Circuit and System Design for Optimal Lightweight AES Encryption on FPGA

Ming Ming Wong, M. L. Dennis Wong, Cishen Zhang and Ismat Hijazin

***Abstract*—The substitution box (or commonly termed as S-Box) is a non-linear transformation, and known as the bottleneck of the overall operation in AES cipher. Due to recent emergence of high performance and lightweight applications, the required optimum AES cipher has to be both hardware cost effective and computationally efficient. In this study, we implemented various S-box architectures in AES encryption in order to perform an in-depth hardware analysis on FPGA platform. These architectures are the hard-coded LUT S-box, the pure combinatorial S-box using composite field arithmetic (CFA), the pipelined version of CFA S-Box, the CFA AES S-box using direct computation and Linear Feedback Shift Register (LFSR) based S-Box. As a result, a total of six AES ciphers with different S-box architectures are synthesized and implemented on FPGA platform. Considering both the hardware size (total Logic Elements (LE)) as well as the performance (throughput (Mbps)) the optimum AES cipher implementation is derived in this work. The presented implementation is proven lower in hardware area occupancy and higher in computational speed compared to the existing works.**

***Index Terms*—Advanced Encryption Standard (AES), AES S-box, Lightweight Cipher, Composite Field Arithmetic (CFA), Pipeline, Linear Feedback Shift Register (LFSR).**

I. Introduction

EVOLUTION of ubiquitous computing has led to the rise of Internet of things since the early twentieth century. As a result, large amount of digital data are constantly exchanged between various embedded devices over the internet. Therefore, ciphers are widely used to ensure the security of these digital data [1]–[3].

In November 2001, National Institute of Standards and Technology (NIST) has chosen Rijndael block cipher as the Advanced Encryption Standard (AES) [4]. Since then, AES has gradually replaced the Data Encryption Standard (DES) and eventually becoming one of the most preferred symmetric cryptographic protocols. Due to its increasing popularity in area-constrained applications, such as the embedded systems, the need of compact AES cipher implementation became more important than ever before [5], [6]. Later, the advent of NFC has redefined the optimality of AES cipher in lightweight applications. These features are not only small in hardware size but also fast in speed performance.

Among the four key AES operations, the SubBytes transformation (S-box) is recognized as the bottleneck of the entire AES encryption process. The transformation computes the most resource demanding multiplicative inversion operation in finite field arithmetic. To deal with the complex computation problem, various design techniques have been proposed in order to achieve low hardware cost (area and power consumptions) and high speed performance in AES S-box implementation. It is also noted that, optimization in the MixColumn transformation gained a considerable amount of research interest which is the second most resource consuming operation after SubBytes Transformation [7].

The focus of this paper is on the S-box design for AES cipher. In order to achieve the optimum features required in the recent lightweight applications, this work aims to design AES ciphers with the minimum hardware cost. This is achieved by a series of optimizations to reduce the critical path and hence enhance the speed performance.

The contribution of this work is twofold. First, it conducts a thorough design analysis and hardware review on different type of S-boxes. The S-boxes will be classified into several categories according to the nature of their architecture design. Furthermore, the Linear Feedback Shift Register (LFSR) based S-Box in AES cipher, which is relatively new in the research area is included in the implementation benchmarking analysis.

Second, the S-box selected from each category is adopted in full AES encryption and implemented on FPGA platform. The hardware AES cipher with various S-box designs is evaluated in terms of its hardware size and speed performance. Further optimization in algorithmic and architectural levels are employed in our the circuit design. Based on the synthesis and implementation results, the best AES cipher that is most suitable for lightweight applications is identified.

The remainder of this paper is organized as follows: Section II briefs the operations in the AES encryption process. Existing works in AES S-box hardware optimization methodologies are reviewed in Section III. The optimized S-box implementations adopted from our previous studies can be categorized into four different major designs which will be presented in Section IV. The FPGA implementations of AES cipher with various S-box architectures and their hardware analysis will be evaluated in Section V. Finally, the conclusion and future work are drawn in Section VI.

II. Advanced Encryption Standard (AES)

AES is a symmetric block cipher that process data blocks of 128-bits with different key lengths; 128, 192 and 256 bits and resulting in 10, 12 or 14 computation round respectively. Therefore, there are three AES variants based on the key-lengths which are namely the AES-128, AES-192 and AES-256. In this study, we focus only on encryption process of AES-128, which requires key lengths of 128-bits and 10 computation rounds.

Manuscript received November 15, 2016; revised August 04, 2017. This work is supported in part of Swinburne Melbourne-Sarawak Research Collaboration Scheme (MSRCS)

M. M. Wong is with School of Computer Science and Engineering, Hardware & Embedded Systems Lab (HESL), Nanyang Technological University, Singapore. Email: mmwong@ntu.edu.sg

M. L. Dennis Wong is with Institute of Sensors, Signals and Systems, Heriot-Watt University Malaysia, Putrajaya, Malaysia

C. Zhang and I. Hijazin are with Faculty of Science, Engineering and Technology, Swinburne University of Technology, Australia.

The encryption begins by copying the input to an array, termed as *State*. The *State* is an array of 16 bytes arranged in four rows and four columns (4×4). Hence the transformations will be performed on each of the blocks of byte in the *State*, which are represented as finite field elements of $GF(2^8)$ with the specified field polynomial $q(x) = x^8 + x^4 + x^3 + x + 1$. The initial key is then added to the input value in the initial round. Next, the consecutive identical 9 rounds take place and end with a slightly modified final round. The normal computation round (9 rounds) in AES consists of four transformations in the following orders; **SubBytes**, **ShiftRows**, **MixColumns** and **AddRoundKey**. The final round is the normal round without the MixColumns transformation.

Symmetry key cryptography such as the DES and AES ciphers utilizes a substitution box (S-box) to provide the substitution-permutation mechanism to achieve both diffusion and confusion properties in their block ciphers. For AES, the S-box is performed in the first transformation, i.e. **SubBytes transformation** which consists of two main operations. First, each of the data byte is substituted by its multiplicative inversion in $GF(2^8)$ over the field polynomial, $q(x) = x^8 + x^4 + x^3 + x + 1$. The operation is relatively expensive in term of hardware and is therefore identified as the bottleneck in achieving compact and fast AES cipher implementation. Second, bit scrambling is performed using affine transformation over $GF(2)$.

The second transformation, the **ShiftRows transformation** cyclically shifts each row of the *State* by certain offset. To be precise, while the first row remained unchanged, the second row is shifted by one, the third row by two and the fourth row by three bytes to the left. This computation is simple in hardware as no additional hardware is required for shifting. Unlike any other transformation, the third transformation, **MixColumns transformation**, processes one column of the *State* at a time. This transformation basically involves matrix multiplication. Each byte is interpreted as the coefficients of four-term polynomial over $GF(2^4)$, the column is then multiplied with a fixed polynomial $a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$ and modulo $x^4 + 1$.

The fourth transformation, the **AddRoundKey** is a simple addition between the *RoundKey* and the 128-bit *State*. Note that the addition operation in finite field arithmetic is realized using logical gate XOR in hardware. The required *RoundKey* is generated from another operation, called the **KeyExpansion**. The **KeyExpansion** uses the initial key to generates all the 10 *RoundKey* that are required in every computation round in AES.

Further details of AES transformation are also available from the original source by Daemen and Rijmen in [8]. The algorithm structure is also as illustrated in Figure 1.

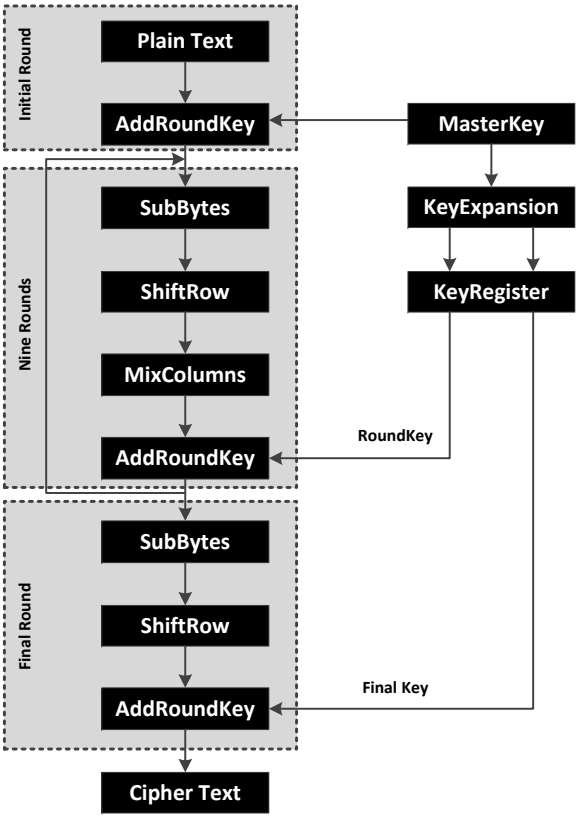

Fig. 1. Description of AES Encryption Algorithm

III. Literature Review

As reviewed in Section II, in AES, the SubBytes transformation (S-box) involves the computation of the most resource consuming operation in finite field arithmetic, which is the multiplicative inversion. The conventional and the most straightforward implementation approach is by the means of hard-coded memory through LUT. The LUT contains the permutation of all 256 possible 8-bit values. Hence, the tedious computation in AES S-box is simplified for mapping each individual input byte to a new output byte through the table. Such implementation often results in high-speed circuitry but is very expensive in terms of the hardware cost as each of the LUT S-box requires $2,048$ memory bits. Furthermore, in a high-speed design of AES-128 cipher, 16 parallel LUT S-box modules are needed for a single round.

