

June 24, 2002
RT0473
Security 6 pages

Research Report

A 10Gbps Full-AES Crypto Design with a Twisted-BDD S-Box Architecture

Sumio Morioka and Akashi Satoh

IBM Research, Tokyo Research Laboratory
IBM Japan, Ltd.
1623-14 Shimotsuruma, Yamato
Kanagawa 242-8502, Japan



Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

Limited Distribution Notice

This report has been submitted for publication outside of IBM and will be probably copyrighted if accepted. It has been issued as a Research Report for early dissemination of its contents. In view of the expected transfer of copyright to an outside publisher, its distribution outside IBM prior to publication should be limited to peer communications and specific requests. After outside publication, requests should be filled only by reprints or copies of the article legally obtained (for example, by payment of royalties).

A 10 Gbps Full-AES Crypto Design with a Twisted-BDD S-Box Architecture

Sumio Morioka and Akashi Satoh
Tokyo Research Laboratory, IBM Japan Ltd.,
1623-14 Shimotsuruma, Yamato-shi, Kanagawa-ken 242-8502, Japan
{E02716,akashi}@jp.ibm.com

Abstract

In this paper, we present a high-speed AES IP-core, which runs at 780 MHz on a 0.13 μm CMOS standard cell library, and which achieves 10 Gbps throughput in all encryption modes, including CBC mode. Although the CBC mode is the most widely used and important, achieving such high throughput was difficult because pipelining techniques cannot be applied. To reduce the propagation delays of the S-Box, the most critical function block, we developed a special circuit architecture that we call twisted-BDD, where the fanout of signals is distributed in the S-Box circuit. Our S-Box is 1.5 to 2 times faster than the conventional S-Box implementations. The T-Box algorithm, which merges the S-Box and another primitive function (MixColumns) into a single function, is also used for an additional speedup.

1 Introduction

DES (Data Encryption Standard) has been used as a de facto standard cipher for more than 20 years. In 2001, NIST (National Institute of Standards and Technology) made Rijndael the new standard cipher AES (Advanced Encryption Standard) [1,2]. Many circuit architectures for AES have been proposed recently and their performances have been evaluated by using ASIC libraries [3,4,5] and FPGAs [6,7]. However, most of them are simple implementations according to the AES specification, and none are yet fast enough for practical use such as optical communication links with a VPN (Virtual Private Network) capability and/or a 10 Gbps WDM (Wavelength Division Multiplex) system.

In particular, no existing AES circuit achieves 10 Gbps throughput in the CBC (Cipher Block Chaining) mode, which is the most widely used and important mode, although more than 10 Gbps throughput was already reported in simple ECB (Electronic Code Block) mode [3]. In the CBC mode, a feedback operation is performed, and therefore pipelining techniques cannot be applied as a speedup method. Propagation delay reduction for the combinational circuits is the only speedup method for the CBC mode.

To reduce the delay of the S-Box, which is the slowest

function block in the entire circuit, we investigated various design techniques for the S-Box. Although many techniques for compact S-Box designs have been proposed [8,9,10], the circuits obtained are too slow. Few techniques for realizing fast S-Boxes have ever reported except for the table-lookup method [11], where the S-Box circuit is automatically synthesized from its truth table by using EDA tools.

In this paper, we propose a new fast S-Box circuit architecture named twisted-BDD. In the conventional BDD (Binary Decision Diagram) [12] representation of the S-Box, various structural characteristics are observed, such as the heavy sharing of input-side nodes and the independence of variable ordering. We tried to decrease the fanout of the primary input signals and reduce the propagation delay of the serially connected selectors. The resulting S-Box is 1.5 to 2 times faster than the conventional S-Box implementations.

As a result, we achieved a throughput of 10 Gbps at a 780 MHz clock rate using a 0.13 μm CMOS standard cell library, by combining the use of the twisted-BDD architecture and the T-Box algorithm [1,7] that was originally developed for high-speed software. As far as the authors know, this is the first 10 Gbps AES circuit which can support all encryption modes.

