This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2018.2869344, IEEE Transactions on Information Forensics and Security

A High Throughput and Secure Authentication-Encryption AES-CCM Algorithm on Asynchronous Multicore Processor

Ali Akbar Pammu, *Student Member, IEEE*, Weng-Geng Ho*, Member, IEEE*, Ne Kyaw Zwa Lwin,
Kwen-Siong Chong, *Senior Member, IEEE* and Bah-Hwee Gwee, *Senior Member, IEEE*

***Abstract*—We propose 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). There are four main features in our proposed AMP-MP. First, we employ the matrix multiplication in GF(2^8) computation to transform the 16 plaintexts into 1 plaintext, hence improving the authentication speed by 32× collectively at the transmitter and receiver. Second, we reschedule the operations of 3 AES encryptions in 3 different cores such that their physical leakages are compensated and equalized, thus reducing the correlation of physical leakage with the processed data by >3×. Third, the intermediate values of AES-CCM are propagated asynchronously between different cores to randomize the physical leakages with the processed data, therefore further enhance the security of AES-CCM against the SCA by another 3×. Fourth, we propose a key adjusting technique based on S-Box byte-key transformation to protect the key against pattern-based attack. 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 is 13.54Gbps while the throughput for both authentication and encryption collectively is 8.32Gbps, which are 17× and 70× faster than the reported counterparty respectively. Based on power dissipation and EM SCA on our proposed AMP-MP, the secret key is unrevealed at 5×10^5 traces, which is ~17× more secured than the standard ASIC AES-CCM implementation.**

***Index Terms* — Authentication, Encryption, AES-CCM, Multicore, Asynchronous Circuit, key adjusting technique**

I. 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 [1]. 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 depicted in Fig.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 Advanced Encryption Standard (AES) [1], 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 depicted in Fig. 2(a). In Fig. 2(a), the AES is used for encryption on the Counter with Chaining Mode (CCM) algorithm, abbreviated as AES-CCM [2]. 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) [3] and FlexRay [4], to secure the data communications against various attacks, such as Man-in-Middle-Attack (MMA) [5] and Side Channel Attack (SCA) [6]. 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 < 2×10^3) [7] 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 [8], memoryless AES-CCM implementation [9], single-core reconfigurable AES [10], Unified Data Authentication Encryption [11] and Ultra-Low Power AES-CCM IP core [12]. The maximum throughput can be achieved by reported techniques [8]-[12] is 3.71Gbps where the need for future communication technologies will be much higher (i.e. >3×). In addition, due to the placement and routing optimization in an FPGA, the physical leakage such as the power dissipation during the encryption process can be vulnerable to SCA [13] where the secret key is breakable at 400 traces. In the reported software implementations of AES-CCM [14], the iterative round computation of AES requires >10^3 clock cycles for each round [6], which is much higher than FPGA with requires only one clock cycle for each round computation of AES algorithm.

In this paper, we propose an Authentication based Matrix-transformation cum Parallel-encryption implemented in

The paper was submitted for review in February 2018. This work was supported in part by the Agency for Science, Technology and Research, Singapore under SERC 2013 Public Sector Research Funding, Grant No: SERC1321202098.

Ali Akbar Pammu*, Weng-Geng Ho, Ne Kyaw Zwa Lwin, Kwen-Siong Chong and Bah-Hwee Gwee are with Virtus, IC Design of Excellent Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798.
(E-mail: *ali1@ntu.edu.sg)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2018.2869344, IEEE Transactions on Information Forensics and Security

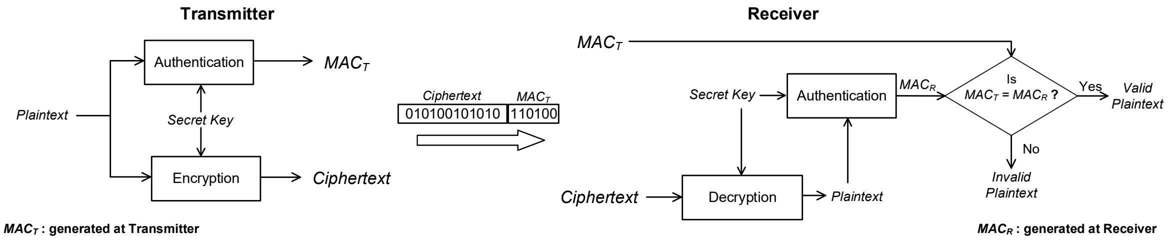
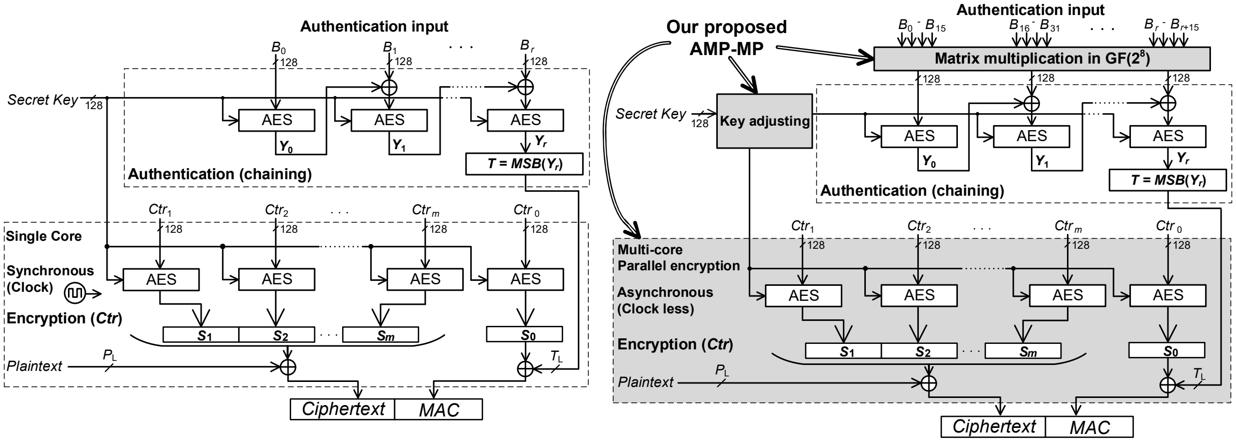


