*ISSN (e): **2250 – 3005** || Volume, 07 || Issue, 09|| September – 2017 ||*
International Journal of Computational Engineering Research (IJCER)

Single- and Multi-core Configurable AES Architectures for Flexible Security

[1]biswaranjan Behera, [2]sushree Saloni Sahoo
Gandhi Institute of Excellent Technocrats, Bhubaneswar, India
Kruttika Institute of Technical Education, Khordha, Odisha, India

ABSTRACT:
As networking technology advances, the gap between network bandwidth and network processing power widens. Information security issues add to the need for developing high-performance network processing hardware, particularly that for real-time processing of cryptographic algorithms. This paper presents a configurable architecture for Advanced Encryption Standard (AES) encryption, whose major building blocks are a group of AES processors. Each AES processor provides block cipher schemes with a novel on-the-fly key expansion design for the original AES algorithm and an extended AES algorithm. In this multicore architecture, the memory controller of each AES processor is designed for the maximum overlapping between data transfer and encryption, reducing interrupt handling load of the host processor. This design can be applied to high-speed systems since its independent data paths greatly reduces the input/output bandwidth problem. A test chip has been fabricated for the AES architecture, using a standard 0.25-μm CMOS process. It has a silicon area of 6.29 mm , containing about 200,500 logic gates, and runs at a 66-MHz clock. In electronic codebook (ECB) and cipher-block chaining (CBC) cipher modes, the throughput rates are 844.9, 704, and 603.4 Mb/s for 128-, 192-, and 256-b keys, respectively. In order to achieve 1-Gb/s throughput (including overhead) at the worst case, we design a multicore architecture containing three AES processors with 0.18-μm CMOS process. The throughput rate of the architecture is between 1.29 and 3.75 Gb/s at 102 MHz. The architecture performs encryption and decryption of large data with 128-b key in CBC mode using on-the-fly key generation and composite field S-box, making it more cost effective (with better thousand-gate/gigabit-per-second ratio) than conventional methods.
Index Terms—Advanced Encryption Standard (AES), configurability, cryptography, encryption, hardware design, multicore architecture, network security.

I. INTRODUCTION

Applications such as electronic transaction and audio/video communication require not only significant network bandwidth but also high security measures [1], [2]. Security processing is computation intensive, which normally includes lookup and fetching/updating of parameters (keys, encryption/authentication algorithms, initial values, and security-related protocol information), encryption and authentication, data transfer, bus contention resolution, etc. Powerful security processing architectures are thus important in high-speed network applications. Some network security designs have been reported

Manuscript received April 29, 2008; revised September 15, 2008. First published July 21, 2009; current version published March 24, 2010.
M.-Y. Wang, C.-P. Su, C.-L. Horng, and C.-W. Wu are with the Department of Electrical Engineering, National Tsing Hua University, Hsinchu 30013, Taiwan.
C.-T. Huang is with the Department of Computer Science, National Tsing Hua University, Hsinchu 30013, Taiwan.
Digital Object Identifier 10.1109/TVLSI.2009.2013231

recently [3], including a network processor (NP) that offloads cryptographic algorithms into a security accelerator, a security coprocessor that handles secure socket layer or IP security header processing and cryptographic algorithms, and an in-line security processor that integrates a packet processing engine and

multiple cryptographic engines. Another recent design method is an NP that contains multiple cryptographic engines
[4].
The symmetric block cipher, Rijndael, is standardized by the National Institute of Standards and Technology as the Advanced Encryption Standard (AES) [5], which is a replacement to the original Data Encryption Standard [6]. It has widely been used intheInternetandwirelesscommunications[7]–[9].Parallelism is a commonpractice for improving hardware performance [10], [11]. Pipelined implementations were often exploited for AES in the past [12]–[14]. In [15], parallel cores and pipelining are used to enhance the throughput of AES. However, for block cipher with feedback, such as cipher-block chaining (CBC) mode, pipelining is limited. A parallel architecture composed of N encryption blocks for several symmetric encryption algorithms is reported in [16], where N is the number of rounds of the block cipher.
In this paper, we propose the "**AES** with **T**sing **H**ua
Ex**T**ended**I**mplicit **C**onfigurability" (AESTHETIC) architecture. It supports the original AES algorithm and also provides the flexibility to configure the parameters of each of the four transforms defined in the AES algorithm. Security-related protocols usually specify critical parameter handshaking (such as the selection of compression algorithms, cryptographic algorithms, keys, and initial values) in the first step. Encryption and/or authentication are done next for sensitive information transmission. With AESTHETIC, the user can select randomly among many extended AES ciphers each time an encryption or decryption session is requested. Such a dynamic configurability reinforces the security in data communication. In our security processing architecture, we use multiple AESTHETIC processors to achieve high speed and enhance security level. In this architecture, each encryption block handles an independent packet encryption session. Shared control interface to host processor through the system bus simplifies management of the multicore architecture. Independent data path for each encryption block (input/output (I/O) first in–first outs and the AESTHETIC processor) ensures high-speed data encryption. In the three-core architecture, e.g., the throughput lies between 1.29 and 3.75 Gb/s, with a 102-MHz clock.
Because the proposed work is a configurable solution, we surveyed several similar works about configurable AES for reasonable comparison. We also list here three examples for the need of configurable solution as follows. The first example is the secure storage [17]. The authors use a diversified AES withdifferent irreducible polynomials to secure the storage. The second example is that, when standard AES is successfully attacked, other AES schemes can be immediately used to avoid the risk. The final example is that specific cipher schemes are used for secure communication between two parties with the same parameters, e.g., the Internet key exchange (IKE) protocol is used for setting up security parameters (cryptographic algorithms) on each side. If other parties do not support extra cipher schemes, they do not have enough knowledge to communicate with the parties supporting these cipher schemes, reducing the security risk. These configurable solutionscanalsoincreasethesecuritysinceadditionalvariables (e.g., configuration bits for selecting one among 2^{19} cipher schemes) is involved. This is similar to increasing key length for security enhancement.
The rest of this paper is organized as follows. Section II describes basic transforms and concepts for the enhanced AES algorithm. In Section III, we present the AESTHETIC architecture, containing I/O block converter, configuration logic, AESTHETIC engine, and novel on-the-fly key generator. In Section IV, we propose a multicore architecture for AESTHETIC and analyze its performance and memory requirement. The implementation results for the two architectures are presented in Section V. Finally, Section VI concludes this paper.

II. CONFIGURABLE AES (AESTHETIC)

The AES algorithm is a symmetric block cipher that processes a series of 128-b data blocks to be encrypted (decrypted), and produces a series of 128-b encrypted (decrypted) data blocks, with a key of 128, 192, or 256 b [5]. The algorithm consists of four transformations on the data block, which are organized in a 4 $\times$ 4 B array, called state. The four transformations are as follows.

1) SubBytes(): Each element of the state is an element in $\mathrm{GF}(2^8)$ with the irreducible polynomial $p(x)$. A nonlinear operation is applied to each element using an S-box (substitution table) [5]. A multiplicative inverse of the element in $\mathrm{GF}(2^8)$ is performed first, followed by an affine transform. The affine transform can be expressed as $b(y) = \mathrm{const}(x) + y \cdot f(x) \bmod (x^8+1)$, where y is the inverse of the element, and $f(x)$ and $\mathrm{const}(x)$ are two polynomials with a degree less than eight.
2) ShiftRows(): A cyclic shifting operation is done on rows of the state with different numbers of bytes (offsets).
3) MixColumns(): Multiplication of a four-term polynomial withafixedpolynomial $c(x)$ modulo (x^4+1) isperformed. Each column of the state is considered a four-term polynomial, with coefficients in $\mathrm{GF}(2^8)$.

