Highly secure and low overhead countermeasures against side-channel attacks

Pammu, Ali Akbar

2019

Pammu, A. A. (2019). Highly secure and low overhead countermeasures against side-channel attacks. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/83538>

<https://doi.org/10.32657/10220/48023>

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

Highly Secure and Low Overhead Countermeasures against Side-Channel Attacks

Ali Akbar Pammu

School of Electrical & Electronic Engineering

2018

Highly Secure and Low Overhead Countermeasures against Side-Channel Attacks

Ali Akbar Pammu

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Doctor of Philosophy

2018

STATEMENT OF ORIGINALITY

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarized materials, and has not been submitted for a higher degree to any other University or Institution.

1 April 2019

.....

Date

.....

ALI AKBAR PAMMU

SUPERVISOR DECLARATION STATEMENT

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

1 April 2019

.....

Date

.....

Assoc Prof. Gwee Bah Hwee

AUTHORSHIP ATTRIBUTION STATEMENT

This thesis contains material from two journal papers, one conference paper and one technical disclosure published in the following peer-reviewed journal where I was the first author for the journal and conference paper and corresponding author for the technical disclosure.

Chapter 3 is published as **A. A. Pammu**, K. S. Chong, Yi Wang and B. H. Gwee, "A Highly Efficient Side Channel Attack with Profiling through Relevance Learning on Physical Leakage Information," *IEEE Transaction on Dependable Secure Computing*, 2018, accepted for publication, July 2018.

The contributions of the co-authors are as follows:

- Assoc Prof Gwee Bah Hwee provided the initial project direction, edited the manuscript draft and the flow of the paper.
- I prepared the manuscript drafts. The manuscript was revised by Dr. Chong Kwen Siong and Dr. Yi Wang.
- I initiated the idea and further developed it into concrete idea with Dr. Chong Kwen Siong and performed all the experiment and the laboratory work at the School of Electrical and Electronic Engineering.
- I analyzed the data collected during the experiments and presenting them in the paper properly.
- Assoc Prof Gwee Bah Hwee, Dr. Chong Kwen Siong and Dr. Yi Wang strongly supported during the reply for the reviewer comments.

Chapter 4 is published as **A. A. Pammu**, W. Ho, N. K. Z. Lwin, K. Chong and B. Gwee, "A High Throughput and Secure Authentication-Encryption AES-CCM Algorithm on Asynchronous Multicore Processor," in *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 1023-1036, April 2019.

The contributions of the co-authors are as follows:

- Assoc Prof Gwee Bah Hwee provided the initial project direction, edited the manuscript draft and the flow of the paper.
- I prepared the manuscript drafts. The manuscript was revised by Dr. Chong Kwen Siong and Dr. Ho Weng Geng.
- I initiated the idea and further developed it into concrete idea with Dr. Chong Kwen Siong and performed all the experiment and the laboratory work at the School of Electrical and Electronic Engineering.
- I analyzed the data collected during the experiments and presenting them in the paper properly.
- Mr Ne K. Z. Lwin assisted me for the experimental set up and data collection during the experiment
- Assoc Prof Gwee Bah Hwee, Dr. Chong Kwen Siong and Dr. Ho Weng Geng strongly supported during the reply for the reviewer comments.

Chapter 5 is published as **A. A. Pammu**, K. S. Chong and B. H. Gwee, "Highly secured state-shift local clock circuit to countermeasure against side channel attack," *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD-USA, 2017, pp. 1-4.

The contributions of the co-authors are as follows:

- Assoc Prof Gwee Bah Hwee provided the initial project direction, edited the manuscript draft and the flow of the paper.
- I prepared the manuscript drafts. The manuscript was revised by Dr. Chong Kwen Siong.
- I initiated the idea and further developed it into concrete idea with Dr. Chong Kwen Siong and performed all the experiment and the laboratory work at the School of Electrical and Electronic Engineering.
- I analyzed the data collected during the experiments and presenting them in the paper properly.
- Assoc Prof Gwee Bah Hwee and Dr. Chong Kwen Siong strongly supported during the reply for the reviewer comments.

Chapter 6 is filed as K.S. Chong, B.H. Gwee and **A. A. Pammu**, “Hardware Security to Countermeasure Side-Channel Attacks,” PCT Patent Applications and Singapore Patent Application No. 10201702226R, 19 Mar 2018.

The contributions of the co-authors are as follows:

- Assoc Prof Gwee Bah Hwee provided the initial project direction, edited the manuscript draft and the flow of the paper.
- I initiated the idea and further developed it into concrete idea with Dr. Chong Kwen Siong and performed all the experiment and the laboratory work at the School of Electrical and Electronic Engineering.

1 April 2019

.....
Date

.....
ALI AKBAR PAMMU

ACKNOWLEDGEMENT

First of all, I would like to express my sincere gratitude to my supervisor, Assoc. Prof. Gwee Bah Hwee for his professional guidance in technical writing/presentation. I also would like to express gratitude to my senior researcher, Dr. Chong Kwen Siong for his full encouragement/support throughout the years of my postgraduate study.

I also would like to thank my colleagues, Dr. Ho Weng Geng and Mr. Ne Kyaw Zwa Lwin for their kind help and support in technical knowledge/skills.

Last but not least, I would like to thank my family for their patience and supports throughout my postgraduate study.

Special thanks and gratitude to my wife

Dr. Rafiqah Nurdin

for strong support and dedication throughout my PhD

to my sons,

Ahmad Syafii Alfatih Ali (Fii)

Abdunnaafii Aziz Al Ayyubi Ali (Ayyub)

The future is yours, aim higher than this achievement

ABSTRACT

Side-Channel Attack (SCA) is an emerging technique to reveal the secret key of crypto-devices by analyzing the processed-data and their Physical Leakage Information (PLI). The crypto-devices refer to electronic devices such as smartphone, smartcards, wearable devices, and Internet of Things (IoT) implemented with cryptographic algorithms. The processed-data can either be the plaintext or the ciphertext. The PLI includes the power dissipation, electromagnetic interference, acoustic, temperature, and timing information of the crypto-device. The Correlation based Attack (CbA) is the most commonly used SCA. In the CbA, the correlation between the processed-data and the PLI is analyzed to reveal the secret key. In contrast, to secure the secret key against the CbA, countermeasure techniques can be applied to reduce the correlation between the processed-data and the PLI.

This thesis pertains to the investigation and design of the highly efficient SCA which is meant to efficiently evaluate the SCA resistance of the proposed countermeasure techniques in the crypto-device. Thereafter, the proposed countermeasure techniques are implemented with emphases on high resilience against SCA and low power/area overhead. Four main research works have been proposed in this thesis including a highly efficient SCA and three highly secured countermeasures against SCA. The four main research works are explained as follows:

First, we proposed a Profiling through Relevance-Learning (PRL) technique on PLI to extract the highly correlated PLI with processed-data, as to achieve a highly efficient yet effective SCA when evaluating the countermeasure techniques. The highly correlated PLI is extracted using the k -nearest-neighbor, k -NN variance clustering by clustering the high variance sampling points and

discarding the low variance sampling points of PLI measurements. Hence, the number of sampling points of PLI is significantly reduced for SCA computation. Based on the experimental results, our proposed PRL technique successfully reduces 98.19% of PLI measurements. When comparing with reported techniques which require $>10^6$ PLI measurements to reveal the secret key, our proposed PRL requires 523 PLI measurements, which is $\sim 2,000\times$ more efficient in evaluating the countermeasure techniques against SCA.

Second, we proposed a highly-secured State-shift Local Clock (SsLC) countermeasure technique to secure the PLI against CbA for low power and medium speed applications. It embodies a Finite State Machine (FSM) to regularly shift the timing operations of the cryptographic algorithm implementations and hence randomizing the PLI. The experimental results show that the embodiment of the FSM can widely distribute the PLI in time domain and hence reduces the correlation between the PLI and processed-data. Furthermore, the SCA resistant of the proposed SsLC is significantly improved as the required traces for both PLI power and EM measurements have increased by $>18\times$ and $>25\times$, respectively.

Third, we proposed an Authentication-based Matrix-transformation cum Parallel-encryption implemented on an asynchronous Multicore Processor (AMP-MP) to achieve a high throughput and yet secure Advanced Encryption Standard based on Counter with Chaining Mode (AES-CCM). The matrix multiplication in $GF(2^8)$ computation is adopted to transform each message of 16 plaintexts into 1 plaintext, hence improving the authentication speed by $32\times$ collectively at the transmitter and receiver. Our proposed AMP-MP is realized on an 8-bit asynchronous 9-core processor fabricated based on 65nm CMOS process. The experimental results show that the throughput of the authentication and encryption is 8.32Gbps, which is $70\times$ faster than the reported

counterparts. When performing the SCA on our proposed AMP-MP, the secret key is unrevealed at 5×10^5 traces, which is $\sim 17 \times$ more secured than the standard ASIC AES-CCM implementation.

Fourth, we proposed a highly secured Nano AES encryption with Controllable Time Shuffling (CTS) for small form-factor, ultra-low power and medium speed Biomedical/IoT applications. The Nano AES encryption with CTS is performed on byte-based encryption which requires $21 \times$ more clock cycles for each round operation. Based on the experimental results on FPGA, the Nano AES features a gate count of 1,745 which is $5.21 \times$ of gate count reduction and $8 \times$ power dissipation reduction when comparing with the conventional AES implementation. The SCA resistance is increased by $9 \times$ when compared with the conventional AES as the secret key is unrevealed with $> 10^6$ PLI measurements.

TABLE OF CONTENTS

Statement of Originality	i
Supervisor Declaration Statement.....	ii
Authorship Attribution Statement	iii
Acknowledgement.....	vi
Abstract	viii
List of Figures.....	xv
List of Tables	xix
List of Author's Publications	xx
Abbreviations	xxii
List of Notations	xxiv
CHAPTER 1 INTRODUCTION	1
1.1 Motivation	1
1.2 Objectives	4
1.3 Contributions	6
1.4 Organization of the thesis	7
CHAPTER 2 LITERATURE REVIEW	9
2.1 Cryptographic Algorithms	9
2.1.1 Asymmetric Encryption Algorithms	12
2.1.2 Symmetric Encryption Algorithms	15
2.2 Side Channel Attack (SCA)	20
2.2.1 Simple Power Analysis (SPA)	21
2.2.2 Differential Power Analysis (DPA).....	21
2.3 Countermeasure against SCA	24
2.4 Masking	26

2.4.1. Software Implementation	26
2.4.2. Hardware Implementation.....	27
2.5 Hiding	27
2.5.1 System Level.....	27
2.5.2 Cell-Level.....	35
2.5.3 Hybrid Countermeasure	39
2.5.4 Multicore Implementation	40
2.6 Summary	40
 CHAPTER 3 A HIGHLY EFFICIENT SIDE CHANNEL ATTACK WITH PROFILING THROUGH	
RELEVANCE-LEARNING ON PHYSICAL LEAKAGE INFORMATION.....	40
3.1 Introduction	40
3.2 Proposed profiling through relevance-learning on physical leakage information	45
3.2.1 k-NN Variance Clustering.....	47
3.2.2 Relevance Learning	51
3.3 Analyses of Computational Complexity	55
3.4 Experimental Results	57
3.4.1 AES-128 performed with Different Frequencies.....	57
3.4.2 Comparison with Various Profiling CbA on AES-128	64
3.4.3 Vertical Hiding Countermeasure on AES-128	65
3.4.4 Horizontal Hiding Countermeasure on AES-128.....	66
3.5 Conclusions	67
 CHAPTER 4 HIGHLY SECURED STATE-SHIFT LOCAL CLOCK CIRCUIT TO COUNTERMEASURE	
AGAINST SIDE CHANNEL ATTACK	68
4.1 Introduction	68
4.2 Clocking System in AES-128 implementation	72
4.3 Experimental Results	74

4.4 Conclusions	79
CHAPTER 5 A HIGH THROUGHPUT AND SECURE AUTHENTICATION-ENCRYPTION AES-CCM	
ALGORITHM ON ASYNCHRONOUS MULTICORE PROCESSOR.....	79
5.1 Introduction	79
5.2 Review of AES-CCM.....	85
5.3 Proposed Authentication Based Matrix-Transformation Cum Parallel-Encryption in Multicore Processor (AMP-MP) on AES-CCM.....	87
5.3.1 Proposed matrix multiplication in $GF(2^8)$	87
5.3.2 Proposed Parallel and Rescheduled AES encryption	92
5.3.3 Proposed novel asynchronous data flow	95
5.3.4 Proposed key adjusting technique	96
5.4 Experimental Results in Asynchronous Multicore	97
5.4.1 Matrix multiplication over $GF(2^8)$ and authentication.....	98
5.4.2 Parallel encryption and rescheduling of the AES in ANoC	101
5.4.3 Key adjusting technique	104
5.5 Experimental Results in SCA	107
5.5.1 Single Channel SCA.....	109
5.5.2 Multi-Channel SCA	110
5.6 Conclusions	113
CHAPTER 6 A CONTROLLABLE TIME SHUFFLING COUNTERMEASURE SCA	
6.1 Introduction	113
6.2 Proposed Controllable Time Shuffling	115
6.3 Design Nano AES embodying a proposed CTS	117
6.4 Experimental Results	119
6.5 Conclusions	124

CHAPTER 7 CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK	124
7.1 Conclusions	124
7.2 Recommendations for Future Work.....	127
Bibliography	129

LIST OF FIGURES

Figure 1.1: Potential attack on the main channel communication by Adversary.....	1
Figure 1.2: Side-Cannel Attacks on main and side-channels communication by Adversary	3
Figure 1.3: The global market for hardware encryption: trends, drivers and projections [4]	4
Figure 2.1: Illustration of potential attack on security algorithm-based communication system	11
Figure 2.2: Illustration of a) Asymmetric and b) Symmetric encryption algorithms.....	12
Figure 2.3: The encryption and decryption of RSA are using public and private keys respectively	13
Figure 2.4: Point P and Q in boundary elliptic curve.....	14
Figure 2.5:The general scheme of AES algorithm.....	16
Figure 2.6:The two sub-modules of a conventional S-Box.....	17
Figure 2.7: General description of DES encryption algorithm	19
Figure 2.8: Simple Power Analysis (SPA) attack	21
Figure 2.9: DPA Scheme	22
Figure 2.10: Illustration of countermeasure against SCA.....	25
Figure 2.11: Countermeasure trend for the last 15 years	25
Figure 2.12: (a) Architecture of a SRCP and (b) PE of SRCP.....	28
Figure 2.13: Ring Oscillator operates in parallel with S-Box.....	29
Figure 2.14: The implementation of ring oscillator	30
Figure 2.15: A CoRe countermeasure technique	31
Figure 2.16: DVFS Cryptosystem.....	32
Figure 2.17: Current Equalizer block with three switching stage capacitors.....	33
Figure 2.18: Four phase current equalizer	34
Figure 2.19: General structure of SABL.....	36
Figure 2.20: Implementation WDDL in crypto module.....	37

Figure 2.21: An AND/NAND cell based on PCSL	38
Figure 2.22: TDPL inverter.....	39
Figure 2.23: The combination of switched capacitor and SABL in hybrid countermeasure	39
Figure 3.1: (a) Profiling on PLI for the CbA before the computation of correlation and (b) Classifications of profiling on PLI	42
Figure 3.2: General flow of our proposed PRL on the PLI measurement SCA which comprises.....	46
Figure 3.3: Illustrations of our proposed PRL technique with three PLI measurements and two clusters, phases ① – ⑥ are corresponding to respective (a) PLI measurements, (b) k -NN variance clustering, (c) extracting the useful sampling points from k -NN, (d) scaling the PLI amplitude, (e) assigning weights for each extracted sampling points and (f) generating result of the amplitude scaling which is amplified with weights for cluster 1 and cluster 2 of PLI	46
Figure 3.4: (a) Flow chart of k -NN variance clustering on the PLI,	51
Figure 3.5: The flow chart of relevance learning on the clustered traces	54
Figure 3.6: Comparison of time complexity on various profiling techniques	56
Figure 3.7: Different power dissipation measurements for the AES-128 running at (a) 2MHz,	59
Figure 3.8: The k -NN variance clustering, clustered (shaded area) and discarded sampling points of the power dissipation measurements in Figure 3.7, (a) 2MHz, (b) 9.6MHz, (c) 12MHz and	59
Figure 3.9: Learning error of the clustered variances in C#10, where the best (2MHz) and worst (16MHz) scenarios require 256 and 538 iterations respectively to obtain the convergence weights	60
Figure 3.10: Correlation coefficient vs power traces of the CPA on AES-128 @2MHz (a) without PRL profiling (b) with our proposed PRL profiling.....	61
Figure 3.11: Correlation coefficient vs power traces of the CPA on AES-128 @16MHz (a) without PRL profiling (b) with our proposed PRL profiling.....	61
Figure 3.12: Fake peaks of the EM signal are noticeable at high frequency	63
Figure 3.13: EM at 16MHz requires 2,103 iterations to achieve $E < 1\%$ due to small peaks hinder the relevance learning	63
Figure 4.1: Clocking systems in a) AES-128 implementation, b) Master Clock,.....	71
Figure 4.2: Timing diagram of a) SsLC_1, b) SsLC_1 and SsLC_2 shifted based on the M_CLK.....	73

Figure 4.3: Block diagrams of a) SsLC_1 and b) SsLC_2.....	74
Figure 4.4: The measured clocks signals from SsLC_2, SsLC_1 and M_CLK.....	75
Figure 4.5: Power dissipation and EM measurements of the AES-128 implementation	76
Figure 4.6: Distribution PLI based on variance value of reported higher order shuffling	77
Figure 4.7: a) Secret key (142D) is revealed in reported technique [47] implementation and	78
Figure 4.8: Correlation coefficient VS Traces of a) reported higher order shuffling technique [60] and b) Proposed SsLC.....	78
Figure 5.1: The Message Authentication Code (MAC) is generated at the Transmitter and verified at the Receiver	82
Figure 5.2: AES-CCM Architecture based on (a) Conventional (b) Our proposed AMP-MP	82
Figure 5.3: Authentication process	85
Figure 5.4: Counter mode is performed to generate ciphertext and MAC.....	86
Figure 5.5: Schematic flow of our proposed matrix multiplication in $GF(2^8)$	88
Figure 5.6: Matrix multiplication in $GF(2^8)$	88
Figure 5.7: Circuit implementation for the matrix multiplication.....	92
Figure 5.8: Design of ANoC based 9-Core processor (a) Overall architecture of multicore and (b) Parallel and rescheduling three AES within three cores	93
Figure 5.9: Main channel interface (L and R) of SAHB circuit and cell template	96
Figure 5.10: Circuit schematic of key adjusting technique for each 8-bit Data_in.....	97
Figure 5.11: Microphotograph of the fabricated multicore (a) 9-Core with respective ANoC.....	98
Figure 5.12: Power dissipation of matrix multiplication and authentication process	100
Figure 5.13: Power dissipation of parallel encryption in three cores which dissipates power $307\mu W$ in $3.26\mu s$	102
Figure 5.14: EM signal of our proposed AMP-MP in multicore and single core synchronous implementation	103
Figure 5.15: Comparison of the variance power dissipation during the encryption	104

Figure 5.16: The EM signal patterns are detected based on (a) reported key updating system and with (b) our proposed key adjusting technique the patterns are omitted	105
Figure 5.17: HD and HW distribution of 256 secret key (a) HD and HW patterns are detected and (b) Performing key adjusting technique to remove patterns.....	107
Figure 5.18: The SR of power and EM analyses based on single core and multicore implementations ..	108
Figure 5.19: CPA attack on AES-CCM based on (a) Single core and.....	110
Figure 5.20: The plot of Single-channel (black) and Multi-channel result (red) based on CPA and CEMA which are unrevealed at 5×10^5 measurements	111
Figure 6.1: Proposed Controllable Time Shuffling countermeasure.....	115
Figure 6.2: Illustration of the CTS operation a) without redundant round, the genuine round is performed at the b) beginning, c) middle and d) last of clock cycles	117
Figure 6.3: The S-Box mode operations parallel and serial mode which require	118
Figure 6.4: The iteration of $16 \times$ for the 8-bit S-Box in the designed Nano AES.	119
Figure 6.5: Measurement result of the proposed CTS for AES-128 algorithm	120
Figure 6.6: The EM measurement of the Nano AES embodying CTS technique	122
Figure 6.7: SCA evaluation for both Parallel and Serial S-Box implementations embodying the CTS technique in FPGA platform	122
Figure 6.8: Micrograph for both Parallel and Serial S-Box in Nano AES implementations	123

LIST OF TABLES

Table 2.1: NUMBER OF LOGIC GATES REQUIRED IN CONVENTIONAL S-BOX	18
Table 2.2: CHARACTERISTIC OF VARIOUS SYSTEM-LEVEL COUNTERMEASURE TECHNIQUES	27
Table 2.3: PERFORMANCE OF CURRENT EQUALIZER SWITCHING CAPACITOR [54].....	34
Table 2.4: CHARACTERISTIC OF VARIOUS CELL-LEVEL COUNTERMEASURE TECHNIQUES	35
Table 3.1: COMPARISON OF N , S , T AND M BASED ON CPA IN FOUR DIFFERENT FREQUENCIES	62
Table 3.2: COMPARISON OF N , S , T AND M BASED ON CEMA IN FOUR DIFFERENT FREQUENCIES	64
Table 3.3: COMPARISON OF COMPUTATIONAL COMPLEXITY, PROFILING AND CBA PERFORMANCES BASED ON OUR PROPOSED PRL AND VARIOUS PROFILING TECHNIQUES AT OPTIMUM FREQUENCY	65
Table 3.4: COMPARISON OF N , S , T AND M IN CBA BASED ON PROPOSED PRL AND COMPENSATOR [45] .	66
Table 3.5: COMPARISON OF N , S , T AND M IN CBA BASED ON PROPOSED PRL AND SsLC [55].....	67
Table 4.1: MEASUREMENT FREQUENCY OF SsLC_2, SsLC_1 AND M_CLK	75
Table 5.1: A GENERATOR OF $GF(2^4)$ FOR CONSTANT VECTOR	90
Table 5.2: THE 1 ST AND 2 ND PREIMAGE ATTACKS BASED ON THE MAC OF	101
Table 5.3: THE THROUGHPUT PERFORMANCE OF PARALLEL ENCRYPTION AND PARALLEL DECRYPTION IN 6 CORES CONFIGURATION	102
Table 5.4: MEASUREMENTS OF TIME, POWER AND EM OF KEY ADJUSTING	107
Table 5.5: COMPARISON OF OUR PROPOSED AMP-MP AES-CCM WITH	112
Table 5.6: THE PERFORMANCE OF OUR PROPOSED AMP-MP COMPARED WITH OTHER THREE OPTIMIZED AEAD ALGORITHMS	112
Table 6.1: THE COMPARISON OF THE PERFORMANCE SHUFFLING COUNTERMEASURE.....	121

LIST OF AUTHOR'S PUBLICATIONS

PATENT

1. K.S. Chong, B.H. Gwee and **A. A. Pammu**, "Hardware Security to Countermeasure Side-Channel Attacks," PCT Patent Applications and Singapore Patent Application No. 10201702226R, 19 Mar 2018.

JOURNAL PAPERS

1. **A. A. Pammu**, K. S. Chong, Yi Wang and B. H. Gwee, "A Highly Efficient Side Channel Attack with Profiling through Relevance Learning on Physical Leakage Information," *IEEE Transaction on Dependable Secure Computing*, July 2018.
2. **A. A. Pammu**, W. Ho, N. K. Z. Lwin, K. Chong and B. Gwee, "A High Throughput and Secure Authentication-Encryption AES-CCM Algorithm on Asynchronous Multicore Processor," in *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 1023-1036, April 2019.
3. **A. A. Pammu**, K.-S. Chong, B.-H. Gwee, "A Highly-Secured Arithmetic Hiding cum Look-Up Table (AHLUT) based S-Box for AES-128 Implementation", *Advances in Science, Technology and Engineering Systems Journal*, vol. 2, no. 3, pp. 420-426, 2017.

CONFERENCE PAPERS

1. **A. A. Pammu**, K. S. Chong, K. Z. L. Ne and B. H. Gwee, "High Secured Low Power Multiplexer-LUT Based AES S-Box Implementation," *2016 International Conference on Information Systems Engineering (ICISE)*, Los Angeles, CA, 2016, pp. 3-7. **(Best Presenter Award)**
2. **A. A. Pammu**, K. S. Chong and B. H. Gwee, "Secured Low Power Overhead Compensator Look-Up-Table (LUT) Substitution Box (S-Box) Architecture," *2016 IEEE International Conference on Networking, Architecture and Storage (NAS)*, Long Beach, CA, 2016, pp. 1-7.
3. **A. A. Pammu**, K. S. Chong, W. G. Ho and B. H. Gwee, "Interceptive side channel attack on AES-128 wireless communications for IoT applications," *2016 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, Jeju, 2016, pp. 650-653.
4. **A. A. Pammu**, K. S. Chong, N. K. Z. Lwin, W. G. Ho, N. Liu and B. H. Gwee, "Success rate model for fully AES-128 in correlation power analysis," *2016 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, Jeju, 2016, pp. 115-118.
5. **A. A. Pammu**, K. S. Chong and B. H. Gwee, "Highly secured arithmetic hiding based S-Box on AES-128 implementation," *2016 International Symposium on Integrated Circuits (ISIC)*, Singapore, 2016, pp. 1-4. **(Best Student Paper Award)**
6. W. G. Ho, **A. A. Pammu**, N. Liu, K. Z. L. Ne, K. S. Chong and B. H. Gwee, "Security analysis of asynchronous-logic QDI cell approach for differential power analysis attack," *2016 International Symposium on Integrated Circuits (ISIC)*, Singapore, 2016, pp. 1-4.

7. **A. A. Pammu**, K. S. Chong and B. H. Gwee, "Highly secured state-shift local clock circuit to countermeasure against side channel attack," *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD-USA, 2017, pp. 1-4.