Fig. 1: The Message Authentication Code (MAC) is generated at the Transmitter and verified at the Receiver



(a)

(b)

Fig. 2: AES-CCM Architecture based on (a) Conventional (b) Our proposed AMP-MP

Multicore Processor (AMP-MP) to achieve comprehensive performance, high throughput and secure AES-CCM, yet low power dissipation overhead as depicted in Fig. 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 [2] as to break the dependency between the authentication input and the messages. In Fig. 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 Fig. 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 [15], aims to reduce the length of the authentication input. Consequently, the speed of the authentication process can be increased by ≥32× 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 physical leakages, i.e. power dissipation and Electromagnetic (EM) are compensated and equalized, thus reducing the correlation of the physical leakage with the processed data by ≥3×. 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×. Furthermore, the power dissipation and EM in asynchronous circuits can be significantly reduced by >2× through reducing the spurious switching due to the clock-less circuit protocol. Fourth, the leakage characteristics of the secret key based on Hamming Weight (HW) and Hamming Distance (HD) are observable in the key management (an IPsec function), due to the limitation-randomization of True-RNG (TRNG) [16] on key updates [17]. 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×. The power dissipation of authentication process based on our proposed AMP-MP is 307μW, while overall authentication and encryption process dissipates 311μW to perform 256 plaintexts, ~2× lower than a reported counterpart [11]. Based on SCA evaluation, single (power) and multi-channel (both power and EM) attacks [18], 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 [1] with 2^{16} sets of plaintext. The experimental results shows 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 paper is organized as follows. Review of the AES-CCM is explained in Section II. The proposed AMP-MP and its implementation in asynchronous-multicore processors are elaborated in Section III. The experimental results and comparisons of the performance with various reported of AES-CCM are presented in Section IV. Evaluation of our proposed AMP-MP based on SCA is presented in Section V. Finally, the paper is concluded in Section VI.

II. 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 [19] 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, …, B_r) for authentication process as depicted in Fig. 3. The authentication output (T) is generated with the bit-length (T_L) where the bit range of the Most Significant Byte (MSB) is from 32-to-64 bits [2], $32 < T_L < 64$.

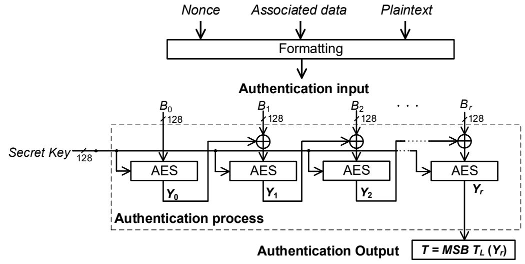


Fig. 3: Authentication process

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) [1], chaining mode. In the operation as depicted in Fig. 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 [6]) by correlating the ciphertext and measured EM emanation during the authentication.

In the encryption process as depicted in Fig. 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 the forward AES encryption [1] to generate MAC_T and MAC_R, hence reducing the area overhead.

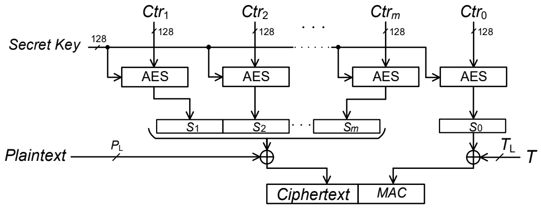


Fig. 4: Counter mode-based encryption to generate ciphertext and MAC

III. Proposed Authentication based Matrix-transformation cum Parallel-encryption in Multicore Processor (AMP-MP) on AES-CCM

The four main features of our proposed AMP-MP on AES-CCM are elaborated as follows.

A. *Proposed matrix multiplication in GF(2^8)*

To achieve high throughput of the authentication-encryption in AES-CCM, Fig. 5 depicts the schematic flow of the matrix multiplication on reducing the inputs for authentication from 256-block to 16-block of inputs. The input: raw-data, in this operation; 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 [1], 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 [15]). The fundamental difference of our proposed matric 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.

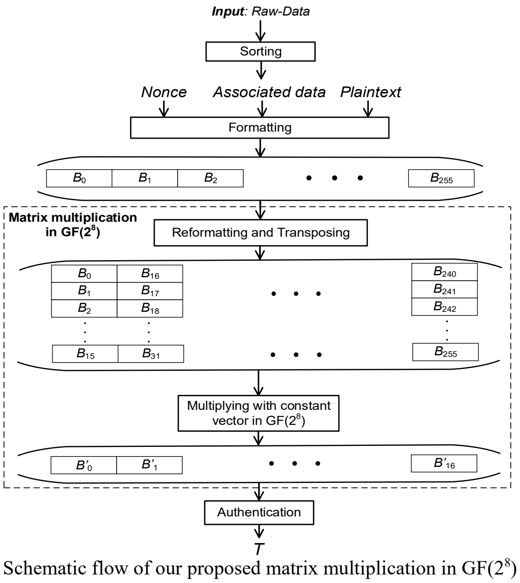


Fig. 5: Schematic flow of our proposed matrix multiplication in GF(2^8)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2018.2869344, IEEE Transactions on Information Forensics and Security

