A High-Throughput Low-Power AES Cipher for Network Applications

Shin-Yi Lin and Chih-Tsun Huang*
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan 30013

Abstract—We propose a full-featured high-throughput low-power AES cipher which is suitable for widespread network applications. Different modes of operation are implemented, i.e., the ECB, CBC, CTR and CCM modes. Our cipher utilizes a cost-efficient two-stage pipeline for the CCM mode by a single datapath. With the design-for-test circuitry, the maximum throughput is 4.27 Gbps using a 0.13μm CMOS technology with a 333MHz clock rate. The hardware cost is 86.2K gates with the power of 40.9mW.

I. Introduction

The rapidly growing number of high-speed network applications, such as multi-gigabit Internet, IEEE802.11 wireless networking, and high-speed serial link, etc., has led to ever increasing demand of high throughput security. Private data such as the identification, password, banking information, confidential document and personal information, has to be protected robustly to ensure the reliability of such kinds of applications, preventing security risks over public channels.

Two types of cryptographic systems, asymmetric (public-key) and symmetric (secret-key) cryptographies, have been developed for the data security. Asymmetric cryptography manipulates two different keys for encryption and decryption and provides a robust mechanism for the key transportation. On the other hand, symmetric cryptography uses an identical key for both encryption and decryption, which is more efficient for large amount of data.

Among many symmetric cryptographies, Advanced Encryption Standard (AES) [1] is an efficient scheme for both hardware and software implementation. In addition to AES encryption and decryption, there is an increasing demand for different modes of operation to further enhance the security level in the widespread network applications [2]. E.g., ECB (Electrical Code Book), CBC (Cipher Block Chaining), CTR (Counter), and CCM (Counter with CBC Message Authentication Code) [3] modes are the most common ones of operations modes.

*This work was supported in part by National Science Council, Taiwan, ROC, under contract NSC 94-2220-E-007-037 and Ministry of Economic Affairs (MOEA), Taiwan, ROC, under contract 94-EC-17-A-01-S1-038.

Most of the previous AES designs focused on its byte-substitution transformation, or S-Box, which is a byte-wise non-linear substitution. S-Box consists of a finite field (also Galois field, or GF) inversion in $GF(2^8)$ and an affine transformation. A straightforward look-up-table (LUT) can be used to implement the S-Box operation [4,5] LUT-based S-Box approaches required a large amount of area. An AES core with full encryption/decryption and key generation needs 36 LUTs: 16 for encryption; 16 for decryption; and another 4 for key generation. In [4], the encryption of 256-bit data blocks is supported, resulting in a total of 48 LUTs: 32 for encryption and 16 for key generation. In [5], a high-throughput programmable AES coprocessor was presented. The ECB, CBC, CTR and CCM modes were supported with its specific instruction set.

Many other AES alternatives implemented the GF inverter, which can be shared between encryption and decryption to reduce the area [6–11]. To prevent from the long timing overhead by using the GF inverter, the composite field technique was utilized, transferring the field operations from $GF(2^8)$ to $GF((2^4)^2)$, even to $GF(((2^2)^2)^2)$. GF-based approaches for the S-Box reduced the hardware cost effectively, but required additional basis conversion circuitry.

In this paper, we present a high-throughput low-power AES cipher core with full AES functions, i.e., AES encryption, decryption, and on-the-fly key generation with 128-, 192-, and 256-bit keys, supporting the ECB, CBC, CTR and CCM modes. We utilize a two-stage pipeline architecture to perform the two data flow of the CCM mode interleavedly inside a single AES datapath. The cost-efficient approach also allows interleaved execution of two separate data streams. The proposed cipher shares LUT-based $GF(2^8)$ inverters between encryption and decryption. Several retiming techniques are applied to further improve the critical path. Finally, the comprehensive AES cipher for networking has been implemented by using a 0.13μm CMOS technology with testability and power issues considered, achieving a high throughput/area ratio.

