvroy, F.-X. Standaert, J.-J. Quisquater, J.-D. Legat, *Design Strategies and Modifed Descriptions to Optimize Cipher FPGA Implementations: Fast and Compact Results for DES and Triple-DES*, in the proceedings of FPL 2003, LNCS, vol 2778, pp 181-193, Lisbon, Portugal, Sept. 2003.

[16] B. Shackleford, M. Tanaka, J. Carter, G.Snider, *FPGA implementation of neighborhood-of-four cellular automata random number generators*, in the proceedings of FPGA 2002, pp 106-112, Monterey, California, Feb 2002.

[17] F.-X. Standaert, S.B. Ors, B. Preneel, *Power Analysis of an FPGA Implementation of Rijndael : Is Pipelining a DPA Countermeasure?*, in the proceedings of CHES 2004, Lecture Notes in Computer Science, vol 3156, pp 30-44, Cambridge, MA, USA, August 2004.

[18] F.-X. Standaert, F. Macé, E. Peeters, J.-J. Quisquater, *Updates on the Security of FPGAs Against Power Analysis Attacks*, in the proceedings of ARC 2006, LNCS, vol 3985, pp 335346, Delft, Netherlands, March 2006.

[19] K. Tiri, M. Akmal, I. Verbauwhede, *A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards*, proceedings of ESSCIRC 2003.

[20] E. Trichina, *Combinational Logic Design for AES SubByte Transformation on Masked Data*, IACR e-print archive, 2003/236, available from http://eprint.iacr.org/2003/236.pdf