2 The AES Algorithm

2.1 Basic Algorithm

An AES encryption process for 128-bit plain text data and a 128-bit secret key is shown in Figure 1. A sequence of four primitive functions, SubBytes, ShiftRows, MixColumns, and AddRoundKey, are executed $N_r - 1$ times. Each loop is called a *round* and the concrete value of N_r is 10, 12, or 14 depending on the key length. Prior to this main loop, AddRoundKey is executed for initialization. After executing the main loop, a sequence of SubBytes, ShiftRows, and AddRoundKey is executed as the final round.

SubBytes is a 16-byte (128-bit) input/output nonlinear transformation that uses one-byte substitution tables (*S-Boxes*). Each S-Box is a multiplicative inversion on a Galois field $\text{GF}(2^8)$ followed by an affine transformation.

The irreducible polynomial used by the field is

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

ShiftRows is a cyclic shift operation in each row of 4×4 -byte data by 0~3-byte offsets. MixColumns treats the 4-byte data in each column as coefficients of a 4-term polynomial, and multiplies the data modulo x^4+1 with the fixed polynomial given by

$$c(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}. \quad (2)$$

AddRoundKey is a simple bit-wise XOR operation on the 128-bit round keys $K_0 \sim K_{Nr}$ and the data.

In the decryption process, the inverse operations of each primitive function are executed. The inverse of AddRoundKey is AddRoundKey itself. InvSubBytes, which is the inverse of SubBytes, executes an affine transformation before the multiplicative inversion. InvShiftRows is a cyclic rotation in the reverse direction. InvMixColumns uses the following polynomial for the multiplications:

$$c^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}. \quad (3)$$

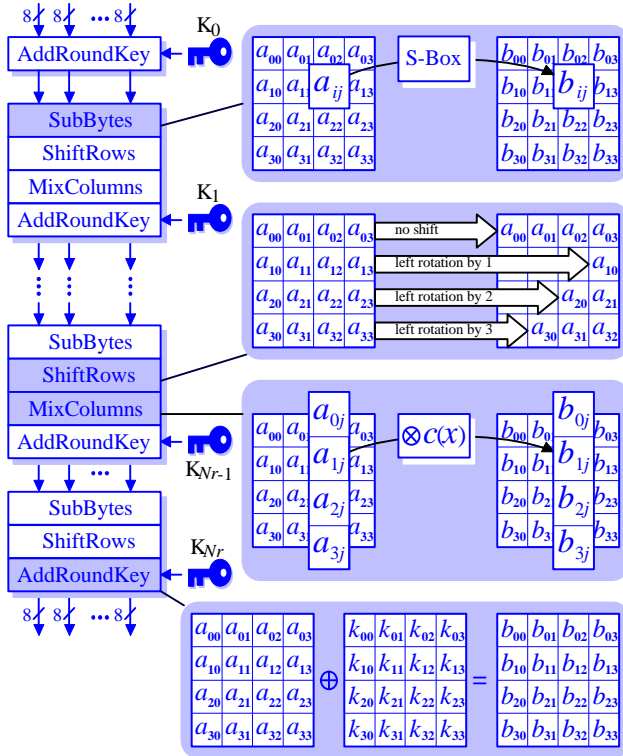


Figure 1. AES Encryption Algorithm.

2.2 T-Box for High-Speed Implementations

The T-Box algorithm is a speedup technique for software where SubBytes and MixColumns are merged in

encryption, and InvSubBytes and InvMixColumns are merged in decryption [1,7]. In the left half of Figure 2, the original AES encryption and decryption processes are shown, while in the other half, the processes for the T-Box algorithm are shown. Some functions are reordered in order to merge the functions. Each one-byte S-Box output of SubBytes and InvSubBytes is multiplied by the four coefficients of the polynomials (2) and (3) respectively, and these two one-byte input and four-byte output relation tables become *T-Boxes*. The concrete circuit implementation of this algorithm will be described in Section 5.1.

