ISSN: 2321-7782 (Online)

Volume 2, Issue 2, February 2014

International Journal of Advance Research in Computer Science and Management Studies

Research Article / Paper / Case Study

Available online at: www.ijarcsms.com

True Random Based Differential Power Analysis Countermeasure Circuit for an AES Engine

S. Saravanakumar
PG Scholar
Department of ECE
Kumaraguru College of Technology
Coimbatore - India

Abstract: ***In cryptography, power analysis is a form of side channel attack in which the attacker studies the power consumption of a cryptographic hardware device (such as a smart card, tamper-resistant "black box", or integrated circuit). The attack can non-invasively extract cryptographic keys and other secret information from the device. Differential power analysis (DPA) is a side-channel attack which involves statistically analyzing power consumption measurements from a cryptosystem. Several methods have been proposed overtime to resist the DPA attack, but they largely increase the hardware cost and severely degrade the throughput. In this brief, a security problem based on ring oscillators is resolved by a new architecture with self-generated true random sequence. This paper presents a novel low-transition linear feedback shift register (LFSR) that is based on some new observations about the output sequence of a conventional LFSR. It ensures the safe and secured encryption and decryption method.***

Keywords: Advanced Encryption Standard (AES), cryptography, differential power analysis (DPA), linear feed back shift register (LFSR).

I. INTRODUCTION

The differential power analysis (DPA) attack proposed by Kocher et al. in 1999 has become a serious issue when designing cryptographic circuits. DPA attack can efficiently disclose the top secret key by the power consumption information leaked from cryptographic devices. It has been verified that the secret key of an Advanced Encryption Standard (AES) chip can be release within 10,000 measurements. Accordingly, the DPA resistance has become the most main consideration for hardware-based cryptographic devices. Several methods have been proposed to counteract the DPA attack, each in the algorithm or in the circuit level. Some of them use a data covering method to randomize the data processed in cryptographic circuits .The data being processed is changed by an inside generated random mask before the en-/decryption process. As a result, a corresponding mask should be used to improve the actual output data at the end of the process. This way, the power consumption of cryptographic circuits will be separate of the predicted power consumption. Some proposal balance the power consumption of several operations by using new logic cells called sense amplify based logic or wave dynamical differential logic (WDDL).Standard cells are replace by this new logic family and then the power consumption of several patterns would be approximately the same. Some proposals isolate the power supply and cryptographic circuits by switching capacitors .The current is charged to a capacitor array, and the current consumed by cryptographic circuits is then supplied by the capacitor array instead of the power supply However, the expand security level results in extra hardware cost and throughput degradation. For example, the WDDL method can increase the security with 3 times bigger silicon area and 75% throughput degradation The switching capacitor method can decrease the area overhead to 27%, but the presentation is still degraded by 50% .A ring-oscillator-based DPA countermeasure circuit can successfully reduce the area overhead and throughput degradation .Details of the ring-oscillator-based same after the system is reset. Therefore, the extra power consumption added by the DPA countermeasure circuit in each cycle would be the same if the attacker resets the system before copy power traces. To solve problem in a several architecture that incorporates a true random number generator was proposed not only to counteract the DPA attack but also to self-generate a true random sequence. With the proposed architecture, the safety level of AES engines can be further improved while the area overhead can be also reduced problem in a different architecture that incorporates a true random number generator is proposed not only to information of the ring-oscillator based DPA countermeasure circuit such as inversion stages.

II. DPA ATTACK

The DPA attack utilizes the statistical analysis to calculate the connection between the leaked power information and the predicted power consumption. Irrelative noises can be eliminated by statistical analysis and therefore, the DPA attack can still be successfully conducted even in a noisy environment. The secret key of a cryptographic circuit can be disclosed from the correlation index of the analysis result. For the AES algorithm, the 128-bit secret key can be divided into 16 8-bit sub keys, and the attacker can disclose each 8-bit sub key at one time. As a result, the array would consists of 2^8 = 256 columns for all key hypotheses. After the measured and the predicted power arrays are available, the secret key can be disclosed by the statistical analysis. Each column of the predicted power array is used to find a correlation index with every column of the measured power array. If the key hypothesis matches the secret key used by the cryptographic circuit, the correlation index would be higher than that of other key hypotheses.

III. DPA COUNTERMEASURE CIRCUIT

The true random-based architecture is introduced first and then the improved architecture with self-generated random sequence is presented.