ABBREVIATIONS

AEAD	: Authentication-Encryption with Associated Data
AES	: Advanced Encryption Standard
AES-CCM	: Advanced Encryption Standard based on Counter with Chaining Mode
AMP-MP	: Authentication based Matrix-transformation cum Parallel-encryption implemented on an asynchronous Multicore Processor
ANoC	: Asynchronous Network-on-Chip
CbA	: Correlation-based Attack
CPA	: Correlation Power Analysis
CEMA	: Correlation Electromagnetic Analysis
CNN	: Convolutional Neural Network
CTS	: Controllable Time Shuffling
DES	: Data Encryption Standard
DH	: Diffie-Hellman
DPA	: Differential Power Analysis
DVS	: Dynamic Voltage Scaling
DVFS	: Dynamic Voltage Frequency Scaling
ECC	: Elliptic Curve Cryptography
FSM	: Finite State Machine
GF	: Galois Field
HD	: Hamming Distance
HW	: Hamming Weight
IoT	: Internet-of-Thing
IPsec	: Internet Protocol security
k -NN	: k Nearest-Neighbor
LUT	: Look-Up-Table
MDPL	: Modified Dual-rail Pre-charge Logic

MLUT	: Multiplexer Look-Up-Table
NCL	: Null Conventional Logic
PCSL	: Pre-Charge Static Logic
PE	: Processing Element
PLI	: Physical Leakage Information
PRL	: Profiling through Relevance-Learning
RSA	: Ravist Shamir Adleman
S-Box	: Substitution Box
SABL	: Sense Amplifier Based Logic
SCA	: Side-Channel Attack
SNR	: Signal-to-Noise Ratio
SoC	: System-on-Chip
SOSD	: Second Order Standard Deviation
SR	: Success Rate
SRCP	: Secured Reconfigurable Crypto Co-Processor
SsLC	: State-shift Local Clock
TDPL	: Three-phase Dual-rail Pre-charge Logic
WDDL	: Wave Dynamic Differential Logic
ZOA	: Zero Order variant Attack

LIST OF NOTATIONS

Notation	Description
$\alpha_t(i)$	Learning rate at sampling point t with i^{th} iterations; initial value is α_0
B	Constant threshold boundary in computational complexity
C	Correction factor during the relevance learning; i.e., 1.2 to 1.5
$d(. , .)$	Euclidian distance to determine the distance between object and centroid
D	Constant for 2-dimensional space in computational complexity
E	Set of object variances after the k -NN variance clustering
E	Factor error in the relevance learning to represent the SNR of the PLI
I	The i^{th} of PLI measurement (plaintext/ciphertext)
l	Number of traces for pre-processing reported techniques
L	Leakage function of PLI measurement
M	Number of traces for correlation computation in SCA
N	Number of traces for k -NN variance clustering
O	Centroid position of the object class
P	PLI measurement of the crypto-device; i.e., power dissipation or EM
S	Number of traces for relevance learning
$S()$	Sigmoid function during the relevance learning
T	Number of sampling points for k -NN variance clustering
t	The t^{th} sampling point for each PLI measurement
w	Weight assigned to each clustered sampling point
σ_x	Variance of PLI measurement at time x
$\Lambda_{x, y}$	Relevance factor between sampling points x and y

Γ	Set of sampling points respect to PLI measurement
λ	Normalized rate-change variance during the learning process
Ω_t	Set of PLI in one sampling point t
χ	Input variable for leakage function (i.e., ciphertext)
$\mathbb{Z}$	Set of clustered variances respect to sampling point

CHAPTER 1

INTRODUCTION

1.1 Motivation

To date, the advanced technologies enable most of the electronic devices to be connected via internet. However, the digital information available online during the communication is highly vulnerable against malicious-attack [1]. Figure 1.1 depicts an unauthorized party or adversary may intercept and obtain the data to retrieve the confidential information (i.e., identity of bank customer or payroll records). Hence, an effective encryption algorithm is required to protect the data from the unintended party. In the encryption algorithm, the users are required to provide a secret key (known only by user) to encrypt the data (i.e., message/plaintext) into protected message (ciphertext). The adversary obtains the ciphertext may not be able to interpret it without knowing the secret key [2], even though the encryption algorithm is known by the adversary.

Figure 1.1: Potential attack on the main channel communication by Adversary

The encryption algorithm can be categorized into two categories [2] based on the types of

secret keys used to encrypt and decrypt the data, namely symmetric and asymmetric encryption. The symmetric encryption algorithm uses the same key (i.e., secret key) for encryption and decryption which encrypt and decrypt the plaintext and ciphertext respectively, while asymmetric encryption algorithm uses different keys. In general protocol of cryptographic communication, key distribution for symmetric encryption algorithm [3] is performed by asymmetric encryption. For instance; asymmetric encryption algorithm, Rivest Shamir Adelman (RSA), with 128-bit data processor is used to establish and share the keys of the symmetric encryption algorithm, Advanced Encryption Standard (AES), from transmitter to one or more receivers. The RSA is the most commonly used asymmetric encryption algorithm due to faster computation [3] than others asymmetric cryptographic algorithms, while in symmetric encryption algorithms is AES which can process the different size of the data at the time (i.e., 128, 192 and 256 bits). In this thesis, our focus is on the symmetric encryption, in particular the AES-128.

A physical hardware-attack approach which is known as Side-Channel Attack (SCA) [4][5] has been reported to successfully extract the secret key of the encryption algorithm by analyzing the Physical Leakage Information (PLI) which are obtained from Side-Channel of the crypto-device leak out during the execution of the algorithm. The PLI such as power dissipation, electromagnetic interference, acoustic (i.e., sound and vibration), temperature and timing of electronic devices (crypto-device) are dependent on the operations and input values of the algorithm. Figure 1.2 illustrates the SCA by adversary whose successfully reveal the secret key by analyzing the PLI and the ciphertext. Power analysis is the most common SCA [5] where it analyzes the power dissipation profile to extract the secret key. The adversary that successfully obtains the secret key by means of SCA is able to further decrypt the ciphertext

and retrieve the original message. In this context, protecting the secret key, which is stored in crypto-device (i.e., smart phone), against SCA is important on the online communication.

Figure 1.2: Side-Channel Attacks on main and side-channels communication by Adversary

In order to protect the crypto-devices against the SCA, countermeasure techniques are required to make the association between the PLI and the plaintext/ciphertext independent. The countermeasure technique can be performed in the crypto-device based on the software or hardware approaches.

The SCA and the countermeasure techniques based on the hardware encryptions are continuously evolving to resolve the major security threat in the crypto-devices. According to the Global Industry Analysts, Inc, the global market for the hardware encryption is projected to reach US\$252 billion by 2020 [4]. Figure 1.3 depicts a summary of the global market for hardware encryption, particularly the key drivers which include; computing devices and networking devices, storage market (USB and HDDs), embedded systems, Internet-of-Things (IoT) and cloud computing to secure the data leakage against SCA.

Figure 1.3: The global market for hardware encryption: trends, drivers and projections [4]

In this thesis, the main focus is to design the highly efficient countermeasure techniques which are implementable in multiple platforms such as FPGA and ASIC platforms. In addition, the implementations of countermeasure techniques leverage the multicore platform based Asynchronous Network-on-Chip (ANoC). Finally, this research aims to achieve the highly secured of crypto-device and yet low overheads on the implementations.

1.2 Objectives

The overall objective of this thesis is to design and realize an effective SCA algorithm which subsequently used to effectively evaluate the proposed highly secured countermeasure techniques. The overall objective of this research can be decomposed into two main objectives where the first objective is to design an effective SCA algorithm to evaluate the robustness of the countermeasure techniques. On the contrary, the second objective is designing a highly secured countermeasure technique to make the SCA technique inefficient to break the secret

keys. The two main objectives are dependent entity in this research where the countermeasure technique will be invalid without being verified with the effective robust SCA.

The effective SCA algorithm aims to reveal the secret key of the implemented cryptographic algorithms in the crypto-device embodying the highly secured countermeasure techniques with shortest time and lowest computational resources. The proposed highly secured countermeasure techniques aim to break the dependency between the PLI and the plaintext/ciphertext without degrading other performances parameters, such as power dissipation, area and speed overheads.

In view of the overall objective, the specific objectives are:

- (a) To design a highly efficient SCA by integrating the machine learning algorithms in the pre-processing of the PLI of crypto-device. In this design, the Profiling through Relevance-Learning (PRL) technique on PLI is used to extract the highly correlated PLI with processed-data, as to achieve a highly efficient yet effective SCA.
- (b) To design a countermeasure technique based on the State-shift Local Clock (SsLC) to hide the PLI against SCA. The design embodies a Finite State Machine (FSM) which can be employed to regularly shift the timing operation of cryptographic algorithm implementations.
- (c) To increase the throughput and security performances of an authentication-encryption algorithm by adopting a Matrix-transformation cum Parallel-encryption. The design is implemented on an asynchronous Multicore Processor to achieve high throughput and yet secure cryptographic algorithm implementations.

- (d) To reduce the form-factor and power dissipation overhead of AES-128 implementation by designing a Nano AES-128 encryption embodying Controller Time Shuffling (CTS). The CTS is to further secure the small embedded devices (i.e., Biomedical devices) against SCA.

1.3 Contributions

This section highlights the contributions of the work described in this thesis. The contributions pertaining to the objectives (a)-(d) are elaborated as follows:

- (i) A PRL technique on PLI is proposed to extract the highly correlated PLI with processed-data, by clustering the highly correlated PLI and discarding the unrelated PLI as to achieve a highly efficient and yet effective SCA.
- (ii) An SsLC countermeasure technique is proposed to protect the PLI against the SCA for ultra-low power and medium speed implementations.
- (iii) An FSM is incorporated to the SsLC which can be employed to regularly shift the timing operation of cryptographic algorithm implementations. Thus, the correlation of the PLI with the processed data is significantly reduced due to dynamically changes the occurrences of encryption operation in time domain.
- (iv) An Authentication based Matrix-transformation cum Parallel-encryption implemented on an asynchronous Multicore Processor (AMP-MP) is proposed to achieve a high throughput and yet secure an Advanced Encryption Standard based on Counter with Chaining Mode (AES-CCM).

- (v) The matrix multiplication in $GF(2^8)$ computation to transform each message of 16 plaintexts into 1 plaintext is employed, hence improving the authentication speed by $32\times$ collectively at the transmitter and receiver.
- (vi) The operations of 3 AES encryptions in 3 different cores are rescheduled such that their PLI measurements are compensated and equalized, thus reducing the correlation of physical leakage with the processed data by $>3\times$.
- (vii) The intermediate values of AES-CCM are propagated asynchronously between different cores to randomize the PLI with the processed data, therefore further enhance the security of AES-CCM against the SCA by another $3\times$.
- (viii) A key adjusting technique based on S-Box byte-key transformation is proposed to protect the key against pattern-based attack, hence further enhance the security level of the AES-CCM.

1.4 Organization of the thesis

The remainder of this thesis is organized in the following manners.

Chapter 2 reviews the encryption algorithms and SCA which have been employed to reveal the secret key implemented in crypto-devices. Different types of PLI are presented including power dissipation, electromagnetic, temperature, acoustic and timing. Several types of attacks including Simple Power Analysis (SPA), Differential Power Analysis (DPA), Correlation Power Analysis (CPA), Correlation Electromagnetic Attack (CEMA) and the reported countermeasure techniques for SCA are reviewed.

Chapter 3 presents a proposed a PRL technique on PLI to extract highly correlated PLI with processed-data, as to achieve a highly efficient yet effective SCA. In other words, our proposed PRL technique is able to break the secret key with low number of PLI. The nearest-neighbor k -NN variance clustering is used to reduce the sampling points of PLI by clustering the high variance sampling points and discarding the low variance sampling points of PLI traces. Subsequently, the relevance learning algorithm is adopted to learn the relevance factor for each clustered sampling point to quantify the degree of leakage associated with the secret key. Chapter 3 also presents the robustness of the proposed PRL technique against noisy PLI and benchmarking against reported profiling techniques. The proposed PRL successfully reduces the number of traces when performing with different noise levels. By comparing with reported techniques which require enormous number of traces, the proposed PRL is more efficient in performing SCA.

Chapter 4 presents a proposed a highly-secured SsLC for countermeasure technique to secure the PLI against SCA. It subsequently presents the features of the SsLC which embody an FSM to regularly shift the timing operation of cryptographic algorithm implementations. Thereafter, it shows the comparison with the reported counterparts where the resistance of our proposed SsLC against SCA is significantly improved where the number of power dissipation and EM traces to reveal the secret key has increased significantly.

Chapter 5 presents a proposed AMP-MP to achieve a high throughput and yet secure AES-CCM. Thorough comparison with the reported authentication-encryption process is subsequently presented in this chapter. Thereafter, the experimental result based on the fabricated ASIC asynchronous multicore is presented by evaluating the throughput and security feature against SCA.

Chapter 6 presents a proposed ultra-low power Nano AES-128 encryption with CTS to secure Biomedical device applications with smallest form-factor in terms of gate counts. It subsequently presents the performance of the proposed idea based on the FPGA and the fabricated ASIC design implementations. Finally, the evaluation against SCA is presented in this chapter and benchmarking the performances against reported countermeasure techniques.

Chapter 7 draws the conclusions of this thesis and provides the recommendations for future work.

CHAPTER 2

LITERATURE REVIEW

This chapter provides a review on different cryptographic algorithms based on their classifications and implementations. The SCA is also reviewed as an effective technique to reveal the secret key of the crypto-devices. Finally, the reported countermeasure techniques are reviewed as the benchmark comparison for our proposed countermeasure designs.

2.1 Cryptographic Algorithms

In this era where most of the electronic devices are connected wirelessly, numerous attacks such as viruses, hackings, electronic eavesdropping and electronic fraud can be the potential threads for these electronic devices. Thus, it is paramount to secure these electronic devices and the processed data against the threads. There are three key principles to secure the processed data in the electronic devices, namely confidentiality, integrity, and availability [6].

The confidentiality refers to the preservation of an authorized restriction to access and disclose the processed data (information), including for the protection of the personal privacy and the proprietary information. A breaching of confidentiality is an unauthorized disclosure of information. The integrity refers to the protection of the information against modification or destruction of the information, including ensuring information nonrepudiation and authenticity. Transferring the information without integrity is the unauthorized modification or destruction of information. For instance; multiple gadgets are synchronized with one account is high potential to loss the integrity commutation. The availability refers to reliability of the accessing

and processing the information. Losing of availability in communication is the disruption of access to or process of the information or the information system. These three objectives are sorted based on security layer in communication system, in which the first layer is confidentiality, followed by integrity and availability. Once the confidentiality is violated, the subsequent security layers are easily accessed by adversary.

In developing a particular security mechanism or algorithm to highly protect confidential information, we must always consider potential attack on every drawback of those security features. In many cases, successful attacks are designed by looking at the problem in completely different way, therefore exploiting an expected weakness in the system or mechanism is important. Figure 2.1 depicts the illustration of potential attack by Adversary on secured message through open channel. The security mechanism typically involves more than a particular algorithm or protocol [6]. In this context, the users are also required in possession of some secret information (e.g., an encryption key) as additional of security features. In the big picture, the security algorithm or cryptosystem comprises of two main categories, Asymmetric and Symmetric encryption algorithms as illustrated in Figure 2.2. The mechanisms of both algorithms are the same in the sense that the encryption and decryption are located at transmitter and receiver respectively.

Figure 2.1: Illustration of potential attack on security algorithm-based communication system

Figure 2.2: Illustration of a) Asymmetric and b) Symmetric encryption algorithms

2.1.1 Asymmetric Encryption Algorithms

The asymmetric encryption employs two different keys, known as the public key and the secret key to encrypt the plaintext and decrypt the ciphertext. The public key is the publicly known key used to encrypt the plaintext into a form of ciphertext. The secret key which is known only by the receiver to decrypt the ciphertext into plaintext. This encryption algorithm is based on mathematical function to encrypt and decrypt the message, therefore requires high power in computation to secure the message. The most widely used asymmetric encryption is RSA[6], in which the difficulties of attacking RSA is based on difficulty of finding the prime factors of a composite number (i.e., all numbers except number one and prime numbers). Other types of asymmetric encryptions which are commonly used in many applications are Elliptic Curve Cryptography (ECC), El-Gamal and Diffie-Hellman (DH) [3].

2.1.1.1 Ravist Shamir Adelman (RSA)

The RSA makes use of an expression with exponentials in which plaintext is encrypted in blocks, with each block having a binary value less than some number, n , which has been specified in advance. Thus, the block size must be less than or equal to $\log_2(n) + 1$; in practice, the block size is i bits, where $2^i < n < 2^{i+1}$. Figure 2.3 depicts the application of an RSA algorithm for encryption

and decryption in the communication channel. The encryption process employs a public key (i.e., Bob's public key) while the decryption process requires private key (i.e., Bob's private key).

Figure 2.3: The encryption and decryption of RSA are using public and private keys respectively

2.1.1.2 Elliptic Curve Cryptography (ECC)

The main operation of ECC is based on elliptic curve arithmetic [4]. The elliptic curve is described as cubic equations, similar to those used for calculating the circumferences of an ellipses. In general, cubic equations for elliptic curves is expressed as the following equation (2.1), known as Weistrass equation.

$$y^2 + axy + by = x^3 + cx^2 + dx + e \quad (2.1)$$

Where coefficient; a , b , c , d and e are the real number, and x and y are the coordinate value of the elliptic curve. The ECC makes use of elliptic curve just enumerated in which variables and coefficients all restricted to elements of finite field.

For example, two points (P, Q) in elliptic curve $y^2 = x^3 + x + 1$ are shown in Figure 2.4, based on field $p = 23$ (prime number). Figure 2.4 also shows the results of addition and inverse addition of two points.

Figure 2.4: Point P and Q in boundary elliptic curve

In the ECC application, multiplicative operation in elliptic curve is normally used to encrypt the message. The result of multiplication (i.e., Q) is made public and the multiplicative value (i.e., P) is a privately selected.

2.1.1.3 El-Gamal

In 1984, T. El-Gamal announced a public-key scheme based on discrete logarithms and named as El-Gamal cryptosystem [6]. The El-Gamal cryptosystem is used in standard applications including the digital signature standard and e-mail security. The global elements of El-Gamal are prime numbers q and α , which are coprime to each other.

2.1.1.4 Diffie-Hellman (DH)

The Diffie-Hellman is invented by Diffie and Hellman that defined public-key cryptography [6] and is generally referred to as DH key exchange. The purpose of this algorithm is to enable two users to securely exchange a key which can then be used for subsequent encryption of the large size of the message as in symmetric encryption does. The algorithm depends on its effectiveness on the difficulty of computing discrete logarithms for q and α which are coprime to each other.

2.1.2 Symmetric Encryption Algorithms

The symmetric encryption is a form of cryptosystem in which encryption and decryption are performed employing the same key (i.e., encryption key) and also known as conventional encryption. It transforms message (known as plaintext) of transmitter into ciphertext using encryption key. At the receiver, by means of the same key and decryption algorithm, the plaintext is recovered from the ciphertext as illustrated in Figure 2.2. The plaintext is processed in fixed block bases, such as 64-bit, 128-bit, 192-bit and 256-bit, therefore it can process unlimited length of plaintext with only use the same key. Two main algorithms classified as symmetric encryption, AES which is based on substitution principle and Data Encryption Standard (DES) which is based on permutation principal.

2.1.2.1 Advanced Encryption Standard (AES)

The AES, which is based on substitution principal, encrypts the plaintext in fixed block length with a key, which could be 128, 192 and 256 bits [6]. The number of iterative rounds required to produce the output is also different for different size of input, which are AES-128, 192 and 256 require 10, 12, and 14 rounds iteration respectively. The most commonly used among those three types is AES-128 which is processing 128-bit plaintext at a time. Each round of the iteration contains four logical and arithmetic operations namely AddRoundKey, ShiftRows, MixColumn and Substitution-Box (S-Box), except for the last round which is excluding only the MixColumn [7]. The general schematic diagram for AES encryption algorithm is depicted in Figure 2.5.

Figure 2.5: The general scheme of an AES algorithm

The explanation of working principle for each operation is presented as follows. The AddRoundKey operation is simply XOR the input plaintext and the key. Each byte of plaintext is calculated using XOR operation corresponding byte of the key, hence the number of byte-keys must be in line with the byte of the plaintext during the operation. The ShiftRows operation is to shift the intermediate data which has been transform into matrix form. In AES-128 algorithm, 16-byte of the plaintext is transformed into 4×4 square matrices and each row is shifted to the left based on the number of rows (i.e., first row starts as row0, following the second, third and fourth as row1, row2 and row3 respectively). The MixColumn is to mix the intermediate data (in the function of plaintext and key) with pre-defined constant value which is independent of the intermediate data value. The S-Box operation is non-linear operation which maps the input value to the unique output by maintaining the size of the data which is 8-bit input and output. The S-Box consist of multiple modules which makes the AES algorithm highly resistance against malicious-attack. In addition, the multiple rounds of AES make the algorithm more secured and resistant against malicious attack compared with other encryption algorithms. The intermediate data of AES, which is in function of plaintext and key, is almost

impossible to predict by mathematical estimation analysis. The key expansion as depicted in Figure 2.5 also implements S-Box operation to highly secure the sub-key for each round of iteration, hence it is almost unpredicted by adversary.

The S-Box is the main module of the AES implementation. This module is a non-linear operation which makes AES highly secured in protecting the secret information (i.e., plaintext and secret key) [7]-[20]. The S-Box consists of two sub-modules, namely the multiplicative inversion sub-module in $GF(2^8)$ and the Affine transformation (operates in matrix multiplication) sub-module as shown in Figure 2.6. Each input to the S-Box is a 1-byte of intermediate data, x , and the S-Box will generate 1-byte of output $S(x)$. In term of power, it dissipates 65% - 80% of the total power dissipation of the AES implementation [21]. Based on these two sub-modules, the S-Box features a non-self-inverse function [17] which effectively protects the data against the CPA attack.

Figure 2.6: The two sub-modules of a conventional S-Box

In Figure 2.6, the isomorphic and inverse isomorphic mappings are denoted as A and A^{-1} respectively [6]. The mapping function of A aims to decompose complex $GF(2^8)$ to lower order $GF(2)$, $GF(2^2)$ and $GF((2^2)2^2)$. In circuit implementation, function of A and A^{-1} require 24 and 23 XOR gates respectively. The square operation in $GF(2^4)$ is denoted as B which requires 4 XOR gates. The sum operation in $GF(2^4)$ is denoted as C requires 4 XOR gates. The multiplication in $GF(2^4)$ is denoted as D , requires 21 XOR and 9 AND gates. The

multiplication with constant operation is denoted as E , requires 3 XOR gates. The inverse operation in $GF(2^4)$ is denoted as F , requires 22 XOR and 12 AND gates. The Affine transformation is performed based on matrix multiplication which requires 23 XOR gates. The total number of gates required in S-Box operation are 120 XOR and 21 AND gates as tabulated in Table 2.1.

Table 2.1: NUMBER OF LOGIC GATES REQUIRED IN CONVENTIONAL S-BOX

Logic Gate	Multiplication inversion in $GF(2^8)$	Affine transformation	Total number of gates
XOR	97	23	120
AND	21	0	21

2.1.2.2 Data Encryption Standard (DES)

The DES algorithm has been reported as one of the most widely used encrypted algorithms since 1977 [23]. Since then, many works [5], [14], [19] and [22] have been reported that the key of DES can be easily broken by means of mathematical attacks. This is due to permutation function is easily predicted by finite possible numbers and even with triple-DES. Afterwards, the DES is replaced by AES for wide range applications such as securing of bank transaction and military application. In addition, the AES is almost impossible to break due to substitutional function embedded on it which is unpredictable output for unique input. Therefore, exploiting all the weaknesses of the AES is our focus of this research and particularly in the hardware implementation.

The DES algorithm is based on the permutation principle which input two values to the encryption function as described in Figure 2.7. In this context, the plaintext is 64 bits in length and the key is 56 bits in length. The processing of plaintext proceeds in three phases. First, the 64-bit plaintext passes through an initial permutation (IP) which rearranges the bits to generate the *permuted input*. The second phase is consisting of 16 rounds of the same function, which

involves both permutation and substitution functions. The output of the last round consists of 64 bits which are a function of the input plaintext and the key. The left and right halves (each 32 bits) of the output are swapped to produce the pre-output. Final phase, the pre-output is passed through a permutation $[IP^{-1}]$ which is the inverse of initial permutation function, to produce the 64-bit ciphertext.

The key expansion process as shown in Figure 2.7, which is processing the key, illustrates how the 56-bit key is expanded into 16 sets of 48-bit subkeys. Initially, the key is passed through a permutation function. Then, for each of the sixteen rounds, a sub-key (K_i) is generated by the combination of a circular shift and a permutation. The permutation function is the same for every rounds, however a different sub-key is produced for the repeated shift of the key bits.

Figure 2.7: General description of DES encryption algorithm

2.2 Side Channel Attack (SCA)

In cryptographic hardware implementations, we must always consider potential attack on every drawback of those security features (i.e., AES). In many cases, successful attacks are designed by looking at the problem in completely different ways, therefore it can exploit an expected weakness in the system or mechanism of the cryptographic implementations. One of the weaknesses of cryptographic implementation is its side effect unintentionally generated from hardware during the encryption process, including SCA [5].

The SCA is a technique which employs a physical parameter generated from the crypto-device during the operation to extract useful information including plaintext and secret key. The most commonly used physical parameters are power dissipation, electromagnetic interference, acoustic, temperature and timing. Power dissipation and electromagnetic interference are mostly used to attack the communication system (i.e., Wi-Fi, Bluetooth and RFID) [14] which implements the symmetric algorithm. This is due to random input data can be correlated with its corresponding to PLI measurements during the encryption process. The acoustic parameter is used in wireless printer (i.e., dot-matrix printer) which is the sound generated during the operation and it is dependent on the processed data [6].

There are two main SCA techniques, i.e., simple and differential based SCA technique. These techniques are commonly implemented in power dissipation and electromagnetic leakage parameters, as these parameters can be easily obtained during the encryption and decryption operations. In power analysis point of view, it includes SPA and DPA which will be elaborated as follows.

2.2.1 Simple Power Analysis (SPA)

The SPA is used to determine the secret key based on the single or few numbers of power dissipation [5]-[7]. In this technique, the adversary must be equipped with the detail information about the encryption algorithm in order to identify the bit value of the data being processed as shown in Figure 2.8. For instance; multiplication and square operations are distinguished and identifiable in power dissipation measurement and further to determine the bit value which is processed in operation, either 0 or 1, ultimately disclose the overall key. The successfulness of employing this technique depends on the noise environment which is measured by its Signal-to-Noise Ratio (SNR). If the SNR value is relatively high, the key value, 0 or 1, may be undistinguished and obstruct the attacking process.

Figure 2.8: Simple Power Analysis (SPA) attack

2.2.2 Differential Power Analysis (DPA)

The DPA reveals the secret key based on large number of power traces by means of statistical analysis [5]. The detail knowledge of algorithm and crypto-device are not required since the main analysis is the changes of the power dissipation for different input values. There are two inputs of this technique, power dissipation measurement and plaintext/ciphertext as depicted in Figure 2.9. The power model is established from either ciphertext or plaintext which can be HD or HW. The HD model is employed when the crypto-device employs register for each state

value while HW model when the device implements dynamic pre-charge cell logic. The DPA has three types of analysis based on the mathematical formulae is used, CPA, Difference of Means and Distance of Means.

