IEEJ TRANSACTIONS ON ELECTRICAL AND ELECTRONIC ENGINEERING
IEEJ Trans 2017; **12**: 96–104
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI:10.1002/tee.22349

Paper

Secure Data Processing with Massive-Parallel SIMD Matrix for Embedded SoC in Digital-Convergence Mobile Devices

Takeshi Kumaki*[a], Member
Tetsushi Koide**, Non-member
Takeshi Fujino*, Non-member

This paper presents secure data processing with a massive-parallel single-instruction multiple-data (SIMD) matrix for embedded system-on-chip (SoC) in digital-convergence mobile devices. Recent mobile devices are required to use private-information-secure technology, such as cipher processing, to prevent the leakage of personal information. However, this adds to the device's required specifications, especially cipher implementation for fast processing, power consumption, low hardware cost, adaptability, and end-user's operation for maintaining the safety condition. To satisfy these security-related requirements, we propose the interleaved-bitslice processing method, which combines two processing concepts (bitslice processing and interleaved processing), for novel parallel block cipher processing with five confidentiality modes on mobile processors. Furthermore, we adopt a massive-parallel SIMD matrix processor (MX-1) for interleaved-bitslice processing to verify the effectiveness of parallel block cipher implementation. As the implementation target from the Federal Information Processing Standardization-approved block ciphers, a data encryption standard (DES), triple-DES, and Advanced Encryption Standard (AES) algorithms are selected. For the AES algorithm, which is mainly studied in this paper, the MX-1 implementation has up to 93% fewer clock cycles per byte than other conventional mobile processors. Additionally, the MX-1 results are almost constant for all confidentiality modes. The practical-use energy efficiency of parallel block cipher processing with the evaluation board for MX-1 was found to be about 4.8 times higher than that of a BeagleBoard-xM, which is a single-board computer and uses the ARM Cortex-A8 mobile processor. Furthermore, to improve the operation of a single-bit logical function, we propose the development of a multi-bit logical library for interleaved-bitslice cipher processing with MX-1. Thus, the number of clock cycles is the smallest among those reported in other related-studies. Consequently, interleaved-bitslice block cipher processing with five confidentiality modes on MX-1 is effective for the implementation of parallel block cipher processing for several digital-convergence mobile devices. © 2016 Institute of Electrical Engineers of Japan. Published by John Wiley & Sons, Inc.

Keywords: digital-convergence; DES; triple-DES; AES; parallel processing; interleaved implementation; bitslice implementation; embedded processor

Received 21 May 2015; Revised 16 November 2015

1. Introduction

Mobile devices have spread with the rapid development of embedded system-on-chip (SoC) technology. Thus, almost everyone uses a cellular phone, smartphone, or notebook PC. The processing capability and functions of these mobile devices becomes more advanced and convenient with each new release. Furthermore, various types of digital applications have been converged and executed on a single mobile device, the so-called digital-convergence [1]. While mobile devices have several convenient functions, they also store huge volumes of private information, such as telephone numbers, photos, e-mail addresses, credit card numbers, and business documents. Thus, mobile devices need to be equipped with private-information-secure technology [2]. In particular, cryptographic processing is an important technology to prevent the leakage of personal information. Many mobile devices, such as smartphones and tablets, support hardware cipher processing. The hard-wired cipher intellectual properties (IPs) for SoCs have often been embedded directly in mobile devices, which can prevent accidental leakage of stored data and allow fast transfer of encrypted or decrypted plain data. However, to satisfy the required capability for digital convergence, these circuits entail additional hardware cost and power dissipation in recent mobile devices. The hard-wired-based implementation of the cryptographic algorithm cannot be updated or modified and may be insufficiently secure to protect private information in the future. Moreover, these circuits automatically encrypt and decrypt everything written to or read from flash memory and have unadaptable secure processing. On the other hand, today's mobile devices also support software cipher processing aside from the hardware cipher circuit [3]. Software cipher processing can apply different types of protection to various pieces of data. Conventional software-based general-purpose CPUs can adaptively execute several cipher algorithms but cannot obtain sufficient processing speed. Hardware accelerators, such as the advanced encryption standard new instructions (AES-NI) [4] instruction set, are implemented in Intel and AMD processors. However, cipher processing is limited to the AES-related algorithm and tends to increase power consumption and hardware cost.

To overcome these serious security-related problems, we propose an interleaved-bitslice cipher processing method, which is a software-based highly parallel block cipher processing method, with a massive-parallel SIMD matrix architecture (MX-1) [5–10]. Since MX-1 uses a software-based processing unit for several multimedia algorithm implementations, it is programmable for block cipher processing required by digital-convergence mobile devices.

[a] Correspondence to: Takeshi Kumaki. E-mail: kumaki@fc.ritsumei.ac.jp

* Department of Electronic and Computer Engineering, Ritsumeikan University, 1-1-1, Noji-Higashi, Kusatsu, Shiga 525–8577, Japan
** Research Institute for Nanodevice and Bio Systems, Hiroshima University, 1-4-2, Kagamiyama, Higashi-Hiroshima, Hiroshima 739–8527, Japan

This paper is organized as follows. Section 2 describes the MX-1 architecture in detail. Section 3 introduces the conventional parallel block cipher processing method and the proposed interleaved-bitslice parallel block cipher processing method with MX-1. Section 4 describes the developing environment for MX-1 and conventional embedded processors and compares the performance of the AES block cipher implementation with the five confidentiality modes. Section 5 discusses the data encryption standard (DES) and the triple-DES implementation results with the five confidential modes. Finally, Section 6 concludes this paper.

2. Massive-Parallel SIMD Matrix Architecture

For processing multimedia applications efficiently, various SIMD architectures have been reported [11,12]. However, few processing elements (PEs) can actually be implemented. For overcoming this parallelism-related problem, we have developed a massive-parallel processor based on an SRAM-embedded matrix architecture (MX-1) [5–10] that overcomes the limitations in parallelism of previous architectures. This massive-parallel SIMD matrix architecture achieves, for example, 40 GOPS (giga operations per second) performance for 16-bit additions at 200 MHz clock frequency and 250 mW power dissipation in 90-nm CMOS technology [5]. It is also programmable for all processing functions required for digital-convergence mobile devices. A block diagram of the SIMD matrix architecture is shown in Fig. 1. Also, 1M-bit SRAM is provided for data registers, and 1024 or 2048 two-bit PEs, connected by a flexible switching network, are integrated on a small area of 3.1 mm^2 with 90-nm low-power CMOS technology. A vertical channel (V-ch) connects the PEs, while a horizontal channel (H-ch) connects the SRAM-register space and PEs. Through V-ch, a communication path between a PE and another PE, being a distance of a power of 2 rows away, is generated and operated in one cycle. For H-ch, three register accesses (2 read, 1 write) in the form of a read-modify-write operation are achieved in one cycle by the division of the SRAM into two simultaneously accessible parts. Since the massive-parallel SIMD matrix consists of a simple SRAM-based architecture, the processed data width and the magnitude of parallelism can be changed and optimized in accordance with the application needs. The MX-1 architecture was introduced for embedded SoC in digital-convergence mobile devices [1] and has been used to implement multimedia applications, such as DCT processing [13], JPEG compression [6], face detection [7], fast multiplication [14], image transform [8], random number [9], and stream cipher [10]. The MX-1 has two types of implementation styles, an evaluation board and a simulator. For AES effectiveness for digital-convergence mobile devices, we report on interleaved-bitslice processing with MX-1.