II. AES Cryptosystem

A. AES Algorithm

The AES cryptography is a block cipher that processes data blocks of 128 bits with a cipher key of 128, 192, or 256 bits.

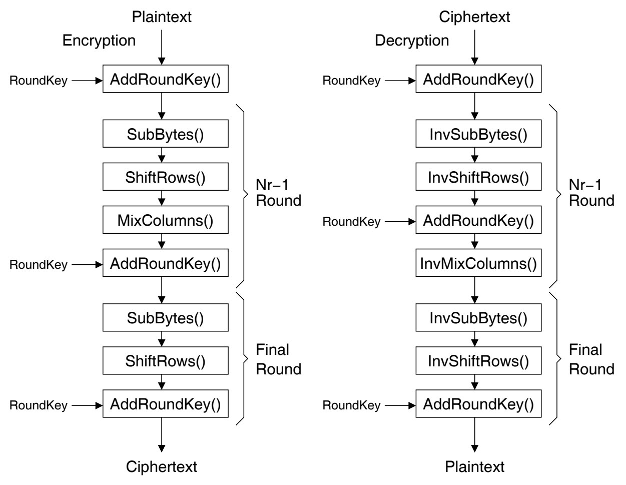


Fig. 1. The procedure of encryption and decryption.

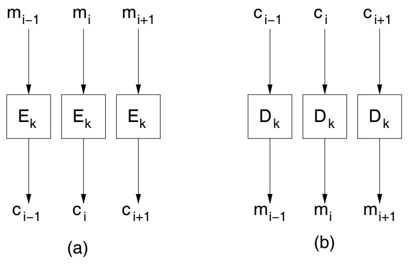

m_{i-1} m_i m_{i+1}
E_k E_k E_k
c_{i-1} c_i c_{i+1}
(a)
c_{i-1} c_i c_{i+1}
D_k D_k D_k
m_{i-1} m_i m_{i+1}
(b)

Fig. 2. (a) Encryption and (b) decryption of ECB mode.

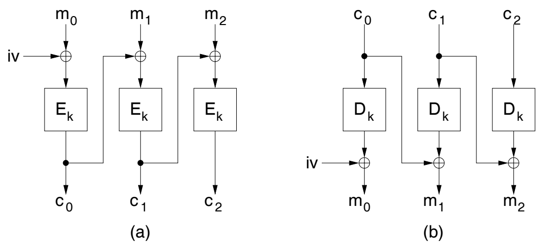


Fig. 3. (a) Encryption and (b) decryption of CBC mode.

A 128-bit data block can be treated as a 4×4 byte array, also known as the *State*. Figure 1 shows the encryption and decryption procedures in our AES cipher. The encryption and decryption have to be performed by N_r rounds, where $N_r = 10$, 12, or 14 for cipher key of length 128, 192, or 256 bits, respectively. An AES round function consists of four transformations operating on bytes, rows and columns on the State [1], which are briefly described as follows.

SubBytes(): a nonlinear byte-oriented substitution, also known as S-Box, consisting of a) a multiplicative inverse in finite field $GF(2^8)$ with irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$; and b) an affine transformation over $GF(2)$. The affine transformation can be done by a matrix multiplication and an addition.

ShiftRows(): a cyclic shifting on each row of the State with different number of byte offsets.

MixColumns(): mixing the bytes of each column by multiplying the four column polynomials of the State with a given polynomial modulo $x^4 + 1$ with their coefficient in $GF(2^8)$. The column polynomial is a polynomial of degree three with the four bytes in that column as its coefficients.

AddRoundKey(): an addition of a round key to the State. The round keys are generated by the key expansion procedure.

Key generation of the AES algorithm consists of the S-Box, AddRoundKey() and RotWord() transformations. The RotWord() performs cyclic shifting of a 4-byte word. Key generation and scheduling of the 192- and 256-bit keys is much more complicated than that of the 128-bit key scheme.