Figure 2.9: DPA Scheme

2.2.2.1 Correlation Power Analysis (CPA)

The CPA attack is the most commonly used of DPA based technique which determine the dependency of processed data and leakage parameters, in this context power model (function of plaintext/ciphertext) and power dissipation measurement, by means of correlation coefficient [5]. The changes value of the processed data which corresponds to the changes of power dissipation is correlated for different key candidates [7]. It is based on byte attack and therefore, there are 256 possible value of key candidates used to guess the correct key. The highest correlation coefficient which corresponds to one of the key candidates indicates the correct key. The correlation coefficient formula used in calculation is shown in (2.2) as follows:

$$\rho(X, Y) = \frac{\text{cov}(X, Y)}{\sqrt{\sigma^2(X) \cdot \sigma^2(Y)}} \quad (2.2)$$

In (2.2), $(\rho(X, Y))$, is a dimensionless quantity and the range of ρ is minus to plus one, $-1 \leq \rho \leq 1$, which determine the dependency the two random variable X and Y . The value of plus one ($\rho = +1$) implies that one group of random variables (i.e. X) has the same changes in term

of values and vector direction with second random variable (i.e., Y), while the minus one ($\rho = -1$) indicates both random variables has the same changes but opposite in vector direction, phase difference is 180° . The zero value ($\rho = 0$) indicates that two random variables are completely uncorrelated, in other word, they are completely independent. In CPA attack, value -1 and 1 of ρ have the same interpretation which are indicated as direct dependent of the power dissipation and processed data (in the function of 256 key candidates). In the CPA attack, the power dissipation and power model are denoted as t and h respectively. The computation of correlation coefficient of the candidates is shown in (2.3) as follows.

$$\rho(t, h) = \frac{\text{cov}(t, h)}{\sqrt{\sigma^2(t) \cdot \sigma^2(h)}} = \frac{\sum_{d=1}^D (t_{d,i} - \bar{t}_i) \cdot (h_{d,i} - \bar{h}_i)}{\sqrt{\sum_{d=1}^D (t_{d,i} - \bar{t}_i)^2 \cdot \sum_{d=1}^D (h_{d,i} - \bar{h}_i)^2}} \quad (2.3)$$

Where D is denoted as the number of power dissipation measurement (i.e., power traces), the number of key candidates (256) is denoted as $i=256$ in byte-based attack and the numbers of sampling points in power measurement is denoted as j .

2.2.2.2 Difference of Means

This technique uses different techniques to determine the relationship between power model which is derived from ciphertext or plaintext and power dissipation. The adversary creates a binary matrix (i.e., Matrix $\mathbf{P}$) which divides the power model into two groups, sequence of zeros and ones in each column of power model. The power model is a function of data input (plaintext/ciphertext) and key candidates. In order to determine the correct key, the adversary divides the measured power dissipation which is in the form matrix $\mathbf{T} = \{t_0, t_1 \dots t_x\}$, into two set of lines [19]. The first data is a file which corresponds to the position of zeros in the matrix $\mathbf{P} = \{p_0, p_1 \dots p_x\}$, and the second file is the rest of the file. Subsequently, the adversary calculates

the mean of the rows of the $\mathbf{T}$ matrix. Vector m_0 is denoted as means of the first files and vector m_1 is denoted as means of second file. The estimated correct key is indicated if the is significant differences of these two vectors (i.e., $|m_1 - m_0| \approx 0$). The calculation for difference means technique is described in Equations (2.4) to (2.6).

$$m_0 = \frac{1}{n} \sum_{l=1}^n p_0 \cdot t_0 \quad (2.4)$$

$$m_1 = \frac{1}{n} \sum_{l=1}^n p_1 \cdot t_1 \quad (2.5)$$

$$C = M_0 - M_1 \quad (2.6)$$

Where n is the number of rows of the matrix $\mathbf{P}$. In other formulas, the n notation also indicates the number of power traces.

2.2.2.3 Distance of Means

This technique is an improvement of the previous technique which is considering the standard deviation. The technique uses a commonly known hypothesis test to compare the quality of the mean values of two different distributions [5]. The adversary divides the matrix $\mathbf{T}$ into two sets of rows for each key candidate as well as in the previous section. Element of C is calculated as shown in (2.7).

$$C = \frac{m_0 - m_1}{S} \quad (2.7)$$

Where S is the standard deviation of the two groups distribution (m_0 and m_1).

2.3 Countermeasure against SCA

Countermeasure against SCA is a technique aims to break the dependency of the leakage parameters (side effect of the hardware) and the processed intermediate data [5], [9]-[20]. The technique can be applied in either processed data or in leakage parameters (i.e., power or

electromagnetic) point of view. There are two main types of countermeasure techniques, i.e., masking and hiding. The general countermeasure is depicted in Figure 2.10.

Figure 2.10: Illustration of countermeasure against SCA

The countermeasure technique has become the emerging topic for the last 15 years. There are many techniques, in different physical parameters (i.e., power, electromagnetic, acoustic and timing) for wide range of implementation, have been reported and summarize in Figure 2.11. In this figure, the power-based attack is dominating over other parameters. However, the power-based attack is more invasive compared with EM based attack and thus relatively easy to implement. Therefore, its countermeasure trend tends to increase for the last 5 years due to the EM signal of the crypto-device is relatively easy to detect and measure.

Figure 2.11: Countermeasure trend for the last 15 years

2.4 Masking

Masking is a type of countermeasures which is achieved by randomizing the intermediate values which are processed by the crypto-device) [5]. An advantage of this approach is that it can be implemented at algorithm level without manipulating the power dissipation characteristic of crypto-device. In other words, masking allows to make power dissipation independent of the intermediate values, even if the device has a data-dependent power dissipation. A masked intermediate value V_M is an intermediate value V which is concealed by a random value m : $V_M = V * m$ [16]. The adversary is almost impossible to access the random value. Typically, the masks are directly applied to the plaintext or to the key of the encryption algorithm. The implementation of the algorithm requires to be slightly changed in order to process the masked intermediate values and in order to keep track of the masks. The result of the encryption is also masked. Hence, the masks need to be removed at the end of the computation in order to obtain the ciphertext.

2.4.1. Software Implementation

A masking implementation refers to Boolean masking [16] scheme where the implementation is relatively easy and requires low cost. The mask (i.e., random number generated by internal algorithm) is operated in XOR with the plaintext (or key), to make sure that all intermediate values are masked throughout the computation, keep track how the masks are changed, and eventually, remove the mask from the output. If all operations of the algorithm are (linear) Boolean operations, then Boolean masking fits nicely and it is easy to implement. The examples of software implementation-based masking are masking LUTs [5] and random Pre-Charging.

2.4.2. Hardware Implementation

The implementation of the masking scheme in hardware has similar considerations as implementation in software [25]. The Boolean masking schemes fit well for many block ciphers. Hence, only for those parts of the round function that require different types of masking more efforts have to be spent. In contrast to software implementations, more trade-offs between size and speed are possible. The examples of hardware-based masking are masking multipliers, random pre-charging, and masking buses.

2.5 Hiding

The hiding countermeasure can be applied in three different abstraction levels namely; system, cell and hybrid level [7]. In this section, the explanation for each technique are stated as follows.

2.5.1 System Level

The system-level countermeasure technique controls the dissipated power by circuit of the crypto-device to be insensitive against the changes of random input value. These techniques include reconfigurable of the crypto-device [25], introduce noise to the power dissipation [26], balancing the operation of the crypto-device [27], passive and active filtering [30] and the latest technique is the equalizing power dissipation by means of tank capacitor at V_{DD} point [32]. The main drawback of this technique is area overhead. Table 2.2 tabulates the various system-level countermeasure techniques.

Table 2.2: CHARACTERISTIC OF VARIOUS SYSTEM-LEVEL COUNTERMEASURE TECHNIQUES

Countermeasure Techniques	Average Power (mW)	Delay (ns)	CPA-resistance ($\times 10^6$)	Area (mm^2)	Cryptographic algorithm(s)
SRCP [25]	31.3 – 33	102.3	1.2	3 - 3.5	AES, DES
Ring Oscillator [26]	83 - 105.6	153.4	1	-	AES
Time-Delayed reshuffle [27]	36.56	107.4	-	-	AES
DVFS [28]	45.8	56.2	-	-	AES, DES
Switching Capacitor Equalizer [54]	44.32	452.46	1	1.37	AES
Intermittent-Equalizer [56]	33.32	535.32	10	1.28	AES

The following are the detail elaborations of the various system-level countermeasure technique which has been implemented on crypto-device.

2.5.1.1 A Secured Reconfigurable Cryptographic Co-Processor (SRCP)

The main concept of this technique is to hide the physical leakage of implemented AES algorithm in crypto-device by utilizing the idle reconfigurable Processing Element (PE) to do dummy operation. The amount of power generated from dummy operation depends on the power dissipated by AES crypto-device. In addition, this crypto-device is able to be self-reconfigured to control the operation of the dummy operation. It uses external source to control the power dissipation of the crypto-device.

This technique, SRCP also supports multiple algorithm such as AES, DES, and other symmetric encryption algorithms. The algorithm can be run in parallel in integrated System-On-Chip (SoC) and implemented in 0.18 μ m CMOS processor as depicted in Figure 2.12. The SRCP consist of a PE arrays, configuration register, function configuration module, countermeasure configuration module, interconnected bus, and buffer output control.

Figure 2.12: (a) Architecture of a SRCP and (b) PE of SRCP

All interconnected network is commonly used which can realize maximum flexibility and expandability, by connecting all the PEs together through a huge number of multiplexors.

However, it has very large network scale which increase exponentially with PE number. This can cause potential data path for the throughput as shown in Figure 2.12(a).

The ability of this system is to countermeasure globally of encrypted algorithm which leads to power and area overhead. The experimental results showed that the power increases by 6% which is from 31.3mW to 33mW and compromising the area of 35% overhead.

The resistance of SRCP has been tested by performing 1.2×10^6 power traces and the result showed the one-byte key of AES is unrevealed at this number while DES is revealed at 27,679 power traces, $36\times$ more secured than the conventional implementation.

2.5.1.2 Ring Oscillator Based Countermeasure Circuit

The S-Box is the main core of the AES which is easily leaking the information of the processed data, thus the focus of the countermeasure is at the S-Box operation. In this design, the ring oscillator is implemented to hide the power dissipation against processed data of the S-Box. An internally generated dummy operation, in this context is the ring oscillator, and input data of the S-Box are implemented to control the countermeasure circuits to dynamically change the power dissipation as illustrated in Figure 2.13. The countermeasure circuit is designed to operate in parallel with the S-Box module without any modification to internal function of the S-Box.

Figure 2.13: Ring Oscillator operates in parallel with S-Box

An 8-bit input is obtained by XOR one data byte with a random hiding and an 8-bit ring oscillator are directly controlled by this 8-bit input. There are two considerations in this countermeasure design circuit: 1) the number of inversion stages in each ring oscillator and 2) the number of ring oscillators required of the S-Box. The main consideration for the number of inversion stages is the hardware cost overhead, so ring oscillator must be as short as possible as shown in Figure 2.14. The ring oscillator is activated to hide the information when the input is high. When the input is at logic 0, the ring oscillator remains idle with any feedback value, and no additional power is dissipated, and the ring oscillator starts to oscillate when the input is at logic 1. The power dissipation of the ring oscillator dominates over S-Box to make power traces independent of processed data.

Figure 2.14: The implementation of ring oscillator

2.5.1.3 Time-Delayed Converter-Reshuffling

The time-delayed converter reshuffling (CoRe) technique is employed as a countermeasure against machine learning based attack where the adversary can synchronize the sampling frequency with the operating frequency of the device under attack as depicted in Figure 2.15.

The technique exploits the distributed nature of multiphase switched capacitor voltage converter where half of the converter stages are delayed to mitigate the risk of having zero Power Trace Entropy (PTE).

Figure 2.15: A CoRe countermeasure technique

A time-delayed CoRe technique aims to scramble the monitored power dissipation hence the adversary may not be able to extract a meaningful information from SCA leakage. In this technique, half of the converter stage in the core scheme will be activated and gated with a time delay.

Time-delayed CoRe technique delays half of the converter stage, making it infeasible to synchronize the attack with the switching frequency. An analytical expression for the PTE is developed to evaluate the security-performance of the conventional and time delayed CoRe technique.

2.5.1.4 A Dynamic Voltage and Frequency Switching (DVFS) Approach

The DVFS approach is the leverage on the hardware techniques to obscure the cryptography's activities, making it more difficult to attack. This reported technique is claimed to be an

effective approach to reduce energy and realized in two ways, low level hardware support and high-level software support.

The cryptosystem is composed of processor core, DVFS feedback loop and DVFS scheduler as depicted in Figure 2.16. The feedback loop takes the value of desired voltage/frequency as an object, which is stored in internal register by scheduler. Feedback loop physically implements the desired supply voltage and operating frequency by using ring oscillator, phase locked loop and eventually output the supply voltage to the crypto-processor core. The scheduler module randomly generates a voltage or frequency (V/F) value under certain limits and store it in register. This unit requires some timing information about the application program from the Operating System.

Figure 2.16: DVFS Cryptosystem

2.5.1.5 Switching Capacitor for Current Equalizer

This countermeasure technique is an on-chip equalizer based on switched capacitor circuits that prevent the leakage of side-channel information. The current equalizer block is comprised of an array of capacitor modules, as depicted in Figure 2.17. Each capacitor module has three different switching states: (S1) replenish charge from the supply, (S2) provides charge for encryption and (S3) continue discharging to a pre-programmed value. Since there are three distinct

switching states, a minimum of three independent capacitor modules are needed to provide uninterrupted operation of the encryption core.

Figure 2.17: Current Equalizer block with three switching stage capacitors

The switching cycle starts with S1, in which the capacitor is charged to full potential by connecting it directly to the supply and disconnecting it from the core and shunt the path. In S2, the capacitor is then disconnected from the supply and connected to the encryption core. At the end of S2, the capacitor contains a variable charge which depends on encryption activities. If at this point the capacitor were connected to the supply, the amount of current drawn would carry side-channel information to the external power supply pads. To prevent this, in S3, the capacitor is disconnected from the core and it shunted until it reaches a known value. This state enforces that the amount of charge replenish by the supply in S1 is the same each time. During the shunt state, the current loop is local to the on-chip capacitor and the parallel shunt transistor. Thus, the discharging process is not externally observable in the common ground node.

An ASIC implementation of an AES encryption engine with 1.2 Gbits/s throughput at 100 MHz was fabricated in 0.13 μm CMOS technologies. Test chip measurement demonstrated the resilient against a DPA. The secured core has been subjected to ten million encryptions, 2,500 $\times$ more than the unprotected core none of its secret key blocks have yet been disclosed. The current equalizer increases the security of the AES engine with additional 7.3% area and 33%

power overheads and $2\times$ lower throughput. The performance of this technique is tabulated in Table 2.3.

Table 2.3: PERFORMANCE OF CURRENT EQUALIZER SWITCHING CAPACITOR [54]

Parameter	Unprotected	Protected
Area for AES Core*	0.35	0.36
Area for I/O + Clk generator*	0.93	0.93
Area for equalizer*	-	0.79
Operating voltage range (V)	0.6	0.78
Power dissipation (mW @1.2 V and 100MHz)	33.32	0.78
Maximum Throughput (Gb/s)	2.56	1.28 ($2\times$ lower)
Number of traces to reveal 1-byte key	4k	>10M
Number of keys can be revealed	16	0

*The area is measured in mm^2

2.5.1.1 An Intermittent-Driven Supply-Current Equalizer

This technique is an improvement from previous technique which activates the equalizer only at the rounds contain a risk of key disclosure which are the first and the last round of the AES encryption algorithm. The switching phase consists of four phases including three phases for equalizer and another phase switch is to by passing the current during second to 9th round of AES as depicted in Figure 2.18.

Figure 2.18: Four phase current equalizer

The experiment result showed that the power overhead is reduced by $11\times$ which is only about 8% of the total power dissipation.

2.5.2 Cell-Level

The Cell-level technique is the countermeasure approach based on individual gate of the crypto-device to balance the power dissipation between 1 and 0. The examples of the technique are Sense Amplifier Based Logic (SABL) [8], Wave Dynamic Differential Logic (WDDL) [9], PCSL [10], Dual-rail Pre-Charge Logic (DPL) [11], Three-phase Dual-rail Pre-Charge Logic (TDPL) [12] and Three-phase Single-rail Logic (TSL) [13]. Table 2.4 tabulates the cell-level countermeasure techniques.

Table 2.4: CHARACTERISTIC OF VARIOUS CELL-LEVEL COUNTERMEASURE TECHNIQUES

Countermeasure Techniques	Max. Energy (fJ)	NED (%)	Average delay (ps)	W/L	Cryptographic algorithm(s)
SABL [8]	3.25	7.7	64	24	AES, AES, RSA
WDDL [9]	4.04	13.9	38	30	AES
PCSL [10]	3.52	11.4	58	39	AES, DES
DPL [11]	3.12	15.4	54	21	AES, DES
TDPL [12]	1.68	71.1	55	28	AES, RSA

2.5.2.1 Sense Amplifier Based Logic (SABL)

The SABL makes the four output logics equal [8], by charging at every event the same capacitance value: one of the balance load capacitances and the sum of all the internal node capacitances.

The SABL gates is based on the StrongArm10 Flop-Flop (SAFF). To realize the basic gate, the sense amplifier is kept half of the flip flop and replace the input differential pair by a differential pull down network as depicted in Figure 2.19.

Figure 2.19: General structure of SABL

The SABL is based on two principles. First, it is dynamically and differential logic and therefore it has exactly one switching event per cycle and it is independently of the input value and sequence. Second, during the switching event, it discharges and charges the sum of all the internal node capacitances together with one of the balanced output capacitances. The evaluation of the circuit has shown that the performance in term of Normalized Energy Deviation (NED) is 0.032 which is much lower than Static CMOS (ScCMOS) with NED equal to 1. However, the design incurs the area overhead which is 47% larger than ScCMOS.

2.5.2.2 Wave Differential Dual Logic (WDDL)

The WDDL is designed based on a Dynamic Logic without a big load on the pre-charge control signal [9]. The gates are pre-charged without disturbing the signal to each individual gate. The other advantages are that during the pre-charge phase, WDDL has a lower peak supply current. As a result, the supply bounce, often a problem for signal integrity, is lowered. There are two ways to launch a pre-charge wave. First, insert a pre-charge operator at the start of every combinatorial logic tree, i.e., the input of the encryption module and the output of the registers

as depicted in Figure 2.20. The second technique is the entire compound register will be reset in every cycle.

Figure 2.20: Implementation WDDL in crypto module

This approach avoids that the differential signals have to be match by the router. The true and false signal sees the ‘same’ environment even though they are physically separated.

2.5.2.3 Pre-Charge Static Logic (PCSL)

The PCSL circuits which is based on the pre-charge logic [10]. Figure 2.21 depicts the AND/NAND cell embodying the balanced PCSL. The balancing is achieved by inserting dummy transistors to balance the charging/discharging paths. Although the concept of such dummy transistor insertion is not new, the application of PSCL can make the overall digital circuit embodying PCSL more secured. Noted that our balanced PCSL cells are applicable to both a synchronous pipeline and an asynchronous pipeline. The former synchronous pipeline is the standard approach where the clocked-based registers are used. The latter asynchronous pipeline can have realized similarly based on that in the PCSL by using asynchronous handshake protocol. The self-timed asynchronous operation is tolerant to process-voltage-temperature variations, further enabling DPA countermeasures to modulate the power dissipation in both time and amplitude. In addition, Dynamic Voltage Scaling (DVS) can be easily applied to asynchronous operations.

Figure 2.21: An AND/NAND cell based on PCSL

The simulation result in Nano-Sim with CMOS 65nm technology shows that the PCSL has successfully increased the resistance of the circuits by 24.2% against SCA [6]. The implementation is relatively competitive [10] with SABL and WDDL in term of power dissipation and area overheads.

2.5.2.4 Three-phase Dual-rail Pre-charge Logic (TDPL)

The TDPL is an improved design of the dual-rail pre-charge logic family [11]-[12]. This technique provides additional phase to ensure that all the internal node is fully discharged and therefore, the input charge is always constant and independent of the input value. The topology of the TDPL design for inverter circuit is depicted in Figure 2.22. The experimental results confirm that the designed logic has a constant energy consumption even in the presence of asymmetric interconnection. The simulated energy dissipation per cycle is up to 100× more balanced than in the corresponding SABL gates without requiring any constraint on the geometry of the complementary wire. In addition, the overhead in terms of mean power dissipation is smaller than the masking logic or masking dual rail pre-charge logic.

Figure 2.22: TDPL inverter

2.5.3 Hybrid Countermeasure

The Hybrid approach is the combination of system and cell level approaches [11] aim to further increase the resistance of the crypto-device. The aims of this technique are to protect the system from more than one physical parameter attack such as power and electromagnetic. The system level countermeasure which has been discussed is mainly on power dissipation protection. The adversary is possible to measure other physical parameter such as electromagnetic or timing to disclose the secret information. One of the example hybrid countermeasures is collaborating the switching capacitor and SABL [8] to protect the circuits from power and electromagnetic attack. The main drawback of this approach is relatively high overhead in term of occupied area. Figure 2.23 portrays the illustration of hybrid countermeasure.

Figure 2.23: The combination of switched capacitor and SABL in hybrid countermeasure

2.5.4 Multicore Implementation

The AES algorithm consists of multiple rounds (i.e., AES-128 has 10 rounds operation). In the SCA point of view, the last and the first rounds are vulnerable due to the input and output are easily accessible, in this context the plaintext and ciphertext. Multicore chip is a suitable platform to perform different rounds of operations in different cores and to apply the countermeasure at the particular core, therefore the countermeasure technique is more secure when compared with other reported techniques.

The implemented algorithm is not limited to AES but also to other types of symmetric encryptions such as DES and 3-DES. The Advantages of this technique are low power dissipation due to the countermeasure is localized at different core. The countermeasure application can be based on the power or the electromagnetic at system and cell level. The application of this technique can support for many emerging technologies in communication such as Vehicle-to-vehicle (V2V) [27]-[29] communication, health care and other IoT implementation [30]-[35].

2.6 Summary

In this chapter, various types of encryption algorithms, the SCA methodologies, the several types of attacks and the countermeasure techniques have been reviewed. In addition, the various topologies of the circuits to countermeasure against the attacks have been reviewed as well to be benchmarking with the proposed countermeasure techniques. The properties of the circuit topologies in system and cell level have been tabulated in Tables 2.2 and 2.4 respectively.

CHAPTER 3

A HIGHLY EFFICIENT SIDE CHANNEL ATTACK WITH PROFILING THROUGH RELEVANCE- LEARNING ON PHYSICAL LEAKAGE INFORMATION

This chapter presents a proposed novel PRL technique to increase the efficiency of SCA. In this technique, the correlated PLI is identified and classified as useful signal for SCA computation. The proposed PRL technique is subsequently employed to evaluate the robustness of the proposed countermeasure techniques in Chapter 4, Chapter 5 and Chapter 6.

3.1 Introduction

The CbA [36] is a form of SCA to reveal the secret key embedded in a crypto-device. The computational complexity of CbA is proportional to the number of PLI measurements (M) and the sampling points (T) of PLI measurements, termed as $O(M \cdot T)$. For instance, a crypto-device is considered ‘secured’ if the CbA is only successful when $O(M \cdot T)$ is significantly large enough to deter the adversary, where a large M is required for establishing highly data-dependency and/or a large of T is required for better fine resolution on PLI.

While the research on effective countermeasures [37] is to enhance the CbA resistance of the crypto-device such that large M and T are required for the successful CbA, the research into pre-processing the PLI (dimensional reduction on SCA) [38], which reduces the required number of M and T , is equally of interest to hardware security communities. The basic objectives of the latter include to evaluate the vulnerability of the crypto-device against countermeasures, to enable a real-time SCA (thereafter with preventions) and to predict any potential risks of security in the future.

Profiling (pre-processing) on PLI has been used to reduce the computation in $O(M \cdot T)$, by reducing M and/or T [39]. Figure 3.1(a) shows a general block diagram where profiling is used to pre-process the PLI and obtain a small M or T (or both) before performing the CbA. The profiling aims to identify the useful data-dependent PLI (highly correlated PLI with the processed data), and if possible to localize only useful sampling points related to the secret key. In general, the profiling techniques can be classified into two main categories, the noise removal and the feature selection, as illustrated in Figure 3.1(b).

The reported noise removal techniques include Averaging [36], Filtering [39], Principle Component Analysis (PCA) [40], Linear Discriminant Analyzer (LDA) [41] and Single Spectrum Analyzer (SSA) [42]. These techniques aim to increase the SNR of the PLI such that only a small number of M is required to reveal the secret key. However, the Averaging and Filtering are conventional pre-processing techniques [40], and the numbers of pre-processing measurements, termed as l , are large (i.e., $>10^5$ measurements). As a result, the computational complexity of these techniques is $O(l^2)$ [39], hence compromising the effectiveness of M reduction for the CbA. The PCA, LDA and SSA techniques involve covariance computations on the PLI to identify data-dependent measurements. However, the reduction of M for these techniques is somewhat limited, and their efficacy could be significantly compromised in the presence of countermeasures such as operation-time randomizations [37].

Figure 3.1: (a) Profiling on PLI for the CbA before the computation of correlation and (b) Classifications of profiling on PLI

On the other hand, the reported feature selection techniques include Point-Of-Interest (POI) detection [43], T-Test [44], Sum-Of-Squared Differences (SOSD) [45] and Sum-Of-Squared Pairwise T-differences (SOST) [44]. These techniques select sampling points of PLI measurements which are highly related to the secret key and eventually reduce both M and T , are more efficient than noise removal where only M is being reduced. However, the effectiveness of POI technique can be greatly reduced if the local maxima of the PLI are less distinguishable, e.g., at high clock operations frequency. Although T-Test, SOSD and SOST involve computations to analyze the features of PLI, they suffer from being able to establish good determinants to distinguish useful data-dependent measurements in the presence of countermeasure techniques [14].

In this thesis, a novel PRL technique [81] is proposed to extract highly correlated sampling points of PLI to perform a successful CbA with small numbers of M and T . Our proposed PRL technique is categorized under the feature selection and it involves four novel features. First, we adopt variance analysis to localize the local maxima of the PLI and subsequently group into various clusters; T is reduced such that the sampling points outside the clusters will be discarded. Second, we employ the k Nearest-Neighbor (k -NN) variance clustering to reduce the number of PLI measurements. This is mainly due to the k -NN algorithm is effective in clustering the objects to the class of single nearest neighbor. In the proposed PRL profiling, the variances of sampling points are akin to the objects of the k -NN algorithm. In term of performance, the k -NN variance clustering is able to filter out the noise and cluster the highly correlated leakage signal with $>90\%$ accuracy. In other words, k -NN algorithm can cluster the highly correlated leakage signal accurately ($>90\%$) for the variance of PLI measurements. Thus, it is selected for the variance clustering in the proposed PRL profiling. Third, we derive

the relevance factor (quantified as the degree leakage associated with the secret key) for the sampling points within cluster. Forth, we adopt Hebbian learning with the relevance factor to update the weight. The weight is proportional to the secret key and it is used to amplify the PLI through amplitude scaling. The last two features are essentially to increase the SNR of PLI such that the correlation between the PLI and the processed data will be more easily noticeable at small M . Therefore, with the proposed PRL, the CbA only requires small numbers of M and T to reveal the secret key.

The proposed PRL which is based on the AES-128 algorithm is implemented on an FPGA encryption board (crypto-device) and shows that a reduction of $31\times$ of sampling points is achieved. The robustness of our proposed PRL is further analyzed by examining with four different frequencies, various reported profiling techniques and two hiding countermeasures applied on the PLI (vertical and horizontal hidings). Implementing at four different frequencies, our proposed PRL technique features up to $55\times$ and $26\times$ reductions of M and T respectively, collectively achieving $1,430\times$ reduction of $M\cdot T$. With the hiding countermeasures on PLI, our proposed PRL technique features $\sim 2,000\times$ and $51\times$ reductions of M and T respectively, collectively achieving $102,000\times$ reduction of $M\cdot T$.

