An Adaptive Cryptographic Accelerator for IPsec on Dynamically Reconfigurable Processor

Yohei Hasegawa, Shohei Abe, Hiroki Matsutani, and Hideharu Amano
Graduate School of Science and Technology, Keio University
3-14-1 Hiyoshi, Kohoku, Yokohama, Kanagawa 223-8522, Japan
drp@am.ics.keio.ac.jp

Kenichiro Anjo
NEC Electronics Corporation
1753 Shimonumabe, Nakahara, Kawasaki, Kanagawa 211-8668, Japan

Toru Awashima
NEC System Devices Research Laboratory
1753 Shimonumabe, Nakahara, Kawasaki, Kanagawa 211-8668, Japan

Abstract

We propose a cryptographic accelerator for IPsec by using the NEC electronics' Dynamically Reconfigurable Processor (DRP). In our system, an embedded processor and DRP are integrated in a System-on-a-Chip (SoC) and multiple cryptographic tasks can be accelerated by DRP. Moreover, the virtual hardware mechanism, which dynamically changes its configuration data set, is introduced to realize more tasks on DRP. The evaluation results show that the throughput of each implemented cryptographic task outperformed a MIPS compatible embedded processor from 1.6 times to 7.8 times. In addition, it is shown that 80.7% of the run-time configuration overhead can be reduced by background configuration based on the double buffering method.

1 Introduction

Recently, the improvements in semiconductor technology made it possible to integrate all of system components, such as microprocessor, memory, and several controllers into a single chip called System-on-a-Chip (SoC). In particular, in mobile phones, PDAs, and other embedded mobile systems, application specific hardware and a simple embedded processor are coupled to perform image/audio processings, encryptions and other multimedia processings. In these systems, an application specific hardware for off-loading heavy weight processes of embedded processors is widely used to achieve both high performance and low power consumption.

However, such a specialized chip cannot be used for other purposes and new standards or techniques are hard to be built in after shipment. Besides, the increase of the development cost is a serious problem when the size and complexity of the system are increased. Since a various kind of new technologies including JPEG2000, H.264, and AES have been proposed one after another, it is not practical to develop the dedicated hardware for each of them.

For example, IP Security (IPsec) [13] has been popular even in embedded systems. It is a protocol suite that provides security feature to the standard Internet Protocol (IP) with combination of multiple encryptions and authentications. Since powerful computational capability is required to deal with data encryption and authentication processing, hardware implementation of them has been a common solution. However, designing special hardware corresponding to each cryptography results in the heavy increase of the development cost. In addition, it is difficult to cope with newly developed cryptographic algorithms.

Dynamically reconfigurable processors are expected to overcome these problems. Recent coarse grain dynamically reconfigurable processors [11, 9, 8, 12] have been developed to achieve high performance and flexibility with smaller cost compared with traditional programmable devices, such as Field Programmable Gate Arrays (FPGAs). These devices consist of coarse grain cells and are prospective for high performance especially for stream applications such as encryption, image/audio compression, and digital signal processings [14, 3].

Furthermore, dynamically reconfigurable processors are believed to be more suitable for mobile terminals whose hardware resource and battery capacity are strictly limited because of *time-multiplexed*

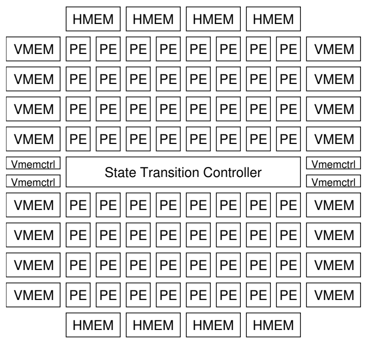


Figure 1. DRP Tile Architecture

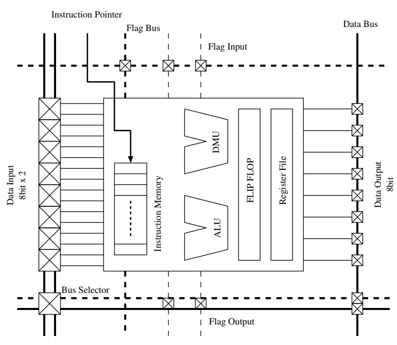


Figure 2. Processing Element (PE)