Back in the early 20th century, with the rise of computers and Internet, cryptography algorithms began to play a vital role in several communication applications. Subsequently, dedicated hardware with high speed performance were needed to handle the complex computations in the ciphers. In order to fulfill these requirements, several efficient and high throughput AES hardware implementations were proposed such as that in [9]–[17].

The need of cryptography algorithms was then further extended to the resource-constrained applications due to the rapid emergence of lightweight embedded systems in the 21st century. Therefore, various area efficient implementations of AES S-boxes using composite field arithmetic (CFA) [18]–[20] were introduced.

These results derived equivalent computation in smaller fields such as $GF((2^4)^2)$ [21]–[24] or $GF(((2^2)^2)^2)$ [25]–[32].

On the other hand, the CFA approach generally results in complex circuitry with long critical path. This prevented efficient pipelining to be performed, leading to low performance in hardware implementation. In other words, there exists a significant trade-off between the hardware cost and the performance in CFA AES S-box implementation.

In addition to the pure combinatorial circuitry mentioned above, several other area efficient AES S-boxes that utilized dedicated Block RAMs available in FPGA platform were reported in [33]–[38]. However, these techniques may not be an attractive solution in terms of of ASIC implementation.

Several other methodologies have been introduced in the literature to design low power AES hardware circuitry [39]–[43]. Among all these low-power techniques, the decoder-permute-encoder structure (Decoder Switch Encoder (DSE) architecture) proposed by Bertoni et al. consumed the least power for AES S-box implementation [43]. The structure of the circuit is short in critical path architecture and induces minimal signal activity within the circuit when the input changes. Though the design resulted in low power consumption, its size is about three times as large as that of the pure CFA work [44].

In recent years, a new approach using the maximum length Linear Feedback Shift Register (LFSR) to compute the multiplicative inversion has been investigated and implemented in the AES S-box. LFSR is often used in cryptographic implementations such as the hash functions, pseudorandom number generators (PRNGs) and stream ciphers. However, it is considerably new in block cipher implementation and is yet to be commonly applied to AES S-box in comparison to the CFA approach. To our best knowledge, LFSR based multiplicative inversion customized for AES cipher was only reported in [45]–[48]. These works managed to produce low hardware cost AES S-box comparable with that implemented using CFA approach.

In general, several different S-box architectures have been proposed in the literature in an attempt to optimize the non-linear transformation in AES cipher in terms of hardware size, speed and power consumption. With an aim of implementing a small hardware size AES cipher with high computation efficiency, our work shall focus on the CFA approach for S-box and followed by optimization works in algorithm and architectural levels. Meanwhile, the extended CFA AES S-boxes with various pipeline stages employed to minimize the long critical path in the complex circuits have been proposed in our previous studies [49]–[51]. With the shorter critical path (reduced due to the insertion of pipelines), our works effectively produces CFA AES S-box designs with higher speed performance and smaller dynamic power consumption at the cost of a slight increment in the hardware cost.

It is noted that, the detailed hardware analysis of the effectiveness and the efficiency of each S-box in AES cipher has not been evaluated in the literature. As an extension of our previous works, this study will therefore, benchmark the cost and the performance of various S-boxes in AES cipher implemented on FPGA platform.

IV. AES S-box Designs

In this section, four different major designs of AES S-box will be discussed and evaluated in details. These are namely the pure composite field arithmetic (CFA) S-box (Section IV-A), the pipelined version of CFA S-box (Section IV-B), S-box using direct computation (Section IV-C) and linear feedback shift register (LFSR) based S-box (Section IV-D).

A. Composite Field Arithmetic (CFA) AES S-box

As opposed to the conventional method that utilized LUT, Rijmen [52] was the first to propose an alternative solution by employing CFA [18]–[20], [53] to compute the multiplicative inversion over $GF(2^8)$ in AES S-box. In CFA, finite field elements are represented in its equivalent composite field, i.e $GF(2^l) \equiv GF((2^m)^n)$, where $l = m\dot{n}$. The composite field is built iteratively from its lower order fields, allowing the actual mathematical manipulation to be performed in the lower fields instead of its original higher order field. Subsequently, in AES S-box, the element in $GF(2^8)$ can be mapped to its subfield and this will yield less complexity in the multiplicative inverse. The following summarizes the steps in performing multiplicative inversion using CFA:

1) Map all elements of field *A* to a composite field *B* using isomorphism function; $b = f(a) = \delta \times a$.
2) Compute the multiplicative inverse over *B*; $x = b^{-1}$ (except if $b = 0$, then $x = 0$).
3) Remap the computation results to *A*, using the inverse isomorphism function; $a = f'(x) = \delta^{-1} \times x$.

Complexity of a field is heavily dependent on several factors: **field of mapping**, **representations of the field elements, (i.e. field polynomials and the basis representations used)** and **isomorphic mapping chosen for the representation**. Thus, one can conveniently take the advantage of the isomorphism to map a computation from one field to another to search for the most efficient implementation. Mapping the finite field from $GF(2^8)$ to $GF(((2^2)^2)^2)$ requires three stages of isomorphism and field polynomials which are listed (in a general form) below:

$$r(y) = y^2 + \tau y + \nu \quad \text{(isomorphism for } GF(2^8)/GF(2^4)) \quad (1)$$

$$s(z) = z^2 + Tz + N \quad \text{(isomorphism for } GF(2^4)/GF(2^2)) \quad (2)$$

$$t(w) = w^2 + w + 1 \quad \text{(isomorphism for } GF(2^2)/GF(2)) \quad (3)$$

Generally, multiplicative inverse over the composite field $GF(((2^2)^2)^2)$ can be performed with respect to either **polynomial basis** or **normal basis**. The computations are also determined by the coefficients of their respective field polynomials; (1), (2) and (3). Hence, the coefficients in the field polynomials have a direct influence towards the computation complexity. As $w^2 + w + 1 = 0$ is the only irreducible polynomial of degree 2 over $GF(2)$, there is no other candidate coefficient for (3). For (1) and (2), there are several possible coefficients of ν, τ, N and T in both normal and polynomial bases. In order to promote simplicity in CFA, one could either have the trace or the norm of $r(y)$ and $s(z)$ equal to unity but not both.

To our best knowledge, most studies such as reported in [23], [25], [26], [28], [31], [52], [54] attempted the optimization with traces of field polynomials equal to unity using polynomial basis representation. Similar case reported in [27] used normal basis representation instead. On the other hand, our previous works have chosen the norms (ν and N) of the field polynomials to be unity for both polynomial

and normal bases [29], [30]. As a result, three CFA S-box constructions (listed as below) were derived. Theoretical and empirical results showed that **Case C** (see Figure 2) successfully strikes a balance between the total number of gate counts (area) and its respective critical path (speed) in comparison with the conventional CFA selections reported in [25]–[28], [54].

1) **Case A**: Using polynomial basis representation with field polynomials' norms equal to unity (both ν and N in (1) and (2) equal to unity).
2) **Case B**: Using normal basis representation with field polynomials' norms equal to unity (both ν and N in (1) and (2) equal to unity).
3) **Case C**: Using normal basis representation with τ in (1) and N in (2) equal to unity.

Due to its promising implementation results, the S-box from **Case C** is chosen to be deployed in AES encryption. The implementation and the in-depth hardware evaluation of the cipher on FPGA platform will be presented in details in Section V. Meanwhile, further optimizations in architectural and algorithmic levels are presented next in Section IV-B and Section IV-C respectively.