This chapter is organized as follows. In Section. 3.2, the proposed PRL on PLI is described. In Section 3.3, the theoretical analysis on computational complexity of the proposed PRL is analyzed. In Section 3.4, the experimental results and the comparisons with various reported profiling techniques are presented. Finally, the conclusion is drawn in Section 3.5.

3.2 Proposed profiling through relevance-learning on physical leakage information

Figure 3.2 delineates a generic flow diagram of the proposed PRL on PLI measurements for CbA. The proposed PRL aims to reduce the number of M and T to perform a successful CbA. Our proposed PRL comprises two main steps, k -NN variance clustering and relevance learning on the clustered sampling points. These two steps can be briefly illustrated by going through 6 phases as numbered ① to ⑥ in Figure 3.2. The corresponding phases are further pictorially described in Figure 3.3(a) - Figure 3.3(f). First, Figure 3(a) shows PLI measurements (traces). Each trace represents one encryption computation to convert a plaintext to a ciphertext, numerous traces are collected and re-aligned for profiling and subsequently proceeded for CbA. The k -NN variance clustering is then performed by identifying the high variance of the PLI measurements. In particular, Figure 3(b) delineates the useful variance analysis, by first defining a boundary of the clusters according to the PLI local maxima, and then calculate the variances of PLI within each cluster with necessary l PLI measurements. The details for the computation of k -NN variance clustering to achieve a reduction of T will be explained in Section. 3.2.1.

As illustrated in Figure 3.3(b), each cluster will determine a variance threshold and those variances of PLI that are higher than the variance threshold will be clustered as useful variance, indicated in the shaded areas. The aim of this k -NN variance clustering step is to retain only the sampling points having variance higher than the variance threshold; otherwise the sampling points will be discarded. In Figure 3.3(c), the sampling points at times (t) 10, 20, 45, 71 and 80 are retained for Group 1, and those at times 154, 159, 188, 223, and 268 are retained for

Figure 3.2: General flow of our proposed PRL on the PLI measurement SCA which comprises 6 phases

Figure 3.3: Illustrations of our proposed PRL technique with three PLI measurements and two clusters, phases ① – ⑥ are corresponding to respective (a) PLI measurements, (b) k -NN variance clustering, (c) extracting the useful sampling points from k -NN, (d) scaling the PLI amplitude, (e) assigning weights for each extracted sampling points and (f) generating result of the amplitude scaling which is amplified with weights for cluster 1 and cluster 2 of PLI

group 2. In phase ④ after performing the k -NN variance clustering, the T will be reduced where trivial T are discarded. It is worthwhile to note that small number of useful T which has been obtained is used to perform CbA with only a small number of M . Figure 3.3(d) delineates the amplitude scaling process by scaling the remaining sampling points within cluster based on their mean value. Figure 3.3(e) then performs the Relevance Learning computation, by analyzing the relevancies of the clustered PLI with respect to the secret key. In other words, the weights, which are corresponding to the degree of leakage of each clustered of PLI variances, can be identified. While there are a number of weights in each cluster, the sum of the weights in each cluster is 1. Finally, the weight is assigned to the PLI. Figure 3.3(f) illustrates the updated PLI with respect to its associated weight. With this amplification, the accumulative correlation between the clustered variance (in one cluster) and the processed data can be increased, particularly using the HD leakage model [36]. Therefore, the required number of M can be significantly reduced to reveal the secret key. We will now describe the details of the two main steps, k -NN variance clustering and relevance learning.

3.2.1 k -NN Variance Clustering

By considering the last round of the AES-128 algorithm, the leakage function (L_t) at time t can be related to the PLI profile of the operations as described in (3.1). The three operations in the last round of the AES-128 include the S-Box, ShiftRow and AddRoundKey (exclusive-OR) operations. The input plaintext (a known variable) and the secret key are denoted as χ and κ , respectively:

$$L_t(\chi, \kappa) \propto f(\text{ShiftRow}(\text{SBox}(\chi \oplus \kappa))) \quad (3.1)$$

The CbA will be performed on the last round of operations, where the ciphertext is correlated

with the PLI traces, which leak the information of the secret key as it is referring to the data-dependency. We denote the PLI measurements of the crypto-device to be $P = \{P_1, P_2, P_3 \dots P_t\}$, and the measured sampling points in the time domain to be $\Gamma = \{1, 2, 3 \dots t\} [0, T]$. Each PLI measurement, $P_t(i)$, at the particular sampling point, t , can be decomposed into the leakage function and the noise, where the noise is denoted as $\aleph$ as shown in (3.2):

$$P_t(i) = L_t(\chi, \kappa) + \aleph_t; \text{ where } t \in \Gamma \text{ and } i = 1 \quad (3.2)$$

Figure 3.4(a) delineates the methodological flow of the k -NN variance clustering in our proposed PRL. Each sampling point of PLI is used to calculate the variance, in which the variance is an object. The variance clustering is initiated by determining the number of clusters (k), in path (A) $\rightarrow$ (B) and Figure 3.4(b). In Figure 3.4(b), the variances are first calculated, then the local maximum of the variances are identified and finally each pair of the local maximum will form a cluster (k).

In Figure 3.4(a), For the PLI measurement, $P_t(i)$, in each iteration from $i = 2$ to $i = N$, the variance of the sampling points at time t can be calculated and updated based on (3.3):

$$(\sigma_t^i)^2 = \frac{\sum_{i=1}^N (P_t(i) - \overline{P_t})^2}{N-1}; \quad \text{with } N \geq 2 \quad (3.3)$$

There is only one centroid of variances of the PLI in each cluster. The centroid of each cluster is initially assigned to be 1. The change of the centroid (ΔO_k) of the cluster k , (O_k), will be updated according to the distribution of the clustered variances. The clustering process will be completed when the change is $< 2.5\%$. The k -NN variance clustering on the PLI is as follows:

3.2.1.1 Euclidean distance and regrouping the objects

The distance $d(\cdot, \cdot)$ in (3.4) is an Euclidean distance [49] between an object and a centroid. The Euclidean distance is used for regrouping the objects for each cluster during the iterations. By comparing its distances with the two nearer centroids, the object with the shortest distance to the centroid will be grouped into cluster of that centroid.

$$d(\sigma_t, o_k) = \left(\sum_{i=1}^Z |\sigma_t^i - o_k^i| \right)^{\frac{1}{a}} \quad (3.4)$$

with $a = 2$ for 2-dimension clustering and Z is a total number of objects/variance in each cluster.

3.2.1.2 Update the centroid

The position of the centroids is updated in each iteration, from $i=2$ to $i=N$ iterations, where the flow of iteration process is delineated in Figure 3.4(a). The centroid of the clustered variances represents the region where the data-dependency occurs, implying that those useful sampling points in the clusters contain more leakage information pertaining to the secret key. The centroid position is determined based on two parameters, the sampling points and the objects (variances). As illustrated in Figure 3.3(b), we denote the x -axis to be the sampling points, t , and the y -axis to be the variance, σ . Therefore, the centroid, O_k , is mathematically formulated according to the x - y coordinates as described in (3.5):

$$o_k^i(x, y) = o_k^i \left(\frac{1}{Z_k} \sum_{E \in z} t_E, \frac{1}{Z_k} \sum_{E \in z} \sigma_E \right), 0 < i \leq N \quad (3.5)$$

Where $\mathbb{Z} = \{1, 2, 3 \dots z\}$ $[0, Z]$

3.2.1.3 Determine the final position of the centroids

The clustering process is completed when $\Delta O_k < 2.5\%$ as described in (3.6), where the updated distance is referring to the Euclidean distance as described in (3.4):

$$\Delta O_k = \frac{|o_k^i(x, y) - o_k^{i-1}(x, y)|}{|o_k^i(x, y)|} = \frac{d(o_k^i(x, y), o_k^{i-1}(x, y))}{|o_k^i(x, y)|} < 2.5\% \quad (3.6)$$

After performing the k -NN as illustrated in Figure 3.4, a large number of unrelated sampling points to secret key are now discarded, thus reducing T . Subsequently, the clustered of related sampling points will be used for the relevance learning to determine the weights. In other words, the effectiveness of CbA will be significantly enhanced due to the smaller number of T and M on PLI measurements to successfully reveal the secret key. It is noted that the variance analysis and the k -NN variance clustering are two different processes performed consecutively.

Figure 3.4: (a) Flow chart of k -NN variance clustering on the PLI,
(b) Flow chart of determining the clusters (k) and centroids

3.2.2 Relevance Learning

The relevance factor (Λ_{t_1, t_2}) in (3.7) is the relevancy between the sampling points at time t_1 and time t_2 . This relevance factor is the probability of the secret key being revealed at t_1 and t_2 based on the CbA and is updated in each iteration. In this context, there are a number of relevance factors in each clustered sampling points, which the highest value of Λ_{t_1, t_2} indicates the required M to reveal the secret key based on the CbA are equivalent at t_1 and t_2 :

$$\Lambda_{t_1, t_2} = \frac{(\Omega_{t_1} \cdot \Omega_{t_2}^T) - S(\mu(\theta_{t_1}))}{\left(\sum_{i=1}^Z \sigma_i \right) \cdot \varepsilon} \quad (3.7)$$

where: Ω_{t_1} represents as a vector of PLI at t_1 ;
 $\Omega_{t_2}^T$ represents a transposed of a vector of PLI at t_2 ;

$S(\mu(\theta_i))$ is the error sigmoid function for $\mu(\theta_i)$;
 $\mu(\theta_i)$ is the normalized variance-distance;
 ε is the factor error correction of the objects (σ_i).

$S(\mu(\theta_i))$, $\mu(\theta_i)$ and ε in (3.7) are respectively formulated in (3.8), (3.9) and (3.10):

$$S(t) = \frac{1}{(1 + \exp(-t))} \quad (3.8)$$

$$\mu(\theta_i) = \frac{\sigma_{i1} - \sigma_{max}}{(\sigma_{i1} + \sigma_{max})^2} \quad (3.9)$$

$$\varepsilon = \frac{\sqrt{SNR}}{c} = \frac{\sqrt{\frac{\sigma_{Signal}^2}{\sigma_{Noise}^2}}}{c} \quad (3.10)$$

In (3.10), σ_{Signal} is the clustered variance whereas σ_{Noise} is the discarded variance (based on the k -NN variance clustering). The optimal correction factors, c , for ASIC and FPGA hardware implementations are 1.2 and 1.5 respectively [45]. The relevance learning process, performed after the k -NN variance clustering, determines the weight (w) of each clustered sampling point. The weight is subsequently used to amplify the scaled PLI clustered traces. The amplification for each clustered sampling point, which is associated with the traces, can increase the correlation with processed data (i.e., HD). The weight of each clustered sampling point is updated based on the Hebbian learning [50] of the relevance factor for the traces. As shown in Figure 3.3(e), the weights of particular cluster sampling points exhibit strong data-dependent link with the processed data. The weight is iteratively updated as described in (3.11):

$$w_i = w_{i-1} + (\alpha_i(i) \cdot \Lambda_{i-1,i}^{-1} \cdot \lambda_i); \quad 1 \leq i \leq N \quad (3.11)$$

where $\alpha_i(i)$ is the learning rate, described in (3.12) and λ_i is the normalized rate-change variance, described in (3.13):

$$\alpha_i(i) = \frac{\alpha_0}{1 + \nu \cdot (2i - 1)} \quad (3.12)$$

$$\lambda_i = \frac{|\sigma_i - \sigma_{i-1}|}{\sigma_i} \quad (3.13)$$

The learning rate in (3.12), which determines the convergence weight, is proportional to the SNR of the PLI where the initial value (α_o) is determined by implementations ASIC and FPGA which are 2.5 and 1.75 respectively [40]. In addition, the correction factor v is required to increase the accuracy of computed weights during the relevance learning and reduce the learning error.

The highest value of w is corresponded to the lowest data-dependency of the clustered traces, since the weight is inversely proportional with $\Lambda_{t1,t2}$ as described in (3.11). The amplification suppresses the noise and increases the related secret key PLI traces. The total value of w in one cluster should be unity as to indicate that the information of the secret key in sampling points is fully clustered during the k -NN variance clustering. The termination criteria of the relevance learning are based on the weight error (e) and the learning error (E) which are described in (3.14) and (3.15) respectively:

$$e_i = \sum_{i \in Z} (w_{i-1} - w_i) \quad (3.14)$$

$$E_i = |e_{i-1} - e_i| \quad (3.15)$$

Each w is initially set to be 1 and it will be updated iteratively during the relevance learning process. The E in the relevance learning is evaluated in every iteration until the termination point of the relevance learning is reached when $E_i < 1\%$ at $i = S$, where S is the number of traces required during relevance learning. Hence, we can obtain the converged weights using S traces of PLI. We select 1% as the threshold of learning error during the relevance learning to achieve an equivalent for the error during the testing (i.e., testing error). Hence, the difference of two probabilities between the learning error and the testing error must be small enough (i.e., $< 10^{-4}$) when achieving the converged weights.

We scale the PLI traces with respect to the mean value, before multiplying with $(1+w)$ where $0 < w < 1$. The scaling aims to amplify the data-dependency within each PLI. Referring to Figure 3.3(d), the PLI differences among various traces at the specific time are originally very small. After the amplification, by referring to Figure 3.3(f), the PLI differences among various traces at the specific time are larger and thus making the CbA easier to reveal the secret key. Equation (3.16) describes the computation of amplification with the scaled PLI:

$$P_t = (1 + w_t) \cdot (P_t - \bar{P}_t) \quad (3.16)$$

The relevance learning process is delineated in Figure 3.5 where the flow chart of relevance learning applied to the remaining clustered sampling points. In this relevance learning, the parameters of the initial state before the iteration process are as follows: weight (w) = 1, covariance ($\Omega_{t1} \cdot \Omega_{t2}^T$) = 1, normalized variance distance = 0 and the most leakage sampling point.

Figure 3.5: The flow chart of relevance learning on the clustered traces

3.3 Analyses of Computational Complexity

To analyze the computational complexity of the proposed PRL in profiling PLI measurements, we analyze the core computation into two main parts: k -NN variance clustering and relevant learning. The computational complexity, in our proposed PRL, refers to the simulation time (computational time) of the algorithm, in which the time duration increases proportionally with the size of the inputs (PLI measurement for the clustering variances), and based on the worst-case scenario, $O(\cdot)$. The computational complexity of the k -NN variance clustering and relevance learning are explained in detail as follows:

A. k -NN variance clustering

The basic principle of the k -NN algorithm is to group all the unstructured variances (objects for clustering) into a regular cluster. The first computation in the k -NN variance clustering is to determine the distance between variances and centroid in 2-dimensional spaces (D) and the distances between centroid and threshold boundary (B) with computational complexity $O(D)$ and $O(B)$ respectively. By computing all the distances respect to the variances at t in one cluster, the computational complexity will be $O(t \cdot D)$ and $O(t \cdot B)$ with t is time (sampling point), $t \in \Gamma$. The two distances are compared to make the clustering decision whether clustering or discarding the variances. The total computational complexity during the k -NN variance clustering is formulated in (3.17):

$$O(t \cdot D) + O(t \cdot B) = O(t \cdot D + t \cdot B) \quad (3.17)$$

Since the B and D are constant during the clustering process, the (3.17) is rearranged in (3.18):

$$O(t \cdot (B + D)) = (B + D) \cdot O(t) \simeq O(t); \text{ where } B + D \gg 1 \quad (3.18)$$

Hence, in the formal notation, (3.18) is termed as $O(n)$; $n > t$.

B. Relevance Learning process and Weight updates

The computational process of relevance learning is mainly used on the relevance factor computation as described in (3.7). The computation of $\Omega_{t1} \cdot \Omega_{t2}^T$, requires $2 \times S$ data matrix dimension with quadratic memory spaces required for learning variables. The computational complexity with memory complexity can be formulated in (3.19):

$$O(2 \cdot S) \approx O(S); \quad \text{where the memory complexity is } O(S^2) \quad (3.19)$$

with $S \gg 1$, PLI measurement. The weight updates will require the same computational time as formulated in relevance factor computation, $O(n)$; $n > S$ and memory is $O(n^2)$.

The overall computational complexity is the sum of $O(t)$ and $O(S)$ which are the k -NN variance clustering and the relevance learning respectively, and described in (3.20):

$$O(t) + O(S) = O(t + S) \quad (3.20)$$

Since S is always higher than t , $S \gg t$, the (3.19) is reformed as $O(S)$, $O(n)$; $n > S$. which is linearly proportional to the number of PLI measurements, hence termed as $O(n)$. For comparison, Figure 3.6 shows the time complexity (in second) of our proposed PRL which is lowest among various reported techniques.

Figure 3.6: Comparison of time complexity on various profiling techniques

3.4 Experimental Results

The AES-128 algorithm is implemented on Sakura-X board [14] embodying an FPGA. For CbA analysis, we contemplate both power dissipation and EM signal as PLI, and for brevity, their CbAs are CPA and CEMA respectively. The PLI is measured in an oscilloscope with the maximum sampling rate of 2.5GS/s to obtain high resolution of PLI.

To evaluate the efficacy of our proposed PRL for CPA and CEMA, we perform four different AES-128 comparison scenarios. The first scenario is the conventional AES-128 algorithm (conventional S-Box) with four different clocking frequencies, i.e., 2MHz, 9.6MHz, 12MHz and 16MHz. The employment of four different frequencies is to examine the robustness of our proposed PRL against different noise levels generated at different frequencies. In the second scenario, the performance of the proposed PRL profiling is compared with various reported profiling techniques. For the third scenario, we apply a vertical hiding countermeasure to resist CbA. The vertical hiding countermeasure adopted here is essentially a compensator [45] and equalizer [54] to balance the magnitude of power dissipation (throughout the AES-128 operations). In the last scenario, we apply horizontal hiding countermeasure to highly secure the CbA against EM. The horizontal hiding countermeasure implemented here is essentially the operation randomization by means of the State-shift Local Clocks [55] (SsLC) to moderate the time occurrences of the AES-128 operations. We now describe the detailed evaluation on these four scenarios as follows:

3.4.1 AES-128 performed with Different Frequencies

We perform the CPA evaluation followed by the CEMA evaluation for the PLI power dissipation and EM respectively. Figure 3.7(a)-Figure 3.7(d) delineate the power dissipation

measurements of AES-128 operating at 2MHz, 9.6MHz, 12MHz and 16MHz respectively. At different frequencies, their power dissipation profiles are different, partly due to the parasitic inductance and capacitance in bonding wires [36] of the circuit board. These parasitic inductance and capacitance can serve as filters to affect the rate of changes of dynamic power dissipation. At the low frequency (2MHz), 10 rounds of AES-128 operations are still noticeable by matching 10 peaks. However, at high frequencies, the peaks to match the 10 rounds of AES-128 are undistinguishable (16MHz). The higher frequency could infer that the k -NN variance clustering requires more PLI measurements to cluster the salient features.

Figure 3.8(a)-Figure 3.8(d) present the result of k -NN variance clustering for AES-128 operating at 2MHz, 9.6MHz, 12MHz and 16MHz respectively. The explanation of Figure 3.8 as follows: First, at the low frequency (2MHz), we cluster 96 sampling points (out of 3,800 or 2.5%) per measurement. These sampling points only the useful sampling points for CbA. Should the last round be only considered for CbA, we require only 13 sampling points per measurement for CbA. Second, when the frequency becomes higher, the remained sampling points are increased, from 96@2MHz $\rightarrow$ 158@9.6MHz $\rightarrow$ 179@12MHz $\rightarrow$ 191@16MHz. Should the last round be only considered for CbA, the clustered sampling points increase from 13@2MHz $\rightarrow$ 16@9.6MHz $\rightarrow$ 21@12MHz $\rightarrow$ 27@16MHz. Put simply, even though the frequency is scaled-up to $8\times$ (2MHz to 16MHz), the remained sampling points are increased only $\sim 2\times$ (13@2MHz to 27@16MHz) to be sufficient for CbA. The low number of useful sampling points (which is highly correlated with processed data) lead to a shorter time to reveal the secret key based on the CbA, low number of M . Figure 3.9 presents the learning error E for the clustered variances in C#10 (last round) at

Figure 3.7: Different power dissipation measurements for the AES-128 running at (a) 2MHz, (b) 9.6MHz, (c) 12MHz and (d) 16MHz, exhibit more switching noise generated at the higher frequency, hence the peaks are undistinguishable

Figure 3.8: The k -NN variance clustering, clustered (shaded area) and discarded sampling points of the power dissipation measurements in Figure 3.7, (a) 2MHz, (b) 9.6MHz, (c) 12MHz and (d) 16MHz respectively

different frequencies, indicating the convergence speed (in terms of iterations) of the relevance learning of clustered sampling points to assigned weights. At the low frequency where the noise is relatively low, it requires 256 iterations to reach $E < 1\%$. Even at the high frequency @ 16MHz, the PRL merely requires 538 iterations (about $\sim 2\times$ compared to that when operating @ 2MHz) to reach $E < 1\%$. We use the converged weights, obtained during

the training based on the criteria that $E < 1\%$, for SCA testing, which SCA is based on the computational of correlation in the CbA.

Figure 3.9: The learning error of the clustered variances in C#10, where the best (2MHz) and worst (16MHz) scenarios require 256 and 538 iterations respectively to obtain the convergence weights

Figures 3.10(a) and (b) present the results of CPA attacks @ 2MHz, by plotting the correlation coefficient (r) [36] vs power traces (M), with the standard technique and with our proposed PRL. Figures 3.11 (a) and (b) present the same except the frequency is at 16MHz. Based on Figure 3.10(a), when the reported technique [38] is applied to power dissipation measurements, the secret key (black line) is successfully revealed at 421 power traces. By applying our proposed PRL to power dissipation measurements, the same secret key can be revealed at only 23 power traces, as shown in Figure 3.10(b). For the reported technique [38], when the operation frequency is increased to 16MHz, the number of traces required to reveal the secret key is increased significantly, i.e., having 7,847 traces as shown in Figure 3.11(a). As shown in Figure 3.11(b), the number of traces is reduced to 142 traces to reveal the secret key with our proposed PRL applied to power dissipation measurement. Our proposed PRL is effective because it is able to filter out the irrelevant sampling points

which can hinder the correlation between the PLI and processed data. In addition to the reduction number traces required to reveal the secret key, the correlation coefficient between PLI and processed data is also increased. At 2MHz, r is increased by $\sim 2\times$ (from 0.26 in Figure 3.10(a) to 0.6 in Figure 3.10(b)) with our proposed PRL applied and by $\sim 7\times$ (from 0.05 in Figure 3.11(a) to 0.35 in Figure 3.11(b)) at 16MHz. The higher r alludes to the easier successful and highly effective CPA compared with the reported technique [14].

Figure 3.10: Correlation coefficient vs power traces of the CPA on AES-128 @2MHz (a) without PRL profiling (b) with our proposed PRL profiling

Figure 3.11: Correlation coefficient vs power traces of the CPA on AES-128 @16MHz (a) without PRL profiling (b) with our proposed PRL profiling

Table 3.1 tabulates the CPA performance on AES-128 based on the reported technique [38] without profiling and with our proposed PRL. Four frequencies are evaluated to compare the number of iterations during the PRL (N and S), sampling points T and number of traces

M . The numbers shown in parentheses are normalized with the performance of the proposed PRL and the symbols $\uparrow$ and $\downarrow$ are respectively used throughout this chapter to indicate the number of times increased and decreased of the performance when compared to the reported techniques [14], [38], [45] and [55]. As high frequency is associated with high noise, the N and S are proportionally increased, with average of 330 and 408 for N and S respectively. On average, the CPA with our proposed PRL reduces $31\times$ of T , $23\times$ of M , collectively having $713\times$ $M\cdot T$ improvement. The N and S can be considered as the processing overhead in our proposed PRL. Nevertheless, N and S are smaller compared to $M\cdot T$ improvement for CbA. Consider now the EM signal of AES-128 where the EM signal @ 16MHz is extracted and presented in Figure 3.12. Although the peaks of EM measurements can be easily identified and matched to each round iteration, there are many smaller local peaks (with smaller amplitudes) during the transition between rounds. These smaller local peaks are (switching) noise due to the transition data in the local BUS [36] of the crypto-device. With the presence of these noises, Figure 3.13 shows the learning error at different frequencies, indicating more iterations to achieve the convergence weights during the relevance learning.

Table 3.1: COMPARISON OF N , S , T AND M BASED ON CPA IN FOUR DIFFERENT FREQUENCIES

Freq. (MHz)	k -NN (N)	$S @ E<1\%$	Sampling Point (T)		Traces (M)	
			Proposed PRL	Reported [38]	Proposed PRL	Reported [38]
2.0	206	256	13(1 $\times$)	600(46 $\times\uparrow$)	23(1 $\times$)	421(18 $\times\uparrow$)
9.6	319	401	16(1 $\times$)	300(19 $\times\uparrow$)	87(1 $\times$)	806 (9 $\times\uparrow$)
12.0	334	437	21(1 $\times$)	700(33 $\times\uparrow$)	117(1 $\times$)	1,054(9 $\times\uparrow$)
16.0	461	538	27(1 $\times$)	700(26 $\times\uparrow$)	142(1 $\times$)	7,847(55 $\times\uparrow$)
Avg.	330	408	(1 $\times$)	(31 $\times\uparrow$)	(1 $\times$)	(23 $\times\uparrow$)

Table 3.2 tabulates the CEMA performance on AES-128 based on the standard technique [14] without any profiling and with our proposed PRL. The four frequencies are evaluated to compare PRL (N and S), M , and T with the numbers shown in parentheses are normalized with the reported technique [14]. The reduction of traces by means of our proposed PRL is

inversely proportional with the operation frequencies. In other words, the fake peaks (noise) hinder the CEMA to reveal the secret key at high frequency, although the useful sampling points have been filtered during the relevance learning. The smaller peaks (noise) at higher frequency can reduce the accuracy of relevance learning and hence requires more number of iterations ($N = 1,887$ and $S = 2,103$) compared with power dissipation to achieve the convergence weights. On average, the CEMA with our proposed PRL reduces $25\times$ of T , $5.5\times$ of M , collectively having $137\times$ $M\cdot T$ improvement.