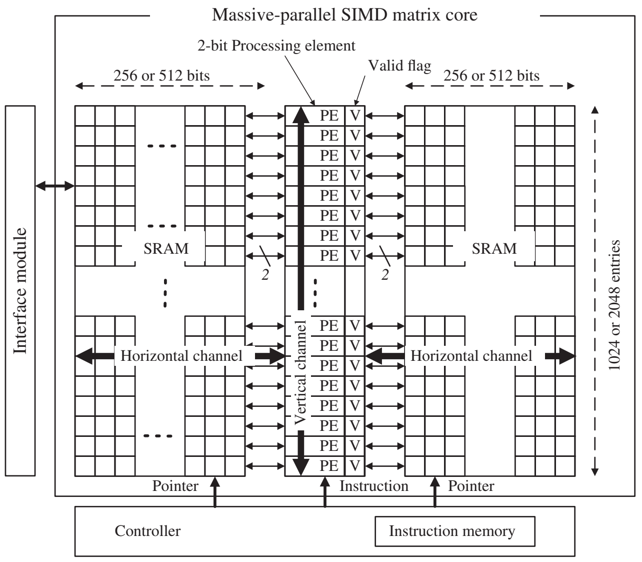

Fig. 1. Block diagram of massive-parallel SIMD matrix architecture

3. Interleaved-Bitslice Implementation for Parallel Block Cipher Processing with MX-1

Software block cipher processing will not be sufficiently fast and secure with digital-convergence mobile devices. Thus, a fast and secure parallel encryption/decryption implementation method is needed.

This section first introduces three types of conventional implementations for parallel block cipher processing in mobile devices: hard-wired and reconfigurable, bitslice, and interleaved. Next, the proposed interleaved-bitslice parallel block cipher processing method with MX-1 is described in detail.

3.1. Conventional parallel block cipher studies

3.1.1. Hard-wired/reconfigurable parallel cipher implementation Various researchers have proposed block cipher algorithms in different parallel implementations, which are based on a hard-wired circuit and a reconfigurable processor [15–19]. A block cryptographic LSI, such as an AES application-specific integration circuit (ASIC), contains the same ten units, which can execute one round of the algorithm. Ten rounds of the algorithm are processed in parallel using an external pipelined design [16]. On the other hand, in reconfigurable logic, such as field-programmable gate array (FPGA) and network-on-chip (NoC), a pipelined implementation for block cipher encryption has been introduced that unrolls the loop of the block cipher algorithm and inserts registers between each round [17]. The default S-Box table-lookup is then decomposed into several small S-Box modules to transform a large amount of byte-data in parallel (Fig. 2). A parallel and pipeline execution method for the block cipher is implemented on an NoC [19]. This implementation is used for programmable implementation in which the parallelized and pipelined tasks are allocated by the mapping strategy to each PE.

Thus, the hard-wired cryptographic chip can enable fast processing for mobile devices. However, it is not completely secure against leakage of private information, so the implemented algorithm or cipher mode in the cryptography chip may have to be changed to a more secure algorithm at some point. The ASIC-based implementation of the cryptographic algorithm cannot be updated or modified. The FPGA and NoC-based implementations, which have expensive hardware and are unsuitable for mobile devices, also make it more difficult to change the current algorithm than with a software-based processor in real time.

3.1.2. Bitslice implementation A bitslice implementation of the block cipher algorithm is a potentially effective software cipher processing method [20–22]. It involves converting the block cipher algorithm into multiple bit-serial logical operations. When implemented on a software-based processor with an n-bit register width, each bit in the register executes as a 1-bit PE executing a different encryption. Therefore, n encryptions/decryptions are processed in parallel.

The concept of bitslice processing is shown in Fig. 3. For ease of explanation, plain text bit width is the same as that of register width. On an n-bit software-based processor, sequential processing (Fig. 3(a)) is executed with one-way n-bit logical operation, such as XOR and AND, between two plain texts. On the other hand, bitslice processing can execute n-way bit-serial word-parallel operations for n plain texts at the same time. The

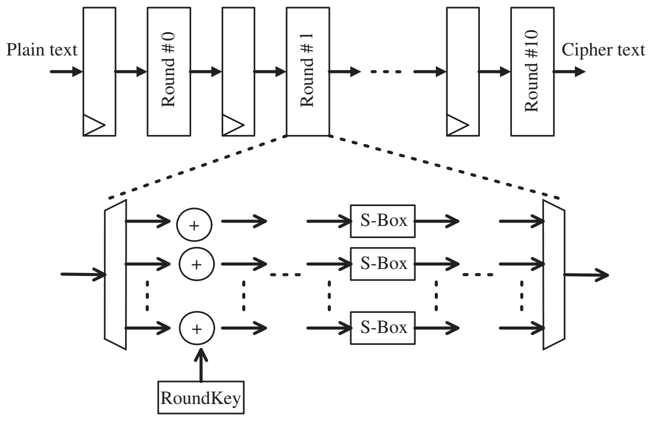

Fig. 2. Parallel implementation using unrolled loop of block cipher algorithm

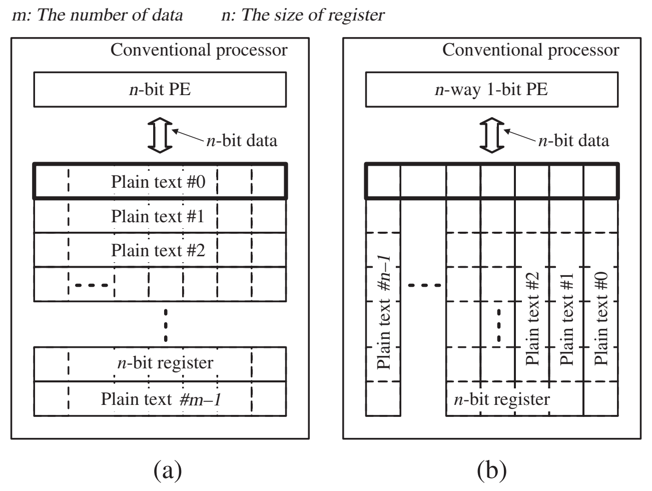

Fig. 3. Two block cipher implementation methods on conventional processor: (a) sequential processing, (b) bitslice processing

n-bit PE acts as an SIMD processor executing n-way 1-bit logical operations in parallel.

The bitslice operation can enable parallel block cipher processing by software programming. This technique has often been successful on a high-spec processor with many long registers. Hence, bitslicing cipher solutions are not widely used in real applications, because they are actually slow on several consumer devices without high-performance processors [22]. Moreover, bitslice processing requires a particular data format; thus, an additional format conversion process is also needed for compatibility with a conventional software program. Moreover, the bitsliced block cipher algorithm can only be used in particular modes that support parallelized encryptions, such as Electronic CodeBook (ECB) and CounTeR (CTR), and parallelized decryption, such as ECB, Cipher Block Chaining (CBC), Cipher FeedBack (CFB), and CTR, which the National Institute of Standards and Technology (NIST) has introduced [23].