B. Pipelined CFA AES S-box

CFA offers an effective solution to deduce a pure combinatorial AES S-box using the composite field $GF(((2^2)^2)^2)$. Without the need of LUT, the mathematical manipulations can be performed through three isomorphism levels, involving several complex sub-operations modules of different complexities. For this reason, several studies have also verified that the resultant circuit tends to have relatively long critical path [55]. Besides such a complex signal path, the circuit might produce massive dynamic hazards propagation and lead to high power dissipation. In order to curb the problems, our previous works in [49], [50] have attempted to equalize the amount of signal transitions within every clock cycle . This can be done by careful placement of pipelines in the circuit such that the signal arrival times from each parallel path is consistent.

Due to the complicated crossing and branching in the composite field S-box, pipelining alone is insufficient to ensure complete consistency within the circuit. Therefore, prior to pipelining, pre-processing procedure is required i.e. to convert the complicated circuit into several logical expressions without altering the functionality of the original circuit. The computation is expressed using the Algebraic Normal Form (ANF), consisting of only AND gates and XOR gates. Hence the sub-operations which include the 4-bit adder, square-scaler, $GF(2^4)$ multiplier and $GF(2^4)$ inverter (refer to Figure 2) will be first translated into ANF expressions. Next, these sub-operations are merged to form three main computation modules in $GF(2^8)$ inversion circuit, together with isomorphism and inverse isomorphism plus affine transformation respectively. All the five ANF modules in the AES S-box are listed in the following.

1) Isomorphism
2) Two 4-bit adders, a square-scaler and a $GF(2^4)$ multiplier.
3) $GF(2^4)$ inverter
4) Two $GF(2^4)$ multipliers
5) Inverse isomorphism plus affine transformation

With the operations in the $GF(2^8)$ multiplicative inverter is derived using pure logic equations, pipelines can be applied at ease in the next step. Note that the position and the number of the pipeline will effect the overall hardware cost and speed performance. One of the earlier works that attempted 3-stage pipelining was reported in [55]. On the other hand, our work in [49] employed 7-stage pipelining; with one pipeline stage for each ANF modules and additional sub-pipelines in both of the $GF(2^4)$ multipliers. The aim of the sub-pipelining is to further shorten the critical path in the multiplier. Based on the hardware analysis in Table I, the design results in high performance and also increase in the number of hardware resources.

In a separate work, another compact and well-transformed three-level isomorphism CFA S-box was reported in [50]. The work presented a highly modular design, which allows efficient 5-stage pipelining. Not only that, the logic equations were arranged in tree structures that the common factors can be eliminated using subsharing factorization. The block diagram for the pipelined CFA AES S-box presented in [49], [50], [55] are as illustrated in Figure 3. Table I shows the comparison in terms of the hardware requirements and performance for three AES S-box designs on Cyclone III EP3C5F256C6 device.

Based on the hardware results, the works in [49], [55] require 24 pipeline registers for 3-stage pipelining and 90 pipeline registers for 7-stage pipelining respectively. On the other hand, the work in [50] only requires 36 pipeline registers for 5-stage pipelining. Not only that, its speed performance is comparable with the circuit in [49] but also has considerably lower power dissipation. Hence, it can be concluded that S-box in [50] is a better selection in term of the hardware cost and performance efficiency.

C. Direct Computation CFA AES S-box

As discussed in the previous section, the exploitation of pipelining is capable of altering the hardware cost and the performance of the CFA AES S-box. A separate study reported in [51] has applied direct computation to produce an alternative architecture that further extend the benefits of pipelining in CFA AES S-box design. The work derived three main computational modules which are namely the adder (4-bit XOR) and the square-scaler $\nu\gamma^2$ (labeled as M1), $GF(2^4)$ multiplier (labeled as M2) and $GF(2^4)$ inverter (labeled as M3), which are shown in Figure 4. In the first step of the algorithmic optimizations, the computational modules (M1, M2 and M3) are merged and reduced into new boolean modules; **P1**(square-scaler merged with multiplier), **P2** and **P3**(inverter merged with multiplier). The block diagram of the resultant circuit is as shown in Figure 4. The new $GF(2^8)$ inverter is now comprised of only three modules; P1, P2 and P3, with several repetitive common sub-expressions in their boolean functions.

1 **Module P1:** Modules M1 (adder and square-scaler) and M2 ($GF(2^4)$ multiplier) are merged to produce a new module, P1. The derivation of the new module P1 is expressed in boolean functions with respect to input $\gamma_1 = \{a, b, c, d\}$ and $\gamma_0 = \{x, y, w, z\}$.