4) AddRoundKey(): A bitwise XOR operation is done, which adds a round key to the state in each iteration, where the round keys are generated by the key expansion procedure. The AESTHETIC engine uses the same encryption/decryption and key expansion procedures as the original AES algorithm. However, the user can respectively select from sets of parameters for , and . Against

$p(x),\ f(x),\ \mathrm{const}(x)$

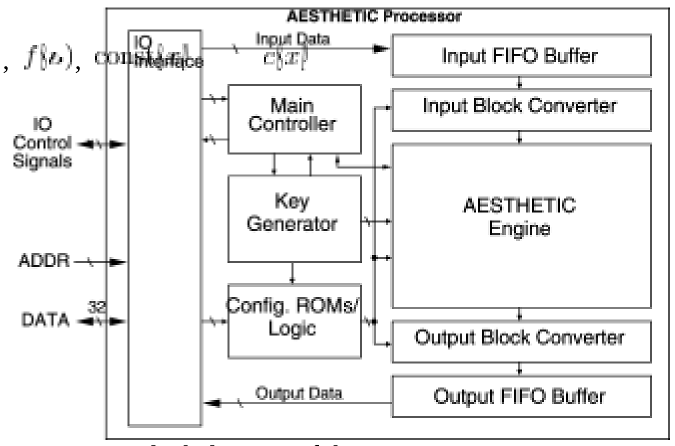


Fig. 1. Block diagram of the AESTHETIC processor.

existing attacks, several design criteria must be satisfied to ensure that the selected $\{p(x), f(x), \mathrm{const}(x)\}$ tuple can generate strong S-boxes. We have selected 16 irreducible polynomials of degree eight for . Each has 256 pairs of $\{f(x), \mathrm{const}(x)\}$, which are selected to provide sufficient algebraic complexity of the S-box. The matrix coefficients of MixColumns() transform $p(x)$ $c(x)$ are designed to provide strong diffusion power and guarantee that the relative inverse matrix exists. The coefficients of are also restricted to fall within the set . Under these constraints, we have selected 128 alternatives for . Therefore, there are totally types of block cipher that can be selected as the extended AES algorithm.

$c(x)$

$\{\{01\}_x, \{02\}_x, \ldots, \{0F\}_x\}$

$c(x)$

$16 \cdot 256 \cdot 128 = 2^{19}$

We need to design hardware for SubBytes() and MixColumns() with extended configurability, considering both hardware overhead and performance impact. Most S-box implementations use either the lookup table (LUT) approach [12]–[16], [18]–[23] or arithmetic computing [14], [24]–[29]. The trade-off between area and computation time for various S-box implementations has been evaluated in [30]. Using the ROM-based LUT approach to support as much as 2^{12}

kinds of S-boxes will consume unacceptable area. Another way is to use a 256 G 8 b SRAM to store the S-box values. Any change on S-box's parameters requires dedicated hardware to recompute all the values, which can lead to long configuration time. We use the arithmetic computing approach to implement the data path of the extended AES algorithm for smaller hardware overhead and easy configuration. The arithmetic computing approach results in a longer critical path than the LUT approach. The drawback, however, can be alleviated by the proposed architecture, as shown in Fig. 1, which implements the field multiplier or inverter with a dedicated irreducible polynomial. The data path can thus be optimized for this particular polynomial instead of using a general-purpose inverter or multiplier. Changing the irreducible polynomial $p(x)$ can be done easily by converting the data between the corresponding fields.

A. Composite $\mathrm{GF}(2^8)$ Field Arithmetic

Let the elements $(a_1x + a_0)$ in be represented as polynomials of degree one, i.e., , where $a_0, a_9 \in \mathrm{GF}(2^4)$. The inverse operation in the S-box can then be computed by taking the inverse of the polynomial with the irreducible polynomial , where $(a_1x + a_0)$ $P(x) = x^2 + Ax + B$ $A, B \in \mathrm{GF}(2^4)$ [29]. Although there are 30 irreducible polynomials in $\mathrm{GF}(2^8)$, only 16 of them are primitive. In order to perform basis transformation between $\mathrm{FF}(2^8)$ and $\mathrm{FF}((1^4)^2)$ (isomorphic mapping), a proper primitive polynomial should be

obtained. We

use the primitive polynomial $p_0(x) = x^8 + x^4 + x^3 + x^2 + 1$
(11D) to construct $\mathrm{GF}(2^8)$ for producing minimal critical path and the smallest area overhead [28], the polynomial $P(x) =$
$x^2 + x + \omega^{14}$ for $\mathrm{GF}((2^4)^2)$, where is the root of $x^4 + x + 1$,
and the irreducible polynomial $Q(x) = x^4 + x + 1$ for $\mathrm{GF}(2^4)$
according to [31], which shows an optimal multiplier using the composite field arithmetic. All the computation logic blocks related to the extended AES algorithm $\mathrm{GF}(2^8)$ are implemented using the composite field arithmetic in $\mathrm{GF}((2^4)^2)$.

B. Field Conversion

When using the data path with a fixed polynomial to perform an extended AES algorithm operating in a different polynomial, it requires data conversion at the I/O of the data path. The multiplication and addition defined in the extended AES algorithm can easily be done by the composite field data path with the help of a data conversion hardware. However, the affine transform is done on the prime field , which does not have an isomorphic operation defined in . Additional computation of the affine transform's parameters is required to fit into the composite field data path. We derive the isomorphic relation of the S-box with polynomials and as follows. Let be the transfer matrix $\mathrm{GF}(2)$ from the field with to the field with , and let be the transfer matrix from the field with the polynomial (11D) $\mathrm{GF}((2^4)^2)$ to the .
Thus, represents the transfer matrix from
with to . Given the field element , we can find its composite field representation . Assume that and is the selected polynomial $p(x)$ and constant $p_0(x)$ of the affine transform. T_β^γ The relation of the S-box with polynomial $p(x)$ and that in the composite field $\mathrm{GF}(2^8)$ can be expressed as $p(x)$

$$\mathrm{Sbox}_\beta(x_\beta) = T_A \cdot \mathrm{Inv}_\beta(x_\beta) + c_A$$

$p_0(x)$ T_α^γ $\mathrm{GF}(2^8)$

$$= T_A \cdot \left(\left(T_\beta^\gamma\right)^{-1} \mathrm{Inv}_\gamma \left(T_\beta^\gamma x_\beta\right)\right) + c_A$$

$T_\beta^\gamma = T_\alpha^\gamma \cdot T_\beta^\alpha$ $p_0(x)$ $\mathrm{GF}((2^4)^2)$ $\mathrm{GF}(2^8)$

where Inv_β is the inverse operation with polynomial $p(x)$ $\mathrm{GF}((2^4)^2)$ $x_\beta \in \mathrm{GF}(2^8)$. It can be substituted by operation in composite field with data conversion. From the aforementioned equation, we have $x_\gamma = T_\beta^\gamma x_\beta$

T_A c_A

$$\mathrm{Sbox}_\gamma(x_\gamma) = T_\beta^\gamma \cdot \mathrm{Sbox}_\beta(x_\beta)$$
$$= T_\beta^\gamma T_A \left(T_\beta^\gamma\right)^{-1} \cdot \mathrm{Inv}_\gamma(x_\gamma) + T_\beta^\gamma c_A.$$

The inverse of the S-box transform can be derived in a similar way, i.e.,

$$\mathrm{Sbox}_\gamma^{-1}(x_\gamma) = \mathrm{Inv}_\gamma \left(T_\beta^\gamma \cdot T_A^{-1}(x_\beta + c_A)\right)$$
$$= \mathrm{Inv}_\gamma \left(T_\beta^\gamma T_A^{-1} \left(T_\beta^\gamma\right)^{-1} x_\gamma + T_\beta^\gamma T_A^{-1} c_A\right).$$

Thus, $T_\beta^\gamma T_A (T_\beta^\gamma)^{-1}$, $T_\beta^\gamma c_A$, $T_\beta^\gamma T_A^{-1} (T_\beta^\gamma)^{-1}$, and $T_\beta^\gamma T_A^{-1} c_A$
must be computed beforehand. Only 16 irreducible polynomials are used in our design, and T_β^γ and $(T_\beta^\gamma)^{-1}$ are precomputed and stored in a small ROM. The parameters of the affine transform can be obtained within a short configuration time by using an 8×8 bitwise matrix multiplier.

III. AESTHETIC ARCHITECTURE

The AESTHETIC architecture is shown as a block diagram in Fig. 1, which supports both ECB and CBC cipher modes. The round keys for encryption and decryption are generated on the fly, without any internal memory. The reconfiguration between twoextendedAESalgorithmscanbedonewithinthreeclockcycles. All data access operations are manipulated by the I/O interface. The 32–128-b input first-in–first-out (FIFO) buffer caches the 32-b input data from the I/O interface to form a block of 128-b data, while the output FIFO buffer is used to cache the 128-b output block from the AESTHETIC engine. The I/O interface also consists of a user key (UK) register, an initial vector (IV) register, and a control register for use in configuring the AESTHETIC engine. The main controller is designed to enable the configuration procedure, start or halt the encryption/decryption procedure of the AESTHETIC engine, and also manage the data flow between the input buffer, the data path, and the output buffer. Details can be found in [29].

A. I/O Block Converter

The input data from the input buffer will be converted to the composite field representation in the input block converter, while the output block converter converts the data in $\mathrm{GF}((2^4)^2)$ back to $\mathrm{GF}(2^8)$.

