ASIC Performance Comparison for the ISO Standard Block Ciphers

Takeshi Sugawara[1], Naofumi Homma[1], Takafumi Aoki[1], and Akashi Satoh[2]

[1] Graduate School of Information Sciences, Tohoku University
Aoba 6-6-05, Aramaki, Aoba-ku, Sendai-shi, Miyagi, 980-8579, Japan
[2] National Institute of Advanced Industrial Science and Technology
1-18-13, Sotokanda, Chiyoda-ku, Tokyo, 101-0021, Japan
{sugawara, homma} @aoki.ecei.tohoku.ac.jp, aoki@ecei.tohoku.ac.jp,
akashi.satoh@aist.go.jp

Abstract. This paper presents performance comparisons of the ISO/IEC 18033 standard block ciphers, AES, Camellia, SEED, TDEA, MISTY1, and CAST-128 in ASIC hardware. All the algorithms are implemented with a loop architecture where one round function block is used iteratively, and S-boxes are generated from lookup tables. In addition to the straightforward implementations, compact data path architectures were designed for SEED and MISTY1 using the characteristics of nested round functions. For the compact AES and Camellia circuits, composite field S-boxes were also used in addition to the lookup table S-box. All of the designs were synthesized by using a 0.18-μm CMOS standard cell library, and the sizes and speeds were evaluated. The highest throughput of 3.35 Gbps with 75.8 Kgates was obtained by the 128-bit block cipher AES, and the 64-bit block cipher TDEA showed the smallest gate counts of 4.6 Kgates with 228 Mbps.

Keywords: ISO/IEC 18033, Block Cipher, AES, Camellia, SEED, TDEA, MISTY1, CAST-128, Cryptographic Hardware

1. Introduction

ISO/IEC 18033-3 [1] specifies three 128-bit block ciphers (AES [2], Camellia [3], SEED [4]) and three 64-bit block ciphers (TDEA [5], MISTY1 [6], and CAST-128 [7]), and these algorithms have been widely implemented as software and hardware in practical use. Performance comparisons of cryptographic algorithms in software implementations were often made on the same processor [13]. There are some reports on hardware comparisons for cipher algorithms using the same platform, but no hardware comparisons for the ISO standard ciphers, as far as the authors know.

This paper presents performance comparisons for all of the ISO/IEC 18033-3 standard block ciphers by using an ASIC library. All of the algorithms were implemented by using a loop architecture where one round function block is repeatedly used, and lookup table logic is used for the S-boxes. In addition to the straightforward implementations, we also designed some variations for compact

hardware. In the next section, datapath architectures are described in detail. Then the designs were synthesized using a 0.18-μm CMOS standard cell library in Section 3, and the speeds and sizes were evaluated to provide the basic characteristics of each cipher in hardware implementations. The conclusions are described in Section 4.

2. Hardware architectures

2.1. 128-bit block ciphers

AES

AES (Advanced Encryption Standard) [2] is a Substitution-Permutation Network (SPN) block cipher standardized by the NIST (National Institute of Standard and Technologies), which supports three key lengths of 128, 192, and 256 bits. Fig. 1 shows our AES hardware architecture where the left part indicates a 128-bit data randomization block and the right part is a key scheduler for a 128-bit key. SPN ciphers require different data paths for encryption and decryption, but we shared 128-bit selectors, registers, and XORs between the paths. The encryption datapath consists of four function blocks: ShiftRows (byte-oriented rotations), SubBytes (sixteen 8-bit s-boxes), MixColumns (four 32-bit matrix multiplications), and AddRoundKey (128-

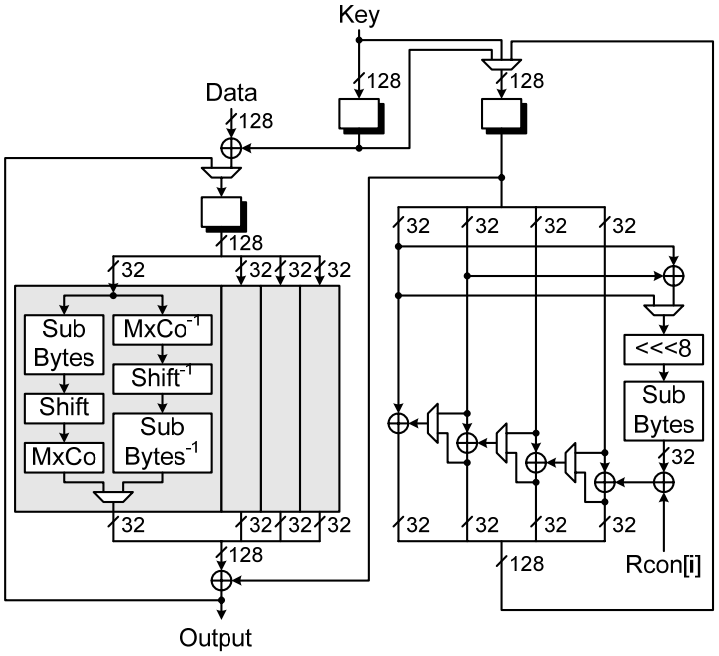

Fig. 1. AES hardware architecture.