3.1.3. Interleaved implementation With most block cipher modes, a data dependency relationship is found between current processing data and previous encrypted or decrypted data. This can often make it impossible to parallelize cipher processing. One solution is to interleave multiple cipher streams [24,25]. For example, this solution can execute parallelized CBC encryption, as shown in Fig. 4. While the ECB and CTR modes are directly suitable for parallel processing, the CBC, CFB, and Output FeedBack (OFB) modes in encryption generally consist of a sequential chain of operations and are more difficult to adopt for parallel

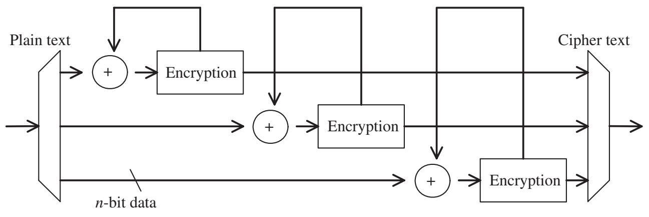

Fig. 4. Interleaved implementation-based parallel CBC encryption

processing. Instead of a single CBC chain, the interleaved CBC mode achieves three CBC encryption paths, as shown in Fig. 4. The first, fourth, and every third block thereafter are encrypted in CBC mode with one initial vector. The second, fifth, and every third block thereafter are encrypted in CBC mode with another initial vector, as is the other text block. Thus, the interleaved CBC mode executes three parallel CBC streams. This concept can also adapt to other cipher modes with any number of parallel streams within the limits of hardware capability.

3.2. Interleaved-bitslice implementation method

3.2.1. Basic idea To achieve parallelized block cipher processing with any cipher mode, we propose an interleaved-bitslice method. The proposed method is aimed at facilitating MX-1, and combines two concepts (bitslice and interleaved implementations) into a parallel cipher processing. This method can be compatible with both parallelized encryption and decryption and adaptive processing for all cipher modes. A parallel XOR example of interleaved-bitslice processing is shown in Fig. 5. All XOR operation paths are stored in m blocks in plain text, which are located at m positions with m different initial vectors. These m blocks are executed in bit-serial XOR operation independently in parallel. As a result, the encryption and decryption of the next m blocks in any cipher mode can start as soon as the previous blocks from m positions have been processed. With these refinements, the same processing performance as in ECB mode is possible.

3.2.2. Implementation on MX-1 This subsection presents a software-based parallel cipher processing solution with MX-1 for fast logical operation and low power consumption on digital-convergence mobile devices. The data width of a widely used conventional processor is limited to 32-, 64- or 128-bits and calculates 32-, 64- or 128-bit mathematical operations in a single access. On the other hand, MX-1 can execute 2-bit 1024 or 2048-way bit-serial and word-parallel operations. In the block cipher algorithms, 64- or 128-bit data often must be handled, and the main operation is bit-wise Boolean functions, such as AND or XOR. These bit-oriented data operations in cipher processing are also unsuitable for a conventional processor [26]. On the other hand, MX-1 is suitable for bit-serial and word-parallel operations. Thus, we explain the interleaved-bitslice parallel block cipher processing method on MX-1 in detail. Figure 6 shows the processing concepts of the interleaved-bitslice processing method for highly parallel block cipher implementation with MX-1. The magnitude of cipher parallelism for a conventional processor depends on the register size n (Fig. 3). The n used in the conventional processor is often 32, 64, or 128 bits. On the other hand, MX-1 can store 1024 or 2048 128-bit plain text blocks in a one-dimensional bitslice line format to take advantage of its high parallelism. The AES algorithm consists of four elementary transformations called SubBytes, ShiftRows, MixColumns, and AddRoundKey, while the decoding algorithm uses the inverse of these four elementary transformations called InvSubBytes, InvShiftRows, InvMixColumns, and InvAddRoundKey. The AddRoundKey transformation example is shown in Fig.

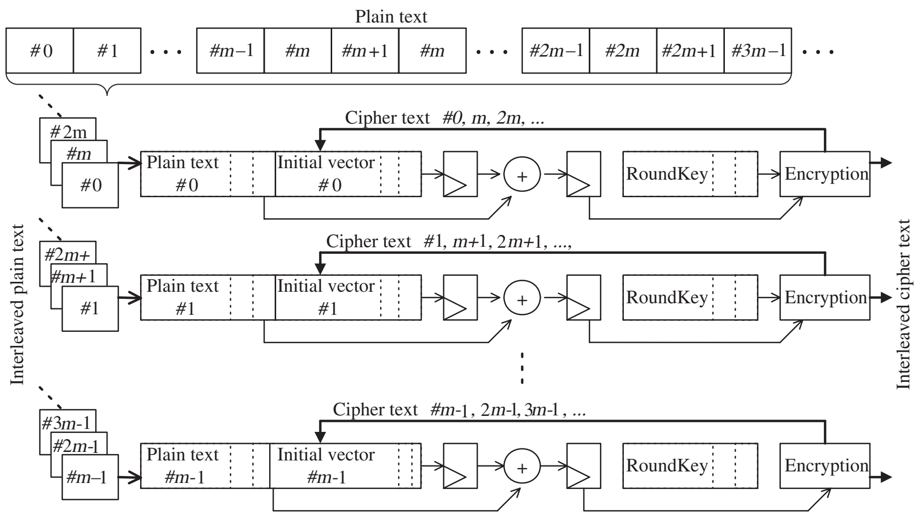

Fig. 5. Parallel cipher processing using proposed interleaved-bitslice parallel block cipher processing method

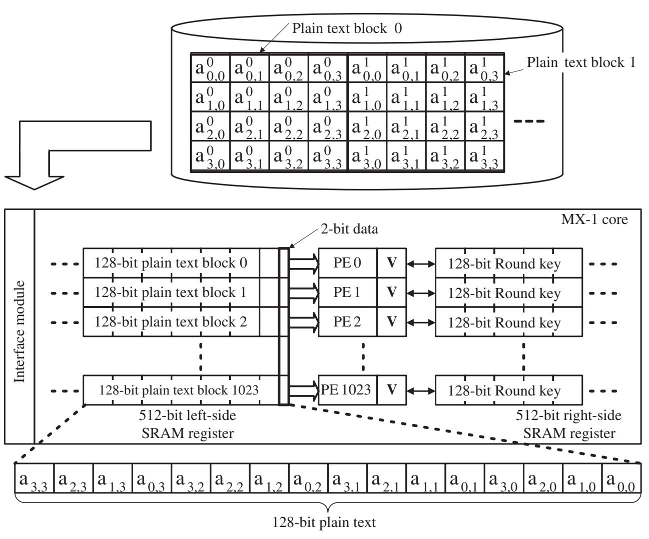
Plain text block 0
Plain text block 1
MX-1 core
Interface module
2-bit data
128-bit plain text block 0
128-bit plain text block 1
128-bit plain text block 2
128-bit plain text block 1023
PE 0
PE 1
PE 2
PE 1023
V
128-bit Round key
512-bit left-side SRAM register
512-bit right-side SRAM register
128-bit plain text

Fig. 6. Processing concept of interleaved-bitslicing on MX-1

6. Additionally, the Federal Information Processing Standardization (FIPS)-approved block ciphers can support five confidentiality modes: ECB, CBC, CFB, OFB, and CTR. To show the effectiveness of highly parallel five confidentiality mode processing with MX-1, this section summarizes the CBC operation, an example of interleaved-bitslice processing with MX-1. Figure 7 shows the procedure for 1024-way interleaved-bitslice CBC encryption on MX-1. To explain effective parallel processing, CBC encryption flow is divided into four steps. The MX-1 operations corresponding to each flow step are explained below.