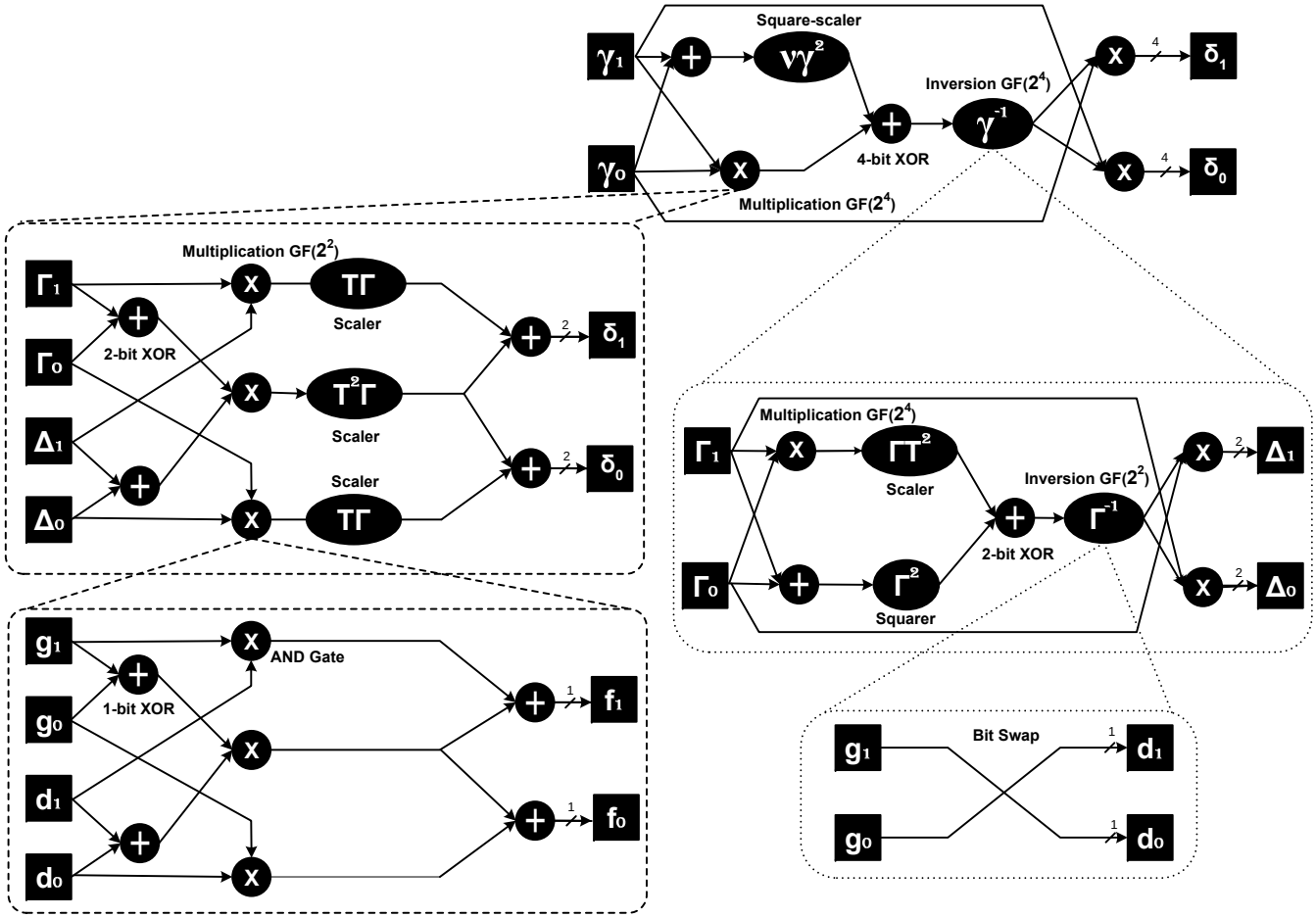

Fig. 2. Multiplicative Inverse over Field $GF(2^8)$

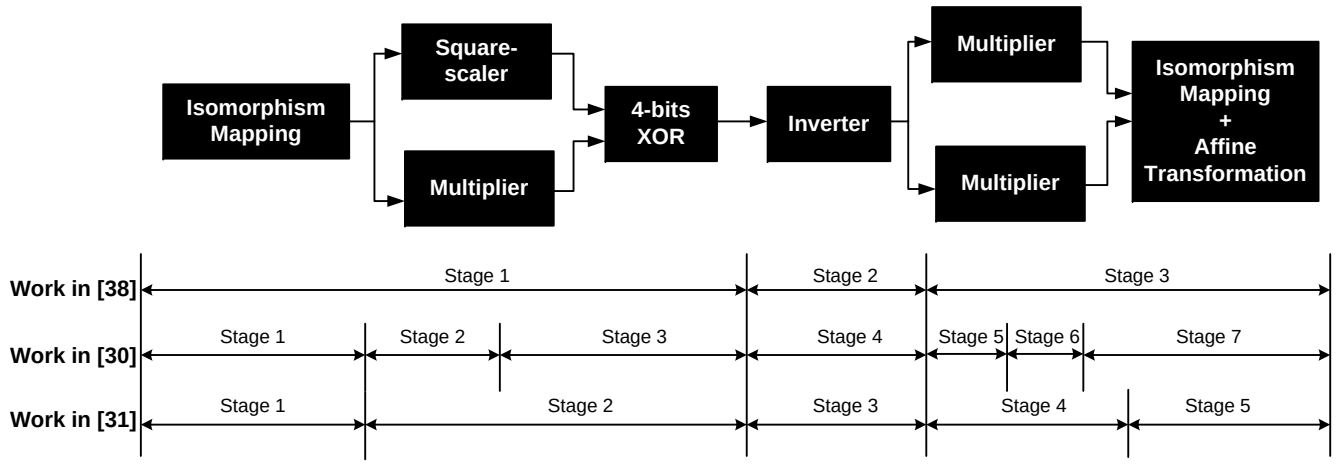

Fig. 3. Pipelined CFA AES S-box

TABLE I
AREA REQUIREMENT, TIMING ANALYSIS AND POWER CONSUMPTION OF FPGA IMPLEMENTATION OF PIPELINED CFA AES S-BOX IN [49], [50], [55]. ALL THE CIRCUITS ARE CLOCKED AT 100MHZ [LE = LOGIC ELEMENT].

Hardware Performance/Requirement	Work presented in [49]	Work presented in [55]	Work presented in [50]
Total LE (5136 LEs)	95	71	66
Combinational Functions (5136 LEs)	75	68	66
Dedicated Logic Registers (5136 LEs)	90	24	36
Total Register (5136 LEs)	90	32	36
Fmax (MHz)	380.66	170.44	346.26
Dynamic Thermal Power Dissipation (mW)	2.26	0.15	1.84

2 **Module P2:** Module P2 is a result of combining modules M2 ($GF(2^4)$ multiplier) and M3 ($GF(2^4)$ inverter). Its boolean functions are expressed as functions of input $\gamma_0 = \{x, y, w, z\}$ and the output from module P1, $P1 = \{k, l, m, n\}$. It is noted that the input γ_0 is obtained from the four least significant bits of the input to the $GF(2^8)$ finite field inverter, as shown in Figure 4. The output of P2 will serve as the four most significant bits of the output of $GF(2^8)$ finite field inverter, δ_1.

3 **Module P3:** Similar to module P2, module P3 is a combination of module M2 ($GF(2^4)$ multiplier) and module M3 ($GF(2^4)$ inverter), with a different set of input data. For module P3, the respective inputs are $\gamma_1 = \{a, b, c, d\}$, which are the four most significant bits of the input to the $GF(2^8)$ finite field inverter (refer in Figure 4) and the output from module P1, $P1 = \{k, l, m, n\}$. Meanwhile, the output of P3 will serve as the four least significant bits of the output of $GF(2^8)$ finite field inverter, notated as δ_0.

Next, the common sub-expressions identified from each individual operator are factorized and eliminated for further cost reduction. The common sub-expressions with respect to the input $\gamma = \{\gamma_1, \gamma_0\}$ and the output from module P1 are as depicted in Figure 5. Effective elimination of these common sub-expressions promotes significant amount of resource sharing and hence leads to efficient computational cost reduction. The mathematical deductions of all the modules M1, M2 and M3 as well as P1, P2 and P3 can be found in details in the original paper [51].

As an extension of the above-mentioned work in [51], 4-stage pipelining is further employed in the design. In this study, the newly derived S-box, together with work from [49] and [50] (discussed in Section IV-B) are also implemented in AES cipher. These implementations and their in-depth hardware evaluations on FPGA platform will be presented in details in Section V.

D. Linear Feedback Shift Register Based AES S-box

In recent years, cryptographic researchers have considered the feasibility of deploying the Linear Feedback Shift Register (LFSR) to perform the multiplicative inversion in the AES S-box. Unlike CFA, the approach is unable to achieve pure combinatorial circuitry. However, without requiring much logical gates, the approach is simple and exhibits high potential in achieving low hardware cost. LFSR uses registers (or flip flops), simple XOR operation of particular bits (the tap position, determined by primitive polynomial) and a shifting operation to generate a collection of cyclic binary states. LFSR updates the current state through direct computation of its predecessor.

In mathematical sense, LFSR transformation at any p cycle is performed as $S(t+p) = T^p \cdot S(t)$ where T is the LFSR transformation matrix, $S(t)$ is the state of the LFSR at the t^{th} instant time or the initial seed and $S(t+1)$ is the state of the LFSR at $(t+1)^{th}$ time instant, i.e. the next clock cycle [46], [47]. With that, multiplicative inversion can be computed with LFSR running with a particular initial seed until the LFSR state matches the given input and the total number of clock cycles lapsed is recorded. Next, the LFSR is re-initiated with the same initial seed and continues to run for the remaining cycles. The total number of cycles in both runs ought to be 255 (for the 8-bit element). Upon arriving at this point, the LFSR will then produce the multiplicative inverses of the given input.

Alternatively, our previous study (in [48]) has proposed a different LFSR based multiplicative inverter that is comprised of two identical 8-bit LFSRs and constructed using primitive polynomial $\{x^8 + x^4 + x^3 + x^2 + 1\}$. There are three non-zero feedback taps, with each implemented using XOR gate. Both LFSRs are loaded with the initial seed (or initial state) $\{00000001\}_2$ and run in parallel but in opposite directions; forward and reverse. With the initial state notated as $S(t)$, the forward LFSR will run continuously in $S(t+i)$ direction, and the reverse LFSR will run simultaneously in $S(t+255-i)$ direction with $i \in \{1, 2, 3...127\}$. Within i clock cycles, when the input matches the current state in either LFSR, the current state of the other LFSR will be the corresponding multiplicative inverse. The block diagram of our LFSR based AES S-box is as depicted in Figure 6.

Sourav Das has also presented two new designs for LFSR based multiplicative inversion that are also applicable to AES S-box hardware implementation [46], [47]. In their work, only one set of 8-bit LFSR is used, which is however constructed using 2-input FFs as reloading is required in their methodology. In order to speed up the overall performance, the computation is split into several stages and a complex mapping function is involved to maneuver the entire procedure.

On the contrary, our work in [48] uses two parallel 8-bit LFSRs. Each of them is constructed using 1-input FFs and hence is smaller in size. Moreover, the need of mapping function is not required. Therefore, the work is implemented in AES encryption for further investigation. The feasibility of LFSR based S-box in comparison to the commonly deployed CFA S-box in AES cipher will be analyzed and evaluated in terms of their hardware size and performance. These will be reported in Section V.