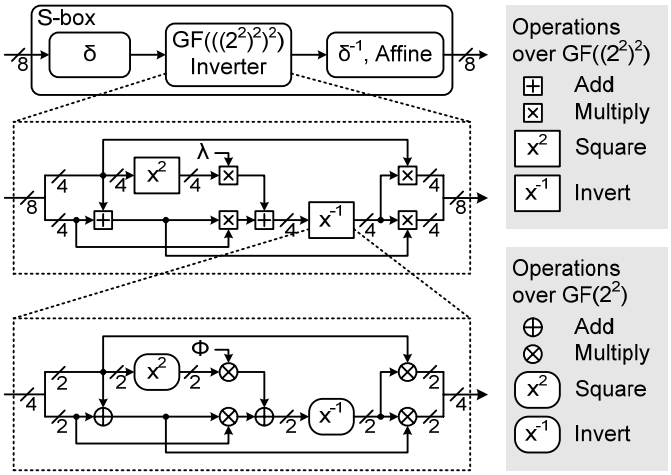

Fig. 2. Compact AES S-box using composite field GF(((2^2)^2)^2) arithmetic.

bit XOR). The decryption datapath consists of four inverse function blocks: InvShiftRows, InvSubBytes, InvMixColumns, and AddRoundKey (shared with the encryption data path). Our architecture uses a pair of sixteen S-boxes for the data randomization, and four of them for the key scheduling. Our architecture takes 10 clock cycles for one 128-bit data block, and thus its throughput is calculated as 128 bits / 10 clocks × operating frequency.

The AES S-box is a combination of a multiplicative inverse on a Galois field $GF(2^8)$ and an affine transformation. It is often implemented as the lookup table logic shown in the specification [2], but that requires a large circuit block. In contrast, the composite field $GF(((2^2)^2)^2)$ S-box shown in Fig. 2 [9] can greatly reduce the hardware resources required. Therefore, we designed both lookup table and composite field S-boxes for the performance comparison.

Camellia

Camellia is a Feistel-type block cipher jointly developed by NTT (Nippon Telegraph and Telephone Corp.) and Mitsubishi Electric [3]. As with AES, Camellia supports 128-, 192-, and 256-bit keys. An advantage of the Feistel cipher is that the same datapath can be used for encryption and decryption. Fig. 3 shows our Camellia hardware where a 64-bit round function (consisting of eight 8-bit S-boxes and a lot of XOR gates) and two 64-bit linear functions for FL and FL^{-1} are used for data randomization. The round function block is also used for key initialization. In encryption (or decryption), the round function is repeated 18 times, the FL/FL-1 functions and key whitening (128-bit XOR) are used twice each, and one clock cycle is required for data I/O, and thus the total number of clocks is 23. The round keys can be generated on the fly, but when the secret key is changed, the key initialization process takes 6 clock cycles.