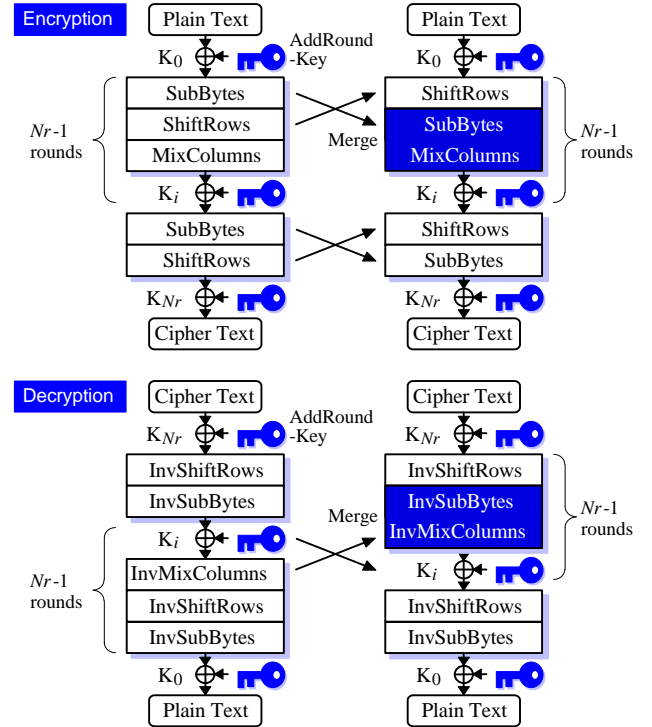


Figure 2. Reordering of Primitive Functions for using the T-Box Algorithm.

3 Issues in Designing Fast S-Box Circuits

3.1 Basic Approaches

There are two approaches for designing S-Box circuits: (1) construct a multiplicative inversion circuit and an affine transformation circuit independently, and then connect these two circuits in serial, and (2) construct a single circuit directly whose input-output relation is equivalent to the S-Box.

In Method (1), circuit area reduction using mathematical theorems over Galois fields (GF) is possible. Various methods for constructing compact inversion circuits over GF have been studied, based on Fermat's Little Theorem, the extended Euclid's Algorithm and so on [11]. In particular, the composite field (or tower field) inversion [8] is effective over $GF(2^8)$, and it can be used

to create compact AES implementations [9,10]. However, these methods are not suitable for achieving 10 Gbps AES circuits due to the large propagation delay.

In Method (2), a fast implementation is possible. The S-Box circuit can be obtained from its truth table by using two-level logic such as SOP (Sum of Products), POS (Product of Sums), PPRM (Positive Polarity Reed-Muller form), etc. [13], or by making a decision diagram such as a BDD [12] or an FDD [14]. In many actual AES implementations the *table-lookup* method is used [11], where the S-Box circuit is automatically synthesized using EDA tools.

Our evaluation results of these various S-Boxes on a 0.13 μm CMOS standard cell library are shown in Table 1. The BDD and table look-up implementations are the fastest of all. These results are obtained in the following steps: (i) implement these S-Boxes as hard-coded VHDL sources (primitive cells are directly called), (ii) adjust cell strengths by a logic synthesis tool without changing circuit structures and (iii) evaluate circuit size and speed using the synthesis tool and static timing analyzer (STA). Although the absolute values of delay/size can vary depending on the ASIC libraries and synthesis tools, the ratios of the circuits' speeds were almost the same, as far as the authors' tests showed.

Table 1. Comparison of S-Box Architectures. (0.13 μm CMOS standard cell, 1 gate = 2 way-NAND)

	Delay (ps)	Size (gate)
Composite field	2,190	354
PPRM	1,010	2,148
SOP	770	1,567
Table look-up	700	1,528
BDD	690	1,399
Proposed method (Twisted BDD)	430	2,818

3.2 Analysis and Issues for BDD Implementation

We selected the BDD architecture as a candidate to achieve the throughput of 10 Gbps, but the speed was not yet adequate. We noticed the following structural characteristics of the S-Box/T-Box BDDs (Figure 3).