B. Main Controller

The main controller controls the AESTHETIC engine, the key generator, and the I/O interface. In brief, it takes three clock cycles for configuration when the control register is set properly. Once the configuration is done, the first round key is stored in the AESTHETIC data path, and the key expansion procedure that generates the round keys will be launched immediately. After the last round key is generated and stored in the AESTHETIC data path, the following step is the encryption or decryption process. The AESTHETIC data path will write the encrypted/decrypted data to the output buffer and continue computation in the following clock cycle.

C. Configuration ROMs/Logics

Fig. 2 shows the configuration logic that generates the necessary coefficients for the AESTHETIC engine. There are

16 256 16 b ROMs in Affine_ROM. Each ROM stores the $f(x)$ $const(x)$ and parameters for a specific polynomial. The InvAffine Polynomial ROM is used to store the inverse polynomial of each . Either $f(x)$ or the inverse polynomial of will be selected as the output according to the state of the signal.

TMatrix_ROMstores $f(x)$ the transfer matrix T_β^γ and its inverse matrix , as discussed previously. The output signals and $f(x)$ represent the values of the T_β^γ

En_De

$(T_\beta^\gamma)^{-1}$

GF256to16 GF16to256

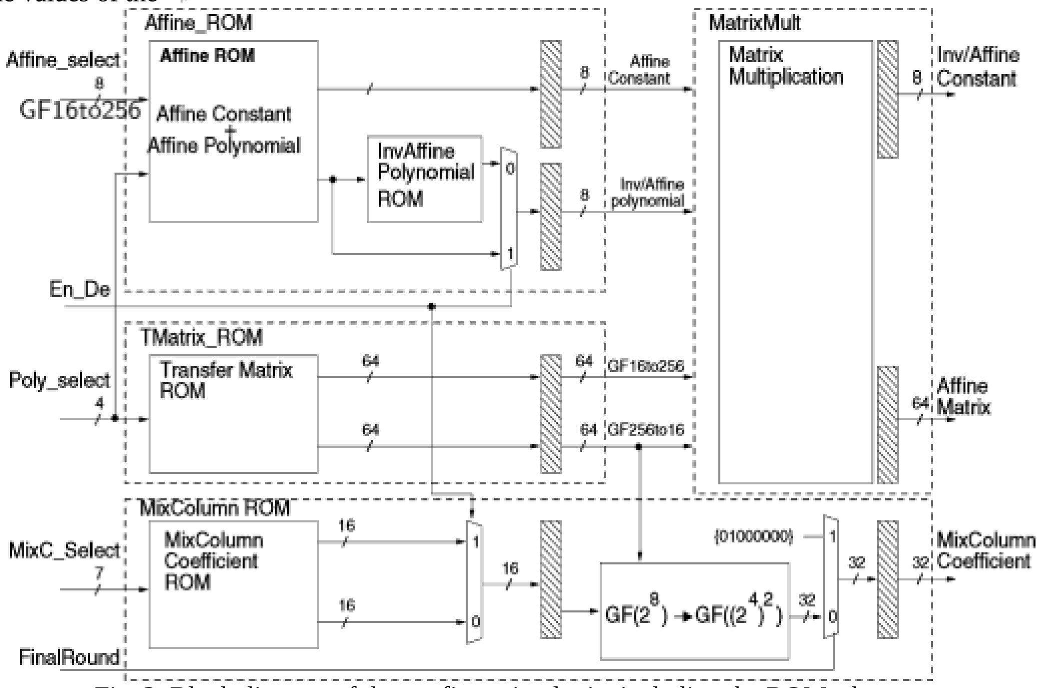


Fig. 2. Block diagram of the configuration logic, including the ROMs that store the coefficients of polynomials, affine transform and MixColumn, and a matrix multiplication circuit. The shaded boxes represent the pipeline registers.

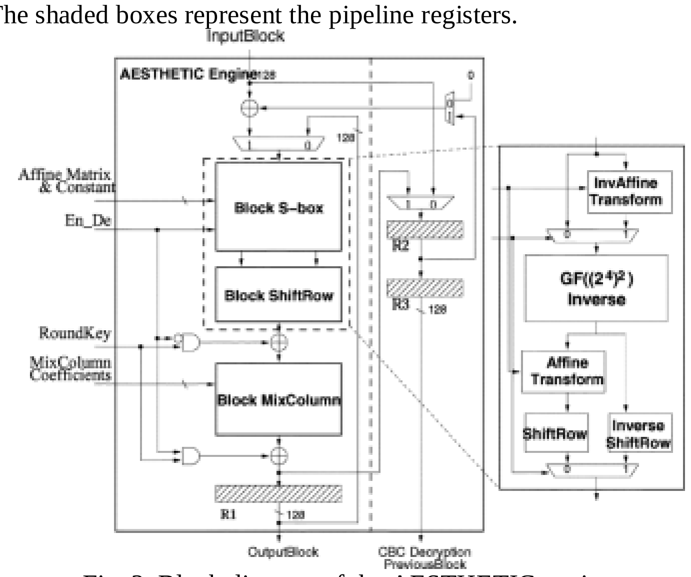


Fig. 3. Block diagram of the AESTHETIC engine.

and $(T_\beta^\gamma)^{-1}$ matrices, respectively. These signals are used by MatrixMultto compute either and or and .

The configurable coefficients of $T_\beta^\gamma T_A (T_\beta^\gamma)^{-1}$ MixColumns() $T_\beta^\gamma c$ transform are stored in MixColumn_ROM. Since the $T_\beta^\gamma T_A^{-1}(T_\beta^\gamma)^{-1}$ coefficients of the polynomial $T_\beta^\gamma T_A^{-1} c(x)$ are within the range, 16-b data is sufficient to store the coefficients of

$\{\{01\}_x, \{02\}_x, \ldots, \{0F\}_x\}$

MixC_select

$c(x)$. All the coefficients of MixColumns() and inverse MixColumns() transformations are selected by and multiplexed by . The selected coefficients are then converted to the composite field. Finally, a control signal, namely, , selects En_De between the converted coefficients and the $\{01000000\}_x$ value. Therefore, in the MixColumns() transform of the AESTHETIC engine, the data will FinalRound multiply with either the MixColumns matrix or a unit matrix to bypass the effect of MixColumns() transform. These parameters can be selected by (Poly_select, Affine_select, MixC_Select, and En_De), which are specified in the control register.

D. AESTHETIC Engine

Fig. 3 shows the block diagram of the AESTHETIC engine. It consists of four transforms in the extended AES algorithm and is used for both encryption and decryption. The Block S-box implements the S-box transform for a 128-b data block. The Sub-
Bytes and InvSubBytestransforms are implemented using one $GF((2^4)^2)$ inverse and two affine transform modules to eliminate the computation loop. The affine transform and inverse affinetransform circuits are identical;however,thefunctionsare different, represented by different input values of affine matrix and constants. ShiftRowsand InvShiftRowsperform the same functions, as defined in the original AES algorithm. Since the shift operations are done byte-wise, it makes no difference when implemented in $GF((2^4)^2)$.
Since all the coefficients of MixColumn transform are programmable, it requires a 4×4 matrix multiplication. Thus, in Block MixColumn, we implement 64 $GF((2^4)^2)$ multipliers to process the data block in parallel. The