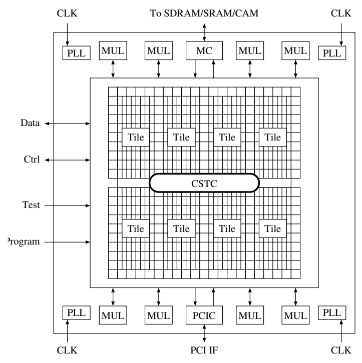


Figure 3. Prototype Chip: DRP-1

execution introduced by multi-context functionality. Multiple configuration data of circuits called *contexts* are stored in the internal configuration memory, and it can be rapidly (often in one clock) interchanged to implement different functions. This multi-context functionality reduces the physical die size and improves power efficiency. It divides a whole task into multiple contexts, and only one context which is required at that point should be activated and executed.

In this paper, we implemented and evaluated the cryptographic accelerator for IPsec based on NEC's Dynamically Reconfigurable Processor (DRP). Moreover, we propose a DRP-based system architecture that targets embedded fields. Our system provides high-throughput cryptographies, cost efficiency, and high flexibility to embedded systems that support IPsec communications.

The rest of this paper is organized as follows. Section 2 explains an architecture of DRP, which is a target device in this paper. In Section 3, we present our DRP-based embedded system architecture including design policies of the cryptographic accelerator. In Section 4, the implementation of experimental system utilizing a DRP evaluation board is described. The results of performance analysis and simulation results are shown in Section 5. Finally, Section 6 describes the previous work on FPGA-based IPsec implementations and this work is concluded in Section 7.

2 DRP Overview

DRP is a coarse grain dynamically reconfigurable processor that was released by NEC Electronics in 2002 [11]. It carries an on-chip configuration data, or contexts, and it dynamically reschedules between them to realize multiple functions with one chip.

Sixty-four of the most primitive 8-bit *processing elements (PEs)* are combined to form what is called a *Tile*, and the DRP core consists of an arbitrary number of these Tiles. Figure 1 is a sketch of a Tile in DRP. The DRP was designed with a focus on coupling with other cores on a SoC, and it has peripheral connections for such purposes. Within each Tile, there are 64 PEs, one State Transition Controller (STC), eight 2-ported memories (VMEMs: Vertical MEMories), four VMEM Controllers (VMCtrl), and four 1-ported memories (HMEMs: Horizontal MEMories). The number of Tiles can be expandable, horizontally and vertically.

The structure of a PE is shown in Figure 2. It has an 8-bit ALU, an 8-bit DMU (for shifts/masks), an 8-bit$\times$16-word register file (RFU), and an 8-bit flip-flop (FFU). These units are connected by programmable wires specified by instruction data (configuration data), and their bit-widths range from 8 Bytes to 18 Bytes depending on the location. PE has 16-depth instruction memories (e.g. 16 contexts) and supports multiple context operations. Its instruction pointer is delivered from the STC.

STC is a simple sequencer in which any finite state machine (FSM) can be stored. The STC has 64 states, and each state is associated with the instruction pointer. The FSM of the STC operates synchronized with the internal clock, and generates the instruction pointer for each clock cycle according to the state. Also, STC can receive event signals from PEs to branch conditionally. The maximum number of branches that can be specified from the PE is four.

As for the memory units, a Tile has eight 2-ported VMEMs on its right and left sides, and four 1-ported HMEMs on upper and lower boundary. The capacity of a VMEM is 8-bit$\times$256-word, and four VMEMs can be handled as a FIFO, using VMCtrl. HMEM is a single-ported memory and it has a larger capacity than the VMEM. It has 8-bit$\times$8K-word entries. Contents of these memories, flip-flops, and register files of PE are shared with the datapath of all the contexts. DRP Core, consisting of several Tiles, can change its contexts every cycle with the instruction pointer distributed from STCs. Also, each STC can run independently by programming different FSMs.