- I. Large numbers of selectors in the first and second stage from the primary inputs.
- II. Heavy sharing of the input side selectors.
- III. Few shared selectors on the output side. The third stage to the 7th stage selectors form a large $2^5:1$ selector.
- IV. The variable ordering of the BDD does not much affect the overall structure and size of the BDD.

These phenomena are often observed in functions involving multiplicative inverses over GF, even if other decision diagrams are used. One of the main reasons is that the truth table of the S-Box/T-Box is an almost perfect random number table. There are few common

parts (terms) in the S-Box/T-Box table, and therefore, few circuits (BDD nodes) can be shared.

As a result, the following issues become the obstacles for speeding up the BDD S-Box/T-Box circuit.

1. Extremely large fanout (up to 150) of the two primary inputs that drive the first and second stage selectors.
2. Large fanout (nearly 30) of each output signal of the first stage selectors, which drives a total of 150 of the second stage selectors.
3. Large propagation delay of the output side selectors, where 2:1 selectors are connected serially in five stages.

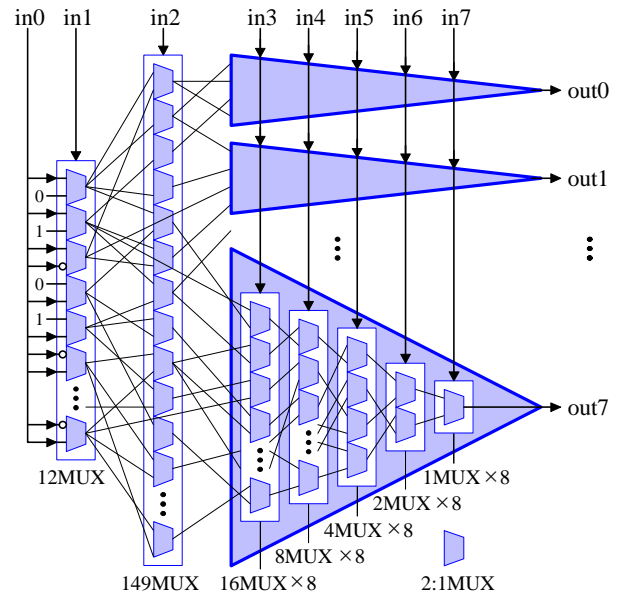


Figure 3. S-Box Implementation using Conventional BDDs.

4 The Proposed Twisted-BDD S-Box/T-box Circuit Architecture

4.1 Twisted variable ordering between primary outputs

In the proposed twisted-BDD architecture (Figure 4), eight BDDs are arranged in parallel, where each BDD corresponds to each primary output. No node is shared between these BDDs and their variable ordering is twisted (rotated) as shown in Figure 4 so that each primary input i ($0 \leq i \leq 7$) drives the $((8+i-j \bmod 8) + 1)$ -th input of BDD $_j$. Each primary input signal is propagated to the next BDDs by passing through drivers (inverters).

As a result, the fanout of each primary input and each driver's output is significantly decreased from 150 down to 30, because the first and second stage selectors are distributed equally between each primary input. Because the BDD structure and its size are almost independent of

the variable ordering, as described in Section 3.2, the fanout of each primary input is almost the same. In the same manner, the fanout of each output of the first stage selectors is decreased from 30 down to 5.

4.2 Parallel Decoding of Selector Control Signals

As shown in Figure 5, we replaced the $2^5:1$ selectors on the output side in each BDD with a combination of a select-signal decoder (5-bit binary to 2^5 -bit one-hot conversion) and a data selection part (1 stage ANDs and 5 stage ORs). As a result, the delay of the $2^5:1$ selectors is reduced, because the decoding of the select-signals and the signal processing in the first and second stage selectors are performed in parallel.