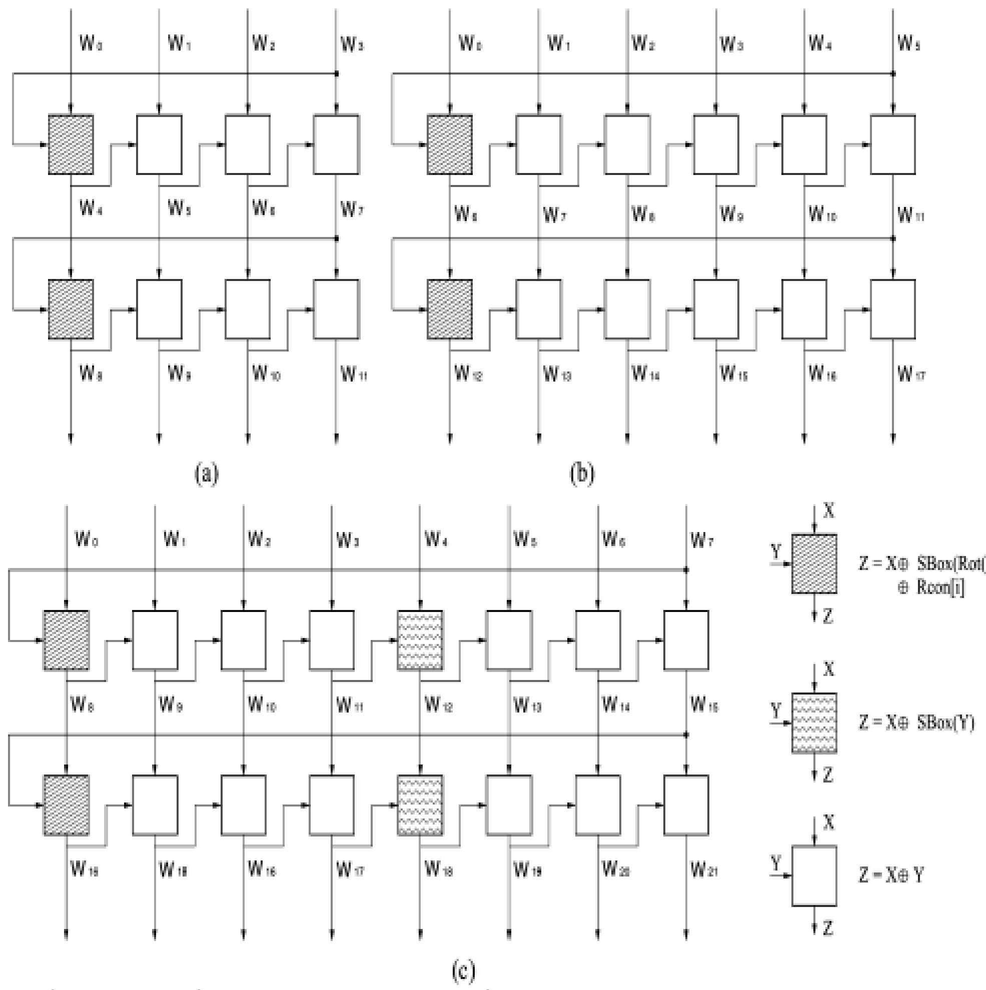


Fig. 4. Data flow graphs of key expansion procedure for (a) 128-b, (b) 192-b, and (c) 256-b keys, respectively.

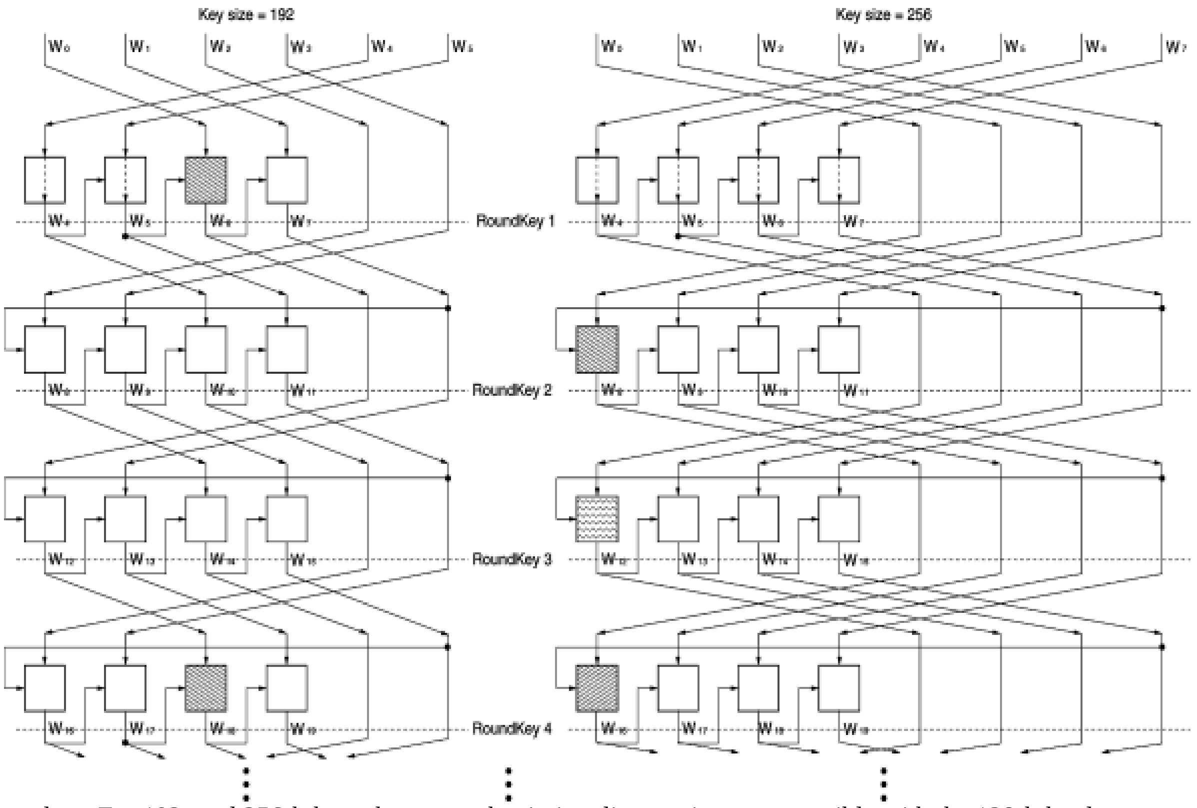
MixColumnand InvMixColumnfunctions can easily run on the same hardware by changing the coefficients according to the processing mode. In the final round of the AES algorithm where the MixColumns() transform is omitted, we configure the value of MixColumn coefficients to perform a unit matrix multiplication. Therefore, no extra hardware is required. The bit-wise AddRoundKey() transform is implemented by two groups of XOR gates that are placed before and after the Block MixColumnmodule, respectively. The En_De signal selects one of the two groups of XOR gates to determine the output data.

In order to balance the computation time in ECB and CBC cipher modes, our data path is designed to perform one round function per clock cycle. In Fig. 3, register R1 latches the output data in each round. The output will be fed back to the input of the data path for the next round of computation. Register R2 is used to support the CBC mode. In CBC mode, R2 holds the latest result of the encrypted data block. It will be XORed with the next input data block. In CBC decryption mode, R2 holds the latest input data block. Another register R3 is used to delay one clock cycle of R2's output, which will be XORed with the current decrypted block in the output block converter.

E. Key Generator

We propose a thee-in-one key generator that is associated with the AESTHETIC engine. Since the key expansion procedure is the same as the original AES algorithm, our design can also be applied to standard AES cipher. All the round keys are generated on the fly without additional memory for the subkeys. Our key generatorproduces one 128-b round key per clock cycle to fit our data path, regardless of the key size.

Fig. 4 shows the data flow graph for three different key sizes based on the key expansion procedure. It can be seen that generating a 128-b round key per clock cycle is straightforward using the 128-b key expansion



procedure. For 192- and 256-b keys, however, the timing diagram is not compatible with the 128-b hardware, so we rearrange the data flow graph for 192and 256-b keys such that only one 128-b key material is produced in each time frame. By properly shuffling the UK and the round keys, we can use only four computing elements to generate the 128-b round key. The result is shown in Fig. 5. The regularity in the data flow graph makes it easy to implement three different key expansion procedures by a unified hardware. For decryption, the key expansion procedure can be derived by reversing the computing order.

Fig. 6 shows the data path of the key generator module based on the rearranged data flow graph, where $R0, R1, \ldots, R7$ are

32-b registers to store the shuffled round keys. All the UKs are converted to $\mathrm{GF}((2^4)^2)$ and stored in $R0, R1, \ldots, R7$ according to the key size. Once the key generator is enabled, the data path will generate the round key by properly controlling the multiplexers in each clock cycle. The data shuffling multiplexer is used to swap the data in the registers and the generated round key iteratively. For 192-b key encryption, the output

Fig. 5. Rearranged data flow graph for 192- and 256-b key expansion.

$\mathrm{RoundKey}$ will be stored in , , $R2$ $R3$, and $R4$, while $R5$ the data in and will be stored in and to produce the next $R4$ round key $R5$ (see Fig. 5). Similarly $R0$, in $R1$ 256-b key encryption, will be stored in , , , and , while the data in , , , and will be swapped to , , , and .

$\mathrm{RoundKey}_i$ The S-box transform in this data $R4$ path $R5$ must $R6$ behave $R7$ in the same way as that in the AESTHETIC engine. Thus, the parameters $R4$ of $R5$ affine $R6$ transform $R7$ must be provided to $R0$ configure $R1$ $R2$ the S-box. All the round constants Rcon_i should also $R3$ be converted to $\mathrm{GF}((2^4)^2)$ before entering the key generator. In order to reduce the critical path and eliminate the computation loop, the output of the S-box is broadcast to each W_i instead of beingconnected to the zero input of mux_00 and mux_12 in Fig. 6. Therefore, additional XOR gates and multiplexers (such as mux_s0, mux_s1, mux_s2, and mux_s3) are used to provide equivalent mathematical relation.