3.1 True Random-Based DPA Countermeasure Circuit

To solve the security weakness in the pseudo random-based architecture, a true random sequence for the DPA countermeasure circuit is required. However, most true random number generators are analog circuits with much higher power consumption. Goli proposed a digital method to generate random data by using ring oscillators in Fibonacci and Galois configurations .The Fibonacci and the Galois ring oscillator consists of a series of inverters connected with feedback polynomial. The proposed architecture incorporates a true random number generator into the DPA countermeasure circuit to resist the DPA attack and the reset problem mentioned earlier. The combination of two FiLFSRs and two GaLFSRs is used as the random source to generate one random sequence. In order to generate eight independent random bits for each data byte, a total of 32 ring oscillators (including Fibonacci and Galois LFSRs) are required in the DPA countermeasure circuit. These sixteen random sources are sampled by flip-flops for further post processing input is obtained by XOR one data byte with a random mask, and 16 s are directly controlled by this16-bit input. The random mask can be. After post processing, these 16 random bits are XOR with data bytes from the cryptographic circuit to dynamically enable LFSR in the FiLFSR and GaLFSR. The FiLFSR and GaLFSR now work not only as random sources into generate random data but also as the digitally controlled linear feedback shift register in to counteract the DPA attack. The FiLFSR will not have a fixed point if and only if f(x) = (1 + x) h(x) and h (1) = 1, where f(x) is the polynomial presentation of the feedback configuration for FiLFSR, and h(x) is a primitive polynomial. Note that a fixed point is a state that the output vector of inverters is an alternating string of 1 and 0 ({01010 · · ·} or {10101 · · ·}). Since each random source is from the combination of four different ring oscillators, at least four different h(x) are required. To have four different forms of h(x), the minimum degree of f(x) for the FiLFSR is 6. Similarly, the condition for the GaLFSR having no fixed point, is f(1) = 0, and the degree of f(x) must be odd Again, in order to have four different configurations, the minimum degree of f(x) for GaLFSRs must be 7.The post processing circuits are composed of LFSRs with different initial seeds. The purpose of the post processing circuit is to remove the bias of the random source. In each post processing circuit, the feedback value is XOR with that from the random source. In this way, even the post processing circuit starts from a deterministic state after the system is reset, the generated random sequence would not be the same because the random source is added into the feedback of the LFSR. The means show that the random sequence would be VDD/2, and the standard deviations. The countermeasure circuit consists of 12 ring oscillators, each of which can be enabled or disabled independently. When a ring oscillator is generated by an internally designed random number generator, whose randomness dominates the DPA resistance of our proposed countermeasure circuit. The remaining four oscillators are controlled by pairs of these 16 inputs. The post processing circuits are composed of LFSRs with different initial seeds. The purpose of the post processing circuit is to remove the bias of the random source. In each post processing circuit, the feedback value is XOR with that from the random source. In this way, even the post processing circuit starts from a deterministic state after the system is reset, the generated random sequence to remove the bias of the random source. In each post processing circuit, the feedback value is XOR with that from the random source. In this way, even the post processing circuit starts from a deterministic state after the system is reset, the generated random sequence sequences generated with the proposed architecture, although the standard deviations are zero in the first few cycles, which means the generated bits in these cycles would be always the same after the system is reset.

Linear Feedback Shift Registers (LFSR) is a collection of cyclic binary states where the current state is a direct computation of its predecessor. A simple XOR of particular bits (the tap positions), and a shifting behaviour allows for a uniform serial computation until the start state repeats. The length of unique states depends on the tap positions that are used to create the 'feedback' bit. If the tap positions are maximal, then there are possible states, spanning all non-zero bit binary numbers. The zero state is not allowed in LFSR because it would infinitely return the zero state since XOR of any number of zeros will always return zero.

3.1.1. Fibonacci LFSRs

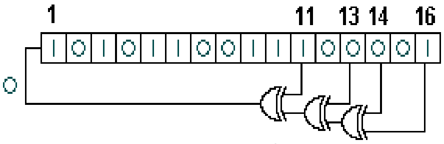


Figure.1.Block Diagram of Fibonacci LFSR

Figure.1. shows the block diagram of a 16-bit Fibonacci LFSR. The feedback tap numbers in white correspond to a primitive polynomial in the table so the register cycles through the maximum number of 65535 states excluding the all-zeroes state. The state shown, 0xACE1 (hexadecimal) will be followed by 0x5670.