4.3 Use of Negative-Output Selectors and Drivers

We used negative-output primitive-gates for implementing the selectors in each BDD and drivers to reduce the circuit delay and to decrease the number of gates. Because most of the CMOS primitive gates with positive outputs usually consist of a negative output gate followed by an inverter, primitive gates with negative output are faster and smaller.

4.4 Evaluation Results and Discussion

As shown in Table 1, the S-Box speed is increased 1.5 to 2 times by the proposed method. We obtained a 430-ps delay S-Box on a 0.13 μm CMOS standard cell library, and this is the fastest we know of. In spite of the incorporated highly parallel circuit structure shown in Figure 4, the total circuit size remains only double the original BDD, because in the original BDD, the selectors in the 3rd to the 7th stage are already unshared and separated between primary outputs.

We believe that further improvement in speed will be quite difficult if any of the other logic circuit structures described in Section 3.1 are used, for the following two reasons:

First, if any two-level logic such as SOP is used, the number of prime terms increases (for example, 150 terms in SOP) and the fanout of the prime inputs becomes large. However, in contrast to our twisted-BDD, distributing and reducing fanout are difficult, because each primary input signal drives almost the same number of prime selectors.

Second, if any decision diagram other than BDD is used, the propagation delay of each node becomes much larger. For example, if FDD is used, each node is implemented by AND+XOR, and this is much slower than a 2:1 selector used as a BDD node. A fast 2:1 selector cell is available in most ASIC libraries.

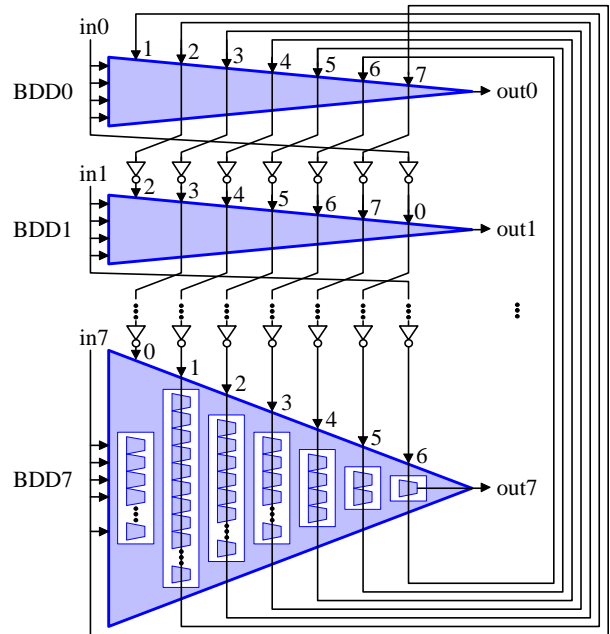


Figure 4. The Proposed Twisted-BDD Architecture.