IV. MULTI-AESTHETIC ARCHITECTURE

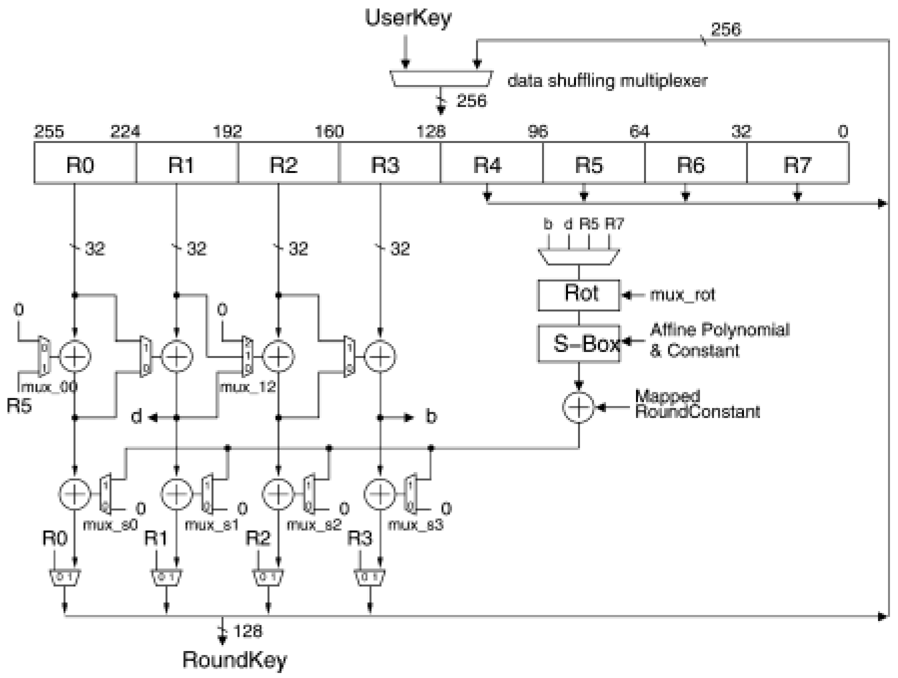
The multicore architecture is shown as a block diagram in Fig. 7. The host processor controls each AESTHETIC processor via an AHB-like interface. First, when the host processor has to performasecurityprocessingtask,itcheckswhichAESTHETIC processor is free with the resource control register of the shared module. If one or more AESTHETIC processors are available, it sets the corresponding bit of the register for the selected AESTHETIC processor. Otherwise, it waits. Second, it sets the descriptor pointer (Initial DescriptorPtr_Reg) for reading descriptors in the linked-list data structure (see Fig. 8), further reducing the interrupt handling load of the host processor. In addition, related registers (e.g., DataInPtr_Reg) are then updated when each descriptor in memory is indexed. In the third step, the host processorwritestheUK,theIV(forCBCmode),andtheconfiguration word to the UK, IV, and control registers, respectively. In the fourth step, each memory controller independently accesses data and result to avoid contention. The data access can be carried out simultaneously with 128-b block encryption (decryption) as well. In the final step, the main controller issues an interrupt to the host processor after an encryption (decryption) task is complete. The host processor then reads the interrupt status register, and it clears the interrupt and corresponding bit of the resource control register for the selected AESTHETIC processor.

In the multicore architecture, the critical path lies in the AESTHETIC engine in Fig. 1, not the control path. The design is good for multigigabit wire-speed applications by properly selecting the number of AESTHETIC processors. The area overhead is smaller than using multiple copies of the original AESTHETIC processors, thanks to shared control path.

A. Performance Analysis and Memory Requirement

For the multicore architecture, the number of clock cycles for data encryption (decryption) is $C_1 + C_2 \times N$, where C_1 is the fixed number of clock cycles, which is composed of items in Table I, while and are the encryption cost (e.g., $C_2 = 10$for 128-b keys) for one 128-b block and the number of 128-b blocks to be encrypted or decrypted C_2 N, respectively. In addition, bus contention overhead is considered if necessary.

zFig. 6. Data path of the key generator module.



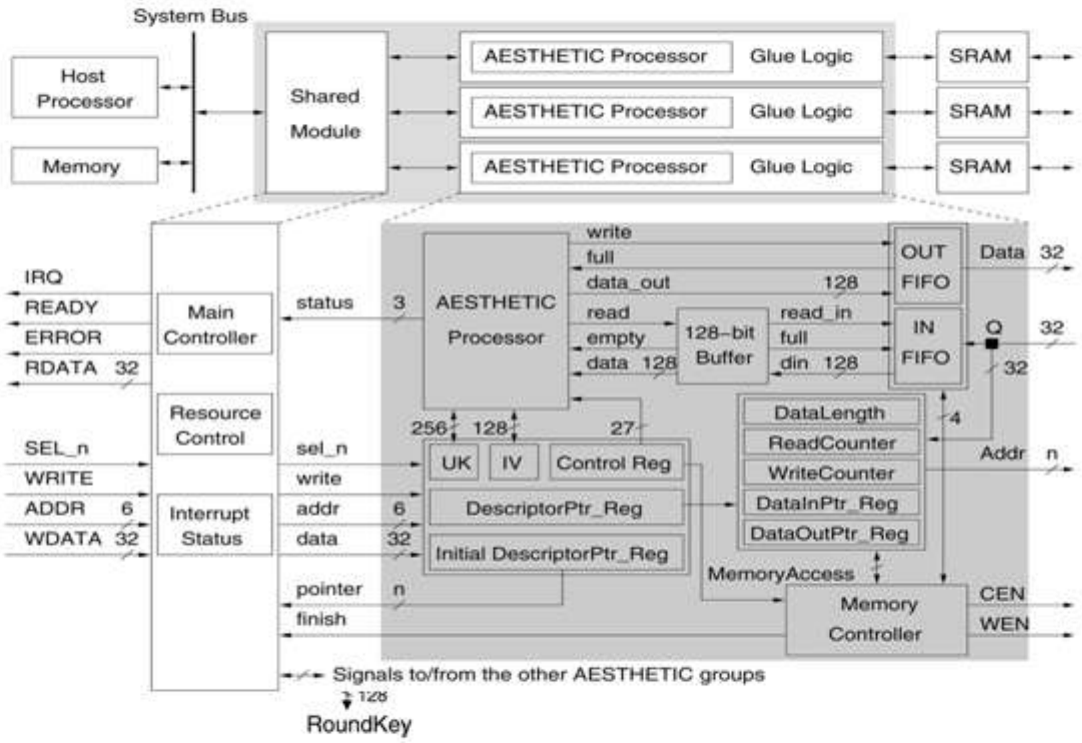


Fig. 7. Block diagram of the multi-AESTHETIC architecture.

To achieve the throughput of 1 Gb/s (with the fixed overhead C_1) in the worst case, we show a three-AESTHETIC architecture. The architecture has been implemented by using a standard 0.18-μm CMOS technology and a cell-based design flow, operating at 102 MHz with around 382,500 gates. The throughput equations and performance results of the three-AESTHETIC architecture are listed in Table II, where N_A is the number of AESTHETIC processors, N is the number of 128-b blocks to be encrypted or decrypted, and f is the operating frequency. After simple analysis for packet-size distribution in the worst

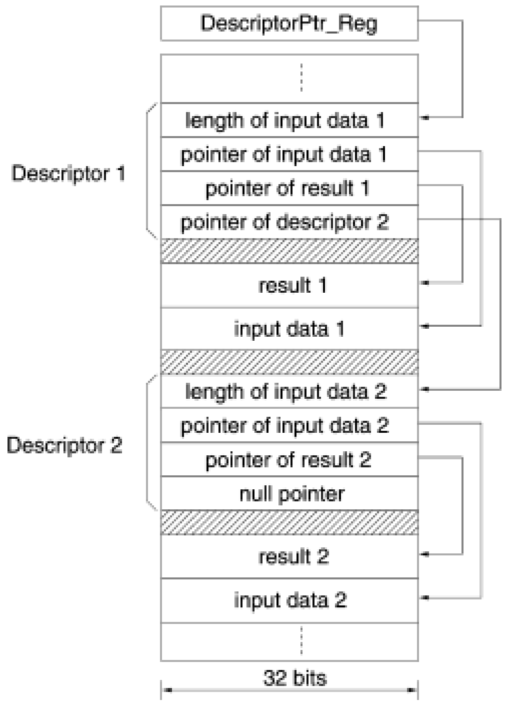


Fig. 8. Structure of I/O data.