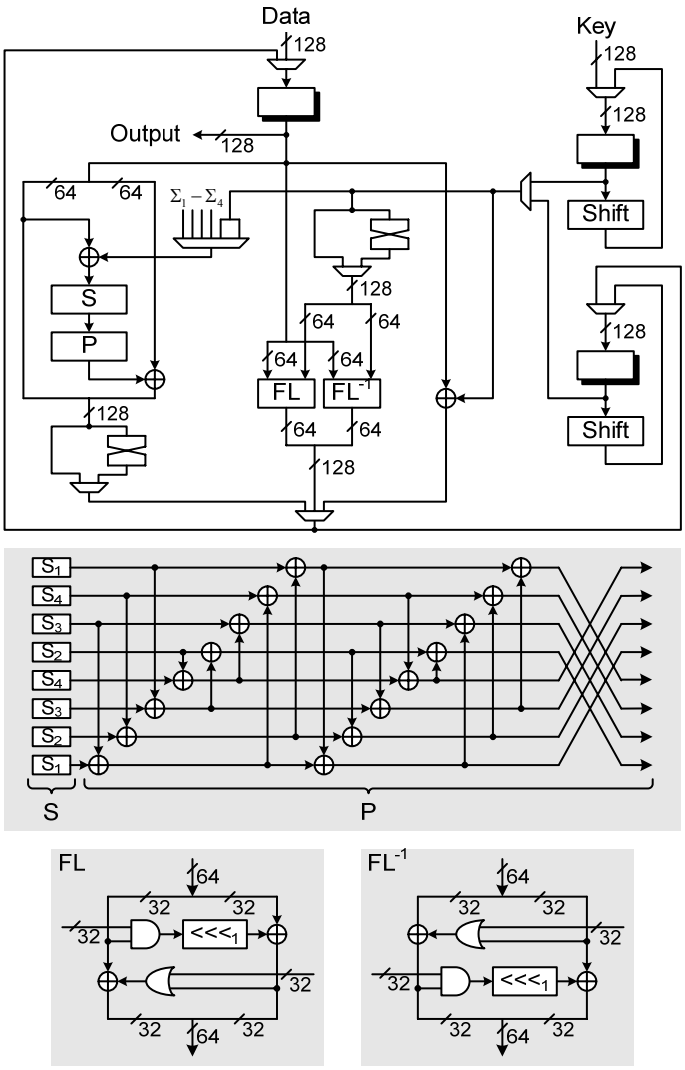

Fig. 3. Camellia hardware architecture.

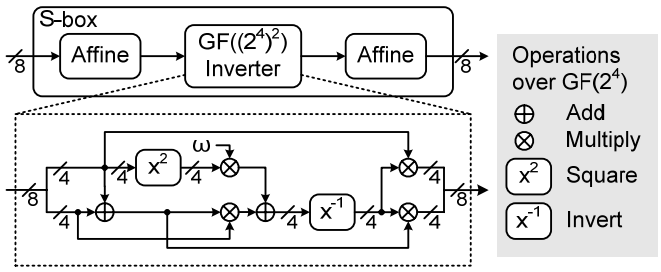

Fig. 4. Camellia S-box using an inverter on the composite field $GF((2^4)^2)$.

The S-box of Camellia is a combination of an affine transformation and multiplicative inverse on a Galois field similar to AES. The Camellia specification [10] shows a lookup table for the inversion, but the field structure is not clearly described. Reference [9] proposed a compact Camellia S-box circuit using the composite field $GF((2^4)^2)$ as shown in Fig. 4. We implemented two types of S-box, look-up table and composite field versions.

SEED

SEED is a Feistel-type block cipher developed by KISA (Korea Information Security Agency) [4], and it only supports a 128-bit key. SEED was designed to optimize its performance on 32-bit processors, and thus 32-bit additions and subtractions are used in data randomization and key scheduling. For the SEED round function, a triplet of a 32-bit G function, a 32-bit XOR, and a 32-bit addition (or subtraction) is executed three times. We designed the two SEED hardware architectures shown in Figs. 5 and 6. Fig. 5 is a straightforward version that executes the round function in one clock cycle, and Fig. 6 is a compact version where the round function is divided into three subfunctions, and one subfunction block is used repeatedly. The G function contains two kinds of 8-bit S-boxes, S1 and S2 defined as lookup tables. In Fig. 5, the data randomization block uses three G function blocks, and the key scheduler has two of them, and the round function is executed in one

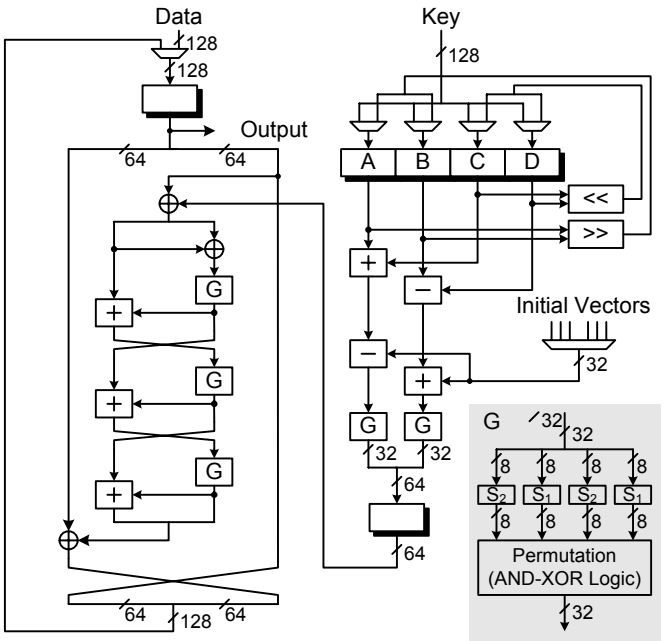

Fig. 5. Straightforward SEED hardware architecture.