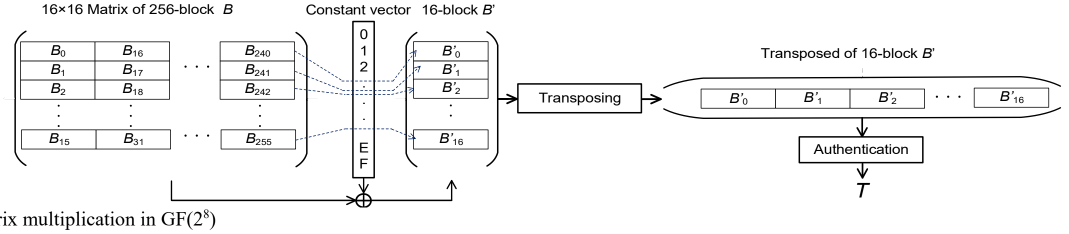


Fig. 6: Matrix multiplication in GF(2^8)

The detail operation of matrix multiplication is explained as follows. Considering the polynomial in 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 (1).

$$x \times f(x) = (a_7x^8 + a_6x^7 + a_5x^6 + a_4x^5 + a_3x^4 + a_2x^3 + a_1x^2 + a_0x) \bmod m(x) \qquad (1)$$

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

$$m(x) = x^8 + x^4 + x^3 + x + 1 \qquad (2)$$

The multiplication of a_7x^8 by $m(x)$ is shown in (3) as follows:

$$\begin{aligned} a_7x^8 \bmod m(x) &= a_7\,((x^8 + x^4 + x^3 + x + 1) - (x^8)) \\ &= a_7\,(x^4 + x^3 + x + 1) \qquad (3) \end{aligned}$$

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 (4) as follows:

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

As a summary, the overall bitwise operation for each byte in matrix multiplication can be expressed in (5) as follows:

$$x \times f(x) = \begin{cases} (a_6a_5a_4a_3a_2a_1a_0 0) & ; a_7 = 0 \\ (a_6a_5a_4a_3a_2a_1a_0 0) \oplus (00011011) & ; a_7 = 1 \end{cases} \qquad (5)$$

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 GF(2^4) to generate the constant vector elements where a generator for 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 GF(2^4) computations are tabulated in Table I as follows:

TABLE I: A GENERATOR OF GF(2^4) 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 (6) as follows:

$$b_0^{\prime 1} = ((b_0^1 \cdot 0) + (b_0^2 \cdot 1) + \cdots + (b_0^{16} \cdot F)) \cdot \mathrm{mod}(m(x)) \qquad (6)$$

Where $b_0^{\prime 1}$ = 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} ⋯ B_{240} (256-byte) with constant vector and B'_1, 16-byte, is generated base on matrix multiplication of B_1, B_{17} ⋯ B_{241} (256-byte) and the constant vector. The reduction bytes 256-to-16 is inherently first preimage resistant [1] which the input is unpredictable based on the output and values of the constant vector. The flow of matrix multiplication over GF(2^8) for 256 of *B* (e.g. plaintext) to generate 16 of *B'* for authentication is depicted in Fig. 6. The matrix multiplication is implemented in both transmitter and receiver to obtain the valid authentication tag. With our proposed matrix multiplication in GF(2^8), the authentication process, for one-time communication from the transmitter to receiver, can be significantly enhanced by >32× 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! ≈ 2.09×10^{13} possibilities, thus securing against preimage attacks.

Fig. 7 depicts 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 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 throughput of authentication process. The input of 128-bit *B* is multiplied with 64-bit randomized constant vector by referring to (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.

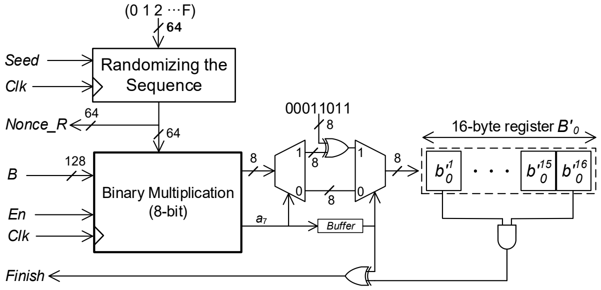


Fig. 7: Circuit implementation of matrix multiplication

B. Proposed Parallel AES with Rescheduling the operations

Fig. 8 depicts the circuit implementation of the AES-CCM authentication-encryption in 9-core synchronous-logic 8051 microcontroller [20] with Asynchronous Network on Chip (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 depicted in Fig. 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.

The main objective of having 3 cores for parallel-pipeline encryption is to further achieve >3× higher 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 [21]. In this analysis, two operations are defined: the vector and the scalar operations. The vector operation involves data transferred 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 (7).