3.1.2. Galois LFSRs

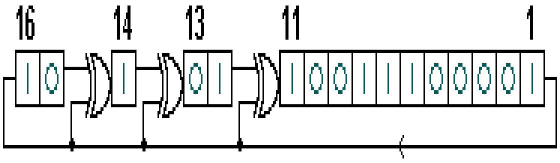
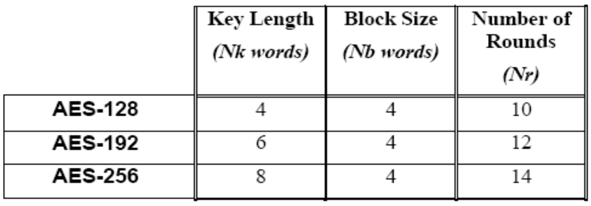


Figure.2. Block Diagram of Galois LFSR

Figure.2. shows the block diagram of the 16-bit Galois LFSR. The register numbers in white correspond to the same primitive polynomial. This register also cycles through the maximal number of 65535 states excluding the all-zeroes state. The state ACE1 hex shown will be followed by E270 hex.

IV. AES ALGORITHM

A symmetric block cipher that can process data blocks of 128 bits, using cipher keys with lengths of 128, 192, and 256 bits. The input and output for AES algorithm each consist of sequences of 128 bits. The Cipher is described in the pseudo code in the individual transformations, SubBytes, ShiftRows, MixColumns, and AddRoundKey.



	Key Length (*Nk words*)	Block Size (*Nb words*)	Number of Rounds (*Nr*)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

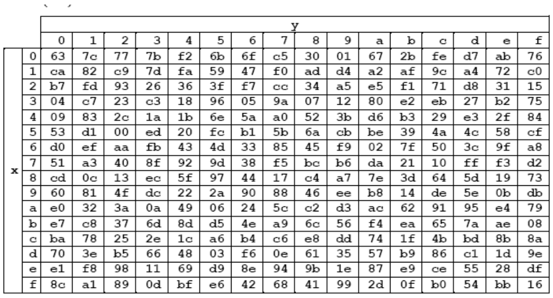
Figure 4. Key-Block-Round Combinations.

4.1. Encryption

To implement the AES encryption algorithm, we proceed exactly the same way as for the key expansion, that is, we first implement the basic helper functions and then move up to the main loop. The functions take as parameter a state, which is, as already explained, a rectangular 4x4 array of bytes. We won't consider the state as a 2-dimensional array, but as a 1-dimensional array of length 16.

4.1.1. SubBytes Transformation

The SubBytes transformation is a non-linear byte substitution that operates independently on each byte of the State using a substitution table (S-box). Take the multiplicative inverse in the finite field GF(28), The S-box used in the SubBytes() transformation is presented in hexadecimal form in Fig. For example, if = 1,1 s {53}, then the substitution value would be determined by the intersection of the row with index '5' and the column with index '3'. This would result in s'1,1 having a value of {ed}.



		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Figure. S-box: substitution values for the byte xy (in hexadecimal format).

4.1.2 ShiftRows Transformation

ShiftRows transformation, the bytes in the last three rows of the State are cyclically shifted over different numbers of bytes.

4.1.3 MixColumns Transformation

The MixColumns transformation operates on the State column-by-column, treating each column as a four-term polynomial. The columns are considered as polynomials.

4.1.4 AddRoundKey Transformation

In this, a Round Key is added to the State by a simple bitwise XOR operation. It will consume additional power to change the power consumption characteristic.

4.2. Decryption

If you managed to understand and implement everything up to this point, you shouldn't have any problems getting the decryption to work either. Basically, we inverse the whole encryption and apply all the operations backwards. As the key schedule stays the same, the only operations we need to implement are the inversed SubBytes, ShiftRows and MixColumns, while addRoundKey stays the same.

4.2.1. Inverse Cipher

The Cipher transformations can be inverted and then implemented in reverse order to produce a straightforward Inverse Cipher for the AES algorithm.

4.2.2. InvShiftRows Transformation

InvShiftRow is the inverse of the ShiftRows transformation. The bytes in the last three rows of the State are cyclically shifted over different numbers of bytes (offsets). The first row, r = 0, is not shifted.

4.2.3. InvSubBytes Transformation

InvSubBytes is the inverse of the byte substitution transformation, in which the inverse S-box is applied to each byte of the State.

4.2.4. AddRoundKey Transformation

In this, a Round Key is added to the State by bitwise XOR operation.