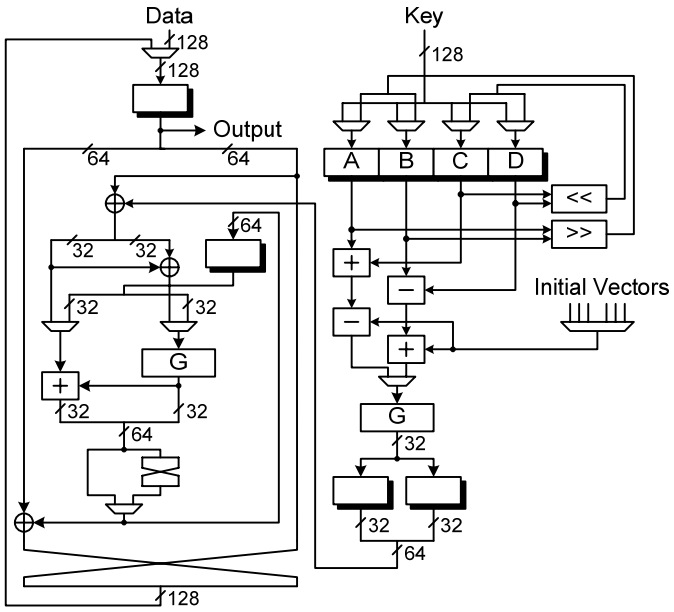

Fig. 6. Compact SEED hardware architecture.

cycle. The total number of clock cycles for one encryption or decryption is 20. The compact architecture in Fig. 6 has only two G function blocks, with one used three times in the data randomization block and the other used twice in the key scheduler. One encryption or decryption takes 52 clock cycles including an extra 4 cycles for initialization and data I/O. Key scheduling is performed on-the-fly in both architectures.

2.2. 64-bit block ciphers

TDEA

The TDEA (Triple Data Encryption Algorithm) repeats the Feistel cipher DES (Data Encryption Standard) algorithm three times by using two 56-bit keys (2-key TDEA) or three keys (3-key TDEA) [5]. We designed a TDEA hardware architecture shown in Fig. 7 that supports the two key options. By using a 32-bit round function block, the TDEA hardware performs the 16-round DES operation three times, and thus 48 cycles are required for one encryption or decryption. According to the TDEA specification [5], we implemented the 6-bit input and 4-bit output S-boxes using lookup tables. Reference [10] provided Boolean expressions for the S-boxes with fewer logic gates. However, the critical path of the logic is rather long, and the S-box

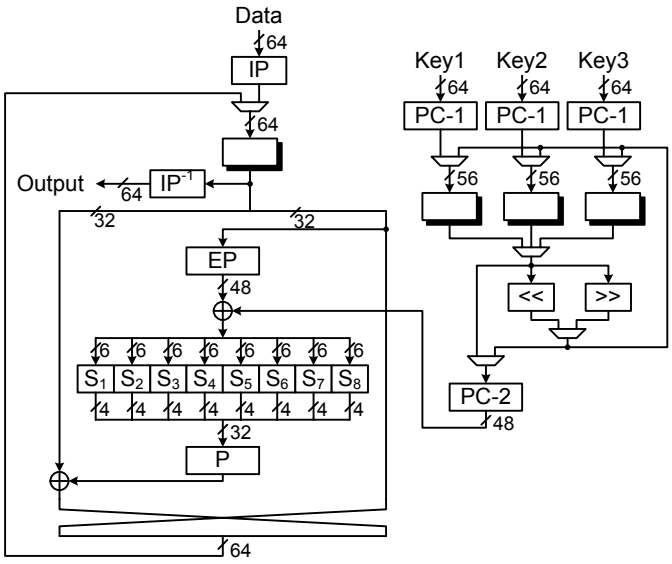

Fig. 7. TDEA hardware architecture.

circuit is rather small even using a lookup table logic. Therefore, we used the lookup table logic that can support a higher operating frequency.

MSTY1

MISTY1 from Mitsubishi Electric [6] is a Feistel-type 64-bit block cipher with a 128-bit key. The data randomization block has a nested structure, and thus it can be divided into several stages to meet performance requirements. Fig. 8 shows a straightforward hardware architecture for MISTY1 where the 64-bit FL/FL^{-1} function and the 32-bit FO function (consisting of three sets of 32-bit FI function and 32-bit XORs) can be performed in the same clock cycle. Fig. 9 is a compact version where the FO function is executed in three clock cycles by repeatedly using one FI function block. The FI function uses 7-bit and 9-bit S-boxes defined as lookup tables. The number of rounds is recommended as eight in the specification [6], and thus we evaluated the throughputs of MISTY1 circuits based on this number, though any multiple of four can be used as the number of rounds. The straightforward architecture in Fig. 8 takes one additional cycle for data I/O, and thus the total cycle count for one encryption or decryption is 9. The compact architecture in Fig. 9 executes each third of the FO sub-function and the FL/FL^{-1} function in different cycles. The FO sub-function block is repeatedly used $3\times8 = 24$ times, the FL/FL^{-1} function block is used 5 times, and one additional I/O cycles is required, requiring 30 cycles in total.