Step 1 1024 PEs in the massive-parallel SIMD matrix core can execute bit-serial XOR operation between plain text blocks (P0, P1, ···, P1023) and the initial vector (IV0, IV1, ···, IV1023).

Step 2 1024 XOR-results (R0, R1, ···, R1023) are encrypted independently by using the ECB algorithm. The CBC cipher text blocks (C0, C1, ···, C1023) are generated in the MX core.

Step 3 The CBC cipher text (C0, C1, ···, C1023) and next plain text blocks (P1024, P1025, ···, P2047) are executed XOR operations in 1024 PEs.

Step 4 Step 3 results (R1024, R1025, ···, R2047) are the processed ECB encryption in 1024 PEs.

After these steps, the MX-1 repeats and continues the above steps for interleaved-bitslice CBC encryption flow. All entries of the MX-1 store 1024 plain text blocks and 1024 different initial vectors. Generally, the initial vectors serve as a randomizer. Such generated values are often referred to as nonces. Nonces merely must change for every encryption. The MX-1 can generate random numbers to execute pseudorandom number algorithms, such as a Mersenne twister [9], in parallel. The MX-1 prepares 1024 different initial vectors relatively easily in the SRAM area. These 1024 blocks are encrypted independently by using the CBC encryption flow. The encryption of the next 1024 blocks can start as soon as the previous encrypted blocks from the 1024 positions have been processed. When the decryption of cipher texts is processed, data compatibility is easily maintained on MX-1. Consequently, MX-1 is clearly effective for highly parallel execution of the practically important five confidentiality modes of the AES block ciphers.

4. Experimental Results for Block Cipher Parallel Processing with MX-1

This section reports on several experimental results, such as the clock cycle number of the five confidentiality modes for parallel block cipher processing, encryption throughput value, and energy efficiency with MX-1 and various well-known embedded processors. Furthermore, an improved logical operation library on MX-1 is proposed for more efficient parallel cipher processing.

Implementation targets from the AES algorithm are selected and implemented, and five confidentiality modes are chosen (ECB, CBC, CFB, OFB, and CTR), which are recommended in the special publication 800-38A from NIST[23].

4.1. Implementation results of five confidentiality modes with block ciphers on MX-1 The number of encryption and decryption clock cycles with the five confidentiality modes is measured using an MX-1 evaluation board. To obtain the number of clock cycles, the AES algorithm is implemented with a C language-based embedded program and executed using the proposed interleaved-bitslice parallel block cipher processing method on MX-1. Figure 8(a) and (b) shows a photograph and a system block diagram of the MX-1 evaluation board, respectively. The main components are the SIMD matrix core, host

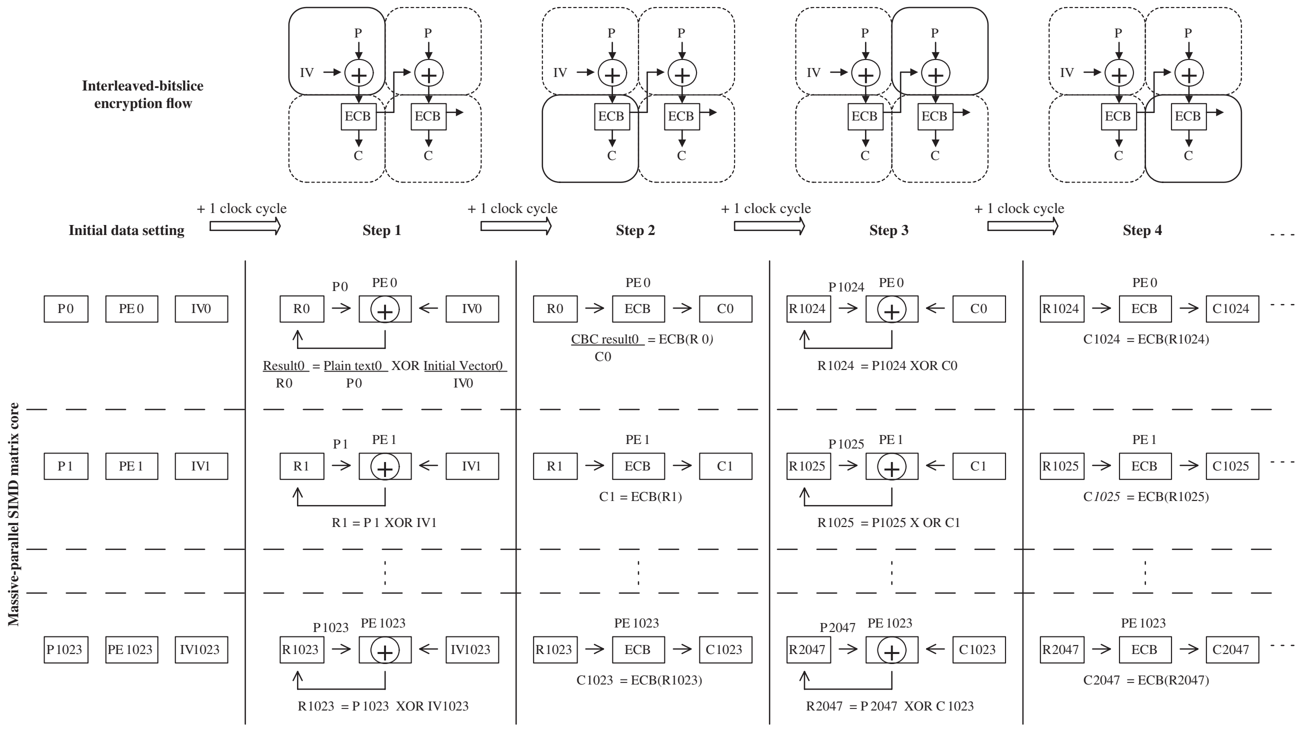

Fig. 7. Basic operations for CBC encryption with MX-1

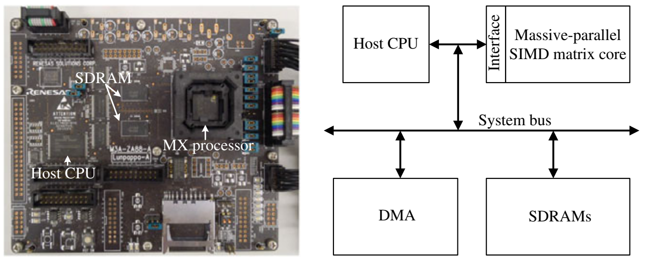

Fig. 8. Photograph and block diagrams of MX-1 evaluation board: (a) evaluation board photograph, (b) evaluation board architecture

CPU (M32R), direct memory access (DMA) controller, and two synchronous dynamic random access memories (SDRAMs). Maximum clock frequencies of the SIMD matrix processor, host CPU, and SDRAM are 162, 81, and 81 MHz, respectively. The specifications of this MX-1 processor are as follows: 1024-parallelism, 512-bit word length, 2-bit serial processing, and 150 mW power dissipation. The host CPU dispatches several tasks to the MX-1 core and the DMA controller and executes serial operations. The DMA controller transfers plain text data between the SDRAMs and SIMD matrix core through the system bus.