DRP-1, shown in Figure 3, is the prototype chip using DRP Core with 4$\times$2 Tiles. It is fabricated with 0.15-μm CMOS processes. It consists of 8 Tiles,

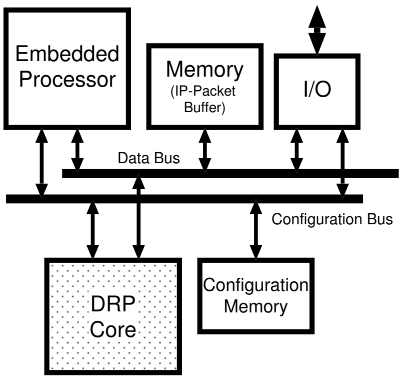


Figure 4. Target System Architecture

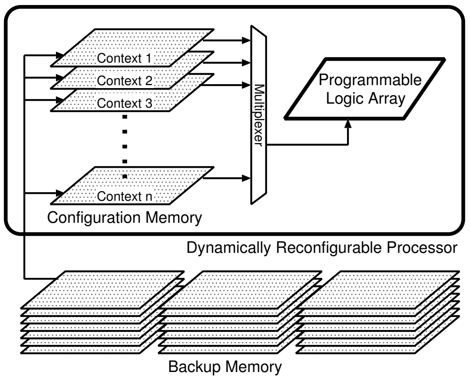


Figure 5. Virtual Hardware

eight 32-bit multipliers, an external SRAM controller, a PCI interface, and 256-bit I/Os. The maximum operation frequency is 100-MHz. Although DRP-1 is used as a stand-alone reconfigurable device, Tiles of DRP can be used as an Intellectual Property (IP) on SoCs with an embedded processor. In this case, the number of Tiles can be chosen so as to achieve the required performance with minimum area.

An integrated design environment, called Musketeer, is provided for DRP-1. It includes a high level synthesis tool, a design mapper for DRP, simulators, and a layout/viewer tool is provided. Applications can be written in a C-based high level hardware description language, synthesized, and mapped directly onto the DRP-1.

3 Cryptographic Accelerator Design

In this paper, we propose the DRP-based cryptographic accelerator for IPsec. We suppose the following system, considering that it is applied for embedded systems.

3.1 Co-Processing System Model

The recent SoCs consist of tightly coupled co-processors with a simple embedded processor. The embedded processor is used to implement control-intensive tasks, while the co-processor that is often dedicated hardware accelerates computation-intensive tasks of applications. The co-processing system has been widely used as a practical solution to optimize cost, performance, and power consumption.

However, designing special hardware customizing various kinds of tasks results in the heavy increase of the development cost. Besides, it is difficult to address new technologies which spring up one after another. In order to solve this problem in SoC, replacing some dedicated hardware cores with a single dynamically reconfigurable processor makes the system flexible and scalable.

Some coarse grain reconfigurable processors like DRP are also used as a co-processor of an embedded processor. The typical stream processing, which is a target of DRP, consists of some tasks, and the transfer of stream data between tasks is quite explicit. For this reason, it is effective to assign tasks to coarse grain reconfigurable datapath and then store and distribute the block of stream data into the internal distributed memory modules.

Hence, we treat a target system shown in Figure 4 here. This is a co-processing system which includes an embedded processor and DRP Core which consists of some Tiles of DRP. The embedded processor, DRP Core and a shared memory are connected by a high-speed on-chip data bus like NECoBus[5], and DMA transfer is supported. Moreover, DRP Core has a special I/O interface considering stream processing.

In the IPsec accelerator proposed here, multiple cryptographic tasks are accelerated by DRP Core, and the other control-intensive tasks including configuration control are executed by the on-chip embedded processor.

3.2 Virtual Hardware

Dynamically reconfigurable processors can realize various functions utilizing dynamic reconfiguration based on the multi-context feature. However, since the on-chip configuration memory is limited (16 contexts in the DRP-1), applications over this limitation cannot be executed.