When the 128-bit secret key is changed, the key scheduler generates a 128-bit subkey in 10 clock cycles, but once the sub-key is generated, the round keys KO, KI, and KL can be generated on-the-fly.

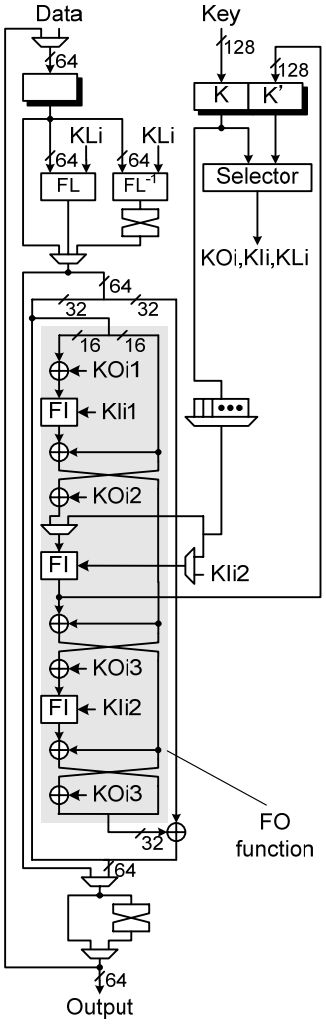

Fig. 8. Straightforward MYSTY1 hardware architecture.

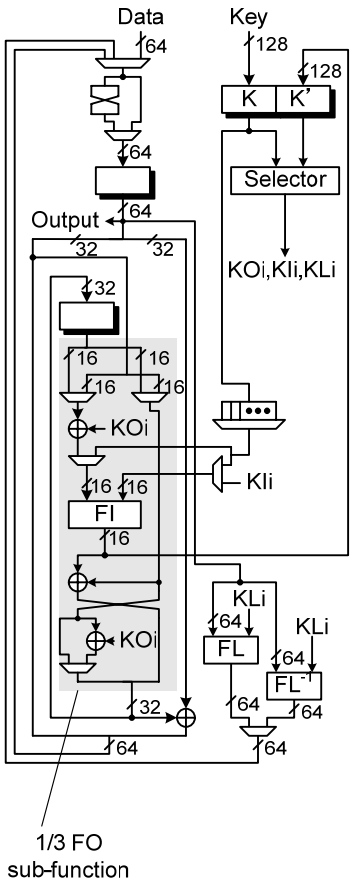

Fig. 9. Compact MYSTY1 hardware architecture.

CAST-128

CAST-128 as developed by Carlisle Adams [7] is a Feistel cipher, where the key length is variable in the range of 40~128 bits in 8-bit steps. The number of rounds is 12 or 16 for 40~80-bits and 88~128-bit keys, respectively. The CSE (Communications Security Establishment) approved CAST-128 for use by the Government of Canada. The popular e-mail ciphering tool PGP (Pretty Good Privacy) uses CAST-128 as the default algorithm. Eight different types of 8-bit input and 32-bit output S-boxes defined as lookup tables are used many times for data randomization and key scheduling. It also uses three different types of 32-bit F functions. The algorithm can be implemented efficiently on 32-bit processors, but the