Figure 3.12: Fake peaks of the EM signal are noticeable at high frequency

Figure 3.13: EM at 16MHz requires 2,103 iterations to achieve $E < 1\%$ due to small peaks hinder the relevance learning

Table 3.2: COMPARISON OF N , S , T AND M BASED ON CEMA IN FOUR DIFFERENT FREQUENCIES

Freq. (MHz)	k -NN (N)	$S @ E < 1\%$	Sampling Points (T)		Traces (M)	
			Proposed PRL	Reported [14]	Proposed PRL	Reported [14]
2.0	823	1,013	21(1×)	500(24×↑)	42(1×)	386(9×↑)
9.6	897	1,707	51(1×)	700(24×↑)	127(1×)	751(6×↑)
12.0	1,098	1,846	31(1×)	780(25×↑)	554(1×)	2,316(4×↑)
16.0	1,887	2,103	33(1×)	900(27×↑)	2,035(1×)	6,876(3×↑)
Avg.	1,176	1,667	(1×)	(25×↑)	(1×)	(5.5×↑)

3.4.2 Comparison with Various Profiling CbA on AES-128

Table 3.3 tabulates the performance of our proposed PRL, comparing with other profiling techniques. We compare these techniques in terms of N , S , T , M and r by performing it with the optimum frequency (MHz). It has been analytically described in Sec. 3, the computational complexity of the proposed PRL is linear ($O(n)$), which shows lower complexity compared with the reported techniques. Our proposed PRL profiling features high constant complexities of B and D which are 205 and 3,864 respectively. The big constants B and D imply that our proposed PRL profiling requires a relatively high computational resource. However, our proposed PRL profiling has an advantage on its linear computational complexity $O(n)$ and therefore the big constant values will not deteriorate the performance significantly of our proposed PRL profiling for large input data (i.e., $> 10^6$ PLI measurements). The profiling performance in terms of N is ranged from $4\times$ to $10\times$ higher for the reported techniques while S is not comparable, since no learning algorithms employed in the reported profiling techniques. The performance of the reported profiling techniques in terms of T and M is normalized with our proposed PRL and the overhead is denoted as $\uparrow$ while the reduction of r is denoted as $\downarrow$. As tabulated in Table 3.3, we show that by profiling the PLI with our proposed PRL, the secret key is easily being revealed, since value of r is noticeable in small M .

Based on Intel Xeon processor @3.7GHz, the time duration required to perform the k -NN variance clustering is 3.36 minutes and for the relevance learning is 7.84 minutes, while overall time duration for PRL profiling is 11.2 minutes. In SCA, it requires another 2.31 minutes to reveal the secret keys and the overall time taken for both PRL profiling and SCA is 13.51 minutes.

We further compared the profiling performance of our PRL and with other machine learning algorithms such as Artificial Neural Network (ANN), Support Vector Machine (SVM) and Random Forest. The data structure of PLI measurement fits well with ANN while unfits for the SVM and random forest. The performance of the ANN is slow and inefficient for both noisy PLI and large inputs.

3.4.3 Vertical Hiding Countermeasure on AES-128

Table 3.3: COMPARISON OF COMPUTATIONAL COMPLEXITY, PROFILING AND CbA PERFORMANCES BASED ON OUR PROPOSED PRL AND VARIOUS PROFILING TECHNIQUES AT OPTIMUM FREQUENCY

Technique	Freq. (MHz)	Computational Complexity	Profiling performance		CbA					
					CPA			CEMA		
			$N(\times\uparrow)$	S	$T(\times\uparrow)$	$M(\times\uparrow)$	$r(\times\downarrow)$	$T(\times\uparrow)$	$M(\times\uparrow)$	$r(\times\downarrow)$
Averaging [36]	2	$O(n^3)$	3,210(9.7)	-	620(31)	432(18)	0.11(5.5)	812(29)	457(10)	0.09(4.8)
Filtering [37]	2	$O(n^2)$	2,051(6.2)	-	840(42)	893(38)	0.24(2.5)	728(26)	927(22)	0.08(5.3)
PCA [39]	2	$O(n^2)$	1,925(5.8)	-	900(45)	354(23)	0.13(4.7)	448(16)	893(21)	0.21(2.0)
LDA [40]	12	$O(n^2)$	1,876(5.7)	-	460(23)	378(16)	0.21(3.0)	756(27)	578(14)	0.11(3.9)
SSA [41]	2	$O(n^2)$	1,862(5.6)	-	420(21)	413(18)	0.11(5.5)	868(31)	624(15)	0.13(3.3)
T-Test [43]	12	$O(n^3)$	3,143(9.5)	-	530(26)	1,504(65)	0.41(1.5)	728(26)	1,958(47)	0.10(4.3)
SOSD [44]	12	$O(n^2)$	1,561(4.7)	-	870(43)	1,251(54)	0.36(1.7)	644(23)	1,587(38)	0.12(3.5)
SyncSCR[48]	9	$O(n^2 \cdot \log(n))$	2,795(8.5)	-	540(27)	1,360(59)	0.28(2.2)	476(17)	1,421(34)	0.05(8.6)
PSD[52]	12	$O(n^3)$	2,989(9.0)	-	620(31)	254(11)	0.54(1.1)	1,148(41)	531(13)	0.16(2.8)
DbLA [53]	12	$O(n^2)$	2,867(8.7)	-	570(28)	301(13)	0.34(1.8)	700(25)	872(20)	0.23(1.9)
Proposed PRL	2-16	$O(n)$	330(1.0)	408	20(1)	23 (1)	0.61(1.0)	28(1)	42(1)	0.43(1.0)

As aforementioned, we apply the compensator [45], which serves as an equalizer [54] to equalize the power dissipation for different input values (plaintext). The compensator is realized by using dual-rail logic for the S-Box circuit which is the most critical block in AES-128 [41]. In the presence of the compensator to hide the magnitude of PLI, it is our interest

to evaluate how the k -NN variance clustering to detect the useful sampling points and how the relevance learning to distinguish the secret-key related PLI measurements.

Table 3.4 tabulates the CPA and CEMA performance on AES-128 based on the dual-rail logic [43] without any profiling and with our proposed PRL, in terms of N , S , T and M respectively. In CPA perspective, the k -NN variance clustering requires 10,327 power dissipation traces to identify the useful sampling points and is $\sim 2\times$ higher than that required in EM signal. The relevance learning requires 2,732 iterations to converge the weights, and similarly $\sim 2\times$ higher than that required in EM signal. Put simply, the dual-rail logic is more effective to equalize the power dissipation but slightly less effective to equalize the EM signal. Our proposed PRL is effective to reduce both M and T for CPA and CEMA. The CPA and CEMA with our proposed PRL, on average, reduce $33\times$ of T , $53\times$ of M , collectively having $1,749\times$ $M\cdot T$ improvement.

Table 3.4: COMPARISON OF N , S , T AND M IN CBA BASED ON PROPOSED PRL AND COMPENSATOR [45]

CbA	k -NN (N)	S @ $E < 1\%$	Sampling Points (T)		Traces (M)	
			Proposed PRL	Reported [45]	Proposed PRL	Reported [45]
CPA	10,327	2,732	27(1 $\times$)	900(33 $\times$ ↑)	439(1 $\times$)	40k (91 $\times$ ↑)
CEMA	5,173	1,453	39(1 $\times$)	1,350(34 $\times$ ↑)	1.9k(1 $\times$)	29k (15 $\times$ ↑)
Avg.	7,750	2,092	(1 $\times$)	(33 $\times$ ↑)	(1 $\times$)	(53 $\times$ ↑)

3.4.4 Horizontal Hiding Countermeasure on AES-128

As stated in Section 3.4, we apply the operation randomization [55] to moderate the time information of AES-operation. The operation randomization is realized by using the SsLC, dynamically varying the clock frequency from 2.4MHz to 24MHz. In the presence of the SsLC to hide the time information of PLI, the variance distribution of PLI will be increased

and dependency with processed data is reduced. It is also our interest to evaluate how the k -NN variance clustering to detect the useful sampling points and how the relevance learning to distinguish the secret-key-related to PLI.

Table 3.5 tabulates the CPA and CEMA performance on AES-128 based on SsLC without profiling and with our proposed PRL in terms of variables N , S , T and M . We show that the AES-128 implementation based on SsLC without profiling requires 1M (10^6) PLI measurements (both power dissipation and EM) reveal the secret key. With our proposed PRL applied on SsLC, it requires 6,224 of N and 6,254 of S , on average, we need merely 1,200 traces to reveal the secret key based on the CbA. The reduction of T and M are $30\times$ and $1,400\times$ respectively, hence it is $42,000\times$ of $M\cdot T$ improvement. The overheads of N and S are significantly lower and can be negligible.

Table 3.5: COMPARISON OF N , S , T AND M IN CbA BASED ON PROPOSED PRL AND SsLC [55]

CbA	k -NN (N)	$S @ E<1\%$	Sampling Points (T)		Traces (M)	
			Proposed PRL	Reported [55]	Proposed PRL	Reported [55]
CPA	5,031	5,531	83(1 $\times$)	800(10 $\times$ ↑)	1,254 (1 $\times$)	1M (800 $\times$ ↑)
CEMA	7,417	7,517	23(1 $\times$)	1,200(51 $\times$ ↑)	523 (1 $\times$)	1M (2.0k $\times$ ↑)
Avg.	6,224	6,524	(1 $\times$)	(30 $\times$ ↑)	(1 $\times$)	(1.4k $\times$ ↑)

3.5 Conclusions

In this chapter, a novel profiling PRL technique has been proposed to extract the highly correlated PLI with the processed data, hence enhancing the efficiency of the SCA. In this chapter, it has been described also that the sampling points of the highly correlated PLI are identified using the k -NN variance clustering in the PRL technique. Thereafter, relevance learning is applied to the clustered sampling points to increase their correlation with the

processed data. Subsequently, the experimental results were shown that the number of PLI measurements for the proposed PRL based on CbA on AES-128 with countermeasure is 523 traces respectively, thus $2,000\times$ more efficient than the reported profiling techniques. Finally, the proposed PRL technique can be employed to effectively evaluate the proposed countermeasure techniques in the subsequent chapters.

CHAPTER 4

HIGHLY SECURED STATE-SHIFT LOCAL CLOCK CIRCUIT TO COUNTERMEASURE AGAINST SIDE CHANNEL ATTACK

4.1 Introduction

As discussed in Chapter 3 that the SsLC is one of the countermeasure techniques used to examine the robustness of the proposed PRL. In this chapter, the proposed SsLC [55] is explained in detail which is categorized as a hiding countermeasure technique. The hiding countermeasure technique can be classified into two types, namely vertical hiding (V-hiding) and horizontal hiding (H-hiding).

The main idea of the V-hiding is to balance the power dissipation of different processed data values performed in the crypto-device. It was reported [47] that the V-hiding techniques can secure the PLI against SCA of 10^6 traces. However, in practice, the resistant level is secured to $\sim 10^3$ traces due to glitch effect [36]. In this context, the resistant level of the countermeasures is defined as the minimum number of measurements required to reveal the secret key. The techniques based on the V-hiding include noise injection [58], reconfigurable IC [59], equalizer [54] and compensator [45]. The noise injection and the reconfigurable IC generate dummy operations which can balance the power dissipation for different values of the processed data. However, it incurs $\sim 53\%$ power overhead and 64.3% of 2×10^3 additional logic gates in implementation. The equalizer and compensator are somewhat vulnerable against EM-based attack (e.g., CEMA) and the secret key can be revealed in ≤ 500 EM measurements.

The H-hiding countermeasure rearranges the sequence of the operations, such as random shuffling [47], Random Process Interrupt [41] and Random Changing of Clock and Frequency [48]. They hide the PLI by introducing random delay in time domain. However, the propagation delay which is randomly implemented can lead to be unequal distribution for the leakage information. Consequently, the secret key can be easily revealed (i.e., $<2 \times 10^3$ traces) at the sampling points where the secret key is highly distributed. In addition, the reported H-hiding techniques are highly vulnerable against the EM-based attack (e.g., CEMA) [45] due to not effectively balanced at certain points (i.e., at V_{DD} and GND) of the encryption-device which most likely leaks the related EM signal against SCA.

In this chapter, a proposed SsLC [55] which features two Local Clock (LC) generator is presented. The first LC (SsLC_1) generator controls the input and output rate of the AES-128 cryptographic device at First-In First-Out (FIFO) module, and the second LC (SsLC_2) generator controls the encryption operation rate of the AES-128 implementation. There are four main features in the proposed SsLC countermeasure technique. First, it embodies a Finite State Machine (FSM) which can be employed to regularly shift the timing operation of cryptographic algorithm implementations. Thus, the correlation of the PLI with the processed data is significantly reduced due to dynamically changes the occurrences of encryption operation in time domain. Second, the PLI which encompasses a secret key is spread over in time domain to reduce the probability of revealing the secret key. Third, the power dissipation overhead is negligible and hence it is highly applicable for low power applications. Fourth, the regular state shifting technique in the SsLC is able to hide multiple

PLIs, i.e., power dissipation and EM signals, concurrently. The power is measured based on the current flowing across the 1Ω resistor while the EM signal is measured by means of the

Figure 4.1: Clocking systems in a) AES-128 implementation, b) Master Clock, c) proposed Local Clock _1 and d) Local Clock _2

EM probe with 0.9mm resolution [14]. The measurement is focused at the area where the EM signal is the highest emission. Based on the experimental results in FPGA, the proposed SsLC countermeasure technique features wide distribution of PLI in time domain, dissipates 2.77mW of power and emits 12.2mV/m of EM signal @ 2.4MHz. By comparing with the conventional AES-128 implementation [51], the dissipated power is 3.8 $\times$ lower with the dissipated power is 10.5mW [51]. Furthermore, the secret key of the cryptographic algorithm is highly secured where it remains unbreakable at 10^6 traces. In comparison with the reported counterparts, the resistance of the proposed SsLC against SCA is significantly improved as the number of power dissipation and EM traces to reveal the secret key has increased by >18 $\times$ and >25 $\times$ respectively. Consequently, the correlation coefficient between the PLI and the processed data is reduced by 3.5 $\times$.

The remainder of the chapter is organized as follows. The AES-128 implemented with the

proposed SsLC is discussed in Section 4.2. The experimental results and comparison with reported techniques are explained in Section 4.3. Finally, the conclusions are drawn in Section 4.4

4.2 Clocking System in AES-128 implementation

Figure 4.1(a) presents a simplified circuit block diagram of an AES-128 system implementation. The input FIFO (FIFO_In) and output FIFO (FIFO_Out) are used to control the input/output data rates for AES-128 implementation, which the FIFO interfaces are adopted in the FPGA board for SCA analysis. Figure 4.1(b) delineates a conventional clocking method where a main clock (M_Clk) is used to connect ports A and B, driving all the registers in the AES-128 implementation. Since the M_CLK has a constant frequency and hence the timing operation of AES-128 implementation is fixed (t). The PLI of the AES-128 implementation which encompasses the secret key can be easily identified. Therefore, this AES-128 implementation is vulnerable against the CPA and CEMA attacks.

To moderate the timing, to shuffle the time sequence, we propose SsLC (SsLC_1 is opposed to M_CLK) as shown in Figure 4.2(c) to regularly shift the time of the local clock pulses during the operation of AES-128 implementation. The assertion of the local clock is controlled by an FSM embedded in SsLC_1. For instance, using a 3-bit counter in the FSM where there are 8 states, the local clock is not only asserted once for every 8 states, but also transited irregularly from one state to the other state. Figure 4.2(a) illustrates the timing diagram of the local clock at various states. The irregular shifting means that the assertion of the local clock can occur at any one of the 8 states. Put simply, the SsLC_1 functions beyond a mere clock divider, randomizing the local clock assertion and yet providing an equal clock

distribution (for all states). The number of states in the FSM can be increased to further increase the shifting range of the local clock, hence enhance the resistance level of AES-128 implementation against the CPA and CEMA attacks. Nevertheless, the speed is inversely proportional to the number of states with the worst-case scenario is that the M_Clk is reduced by factor of states number in the FSM.

Figure 4.2: Timing diagram of a) SsLC_1, b) SsLC_1 and SsLC_2 shifted based on the M_Clk

To further enhance the resistant level against the SCA, in time domain, we further propose two degrees of local clock shifting by augmenting another local clock as shown in Figure 4.1(d), terms as the SsLC_2 which embeds two counters. The first counter in the SsLC_2 has the number of bit in the SsLC_1, aiming to tune the state numbers and synchronizing data rates with SsLC_1. The second counter in the SsLC_2 generates the local clock by comparing the shifting value with the state numbers. Figure 4.2(b) illustrates the timing diagram of the SsLC_2. By integrating the local clock and main clock, the local clock in the SsLC_2 can be regularly shifted relative to the SsLC_1 and M_Clk, without disrupting the AES-128 operation. In this context, the adversary might find it even harder to synchronize between the rates of the output data (SsLC_1) with the rate of the PLI (SsLC_2) to align the traces before the CPA computation. This is due to the SsLC_2 creates another degree of shifting direction

in the local clock for AES-128 relative to input/output data. In addition, the local clock frequency in the SsLC_2 will be varied (based on different shifting) to misalign the traces in the correlation computation and hence protecting the secret key of AES-128 against multiple PLIs based SCA.

Figure 4.3: Block diagrams of a) SsLC_1 and b) SsLC_2

For completeness, Figure 4.3(a) and Figure 4.3(b) delineate the block diagrams of the SsLC_1 (one 3-bit counter) and the SsLC_2 (two 3-bit counters) implementations, which can be easily modeled in Verilog HDL. The FSMs therein are cyclic machines and the state sequences are controllable, featuring an equal distribution when compared with the random distribution in the reported higher order masking technique [60].

4.3 Experimental Results

We employ Sakura-X, FPGA board, [14] with the M_Clk frequency of 12MHz to implement the AES-128 algorithm incorporating our proposed SsLC circuit as shown in Figure 4.1(d) and Figure 4.3. The 16-byte plaintexts are randomly generated and input to the FPGA board whereas the 16-byte secret key is fixed. The frequency of the SsLC_2, SsLC_1 and M_Clk are measured in oscilloscope with sampling rate of 2.5GS/s as illustrated in Figure 4.4. Based on the Figure 4.4, we observe that the shifting of the local clock in SsLC_1 is somewhat

irregular, and that in SsLC_2 is even more irregular. Table 4.1 tabulates the measurements on these clocks, including the maximum (Max) frequency, the minimum (Min) frequency, the average (Avg) frequency, and the standard deviation (SD). The SsLC_2 varies from 0.39MHz to 2.4MHz whereas SsLC_1 is fixed @ 1.33MHz. The SD of SsLC_2 is the highest among other clocks, indicating that the SsLC_2 widely distributes the PLI in time domain to hide the leakage information against the SCA (i.e., $r_{i,j,t} \simeq 0$).

Figure 4.4: The measured clocks signals of SsLC_2, SsLC_1 and M_CLK

Table 4.1: MEASUREMENT FREQUENCY OF SsLC_2, SsLC_1 AND M_CLK

	Max @ MHz	Min @ MHz	Avg @ MHz	SD @ kHz
SsLC_2	2.4	0.39	0.76	133.50
SsLC_1	1.3	1.19	1.31	46.97
M_Clk	12.0	11.80	11.93	9.75

Figure 4.5 presents the measured power dissipation and EM signals generated during the AES-128 encryption process. The AES-128 system requires 11 (local) clock cycles (1 key expansion and 10 round iterations) to encrypt one plaintext. The measurement of the power dissipation and EM emission are 2.77mW and 12.2mV/m respectively.

Figure 4.5: Power dissipation and EM measurements of the AES-128 implementation

To evaluate the timing shifting effect, we compare our proposed technique against the reported technique [60] using the software time shuffling. The variance of 10^3 power dissipation is computed and compared with the reported technique [60] as presented in Figure 4.6. As shown in the Figure 4.6, the reported technique has one peak region where the high-power variance concentrated at the sampling point range of 40 to 60. This implies that PLI is most likely leaking in this range, hence the adversary finds it easier to analyze the PLI. However, our proposed SsLC technique generates multiple (local) peak regions. In other words, the PLI of our proposed SsLC is well distributed in the time domain, and the adversary would be hard to identify the correlated PLI. Viewed differently, the adversary would require substantially higher computational time to align the PLI traces (measurements) due to the multiple states (time) shift operation in the SsLC. In addition, the active attacks such as fault injection will be deterred as the randomization will not be synchronized and the encryption process will generate incorrect ciphertext.

Figure 4.6: Distribution PLI based on variance value of reported higher order shuffling technique [60] (red) and proposed SsLC (black)

In contrast to power dissipation, EM signals are relatively sensitive and easy to be measured without the need of modifying the hardware interconnections (fully non-invasive), in this context, the CEMA is highly effective SCA approach. To the best of our knowledge, none of the reported techniques has resistant level against CEMA with more than 10^6 EM traces. Figure 4.7 presents the CEMA attack on single byte of AES-128 secret key with 10^6 of EM traces. The secret key (142D) is clearly distinguished with other key candidates ($r_{i,j,t} = 0.1$) based on the M_Clk implementation as shown in the Figure 4.7(a). Conversely, during the SsLC implementation, the correlation is reduced by $3.5\times$ ($r_{i,j,t} = 0.1$ to be $\tilde{r}_{i,j,t}=0.028$) as shown in the Figure 4.7(b) and make the secret key remains undistinguishable with other key candidates. Therefore, the adversary may not be able to identify the secret key in our proposed SsLC implementation based on the CEMA attack.

Figure 4.7: a) Secret key (142D) is revealed in reported technique [47] implementation and b) Secret key is not revealed in the proposed SsLC implementation

We further perform the CPA and CEMA attacks based on the reported technique [50] and our proposed SsLC technique. Figure 4.8 delineates the minimum PLI traces required to break the key based on the reported technique [60] and the proposed SsLC. Figure 4.8(a) shows that the correlation of correct secret key of the reported techniques [50] is revealed at 5.4×10^4 and 4×10^4 traces for CPA (black line) and CEMA (red line) respectively. Whereas Figure 4.8(b) shows that the correlation of correct secret key of the proposed SsLC technique is still lower than other key candidates (grey line) at 10^6 traces ($N=10^6$). The performance is significantly improved $>18\times$ and $>25\times$ based on the CPA and CEMA attacks respectively.

Figure 4.8: Correlation coefficient VS Traces of a) reported higher order shuffling technique [60] and b) Proposed SsLC

4.4 Conclusions

In this chapter, a highly secured SsLC countermeasure technique has been proposed with four main features. First, it embodies an FSM which can be employed to regularly shift the timing operation of the AES-128 implementation. Second, the PLI which encompasses a secret key is spread over in time domain to reduce the probability of revealing the secret key. Third, the power dissipation overhead is negligible and hence it is highly applicable for low power applications. Fourth, the regular state shifting technique in the SsLC is effectively hide multiple PLIs, i.e., power dissipation and EM signals, concurrently. In this chapter, the experimental results in the FPGA is also presented. It shows that the proposed SsLC countermeasure technique features wide distribution of PLI in time domain, dissipates 2.77mW of power and emits 12.2mV/m of EM signal @ 2.4MHz. Furthermore, the correlation coefficient is reduced by $3.5\times$ and with 10^6 traces of power dissipation and EM measurements, the secret key of the cryptographic algorithm remains unbreakable. Consequently, the resistance against SCA is significantly improved by $>18\times$ and $>25\times$ based on the CPA and CEMA attacks respectively when comparing with the reported counterparts.

CHAPTER 5

A HIGH THROUGHPUT AND SECURE AUTHENTICATION-ENCRYPTION AES-CCM ALGORITHM ON ASYNCHRONOUS MULTICORE PROCESSOR

This chapter presents the proposed AMP-MP in AES-CCM based on an asynchronous multicore platform implementation to achieve high throughput and highly secure against SCA.

5.1 Introduction

Internet Protocol security (IPsec) is a vigorous layer-network protocol to enhance the security level of digital communication systems and it encompasses three processes: key management, authentication and confidentiality [6]. The key management, which is based on the Random Number Generator (RNG) module, establishes secure key distribution using the encryption algorithm and key updates between the transmitter and the receiver afore-exchanging the messages (plaintexts).

As delineated in Figure 5.1, the authentication process verifies the originality of the plaintext while the confidentiality is to ensure the security of the messages by performing encryption-decryption process at the transmitter and receiver respectively. At the transmitter, the Message Authentication Code (MAC) is generated based on the plaintext and secret key through an authentication algorithm. The MAC is used to validate the plaintext upon received at the receiver. The plaintext is authenticated when the MAC of the plaintext at the receiver (MAC_R) is corresponding with the transmitted MAC (MAC_T). On the other hand, the plaintext is encrypted into ciphertext to protect the confidentiality of the plaintext against adversary.

The authentication and confidentiality can be performed simultaneously using a symmetric encryption algorithm such as AES [6], which is widely used for encryption. The authentication adopts the chaining mode operation while the confidentiality (encryption) is based on the counter mode operation as illustrated in Figure 5.2(a). In Figure 5.2(a), the AES is used for encryption on the Counter with Chaining Mode (CCM) algorithm, abbreviated as AES-CCM [61]. Thus, the AES-CCM can provide two attributes simultaneously, the high assurance of authentication and confidentiality of the messages during the communication.

The AES-CCM algorithm is largely employed in standard protocol of vehicular communication, i.e., Control Area Network Bus (CAN-Bus) [62] and FlexRay [63], to secure the data communications against various attacks, such as Man-in-Middle-Attack (MMA) [64] and SCA [36]. In addition, the applications of the AES-CCM algorithm are equally propitious for Internet of Thing (IoT) and wearable devices with the employment of low power and small area AES accelerator (gate counts $< 2 \times 10^3$) [65] in the AES-CCM.

Besides the aforementioned advantages, the AES-CCM algorithm implementation remains challenging, particularly on achieving the high throughput for communication systems and the high security against physical hardware attacks. Further, the throughput and security are somewhat limited in synchronous platform. Various techniques have been reported to overcome the said challenges of the AES-CCM which can largely be categorized into two groups: hardware FPGA and software approaches. The hardware FPGA approaches are pipeline-reconfigurable AES-CCM in FPGA implementation [66], memoryless AES-CCM implementation [57], single-core reconfigurable AES [68], Unified Data Authentication Encryption [69] and Ultra-Low Power AES-CCM IP core [70]. Due to the placement and routing optimization in an FPGA, the physical leakage such as the power dissipation during

Figure 5.2: The Message Authentication Code (MAC) is generated at the Transmitter and verified at the Receiver

Figure 5.1: AES-CCM Architecture based on (a) Conventional (b) Our proposed AMP-MP

the encryption process can be vulnerable to SCA [45] where the secret key is breakable at 400 traces. In the reported software implementations of AES-CCM [71], the iterative round computation of AES requires $>10^3$ clock cycles for each round [36], which is much higher than FPGA with requires only one clock cycle for each round computation of AES algorithm.