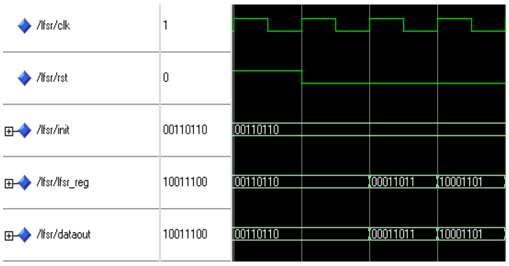
4.2.5. InvMixColumns Transformation

InvMixColumns is inverse of MixColumns transformation. It operates column by column, treating each column as four term polynomial.

V. RESULTS AND ANALYSIS

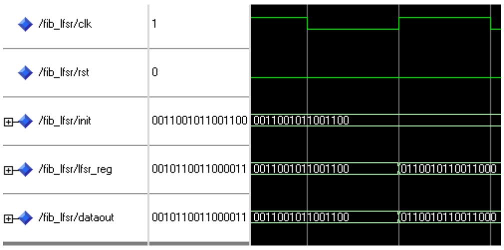
5.1 LFSR Result

The 128 bit input is given to LFSR. The shifted bit is shown as below.



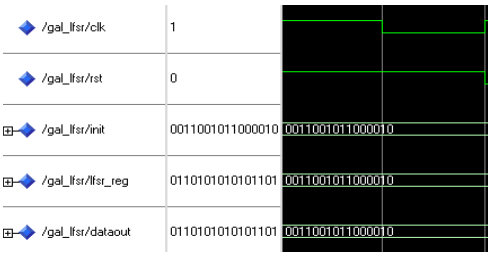
5.2 Fibonacci LFSR Result

Fibonacci LFSR is an advanced LFSR. It can shift 16 bits at the same type.



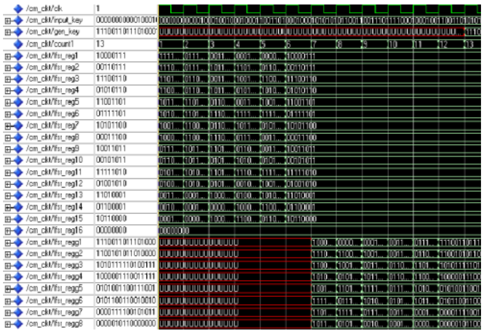
5.3. Galois LFSR Result

Galois LFSR is also an advanced LFSR it also shift 16 bit at the same type.


/gal_lfsr/clk 1
/gal_lfsr/rst 0
/gal_lfsr/init 0011001011000010
0011001011000010
/gal_lfsr/lfsr_reg 0110101010101101
0011001011000010
/gal_lfsr/dataout 0110101010101101
0011001011000010

5.4 Counter measure circuit

Counter measure circuit merges the output of the result of the Fibonacci and Galois LFSR result.

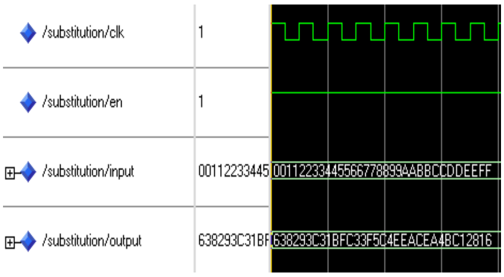


```
/cm_ckt/clk         1
/cm_ckt/input_key   0000000000010001
/cm_ckt/gen_key     1110011011101000
/cm_ckt/count1      13
/cm_ckt/lfsr_reg1   10000111
/cm_ckt/lfsr_reg2   00110111
/cm_ckt/lfsr_reg3   11100110
/cm_ckt/lfsr_reg4   01010110
/cm_ckt/lfsr_reg5   11001101
/cm_ckt/lfsr_reg6   01111101
/cm_ckt/lfsr_reg7   10101100
/cm_ckt/lfsr_reg8   00011100
/cm_ckt/lfsr_reg9   10011011
/cm_ckt/lfsr_reg10  00101011
/cm_ckt/lfsr_reg11  11111010
/cm_ckt/lfsr_reg12  01001010
/cm_ckt/lfsr_reg13  11010001
/cm_ckt/lfsr_reg14  01100001
/cm_ckt/lfsr_reg15  10110000
/cm_ckt/lfsr_reg16  00000000
/cm_ckt/lfsr_regg1  1110011011101000
/cm_ckt/lfsr_regg2  1100101011010000
/cm_ckt/lfsr_regg3  1010111110100111
/cm_ckt/lfsr_regg4  1000001110011111
/cm_ckt/lfsr_regg5  0101001100111001
/cm_ckt/lfsr_regg6  0101100110010010
/cm_ckt/lfsr_regg7  0000111100101011
/cm_ckt/lfsr_regg8  0000010110000000
```