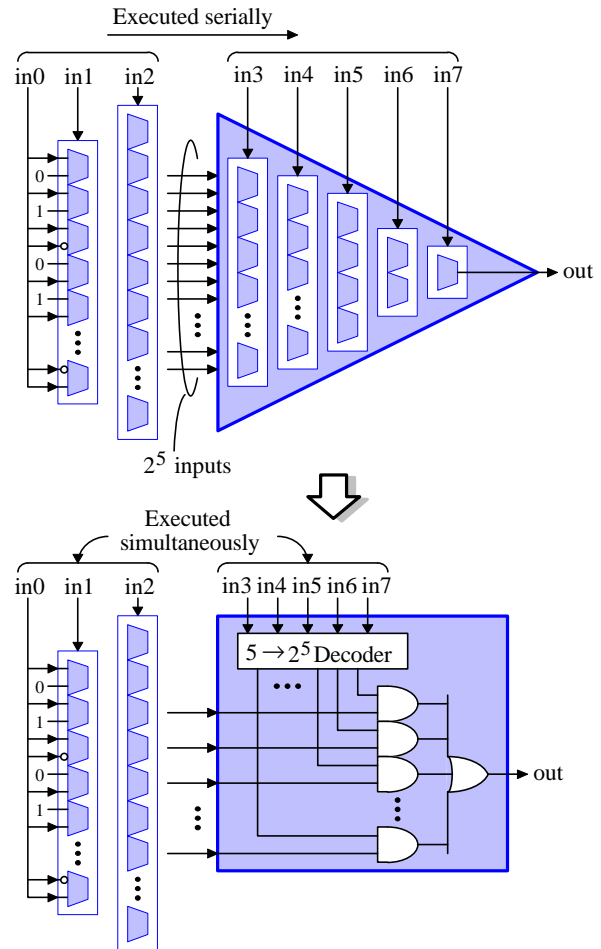


Figure 5. Speedup by Parallel Decoding of Selector Control Signals.