In this chapter, we propose an Authentication based Matrix-transformation cum Parallel-encryption implemented in Multicore Processor (AMP-MP) [82] to achieve comprehensive performance, high throughput and secure AES-CCM, yet low power dissipation overhead as shown in Figure 5.2(b). Our proposed AMP-MP comprises four main features as follows:

First, we propose to use the matrix multiplication in $GF(2^8)$ computation for authentication to

transform each message of 16 plaintexts into 1 plaintext which is inherently irreversible [61] as to break the dependency between the authentication input and the messages. In Figure 5.2(a), the authentication input (e.g., B_0) of conventional AES-CCM is operated sequentially (in chaining mode) to generate the authentication key (Y). With our proposed matrix multiplication in $GF(2^8)$ delineated in Figure 5.2(b), the 16 authentication inputs (e.g., B_0 - B_{15}) are transformed into one authentication key (e.g., Y_0). In other words, the transformation, such as in hash function [72], aims to reduce the length of the authentication input. Consequently, the speed of the authentication process can be increased by $\geq 32\times$ for both the transmitter and receiver collectively when compared to conventional implementations. Second, we reschedule the operations of 3 AES encryptions in 3 different cores such that their PLI, i.e., power dissipation and EM are compensated and equalized, thus reducing the correlation of the physical leakage with the processed data by $\geq 3\times$. Third, the intermediate result of the AES-CCM is propagated asynchronously within the cores of the multicore processor, as to randomize the physical leakage information with processed data. Therefore, the security feature of the AES-CCM can be further enhanced against SCA by another $3\times$. Furthermore, the power dissipation and EM in asynchronous circuits can be significantly reduced by $>2\times$ through reducing the spurious switching due to the clock-less circuit protocol. Fourth, the leakage characteristics of the secret key based on HW and HD are observable in the key management (an IPsec function), due to the limitation-randomization of True-RNG (TRNG) [73] on key updates [74]. Hence, we further propose a key adjusting technique by leveraging the function of AES algorithm (i.e., S-Box) and byte circular-shifting when similar patterns of HW and HD of the secret key are detected. Hence, the secret key is protected against the key updates. The fourth feature provides highest level of security protection of AES-CCM if the SCA successfully can reveal all the secret key during key updates and subsequently the similar

pattern of the secret key is revealed. Furthermore, our proposed AMP-MP is realized on an 8-bit 9-core asynchronous processor and fabricated based on 65nm CMOS process.

The experimental results show that the throughput of the authentication process of AES-CCM based on our proposed AMP-MP implementation is 13.54Gbps while the throughput for authentication and encryption collectively is 8.32Gbps. The multicore based on the asynchronous-logic exhibits uniform distribution of power variance ($\sigma_{power} < 20\%$) for AES operation which can reduce the correlation in SCA by $>2\times$. The power dissipation of authentication process based on our proposed AMP-MP is $307\mu\text{W}$, while overall authentication and encryption process dissipates $311\mu\text{W}$ to perform 256 plaintexts, $\sim 2\times$ lower than a reported counterpart [69]. Based on SCA evaluation, single (power) and multi-channel (both power and EM) attacks [28], the secret key is unrevealed even after 5×10^5 plaintext measurements. Furthermore, we evaluate the robustness of our proposed AMP-MP against collision attacks [6] with 2^{16} sets of plaintexts. The experimental results show that the MAC of our proposed AMP-MP has a zero collision. In addition, with our proposed key adjusting technique on 256 randomly generated secret keys, the 4 similar patterns within the 256 secret keys are completely removed.

The remainder of this chapter is organized as follows. Review of the AES-CCM is presented in Section 5.2. The proposed AMP-MP and its implementation in asynchronous-multicore processors are described in Section 5.3. The experimental results and comparisons of the performance with various reported of AES-CCM are presented in Section 5.4. Evaluation of our proposed AMP-MP based on SCA is presented in Section 5.5. Finally, the conclusions are drawn in Section 5.6.

5.2 Review of AES-CCM

The operation of AES-CCM, which simultaneously provides two essential processes: data authentication and encryption to secure digital communication systems, is based on IEEE Standard 802.11i [76] of wireless local area network. There are 3 inputs to the AES-CCM: (1) The Data which will be both authenticated and encrypted, named as plaintext (P), bit-length: P_L ; (2) The Associated-Data (A) with bit-length: A_L , which is a header of P , will be authenticated but not encrypted; and (3) A unique value generated randomly by TRNG for every block of plaintext and independent of secret key, denoted as a Nonce (N), bit-length: N_L . The N will be assigned to the A and P in the sequence of N, A, P to form a sequence of blocks B (i.e., $B_0, B_1, \dots, B_r$) for authentication process as presented in Figure 5.3. The authentication output (T) is generated with the bit-length (T_L) where the bit range of the MSB is from 32-to-64 bits [6], $32 < T_L < 64$.

Figure 5.3: Authentication process

The bit-length for each block B of authentication input is 128 bits and the first block (B_0) is assigned exclusively for Nonce, thus $N_L < 128$ bits. The number of blocks (r) for an

authentication input is dependent with N_L , A_L and P_L which the authentication process is based on Cipher Block Chaining (CBC) [6], chaining mode. In the operation as presented in Figure 5.2(a), the authentication output of the previous block (B_{N-1}) will be mixed (e.g., XOR) with the input of the current block (B_N), which is $0 < N < r$. In hardware implementations, the number of clock cycle (Clk) required to perform authentication of AES-CCM depends on r , $Clk \sim r$. Therefore, the bigger value of r , the authentication process will require longer time duration. The CBC in AES-CCM could leak information of secret key (by means of SCA [36]) by correlating the ciphertext and measured EM signal during the authentication.

In the encryption process as shown in Figure 5.4, the generation of ciphertext and MAC, in parallel or Counter (Ctr) mode encryption is performed to both the plaintext and the authentication key. The Ctr block, which consists of 128 bits for each block, is including the value of N . The number of Ctr blocks is m which depends on r , where $0 \leq r \leq m$, (Ctr_m). The input for AES is Ctr_m value and the output AES encryption (S_m) is XOR-ed with plaintext, except for Ctr_0 which is XOR-ed with T . The ciphertext generated based on the plaintext is appended with the ciphertext generated based on the authentication message, MAC. It is worthwhile to note that the AES-CCM employs only forward AES encryption [6] to generate MAC_T and MAC_R , hence reducing the area overhead.

Figure 5.4: Counter mode is performed to generate ciphertext and MAC

5.3 Proposed Authentication Based Matrix-Transformation Cum Parallel-Encryption in Multicore Processor (AMP-MP) on AES-CCM

There are four main features of our proposed AMP-MP on AES-CCM which are elaborated as follows:

5.3.1 Proposed matrix multiplication in $GF(2^8)$

Figure 5.5 presents the schematic flow of the matrix multiplication on reducing the inputs for authentication from 256-block to 16-block of inputs. The raw-data as an input comprises; password, security code and message (plaintext), are sorted according to conventional order, N , A , P . The sorted data is then proceeded for formatting to form blocks of data (i.e., B_0 to B_{255}), which is one block consists of 16 bytes of data. The N is corresponded to B_0 and appended to 0 value (byte) if $N_L < 16$ bytes. Based on conventional implementation [6], the A can be formatted in two or more blocks if $A_L > 16$ bytes. In our proposed matrix multiplication, we employ 16 bytes for A_L , hence it is assigned only to one block (i.e., B_1) of authentication input. Finally, the P is assigned to the remaining blocks (i.e., B_2 - B_{255}).

The block B , in the format of row vector, is the input for matrix multiplication. The row vector matrix is equivalent to 1×255 matrix in which one block (i.e., B_0) is categorized as a vector element. The input of matrix multiplication is reformatted and transposed into square matrix 16×16 matrix and subsequently multiplied with constant-column matrix (constant vector) based on $GF(2^8)$ computation to generate 1×16 vector matrix of B' . The block B' is the input for authentication process to generate T (based on Y_r) has been reduced from 256 to 16 blocks through matrix multiplication with constant vector. The multiplication process in $GF(2^4)$ is equivalent to the computation in one of the AES operations, *Mix Column* operation, using a

constant square matrix (4×4) to be multiplied with 16 bytes of intermediate result of AES round computation. Our proposed matrix multiplication has similar principle as a length reduction technique on the plaintext (i.e., using the hash function [72]) which can be applied for authentication in AES-CCM. The fundamental difference of our proposed matrix multiplication with the conventional AES-CCM is different T value. In order to obtain correct T and successfully validate the plaintext, the pre-requisite of our proposed matrix multiplication is to be implemented in both transmitter and receiver.

Figure 5.5: Schematic flow of our proposed matrix multiplication in $GF(2^8)$

Figure 5.6: Matrix multiplication in $GF(2^8)$

The detail operation of matrix multiplication is explained as follows. Considering the

polynomial in $\text{GF}(2^8)$, which has the form $f(x) = a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$ represents one byte data in block B . The $f(x)$ is subsequently multiplied with x which is the representation for one-byte constant vector, as expressed in (5.1).

$$x \times f(x) = (a_7x^8 + a_6x^7 + a_5x^6 + a_4x^5 + a_3x^4 + a_2x^3 + a_1x^2 + a_0x) \bmod m(x) \quad (5.1)$$

Where $m(x)$ is the finite field in family of $\text{GF}(2^8)$ used in one of AES functions [1], *Mix Column*, as shown in (5.2) as follows:

$$m(x) = x^8 + x^4 + x^3 + x + 1 \quad (5.2)$$

The multiplication of a_7x^8 by $m(x)$ is shown in (5.3) as follows:

$$a_7x^8 \bmod m(x) = a_7((x^8 + x^4 + x^3 + x + 1) - (x^8)) = a_7(x^4 + x^3 + x + 1) \quad (5.3)$$

If $a_7 = 0$, the result of $x \times f(x)$ is a polynomial degree less than 8 and polynomial result of a_7 is negligible. Otherwise, polynomial degrees are reduced in modulo $m(x)$ which is expressed in (5.4) as follows:

$$x \times f(x) = (a_6x^7 + a_5x^6 + a_4x^5 + a_3x^4 + a_2x^3 + a_1x^2 + a_0x) + a_7 \cdot (x^4 + x^3 + x + 1) \quad (5.4)$$

As a summary, the overall bitwise operation for each byte in matrix multiplication can be expressed in (5.5) as follows:

$$x \times f(x) = \begin{cases} (a_6a_5a_4a_3a_2a_1a_0) & ; a_7 = 0 \\ (a_6a_5a_4a_3a_2a_1a_0) \oplus (00011011) & ; a_7 = 1 \end{cases} \quad (5.5)$$

The constant vector, which is used in multiplication of each byte in block B , consists of 16 bytes vector elements. We use a generator of a finite field $\text{GF}(2^4)$ to generate the constant vector elements where a generator for $\text{GF}(2^4)$ is based on the family of polynomial equation: $g^4 + g^3 + g^2 + g + 1$. The list of possible values generated by $\text{GF}(2^4)$ computations are tabulated in Table 5.1 as follows:

Table 5.1: THE LIST OF GF(2⁴) FOR CONSTANT VECTOR

Polynomial	Binary	Hexadecimal
0	0000	0
1	0001	1
g	0010	2
$g + 1$	0011	3
g^2	0100	4
$g^2 + 1$	0101	5
$g^2 + g$	0110	6
$g^2 + g + 1$	0111	7
g^3	1000	8
$g^3 + 1$	1001	9
$g^3 + g$	1010	A
$g^3 + g + 1$	1011	B
$g^3 + g^2$	1100	C
$g^3 + g^2 + 1$	1101	D
$g^3 + g^2 + g$	1110	E
$g^3 + g^2 + g + 1$	1111	F

The corresponding bytes in block B (i.e., B_0) is multiplied by a constant vector (i.e., default sequence is 0 - F) and summed up as byte component in B'_0). The computation of multiplication is expressed in (5.6) as follows:

$$b_0^1 = ((b_0^1 \cdot 0) + (b_0^2 \cdot 1) + \dots + (b_0^{16} \cdot F)) \cdot \text{mod}(m(x)) \quad (5.6)$$

Where b_0^1 = first byte of block B'_0 and b_0^1 = first byte of block B_0 . For instance: the input of block B'_0 , 16-byte, is generated from matrix multiplication of $B_0, B_{16} \dots B_{240}$ (256-byte) with constant vector and B'_1 , 16-byte, is generated base on matrix multiplication of $B_1, B_{17} \dots B_{241}$ (256-byte) and the constant vector. The reduction bytes 256-to-16 is inherently first preimage resistant [6] which the input is unpredictable based on the output and values of the constant vector. The flow of matrix multiplication over GF(2⁸) for 256 of B (e.g., plaintext) to generate 16 of B' for authentication is illustrated in Figure 5.6. The matrix multiplication is implemented

in both transmitter and receiver to obtain a valid authentication tag. With our proposed matrix multiplication in $GF(2^8)$, the authentication process, for one-time communication from the transmitter to the receiver, can be significantly enhanced by $>32\times$ faster than conventional AES-CCM implementation. The generator for $GF(2^4)$ will randomize the byte sequences for every specific length of plaintext (e.g., 256 plaintexts) to provide another security layer in authentication. In this context, the sequence of the number is not fixed and unpredictable by adversary. The sequence of constant vector element is randomized with $16! \approx 2.09 \times 10^{13}$ possibilities, hence secure against preimage attacks.

Figure 5.7 presents the circuit implementation of our proposed matrix multiplication. The sequences of pre-stored constant vector are randomized based on TRNG which is initiated with random binary *Seed* input. The randomized sequence will be propagated to the binary multiplication module and *Nonce_R* which the additional security feature is appended to *Nonce* for authentication. It is worthwhile to note that the randomization of the sequence is performed once during the authentication and encryption. In other words, randomization of the sequence will not affect the throughput of the authentication process. The input of 128-bit *B* is multiplied with 64-bit randomized constant vector by referring to (5.6) where the output is restricted by $GF(2^8)$. If the value of an a_7 is high, the multiplication result is XOR-ed with binary $GF(2^8)$ values, 00011011. Otherwise, the result of multiplication is made transparent to the output. The multiplication process is continuously performed until the 16-byte register of B_0 is fully occupied by the multiplication result and subsequently the *Finish* signal is triggered to 1.

Figure 5.7: Circuit implementation for the matrix multiplication

5.3.2 Proposed Parallel and Rescheduled AES encryption

Figure 5.8 shows the circuit implementation of the AES-CCM authentication-encryption in 9-core synchronous-logic 8051 microcontroller [26] with ANoC protocol. To optimize the performances, 3 cores are used for parallel encryption, 2 cores are used for authentication and the other 4 cores are assigned for receiver mode AES-CCM. Input/output (*I/O*) interface of 12 ports is grouped into four blocks (*North-East-South-West*) as presented in Figure 5.8(a). The *N*, *A* and secret key are propagated through *North* (*I/O* 0-2) while the plaintext and ciphertext are interfaced to *West* (*I/O* 9-11) and *East* (*I/O* 3-5) respectively. The 32-bit of control signal for multicore processors consists of three signals: *Address_data* is to determine the address location of the data in the respective core, *Rec_core* is reconfigurable signal to control the flow of data from ANoC to the core during the AES encryption and *En_core* is the enabled signal to control on and off the respective core. The data valid signal (*Dvld*) is activated when the ciphertext has been completed and finish signal (*Finish*) is high when the encryption of the total plaintexts (i.e., 254 plaintexts) are complete.

Figure 5.8: Design of ANoC based 9-Core processor (a) Overall architecture of multicore and (b) Parallel and rescheduling three AES within three cores

The main objective of having 3 cores for parallel-pipeline encryption is to achieve high throughput encryption and EM compensation of AES concurrently. The performance of parallel-pipeline encryption is analyzed based on Multiple Instructions and Multiple Data (MIMD) mode [77]. In this analysis, two operations are defined: the vector and the scalar operations. The vector operation involves data transfer protocol between cores while scalar operation is operating exclusively in a core. The fraction of vector operation is F while the scalar is $(1 - F)$ and their throughputs are denoted as b_v and b_s for vector and scalar operations respectively. The b_v is dependent with b_s in terms of the number of cores n used for parallel-pipeline ($b_v = n.b_s$) and total throughput (b_{tot}) is determined based on parallel current flow

formula as expressed in (5.7).

$$\frac{1}{b_{tot}} = \frac{F}{b_s} + \frac{1-F}{nb_s} = \frac{nF-F+1}{nb_s} = \frac{1+(n-1)F}{nb_s}; b_{tot} = \frac{nb_s}{1+(n-1)F} \quad (5.7)$$

The analysis of EM compensation of parallel-pipeline is explained as follows. The EM flux (Φ_x) of EM field ($\vec{B}_x$) generated in core x is linearly proportional with the dissipated current (I) and core area (A) as expressed in (5.8), where μ , l and r are constant permeability of the conductor, length and radius of the measured point respectively.

$$\Phi_x = \vec{B}_x \cdot A = \frac{\mu \cdot I_x \cdot l \cdot A}{2\pi r^2} \quad (5.8)$$

With vector operations performed during the encryption, the Φ_x at individual core is randomized for different instructions (round iterations) of AES. Since EM flux is a vector form, the magnitude will be cancelled to each other when two EM flux generated by two different cores with opposite directions (i.e., $\Phi_x + \Phi_y \approx 0$). Furthermore, the AES encryption is reconfigured (reschedule the operations) within three cores base on MIMD such that the critical rounds (e.g., first and last round of AES-128) are performed at different cores. This is to de-correlate the leakage information of EM signal with processed data in the same core encryption (e.g., Core _{x}) as expressed in (5.9).

$$\sum \Phi_{x,total} = \Phi_{x,first} + \Phi_{x,middle} + \Phi_{x,last} \xrightarrow{\text{uncorrelated}} \text{processed_data} \quad (5.9)$$

The four main functions of AES algorithm (*Add Round Key*, *S-Box*, *Shift Row* and *Mix Column*), which are constituted in static and vector operations, are assigned to individual cores. To reduce the delay operation and static-operation complexity, the secret key is expended (1-to-10 round of secret key) only at Core₁ and the result is propagated to Core₂ and Core₃. Each core encrypts the *Ctr* with secret key up to 9th round and transmit the intermediate result (i.e., *int. plaintext₀*) to the next core for the last round operation. For instance: Core₁ receives

$plaintext_0$ (T_L) and generates $ciphertext_1$ as the output while the Core_2 receives $plaintext_1$ and generate $ciphertext_2$ and eventually Core_3 receives and generates the $plaintext_2$ and $ciphertext_0$ respectively.

5.3.3 Proposed novel asynchronous data flow

The main advantage of asynchronous circuits in terms of countermeasure against SCA is the ability to randomize the time occurrences of the sensitive leakage information (i.e., secret key) to reduce the correlation of physical leakage with the processed data. Furthermore, in terms of implementation with its clock-less architecture, the asynchronous circuit dissipates lower energy [75] and lower EM signal compared with synchronous circuits. In this chapter, we propose to implement a novel asynchronous architecture of Sense Amplifier Half Buffer (SAHB) [77] in ANoC protocol to achieve fastest data flow between cores and yet high resistance of AES-CCM against SCA (i.e., lower correlation coefficient).

Figure 5.9 delineates the SAHB circuit architecture, constituted in ANoC, features dual-rail logic interface, where each signal interface accompanied with complementary logical signal which is indicated as n coefficient). The dual-rail feature can achieve high robustness, high speed and low energy dissipation [26]. The data flow is based on handshake protocol, where the acknowledge signal at the right channel (R_{ack} and nR_{ack}) is activated when output (D_{out} and nD_{out}) is completed. The left acknowledgement signal (L_{ack} and nL_{ack}) when the SAHB block is ready to receive the input (D_{in} and nD_{in}). The SAHB embodies evaluation block and Sense Amplifier (SA) to accommodate latency during the parallel AES encryption and the SA is based on cross couple latch and positive feedback to speed up and latch the output. Finally, SA is tightly coupled to reduce the switching nodes, hence reducing the dynamic power dissipation during the authentication and encryption.

Figure 5.9: Main channel interface (L and R) of SAHB circuit and cell template

5.3.4 Proposed key adjusting technique

The key update is one of the methods to countermeasure the SCA [74] by changing the secret key periodically based on timing and size of the messages; i.e., every 65 minutes or 500 megabytes plaintexts. The main disadvantage of this method is compromising the throughput caused by the additional time required for both key exchange mechanism and public key management system to update the secret key of AES for both transmitter and receiver. For the worst-case scenario, the secret key can be successfully revealed by means of SCA within the time frame of key update and the adversary can learn the patterns of the key. Based on the observed key patterns, the adversary (i.e. MMA) can predict the subsequent secret key without performing the attack (i.e., SCA). To encounter against the leakage patterns, we propose a key adjusting technique for AES, performed by circular shifting 128-bit (8-bit basis) secret key and transform the value in non-linear function S-Box as shown in Figure 5.10 to make the subsequent key unpredictable.

Our proposed key adjusting technique is activated when the HW and HD patterns are detected (HW_dctd and HD_dctd). The HW pattern is based on the number of bit-one ('1') in each byte of secret key while the HD pattern is detected by observing the bit changes of two consecutive secret key, using XOR logic for each bit. Each 8-bit input secret key ($Data_in$) can be circulated up to $8\times$ until the pattern is fully randomized (in $Data_out$).

Figure 5.10: Circuit schematic of key adjusting technique for each 8-bit $Data_{in}$

The key adjusting technique precedes the key expansion in AES when operation mode is either encryption or decryption. To illustrate the security point of view, suppose the adversary successfully eavesdrops and reveals the pattern of secret key through the SCA in key management system. By predicting the subsequent key (in HW and HD), the ciphertext will be unsuccessfully recovered by adversary due to the key has been corrected by key adjusting technique. The synchronization of the adjusted key between the transmitter and the receiver can be performed in two methods. First, the key is partitioned into two partitions and encrypted using the previous key while the second one is using asymmetric cryptographic algorithms (i.e., RSA, Diffied-Hellman or ECC) [6]. Finally, the validation of authentication key is performed with $< 0.3\%$ latency overhead.

5.4 Experimental Results in Asynchronous Multicore

The proposed AMP-MP is implemented in multicore ANoC embody the SAHB architecture. In this context, we designed the multicore ANoC in full-custom IC based on the 65nm CMOS process where the input and output interfaces are exactly the same as portrayed in Figure 5.8. Microphotograph of multicore ANoC, leveraging the SAHB architecture, is presented in Figure 5.11 which occupies 0.105mm^2 area. Based on the 9-core processor with our proposed

AMP-MP, we assign three cores (Core_1 to Core_3) for parallel encryption and two cores (Core_4 and Core_5) for authentication, while the other four cores (Core_6 to Core_9) are reserved for decryption process (receiver mode).

Figure 5.11: Microphotograph of the fabricated multicore (a) 9-Core with respective ANoC
(b) A unit core is interfaced with ANoC embodies SAHB

To obtain the optimum performance for parallel computations, we assign the maximum number of cores for parallel encryption/decryption. In addition, we set the supply voltages (V_{DD_ANoC} and V_{DD_Core}) and frequency operation (f_{Core}) to 0.8V, 1.2V and 100MHz respectively for the optimum performance of asynchronous multicore. To evaluate the efficacy of our proposed AMP-MP leveraged on multicore ANoC, we perform three experiments, based on main features of our proposed AMP-MP implemented in AES-CCM and compare the results with conventional implementation and the reported techniques.

5.4.1 Matrix multiplication over $GF(2^8)$ and authentication

We generate 4,096 bytes randomly in Hex number as an input for *Raw-Data* and subsequently sorted to respective; 254 plaintexts, 6 bytes N and 12 bytes A . The *Nonce_R* (64 bits/8 bytes)

is generated by sequence randomization module which is then appended to N before formatting to series blocks (256 blocks B). The operation of matrix multiplication in $GF(2^8)$ is performed after reformatting and transposing of 256 blocks B as has been presented in Figure 5.5. We measure power dissipation during the matrix multiplication and authentication of the authentication input block B (i.e., plaintexts) in oscilloscope with a fixed sampling rate of 2.5GS/s. The power dissipation measurement profile is shown in Figure 5.12 where the first 16 peaks represent the dissipated power during the matrix multiplication operation in $GF(2^8)$ computation and higher peaks are generated during the authentication process. The 256 blocks of B are transformed into 16 blocks of B' (16 peaks) in $2.079\mu s$ with average dissipated power is $357\mu W$ and followed by the authentication process which is performed in Core_4 and Core_5 through ANoC in $0.34\mu s$ where its average of power dissipation is $631\mu W$. The overall throughput of matrix multiplication for 256 blocks B and followed by authentication process is 13.54Gbps in $2.42\mu s$ with average power dissipation is $\sim 500\mu W$. The additional 76% power during authentication is dissipated due to overflowing data in ANoC to enable the continuity of parallel AES in two cores. The continuity of encryption is realized by propagating the plaintext to the AES for every round computation without waiting the last round AES computation.

Figure 5.12: Power dissipation of matrix multiplication and authentication process

By comparing the throughput performance based on the same parameters (i.e., clock, supply voltage and single core processor) and same platform, our proposed AMP-MP on AES-CCM is $35.91\times$ faster than conventional implementation. In this comparison, the conventional implementation [71] dissipates $721\mu\text{W}$ power and requires $86.7\mu\text{s}$ to both authenticate and encrypt 256 blocks B .

We further investigate the preimage resistant of the proposed AMP-MP on the input plaintext (message) transformation by performing the experiment on second preimage attack to MAC of 2^{16} (65,536) sets of input plaintext. The second preimage resistant is based on the weak collision attack where it is infeasible to obtain another 16 input plaintexts of B^{11} which has the same output transformation B' with the original 16 plaintexts of B^1 . Each set of input plaintext comprises 16 blocks of input plaintext as authentication input B_0 - B_{15} . The MAC of two different authentication inputs (i.e., B_{x0} - B_{x15} and B_{y0} - B_{y15}) are compared to investigate the collision. Table 5.2 tabulates the performances of the first (1st) and second (2nd) preimage attacks based on the MAC of our proposed message transformation. The performance is compared with two reported hash functions [61] such as SHA-256 and SHA-512 which are

commonly used for message authentications (i.e., MAC).

Table 5.2: THE 1ST AND 2ND PREIMAGE ATTACKS BASED ON THE MAC OF 2^{16} SETS OF INPUT PLAINTEXT (MESSAGE)

Algorithm	1 st	2 nd (coll.)
SHA-256	Yes	No (3)
SHA-512	Yes	No (1)
Our proposed message transformation	Yes	Yes (0)

As tabulated in Table 5.2, no collision occurs in MAC of our proposed input plaintext transformation in 65,536 sets of input plaintext as indicated the number of collision (coll.) is 0. On the other hand, the number of collisions for SHA-256 and SHA-512 algorithms are 3 and 1 respectively. This partly due to “birthday paradox” [64] from 2^{16} sets of input plaintext. Hence, our proposed message transformation is highly secured against second preimage attack. Based on our analysis on the second preimage attacks, it is worthwhile to note that our proposed matrix multiplication in $GF(2^8)$ also secured against strong collision attacks as no pairs of input plaintext are found to be collided in MAC.

5.4.2 Parallel encryption and rescheduling of the AES in ANoC

Figure 5.13 presents the power dissipation profile measured from three encryption cores to generate ciphertext and MAC through ANoC. The parallel encryption requires $3.26\mu s$ and dissipate $307\mu W$ average power to encrypt 256 *Ctr* blocks and XOR operation with plaintext. The sudden impulsive spike of power dissipation is generated when loading the 255 plaintexts into XRAM before the encryption process. With rescheduling mode in three cores ANoC, the throughput of parallel 3 AES encryption is 10.05Gbps. On the other hand, the conventional AES encryption in single core requires $20.94\mu s$ to encrypt 256 plaintext which the throughput is 1.56Gbps. As a result, the parallel encryption with reschedule the AES has increased the encryption process by $6.42\times$. It is worthwhile to note that the encryption process can be

performed in parallel with authentication, 5 cores, hence the overall throughput is mainly determined by encryption, which dissipates $315\mu\text{W}$ nominal power during $3.93\mu\text{s}$, thus the throughput is 8.32Gbps .