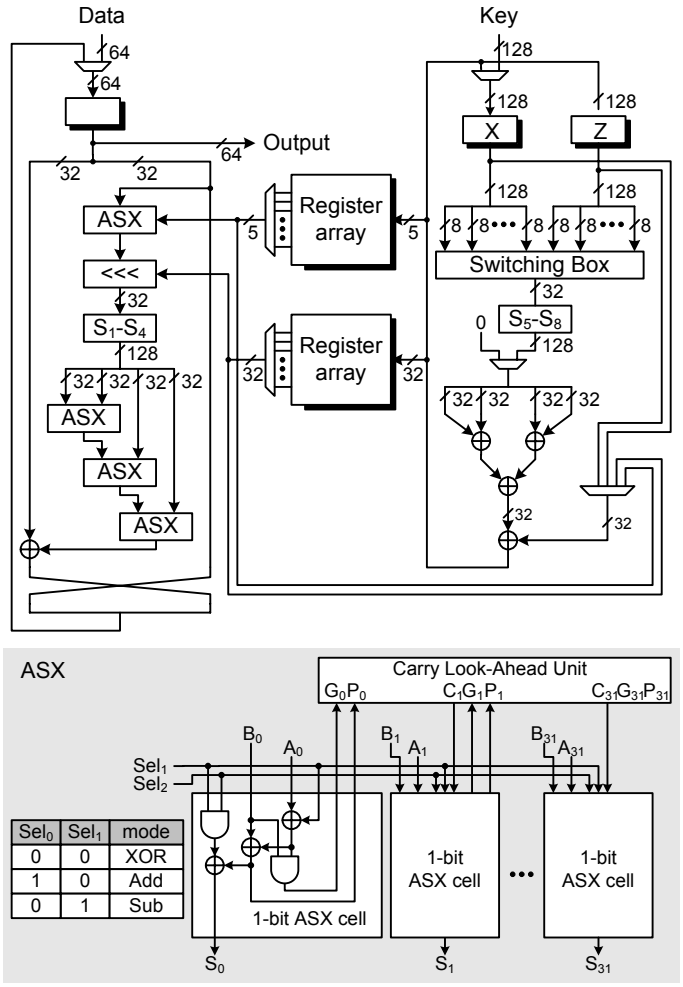

Fig. 10. CAST-128 hardware architecture.

large S-boxes and the use of three different round functions are problematic in the development of compact hardware. In order to achieve compact CAST-128 hardware, we used a minimum set of S-boxes and merged the three round functions by using a unified arithmetic unit called ASX (Add-Sub-XOR), which switches among three arithmetic operations (32-bit addition, subtraction, and XOR) [11] as shown in Fig. 10. A carry-look-ahead scheme was used for the ASX block considering the balance between speed and gate count. Each encryption or decryption requires 16 round functions and 1 data I/O, and thus takes 17 cycles in total.

3. Performance evaluation

Table 1 shows the hardware performance comparisons of the proposed architectures for six cipher algorithms. The designs were synthesized with the Synopsys Design Complier (Version 2005.09) with two optimizations, size and speed. The speeds and sizes were evaluated by using a 0.18-μm CMOS standard cell library [12] under the worst case conditions. The lookup table and the composite field S-boxes were used for AES and Camellia, and the straightforward and the compact round functions were applied to SEED and MISTY1. "Hardware efficiency" in the

Table 1. Performance comparison in ASIC

Algorithm	Mode	Block Length (bits)	Key Length (bits)	Cycle	S-box	Opti-mize	Freq. (MHz)	Thr'put (Mbps)	Area (Kgates)	Efficiency (Kbps/ gates)
AES	Enc.	128	128	10	Table	Area	128.7	1,647.4	31.2	52.7
						Speed	172.4	2,206.9	36.6	60.3
							261.8	3,350.8	75.8	44.2
					$GF(((2^2)^2)^2)$	Area	102.0	1,306.1	18.3	71.4
						Speed	128.0	1,638.9	28.7	57.1
Camellia	Enc./ Dec.	128	128	23	Table	Area	113.6	632.4	11.9	53.1
						Speed	188.3	1,048.1	21.5	48.7
					$GF((2^4)^2)$	Area	101.9	567.3	9.1	62.0
						Speed	146.6	816.0	16.5	49.5
SEED	Enc./ Dec.	128	128	16	Table	Area	67.5	540.2	23.5	23.0
						Speed	84.6	676.8	36.2	18.7
				52	Table	Area	92.4	227.5	11.5	19.8
						Speed	170.1	418.6	18.9	22.2
TDEA	Enc./ Dec.	64	56, 112, 168	48	Table	Area	170.9	227.9	4.6	49.4
						Speed	311.5	415.4	7.0	59.7
MISTY1	Enc./ Dec.	64	128	9	Table	Area	40.2	286.0	14.1	20.3
						Speed	96.9	689.1	29.4	23.4
				30	Table	Area	92.6	197.5	9.3	21.3
						Speed	171.5	365.9	17.6	20.8
CAST-128	Enc./ Dec.	64	128	17	Table	Area	50.5	189.9	26.4	7.2
						Speed	77.6	292.1	32.8	8.9