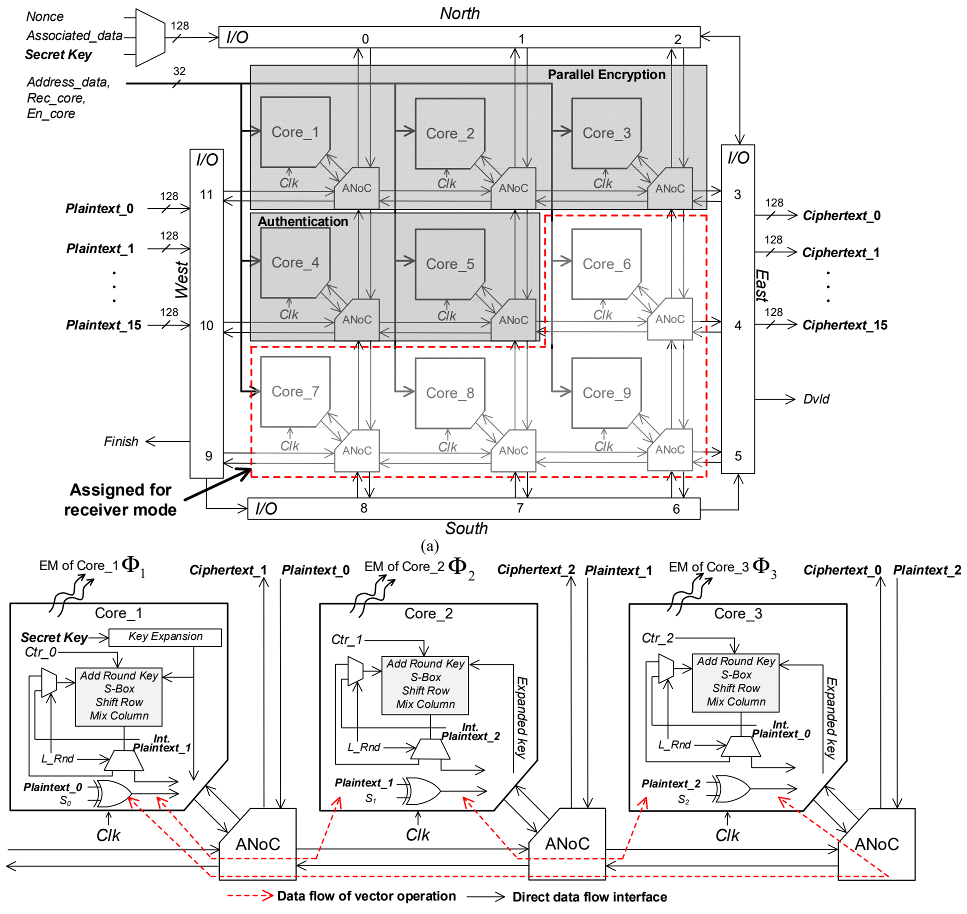
$$\frac{1}{b_{tot}}=\frac{F}{b_s}+\frac{1-F}{nb_s}=\frac{nF-F+1}{nb_s}=\frac{1+(n-1)F}{nb_s};b_{tot}=\frac{nb_s}{1+(n-1)F} \quad (7)$$

The analysis of the 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 the core area (A) as expressed in (8), where *μ*, *l* and *r* are constants permeability of the conductor, length and radius of the measured point respectively.

$$\Phi_x=\overline{B_x}\cdot A=\frac{\mu\cdot I_x\cdot l\cdot A}{2\pi r^2} \quad (8)$$

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 emanation with processed data in the same core encryption (e.g. Core_*x*) as expressed in (9).

$$\sum\Phi_{x,total}=\Phi_{x,first}+\Phi_{x,middle}+\Phi_{x,last}\xrightarrow{uncorrelated}processed\_data \quad (9)$$



(a)

(b)

Fig. 8: Design of ANoC based 9-Core processor (a) Overall architecture of multicore and (b) Parallel and rescheduling three AES within three cores

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 Core_1 and the result is propagated to Core_2 and Core_3. Each core encrypts the *Ctr* with secret key up to 9th round and transmit the intermediate result (i.e. *int. plaintext_0*) to the next core for the last round operation. For instance: Core_1 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.

C. Proposed a 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. Comparing with the synchronous counterparts, the physical leakages of the asynchronous circuits are able 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 [18] and lower EM emanation compared with synchronous circuits. In this paper, we propose to implement a novel asynchronous architecture of Sense Amplifier Half Buffer (SAHB) [21] in ANoC protocol to achieve fastest data flow between cores and yet high resistance of AES-CCM against SCA (i.e. lower correlation coefficient).

Fig. 9 depicts 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 [20]. 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.

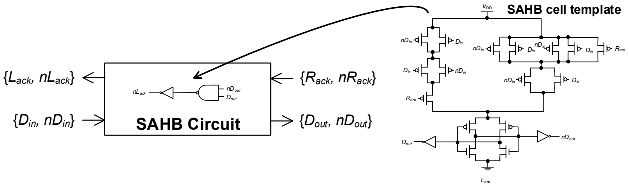


Fig. 9: Main channel interface (*L* and *R*) of SAHB circuit and cell template

D. Proposed key adjusting technique

The key update is one of the methods to countermeasure the SCA [17] by changing the secret key periodically based on timing and size of the messages; i.e. every 65 minutes or 500 megabytes plaintexts. The disadvantage of this method is compromising the throughput caused by additional time for both key exchange mechanism and public key management system to update the secret key of AES for both transmitter and receiver. 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 further protect of AES-CCM implementation against the patterns leakage, 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 to make the subsequent key unpredictable. Fig. 10 depicts circuit implementation of the proposed key adjusting technique.

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× until the pattern is fully randomized (in *Data_out*).

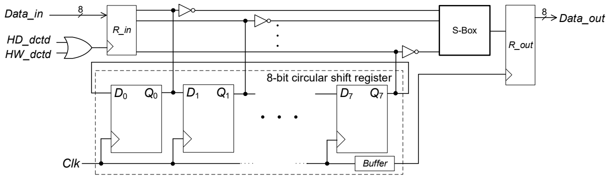
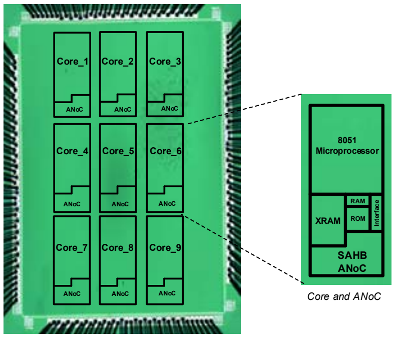


Fig. 10: Circuit schematic of key adjusting technique for each 8-bit *Data_in*

The key adjusting technique precedes the key expansion. 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 method is using asymmetric cryptographic algorithms (i.e. RSA, Diffie-Hellman or ECC) [1]. The validation of authentication key is performed with < 0.3% latency overhead.