B. Common Modes of Operation

Many different modes of operation have been proposed to assist with the symmetric block cipher [2] for various applications. Our AES cipher can support the common ones, which are described as follows.

Fig. 2 shows the procedure of the ECB (Electronic Code Book) mode, which is the most straightforward operation mode. In the ECB mode, the $(i-1)^{th}$, i^{th}, and $(i+1)^{th}$ 128-bit data blocks are encrypted and decrypted individually, where m_i denotes the i^{th} message block, c_i is the i^{th} ciphertext block, E_k is the encryption function and D_k the decryption function. Because there is no dependency between the encryption of two consecutive blocks, they can be processed in parallel.

The CBC (Cipher Block Chaining) mode requires an initial vector (IV) as one input (see Fig. 3) to the encryption of the m_0. The c_0 then becomes the IV to the encryption of the m_1, and so on. The decryption procedure is a reverse of the encryption. Because of the dependency between consecutive blocks, pipelined AES datapath suffers for the throughput degradation.

The CTR (Counter) mode avoids the data dependency of the CBC mode. As shown in Fig. 4, the only value to be encrypted is the increasing counter sequence (ctr), instead of the message itself. The decryption of the CTR mode does not involve with the AES decryption, as long as the receiver can maintain the same counter sequence. The management of the counter sequence becomes crucial because the differential attack can be applied if one obtains two different ciphertext blocks with identical counter number and key.

To prevent the weakness of the CTR mode, CCM (Counter with CBC-MAC) mode was proposed to provide both encryption and authentication [3]. The CCM mode uses the CTR operation for the data encryption, and the CBC operation to produce the MAC (Message Authentication Code). Our cipher supports the CCM protocol defined in IEEE 802.11i. Two

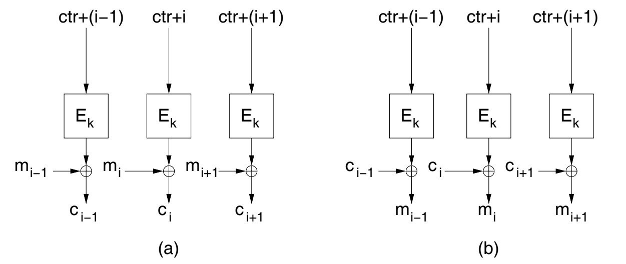


Fig. 4. (a) Encryption and (b) decryption of CTR mode.

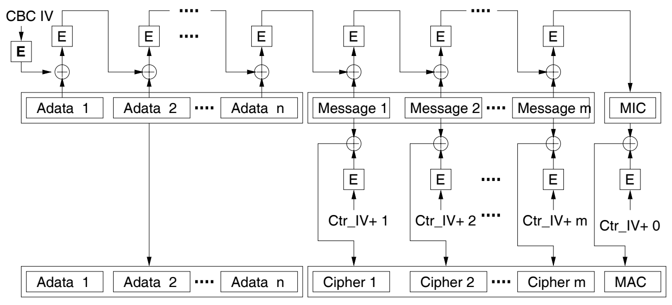


Fig. 5. The CCM encryption.

or more blocks, called *Adata*, are used as the authentication header. The Adata blocks, together with the message blocks are encrypted by the CBC operation to produce the MIC (Message Integration Code) (see the upper part of Fig. 5). Then the message blocks and the MIC are again encrypted by the CTR operation. After the encryption of the MIC, we can obtain the MAC block. For the decryption of the CCM, as shown in Fig. 6, the ciphertext blocks are decrypted by the CTR operation (see the lower part of the figure). The decrypted messages are further encrypted by the CBC operation, producing the MIC block (see the upper part of the figure). The MIC block from the CBC operation is compared with the decrypted MAC to guarantee the data integrity.