In order to conquest this problem and implement more functions than the capacity of the on-chip configuration memory, a virtual hardware mechanism has been proposed [10, 4]. It is a technique which realizes large scale applications by dynamically changing the set of contexts from an external memory. As shown in Figure 5, this method allows reducing hardware cost by escaping unused contexts to the external memory and executes them in the time-multiplexed manner at task level. The virtual hardware is a useful technique to implement various tasks in embedded systems that have limited with computation resources.

We introduced the virtual hardware mechanism to our cryptographic accelerator so that can switch multiple cryptographic tasks on demand. In IPsec communications, it is required that multiple cryptographic tasks can be switched according to the situation. The configuration data corresponding to each cryptographic task is implemented individually. So, if users generate new configuration data of DRP, it is easy to add any cryptographic algorithms to the system.

As shown in Figure 4, an embedded processor controls the configuration access of DRP Core and switches cryptographic tasks by loading the configuration data dynamically. It is transferred by the dedicated configuration bus between DRP Core and the external configuration memory.

3.3 Run-Time Configuration based on Double Buffer

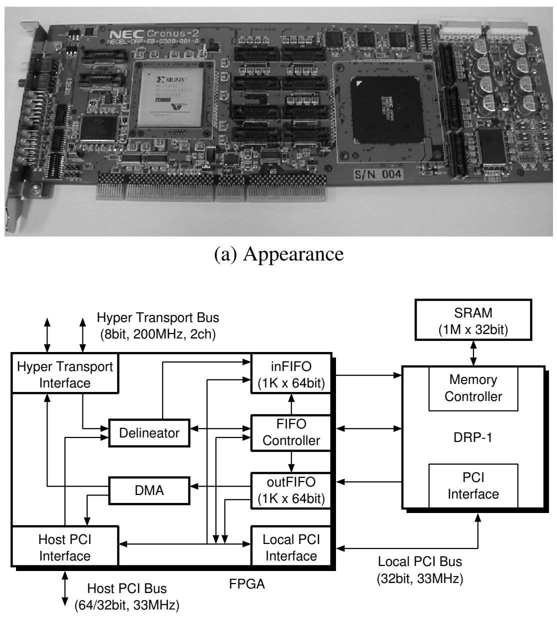
In the virtual hardware system, the dynamic loading time of configuration data sets can be a bottleneck of the system. Since task switching may be frequently occurred when many nodes communicate using several cryptographic tasks on IPsec, the configuration overhead is a critical problem in our cryptographic accelerator.

To address this problem, the run-time configuration method based on double buffering is introduced. In this method, at first, available contexts of DRP Core are divided into two groups. Then, a certain task is executed in one context group, while the configuration of the next task is done to the other context group. Consequently, the execution and the configuration of tasks are overlapped, and the configuration is completed in background.

In order to hide the run-time configuration time, however, the next configuration must be predictable. Furthermore, the next configuration must be finished during the execution of the current task to switch the task without inhibition of system operations. This overhead is evaluated later.

3.4 Acceleration Flow

The acceleration flow of our cryptographic accelerator is as follows. At first, the embedded processor checks IP packet headers in the IP-packet buffer. It determines whether the IPsec is applied for the packet. In IPsec, two communicating parties negotiate and establish a security association (SA) for protecting the transferred data preliminarily. An SA specifies the cryptography and the related keys to be utilized. The embedded processor decodes the IP header and finds out the corresponding SA from its database (SAD). After that, it searches the configuration data corresponding to the cryptography indicated by the SA, and sends its pointer and data to DRP Core.



(a) Appearance

(b) Block Diagram

Figure 6. DRP Evaluation Board

For example, if an SA indicates DES-CBC as an encryption of data, the configuration data is loaded to the DRP Core according to the pointer from the embedded processor. After the configuration of DRP Core, the embedded processor sends the data to the input buffer of DRP Core. Then, it starts processing of DES-CBC. The output data is written in the output buffer and the embedded processor pulls it. Finally, an IPsec packet is transferred to a network interface or an upper layer application.

Furthermore, for the run-time configuration based on the double buffer, the embedded processor checks the IP-packet buffer ahead and holds the event sequence of the future configurations. Thus, it is possible that the configuration for the next task is overlapped with the current task execution.