However, as LFSR based stream cipher are susceptible towards cache timing attack, there are chances that the proposed LFSR based S-box may cause vulnerability of the

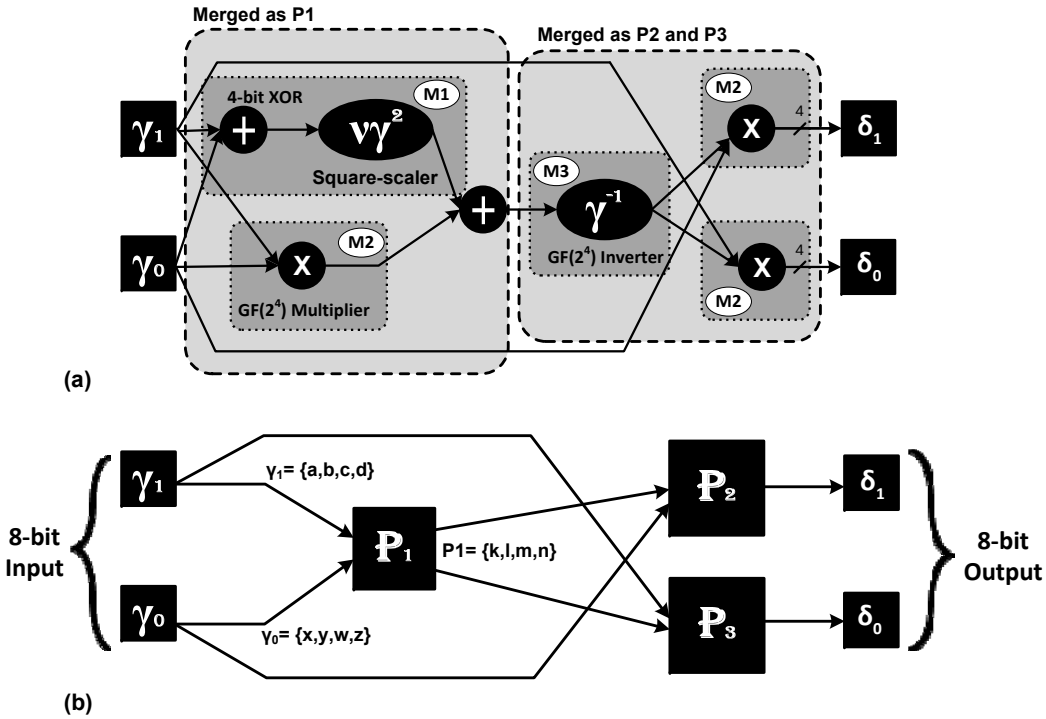

Fig. 4. (**a**) $GF(2^8)$ finite field inverter with its inner modules denoted as M1, M2 and M3. The inner modules are defined over finite field $GF(2^4)$. (**b**) Block diagram of the proposed $GF(2^8)$ finite field inverter after merging.

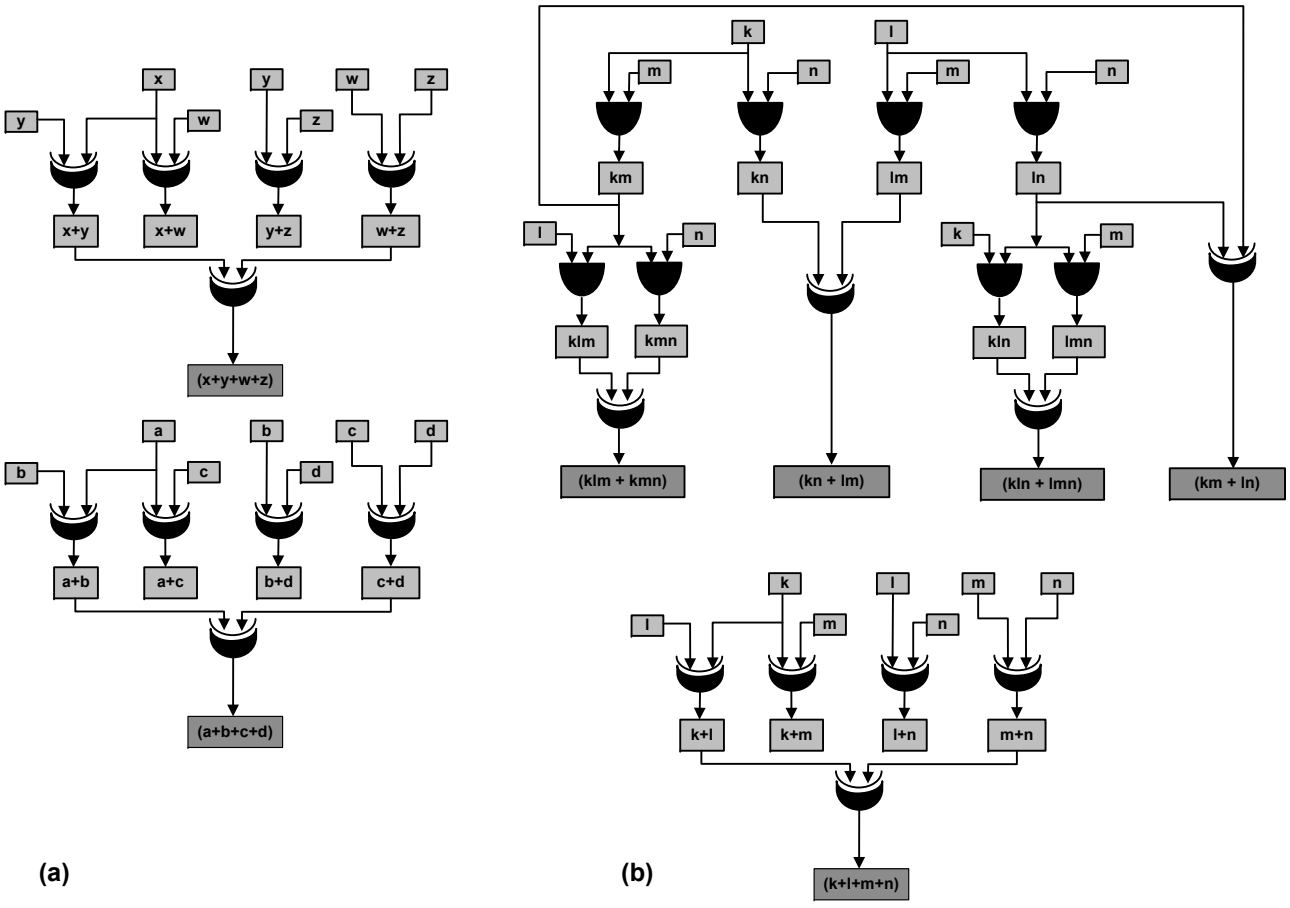

Fig. 5. (**a**) Common sub-expressions with respect to the input γ; $\gamma_1 = \{a, b, c, d\}$ and $\gamma_0 = \{x, y, w, z\}$. (**b**) Common sub-expressions with respect to the output from module $P1 = \{k, l, m, n\}$.

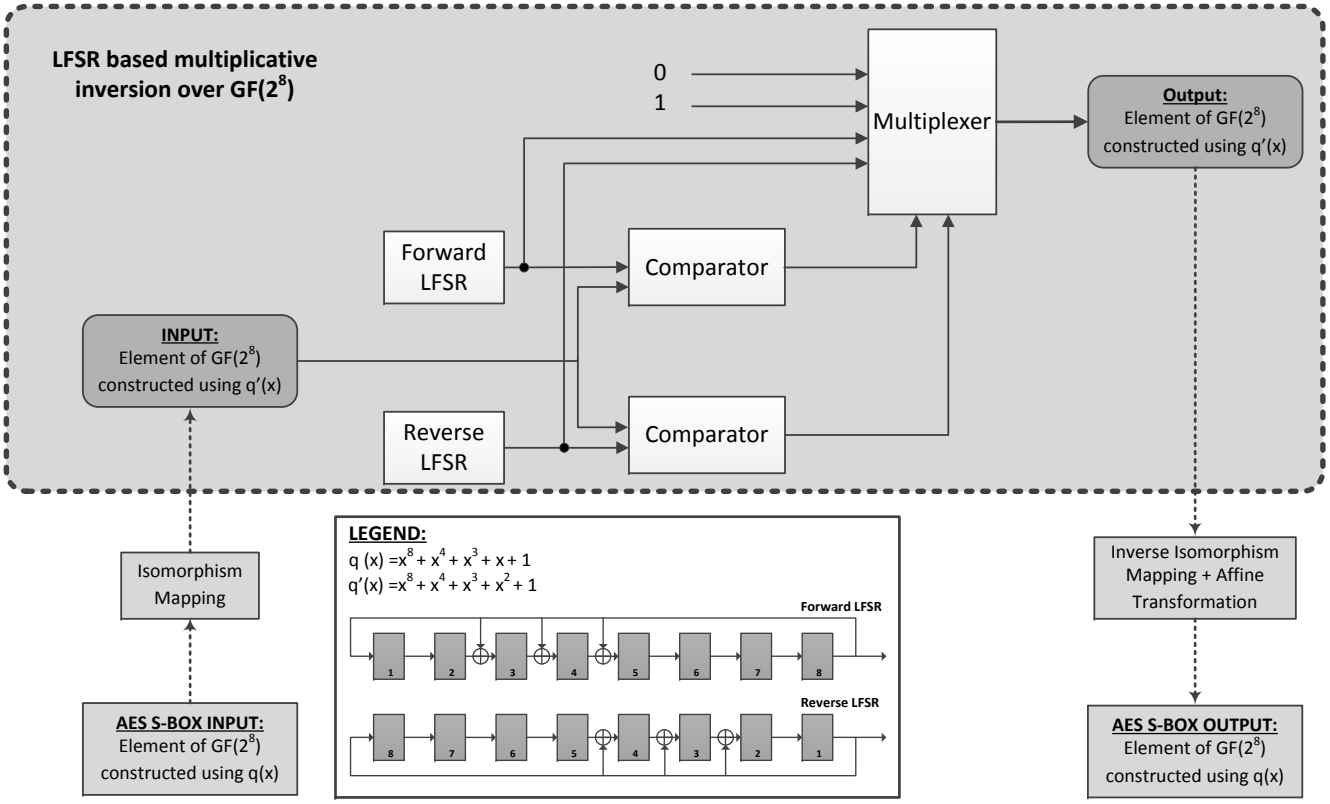

Fig. 6. The proposed LFSR based AES S-box $GF(2^8)$

cipher towards the side-channel attack as well.