TABLE I

FIXED NUMBER OF CLOCK CYCLES

Item	Cycle Count
IV	4
UK	4, 6, 8
Control	7
Read One Descriptor	4
Key Expansion	10, 12, 14
Read 128-bit Block	4
Internal Processing	8
Total	41, 45, 49 for three key sizes

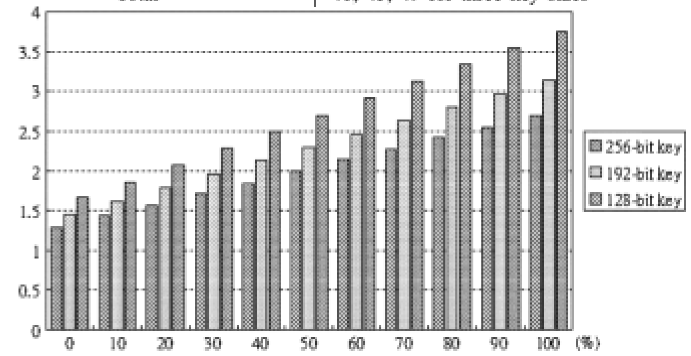


Fig. 9. Throughput(in gigabits per second) foramix of 46-and1500-Bpackets and three key sizes.

case (46-B packet) and the best case (1500-B packet), the performance lies between 1.29 and 3.75 Gb/s (see Fig. 9). For example, 90% in Fig. 9 means that 90% of the total packets are 1500-B packets $(N = 94)$, while 10% of them are 46-B packets $(N = 3)$. From Fig. 9, the fixed processing cost C_1 has a significant performance impact in the worst case. The minimum memory requirement for maximum performance in the three-AESTHETIC architecture equals

TABLE II

THROUGHPUT (IN GIGABITS PER SECOND) RESULTS FOR THREE AESTHETIC PROCESSORS

	128-bit Key	192-bit Key	256-bit Key
Throughput	$N_A \times \frac{128N}{41+10N} \times f$	$N_A \times \frac{128N}{45+12N} \times f$	$N_A \times \frac{128N}{49+14N} \times f$
1500-byte Data	3.75	3.14	2.7
46-byte Data	1.65	1.45	1.29

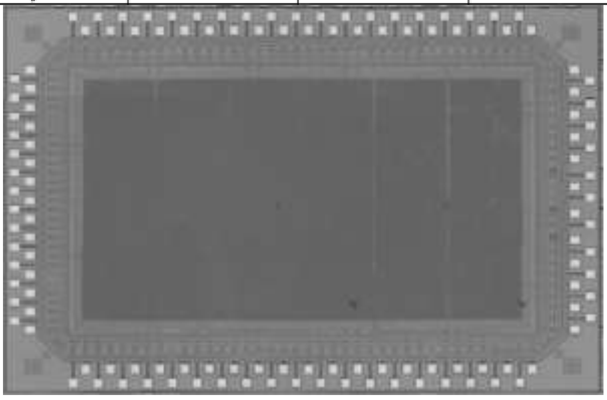


Fig. 10. AESTHETIC processor die photograph.

TABLE III

AREA STATISTICS OF THE AESTHETIC PROCESSOR

Module Name	Gate Count	%
I/O Interface	6121.3	3.05%
Input FIFO	1539.4	0.77%
Output FIFO	1774.3	0.88%
Main Controller	221.0	0.11%
Input Block Converter	9723.7	4.85%
Output Block Converter	4532.3	2.26%
AESTHETIC Engine	103523.3	51.63%
Key Generator	26639.3	13.3%
Configuration ROMs/Logics	49261.0	24.57%
Total	200501.0	100.00%

$2 \subset N \cdot 128(250N + 128)$; where 128 is the size of one deis the size of input data and result, and 2 means that there are two memory spaces available for the current and the following security tasks, i.e., successive cryptographic processing without waiting. For encrypting a 1500-B data in the three-AESTHETIC architecture, the total memory size is 142 kb.

V. IMPLEMENTATION RESULTS

The AESTHETIC chip was first fabricated. Then, we designed the multicore architecture based on the AESTHETIC processor. Therefore, two different technologies are used for the AESTHETIC chip and the multicore architecture.

A. AESTHETIC Chip

Thechipisfabricatedinamature0.25-μm CMOStechnology with five-level metallization. Fig. 10 shows a microphotograph of the processor whose core contains 88,000 instances with 200,500 equivalent gates, measuring 3.24 mm × 1.94 mm and 259.1 mW. It is implemented by standard cells. The gate counts of the modules in AESTHETIC are summarized in Table III. A full-scan test is implemented—there are 3906 test patterns that achieve 99.33% fault coverage. Our chip runs at 66 MHz, so the throughput values are 844.8 Mb/s for 128-b

TABLE IV COMPARISON OF AES DESIGNS

		Verbauwhede [23]	Mangard [26]	Satoh [24]	Su [28]	Jing [32]	Yen [33]	AESTHETIC Processor
Configurable		No	No	No	No	Yes	Yes	Yes
Technology		ASIC 0.18μm	ASIC 0.6μm	ASIC 0.11μm	ASIC 0.35μm	Altera FPGA EP1S40F1020C5	ASIC 0.18μm	ASIC 0.25μm
Clock Rate		125MHz	64MHz	224.22MHz	200MHz	290.87MHz	153.84MHz	66MHz
Throughput (Gbps)	128-bit key	1.6	0.241	2.21	2.381	36.36	1.7902	0.844
	192-bit key	1.33			2.008			0.704
	256-bit key	1.14			1.736			0.603
Gate Count		173K	15.493K	21.337K	58.430K	N/A	83.094K	200.5K
Gate Count/Throughput (Kgates/Gbps)		108.13 130.08 151.75	64.29	9.65	24.54 29.10 33.66	N/A	46.416	237.56 284.80 332.50
Parameters		-	-	-	-	FPGA	Off-chip	On-chip
# of Irreducible Polynomials		1	1	1	1	30	30	16
# of Affine Polynomials		1	1	1	1	$\prod_{i=0}^{7}(2^8 - 2^i)$‡	$\prod_{i=0}^{7}(2^8 - 2^i)$	256*
# of MixColumns Polynomials		1	1	1	1	10	Non-specified	128

*The number means 256 pairs of {affine polynomial f(x), constant const(x)}.
‡ The number of the constants is 255.

TABLE V COMPARISON FOR SINGLE AES (ASIC)

	Hodjat [14] (LUT S-box)	Hodjat [14] (Composite S-box)	Ours
Configurable	No	No	Yes
ASIC 0.18μm (Kgates)	175-380*	137.5-245*	127.5
Mode	ECB	ECB	ECB CBC
Gbps (ECB mode)	31.67-68*	33-66*	128: 1.25 192: 1.05 256: 0.9
Gbps (CBC mode)	2.05-4.36*	1.06-1.61*	128: 1.25 192: 1.05 256: 0.9
Kgates/Gbps (ECB mode)	3.83-7.81*	2.73-5.3*	128: 102 192: 121.43 256: 141.67
Kgates/Gbps (CBC mode)	57.62-117.35*	109.2-172.2*	128: 102 192: 121.43 256: 141.67