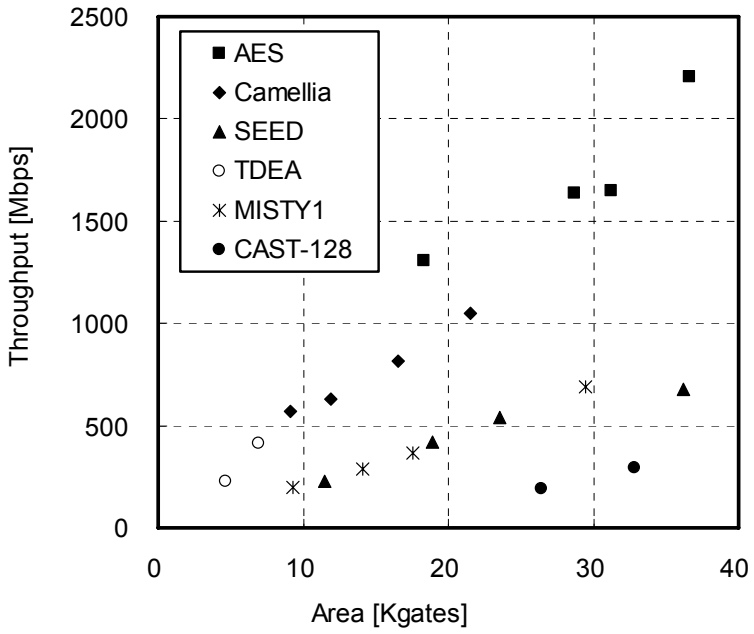

Fig. 11. Throughput versus Area of each implementation

last column means throughput per gate, and thus implementations with higher throughput and smaller gate counts show higher values. Fig. 11 plots throughput versus gate count of each implementation. The horizontal axis is gate count, and the vertical axis is the throughput, so dots farther to the left and moving up show higher hardware efficiency.

The AES hardware with lookup table S-boxes achieved the highest-throughput of 3.35 Gbps with 75.8 Kgates because SPN ciphers generally require fewer iteration rounds in comparison with Feistel ciphers. The throughput is more than three times higher than that of any other ciphers, but it does not fit in the Fig. 11. Therefore, we also synthesized a middle-speed version with the lookup table S-boxes. The composite field S-box version also showed the highest hardware efficiency of 71.4 Kbps/gate, which is at least 15 % higher than the other ciphers. The Camellia hardware with the composite field S-box had the smallest gate count of 9.1 Kgates among the three 128-bit block ciphers. This is because the Feistel network uses a 64-bit round function that is half the size of AES and the basic components in the function are similar to AES. However, as a natural result its throughputs are less than half compared to AES. However, Camellia obtained the maximum throughput of 1.05 Gbps with 21.5 Kgates, and only Camellia and AES showed throughputs higher than 1 Gbps.

The SEED hardware shows a wide variety of performance from 166.2 Mbps with 10.6 Kgates to 676.8 Mbps with 36.2 Kgates depending on the architectures of the round function block. The area for the compact architecture is comparable to Camellia

hardware, but the throughput is much lower due to the large number of clock cycles (52), since 3 clocks are required for each round. The straightforward version performs one round in one clock, but then the round function block has the long critical path, and thus the operating frequencies stay low.

The TDEA hardware had extremely small gate counts of 4.6~7 Kgates and high operating frequencies of 170.9~311.5 MHz since the S-boxes are very small and the permutation function P is implemented simply with wire twisting that needs no transistors. However, the typical throughputs of 227.9~415.4 Mbps are on average among the 64-bit ciphers because the large number of iteration rounds (48). Note that TDEA achieved very high hardware efficiencies of 49.4~59.7 Kbps/gate, comparable to the 128-bit cipher Camellia, and higher than SEED.

The MISTY1 hardware had the highest throughput of 689.1 Mbps among the 64-bit block ciphers. Not only for speed, but the compact version also showed a small gate count of 9.3 Kgates. These results were made possible by the flexibility of the MISTY1 algorithm with the nested network structure, and many architectures can be designed, while the DES algorithm is simple but has almost no flexibility.

The CAST-128 had large gate counts of 26.4 ~ 32.8 Kgates, which is 6~7 times larger than TDEA, and also larger than the 128-bit ciphers in many cases, even though it uses a Feistel network that should be suitable for a compact implementation. The reason is obviously because the total size of the CAST-128 S-boxes is 8 Kbytes while DES needs only 256 bytes. The low frequencies of 50.5~77.6 MHz are caused by the four 32-bit arithmetic operations in the ASX blocks in Fig. 9, in addition to the signal delay in the large S-boxes, even though these components cause no problems in software implementations on 32-bit microprocessors. In contrast, other ciphers use small S-boxes and/or binary field arithmetic where no carry propagation occurs.

From these implementation results, we can basically say that AES and TDEA are the best algorithms for ASIC hardware implementations among the 128-bit and 64-bit block ciphers, respectively.

Lastly, we briefly take a look at software performance. Table 2 shows the speeds of the cryptographic software on a Pentium 4 2.1 GHz processor [13]. AES also shows the best performance in software. It is interesting to note that CAST-128 software achieved a throughput of 342.0 Mbps, which is faster than the 292.1 Mbps in our hardware. In addition to the speed advantage of the Pentium processor fabricated by using advanced LSI processing technology, the CAST-128 algorithm was optimized

Table 2. Software performance on Pentium 4 processor [13]

Algorithm	Throughput (Mbps)
AES	488.08
Camellia	152.64
SEED	-
TDEA	78.80
MISTY1	-
CAST-128	342.00

for 32-bit MPU platforms where a fast customized 32-bit arithmetic unit and a large cache memory are available. In contrast, the TDEA software shows low throughput of 78.8 Mbps where the hardware achieved 415.4 Mbps. This is mainly because of the tedious bit manipulations in the permutation functions such as the P function, which requires a very heavy workload for a microprocessor while it needs no gate logic in hardware. Consequently, we have to be aware that the best performance for a cipher algorithm depends highly on the platform to be used.