V. AES Cipher Implementations and Results

Various AES S-box architectures presented in previous sections are implemented in full AES cipher in order to search for the smallest hardware area occupancy AES with optimum speed performance. These include the S-box realized using LUT (pure hard-coded memory), pure combinatorial circuit through CFA, enhanced version of CFA AES S-box with different number of pipelining stages and followed by algorithmic optimization, as well as the LFSR based implementation.

As a result, a total of six implementations of AES cipher with different S-box architectures (Case I-VI) listed in the following are presented in this section. Furthermore, for fair benchmarking among the various S-box designs, all the other transformation rounds (ShiftRow, MixedColumn, AddRoundKey and KeyExpansion) shall remain unchanged for all implementations.

1) Case I: LUT-based AES S-box
2) Case II: CFA AES S-box [29]
3) Case III: 7-stage pipelined CFA AES S-box [49]
4) Case IV: 5-stage pipelined CFA AES S-box [50]
5) Case V: 4-stage pipelined CFA AES S-box [51]
6) Case VI: LFSR-based AES S-box [48]

These AES ciphers are implemented in Cyclone IVE EP4CE30F29C6 using Quartus II 11.1. Having the architectures clocked at 250MHz, 125MHz and 100MHz, the timing analysis of the architectures are deduced using TimeQuest Timing Analyzer. Table II presents the comparison in terms of the hardware requirements and performance specifications of these six implementations.

The results in Table II reflected that AES cipher in Case I requires the least LE but large hard-coded ROM for the LUT-based S-box. In terms of speed analysis, the result further showed that architectural optimization (pipelining) in CFA AES S-box (in Cases III-V) has effectively enhanced the speed performance of the overall AES cipher implementation.

It can be seen that, pipelining is indeed an effective measure to reduce the critical paths in the complicated CFA S-box circuitry. Note that additional pipeline may contribute to an increase in hardware occupancy. For instance, the cipher has an increase of 5.4% and 5.9% of total LE in Case III and Case IV respectively compared to the pure CFA AES S-box in Case II. However, the effect can be minimized with proper planning during pipeline placement (refer Section IV-B) as well as algorithmic optimizations employed in design (refer Section IV-C). This is proven in Case V where the 4-stage pipelined AES S-box resulted in a reduction of implementation size by 23.7% in the AES cipher.

Among all the cases presented, the AES cipher in Case V strikes a balance in hardware area and performance efficiency. The architecture has the smallest number of LE and its resultant speed performance is better than the pure combinatorial counterpart. It is therefore a suitable candidate for compact AES cipher implementation with high performance.

For detailed verification, the derived best case (AES cipher in Case V) is benchmarked with the related works reported in recent literature and the comparisons are as summarized in Table III. The synthesis results show that our implementation uses a total of 2059 LEs (no LUTs required) and has a throughput of 575 Mbps. The works presented by Chodowiec

TABLE II
AREA REQUIREMENT AND TIMING ANALYSIS AES CIPHERS IMPLEMENTED ON ALTERA FPGA CYCLONE IVE EP4CE30F29C6 USING AES S-BOX IN CASES I-VI. NOTE THAT THE AES IMPLEMENTATIONS FOR CASE I AND CASE II ARE UNABLE TO BE CLOCKED WITH 250MHZ DUE TO TIMING VIOLATION. [LE = LOGIC ELEMENT]

	Case I LUT	**Case II** Pure CFA	**Case III** Pipeline-7	**Case IV** Pipeline-5	**Case V** Pipeline-4	**Case VI** LFSR
Total LE	2149	2699	2845	2859	2059	2370
Total Combinational Functions	2149	2699	2824	2673	2509	2317
Dedicated Logic Registers	957	1084	2284	1998	1678	1309
Memory Bits	32768	0	0	0	0	0
Fmax(clocked at 250MHz)	-	-	307.5	325.41	314.66	155.69
Fmax(clocked at 125MHz)	206.06	150.53	248.14	216.08	231.05	163.59
Fmax(clocked at 100MHz)	215.84	136.54	229.15	234.03	241.6	142.9

TABLE III
COMPARISON OF RELATED WORKS WITH THE PROPOSED AES ENCRYPTION ARCHITECTURE.

Work	Platform	Throughput (Mbps)	Slices/ LEs	LUTs (Kbits)
[56] A	EP2C20F484C7	199	1818 LE	-
[56] B	EP2C20F484C7	306	2234 LE	-
[56] C	EP2C20F484C7	598	3408 LE	-
[57] (Area)	0.13μm	121	3100 Gates	-
[57] (Speed)	0.13μm	232	3900 Gates	-
[33]	Spartan II-6 (0.22μm)	166	444 LE	12,288
[34]	Virtex-6 (0.22μm)	577	2460 LE	73,728
[37]	Virtex-6 (0.22μm)	353	11346 LE	0
[36]	Virtex-6 (0.22μm)	294	7056 LE	0
[35]	FLEX 10KE-1 (0.22μm)	451	2530 LE	98,304
[35]	ACEX 1K-1 (0.22μm)	212	2923 LE	49,152
[35]	APEX 20KE-1 (0.18μm)	612	2493 LE	102,400
[38]	Virtex-6 (0.22μm)	250	480 LE	32,768
Our Work (Case V)	EP4CE30F29C6	575	**2059 LE**	-

et al. in [33] and McMillan et al. in [38] utilized minimal amount of LEs (a total of 444 LEs and 480 LEs) but both architectures utilized hard-coded memory blocks with 12,288 bits and 32,768 bits respectively.

Another architecture reported in [56] Case A required 1818 LEs without involving LUTs. However, our architecture is able to achieve approximately 3 times the throughput performance with just a slight addition of LEs. Various pipelined architectures were proposed in [34], [36], [37], [56] in order to achieve throughput enhancement. It is worth noting that our implementations performance level is on par with the architectures presented in [56] Case C and [34] but with much smaller hardware circuit.

The above results can verify that our proposed work is able to meet the stringent requirement of lightweight implementation of AES cipher in terms of the low hardware area occupancy. Furthermore, our compact cipher also offers high throughput performance, which is also highly demanded in lightweight applications.

VI. CONCLUSION AND FUTURE WORK

This study presented a detailed overview of the various design of AES S-boxes, which are classified as the hard-coded LUT S-box, the pure combinatorial S-box using composite field arithmetic (CFA), the pipelined version of CFA S-Box, the AES S-box using direct computation and Linear Feedback Shift Register (LFSR) based S-Box. As a result, using various S-box architectures, six full AES cipher (Case I-VI) implementations on FPGA were derived and evaluated in details in terms of hardware cost and speed performance.

Based on the synthesis result, the AES encryption from Case V, which employed both architectural and algorithmic optimization, was proven to have low hardware area occupancy and high computational speed. Thus, it is able to fulfill the demand of optimum lightweight cipher for high speed and area-constrained applications.

For future work, we shall look deeper into the algorithmic optimization in the finite field sub-circuit. The recent work reported by Ueno et. al has proven that finite field arithmetic using different field and basis representations will have a great impact in the computation gate counts and its critical path [21]. In addition, another work by Boyar et al. has presented a new technique for combinatorial logic optimization which effectively simplifies the operations in the subfield $GF(2^4)$. Referring to these results, we shall extend our work by employing computation optimization in $GF(2^4)$ inversion and multiplications in order to achieve further hardware cost reduction.

REFERENCES

[1] T. Sivakumar and R. Venkatesan, "A novel approach for image encryption using dynamic SCAN pattern," *IAENG International Journal of Computer Science*, vol. 41, no. 2, pp. 91–101, 2014.