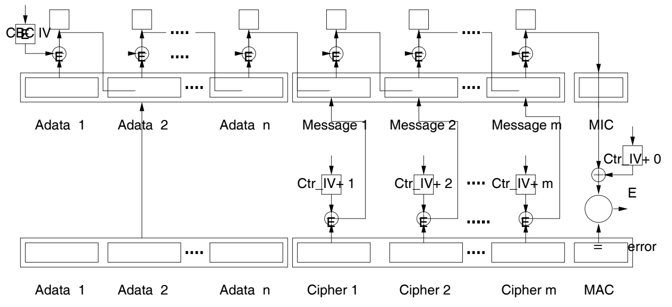


Fig. 6. The CCM decryption.

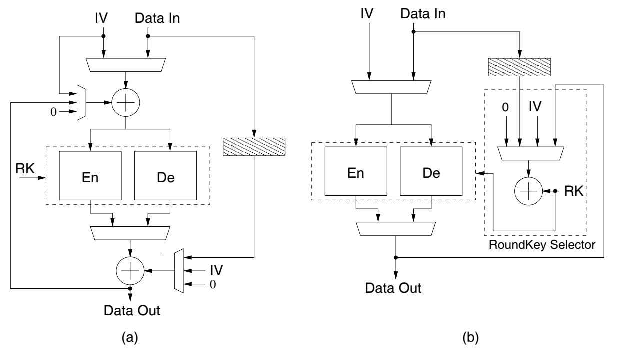


Fig. 7. Integrated AES encryption/decryption for the ECB, CBC, CTR, and CCM modes.

III. Improved Architecture

A. The Support of Various Modes

Table I summarizes the usage of AES encryption and decryption functions in various modes of operation. We can observe that the ECB and CBC modes require AES decryption function for the mode decryption, while the CTR and CCM modes uses AES encryption function for both mode encryption and decryption.

TABLE I
The AES usage in different operation modes.

	ECB	CBC	CTR	CCM
Mode Encryption	$E_k{}^1$	E_k	E_k	E_k
Mode Decryption	$D_k{}^2$	D_k	E_k	E_k

^{1}AES encryption function; ^{2}AES decryption function.

In our approach, the AES encryption and decryption functions are tightly coupled for the resource sharing. Fig. 7(a) shows an integrated AES datapath with both AES encryption and description functions. The multiplexers can be manipulated to perform the different modes of operation. In addition, the upper and lower XORs in Fig. 7(a) can be merged and retimed because the first and last transformations of AES encryption/decryption are both AddRoundKey(), resulting in the block diagram in Fig. 7(b). The improved diagram has reduced critical path and simpler mode control.

B. Improved Round Function and Its Datapath

The proposed AES datapath is improved from the GF-based architecture, which decomposes the S-Box to the multiplicative inversion and affine transformation. Fig. 8(a) illustrates a typical implementation for a round function. The upper data path performs the encryption while the lower path for the decryption. The $GF(2^8)$ inverter is shared. With a retiming technique, Fig. 8(b) shows that the InvAffine() in the first found

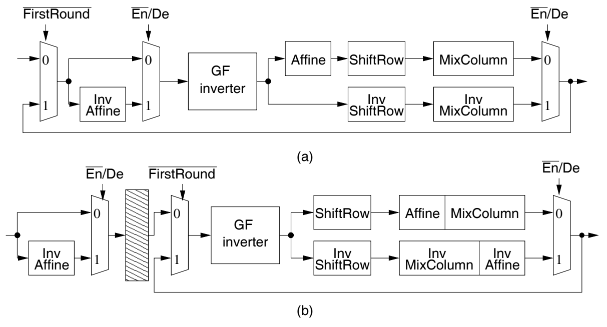


Fig. 8. (a) Original round function and (b) improved one with retiming technique.

function can be moved out from the round function. And inside the round function the InvMixColumn() and InvAffine() transformations, which are both matrix operations, can be merged as one. In addition, the Affine() and ShiftRow() can be exchanged because they are both linear. Thus Affine() and MixColumn() transformations can also be merged as one. Similar retiming technique is used in [8] which is a GF-based approach, however, it is more effective when the LUT-based $GF(2^8)$ inverters are used.