Tables I(a) and (b) list the number of encryption and decryption clock cycles per byte, respectively. Two parallelism architectures, 1024 and 2048, are measured and estimated with the AES algorithm, respectively. MX-1 can execute all transformations in block cipher algorithms within two SRAMs and PEs in MX-1. Thus, 2048 parallelism results can become one-half of the 1024 parallelism results. Moreover, the value of four extension confidentiality modes is almost the same as that of the ECB mode in the interleaved-bitslicing effect.

4.2. Performance comparison between MX-1 and conventional mobile processors

For comparison with the processing speed requirements of plain text transformation, the

Table I. Implementation results of AES algorithm with five confidentiality modes on the MX-1: (a) encryption clock cycles, (b) decryption clock cycles

		(a) The number of encryption clock cycles/bytes				
Algorithm	Parallelism	ECB	CBC	CFB	OFB	CTR
AES	1 024	25.33	25.44	25.44	25.46	25.46
	2 048	12.66	12.72	12.72	12.73	12.73
		(b) The number of decryption clock cycles/bytes				
AES	1 024	28.59	28.80	25.44	25.46	25.46
	2 048	14.29	14.40	12.72	12.73	12.73

necessary values of encryption and decryption clock cycles per byte for the AES algorithm are calculated using the 2048 parallelism MX-1 (81 MHz) and four conventional mobile processors. These are Intel Atom N270 (1.60 GHz); AMD Geode LX800 (500 MHz); Texas Instruments (TI) DM3730 (1.00 GHz), which exploits the Advanced RISC Machines (ARM) Cortex A8 core 32K/32K; and TI OMAP3530 (720 MHz), which exploits the ARM Cortex A8 core 16K/16K. Table II shows the AES encryption and decryption processing efficiency with five confidentiality modes expressed as the number of clock cycles per byte with the above five processors. To verify the normal operation condition of these processors, conventional processors and MX-1 (2048 parallelism) are applied with commercial sequential processing (Fig. 3(a)), which is executed using the open C-language source [27] and the interleaved-bitslice parallel block cipher processing method, respectively. The obtained results show that low-power-specific embedded processors, such as Atom, Geode, and TI products, require over 100 clock cycles. On the other hand, MX-1 can execute 2048 interleaved-bitslice processing and satisfy AES processing with both low power consumption and fast cipher processing. The number of clock cycles per byte of MX-1 can be reduced by up to 93% that of other mobile processors. Furthermore, while

Table II. Comparison of AES algorithm with five confidentiality modes for MX-1 and conventional mobile processors

	The number of AES encryption clock cycles/bytes					The number of AES decryption clock cycles/bytes				
Processor	ECB	CBC	CFB	OFB	CTR	ECB	CBC	CFB	OFB	CTR
Intel Atom N270	115.79	124.50	120.48	119.40	128.74	116.39	130.35	121.17	119.40	128.74
AMD Geode LX800	97.63	105.96	102.47	100.51	110.17	97.88	110.84	103.41	100.51	110.17
TI DM3730 (ARM Cortex A8)	171.27	179.60	177.02	175.82	186.25	172.10	184.88	177.98	175.82	186.25
TI OMAP3530 (ARM Cortex A8)	175.85	184.96	181.95	182.56	192.69	177.24	190.68	183.40	182.56	192.14
MX-1 (2,048-parallel), This work	12.66	12.72	12.72	12.73	12.73	14.29	14.40	12.72	12.73	12.73

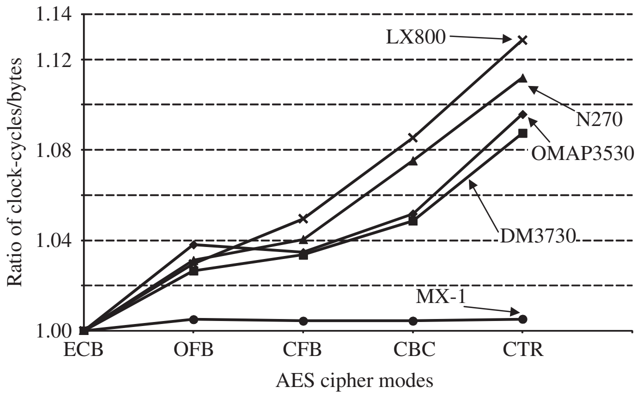

Fig. 9. Transition chart of AES algorithm with five confidentiality modes for MX-1 and conventional mobile processors

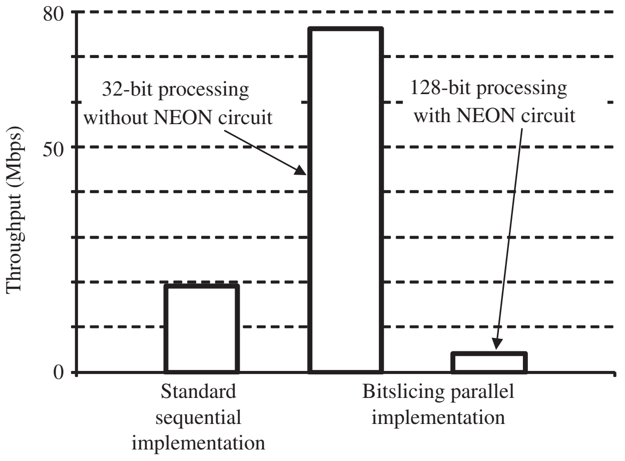

Fig. 10. Comparison of ARM processing results for AES algorithm with/without NEON circuit

the results of the conventional processors gradually increase in accordance with the mode condition, as shown in Fig. 9, for visualizing the results ratios in Table II, those of MX-1 can be almost constant for all AES modes, which is indicated on the vertical axis. Thus, the interleaved-bitslice parallel block cipher processing method with MX-1 can enable effective highly parallel block cipher processing.

This section also verifies the practical effectiveness of bitslice-based parallel cipher implementation (see Section 3.1.2) for mobile embedded processors. In previously reported bitslice implementations [20–22], one software logical instruction is considered as a parallel bit-serial logical operation of n hardware logical gates, as shown in Fig. 3(b). Hence, bitslice processing tends to be efficient when the entire hardware implementation for block cipher algorithms is not complex and processor architecture has many long registers. Therefore, effective bitslice implementation results have been reported that use high-end RISC processors. However, modern mainstream processors, which apply x86 architecture, are often not suitable because of the small number of registers [22]. While the recent Intel Core processor series achieves fast block cipher processing, the effectiveness of the bitslice implementation with mobile embedded processors has not been sufficiently reported. This section discusses the calculation of the encryption throughput value for the AES algorithm by using two mobile embedded processors: ARM Cortex-A8 at 1 GHz, which is a well-known conventional embedded processor for digital-convergence devices and exploited on the BeagleBoard-xM evaluation board [28]; and the MX-1 evaluation board (Fig. 8). The ARM Cortex-A8 has a fixed instruction circuit of 32 bits and the NEON general-purpose 128-bit SIMD engine [29]. We chose these two boards because they are of similar construction, which includes a host CPU and an SIMD circuit. Figure 10 shows three types of throughput results of standard sequential implementation and two bitslicing parallel implementations with ARM Cortex-A8. For 32-bit processing without the NEON unit, the bitslicing result is about 4 times faster than the sequential result. However, the 128-bit processing result with the NEON circuit is about 0.2 times lower than the sequential one. This is because, in the case of 128-parallelism implementation using the NEON circuit, n in the NEON unit is shorter than that of 128-way bitslicing implementation. Thus, the data transfer process occurs frequently during bitslice cipher processing. Figure 11 shows two circular percentage graphs for total NEON instruction operation (veor, vld, vmov, and vst). The vmov command, which is a data transfer operation between the ARM core and the NEON circuit, in 128-parallelism implementation increases drastically and is about 42% larger than that of 32-parallelism.