5.5 AES Algorithm Encryption Results

5.5.1. SubBytes Transformation Result

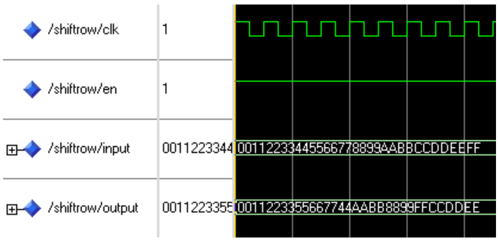
Subbytes transformation result is obtained by applying 128 bit input to S box. Depending on the input a value is selected from the S box.



```
/substitution/clk      1
/substitution/en       1
/substitution/input    00112233445   00112233445566778899AABBCCDDEEFF
/substitution/output   638293C31BF   638293C31BFC33F5C4EEACEA4BC12816
```

5.5.2. ShiftRow Transformation Result

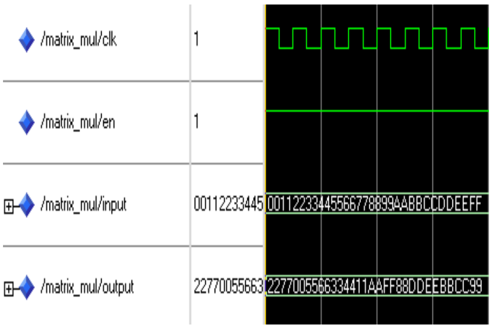
In shiftrow transformation there are four rows. In which there is no change in first row and one change in second row and two change in third row and three change in fourth row.



```
/shiftrow/clk      1
/shiftrow/en       1
/shiftrow/input    0011223344   00112233445566778899AABBCCDDEEFF
/shiftrow/output   0011223355   00112233556677 44AABB8899FFCCDDEE
```

5.5.3. Mixed Column Transformation Result

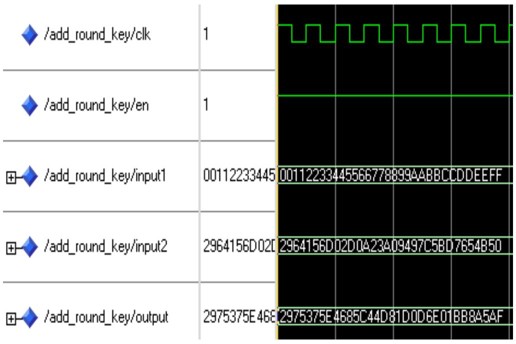
The mixed transformation result is obtained by multiplying a 128 bit matrix to a constant matrix .



```
/matrix_mul/clk      1
/matrix_mul/en       1
/matrix_mul/input    00112233445   00112233445566778899AABBCCDDEEFF
/matrix_mul/output   22770055663   22770055663344 11AAFF88DDEEBBCC99
```

5.5.4. AddRoundKey Transformation Result

This result is obtained by xor of the result of mixed column transformation and 128 bit input.

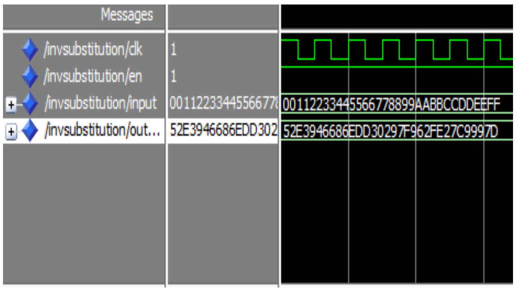


```
/add_round_key/clk      1
/add_round_key/en       1
/add_round_key/input1   00112233445  00112233445566778899AABBCCDDEEFF
/add_round_key/input2   2964156D02D  2964156D02D0A23A09497C5BD7654B50
/add_round_key/output   2975375E468  2975375E4685C44D81D0D6E01BB8A5AF
```

5.6 AES Algorithm Decryption Results

5.6.1. Inverse Subbyte Transformation Result

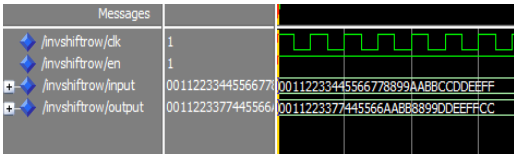
This result is obtained by the inverse operation of subbyte operation.



```
Messages
/invsubstitution/clk      1
/invsubstitution/en       1
/invsubstitution/input    00112233445566778  00112233445566778899AABBCCDDEEFF
/invsubstitution/out...   52E3946686EDD302   52E3946686EDD30297F962FE27C9997D
```