IV. Experimental Results in Asynchronous Multicore

Our proposed AMP-MP is implemented in multicore ANoC embody the SAHB architecture. In this context, we adopt full-custom approach based on the 65nm CMOS process where the input and output interfaces are exactly the same as portrayed in Fig. 8. Microphotograph of multicore ANoC, leveraging the SAHB architecture, is depicted in Fig. 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).



(a) (b)

Fig. 11: Microphotograph of 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 the main features of our proposed AMP-MP implemented in AES-CCM and compare the results with conventional implementation and the reported techniques.

A. *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 depicted in Fig. 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 2.5GSamples/second. The power dissipation measurement profile is depicted in Fig. 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 2.079µs with average dissipated power is 357µW and followed by the authentication process which is performed in Core_4 and Core_5 through ANoC in 0.34µs where its average of power dissipation is 631µW. The overall throughput of matrix multiplication for 256 blocks *B* and followed by authentication process is 13.54Gbps in 2.42µs with average power dissipation is ~500µW. 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.

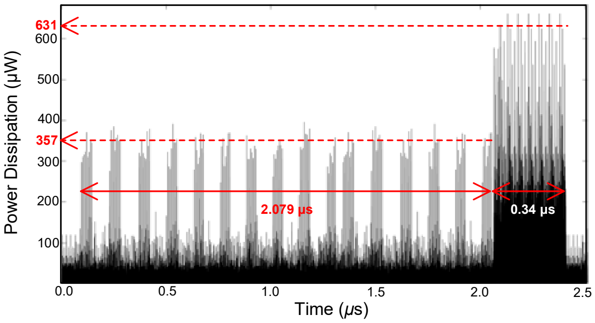


Fig. 12: Power dissipation of matrix multiplication and authentication process

By comparing the throughput performance, based on the same parameters (i.e. clock, supply voltage and single core processor) and platform, our propose AMP-MP on AES-CCM is 35.91× faster than conventional implementation. In this comparison, the conventional implementation [14] dissipates 721µW power and requires 86.7µs to both authenticate and encrypt 256 blocks *B*.

We further investigate the preimage resistant of our proposed on 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 II 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 [1] such as SHA-256 and SHA-512 which are commonly used for message authentications (i.e. MAC).

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

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

As tabulated in Table II, 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" [5] from 2^{16} sets of input plaintext. The collisions can be analyzed based on $2^{2,048}$ bits (2^{16} sets of input) which is considered as a large search space to identify birthday paradox. Based on our analysis on 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.

B. *Parallel AES with rescheduling the operations in ANoC*

Fig. 13 depicts the power dissipation profile measured from three encryption cores to generate ciphertext and MAC through ANoC. The parallel encryption requires 3.26µs and dissipates 307µW average power to encrypt 256 *Ctr*(s) and XOR operation with plaintexts. 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 20.94µs 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×. 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 315µW nominal power during 3.93µs, thus the throughput is 8.32Gbps.

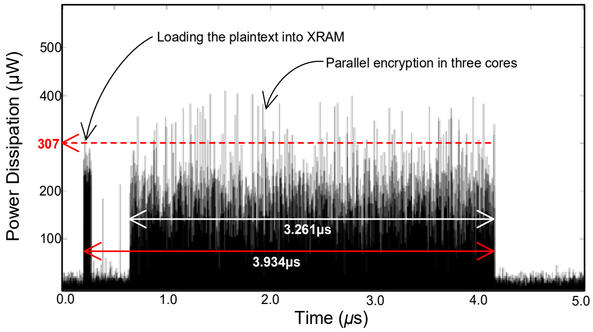


Fig. 13: Power dissipation of parallel encryption in three cores which dissipates power 307µW in 3.26µs

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 III 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 III: 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 emanation of our proposed AMP-MP during the authentication and encryption in AES-CCM and compare with a single core synchronous implementation @100MHz as depicted in Fig. 14. The EM emanation of 132 dB (in µA/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 emanation 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 emanation 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 (9) can improve the resistance level against SCA by breaking the correlation of EM emanation with processed data.

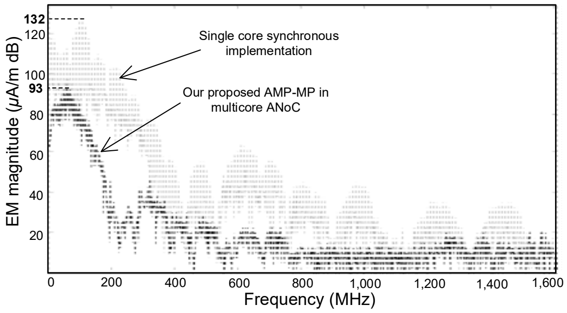


Fig. 14: EM emanation of our proposed AMP-MP in multicore and single core synchronous implementation

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. Fig. 15 depicts 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%.

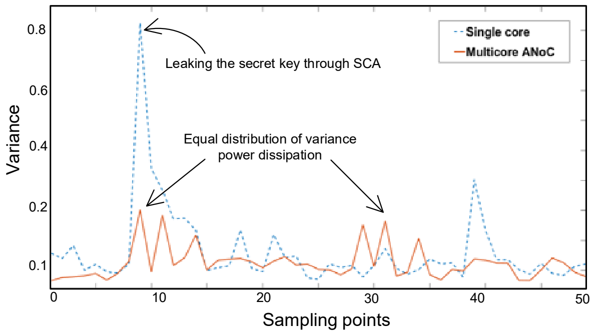
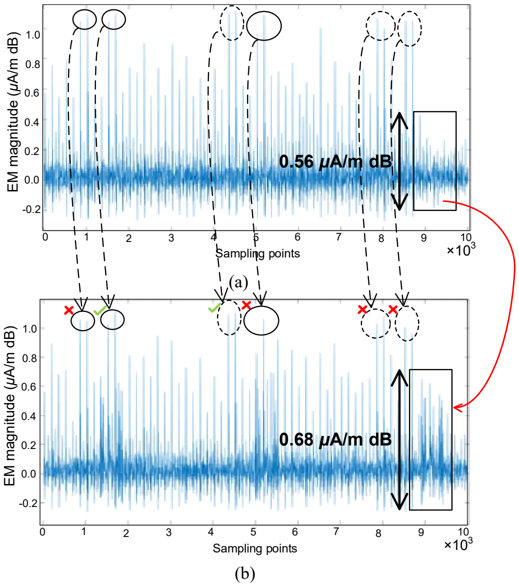