To increase the degree of parallel processing, MX-1 applies a bit-serial and word-parallel mode of operation, directly connecting many small PEs to SRAM arrays. Thus, the MX-1 can be expected to reveal the effectiveness of the bitslicing concept. Figure 12 shows the 1024-parallelism throughput value for the MX-1 with four patterns of operating frequency. The results at 81 MHz were obtained using the evaluation board, and the other results were obtained from a C-language-based HEW simulator dedicated to MX-1 [30]. The MX-1 throughput at 200 MHz, the operating frequency of which was referred to in a previous report [5], is nearly the same as that of ARM Cortex-A8 at 1 GHz. Moreover, if MX-1 executes AES encryption at 1 GHz, the throughput value can achieve about 300 Mbps and is about 4 times faster than that of ARM Cortex-A8 at 1 GHz.

Generally, low power dissipation is a required capability for mobile devices. Table III also shows the energy efficiency expressed in Mbps/mW of the above processors. The MX-1, which is operated using the evaluation board and HEW simulator, consists

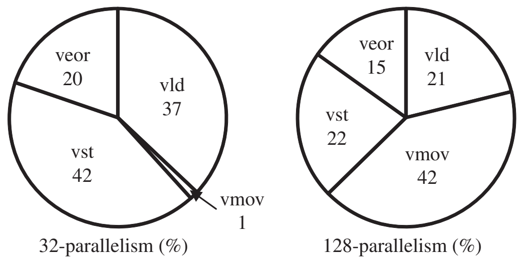

Fig. 11. Percentage graphs for the executed NEON command in bitslice AES encrypting for the ARM processor

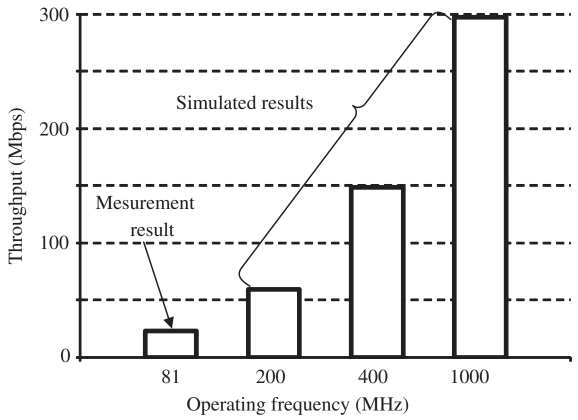

Fig. 12. Interleaved-bitslice implementation results of AES algorithm on MX-1

Table III. Comparison of MX-1 and ARM processors in terms of energy efficiency for parallel AES processing

Processor	M32R + MX core (Evaluation board)	SH-2A + MX core (HEW simulator)	TI DM3730 (ARM AM3715 Coretex-A8)
CMOS process (nm)	90	90	65
Operating frequency (MHz)	81	200	1000
Power consumption (mW)	200	124	731
Throughput (Mbps)	22.68	59.51	76.28
Energy efficiency (Mbps/mW)	0.11	0.48	0.10

of an SIMD matrix core and two types of CPUs. The total power consumption was measured and estimated to be about 200 mW at 81 MHz and 124 mW at 200 MHz. On the other hand, the power consumption of the DM3730, which is based on ARM Cortex-A8, was calculated, using the Power Estimation Spreadsheet [31], to be about 731 mW. In comparison, the energy efficiency of the MX-1 can be up to 4.8 times higher than that of an ARM-based processor.

According to the results from the above evaluations, interleaved-bitslice processing with MX-1 can achieve highly parallel cipher operation and a low-power and efficient solution for mobile devices.

4.3. Performance comparison between MX-1 and those from related studies Block ciphers are repeated any number of times in several logical operations, such as XOR, AND, OR, and INV, for encrypting and decrypting plain text. Thus, for effective parallel cipher processing, logical microcode libraries, which are dedicated to MX-1, need to be designed to correspond to any condition. However, the current MX-1 library only supplies 1-bit processing logical functions. For example, if MX-1 executes an n-bit long XOR operation, the following XOR function has to repeat n times.

Table IV. Comparison of reduction ratios for MX-1 libraries

Transformation	The number of instruction clock cycles: mx_xor	mx_xor_h	Reduction ratio (%)
AddRoundKey	618	150	76
SubBytes	36 853	34 165	7
MixColumn	4 586	2 295	50

```
//**1-bit XOR function in MX-1 library**//
 void mx_xor(int ap, int bp);
   ap: bit pointer of storage location
   bp: bit pointer
```

The number of clock cycles is then calculated from (1). The variables $C1$, $C2$, and $C3$ correspond to the "register setting", "read from SRAM", and "execution in PE and write to SRAM", respectively.

$$(C_{mx\_xor}) = (C1 + C2 + C3) \times n \tag{1}$$

For improving the operation of 1-bit logical functions, we propose the development of a multibit logical function to substantially reduce the number of cipher clock cycles. The multibit XOR function (mx_xor_h) can be represented in the following function:

```
//**multi-bit XOR function in MX-1 library**//
  void mx_xor_h(int ap, int bp, int n);
    ap: bit pointer of storage location
    bp: bit pointer
    n: bit length
```

If this function is added to the MX-1 library, the number of clock cycles can be estimated from (2):

$$(C_{mx\_xor\_h}) = C1 + C2 + C3 + n \tag{2}$$

Therefore, the number of clock cycles for three transformations, which are used in multibit logical functions, in the AES algorithm can be up to 79% smaller than that for current logical-function-based transformations (see Table IV).

To verify the effectiveness of the multibit XOR function, MX-1, which exploits the mx_xor_h, was compared with those from four cipher-related studies [32–35] for digital-convergence mobile devices in Table V. Since each result varies according to the implementation conditions, the number of clock cycles per byte was used for an objective and unified evaluation. As a result, MX-1 showed the smallest value of these functions as well as effective processing capability for digital-convergence mobile devices.