The datapath in [8] with previous retiming technique is shown in Fig. 9(a), where *A* denotes the Affine(), *IA* denotes the InvAffine(), *MC* represents the MixColumn() and *IMC* for InvMixColumn(). Note that in order to perform the composite field $GF((2^4)^2)$ inversion, the byte-level data has to be converted from $GF(2^8)$ to $GF((2^4)^2)$ (GF256/16). Afterward the composite field data has to be converted from $GF((2^4)^2)$ back to $GF(2^8)$ (GF16/256). In Fig. 9(b), the proposed datapath design uses an LUT-based $GF(2^8)$ inverter, which can be optimized by the commercial synthesis tool. The boldface blocks in Fig. 9(a) can be removed or simplified. Our synthesis experiment shows that the critical path of our LUT inverter can achieve 2ns or shorter, while the composite-field GF-based inverter stops at around 3ns. If a timing constraint of 8ns is applied to the entire datapath, the synthesis result of the entire AES core with encryption/decryption and key generation shows that the composite-field approach requires about 62.4K gates, while the hardware cost of our LUT approach is about 54K gates, by using a $0.18\mu m$ CMOS technology.

The retiming technique can be applied again to our LUT-based datapath. In Fig. 9(b), the rightmost XOR in Fig. 9(a) is moved backward along the encryption path, and the multiplexing to the three XORs in the round function can be simplified further.

C. Pipelined Architecture

Pipelined architecture is used to improve the throughput of the AES design [6, 10, 11]. However, the more the pipeline stages, the worse the utilization when the feedback operation mode, e.g., the CBC mode, is required. On the other hand, a straightforward approach to realize the CCM mode is to use two AES cores for the CBC and CTR operations, respectively, as in [5]. We present a cost-effective two-stage pipelined datapath, which can also perform two AES operations interleavedly. Although the CCM mode needs the AES encryption function only, our cipher with full-featured encryption/decryption provides the flexibility to process two separate data streams in parallel, which is suitable for the widespread high-speed network applications. The utilization will be 50% for the CBC mode with single data stream. But the datapath can still be fully utilized if two data streams can be manipulated interleavedly.

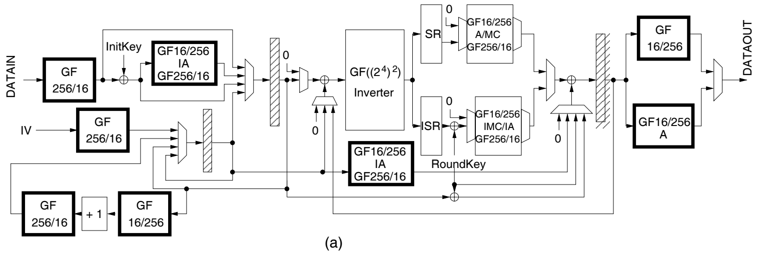
The datapath is partitioned at the output of the LUT-based GF inverter. The two stages can be well balanced because the LUT has more degree of freedom to synthesize than the GF-based operation does. The critical path of the first stage consists of a multiplexer, an XOR, and the GF inverter; while the critical path of the second stage contains ShiftRow(), Affine()+MixColumn(), three multiplexers and one XOR. The preliminary analysis using a $0.18\mu m$ technology shows that the path delay of the first stage is about 2.37ns, and the delay of the second stage is about 2.16ns. This timing balance also remains with the technology shrinking. Our analysis also indicates that our overall datapath delay is about 20% shorter than the delay in [8].