Fig. 15: Comparison of the variance power dissipation during the encryption

With lower leakage probability (i.e. < 20%), the adversary will require the higher number of traces to reveal the secret key in the SCA, even with Higher Order (HO) attack [25]. This is mainly due to the leakage information of the secret key are sparse and randomized in the sampling points with lower leakage probability. With HO attack, the number of traces required to reveal the secret key will be marginally reduced with additional computational resources to perform the SCA.

C. *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 emanations of the stateful key updating [17] for synchronous applications. The secret key is udapted based on the master key (the intial secret key) to determine the subsequent secret keys. Fig. 16 depicts the EM emanation of the key updating system in 10^4 sampling points. It shows that six similar patterns are detected in Fig. 16(a) which lead to the future computational attacks of the subsequent secret key. However, with our proposed key adjusting technique, in Fig. 16(b) four similar patterns on key updating system are omitted hence secured against patterns based attack. The additional ~22.4% of the EM noise (from 0.56 to 0.68 µA/m dB) are generated by circuit of shift register.



(a)

(b)

Fig. 16: The EM emanation 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 [16] 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× key updates [17] 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 depicted in Fig. 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 depicted in Fig. 17(a). Upon detecting the patterns, we perform our proposed key adjusting technique, which has been shown in Fig. 10, during the key updates and hence, the similar patterns are completely removed as depicted in Fig. 17(b).

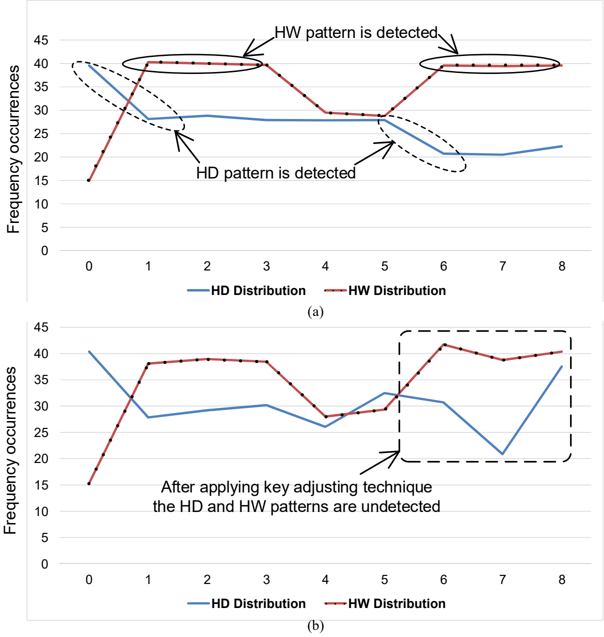


Fig. 17: HD and HW distribution of 256 secret key (a) HD and HW patterns are detected and (b) Performing key adjusting technique to remove patterns

Table IV tabulates the measurement results on variance distribution, time, power dissipation, EM emanation 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 0.12µs which dissipates 41.35µW and 7.6dB for power and EM respectively. In this experiment, the resulted of 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 bit randomly. The adversary still requires huge 2^{128} possible keys to predict the correct key.

TABLE IV: MEASUREMENTS OF TIME, POWER AND EM OF KEY ADJUSTING

Patterns detection		Distribution (σ)		Time (µs)	Power (µW)	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

V. EXPERIMENTAL RESULTS ON SCA

We measure the Success Rate (SR) of the leakage information SCA based on Test Vector Leakage Assessment (TVLA) [22] to detect the presence of the leakage information in the measurements (power or EM) and determine the leakage probability. The values of the TVLA computed based on Welch's t-test is generally determined before quantifying the SCA traces [6]. Fig. 18 depicts the SR of the power and EM analyses based on single core and multicore implementations where the multi leakages measurements are power and EM obtained based on asynchronous multicore implementation.

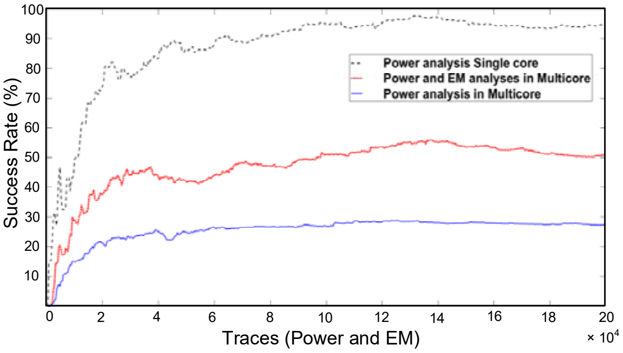


Fig. 18: The SR of power and EM analyses based on single core and multicore implementations

The SR of the leakage measurements shows that the conventional AES in single core implementation has a probability of 70% to leak the secret key at $2{\times}10^4$ power traces (i.e. 70% of the secret keys can be revealed at $2{\times}10^4$ traces). On the contrary when AES is implemented in asynchronous multicore, the SR is reduced to ~28% even after $2{\times}10^5$ 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 $2{\times}10^5$ traces. The result implies that the leakage information of the secret key can only be revealed mostly 50% at $2{\times}10^5$ traces based on our proposed asynchronous multicore implementation.