5. Cryptographic Implementation Capability Discussion of MX-1 for Digital-Convergence Devices

As an expression of the MX-1 capability, which has the high flexibility of software processing and low power consumption for digital-convergence mobile devices, other cryptographic implementation results are also discussed in this section for reference. The DES and triple-DES algorithms are implemented on the MX-1 for efficient interleaved-bitslice cipher processing. The DES algorithm is a previous standard cryptographic algorithm but is no longer considered for providing the data security needed to protect a large amount of information. On the other hand, the triple-DES algorithm is an important algorithm and is based on the DES algorithm. It uses three keys and encrypts the data 3 times by using

Table V. Comparison of MX-1 and four cipher-related studies in terms of the number of clock cycles per byte for digital-convergence mobile devices

Reference	Architecture	AES (ECB)
Nadehara, *et al.* (2004) [36]	Extentned MIPS	12.50
Gonzalez, *et al.* (2006) [37]	MBlaze-D FSL	12.62
Irwansysh, *et al.* (2009) [38]	NiosII + TC hardware	13.31
Khaddour, *et al.* (2009) [39]	MicroBlaze TM + 16 processors	15.62
This work	MX-1 (multi-bit XOR) 2 048-parallel	11.12

Table VI. Implementation results of DES and triple-DES algorithms with five confidential modes on MX-1: (a) encryption clock cycles, (b) decryption clock cycles

		(a) The number of encryption clock cycles/bytes				
Algorithm	Parallelism	ECB	CBC	CFB	OFB	CTR
DES	1 024	23.11	23.23	23.23	23.27	23.27
	2 048	11.56	11.62	11.62	11.63	11.64
Triple-DES	1 024	69.03	69.15	69.15	69.18	69.19
	2 048	34.51	34.57	34.57	34.59	34.59
		(b) The number of decryption clock cycles/bytes				
DES	1 024	23.11	23.23	23.23	23.26	23.27
	2 048	11.56	11.61	11.61	11.63	11.64
Triple-DES	1 024	69.02	69.14	69.14	69.17	69.18
	2 048	34.51	34.57	34.57	34.59	34.59

the single-DES algorithm. Though the triple-DES algorithm takes more time than the DES algorithm, it is more secure. Thus, the triple-DES and AES algorithms have adopted the FIPS-approved cipher algorithms.

Table VI(a) and (b) lists the number of encryption and decryption clock cycles per byte, respectively. Two parallelism architectures, 1024 and 2048, are measured using the DES and tripe DES algorithms on the MX-1 evaluation board. Thus, the clock cycles of the triple-DES algorithm are about 3 times larger than that of the DES algorithm. The results of 2048 become one-half those of the 1024 parallelism.

Table VII also shows energy efficiency expressed in Mbps/mW of the MX-1 evaluation board. Although the throughput and energy efficiency of the triple-DES algorithm is about 0.41 and 0.45 times smaller than those of the AES algorithm, respectively, MX-1 can flexibly execute three common block cipher algorithms by software programming.

6. Conclusion

We presented secure data processing with the MX-1 for embedded SoC in digital-convergence mobile devices. This cipher processing is based on the proposed interleaved-bitslice parallel block cipher processing method, which is compatible with both parallelized encryption and decryption and adaptive processing for five confidentiality modes. It can use three parallel FIPS-approved block cipher algorithms: DES, triple-DES, and AES. For the AES algorithm, the number of clock cycles per byte with the MX-1 implementation is reduced by up to 93% of that with the other four mobile processors. Additionally, MX-1 results were almost

Table VII. Results of MX-1 in terms of energy efficiency for parallel DES and triple-DES processing

	DES	Triple-DES	DES	Triple-DES
Algorithm	Encryption		Decryption	
Parallelism	1024		1024	
CMOS process (nm)		90		90
Operating frequency (MHz)		81		81
Power consumption (mW)		200		200
Throughput (Mbps)	28.04	9.39	28.04	9.39
Energy efficiency (Mbps/mW)	0.14	0.05	0.14	0.05

constant for all confidentiality modes. If energy efficiency is compared between MX-1 and ARM cortex-A8, the former can be 4.8 times more efficient. To improve the operation of single-bit logical function, we also proposed the development of a multibit logical library for interleaved-bitslice cipher processing with MX-1. Thus, the number of clock cycles is smallest among those reported in other related-studies. Consequently, the interleaved-bitslice parallel block cipher processing method with five confidentiality modes on MX-1 is an effective solution for digital-convergence mobile devices.

Acknowledgments

The reported results were obtained as part of collaborative research undertaken with Renesas Electronics Co. We are deeply grateful to Messrs. K. Murata H. Noda, H. Yoshikawa, and T. Takeda for their support and encouragement. Part of this work has been supported by a Grant-in-Aid for Young Scientists (B) (No. 24710190), Ministry of Education, Culture, Sports, Science and Technology, Japanese government.

References

(1) Uchiyama K. Processor technologies for system LSI's. *Journal of IEICE* 2012; **95**(**7**):582–588.

(2) Abe Y, Ikeda H, Emura M. Security technologies for smartphones. *Magazine FUJITSU* 2012; **65**(**5**):603–609.

(3) Phifer L. How mobile device encryption works to protect sensitive data. *TechTarget.* 2015. http://searchconsumerization.techtarget.com/tip/How-mobile-device-encryption-works-to-protect-sensitive-data. Accessed December 25, 2013.

(4) Intel Corporation. Securing the Enterprise with Intel AES-NI. *TechTarget* 2015. http://www.intel.com/content/dam/doc/white-paper/enterprise-security-aes-ni-white-paper.pdf.

(5) Nakajima M, Noda H, Dosaka K, Nakata K, Higashida M, Yamamoto O, Mizumoto K, Kondo H, Shimazu Y, Arimoto K, Saitoh K, Shimizu T. A 40GOPS 250 mW massively parallel processor based on matrix architecture. *ISSCC Digest of Technical Papers.* 2006; 410–412.

(6) Kumaki T, Koide T, Mattausch HJ, Kuroda Y, Gyohten T, Noda H, Dosaka K, Arimoto K, Saito K. Integration architecture of content addressable memory and massive-parallel memory-embedded SIMD matrix for versatile multimedia processor. *IEICE Transactions on Electronics* 2008; **E91-C**(**9**):1409–1418.

(7) Kumaki T, Imai Y, Hiramoto H, Koide T, Mattausch HJ. Realization of efficient and low-power parallel face-detection with massive-parallel memory-embedded SIMD matrix. *Proceedings of IEEE International Midwest Symposium on Circuits And Systems (MWSCAS'10).* 2010; 359–362.

(8) Kumaki T, Osawa M, Itaya S, Ogura T, Fujino T. Decomposition/reconstruction acceleration of max-plus algebra morhological wacelet transform with massive-parallel SIMD matrix processor. *Journal of Signal Processing* 2011; **15**(**6**):425–434.

(9) Mochizuki Y, Yoshida N, Matsumoto N, Murakami Y, Kumaki T, Fujino T. Parallel processing implementation and evaluation of

mersenne twister with SIMD embedded processor. *IEICE Transactions on Information and Systems* 2012; **J95-D**(**3**):376–386.

(10) Honda T, Mochizuki Y, Kumaki T, Fujino T. Implementation and evaluation of data-parallelized CryptMT stream cipher with SIMD embedded processor. *IEICE Transactions on Information and Systems* 2013; **J96-D**(**3**):495–505.

(11) Kuroda I, Kyo S. Media processing LSI architectures for automotives-challenges and future tends. *IEICE Transactions on Electronics* 2007; **E90-C**(**10**):1850–1857.