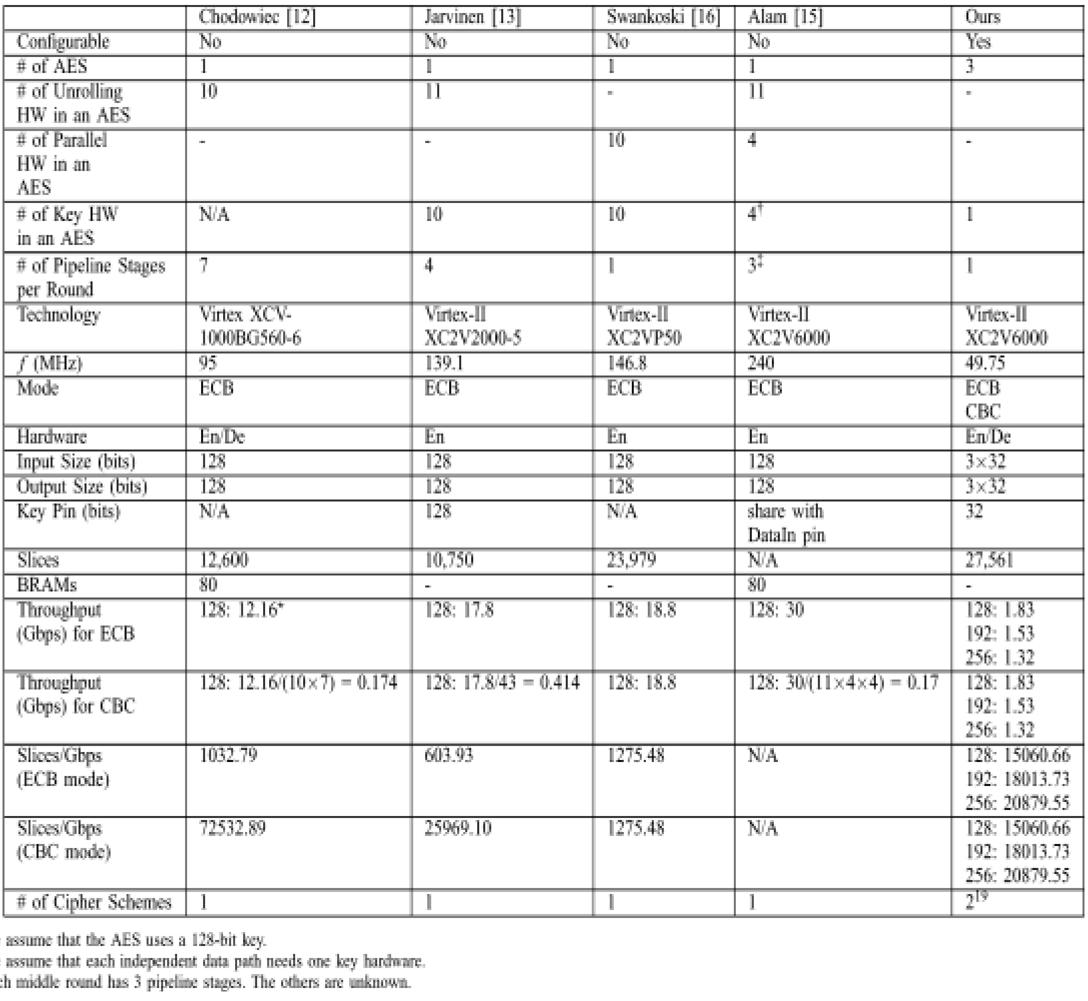
*Results are estimated from [14].

keys, 704 Mb/s for 192-b keys, and 603.4 Mb/s for 256-b keys, respectively. The comparison of our implementation with some other AES designs is shown in Table IV. Configurable designs have larger area than nonconfigurable ones, except [23]. In the table, the configurable AES designs with variable parameters are implemented by field-programmable gate array (FPGA) or application-specific integrated circuit [32], [33]. In [32], the authors use a function generator to produce a set of LUT contents for the LUT-based AES according to the selection of parameters. They provide huge parameter space for the configurable AES. However, the work does not report total area result of AES design including the function generator. In [33], the authors also provide high configurability, although the number of MixColumn polynomials is not clearly specified. Nevertheless, these parameters are stored in a lot of external memories instead of inside the AES chip. In other words, the approach increases additional cost and security risk, e.g., the attacker can obtain sensitive security parameters by monitoring the data between these memories and the AES chip [34]. Our AESTHETIC is designed with medium number of parameters on a single chip, and the user simply needs to select the AES cipher scheme to perform encryption. Moreover, the security risk can be avoided by using the parameter-embedded design with extra silicon cost (49,260 gates).

For the key generator, on-the-fly key generation has been proposed in [21], [26], and [35]. In [21], the architecture was designed to support 128-, 192-, and 256-b blocks with three different key sizes. Our design provides a more compact hardware dedicated for 128-b blocks, as is defined in standard AES. The key generators in [26] and [35] are designed for only 128-bkeys, but our key generator can be used for all three encryption and decryption key sizes, as defined in the standard.

B. Results of Multi-AESTHETIC Architecture

TABLE VI
COMPARISON OF PARALLEL AND PIPELINING FPGA IMPLEMENTATIONS OF AES



	Chodowiec [12]	Jarvinen [13]	Swankoski [16]	Alam [15]	Ours
Configurable	No	No	No	No	Yes
# of AES	1	1	1	1	3
# of Unrolling HW in an AES	10	11	-	11	-
# of Parallel HW in an AES	-	-	10	4	-
# of Key HW in an AES	N/A	10	10	4†	1
# of Pipeline Stages per Round	7	4	1	3‡	1
Technology	Virtex XCV-1000BG560-6	Virtex-II XC2V2000-5	Virtex-II XC2VP50	Virtex-II XC2V6000	Virtex-II XC2V6000
f (MHz)	95	139.1	146.8	240	49.75
Mode	ECB	ECB	ECB	ECB	ECB CBC
Hardware	En/De	En	En	En	En/De
Input Size (bits)	128	128	128	128	3×32
Output Size (bits)	128	128	128	128	3×32
Key Pin (bits)	N/A	128	N/A	share with DataIn pin	32
Slices	12,600	10,750	23,979	N/A	27,561
BRAMs	80	-	-	80	-
Throughput (Gbps) for ECB	128: 12.16*	128: 17.8	128: 18.8	128: 30	128: 1.83 192: 1.53 256: 1.32
Throughput (Gbps) for CBC	128: 12.16/(10×7) = 0.174	128: 17.8/43 = 0.414	128: 18.8	128: 30/(11×4×4) = 0.17	128: 1.83 192: 1.53 256: 1.32
Slices/Gbps (ECB mode)	1032.79	603.93	1275.48	N/A	128: 15060.66 192: 18013.73 256: 20879.55
Slices/Gbps (CBC mode)	72532.89	25969.10	1275.48	N/A	128: 15060.66 192: 18013.73 256: 20879.55
# of Cipher Schemes	1	1	1	1	2^{19}

* We assume that the AES uses a 128-bit key.
† We assume that each independent data path needs one key hardware.
‡ Each middle round has 3 pipeline stages. The others are unknown.

In order to increase throughput, pipelining or parallel techniques are often used in high-speed AES designs [12]–[16]. For ASIC designs [14], the comparison of our multicore implementation (processing 1500-B packets) with them is shown in Table V. For 128-b AES in CBC mode, using on-the-fly key generator and composite field S-box, our architecture obtains better thousand-gate/gigabit-per-second ratio. In addition, the three-AESTHETIC architecture operates at 102 MHz with 382,500 gates and consumes around 0.19 W. For FPGA designs [12], [13], [15], [16], the implementation results are shown in Table VI. These works use various numbers of hardware units or pipeline stages to enhance the total AES throughput. Most of them support 128-b AES encryption (En) in ECB mode only. In comparison with them, our design provides additional mode (CBC), decryption, and two key sizes (192 and 256 b). Our AES I/O sizes are smaller than others (128 b) as well. These works [12], [15], [16] exploit ROMs or FPGA block RAMs (BRAMs) that reduce the critical path. In [15], high-speed cells are used to implement the AES design for maximum performance. Also, onlythemaindataroundwasimplemented, so thisisnotthe performance for the entire AES computation. In addition, the maximum degree of parallelism and utilization rate are limited by encryption cycles of block cipher and I/O bandwidth [15], [16]. From Tables V and VI, we know that pipelining is inefficient for performing encryption in CBC mode without multiple round units [12]–[15]. The proposed configurable multicore architecture in CBC mode achieves medium slices/gigabit-per-second ratio. In particular, the architecture can be configured as one of 2^{19} cipher schemes.

In cryptography, CBC mode is better than ECB mode. Different key sizes imply multiple security choices as well. For key-agility applications, pipelined design with one key module is not enough and insecure. Hardware parallelism improves performance as well as security [16]. We can use native hardware redundancy to avoid faulty function induced by attack, material defects, or radiation. For high degree of security, our parameter-configurable design provides 2^{19} extended-security AES cipher schemes. Our multigigabit architecture with four security properties leads to higher security than other AES architectures.

VI. CONCLUSION

We have designed a configurable AES chip, called AESTHETIC, which enhances security over standard AES designs.

The implicit configurability allows the user to switch among
2^{19} AES-extended block ciphers. The chip supports ECB and CBC cipher modes with 128-, 192-, and 256-b keys. The maximum throughput is around 844.8 Mb/s for 128-b keys, 704 Mb/s for 192-b keys, and 603.4 Mb/s for 256-b keys under a 66-MHz clock. An on-the-fly key generator is also developed that provides exactly one 128-b round key per clock cycle. It can be used in not only the extended AES algorithm but also the original AES algorithm.