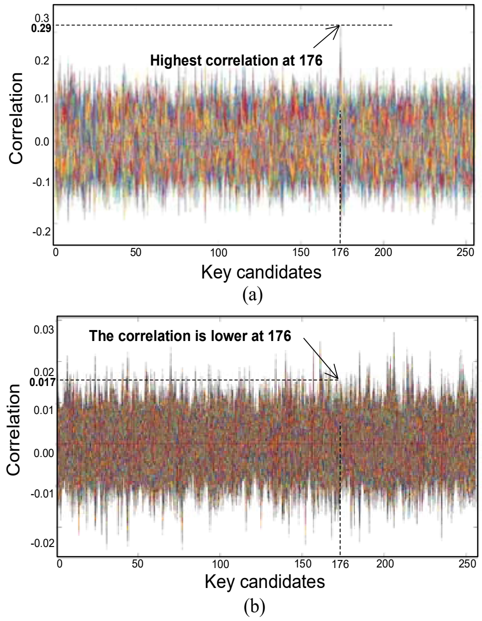
In addition, the SR of the key adjusting technique is also determined based on TVLA which refers to four scenarios in Table IV. The SR of the key adjusting technique when implemented in 2 cores for both power dissipation and EM emanation measurements are 14.87% and 9.32% respectively. The 2 cores refer to the highest distribution of HD and HW. The lower SR is obtained due to the random delay from ANoC and low Signal-to-Noise Ratio (SNR) of the leakage 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 Correlation Power Analysis (CPA) and Correlation EM Analysis (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 [13] which measures only one physical leakage information (i.e. power dissipation) and Multi-Channel [17] which is employing more than one physical leakage information (i.e. both power dissipation and EM emanation) 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 leakage measurements and processes data.

A. Single Channel SCA

Figs. 19(a) and (b) depict 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 depicted in Fig. 19(a). On the contrary, by activating the ANoC between Core_4 and Core_5, the correlation is reduced by 17× (0.29 → 0.017) with SNR of 0.96 dB as depicted in Fig. 19(b). The reduction of correlation is mainly due to the dual-rail asynchronous SAHB in ANoC which distributes equally the variance of leakage information in time domain.



(a)

(b)

Fig. 19: CPA attack on AES-CCM based on (a) Single core and (b) Multi-core with ANoC protocol

B. 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 depicted in Fig. 20. It shows that, based on single channel using CPA (black) and multi-channel attacks based on both CPA and CEMA (red) concurrently [17], the secret key is still unrevealed with 5×10^5 measurements for both power dissipation and EM emanation. The correlation of the secret key is remain 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×. The comparison result is determined based on the conventional AES implementation which has been depicted in Fig. 19(a) and the multi-channel attacks in Fig. 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.

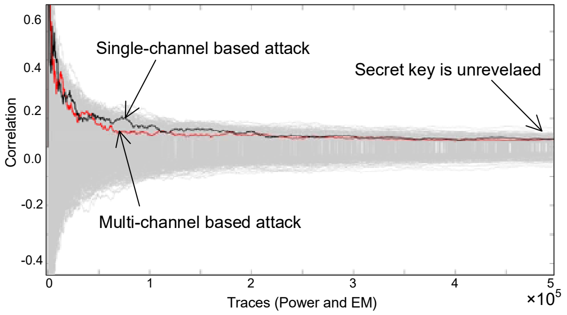


Fig. 20: The plot of Single-channel (black) and Multi-channel result (red) based on CPA and CEMA which are unrevealed at 5×10^5 measurements

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 V 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× to 70× faster than reported techniques during the authentication-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 [23] have been reported as highly efficient algorithms, Deoxys, NORX and CLOC. Table VI tabulates the performance of our proposed AMP-MP in AES-CCM and compare with three AEAD algorithms. The area is compared in terms of Kilo-Gate Equivalent (KGE) which indicates the active area used for algorithm implementation. The efficiency parameter used in this comparison is the throughput/area, Gbps/KGE.

TABLE VI: 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 [23]
#implemented using Global Foundries 65nm CMOS technology

TABLE V: 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 [8]	Three control modules	2	100.00	1.05 (8×)	Spartan 3 3S4000	Yes	N/A
Parallel two AES [9]	Reordering iterations	2	264.00	2.69 (3×)	CMOS SAED 90nm	N/A	N/A
Open System Inter AES [10]	Interconnection AES	2	152.42	1.95 (4×)	XC4VLX40	N/A	N/A
Unified data [11]	Redundancy Check	1	341.58	3.71 (2×)	XC7Z020	Yes	N/A
Ultra-low power AES [12]	8-bit AES core enc.	1	149.00	0.12 (70×)	ASIC 65nm CMOS	N/A	N/A
Conventional AES-CCM [14]	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

The NORX is an online cipher which computes the intermediate states and authentication tag based on the secret key [24]. In Table IV, the NORX outperforms our proposed AMP-MP in asynchronous multicore in terms of throughput (0.1×) and efficiency (0.13×). However, the NORX is still vulnerable against SCA (i.e. CPA decryption attacks) [24], 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 NORX implementation is noticeable during the authentication process at the leakage measurements.

VI. Conclusions

We have proposed an Authentication based Matrix-transformation cum Parallel-encryption implemented on Multicore Processor 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 is 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 (>5×10^5 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 future computational attacks and the leakage patterns of HW and HD during key updates and MMA. With all these advantageous, our proposed AMP-MP in AES-CCM is suitable for secured IoT applications.

References

[1] W. Stallings, *CRYPTOGRAPHY AND NETWORK SECURITY PRINCIPLES AND PRACTICE*, Fifth Edit., vol. 139, no. 3. Pearson Education, Inc, 2011.