Figure 5.13: Power dissipation of parallel encryption in three cores which dissipates power $307\mu\text{W}$ in $3.26\mu\text{s}$

We further analyze the optimum throughput performance for both parallel encryption/decryption (to demonstrate the transmission and receiver modes for AES-CCM). The configuration is based on 6 cores in multicore while the other 3 cores are assigned for the authentication of encryption-decryption. Table 5.3 tabulates the throughput performance of 6 cores configurations for parallel encryption and decryption. The optimum throughput performance is achieved at 3 cores for both parallel encryption and decryption where the throughputs are 8.32Gbps and 8.27Gbps respectively.

Table 5.3: THE THROUGHPUT PERFORMANCE OF PARALLEL ENCRYPTION AND PARALLEL DECRYPTION IN 6 CORES CONFIGURATION

Number of cores for parallel Enc./Dec.		Throughput (Gbps)	
Encryption	Decryption	Encryption	Decryption
1	5	2.74	14.86
2	4	5.17	11.02
3	3	8.32	8.27
4	2	11.24	4.91
5	1	15.33	2.53

For SCA evaluation, we measure the EM signal of our proposed AMP-MP during the authentication and encryption in AES-CCM and compare with a single core synchronous implementation @100MHz as presented in Figure 5.14. The EM signal of 132 dB (in $\mu\text{A/m dB}$) is generated from single core implementation potentially leak the information of the secret key AES-CCM through SCA. With our proposed AMP-MP in multicore, the EM signal of ANoC with dual-rail SAHB is reduced to 93 dB (reducing 29.5%) which shows lower data dependency (noisy) than synchronous counterpart during the authentication-encryption in AES-CCM. Besides rescheduling 3 AES encryption in 3 cores, the expanded keys are distributed to 3 cores with random route through ANoC protocol. With the random route key distributions, the direct attack on EM signal is further secured with only 0.21% of the latency overhead. In addition, the total EM generated from the reconfigured of three cores in the AES operation in (5.9) can improve the resistance level against SCA by breaking the correlation of EM signal with processed data.

Figure 5.14: EM signal of our proposed AMP-MP in multicore and single core synchronous implementation

To further evaluate the robustness of ANoC on leakage distribution against SCA, we determine the variance of 256 aligned power dissipation based on conventional implementation single core synchronous and with our proposed AMP-MP in multicore ANoC. Figure 5.15 presents

the variance of two consecutive operations *S-Box* and *add round key* which are comprising the value of the secret key. It shows that the power variance of AES in ANoC implementation is equally distributed (uniform) due to the result of dual-rail asynchronous SAHB architecture. Comparing with single core implementation (dashed line) which is leaking information at highest variance (probability 80% of key leaked at particular sampling point: 10), our proposed AMP-MP in multicore ANoC implementation is secured against SCA with leakage probability is $< 20\%$.

Figure 5.15: Comparison of the variance power dissipation during the encryption

With lower leakage probability (i.e., $< 20\%$), the adversary will require the higher number of traces to reveal the secret key in the SCA. This is mainly due to the leakage information of the secret key are sparse and randomized in the sampling points with low leakage probability. Consequently, it will increase the search space for the secret key in sampling points of measurements. Therefore, the higher number of traces required to reveal the secret key for each sampling point.

5.4.3 Key adjusting technique

The fundamental idea of the key updating system is to restrict the use of similar secret key for

one time communication. Thereafter, the secret key has to be updated to further thwart the adversary against deciphering the correct ciphertext. To demonstrate the pattern based attack against key updating system, we measured the EM leakage signal of the stateful key updating [74] for synchronous applications. The secret key is updated based on the master key (the initial secret key) to determine the subsequent secret keys. Figure 5.16 shows the EM signal of the key updating system in 10^4 sampling points. It shows that six similar patterns are detected in Figure 5.16(a) which lead to the future computational attacks of the subsequent secret key. However, with our proposed key adjusting technique, in Figure 5.16(b) four similar patterns on key updating system are omitted hence secured against patterns based attack. The additional $\sim 22.4\%$ of the EM noise (from 0.56 to 0.68 $\mu\text{A/m dB}$) are generated by circuit of shift register.

Figure 5.16: The EM signal patterns are detected based on (a) reported key updating system and with (b) our proposed key adjusting technique the patterns are omitted

To further evaluate the robustness of our key adjusting technique, we adopt the TRNG to

generate the random key during the key updates. The randomness of bit sequence, by leveraging de-correlator [73] based on circular shift register to ensure the key value, is uniformly distributed and independent while the XOR gate is to restrict the value under $GF(2^8)$ computation. We randomly generate 256 the secret key based TRNG to simulate $256 \times$ key updates [74] and observe the HD and HW patterns. It turns out that, although the values are independent and unpredictable, the HD and HW patterns are repeatable, predictable and recurrence as presented in Figure 5.17. Two HW patterns are detected at 1-to-3 and 6-to-8 while the two HD patterns are detected at 0-to-2 and 5-to-6, collectively 4 patterns detected, as shown in Figure 5.17(a). Upon detecting the patterns, we perform our proposed key adjusting technique, which has been shown in Figure 5.10, during the key updates and hence, the similar patterns are completely removed as presented in Figure 5.17(b).

Table 5.4 tabulates the measurement results on variance distribution, time, power dissipation, EM signal and number of cores used during key adjusting technique implementation. It shows that, when both patterns are detected, the distribution of HD and HW are highest which lead to reveal and predict the secret key easily. The performance of latency is $0.12\mu s$ which dissipates $41.35\mu W$ and 7.6dB for power and EM respectively. In this experiment, the resulted overheads are 4.95%, 6.33% and 8.17% for latency, power and EM respectively which is proportional to the number of cores used for key adjustment. The brute-force method is not applicable to attack the key adjustment process since the key is corrected in 128 bits randomly. The adversary still requires huge 2^{128} possible keys to predict the correct key.

Figure 5.17: HD and HW distribution of 256 secret key (a) HD and HW patterns are detected and (b) Performing key adjusting technique to remove patterns

Table 5.4: MEASUREMENTS OF TIME, POWER AND EM OF KEY ADJUSTING

Patterns detection		Distribution (σ)		Time (μ s)	Power (μ W)	EM (dB)	Core(s)
HD	HW	HD	HW				
0	0	0.32	0.37	0.00	0.00	0.00	1
0	1	0.87	0.29	0.08	10.33	4.7	1
1	0	0.14	0.86	0.06	34.21	5.1	1
1	1	0.93	0.84	0.12	41.35	7.6	2

5.5 Experimental Results in SCA

We measure the Success Rate (SR) of the leakage information SCA based on Test Vector Leakage Assessment (TVLA) [74] to detect the presence of the leakage information in the measurements (power or EM) and determine the leakage probability. The SR is generally

determined before quantifying the SCA resistant in the correlation based SCA [36]. Figure 5.18 presents the SR of the power and EM analyses based on single core and multicore implementations where the multiple PLI measurements are power and EM obtained based on asynchronous multicore implementation.

Figure 5.18: The SR of power and EM analyses based on single core and multicore implementations

The SR of the PLI measurements shows that the conventional AES in single core implementation has a probability of 70% to leak the secret key at 2×10^4 power traces (i.e., 70% of the secret keys can be revealed at 2×10^4 traces). On the contrary when AES is implemented in asynchronous multicore, the SR is reduced to ~28% even after 2×10^5 traces. We further determine the SR of the multi leakages measurements where two leakages measurements (power and EM) are incorporated for SCA. The SR of multi leakages measurements is saturated at 50% after 2×10^5 traces. The result implies that the leakage information of the secret key can only be revealed mostly 50% at 2×10^5 traces based on our proposed asynchronous multicore implementation.

In addition, the SR of the key adjusting technique is also determined based on TVLA which refers to four scenarios in Table 5.4. The SR of the key adjusting technique when implemented in 2 cores for both power dissipation and EM signal measurements are 14.87% and 9.32% respectively. The 2 cores refer to the highest distribution of HD and HW. The lower SR is obtained due to the random delay from ANoC and low SNR of the PLI measurement generated from the operation of the shift register circuits. Therefore, information of the secret key is secured either in sampling points or in amplitude of the measurements.

To further quantify the resistance of our proposed AMP-MP against SCA, we analyze both CPA and CEMA attacks. Our attacking point is focusing on Core_1 to Core_5 where the authentication and encryption of AES-CCM are performed. The SCA evaluation is constituted into two parts, Single Channel [45] which measures only one physical leakage information (i.e., power dissipation) and Multi-Channel [74] which is employing more than one physical leakage information (i.e., both power dissipation and EM signal) in the computation of correlation to reveal the secret key. In this SCA evaluations, we targeted two leakage functions: first round decryption (i.e., *inverse S-Box* and *Add Round Key*) and last round encryption (i.e., *S-Box*, *Shift Row* and *Add Round Key*). The two leakage functions are based on the HW leakage model to determine the intermediate values (i.e., processed data). Finally, secret key is analyzed based on the correlation between PLI measurements and processed data.

5.5.1 Single Channel SCA

Figure 5.19(a) and Figure 5.19(b) shows the CPA plots of correlation vs key candidates at 5×10^4 power traces for single core and multicore ANoC implementations respectively. In the single core implementation, the secret key 176 is successfully revealed at 39,000 traces with 0.29 correlation and SNR 4.4 dB as presented in Figure 5.19(a). On the contrary, by activating

the ANoC between Core_4 and Core_5, the correlation is reduced by $17\times$ ($0.29 \rightarrow 0.017$) with SNR of 0.96 dB as presented in Figure 5.19(b). The correlation reduction is mainly due to the dual-rail asynchronous SAHB in ANoC which uniformly distributes the variance of leakage information in time domain.

Figure 5.19: CPA attack on AES-CCM based on (a) Single core and (b) Multicore with ANoC protocol

5.5.2 Multi-Channel SCA

We measure 5×10^5 of both power dissipation and EM traces as two physical leakages channel concurrently during the AES-CCM encryption operations with our proposed AMP-MP in three cores ANoC and plot the SCA result (CPA and CEMA) as shown in Figure 5.20. It shows that, based on single channel using CPA (black) and multi-channel attacks based on both CPA and CEMA (red) concurrently [75], the secret key is still unrevealed with 5×10^5 measurements for

both power dissipation and EM signal. The correlation of the secret key is remained lower (< 0.2) compared with other key candidates at 5×10^5 measurements. For comparison, the experimental results on SCA show that the resistance against SCA has been increased by $> 12.82 \times$. The comparison result is determined based on the conventional AES implementation which is presented in Figure 5.19(a) and the multi-channel attacks in Figure 5.20. Eventually, based on our proposed AMP-MP in AES-CCM, the secret key is secured against multi-channel attacks and the adversary will not be able to reveal the secret key in $< 5 \times 10^5$ traces.

Figure 5.20: The plot of Single-channel (black) and Multi-channel result (red) based on CPA and CEMA which are unrevealed at 5×10^5 measurements

For completeness, we compare the performance of our proposed AMP-MP with the reported techniques on AES-CCM and other Authentication-Encryption with Associated Data (AEAD) algorithms. Table 5.5 tabulates a comparison of our proposed AMP-MP with conventional implementation and various reported techniques on AES-CCM. Based on multicore ANoC, running @100MHz for 5-core processors, our proposed AMP-MP features 8.32Gbps of throughput which is $2 \times$ to $70 \times$ faster than reported techniques during the authentication-

Table 5.5: COMPARISON OF OUR PROPOSED AMP-MP AES-CCM WITH
THE REPORTED AES-CCM TECHNIQUES

	Technique	Architecture (AES core(s))	Speed (MHz)	Throughput (Gbps)	Platform	SCA resistant	
						Single- Channel	Multi- Channel
AES control unit FPGA [66]	Three control modules	2	100.00	1.05 (8×)	Spartan 3 3S4000	Yes	N/A
Parallel two AES [67]	Reordering iterations	2	264.00	2.69 (3×)	CMOS SAED 90nm	N/A	N/A
Open System Inter AES [68]	Interconnection AES	2	152.42	1.95 (4×)	XC4VLX40	N/A	N/A
Unified data [69]	Redundancy Check	1	341.58	3.71 (2×)	XC7Z020	Yes	N/A
Ultra-low power AES [70]	8-bit AES core enc.	1	149.00	0.12 (70×)	ASIC 65nm CMOS	N/A	N/A
Conventional AES-CCM [71]	CCM and CBC	1	100.00	0.47 (17×)	XC7S75	Yes	N/A
Our Proposed AMP-MP	Mat. Mult GF(2 ⁸)	5	100.00	8.32 (1×)	Multicore ANoC 65nm	Yes	Yes

encryption. In addition to the security level of the secret key against SCA, both single and multi-channel attacks, are unbreakable at 5×10^5 traces measurements which have been indicated as lower correlation with processed data (<0.29) compared with other key candidates.

In addition to the comparisons with reported AES-CCM implementations, we further compare the performance of our proposed AMP-MP with AEAD algorithms. Three AEAD algorithms [79] have been reported as highly efficient algorithms, Deoxys, NORX and CLOC. Table 5.6 tabulates the performance of our proposed AMP-MP in AES-CCM and compare with three AEAD algorithms. The area is in terms of Kilo-Gate Equivalent (KGE) which indicates the active area used for the algorithm implementation. The efficiency parameter used in this comparison is the throughput/area, i.e., Gbps/KGE.

Table 5.6: THE PERFORMANCE OF OUR PROPOSED AMP-MP COMPARED WITH OTHER THREE
OPTIMIZED AEAD ALGORITHMS

Algorithms	Area (KGE)	Max. Freq. (MHz)	Throughput (Gbps)	Efficiency (Gbps/KGE)
Deoxys*	59.53(1.2×)	847	7.22 (1.2×)	0.12 (1.33×)
NORX*	70.13(1.3×)	757	83.11(0.1×)	1.18 (0.13×)
CLOC*	67.09(1.3×)	746	2.85 (2.9×)	0.04 (4.00×)
Our proposed AMP-MP [#]	51.31(1.0×)	100	8.32 (1.0×)	0.16 (1.00×)

*the optimized result is obtained by running it at maximum frequency and high-power dissipation using TSMC 65nm technology [79]

[#]implemented using Global Foundries 65nm CMOS technology

The NORX is an online cipher which computes the intermediate states and the authentication tag based on the secret key [80]. In Table 5.4, the NORX outperforms our proposed AMP-MP in asynchronous multicore in terms of throughput ($0.1\times$) and efficiency ($0.13\times$). However, the NORX is still vulnerable against SCA (i.e., CPA decryption attacks) [80], particularly when the target of attack is at the output of the authentication process (i.e., MAC) where the secret key is reused two times. Hence, the secret key of the NORX implementation is vulnerable during the authentication process at the PLI measurements. In addition, by considering a passive attack such as fault injection, the attack can be detected through the authentication with $\sim 5\times$ faster than the conventional authentication process.

5.6 Conclusions

In this chapter, an AMP-MP in AES-CCM [82] has been proposed to achieve a comprehensive high throughput and secure AES-CCM yet low power dissipation and latency overheads. The high throughput of 8.32Gbps is achieved by implementing the matrix multiplication in $GF(2^8)$ computation which transforms 16 plaintexts to 1 plaintext for authentication process. While the highly resistance of SCA ($>5\times 10^5$ traces) and low overheads are achieved concurrently by leveraging the multicore ANoC with our propose AMP-MP. The additional security feature, key adjusting technique, is implemented to further secure the AES-CCM against extrapolation key value based on the computational attacks, the similar patterns attacks on HW and HD during the key updates and MMA. With all these advantages, our proposed AMP-MP in AES-CCM is suitable for secured IoT applications.

CHAPTER 6

A CONTROLLABLE TIME SHUFFLING COUNTERMEASURE SCA

This chapter presents the proposed CTS technique as an advanced countermeasure technique against SCA. It also presents the Nano AES implementation which is based on serial mode S-Box operation and embodying the proposed CTS technique. The Nano AES is designed and fabricated based on 65nm CMOS process. The smaller form-factor Nano AES is targeted for highly secured Biomedical and IoT applications.

6.1 Introduction

In this chapter, a highly secured CTS is proposed to hide the PLI measurements against the SCA by randomly shift the time operations of the cryptographic algorithm implementations. The proposed CTS technique [83] is essentially to introduce at least one additional round of iterations (i.e., 11th round in AES-128 algorithm) defined as a redundant round iteration while the AES-128 round of iterations (i.e., 1st to 10th rounds in AES-128 algorithm) can be defined as genuine round of iterations. The terms of redundant and genuine rounds will be used thorough this chapter as to describe the proposed CTS countermeasure. The PLI measurement generated by the redundant round is similar to the PLI measurements (i.e., power dissipation or an EM signal) which generated by a genuine round (i.e., $\Delta\text{PLI} \leq 5\%$). The redundant round is performed using the different secret keys and different plaintexts to make the PLI measurements uncorrelated with the processed data in genuine round. Therefore, the presence of redundant round would significantly hinder the CPA and CEMA from revealing the secret key, hence serving as a highly secured countermeasure technique for hardware security. Our

proposed CTS countermeasure technique features $10^3\times$ more effective than the reported techniques due to its shuffling technique between redundant and genuine rounds. In addition, we further design a small form-factor and ultra-low power AES implementations, namely Nano AES, which embodies a proposed CTS technique for highly secured Biomedical and IoT applications.

The remainder of this chapter is organized as follows. The proposed CTS technique is presented in Section 6.2. Subsequently, the serial S-Box in Nano AES embodying the proposed CTS technique is described in Section 6.3. The experimental results and the comparison with the reported countermeasure techniques are presented in Section 6.4. Finally, conclusions are drawn in Section 6.5.

6.2 Proposed Controllable Time Shuffling

Figure 6.1 presents our proposed CTS countermeasure technique which embodies a control signal (Ctr_Sffl). The control signal is employed to prevent the output of the redundant round to be encrypted for the next rounds of iterations. In this technique, the redundant round has

Figure 6.1: Proposed Controllable Time Shuffling countermeasure

the same number of operations as in the genuine rounds, such as S-Box, Mix Column, Shift Row and Add Round Key and therefore it will generate a similar PLI measurements as generated by the genuine rounds. To prevent the adversary from extracting the secret key using the CPA or CEMA, the redundant round is performed by using different secret keys and different plaintext. Subsequently, the output of the redundant round will be blocked by *Ctr_Sffl* and the multiplexor (MUX_F) due to the wrong data generated from encrypting the different secret keys and plaintexts. Thus, the ciphertext will be generated at the output based on the genuine rounds. The signal which drives the *Ctr_Sffl* may generated independently from plaintext to make the shuffling rounds unpredictable by attackers.

The control signal can be employed to interchange the occurrences between the redundant and the genuine rounds and therefore the adversary will be difficult to synchronize the PLI measurement of the genuine rounds when revealing the secret key. In addition, since the PLI measurements between the redundant and the genuine rounds are similar with differences of $\leq 5\%$, the adversary will be more difficult to search and distinguish between redundant and genuine rounds in the PLI measurement. Therefore, the proposed CTS technique will be more secured against SCA. It is worthwhile to note that the number of the redundant rounds is exponentially related with the resistance level against SCA. For instance: by adding one redundant round, the adversary would require ~43 hours to search the genuine PLI measurement to reveal the secret key. Furthermore, by adding two redundant rounds, the adversary would require ~283 hours to successfully reveal the secret keys.

Figure 6.2 illustrates the implementation of the proposed CTS in four different scenarios for the occurrences of the genuine (G) round; a) without redundant round, b) genuine round at the beginning, c) middle and d) last of the clock cycles. The number of additional clock cycles

(a) depends on the required number of redundant (R) rounds. In this countermeasure technique, the higher number of R performed will increase the resistance level of the cryptographic implementation (i.e., AES-128) against the SCA, however, it will reduce the throughput of the encryption. Therefore, to maintain the throughput with highly secured crypto-device, at least one R is performed during the encryption.

Figure 6.2: Illustration of the CTS operation a) without redundant round, the genuine round is performed at the b) beginning, c) middle and d) last of clock cycles

6.3 Design Nano AES embodying a proposed CTS

In Section 6.2, the concept of the proposed CTS has been explained where the hardware implementation for each round within AES can be implemented differently depending on the applications. For instance, the main operation of the AES is the S-Box operation which can be implemented in serial mode to reduce the form-factor for medium speed and low power applications. Figure 6.3 shows two different implementations for the S-Box operation, parallel and serial S-Box. The parallel S-Box which is referring to the conventional implementation requires one clock cycle to compute 128-bit input plaintext in one round while the serial S-Box requires 16 clock cycles to compute 128-bit input plaintext, 8-bit plaintext for each clock

cycle. This serial approach is advantageous for small form-factor implementation as only one 8-bit S-Box circuit is required, hence reducing the form-factor by $16\times$. The serial S-Box implementation in AES algorithm is termed as Nano AES. Furthermore, the design Nano AES is able to reduce the power dissipation by $10\times$ by reducing the number of registers in the crypto-device, i.e., reducing the number of switching registers for each round computation.

Figure 6.3: The S-Box mode operations parallel and serial mode which require one and 16 clock cycles respectively

Figure 6.4 presents a serial architecture by serializing the S-Box operations where the inputs are *Sin* [127:0] and the corresponding outputs are *Sout* [127:0]. Instead of having 128-bit S-Box operations simultaneously within each round, the S-Box operations are serialized by performing multiple iterations of S-Box operations in one round of iterations. For instance, each iteration of S-Box operations can perform an 8-bit S-Box operation which will be iterated $16\times$, hence completing $128\text{-bit} = 8 \times 16$ or 16 bytes plaintext. The iterations of S-Box operations are controlled by an embedded Counter Controller which is synchronizing the inputs via input registers (FF) to the S-Box circuits and by synchronizing the outputs via output registers (FF) to the Shift Row operations of the AES algorithm. In the design Nano AES implementation, the output registers can be optional where *Sout* [127:0] can be indirectly feedback via combinational logic to the input registers to perform all round of iterations. By adopting the concept of the proposed CTS countermeasure technique, a redundant S-Box can be added during the iteration of S-Box operations (i.e., determine the S-Box) in each round

computation of Nano AES. In this technique, only one clock cycle required to perform redundant S-Box while 122 clock cycles required for Nano AES to encrypt 128-bit plaintext. In other words, the smaller overhead of 0.82% can increase the resistance level exponentially against the SCA.

Figure 6.4: The iteration of 16× for the 8-bit S-Box in the designed Nano AES.

6.4 Experimental Results

The experiment on the AES-128 implementation is performed based on the Sakura-X FPGA board [14] with frequency of 3MHz. The 16-byte plaintext is randomly generated to the board whereas the 16-byte secret key is fixed. The power dissipation and EM signal generated during the encryption process are measured in an oscilloscope with a sampling rate of 2.5GS/s as presented in Figure 6.5. The AES-128 algorithm requires 11 clock cycles to process one plaintext into ciphertext. The proposed CTS countermeasure is implemented at the last round of the AES-128 algorithm with 2 additional clock cycles. The last round operation is shuffled along 3 clock cycles (last round + 2 additional clock). The total number of clock cycles to process the plaintext into ciphertext is 13. The Probability of detecting the leakage (P_r) based on (6.1) below is $3.636 \times 10^{-3} \%$.

$$P_r = \frac{1}{S \cdot T(a+1)} \quad (6.1)$$

Where a : additional clock cycles;

S : rate of sampling points (in Samples/second);

T : time required to encrypt a plaintext into ciphertext (in second).

In Figure 6.5, the resistance level of against the SCA for power dissipation and EM signal are evaluated based on the CPA and CEMA computations respectively. Additional two peaks in the measurements (to be 13 peaks from original number of peak is 11) is undistinguishable to other peaks with $\sim 1\%$ differences and hence the adversary will be more difficult to identify the redundant operation. In this section, the performances of the proposed technique on CPA and CEMA are compared with the reported techniques [37], [50].

Figure 6.5: Measurement result of the proposed CTS for AES-128 algorithm

The comparison of the AES-128 performance without and with shuffling is tabulated in Table 6.1. It shows that the speed of the time shuffling AES-128 is decreased by 17.98% and the power dissipation (P) generated shuffling process is 9.95% power overhead. However, the

minimum resistance level ($R_{\min}$) against the SCA is $>10^6$ measurements which is $\sim 66.67\times$ and $\sim 18.52\times$ higher than N_Shuffle [47] and Rnd_Shuffle [60] respectively.

Table 6.1: THE COMPARISON OF THE PERFORMANCE SHUFFLING COUNTERMEASURE

	$P_r (\times 10^{-3} \%)$	$t (\mu s)$	$R_{\min}$	P (mW)	Peaks
Non-Shuffling [45]	120	3.67	400 (1 $\times$)	15.07	11
N_Shuffle [47]	151.4	3.67	15k (37.5 $\times$)	15.34	11
Rnd_Shuffle [60]	63.5	4.53	54k (135 $\times$)	20.12	11
Proposed CTS	3.636	4.33	1M (2,500 $\times$)	16.57	13

Furthermore, the designed Nano AES embodying a proposed CTS is implemented in FPGA board and the resistance level against SCA is evaluated based on the EM signal generated during the encryption. Figure 6.6 presents the EM signal of the designed Nano AES where it requires 122 clock cycles in 2.5×10^5 sampling points to complete one AES encryption. Multiple peaks of the EM signal are generated due to the serial architecture where each 8-bit S-Box computation will generate the EM signal. Therefore, each round of iteration of AES-128 will emit 10 peaks EM signal. The SCA resistant for both parallel and serial S-Box computation are evaluated based on 10^6 PLI measurement traces. The SCA resistant is evaluated by correlating the EM signal and the processed data of the crypto-device. Figure 6.7 presents the correlation coefficient vs number of traces of the secret key for both parallel and serial S-box computation which are embodying the proposed CTS countermeasure technique. It shows that the secret key is remained unbreakable at 10^6 traces and lower correlation coefficient compared with the other key candidates.

Figure 6.6: The EM measurement of the Nano AES embodying CTS technique

Figure 6.7: SCA evaluation for both Parallel and Serial S-Box implementations embodying the CTS technique in FPGA platform

The parallel S-Box and designed Nano AES embodying the CTS technique are further implemented in full-custom IC based on the 65nm CMOS process. The microphotograph of the IC implementation is portrayed in Figure 6.8 which occupies 1.37mm^2 ($1.055\text{mm} \times 1.304\text{mm}$). Based on the layout of the IC design, the number of gate counts in Nano AES is 1,745 where it is significantly reduced by 80.78% which is from 9,080 gate counts in conventional parallel S-Box.

Figure 6.8: Micrograph for both Parallel and Serial S-Box in Nano AES implementations embodying the CTS technique in ASIC

In addition, we further analyze the active attacks such as fault attack on our proposed technique as follows. The most critical point for the attacker to inject the fault is at the controller part where the pattern of the countermeasures is leaked out. However, since all the controllers (i.e., *Ctr_Sffl*, *En_In*, *En_Out* and *Ctr_In*) in our proposed controllable shuffling has integrated each other, if one of the controllers has been injected to flip the signal, the other controller will quickly adapt with the current condition to generate a true logic. Therefore, the result of the fault injection may not be observed at the output and the attack may not be successful.