Based on our AESTHETIC processor (with configurable S-box and MixColumns transforms), we also propose a high-performance multicore architecture, where independent data path for each AESTHETIC processor provides multigigabit security processing without I/O bandwidth problem. The memory controller coordinates the maximum overlapping between data transfer and encryption (decryption) process. As an example, a three-AESTHETIC architecture has been designed to provide 1.28-Gb/s throughput in the worst case (256-b key and 46-B data). When it encrypts 1500-B packets with 128-b key, the maximum performance is 3.61 Gb/s.

REFERENCES

[1] S. Kent and R. Atkinson, Security architecture for the internet protocol IETF Netw. Working Group, RFC 2401, 1998 [Online]. Available: http://www.rfc-editor.org/RFCeditor.html

[2] A. Frier, P. Karlton, and P. Kocher, The SSL Protocol Version 3.0.Loudoun County, VA: Netscape, Nov. 1996.

[3] S. Stanley, “Security processors,” Feb. 2003 [Online]. Available: http:// www.lightreading.com/document.asp?doc_id=28307

[4] “Intel IXP2850 Network Processor Product Brief,” Intel Corp., Santa Clara, CA, 2003 [Online]. Available: http://www.intel.com/ [5] NIST, Springfield, VA, “Advanced Encryption Standard (AES),” Nov. 2001.

[6] NIST, Springfield, VA, “Data Encryption Standard (DES),” Oct. 1999.

[7] P. Chown, “Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS),” IETF Netw. Working Group, RFC 3268, 2002 [Online]. Available: http://www.rfc-editor.org/RFCeditor.html

[8] S. Frankel, R. Glenn, and S. Kelly, “The AES-CBC cipher algorithm and its use with IPsec,” IETF Netw. Working Group, RFC 3602, 2003[Online]. Available: http://www.rfc-editor.org/RFCeditor.html [9] IEEE 802.11i Standard, IEEE Std 802.11i]2004, Jul. 2004.

[10] J. L. Hennessy and D. A. Patterson, Computer Architecture: A Quantitative Approach, 2nd ed. San Francisco, CA: Morgan Kaufmann, 1996.

[11] D. Carlson, D. Brasili, A. Hughes, A. Jain, T. Kiszely, P. Kodandapani, A. Vardharajan, T. Xanthopoulos, and V. Yalala, “A high performance SSL IPSEC protocol aware security processor,” in Proc. IEEE Int. Solid-State Circuits Conf., San Francisco, CA, Feb. 2003, pp. 142–483.

[12] P. Chodowiec, P. Khuon, and K. Gaj, “Fast implementations of secret-key block ciphers using mixed inner- and outer-round pipelining,” in Proc. Int. Symp. Field Programmable Gate Arrays, Monterey, CA, Feb. 2001, pp. 94–102.

[13] K. U. Jarvinen, M. T. Tommiska, and J. O. Skytta, “A fully pipelined memoryless 17.8 Gbps AES-128 encryptor,” in Proc. Int. Symp. FPGA, Monterey, CA, 2003, pp. 207–215.

[14] A. Hodjat and I. Verbauwhede, “Minimum area cost for a 30 to 70 Gb/s AES processor,” in Proc. IEEE Comput. Soc. Annu. Symp., Lafayette, LA, Feb. 2004, pp. 83–88.

[15] M. Alam, W. Badawy, and G. Jullien, “A novel pipelined threads architecture for AES encryption algorithm,” in Proc. IEEE Int. Conf. Appl.-Specific Syst., Architectures, Process., San Jose, CA, Jul. 2002, pp. 296–302.

[16] E. J. Swankoski, R. R. Brooks, V. Narayanan, M. Kandemir, and M. J. Irwin, “A parallel architecture for secure FPGA symmetric encryption,” in Proc. 18th Int. Parallel Distrib. Process. Symp., Santa Fe, NM, Apr. 2004, p. 132.

[17] M.-H. Jing, S.-Y. Ko, and W.-C. Wu, “The SOC design of a highly secure and reliable storage using a conceptual environment,” in Proc. IEEE Asia-Pacific Conf. Circuits Syst., Tainan, Taiwan, Dec. 2004, pp. 865–868.

[18] V. Fischer and M. Drutarovsky, “Two methods of Rijndael implementation in reconfigurable hardware,” in Cryptographic Hardware and Embedded Systems (CHES) 2001. Berlin, Germany: Springer-Verlag, May 2001, vol. 2162, LNCS, pp. 77–92.

[19] S. McMillan and C. Patterson, “JBits implementations of the advanced encryption standard (Rijndael),” in Proc. 11th Int. Conf. FPL Appl., Aug. 2001, vol. 2147, LNCS, pp. 162–171.

[20] P. Chodowiec,K. Gaj, P.Bellows, andB. Schott, “Experimental testing of the gigabit IPSec-compliant implementations of Rijndael and triple DES using SLAAC-1V FPGA accelerator board,” in Proc. ISC, Oct. 2001, vol. 2200, LNCS, pp. 220–234.

[21] H. Kuo and I. Verbauwhede, , Ç K. Koç, D. Naccache, and C. Paar, Eds., “Architectural optimization for a 1.82 Gb/s VLSI implementation of the AES Rijndael algorithm,” in Cryptographic Hardware and Embedded Systems (CHES) 2001. Berlin, Germany: Springer-Verlag, May 2001, vol. 2162, LNCS.

[22] S. Morioka and A. Satoh, “A 10 Gbps full-AES crypto design with a twisted-BDD S-Box architecture,” in Proc. IEEE ICCD, Freiburg, Germany, Sep. 2002, pp. 98–103.

[23] I. Verbauwhede, P. Schaumont, and H. Kuo, “Design and performance testing of a 2.29-GB/s Rijndael processor,” IEEE J. Solid-State Circuits, vol. 38, no. 3, pp. 569–572, Mar. 2003.

[24] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, “A compact Rijndael hardware architecture with S-box optimization,” in ASIA CRYPT 2001. Berlin, Germany: Springer-Verlag, 2001, vol. 2248, LNCS, pp.239–254.

[25] J. Wolkerstorfer, E. Oswald, and M. Lamberger, “An ASIC implementation of the AES SBoxes,” in CT-RSA 2002.. Berlin, Germany: Springer-Verlag, 2002, vol. 2271, LNCS, pp. 67–78.

[26] S. Mangard, M. Aigner, and S. Dominikus, “A highly regular and scalable AES hardware architecture,” IEEE Trans. Comput., vol. 52, no. 4, pp. 483–491, Apr. 2003.

[27] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, “Unified hardware architecture for 128-bit block ciphers AES and Camellia,” in Cryptographic Hardware and Embedded Systems (CHES) 2003. Berlin, Germany: Springer-Verlag, Aug. 2003.

[28] C.-P. Su, T.-F. Lin, C.-T. Huang, and C.-W. Wu, “A high-throughput low-cost AES processor,” IEEE Commun. Mag., vol. 41, no. 12, pp. 86–91, Dec. 2003.

[29] C.-P. Su, C.-L. Horng, C.-T. Huang, and C.-W. Wu, “A configurable AES processor for enhanced security,” in Proc. ASP-DAC, Shanghai, China, Jan. 2005, pp. 361–366.

[30] U. Mayer, C. Oelsner, and T. Kohler, “Evaluation of different Rijndael implementations for high end servers,” in Proc. IEEE Int. Symp. Circuits Syst., May 2002, vol. 2, pp. 348–351.

[31] C. Paar, “A new architecture for a parallel finite field multiplier with low complexity based on composite fields,” IEEE Trans. Comput., vol. 45, no. 7, pp. 856–861, Jul. 1996.

[32] M.-H. Jing, Z.-H. Chen, J.-H. Chen, and Y.-H. Chen, “Reconfigurable system for high-speed and diversified AES using FPGA,” Microprocess. Microsyst., vol. 31, no. 2, pp. 94–102, Mar. 2007.

[33] C.-H. Yen, T.-Y. Pai, and B.-F. Wu, “The implementations of the reconfigurable Rijndael algorithm with throughput of 4.9 Gbps,” in Proc. 16th VLSI Des./CAD Symp., Hualien, Taiwan, Aug. 2005.

[34] M. Neve, E. Peeters, D. Samyde, and J.-J. Quisquater, “Memories: A survey of their secure uses in smart cards,” in Proc. IEEE Int. Security Storage Workshop, Washington, DC, Oct. 2003, pp. 62–72.

[35] J. H. Shim, D. W. Kim, Y. K. Kang, T. W. Kwon, and J. R. Choi,“A rijndaelcryptoprocessor using shared on-the-fly key scheduler,” in Proc. 3rd IEEE Asia-Pacific Conf. ASIC, Taipei, Taiwan, Aug. 2002, pp. 89–92.