[2] M. J. Dworkin, "Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication," *NIST Special Publication 800-38B*, 2007.

[3] 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.

[4] 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.

[5] 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.

[6] S. Mangard, E. Oswald, and T. Popp, *Power Analysis attacks: Revealing the secrets of smart cards*. 2007.

[7] 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.

[8] 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.

[9] 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.

[10] I. Algredo-Badillo, C. Feregrino-Uribe, 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] K. L. Chang, J. S. Chang, B. H. Gwee, and K. S. Chong, "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 Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 3, no. 1, pp. 23–34, 2013.

[21] 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.

[22] "Leak Me If You Can: Does TVLA Reveal Success Rate?", : [Online]. Available: https://eprint.iacr.org/2016/1152.pdf

[23] Kumar, S., Yahya, J. H., Khairallah, M., Elmohr, M. A., Chattopadhyay, A., "A Comprehensive Performance Analaysis of Hardware Implementations of CAESAR Candidates", 2018. [Online]. Available: https://eprint.iacr.org/2017/1261.pdf

[24] Vaudenay, S., Vizàr, D., "Under Pressure: Security of Caesar Candidates Beyond Their Guarantees", 2018. [Online]. Available: https://eprint.iacr.org/2017/1147.pdf

[25] Gierlichs, B., Batina, L.,Verbauhede, I., "Revisiting Higher-Order DPA Attacks: multivariate mutual information analysis", [Online]: https://eprint.iacr.org/2009/228.pdf

Ali Akbar Pammu (S'15) received the B.Eng. (Hons.) degree in electrical and electronic engineering from Nanyang Technological University (NTU), Singapore, in 2014. He was a recipient of the NTU Graduate Research Scholarship. He was awarded as the best presenter in *ICISE2016* conference, California USA and the best student paper award in *ISIC2016* conference, Singapore.

Mr. Ali is currently a PhD student with the Hardware Assurance Team, Temasek Laboratories @ NTU, Singapore. His current research interests include digital hardware security, profiling leakage measurements for SCA by leveraging machine learning algorithms, fastest leakage assessment SCA and countermeasure SCA.

Weng-Geng Ho (S'10-M'16) received the B.Eng. (Hons.) and Ph.D. degrees in electrical and electronic engineering from Nanyang Technological University (NTU), Singapore, in 2009 and 2016 respectively.

Dr. Ho is currently a Research Scientist with the Hardware Assurance Team, Temasek Laboratories @ NTU, Singapore. His current research interests include low power secured memory design, digital VLSI design, asynchronous-logic circuit design, NoC-based multicore platform design and side-channel-attack countermeasures. Dr. Ho was a recipient of the NTU Graduate Research Scholarship.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2018.2869344, IEEE Transactions on Information Forensics and Security

Ne Kyaw Zwa Lwin received the B.Eng. and M.Sc. degrees in electrical and electronic engineering from Nanyang Technological University (NTU), Singapore, in 2011 and 2014 respectively.

He is currently a Research Associate with School of Electrical and Electronic Engineering, NTU. His current research interests include hardware security, space-grade resilient circuits and systems, and digital VLSI design.

Kwen-Siong Chong (S'03-M'09-SM'13) received the B.Eng., M.Phil. And Ph.D. degrees in electrical and electronic engineering from Nanyang Technological University (NTU), Singapore, in 2001, 2002, and 2007 respectively.

He is presently a Senior Research Scientist with Temasek Laboratories @ NTU, Singapore. He was a visiting researcher in Nara Institute of Science and Technology, Japan, in 2010, and in the University of Michigan, USA, in 2012. He is/was principal investigator (PI), co-PI, and collaborator of several research projects, including the projects supported from National Science Foundation (Singapore), Defense Advanced Research Projects Agency (USA), Ministry of Education (Singapore), and Public Sector Research Funding (Singapore). His research interests include hardware security, space-grade resilient circuits and systems, asynchronous VLSI designs, low-voltage low power VLSI circuits, and audio signal processing.

Dr. Chong was the Chair of IEEE Circuits and Systems (CAS) Society, Singapore Chapter, in 2017 and 2018. He has served as an organizing committee for several conferences, including the ASP-DAC 2014, DSP-2015 and DSP-2018. He has been a member of IEEE CAS Society VLSI Systems and Applications Technical Committee since 2009. He is an IEEE senior member.

Bah-Hwee Gwee (S'93-M'97-SM'03) received the B.Eng. degree in Electrical and Electronic Engineering from University of Aberdeen, U.K., in 1990. He received the M.Eng. and Ph.D. degrees from Nanyang Technological University (NTU), Singapore, in 1992 and 1998 respectively.

He was an Assistant Professor in School of EEE, NTU from 1999 to 2005 and has been an Associate Professor since 2005. He was Assistant Chair (Students) from 2010 to 2014 and he is currently the Assistant Chair (Outreach) of School of EEE. He was the Principal Investigators (PIs) of a number of research projects amounting to more than US$12M. He has published more than 100 technical papers, 7 patents (3 granted in USA) and started a Start-up Company in 2005.

He was the Chairman of IEEE-Singapore Circuits and Systems Chapter in 2005, 2006, 2013 and 2016. He is currently the Chair of the IEEE Circuits and Systems Society DSP TC. He was the TPC Chair for ISIC-2011, 2014 and 2016 and the General co-chair for IEEE DSP 2018. He has served as Associate Editors of a number of journals, including IEEE TCAS-1 (2012-2013), IEEE TCAS-II (2010-2011, 2018-on going) and Journal of Circuits, Systems and Signal Processing (2007-2012). He was an IEEE Distinguished Lecture for Circuits and Systems Society in 2009/10 and in 2017/18. He was awarded the Singapore Defense Technology Prize in 2016.