4 Experimental System

For the purpose of the verification, we established the experimental system for our cryptographic accelerator by using a DRP evaluation board.

4.1 DRP Evaluation Board

For the implementation and the verification of our cryptographic accelerator, we established the experimental system by using a DRP evaluation board shown in Figure 6. A DRP-1, an FPGA as I/O controller, a PCI interface, and an external SRAM are mounted on a PCI-card. In this experimental environment, DRP-1 is used as a DRP Core with 8 Tiles and a host PC controls the DRP-1 using a set of APIs. We developed and debugged the cryptographic accelerator on the environment with the host PC and the DRP-1. We could use this evaluation board as an emulation platform while it doesn't support run-time configuration at the moment.

4.2 Cryptography Designs

The cryptographic tasks implemented on DRP-1 are five private-key encryptions/decryptions: DES, AES (Rijndael), CAST-128, CAST-256, and RC6. We also implemented two one-way hash functions: MD5 and SHA-1. They are described by C-level description language, Behavioral Design Language (BDL), and compiled using the DRP compiler and the integrated development environment called Musketeer. The Cipher Block Chaining (CBC) mode, which is general in IPsec, is adopted as the operation mode for all of private-key encryption algorithms.

The DRP compiler supports two context scheduling methods, automatic and manual scheduling. the manual scheduling in which the timing of changing the state/contexts are directed manually by specifying timing directive in a code, while automatic scheduling mode does not need it. For the high-accuracy context scheduling, we adopted the manual scheduling.

There are various task distributions between embedded processor and DRP-1. Key schedulers for DES and AES are also supported on DRP-1, while software supports for the other algorithms. The AES, CAST-256, and RC6 modules are designed to be able to select the key-length flexibly. The MD5 and SHA-1 modules are designed to calculate the hash value of the original message by 512-bit blocks. The other operation of MD5 and SHA-1, such as zero-padding, and appending a message length are done by software. Furthermore, by processing with software and DRP-1, it is possible to compute the HMAC[2] which is an authentication algorithm used in IPsec.

4.3 Case Design: AES-CBC

A number of cryptographic tasks have been implemented on DRP-1 for the cryptographic accelerator proposed here. From the page limitation, an example, the implementation of AES-CBC is described in this section.

Advanced Encryption Standard (AES) [1] is a 128-bit encryption cipher which has been selected as a successor to the venerable Data Encryption Standard (DES) by the National Institute of Standards and Technology (NIST). The AES algorithm is capable of using cryptographic keys of 128, 192, and 256 bits to encrypt and decrypt data in blocks of 128 bits.

The AES is designed to use only simple whole-byte operations. In the case of 128-bit key, AES iterates some steps for 10 rounds. Each regular round involves four steps: Byte Sub, Shift Row, Mix Column,

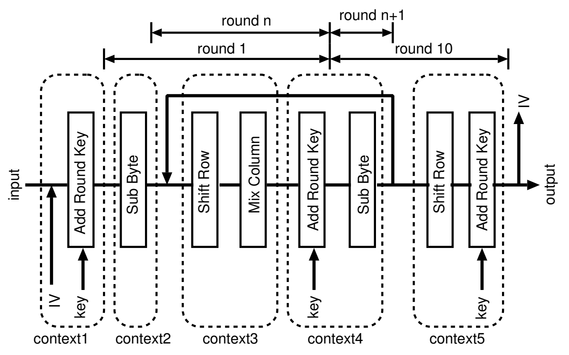


Figure 7. AES Context Scheduling

and Add Roundkey. Mix Column step is omitted in the final round. In the CBC mode, the expanded keys are used in each round and the 128-bit cipher block is fed back to the next encryption as Initialization Vector (IV).

The context scheduling of AES-CBC is shown in Figure 7. Since AES uses only 8-bit operations, DRP deals with it well due to the granularity of the PE and makes the most of the byte-level parallelism especially in Mix Column step. First is the Byte Sub step, where each byte of the block is replaced by its substitute called an S-box operation. We implemented the S-Box which uses VMEMs as LUTs to achieve a balanced design in resource and delay. Since the VMEM is a 2-ported memory and distributed around a Tile of DRP, required data sets can be read from the VMEM-based S-box simultaneously.