4. Conclusion

This paper compared the hardware performances for all of the ISO/IEC 18033-3 standard block ciphers by using a 0.18-μm CMOS standard cell library. Each algorithm was implemented as a simple loop architecture with lookup table S-boxes. The round functions of SEED and MISTY1 have nested network structures, and thus we also designed compact versions for them by dividing each round function into sub-function blocks, and repeatedly using one block. For AES and Camellia, we use composite field S-boxes in addition to the lookup table versions.

AES is only the SPN cipher while the others have Feistel networks, and thus we implemented two data paths for the AES data randomization block. Even with this in mind, the highest throughput of 3.35 Gbps and the highest hardware efficiencies of 71.4 kbps/gate clearly show the high advantages of AES in ASIC implementations over all of the other ciphers. The Camellia hardware achieved the smallest gate counts of 9.1 Kgates among the 128-bit ciphers as a consequence of the 64-bit round function of the Feistel cipher, but the larger number of rounds limited the throughput to 1.01 Gbps.

Among the 64-bit block ciphers, the simple structure of TDEA resulted in an extremely small size of 4.7 Kgates with area optimization, and a very high operating frequency of 311.5 MHz and high hardware efficiency of 59.7 Kbps/gate with speed optimization, but the throughput of 415.4 Mbps is not so high, because the TDEA hardware needs 48 cycles for one encryption or decryption while AES and Camellia require only 10 and 23 cycles, respectively. Still, TDEA is the best algorithm among the three 64-bit ciphers from the hardware performance viewpoint.

In this paper, each cipher algorithm was designed in a straightforward way using simple loop architecture, and no special optimizations for each algorithm were done, except for the AES and Camellia S-box. Therefore, algorithm-specific hardware architectures would achieve better performances. However, we believe that our results based on the circuits using the loop architecture and synthesized with the same ASIC library fairly represent the basic characteristics of each cipher. We are now designing algorithm-specific hardware for each cipher, and will report the results in the near future.

References

1. ISO/IEC 18033-3 “Information technology -- Security techniques -- Encryption algorithms -- Part 3: Block ciphers,” Jul. 2005, http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=37972
2. NIST, “Advanced Encryption Standard (AES),” FIPS PUB 197, Nov. 2001, http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
3. K. Aoki, et al., “Specification of Camellia – a 128-bit Block Cipher Version 2.0,” http://info.isl.ntt.co.jp/camellia/dl/01espec.pdf
4. KISA, “SEED Algorithm Specification,” http://www.cyberprivacy.or.kr/kisa/seed/data/Document_pdf/SEED_Specification_english.pdf
5. NIST, “Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher,” SP 800-67, May 2004. http://csrc.nist.gov/publications/nistpubs/800-67/SP800-67.pdf
6. M. Matsui, “Specification of MISTY1 – a 64-bit Block Cipher,” NESSIE Project, https://www.cosic.esat.kuleuven.ac.be/nessie/workshop/submissions.html
7. C. Adams, “The CAST-128 Encryption Algorithm,” RFC 2114, May 1997, http://www.ietf.org/rfc/rfc2144.txt
8. A. Satoh, et al., “A Compact Rijndael Hardware Architecture with S-box Optimization,” Advances in Cryptology – ASIACRYPT 2001, LNCS 2248, pp. 239-254, Dec. 2001.
9. A. Satoh, et al., “Hardware-Focused Performance Comparison for the Standard Block Ciphers AES, Camellia, and Triple-DES,” Information Security Conference - ISC 2003, LNCS 2851, pp. 252-266, Oct. 2003.
10. M. Kwan, “Bitslice DES,” http:/www.darkside.com.au/bitslice/
11. T. Sugawara, et al., “A High-Performance ASIC Implementation of the 64-bit Block Cipher CAST-128,” 2007 IEEE International Symposium on Circuits and Systems - ISCAS 2007, May 2007.
12. M. Hashimoto, et al. “Standard cell libraries with various driving strength cells for 0.13, 0.18, and 0.35 um technologies, Proc. of Asia and South Pacific Design Automation Conference 2003, pp. 589-590, Jan. 2003.
13. W. Dai, “Crypto++ 5.2.1 Benchmarks.” http://www.cryptopp.com/benchmarks.html