6.5 Conclusions

In this chapter, a highly secured CTS countermeasure technique has been proposed with three main features. First, it embodies a control signal which can control the occurrences of the redundant and genuine rounds and hence shuffling the PLI measurements in time domain. Second, the PLI measurements of the redundant round is generated similarly (i.e., $\Delta\text{PLI} < 5\%$) to the genuine round hence make more difficult for the adversary to identify the PLI of the genuine round. Consequently, the secret key is highly secured against the SCA. Third, the shuffling technique in the proposed CTS is able to hide dual PLI measurements concurrently, power dissipation and EM signal with $>10^6$ traces. The experimental results are also presented in this chapter showed that the proposed CTS for countermeasure technique features $\sim 1\%$ different PLI measurement between redundant and genuine rounds. As results, correlated PLI is unidentified by the adversary and hence the correlation of the PLI and the processed data is significantly reduced. Subsequently, the designed Nano AES is also presented which embodies a proposed CTS and implemented in FPGA. When compared with the reported counterparts, the SCA resistant has been significantly increased by $2,500\times$ and the secret key of AES-128 algorithm is unrevealed at 10^6 traces. With the designed Nano AES embodying a proposed CTS, the number of gate-counts for highly secured AES-128 implementation is significantly reduced by 80.78%. Therefore, the designed Nano AES with proposed CTS is highly secured against SCA with smaller form-factor which is suitable for Biomedical and IoT applications.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK

7.1 Conclusions

This thesis has described the investigation and the design of the highly efficient SCA which was meant to evaluate efficiently the SCA resistant of the proposed countermeasure techniques in the crypto-device. Subsequently, the proposed countermeasure techniques have been implemented with emphases on highly secure against power and electromagnetic SCA with low overhead. The targeted application of the proposed countermeasure technique in this thesis is for low overhead crypto-devices such as secured Biomedical devices and IoT implementations.

A novel PRL technique for profiling SCA has been proposed to extract highly correlated PLI with the processed data of the crypto-device to enhance the efficiency of the SCA. In the proposed technique, the sampling points of the highly correlated PLI are identified and clustered using the k -NN variance clustering and discarded the unrelated PLI. Therefore, the computational power is significantly reduced. The experimental results showed that the proposed PRL is $2,000\times$ more efficient than the reported profiling techniques when evaluating the vulnerability of the countermeasure techniques against SCA. The computational complexity of the proposed PRL is linearly proportional to the number of PLI measurements, $O(n)$, which is highly effective for SCA.

A highly secured SsLC for countermeasure technique has been proposed by regularly shifting the time operation of the cryptographic algorithm. It embodies an FSM which regularly shift the timing operation of the AES-128 implementation and hence the PLI which encompasses a secret key is spread over in time domain. Consequently, the probability of revealing the secret key is reduced significantly. The regular state shifting technique in the proposed SsLC can effectively hide the PLI measurements, i.e., power dissipation and EM signals, concurrently. Based on the experimental results in the FPGA, with 10^6 power dissipation and EM measurements, the secret key remains unbreakable while the correlation coefficient is reduced by $3.5\times$. When compared with the reported techniques, the SCA resistant is significantly improved by $>18\times$ and $>25\times$ based on the CPA and CEMA attacks respectively.

An AMP-MP in an asynchronous multicore has been proposed in this thesis to achieve a comprehensive performance, high throughput and secure AES-CCM, yet low overheads of the power dissipation and the latency. The high throughput of 8.32Gbps has been achieved by implementing matrix multiplication in $GF(2^8)$ computation which transforms 16 plaintexts to 1 plaintext for authentication process. While the highly resistance of SCA ($>5\times 10^5$ traces) and low overheads were achieved concurrently by leveraging the multicore ANoC with our proposed technique. The additional security feature, key adjusting technique, was implemented to further secure the AES-CCM against future computational attacks and the leakage patterns of HW and HD during key updates and MMA. With all these advantageous, the proposed technique in the AES-CCM is suitable for secured IoT applications.

A highly secured Nano AES with CTS countermeasure technique has been proposed by serializing the operations of AES implementation to achieve a small form-factor and highly

secured crypto-device. The proposed technique targeted the low power and medium speed application such as Biomedical device and IoT applications. Based on the experimental results, the embodiment of the CTS countermeasure technique in the Nano AES has successfully reduced 80.78% of the gate counts. In addition, the secret key of AES-128 algorithm is unbreakable at 10^6 traces of power dissipations and EM measurements. When compared with reported counterparts, the SCA resistant has been significantly improved by $>18.51\times$ and $>25\times$ based on the CPA and the CEMA respectively. With high secured and smallest form-factor, the proposed Nano AES with CTS countermeasure technique is suitable for the secured Biomedical device and IoT applications.

7.2 Recommendations for Future Work

The followings are recommended for future work to further improve the resistance level of the countermeasure techniques against SCA.

7.2.1 Integration of the machine learning algorithms and SCA

In Chapter 3, the proposed Relevance Learning and k -NN as Machine Learning algorithms are implemented in the pre-processing SCA to speed up the SCA evaluation. The advanced Machine Learning can be further adopted such as Convolutional Neural-Network (CNN) to further increase the reliability and accuracy of the SCA. By learning various leakage models for different implementations (i.e., microprocessors and FPGA) and stored the learning variables online, the subsequent SCA can interpret the secret key based on the CNN algorithms. Therefore, the time duration to reveal the secret keys in the SCA can be reduced significantly.

7.2.2 Dynamic Frequency Scaling integrated with State-shift Local Clock

In Chapter 4, the proposed State-shift Local Clock is implemented based on the constant clock frequency. With advanced SCA, the adversary may interpret the operational frequency and synchronize the PLI measurements to compute the correlation coefficient. Therefore, by integrating the Dynamic Frequency Scaling with the State-shift Local Clock, the adversary will be more difficult to extract the highly correlated PLI measurements in time domain (i.e., based on the sampling point measurements) and increase the resistance level of the crypto-device against SCA.

7.2.3 Second order masking for NANO AES with Controllable Time Shuffling

In Chapter 6, the proposed Nano AES with CTS has been proven to be highly resistance against SCA. However, the second order SCA is somewhat compromised to the proposed Nano AES with CTS. To further enhance the shuffling technique, it is recommended to incorporate the second order masking in the register for each round computation of the AES-128. The second order masking can be performed by generating two random variables m_1 and m_2 at input registers for each round. The two random variables are subsequently unmasked at the output of the register to recover the correct intermediate data. The main challenges of this recommended design are the power dissipation and area overheads could be increased. The additional overheads could be due to both masking and unmasking processes in the register.

BIBLIOGRAPHY

- [1] A. M. Elmisery, S. Rho and D. Botvich, "A Fog Based Middleware for Automated Compliance With OECD Privacy Principles in Internet of Healthcare Things," in *IEEE Access*, vol. 4, pp. 8418-8441, 2016.
- [2] Weik M.H. (2000) encryption algorithm. In: Computer Science and Communications Dictionary. Springer, Boston, MA
- [3] F. Gandino, R. Ferrero and M. Rebaudengo, "A Key Distribution Scheme for Mobile Wireless Sensor Networks: q -s-Composite," in *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 34-47, Jan. 2017.
- [4] http://www.strategyr.com/MarketResearch/Hardware_Encryption_Market_Trends.asp
- [5] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *Advances in Cryptology — CRYPTO' 99*. vol. 1666, M. Wiener, Ed., ed: Springer Berlin Heidelberg, 1999, pp. 388-397.
- [6] W. Stallings, *Cryptography and network security: principles and practice*: Boston : Prentice Hall, c2011.5th ed., 2011.
- [7] K-S. Chong, K. Z. L. Ne, W-G. Ho, L. Nan, A. Akbar, B-H. Gwee and J. S. Chang, "Counteracting differential power analysis: Hiding encrypted data from circuit cells," *International Conference on Electron Devices and Solid-State Circuits (EDSSC)*, IEEE, 2015, pp. 297-300.
- [8] K. Tiri and I. Verbauwhede, "Charge recycling sense amplifier-based logic: securing low power security ICs against DPA [differential power analysis]," *Solid-State Circuits Conference, 2004. ESSCIRC 2004. Proceeding of the 30th European*, 2004, pp. 179-182.
- [9] S. Guilley, L. Sauvage, P. Hoogvorst, R. Pacalet, G. M. Bertoni and S. Chaudhuri, "Security Evaluation of WDDL and SecLib Countermeasures against Power Attacks," in *IEEE Transactions on Computers*, vol. 57, no. 11, pp. 1482-1497, Nov. 2008.
- [10] T. Lin, K. S. Chong, J. S. Chang and B. H. Gwee, "An Ultra-Low Power Asynchronous-Logic In-Situ Self-Adaptive V_{DD} System for Wireless Sensor Networks," in *IEEE Journal of Solid-State Circuits*, vol. 48, no. 2, pp. 573-586, Feb. 2013.
- [11] M. Bucci, L. Giancane, R. Luzzi and A. Trifiletti, "A Flip-Flop for the DPA Resistant Three-Phase Dual-Rail Pre-Charge Logic Family," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 11, pp. 2128-2132, Nov. 2012.
- [12] N. E. C. Akkaya, B. Erbagci, R. Carley and K. Mai, "A DPA-resistant self-timed three-phase dual-rail pre-charge logic family," *2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, Washington, DC, 2015, pp. 112-117.
- [13] E. Menendez and K. Mai, "Extended abstract: A high-performance, low-overhead, power-analysis-resistant, single-rail logic style," *2008 IEEE International Workshop on Hardware-Oriented Security and Trust*, Anaheim, CA, 2008, pp. 33-36.

- [14] Y. Hori, T. Katashita, A. Sasaki, and A. Satoh, "SASEBO-GIII: A hardware security evaluation board equipped with a 28-nm FPGA," in *1st Global Conference on Consumer Electronics (GCCE)*, IEEE, 2012, pp. 657-660.
- [15] C. Arnaud and P.-A. Fouque, "Timing Attack against Protected RSA-CRT Implementation Used in PolarSSL," in *Topics in Cryptology – CT-RSA 2013*. vol. 7779, E. Dawson, Ed., ed: Springer Berlin Heidelberg, 2013, pp. 18-33.
- [16] M. Nassar, Y. Souissi, S. Guilley, and J.-L. Danger, "'Rank Correction': A New Side-Channel Approach for Secret Key Recovery," in *Security Aspects in Information Technology*. vol. 7011, M. Joye, D. Mukhopadhyay, and M. Tunstall, Eds., ed: Springer Berlin Heidelberg, 2011, pp. 128-143.
- [17] O. X. Standaert, E. Peeters, G. Rouvroy, and J. J. Quisquater, "An Overview of Power Analysis Attacks Against Field Programmable Gate Arrays," *Proceedings of the IEEE*, vol. 94, pp. 383-394, 2006.
- [18] Z. Martinasek, V. Clupek, and T. Krisztina, "General scheme of differential power analysis," *36th International Conference on Telecommunications and Signal Processing (TSP)*, IEEE, 2013, pp. 358-362.
- [19] S. Mangard, "Hardware Countermeasures against DPA – A Statistical Analysis of Their Effectiveness," in *Topics in Cryptology – CT-RSA 2004*. vol. 2964, T. Okamoto, Ed., ed: Springer Berlin Heidelberg, 2004, pp. 222-235.
- [20] A. Thillard, E. Prouff, and T. Roche, "Success through Confidence: Evaluating the Effectiveness of a Side-Channel Attack," in *Cryptographic Hardware and Embedded Systems - CHES 2013*. vol. 8086, G. Bertoni and J.-S. Coron, Eds., ed: Springer Berlin Heidelberg, 2013, pp. 21-36.
- [21] W. Jun, S. Yiyu, and C. Minsu, "Measurement and Evaluation of Power Analysis Attacks on Asynchronous S-Box," *IEEE Transactions on Instrumentation and Measurement*, vol. 61, pp. 2765-2775, 2012.
- [22] D. D. Hwang, K. Tiri, A. Hodjat and B.-C. Lai, "AES-Based Security Coprocessor IC in 0.18- μ m CMOS With Resistance to Differential Power Analysis Side-Channel Attacks," *IEEE JSSC*, vol. 41, no. 4, pp. 781-792, Apr 2006.
- [23] F. Gurkaynak, S. Oetiker, H. Kaeslin, N. Felber, and W. Fichtner, "Improving DPA security by using globally-asynchronous locally-synchronous systems," in *Solid-State Circuits Conference, ESSCIRC 2005*. Proceedings of the 31st European, pp. 407-410, 2005.
- [24] K. Woo Hyun, H. Soohee, and L. Jang Gyu, "An Optimal FIR Filter With Fading Memory," *Signal Processing Letters*, IEEE, vol. 18, pp. 327-330, 2011.
- [25] S. Weiwei, F. Xingyuan, and X. Zhipeng, "A Secure Reconfigurable Crypto IC With Countermeasures Against SPA, DPA, and EMA," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, pp. 1201-1205, 2015.
- [26] K.-L. Chang, "Synchronous-logic and asynchronous-logic 8051 microcontroller cores for realizing the internet of things: a comparative study on dynamic voltage scaling and variation effects," *IEEE JETCAS*, v3, pp. 23–34, Mar. 2013.

- [27] R. Gomez Cid-Fuentes, A. Cabellos-Aparicio and E. Alarcon, "Area Model and Dimensioning Guidelines of Multisource Energy Harvesting for Nano–Micro Interface," in *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 18-26, Feb. 2016.
- [28] S. Jens and F. Steve, "Principle of Asynchronous Circuit Design - A System Perspective," in *Kluwer Publisher*, United Kingdom, Sep 2001.
- [29] W. G. Ho, K. S. Chong, N. K. Z. Lwin, B. H. Gwee and J. S. Chang, "High robustness energy- and area-efficient dynamic-voltage-scaling 4-phase 4-rail asynchronous-logic Network-on-Chip (ANoC)," *Circuits and Systems (ISCAS), 2015 IEEE International Symposium on*, Lisbon, 2015, pp. 1913-1916.
- [30] J. Chang, B.-H. Gwee & K.-S. Chong, "Digital cell," *US Patent Application Publication*, US 2013/0342253 A1, Dec. 2013.
- [31] H. Zakaria & L. Fesquet, "Designing a process variability robust energy-efficient control for complex SOCs," *IEEE JETCAS*, v1, n2, pp. 160–172, Jun 2011.
- [32] Y. Thonnart, P. Vivet, & F. Clermidy, "A fully-asynchronous low-power framework for GALS NoC integration," *DATE*, pp. 33-38, Mar. 2010.
- [33] T. Bjerregaard & J. Sparso, "Implementation of guaranteed services in the MANGO clock less network-on-chip," *IEE Proc. Comp. and Dig. Tech.*, v153, n 4, pp. 217-229, Jul. 2006.
- [34] J. Pontes, R. Soares, E. Carvalho & N. Calazans, "SCAFFI: an intrachip FPGA asynchronous interface based on hard macros," *ICCD*, pp. 541-546, Oct. 2007.
- [35] A. Sheibanyrad, A. Greiner & I. Miro-Panades, "Multi-synchronous and fully asynchronous NoCs for GALS architectures," *IEEE Design & Test of Comp.*, v25, n6, pp. 572-580, Nov. 2008.
- [36] S. Mangard, E. Oswald, and T. Popp, *Power Analysis attacks: Revealing the secrets of smart cards*. 2007.
- [37] N. Bruneau, S. Guilley, A. Heuser, and O. Rioul, "Taylor Expansion of Maximum Likelihood Attacks for Masked and Shuffled Implementations," in *ASIACRYPT 2016*, 2016, vol. 10031, pp. 573–601.
- [38] C. Whitnall and E. Oswald, "Robust Profiling for DPA-Style Attacks," in *Cryptographic hardware and embedded systems – CHES 2015: 17th international workshop Saint-Malo, France, september 13–16, 2015 proceedings*, 2015, vol. 9293, pp. 3–21.
- [39] N. Bruneau, S. Guilley, A. Heuser, D. Marion, and O. Rioul, "Less is More: Dimensionality Reduction from A Theoretical Perspective," in *Cryptographic hardware and embedded systems – CHES 2015: 17th international workshop Saint-Malo, France, september 13–16, 2015 proceedings*, 2015, vol. 9293, pp. 22–41.
- [40] S. M. Del Pozo and F.-X. Standaert, "Blind Source Separation from Single Measurements Using Singular Spectrum Analysis," in *Cryptographic hardware and embedded systems – CHES 2015: 17th international workshop Saint-Malo, France, september 13–16, 2015 proceedings*, 2015, vol. 9293, pp. 42–59.
- [41] C. Clavier, J. S. Coron, and N. Dabbous, "Differential Power Analysis in the Presence of Hardware

- Countermeasures,” *Proceedings of CHES 2000*, pp. 252–263, 2001.
- [42] B. Gierlichs, K. Lemke-rust, C. Paar, K. U. Leuven, and E. Cosic, “Templates vs. Stochastic Methods A Performance Analysis for Side Channel Cryptanalysis,” in *Cryptographic hardware and embedded systems – CHES 2006*, 2006, pp. 15–29.
 - [43] A. Heuser, O. Rioul, and S. Guilley, “Good Is Not Good Enough: Deriving Optimal Distinguishers from Communication Theory,” in *Cryptographic hardware and embedded systems – CHES 2014*, 2014, pp. 55–74.
 - [44] T. Güneysu and H. Handschuh, “SoC It to EM: ElectroMagnetic Side-Channel Attacks on a Complex System-on-Chip,” in *Cryptographic hardware and embedded systems – CHES 2015: 17th international workshop Saint-Malo, France, september 13–16, 2015 proceedings*, 2015, vol. 9293, pp. 620–640.
 - [45] A. A. Pammu, K.-S. Chong, and B.-H. Gwee, “Secured Low Power Overhead Compensator Look-Up-Table (LUT) Substitution Box (S-Box) Architecture,” in *IEEE International Conference on Networking, Architecture and Storage (NAS), Aug. 2016*, 2016, pp. 1–7.
 - [46] M. Rivain, E. Prouff, and J. Doget, “Higher-order masking and shuffling for software implementations of block ciphers,” in *Cryptographic hardware and embedded systems – CHES 2009: 11th International Workshop Lausanne, Switzerland, September 6-9, 2009 Proceedings*, 2009, vol. 5747 LNCS, pp. 171–188.
 - [47] C. Herbst, E. Oswald, and S. Mangard, “An AES smart card implementation resistant to power analysis attacks,” in *International Conference on Applied Cryptography and Network Security ACNS 2006: Applied Cryptography and Network Security*, 2006, vol. 3989, pp. 239–252.
 - [48] C. O’Flynn and Z. Chen, “Synchronous sampling and clock recovery of internal oscillators for side channel analysis and fault injection,” *Journal of Cryptographic Engineering*, vol. 5, no. 1, pp. 53–69, 2015.
 - [49] A. Sato and K. Yamada, “Generalized Learning Vector Quantization,” in *Advances in Neural Network Information Processing Systems*, 1996, pp. 423–429.
 - [50] P. Schneider, K. Bunte, H. Stiekema, B. Hammer, T. Villmann, and Bie, “Regularization in matrix learning,” *IEEE Transactions on Neural Networks*, vol. 21, no. 5, pp. 1–47, 2010.
 - [51] A. A. Pammu, K.-S. Chong, K. Z. L. Ne, and B.-H. Gwee, “High Secured Low Power Multiplexer-LUT Based AES S-Box Implementation,” in *2016 International Conference on Information Systems Engineering (ICISE)*, 2016, pp. 3–7.
 - [52] P. Hodgers, N. Hanley, and M. O’Neill, “Pre-processing power traces with a phase-sensitive detector,” in *Proceedings of the 2013 IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2013*, 2013, pp. 131–136.
 - [53] W. Yang, Y. Cao, Y. Zhou, H. Zhang, and Q. Zhang, “Distance based leakage alignment for side channel attacks,” *IEEE Signal Processing Letters*, vol. 23, no. 4, pp. 419–423, 2016.
 - [54] C. Tokunaga and D. Blaauw, “Securing encryption systems with a switched capacitor current equalizer,” *IEEE Journal of Solid-State Circuits*, vol. 45, no. 1, pp. 23–31, 2010.

- [55] A. A. Pammu, K.-S. Chong, and B.-H. Gwee, "Highly Secured State-Shift Local Clock Circuit to Countermeasure Against Side Channel Attack," *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD, USA, pp. 1–4, 2017.
- [56] S. Ghosh and I. Verbauwhede, "BLAKE-512-based 128-bit CCA2 secure timing attack resistant McEliece cryptoprocessor," *IEEE Transactions on Computers*, vol. 63, no. 5, pp. 1124–1133, 2014.
- [57] L. Zhuang, F. Zhou, and J. D. Tygar, "Keyboard acoustic emanations revisited," in *Proceedings of the 12th ACM conference on Computer and communications security - CCS '05*, 2005, vol. V, no. November, p. 373.
- [58] P. C. Liu, H. C. Chang, and C. Y. Lee, "A low overhead DPA countermeasure circuit based on ring oscillators," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 57, no. 7, pp. 546–550, 2010.
- [59] W. Shan, X. Fu, and Z. Xu, "A Secure Reconfigurable Crypto IC With Countermeasures Against SPA , DPA , and EMA," *IEEE Transaction on Computer Aided Design of Integrated Circuits and Systems*, vol. 34, no. 7, pp. 1201–1205, 2015.
- [60] M. Rivain, E. Prouff, and J. Doget, "Higher-order masking and shuffling for software implementations of block ciphers," *Lect. Notes Comput. Sci. Artif. Intell. Bioinforma.*, vol. 5747 LNCS, pp. 171–188, 2009.
- [61] M. J. Dworkin, "Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication," *NIST Special Publication 800-38B*, 2007.
- [62] W. Prodanov, M. Valle, and R. Buzas, "A Controller Area Network Bus Transceiver Behavioral Model for Network Design and Simulation," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 9, pp. 3762–3771, 2009.
- [63] C. C. Wang, C. L. Chen, G. N. Sung, C. L. Wang, and C. Y. Juan, "A FlexRay transceiver design with bus guardian for in-car networking systems compliant with FlexRay standard," *Journal of Signal Processing Systems*, vol. 74, no. 2, pp. 221–233, 2014.
- [64] M. Conti, N. Dragoni, and V. Lesyk, "A Survey of Man in the Middle Attacks," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 3, pp. 2027–2051, 2016.
- [65] S. K. Mathew *et al.*, "340 mV–1.1 V, 289 Gbps/W, 2090-Gate NanoAES Hardware Accelerator With Area-Optimized Encrypt/Decrypt GF(24)2 Polynomials in 22 nm Tri-Gate CMOS," *IEEE Journal of Solid-State Circuits*, vol. 50, no. 4, pp. 1048–1058, 2015.
- [66] E. López-Trejo, F. Rodríguez-Henríquez, and A. Díaz-Pérez, "An FPGA Implementation of CCM Mode Using AES," in *Information Security and Cryptology-ICISC 2005.*, 2006, pp. 322–334.
- [67] K. Nguyen, L. Lanante, Y. Nagao, M. Kurosaki, and H. Ochi, "Implementation of 2.6 Gbps super-high speed AES-CCM security protocol for IEEE 802.11i," in *13th International Symposium on Communications and Information Technologies: Communication and Information Technology for New Life Style Beyond the Cloud, ISCIT 2013*, 2013, pp. 669–673.
- [68] I. Algreto-Badillo, C. Feregrino-Urbe, R. Cumplido, and M. Morales-Sandoval, "FPGA Implementation and Performance Evaluation of AES-CCM Cores for Wireless Networks," in *2008 International Conference on Reconfigurable Computing and FPGAs*, 2008, pp. 421–426.

- [69] Y. Wang, J. An, and Y. Ha, "Unified Data Authenticated Encryption for Vehicular Communication," in *2016 IEEE International Midwest Symposium on Circuits and System (MWSCAS)*, Abu Dhabi, UAE, 2016, no. October, pp. 16–19.
- [70] Van-Phuc Hoang, Thi-Thanh-Dung Phan, Van-Lan Dao, and Cong-Kha Pham, "A compact, ultra-low power AES-CCM IP core for wireless body area networks," in *2016 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, 2016, pp. 1–4.
- [71] J. H. Yoo, "Fast software implementation of AES-CCM on multiprocessors," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011, vol. 7017 LNCS, no. PART 2, pp. 300–311.
- [72] K. Atighehchi and R. Rolland, "Optimization of Tree Modes for Parallel Hash Functions: A Case Study," *IEEE Transactions on Computers*, vol. 66, no. 9, pp. 1585–1598, 2017.
- [73] S. K. Mathew *et al.*, "µrNG: A 300-950 mV, 323 Gbps/W All-Digital Full-Entropy True Random Number Generator in 14 nm FinFET CMOS," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 7, pp. 1695–1704, 2016.
- [74] M. Taha and P. Schaumont, "Key updating for leakage resiliency with application to AES modes of operation," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 3, pp. 519–528, 2015.
- [75] W. Yang, Y. Zhou, Y. Cao, H. Zhang, Q. Zhang, and H. Wang, "Multi-Channel Fusion Attacks," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1757–1771, 2017.
- [76] I. 802. 1. W. Group, "IEEE Standard for High Data Rate Wireless Multi-Media Networks," *IEEE Std 802.15.3-2016 (Revision of IEEE Std 802.15.3-2003)*, vol. 2016, pp. 1–510, 2016.
- [77] K. S. Chong, W. G. Ho, T. Lin, B. H. Gwee, and J. S. Chang, "Sense Amplifier Half-Buffer (SAHB) A Low-Power High-Performance Asynchronous Logic QDI Cell Template," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 2, pp. 402–415, 2017.
- [78] "Leak Me If You Can: Does TVLA Reveal Success Rate?", : [Online]. Available: <https://eprint.iacr.org/2016/1152.pdf>
- [79] Kumar, S., Yahya, J. H., Khairallah, M., Elmohr, M. A., Chattopadhyay, A., "A Comprehensive Performance Analysis of Hardware Implementations of CAESAR Candidates", 2018. [Online]. Available: <https://eprint.iacr.org/2017/1261.pdf>
- [80] Vaudenay, S., Vizàr, D., "Under Pressure: Security of Caesar Candidates Beyond Their Guarantees", 2018. [Online]. Available: <https://eprint.iacr.org/2017/1147.pdf>
- [81] A. A. Pammu, K. S. Chong, Yi Wang and B. H. Gwee, "A Highly Efficient Side Channel Attack with Profiling through Relevance Learning on Physical Leakage Information," *IEEE Transaction. on Dependable Secure Computing*, 2018, accepted for publication, July 2018.
- [82] A. A. Pammu, W. G. Ho, K. Z. L. Ne, K. S. Chong and B. H. Gwee, "A High Throughput and Secure Authentication-Encryption AES-CCM Algorithm on Asynchronous Multicore Processor," *IEEE Transactions on Information Forensics and Security*, 2018, accepted with mandatory minor revision, July 2018.

- [83] K.S. Chong, B.H. Gwee and A. A. Pammu, “Hardware Security to Countermeasure Side-Channel Attacks,” PCT Patent Applications and Singapore Patent Application No. 10201702226R, 19 Mar 2018.