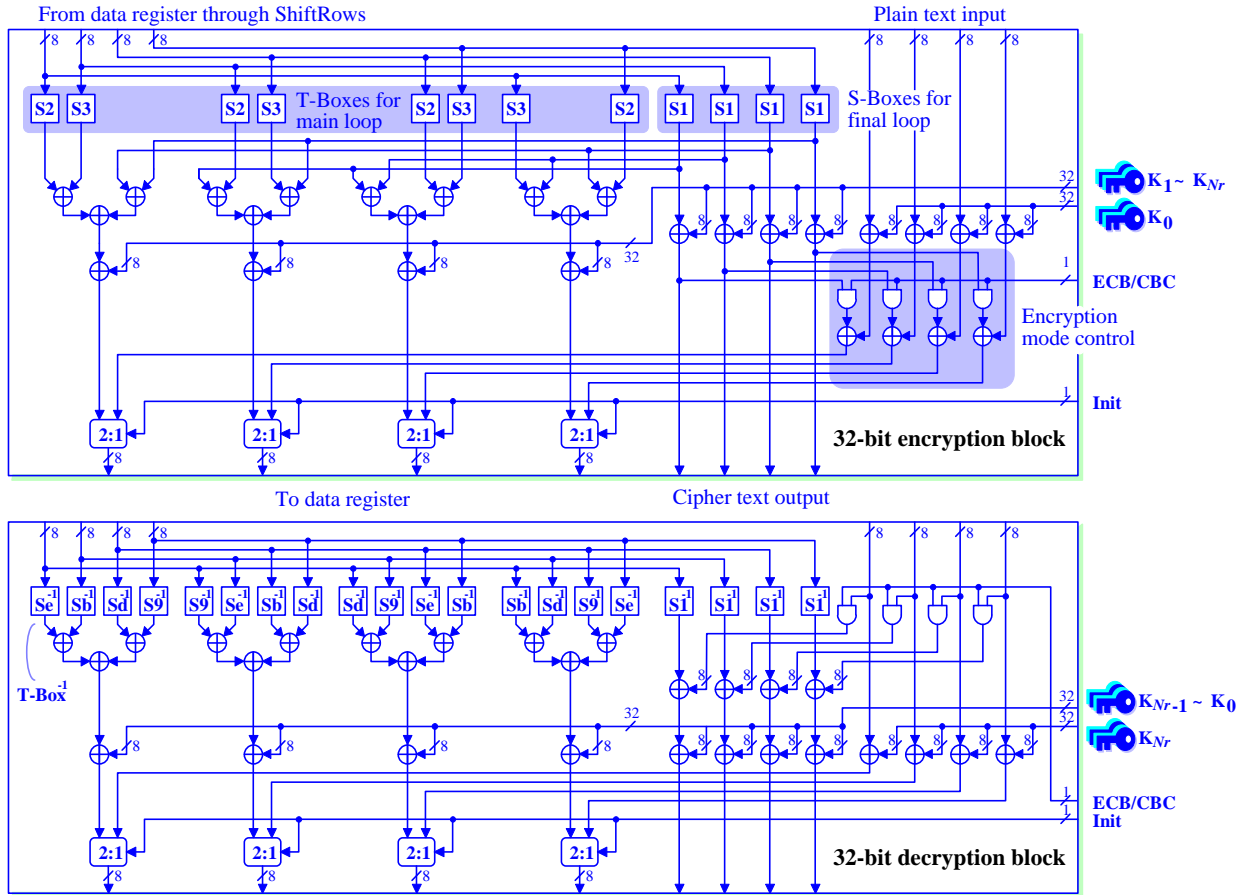


Figure 6. Our Data-path Implementation of the T-Box Algorithm.

5 Implementation Results of 10 Gbps AES Circuit

5.1 Combining T-Box Algorithm and Twisted S-Box/T-Box Circuit Architecture

To achieve 10 Gbps throughput with a 780 MHz clock, we used the T-Box algorithm described in Section 2.2, together with our twisted-BDD method. The reason was to minimize the delay of the MixColumns and InvMixColumns operations.

The 32-bit data path of our circuit implementation using the T-Box algorithm is shown in Figure 6. The complete circuit consists of four identical blocks and data registers (Figure 6 shows only one combinational logic block). The encryption path and decryption path are separated and unshared. In both paths, data is stored in a register and the encryption and decryption are done by iterative execution of the round operation, which corresponds to the combinational circuit block. If only a slower clock is available, it is still possible to achieve 10 Gbps throughput by duplicating the combinational circuit blocks and connecting them in serial, i.e., by using the unrolling technique.

In the encryption path shown in the upper half of

Figure 6, the output of the data register is connected to multiple T-Boxes following the ShiftRows operation. In Figure 6, T-Boxes S_k and S_k^{-1} compute $S(x) \times k$ and $S^{-1}(x) \times k$ respectively, where $S(x)$ is the S-Box and k is a constant value. Then the outputs of the T-Boxes and the round-key data are XORed and returned to the data register, except for the final round operation. Because the final operation is different from the other rounds (MixColumns is unnecessary, and an initial-data load and multiple-mode support are required), an independent computation path exists in parallel.

In summary, each 8-bit block of the data register drives 5 T-Boxes, and 80 T-Boxes are used in the complete circuit. Each T-Box is constructed using the twisted-BDD architecture. However, in the encryption path, some T-Boxes can be shared, so only 48 T-Boxes are used in the whole circuit.

5.2 Implementation Results and Discussion

We achieved 10 Gbps throughput and a 780 MHz clock cycle on a 0.13 μm CMOS standard cell library. We did not implement any key scheduler under the assumption that the round-keys are stored in an external register file, but on-the-fly generation of round-keys is possible

without difficulty.

The evaluation of the speed and size of our implementation was done by the same method described in Section 3.1. The power consumption was estimated by a simulation-based method. In this method, a timing simulation is performed using a synthesized net-list and a given set of test input data, and the number of switching events for all internal gates are counted. The effects of dynamic hazards is reflected in the power estimation results.

The T-Box architecture is almost 20% faster than the basic algorithm in Section 2.1 (Table 2), although strong T-Box drivers are necessary. The circuit size and power are still reasonable. Much of the critical path delay is used by the T-Box (Table 3) and this shows that it will be difficult to achieve the maximum throughput without using the twisted-BDD architecture.

Regarding the T-Box design, the twisted-BDD architecture is suitable for a high clock-speed AES implementation because of its low propagation delay (see Table 1 in Section 3.1).

Table 2. Performance of our AES Circuit in CBC Mode.