IV. Overall AES Cipher

The overall architecture of the proposed AES cipher is shown in Fig. 10. Our design is composed of three major parts: IO Interface, Input/Output FIFOs, and AES core. The interface follows the standard AHB (Advanced High-Performance) interface, which makes our core ready to be integrated. A slightly modification can be made to fit for other common interfaces. In addition to the 128-bit IV and 256-bit initial key (IK) registers, a 9-bit control (CTRL) register is used to select the encryption or decryption function; ECB, CBC, CTR or CCM mode; interleaved data flow or not; 128-, 192, or 256-bit key; enable or pause; the change of the IV; and the re-computation of the round key.

To support the two-stage pipeline architecture, input and output FIFOs are both 256 bits, i.e., they are capable of two data blocks. Larger FIFOs also work well in our design for different system environment.

The AES core is controlled by the Main Controller, which manages the datapath for the round functions, the Key Generator for the round keys, and also the Key Controller to schedule the round key properly. The key generation is also pipelined to fit the timing of the datapath. Before the data enters the AES datapath, the BlockInit module performs the mode operations with the IV and key, also the retimed InvAffine() transformation accordingly.



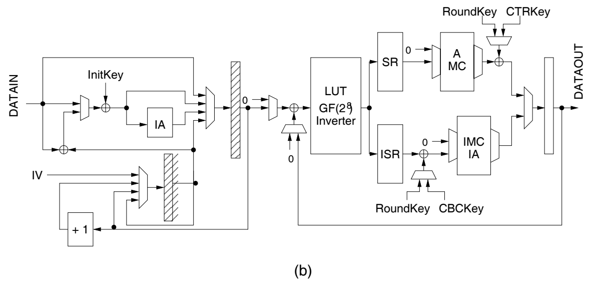


Fig. 9. (a) GF-based datapath and (b) our LUT-based datapath.

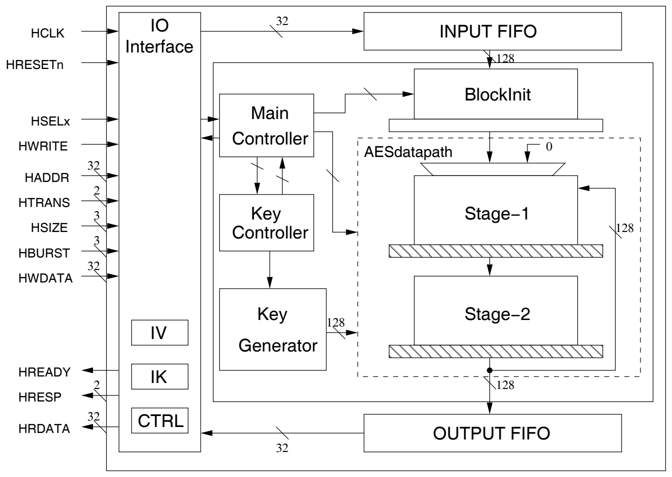


Fig. 10. The overall architecture of the proposed AES cipher.

V. Implementation Results

A. RTL Verification and FPGA Prototyping

A verification flow was applied before the realistic implementation. We constructed a random simulation flow for our AES design. The random generator produces the stimulus of random messages, different key sizes and operation modes. The RTL/netlist simulation results of our AES design are then compared with the golden responses of the C-based behavior model. More than half a billion patterns are verified during the development phase.

Besides the simulation, the design has also been implemented by an Altera FPGA board. The FPGA design contains about 13,260 logic elements, 68,480 memory bits and 89 pins, with maximum clock rate of 42MHz. In a simple prototype a desktop computer is used to transmit the encrypted video stream via an USB connection to the FPGA board. The data is decrypted by our AES cipher, and sent back to the computer for the playback.