To hide the 1-clock latency for the memory access and the gate delay, round n and round $n + 1$ are mixed in the same context as shown in Figure 7. This causes the inefficient resource utilization, since the same functions are required in the different contexts. The number of the contexts is increased instead of the improvement of the performance.

5 Evaluation Results

5.1 Resource Usage and Performance

Table 1 shows the resource usage and throughput of each cryptographic task. The required contexts (*Ctxt*), the maximum number of required PEs (*Max*), the total utilization of PEs (*All*), and the maximum number of required VMEM modules (*Vm*) are summarized in this table. The value of *Max* is the maximum number of PEs used in the context. This means the number of PEs used spatially. In contrast, the value of *All* means the total number of PEs used in all contexts, i.e. the number of PEs used temporally.

One Tile of DRP-1 has 64 available PEs. So, all of cryptographic tasks are implemented within 2 Tiles of DRP-1 (128 PEs and 32 VMEMs). We can optimize the area efficiency by adopting the DRP core

Table 1. Resource Usage and Throughput

	Required Resources				Freq.	Throughput[Mbps]		
name	Ctxt	Max	All	Vm	[MHz]	DRP-1	DSP	MIPS
DES	16	121	1357	10	28.6	107.6	84.2	40.3
CAST128	9	32	140	20	27.4	109.8	95.4	46.7
AES	8	128	372	30	56.7	363.1	16.9	46.4
CAST256	16	38	327	21	28.9	37.4	36.1	23.2
RC6	6	56	164	8	34.0	198.1	96.6	76.9
MD5	7	75	279	14	24.7	194.2	135.5	95.5
SHA-1	8	65	337	20	35.1	202.1	44.0	38.7

with 2 Tiles for all cryptographic tasks. The context which utilizes the largest number of PEs includes a core round function of the encryption process, which is the critical path of all contexts.

Besides, we evaluated the throughput of each cryptographic task. Table 1 also shows the throughput on DRP-1 comparing with DSP and an embedded processor for each task. The throughput is measured from the minimum number of clock cycles which is required to operate 10000 blocks.

The throughput on DRP-1 is compared with a widely used DSP: Texas Instruments TMS320C6713. It is an 8-way VLIW DSP with floating operations that runs at 225MHz. 4-KB L1 instruction cache and 4-KB L1 data cache backed up with 256-KB L2 cache are equipped. A custom C compiler is used to compile the code written in C language. The options were set so as to maximize the throughput. We measured the throughput with all of data stored in the L1 data cache.

NEC's MIPS64 compatible VR5500 was also used for the evaluation. VR5500 is a 10-stage pipeline, 2-way out-of-order SuperScaler processor that runs at 400 MHz with 32KB instruction cache and 32KB data cache. Evaluation programs for each cryptographic task were written in C language and compiled with a MIPS compatible gcc cross-compiler in the T-Engine design environment. A '-O3' optimization option was used for compiling the program.

Evaluation result shows that the throughput of all cryptographic tasks on DRP-1 outperforms VR5500 from 1.6 times to 7.8 times. So, they can accelerate encryptions and authentications for IPsec communications. Because of the performance over DSP, DRP has advantages as an accelerator. In terms of resource usage, all cryptographic tasks have been implemented within 2 Tiles of DRP-1. If we use 2 Tile as DRP Core, it is possible to accelerate cryptographic tasks with improved area-efficiency. Furthermore, since the operation frequency is about 30MHz, the power consumption is expected to be reduced compared with other architectures.