(12) Amano H. A survey on dynamically reconfigurable processors. *IEICE Transactions on Communications* 2006; **E89-B**(**12**):3179–3187.

(13) Kumaki T, Ishizaki M, Koide T, Mattausch HJ, Kuroda Y, Noda H, Dosaka K, Arimoto K, Saito K. Acceleration of DCT processing with massive-parallel memory-embedded SIMD matrix processor. *IEICE Transactions on Information and Systems* 2007; **E90-D**(**8**):1312–1315.

(14) Kurafuji T, Haraguchi M, Nakajima M, Gyoten T, Nishijima T, Yamasaki H, Imai Y, Ishizaki M, Kumaki T, Okuno Y, Koide T, Mattausch HJ, Arimoto K. A scalable massive parallel processor for real-time image processing. *ISSCC Digest of Technical Papers.* 2010; 15–17.

(15) Nastou P, Stamatiou YC. Enhancing the security of block ciphers with the aid of parallel substitution box construction. *Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops (ICDCSW02).* 2002.

(16) Kotturi D, Yoo SM, Blizzard J. AES crypto chip utilizing high-speed parallel pipelined architecture. *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS'05).* 2005; 4653–4656.

(17) Huang CW, Chang CJ, Lin MY, Tai HY. The FPGA implementation of 128-bits AES algorithm based pn four 32-bits parallel operation. *First International Symposium on Data, Privacy and E-Commerce (ISDPE).* 2007; 462–464.

(18) Dai Z, Li W, Yang X, Chen T, Ren Q. The research and implementation of reconfigurable processor architecture for block cipher processing. *Proceedings of International Conference on Embedded Software and Systems (ICESS).* 2008; 587–594.

(19) Yang YS, Bahn JH, Lee E, Bagherzadeh N. Parallel and pileline processing for block cipher algorithm on a network-on-chip. *Proceedings of International Conference on Information Technology: New Generations.* 2009; 849–854.

(20) Biham E. A fast new DES implementation in software. *ITechnion-Computer Science Department Technical Report.* 1997; (CS0891).

(21) Rebeiro C, Selvakumar D, Devi ASL. Bitslice implementation of AES. *Lecture Notes in Computer Science.* 2006; 203–212.

(22) Matsui M, Nakajima J. On the power of bitslice implementation on Intel Core2 processor. *Lecture Notes in Computer Science.* 2007; 121–134.

(23) Recommendation for Block Cipher Modes of Operation. *Special Publication 800-38A (SP 800-38A), 2001 ed. National Intstitute of Standards and Technology (NIST),* 2001.

(24) Schneier B. *Applied Cryptography*. 2nd ed. John Wiley & Sons, Inc: New Jersey; 1996; 210–211.

(25) Viega J, Messier M. Secure Programming Cookbook for C and C++. *O'Reilly Media*: Tokyo, Japan; 2003; 208–211.

(26) Hirose Y, Saito M, Xouzijn WES. Embedded micro processor with reconfigurable functional unit. *IEICE Transactions on Electronics* 2003; **J76-C**(**8**):808–816.

(27) http://free.pjc.co.jp/AES/index.html. Accessed March 1, 2012.

(28) http://beagleboard.org/. Accessed April 7, 2011.

(29) http://www.arm.com/products/processors/technologies/neon.php. Accessed October 5, 2011.

(30) http://www.renesas.com/products/tools/ide/ide_hew/ide_hew_tools_product_landing.jsp. Accessed April 7, 2011.

(31) http://processors.wiki.ti.com/index.php/AM/DM37x_Power_Estimation_Spreadsheet. Accessed June 25, 2013.

(32) Nadehara K, Ikekawa M, Kuroda I. Extended instructions for the AES cryptography and their efficient implementation. *Proceedings of IEEE International Conference on Signal Processing Systems.* 2004; 152–157

(33) Gonzalez I, Gomez-Arribas FJ. Ciphering algorithms im MicroBlaze-based embedded systems. *IEE Proceedings-Computers and Digital Techniques* 2006; **153**(**2**):87–92.

(34) Irwansyah A, Nambiar VP, Khalil-Hani M. An AES tightly coupled hardware accelerator in an FPGA-based embedded processor core. *Proceedings of IEEE International Conference on Computer Engineering and Technology.* 2009; 521–525.

(35) Khaddour M, Wang Z, Hammami O. Implementing block cipher on embedded multiprocessors platform. *Proceedings of IEEE International Conference on Multimedia Computing and Systems.* 2009; 193–198.

Takeshi Kumaki (Member) received the B.S. degree from the Department of Mathematics, Faculty of Science, and completed the first half of the M.E. program in information mathematics from the National Defence Academy, Kanagawa, Japan, in 1998 and 2003, respectively. He received the Ph.D. degree in electric engineering from Hiroshima University, Hiroshima, Japan, in 2006. From 2003 to 2004, he was with the Japan Air Self-Defence Force Electric Experimentation Group. From 2005 to 2009, he was with the Research Center for Nanodevices and Systems (RCNS) and the Research Institute for Nanodevice and Bio Systems (RNBS), Hiroshima University, Japan, where he was engaged in system design and architecture research. From 2010 to 2012, he was an Assistant Professor with the Department of VLSI System Design, Ritsumeikan University (RU). Since 2013, he has been a Lecturer with the Department of Electronic and Computer Engineering, RU. His research interests include content addressable memory, SIMD processing architecture, hardware Trojan detection, and their applications. Dr. Kumaki is a member of the Institute of Electrical and Electronics Engineers (IEEE) and the Institute of Electronics, Information and Communication Engineers of Japan (IEICE). Since January 2011, he has been serving as the Secretary for the Membership Development Committee (MDC) of the IEEE Kansai Section.

Tetsushi Koide (Non-member) was born in Wakayama, Japan, in 1967. He received the B.E. degree in physical electronics, and the M.E. and the Ph.D. degrees in systems engineering from Hiroshima University in 1990, 1992, and 1998, respectively. He was a Research Associate and an Associate Professor with the Faculty of Engineering, Hiroshima University, in 1992-1999 and 1999, respectively. From 1999, he was with the VLSI Design and Education Center (VDEC), University of Tokyo, as an Associate Professor. Since 2001, he has been an Associate Professor with the Research Center for Nanodevices and Systems, Hiroshima University. His research interests include system design and architecture issues for memory-based systems, real-time image processing, VLSI CAD/DA, genetic algorithms, and combinatorial optimization. Dr. Koide is a member of the Institute of Electrical and Electronics Engineers, the Association for Computing Machinery, the Institute of Electronics, Information and Communication Engineers of Japan, and the Information Processing Society of Japan.

Takeshi Fujino (Non-member) received the B.E., M.E., and Dr. E. degrees in electronic engineering from Kyoto University, Kyoto, Japan, in 1984, 1986, and 1994, respectively. In 1986, he joined the LSI Research and Development Center, Mitsubishi Electric Corporation, where he had been engaged in the development of electron beam lithography and embedded DRAM circuit design. In 2003, he moved to Ritsumeikan University, Shiga, Japan, where he is currently a Professor with the Department of VLSI System Design. His research interests include low-cost SoCs using programmable logic and their applications for security system such as tamper-resistant LSIs.