B. ASIC Implementation

Our AES cipher has been implemented by using the 0.13μm CMOS technology. Figure 11 shows the layout of the chip which contains 97.8K gates and measures 1.825mm^2 (the core without pads measures 0.721mm^2 including the power rings) with about 89% core utilization. The full-scan design-for-testability was considered. The ATPG coverage is 98.38% with 4 scan chains and 212 patterns. The chip can run with the 125MHz clock rate under the post-layout simulation. This clock rate is limited by the speed of IO pads. Without the IO pads, the core can operate under 333MHz by the post-layout simulation. During the development, the power consumption was taken into consideration. Unnecessary blocks can be suspended functionally by the Main Controller. In addition, the commercial power synthesis tool was also used to optimize the power. The dynamic estimation shows that the power is 40.9mW after simulating about 90 random data blocks, each with encryption, decryption and all the operation modes.

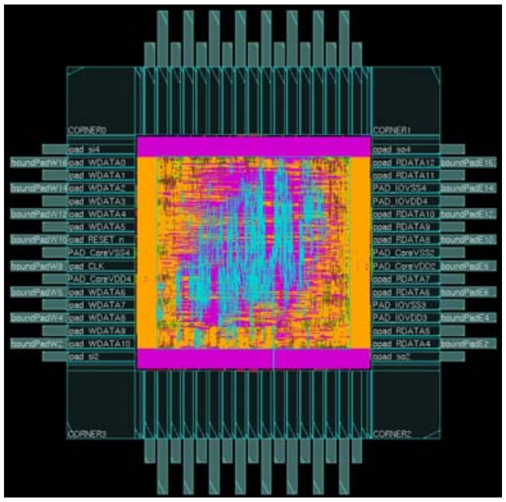


Fig. 11. Layout of our AES chip.

C. Analysis and Comparisons

Table II summarizes the comparison among different AES designs. The AES in [4] supports 128-, 192- and 256-bit data blocks, which needs 48 LUTs for the S-Box. Two pipelined AES designs are listed [6, 10], which only support the ECB mode. In [6] there are 4 pipeline stages. Different key sizes are supported without the on-the-fly key generation. The design in [10] is only capable of the 128-bit key. In addition, the AES designs with different modes of operation are also compared [5, 7, 8]. In [7] the composite field of $GF(((2^2)^2)^2)$ was used to reduce the hardware cost of the AES design with the key length of 128 bits. The ECB and CBC modes were also implemented. The small area leads to a very high throughput-area (T/A) ratio. In [5] a programmable AES coprocessor was presented with the throughput of 3.43Gbps and the clock rate of 295MHz. The ECB, CBC, CTR and CCM modes were implemented as well. Two AES encryption cores were implemented to realize the CCM mode. However, only 128-bit key was supported. The coprocessor can perform the CTR and CCM mode efficiently, but is limited for the ECB and CBC decryption. The AES cipher in [8] supported three different key sizes, on-the-fly key generator, ECB, CBC and CTR modes, but no CCM mode. Finally, our design can achieve the highest throughput of 4.27Gbps with a clock rate of 333MHz. The 128-, 192-, and 256-bit AES encryption, decryption, and on-the-fly key generator were implemented with the ECB, CBC, CTR and CCM modes. A two-stage pipelined datapath was used for the CCM mode. The area of the entire AES cipher is 97.8K gates. And the area of the AES core is 86.2K gates excluding the IO Interface and Input/Output FIFOs (see Fig. 10). The area was obtained by a timing-driven synthesis. Although our T/A ratio is the second highest, the proposed AES cipher is the most cost-efficient in terms of the throughput, functionally, and power.

The supply voltages of the $0.18\mu m$ and $0.13\mu m$ technologies are 1.8V and 1.2V, respectively. The voltage drop leads to a power reduction of about 56%. However, in spite of the power reduction by the technology shrinking, the 40.9mW power consumption of our AES cipher is still the most effective under the consideration of the functionality, clock rate, and architecture.