[2] R. E. Boriga, A. C. Dascalescu, and A. V. Diaconu, "A new fast image encryption scheme based on 2D chaotic maps,," *IAENG International Journal of Computer Science*, vol. 41, no. 4, pp. 249–258, 2014.

[3] F. Riaz, S. Hameed, I. Shafi, R. Kausar, and A. Ahmed, *Enhanced Image Encryption Techniques Using Modified Advanced Encryption Standard*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 385–396. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-28962-0-37

[4] "Advanced encryption standard (AES)," *National Institute and Technology of Standards NIST FIPS PUB 197*, 2001.

[5] X. Zhang, N. Wu, G. Yan, and L. Dong, "Hardware implementation of compact AES S-box," *IAENG International Journal of Computer Science*, vol. 42, no. 2, pp. 125–131, 2015.

[6] L. Fu, X. Shen, L. Zhu, and J. Wang, "A low-cost UHF RFID tag chip with AES cryptography engine," *Sec. and Commun. Netw.*, vol. 7, no. 2, pp. 365–375, Feb. 2014. [Online]. Available: http://dx.doi.org/10.1002/sec.723

[7] S. Banik, A. Bogdanov, and F. Regazzoni, *Exploring Energy Efficiency of Lightweight Block Ciphers*. Cham: Springer International Publishing, 2016, pp. 178–194. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-31301-6-10

[8] J. Daemen and V. Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*, ser. Information Security and Cryptography. Springer Berlin Heidelberg, 2002. [Online]. Available: https://books.google.com.my/books?id=tfjd6icCUoYC

[9] F. Opritoiu, M. Vladutiu, L. Prodan, and M. Udrescu, "A high-speed aes architecture implementation," in *Proceedings of the 7th ACM international conference on Computing frontiers*, ser. CF '10. New York, NY, USA: ACM, 2010, pp. 95–96. [Online]. Available: http://doi.acm.org/10.1145/1787275.1787300

[10] S.-Y. Lin and C.-T. Huang, "A high-throughput low-power aes cipher for network applications," in *Design Automation Conference, 2007. ASP-DAC '07. Asia and South Pacific*, 2007, pp. 595–600.

[11] X. Zhang and K. K. Parhi, "High-speed VLSI architectures for the AES algorithm," *IEEE Trans. VLSI Syst.*, vol. 12, no. 9, pp. 957–967, 2004.

[12] R. Liu and K. K. Parhi, "Fast composite field S-box architectures for Advanced Encryption Standard," in *GLSVLSI '08: Proceedings of the 18th ACM Great Lakes symposium on VLSI*. New York, NY, USA: ACM, 2008, pp. 65–70.

[13] A. Hodjat and I. Verbauwhede, "Area-throughput trade-offs for fully pipelined 30 to 70 Gbits/s AES processors," *IEEE Trans. Comput.*, vol. 55, no. 4, pp. 366–372, 2006.

[14] T. Good and M. Benaissa, "Very small FPGA application-specific instruction processor for AES," *IEEE Trans. Circuits Syst. I*, vol. 53, no. 7, pp. 1477 – 1486, 2006.

[15] ——, "Pipelined AES on FPGA with support for feedback modes (in a multi-channel environment)," *ET Information Security*, vol. 1, no. 1, pp. 1 – 10, 2007.

[16] G. Gogniat, T. Wolf, W. Burleson, J.-P. Diguet, L. Bossuet, and R. Vaslin, "Reconfigurable hardware for high-security/ high-performance embedded systems: The SAFES perspective," *IEEE Trans. VLSI Syst.*, vol. 16, no. 2, pp. 144–155, 2008.

[17] W. N. Chelton and M. Benaissa, "Fast elliptic curve cryptography on FPGA," *IEEE Trans. VLSI Syst.*, vol. 16, no. 2, pp. 198–205, 2008.

[18] D. R. Wilkins, "Part III: Introduction to Galois Theory," in *World Wide Web - http://www.ercangurvit.com/abstractalgebr/galois.pdf*, 2000.

[19] J. L. Fan and C. Paar, "On efficient inversion in tower fields of characteristic two," in *Proc. IEEE ISIT*, 1997, p. 20.

[20] C. Paar, "Some remarks on efficient inversion in Finite Fields," in *Proc. IEEE ISIT*, 1995, pp. 5–8.

[21] R. Ueno, N. Homma, Y. Sugawara, Y. Nogami, and T. Aoki, *Highly Efficient $GF(2^8)$ Inversion Circuit Based on Redundant GF Arithmetic and Its Application to AES Design*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 63–80. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-48324-4-4

[22] J. YONG-SUNG, K. YOUNG-JIN, and L. DONG-HO, "A compact memory-free architecture for the aes algorithm using resource sharing methods." *Journal of Circuits, Systems & Computers*, vol. 19, no. 5, pp. 1109 – 1130, 2010.

[23] A. Rudra, P. K. Dubey, C. S. Jutla, V. Kumar, J. R. Rao, and P. Rohatgi, "Efficient Rijndael encryption implementation with composite field arithmetic," in *Proc. CHES*. Paris, France: Springer-Verlag, May 2001, pp. 171–184.

[24] K. Nekado, Y. Nogami, and K. Iokibe, *Very Short Critical Path Implementation of AES with Direct Logic Gates*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 51–68. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-34117-5-4

[25] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A compact Rijndael hardware architecture with S-box optimization," in *Proc. ASIACRYPT*. Gold Coast, Australia: Springer-Verlag, Dec. 2000, pp. 239–245.

[26] N. Mentens, L. Batinan, B. Preneeland, and I. Verbauwhede, "A systematic evaluation of compact hardware implementations for the Rijndael S-box," in *Proc. Topics in Cryptology - CT-RSA 2005*, vol. 3376/2005. San Francisco, CA: Springer Berlin / Heidelberg, Feb. 2005, pp. 323–333.

[27] D. Canright, "A very compact Rijndael S-box," Naval Postgraduate School, Tech. Rep. NPS-MA-04-001, 2005.

[28] X. Zhang and K. K. Parhi, "On the optimum constructions of composite field for the AES algorithm," *IEEE Trans. Circuits Syst. II*, vol. 53, no. 10, pp. 1153–1157, 2006.

[29] M. M. Wong, M. L. D. Wong, A. Nandi, and I. Hijazin, "Construction of optimum composite field architecture for compact high-throughput aes s-boxes," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 20, no. 6, pp. 1151–1155, 2012.

[30] M. Wong, M. Wong, A. Nandi, and I. Hijazin, "Composite field $gf(((2^2)^2)^2)$ advanced encryption standard (aes) s-box with algebraic normal form representation in the subfield inversion," *Circuits, Devices Systems, IET*, vol. 5, no. 6, pp. 471–476, November 2011.

[31] A. Moradi, A. Poschmann, S. Ling, C. Paar, and H. Wang, "Pushing the limits: A very compact and a threshold implementation of aes," in *Advances in Cryptology EUROCRYPT 2011*, ser. Lecture Notes in Computer Science, K. Paterson, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, vol. 6632, ch. 6, pp. 69–88. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-20465-46

[32] Y. Nogami, K. Nekado, T. Toyota, N. Hongo, and Y. Morikawa, "Mixed bases for efficient inversion in f((22)2)2 and conversion matrices of subbytes of aes," in *Proceedings of the 12th International Conference on Cryptographic Hardware and Embedded Systems*, ser. CHES'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 234–247. [Online]. Available: http://dl.acm.org/citation.cfm?id=1881511.1881532

[33] P. Chodowiec and K. Gaj, *Very Compact FPGA Implementation of the AES Algorithm*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 319–333. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-45238-6-26

[34] P. Chodowiec, K. Gaj, P. Bellows, and B. Schott, "Experimental testing of the gigabit ipsec-compliant implementations of rijndael and triple des using slaac-1v fpga accelerator board," in *Proceedings of the 4th International Conference on Information Security*, ser. ISC '01. London, UK, UK: Springer-Verlag, 2001, pp. 220–234. [Online]. Available: http://dl.acm.org/citation.cfm?id=648025.744360

[35] V. Fischer and M. Drutarovsk, "Two methods of rijndael implementation in reconfigurable hardware," in *Cryptographic Hardware and Embedded Systems - CHES 2001, Third International Workshop, Paris, France, May 14-16, 2001, Proceedings*, ser. Lecture Notes in Computer Science, vol. 2162. Springer, 2001, pp. 77–92.