5.6.2. Inverse Shiftrow Transformation Result

This result is obtained by the inverse operation of shift row transformation.



```
Messages
/invshiftrow/clk      1
/invshiftrow/en       1
/invshiftrow/input    00112233445566778  00112233445566778899AABBCCDDEEFF
/invshiftrow/output   0011223377445566A  0011223377445566AABB8899DDEEFFCC
```

VI. Conclusion

In the proposed method a security is provided for the AES algorithm by counter measure circuit. This method provides higher security. This method is extended by bit swapping LFSR and it is used in countermeasure circuit and this method reduces the area and power consumption.

References

1. M. Alioto, L. Giancane, G. Scotti, and A. Trifiletti, "Leakage Power Analysis Attacks: a Novel Class of Attacks to Nanometer Cryptographic Circuits," IEEE Trans. Circuits and Systems, part I, vol. 57, no. 2, pp. 355-367, Feb. 2010.
2. M. Alioto, M. Poli, S. Rocchi, "A General Power Model of Differential Power Analysis Attacks to Static Logic Circuits," IEEE Trans. VLSI Systems, vol. 18, no. 5, pp. 711-724, May 2010.
3. C. Tokunaga and D. Blaauw, "Securing encryption systems with a switched capacitor current equalizer," IEEE J. Solid-State Circuits,vol. 45, no. 1, pp. 23–31, Jan. 2010.
4. C. Tokunaga and D. Blaauw, "Securing encryption systems with aswitched capacitor current equalizer," IEEE J. Solid-State Circuits,vol. 45, no. 1, pp. 23–31, Jan. 2010.
5. A. Schuster and E. Oswald, "Differential Power Analysis of an AES Implementation," SCA-Lab technical report series, www.iaik.tu-graz.ac.at/research/sca-labindex.php , 2009.
6. MIPS Technologies Inc., http:/www.mips.com, 2009.
7. D. Hwang, K. Tiri, A. Hodjat, B.-C. Lai, S. Yang, P. Schaumont, and I. Verbauwhede, "AES-based security coprocessor IC in 0.18- µmCMOS with resistance to differential power analysis side-channel attacks," IEEEJ. Solid-State Circuits, vol. 41, no. 4, pp. 781–792, Apr. 2006.

8. G.F. Bouesse, M. Renaudin, and S. Dumont, "DPA on Quasi Delay Insensitive Asynchronous Circuits: Formalization and Improvement,"Proc. Design, Automation and Test in Europe (DATE '05), pp. 424-429, 2005.

9. E. Trichina, T. Korkishkoand, and K. H. Lee, “Small size, low power, sidechannel-immune AEScoprocessor: Design and synthesis results,” in Proc.AES, vol. 3373, Lecture Notes in Computer Science, 2005, pp. 113–127.

10. J. Irwin and D. Page, "Using Media Processors for Low-Memory AES Implementation," Proc. IEEE Int. Conf. Appl.-Specific Systems, Architectures, Processors, pp. 144-154, June 2003.

11. K. Tiri, D. Hwang, A. Hodjat, B. Lai, S. Yang, P. Schaumont, and I. Verbauwhede, "A Side-Channel Leakage Free Coprocessor IC in 0.18mm CMOS for Embedded AES-Based Cryptographic and Biometric Processing," Proc. Design Automation Conf. (DAC 2005), pp. 222-227, 2005.

12. den Hartog and E.P. de Vink, "Virtual Analysis and Reduction of Side-Channel Vulnerabilities of Smartcards," Proc. Second Int'l Workshop Formal Aspect of Security and Trust (FAST '04), pp. 85-98, 2004.

13. P. Kocher, J. Jaffe, and B. Jun, “Differential power analysis,” in Proc. 19thAnnu. Int. Cryptology Conf. Adv. Cryptology, 1999, pp. 388–397.

14. M.-L. Akkar and C. Giraud, “An implementation of DES and AES, secureagainst some attacks,” in Proc. CHES, 2001, pp. 309–318.

15. D. Suzuki, M. Saeki, and T. Ichikawa, “Random switching logic: Acountermeasure against DPA based on transition probability,” CryptologyePrint Archive, Rep. 2004/346, 2004. [Online]. Available:http://eprint.iacr.org