VI. Conclusions

We have presented a high-throughput low-power full-featured AES cipher with the ECB, CBC, CTR and CCM modes. Our AES cipher implements a two-stage pipeline architecture to perform the two data flows of the CCM mode efficiently by a single AES datapath. It also allows interleaved execution of two separate data streams. The AES datapath is area-efficient with the resource sharing of the LUT-based $GF(2^8)$ inverter, and the prevention of the basis conversion. Several retiming techniques were applied to further improve the critical path. The AES cipher has been implemented by using a $0.13\mu m$ CMOS technology with testability and power issues considered. The maximum throughput is 4.27 Gbps with a 333MHz clock. The hardware cost is about 86.2K gates ($0.721 mm^2$). The power consumption is about 40.9mW.

TABLE II
THE COMPARISON AMONG DIFFERENT AES DESIGNS.

	Kuo [4]	Su [6]	Lai [10]	Satoh [7]	Hodjat [5]	Horng [8]	Ours
Technology	0.18	0.35	0.25	0.11	0.18	0.18	0.13
Clock Rate (MHz)	154	200	125	224	295	125	333
Area (Gates)	173K	58.4K	80K	21.3K	73.2K	67.9K	86.2K
Throughput (Gbps)	1.60 1.33 1.14	2.381 2.008 1.736	1.454	2.61	3.43	1.60 1.33 1.14	4.27 3.56 3.05
Power (mW)	56	230.6	N/A	N/A	86	56	40.9
T/A* (Kbps/KGates)	9.25 7.69 6.59	40.77 34.38 29.73	18.2	122.28	46.9	23.56 19.63 16.83	49.54 41.30 35.38
Operation Modes	ECB	ECB	ECB	ECB CBC	ECB CBC CTR CCM	ECB CBC CTR	ECB CBC CTR CCM
Pipeline Stage(s)	1	4	6	1	1	1	2
Key Size	All	All	128	128	128	All	All
AES En/De	En	En/De	En/De	En/De	En	En/De	En/De

*Throughput/Area (Kbps/KGates) ratio.

References

[1] National Institute of Standards and Technology (NIST), *Advanced Encryption Standard (AES)*, National Technical Information Service, Springfield, VA 22161, Nov. 2001.

[2] M. Dworkin, “Recommendation for block cipher modes of operation”, Technical report, National Institute of Standards and Technology (NIST), Gaithersburg, MD, Dec. 2001, http://csrc.nist.gov/CryptoToolkit/modes/.

[3] M. Dworkin, “Recommendation for block cipher modes of operation: The CCM mode for authentication and confidentiality”, Technical report, National Institute of Standards and Technology (NIST), Gaithersburg, MD, May 2004, http://csrc.nist.gov/CryptoToolkit/modes/.

[4] I. Verbauwhede, P. Schaumont, and H. Kuo, “Design and performance testing of a 2.29 Gb/s Rijndael Processor”, *IEEE Jour. of Solid-State Circuits*, pp. 569–572, 2003.

[5] A. Hodjat, P. Schaumont, and I. Verbauwhede, “Architectural design feature of a programmable high throughput aes copressor”, in *Proc. IEEE Coding and Computing*, Oct. 2004.

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

[7] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, “A compact Rijndael hardware architecture with S-box optimization”, in *ASIACRYPT 2001*. 2001, vol. 2248 of *LNCS*, pp. 239–254, Springer-Verlag.

[8] C.-L. Horng, “An AES cipher chip design using on-the-fly key scheduler”, Master Thesis, Dept. Electrical Engineering, National Tsing Hua University, Hsinchu, Taiwan, June 2004.

[9] X. Zhang and K. Parhi, “High-speed VLSI architecture for the AES algorithm”, *IEEE Trans. on VLSI Systems*, vol. 12, no. 9, pp. 957–967, 2004.

[10] Y.-K. Lai, L.-C. Chang, L.-F. Chen, C.-C. Chou, and C.-W. Chiu, “A novel memoryless AES cipher architecture for networking applications”, in *Proc. IEEE Circuit and Systems Symp*, May 2004.

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