(0.13 μ m CMOS standard cell, 1gate = 2way-NAND)

	Size (gates)	Max Clock freq. (MHz)	Through-put (Gbps)	Power (Watts, 780 MHz, 1.5V)
Twisted-BDD + T-Box Algorithm (Encryption)	167,566	909	11.6	1.92
Twisted-BDD + T-Box Algorithm (Decryption)	282,494	885	11.3	3.31
Twisted-BDD + Basic Algorithm (Decryption)	61,841	699	8.9	(1.02)

Table 3. Delay of Each Primitive Function in Decryption Circuits.

(0.13 μ m CMOS standard cell, 1gate = 2way-NAND)

	Twisted-BDD + T-Box Algorithm	Twisted-BDD + Basic Algorithm
Register delay	110 ps	100 ps
AddRoundKey and InvMixColumns	---	700
T-Box drivers	260	---
Inv T-Box or Inv S-Box	430	430
Sum of T-Box results and AddRoundKey	100	---
Register input selector	50	50
Register setup-time	180	150
Total	1,130	1,430

6 Conclusion

In this paper, we presented a high-speed AES circuit design, running at speeds over 780 MHz and achieving 10 Gbps throughput in all encryption modes including the CBC mode. To reduce the propagation delay of the S-Boxes, we developed a special logic circuit architecture named twisted-BDD, where the fanout of signals is distributed in the S-Box. The T-Box algorithm that merges the S-Box and MixColumns function is also used. As far as the authors know, this is the first 10 Gbps AES circuit which can support all encryption modes.

References

- [1] J. Daemen et al., "AES Proposal: Rijndael," <http://csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf>.
- [2] "Advanced Encryption Standard (AES)," <http://csrc.nist.gov/encryption/aes/index.html>
- [3] T. Ichikawa et al., "Hardware Evaluation of the AES Finalists," *Proc. Third AES Candidate Conference*, pp. 279-285, 2000.
- [4] H. Kuo et al., "Architectural Optimization for a 1.82 Gbits/sec VLSI Implementation of the AES Rijndael Algorithm," *Proc. CHES2001, LNCS Vol. 2162*, pp. 53-67, 2001.
- [5] B. Weeks et al., "Hardware Performance Simulation of Round 2 Advanced Encryption Standard Algorithm," <http://csrc.nist.gov/encryption/aes/round2/NSA-AESfinalreport.pdf>.
- [6] M. McLoone et al., "High Performance Single-chip FPGA Rijndael Algorithm Implementations," *Proc. CHES2001*, pp. 68-80, 2001.
- [7] V. Fischer et al., "Two Methods of Rijndael Implementation in Reconfigurable Hardware," *Proc. CHES2001, LNCS Vol. 2162*, pp. 81-96, 2001.
- [8] J. Guajardo and C. Paar, "Efficient Algorithms for Elliptic Curve Cryptosystems," *Proc. of 17th Annual Intl. Cryptology Conf. (CRYPTO'97), LNCS Vol. 1294*, pp. 342-356, 1997.
- [9] A. Rudra, et al., "Efficient Rijndael Encryption Implementation with Composite Field Arithmetic," *Proc. CHES2001 LNCS Vol. 2162*, pp. 175-188, 2001.
- [10] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A Compact Rijndael Hardware Architecture with S-Box Optimization," *Proc. of ASIACRYPT2001, LNCS Vol. 2248*, pp. 239-254, 2001.
- [11] U. Mayer, C. Oelsner and T. Kohler, "Evaluation of Different Rijndael Implementations for High-end Servers," *Proc. of IEEE Intl. Symp. On Circuits and Systems (ISCAS2002)*, 2002.
- [12] R.E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation," *IEEE Transactions on Computers*, Vol. C-35, No. 8, pp. 677-691, 1986.
- [13] T. Sasao, "AND-EXOR expressions and their optimization", in Sasao, editor: *Logic Synthesis and Optimization*, Kluwer Academic Publishers, 1993.
- [14] U. Kebshull and W. Rosenstiel, "Efficient Graph-Based Computation and Manipulation of Functional Decision Diagrams," *European Design Automation Conf. '93*, pp. 278-282, 1993.