[36] A. J. Elbirt, W. Yip, B. Chetwynd, and C. Paar, "An fpga-based performance evaluation of the aes block cipher candidate algorithm finalists," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 9, no. 4, pp. 545–557, Aug 2001.

[37] A. Dandalis, V. K. Prasanna, and J. D. Rolim, *A Comparative Study of Performance of AES Final Candidates Using FPGAs*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 125–140. [Online]. Available: http://dx.doi.org/10.1007/3-540-44499-8-9

[38] S. McMillan and C. Patterson, *JBits™ Implementations of the Advanced Encryption Standard (Rijndael)*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 162–171. [Online]. Available: http://dx.doi.org/10.1007/3-540-44687-7-17

[39] Y. Zeng, X. Zou, Z. Liu, and J. Lei, "A low-power Rijndael S-box based on pass transmission gate and composite field arithmetic," *Journal of Zhejiang University - Science A*, vol. 8, no. 10, pp. 1553–1559, Oct. 2007.

[40] M. Kim, J. Ryou, Y. Choi, and J. Sungik, "Low power AES hardware architecture for radio frequency identification," in *IWSEC 2006*. Springer-Verlag Berlin Heidelberg, Oct. 2006, pp. 353–363.

[41] Z. Liu, Y. Zeng, X. Zou, Y. Han, and Y. Chen, "A high-security and low-power aes s-box full-custom design for wireless sensor network," in *2007 International Conference on Wireless Communications, Networking and Mobile Computing*, Sept 2007, pp. 2499–2502.

[42] J. Zhang, Q. Zuo, and J. Zhang, "Reducing the power consumption of the AES S-Box by SSC," in *International Conference on Wireless Communications, Networking and Mobile Computing*, 2007, pp. 2226–2229.

[43] G. Bertoni, M. Macchetti, L. Negri, and P. Fragneto, "Power-efficient asic synthesis of cryptographic sboxes," in *Proceedings of the 14th ACM Great Lakes Symposium on VLSI*, ser. GLSVLSI '04. New York, NY, USA: ACM, 2004, pp. 277–281. [Online]. Available: http://doi.acm.org/10.1145/988952.989019

[44] S. Tillich, M. Feldhofer, T. Popp, and J. Großschädl, "Area, delay, and power characteristics of standard-cell implementations of the aes s-box," *J. Signal Process. Syst.*, vol. 50, no. 2, pp. 251–261, feb 2008. [Online]. Available: http://dx.doi.org/10.1007/s11265-007-0158-2

[45] L. Gaspar, M. Drutarovsky, V. Fischer, and N. Bochard, "Efficient aes s-boxes implementation for non-volatile fpgas," in *Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on*, Aug 2009, pp. 649–653.

[46] S. Das, "Halka: A lightweight, software friendly block cipher using

ultra-lightweight 8-bit s-box," *IACR Cryptology ePrint Archive*, vol. 2014, p. 110, 2014.
[47] ——, "Ultra-lightweight 8-bit multiplicative inverse based s-box using LFSR," *IACR Cryptology ePrint Archive*, vol. 2014, p. 22, 2014. [Online]. Available: http://eprint.iacr.org/2014/022
[48] M. M. Wong and M. L. D. Wong, "New lightweight aes s-box using lfsr," in *Intelligent Signal Processing and Communication Systems (ISPACS), 2014 International Symposium on*, Dec 2014, pp. 115–120.
[49] ——, "A high throughput low power compact aes s-box implementation using composite field arithmetic and algebraic normal form representation," in *Quality Electronic Design (ASQED), 2010 2nd Asia Symposium on*, 2010, pp. 318–323.
[50] ——, "A new lightweight and high performance aes s-box using modular design," in *Circuits and Systems (ICCAS), 2013 IEEE International Conference on*, Sept 2013, pp. 65–70.
[51] M. M. Wong, M. L. D. Wong, C. Zhang, and I. Hijazin, "Compact and short critical path finite field inverter for cryptographic s-box," in *2015 IEEE International Conference on Digital Signal Processing (DSP)*, July 2015, pp. 775–779.
[52] V. Rijmen, "Effcient implementation of the Rijndael S-box," in *World Wide Web - http://ftp.comms.scitech.susx.ac.uk/fft/crypto/rijndael-sbox.pdf*.
[53] C. Paar, "Efficient VLSI architectures for bit-parallel computation in Galois Fields," Ph.D. dissertation, University of Essen, Germany, 1994.
[54] J. Wolkerstorfer, E. Oswald, and M. Lamberger, "An ASIC implementation of the AES S-boxes," in *Proc. RSA Conf.* San Jose, CA: Springer-Verlag, Feb. 2002, pp. 67–78.
[55] S. Morioka and A. Satoh, "An optimized s-box circuit architecture for low power AES design," in *Revised Papers from the 4th International Workshop on Cryptographic Hardware and Embedded Systems*, ser. CHES '02. London, UK, UK: Springer-Verlag, 2003, pp. 172–186. [Online]. Available: http://dl.acm.org/citation.cfm?id=648255.752730
[56] J. J. Tay, M. M. Wong, and I. Hijazin, "Compact and low power aes block cipher using lightweight key expansion mechanism and optimal number of s-boxes," in *Intelligent Signal Processing and Communication Systems (ISPACS), 2014 International Symposium on*, Dec 2014, pp. 108–114.
[57] P. Hamalainen, T. Alho, M. Hannikainen, and T. D. Hamalainen, "Design and implementation of low-area and low-power aes encryption hardware core," in *9th EUROMICRO Conference on Digital System Design (DSD'06)*, 2006, pp. 577–583.

M. M. Wong received her B.E. degree in Computer Systems Engineering from Curtin University of Technology (Sarawak Campus) in 2008. After her first degree, she joined Faculty of Engineering, Computing and Science (FECS) at Swinburne University of Technology (Sarawak Campus) as a Ph.D. research student. She completed her Ph.D in July 2012 and appointed as a lecturer in the same institution until 2017. She is currently a research fellow in Nanyang Technological University, Singapore. Her research interests include VLSI design for cryptology application, digital signal processing and hardware description and implementation.

M. L. D. Wong received his BEng(Hons) in Electronic and Communication Engineering and PhD in Signal Processing from the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, UK. In 2004, Dr. Wong joined the School of Engineering, Swinburne University of Technology Sarawak Campus as a Lecturer. Subsequently, he was appointed a Senior Lecturer in 2007 at the same institution. From 2011 to mid 2012, he was an Associate Professor at Xian Jiaotong Liverpool University, Suzhou, China. From June 2012 to Sept 2016, he was the Dean, Faculty of Engineering, Computing and Science at Swinburne Sarawak. Since October 2016, Professor Wong has been appointed as a Professor and Deputy Provost for Heriot-Watt University Malaysia. Professor Wong is also a Fellow of IET, a Fellow and Chartered Professional Engineer with IEAust, a Chartered Engineer with Engineering Council UK; and a Senior Member of IEEE. In 2014, he was electecd as the inaugural treasurer of the IEEE CIS Malaysia Chapter and in 2015 he was elected as the Vice Chair for the chapter. In 2016, he was elected as the inaugural chair for IEEE Malaysia Sarawak Subsection. His research interests include statistical signal processing and pattern classification, machine condition monitoring, and VLSI for digital signal processing.

C. Zhang received the B.Eng. degree in Computer Engineering from Tsinghua University, China, in 1982 and Ph.D. degree in Electrical Engineering from Newcastle University, Australia, in 1990. He was an Electrician with Changxindian (February Seven) Locomotive Manufactory, Beijing, China during 1970-1978, carried out research on control systems during 1983-1985 at Delft University of Technology, the Netherlands, was with the Department of Electrical and Electronic Engineering at the University of Melbourne, Australia, as a Lecturer, Senior Lecturer, and Associate Professor and Reader in 1989-2002 and with the School of Electrical and Electronic Engineering and School of Chemical and Biomedical Engineering during 2002-2010 at Nanyang Technological University, Singapore. Since 2010, he has been the Professor of Electrical and Electronic Engineering with the School of Software and Electrical Engineering at Swinburne University of Technology in Melbourne, Australia. His research interests include control, signal processing and medical imaging.

I. Hijazin is the Quality Education and Academic Accreditation director of the Faculty of Science , Engineering and Technology at Swinburne University of Technology in Melbourne Australia. In addition, he is also the Program Coordinator for Electrical and Electronic Engineering. He completed his undergraduate and postgraduate studies at Bradley University, Peoria Illinois all in Electrical Engineering. Ismat Hijazin lectures and supervises students in Electrical Engineering at Swinburne. His research interests include error control coding, VLSI digital signal processing and hardware description and implementation.