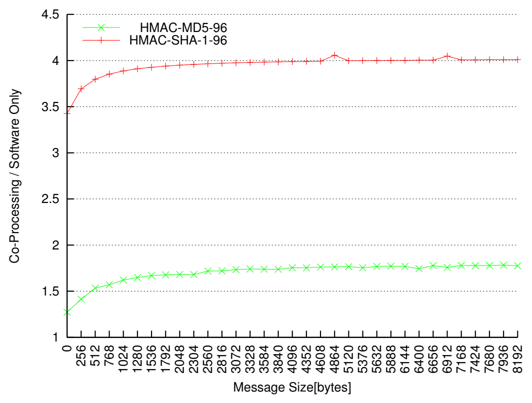


Figure 8. Performance Improvement

5.2 Co-Processing Evaluation

From evaluation results described in the previous section, it appears that cryptographic tasks can be efficiently accelerated by DRP Core with 2 Tiles. Here, we evaluated the performance of co-processing with an embedded processor and DRP Core. In this case, we simulated the system such that NEC's VR5500 is combined with DRP Core that consists of 2 Tiles. The capability of data transfer between VR5500 processor and DRP Core, and the configuration method of DRP Core were assumed in Section 3.

Target applications to evaluate the impact of co-processing are HMAC-MD5-96 and HMAC-SHA-1-96 for authentication algorithms on IPsec. We supposed that MD5 and SHA-1, the core of both applications, are executed on DRP Core. Moreover, the message data to process is generated from a random number by 256-bit block automatically. The preliminary profiling shows that the operation time of both hash functions accounts over 90% of the execution time. Besides, as the message size to operate becomes large, the ratio of both hash function increases.

Figure 8 shows the performance improvement by co-processing to the case that HMAC-MD5-96 and HMAC-SHA-1-96 are executed by software on VR5500 only. The vertical axis shows the ratio of performance improvement which is normalized by the execution time for the software only, while the horizontal axis shows the message size.

The data transfer between VR5500 and DRP Core is assumed to be performed through a shared memory connected via an on-chip bus like NECoBus, and DMA transfer is supported. The throughput of the on-chip bus is assumed to be 64bit at 100MHz. Under the above assumptions, we estimated the time required to communicate the message data. Furthermore, the pure operation time of evaluation programs was measured, assuming that the configuration of DRP Core had been finished completely.

Figure 8 shows that if DRP Core and VR5500 run together, it is possible to improve the perfor-

Table 2. Configuration Data Size of Tasks on DRP-1

Task	Config.Size[bit]
DES (Encryption + KeyScheduler)	331104
CAST128 (Encryption)	104928
AES (Encryption + KeyScheduler)	201184
CAST256 (Encryption)	207744
RC6 (Encryption)	90240
MD5	113760
SHA-1	134560

mance compared with a case of the software execution on VR5500 only. In particular, the co-processing is more effective if the message size to operate becomes large. Consequently, the achieved improvement is about 1.8 times in HMAC-MD5-96 and about 4.1 times in HMAC-SHA-1-96.

5.3 Analysis of Run-Time Configuration Overhead

5.3.1 Configuration Data Size The previous results show that multiple cryptographic tasks can be accelerated by co-processing with an embedded processor and DRP Core. In our system, however, cryptographic tasks are switched by a virtual hardware mechanism. Namely, they are switched by replacing the configuration data corresponding to each task on demand. Thus, the run-time configuration overhead in task switching must be evaluated.

The size of configuration data of each cryptographic task is shown in Table 2. About 40KB of configuration data in a large case must be transferred from the external memory into DRP Core. Although it is much smaller than the configuration data of common FPGAs, its configuration time is considerable compared with the execution time. So, the virtual hardware mechanism with double buffering is introduced to hide the run-time configuration overhead.

5.3.2 Simulation Results We simulated the virtual hardware system with the double buffer using the actual IPsec traffic, and evaluated the effect of overhead hiding. Each cryptographic task is implemented within 16 contexts of the DRP-1. Thus, we supposed that 32 contexts are available to allow using the double buffer. It is also assumed that the bandwidth of DRP configuration is 128bit/100MHz. The configuration access is controlled by an on-chip embedded processor.

The traffic data of IPsec is traced from the following actual network with three PCs (A, B, and C) communicating with each other with IPsec. The PC-B and -C download a certain size of a file from the PC-A simultaneously. The traffic data streaming in the above network is measured. The PC-A and -B communicate with encryption by AES-CBC and authentication by HMAC-MD5-96. Similarly, the PC-A and -C communicate with RC6-CBC and HMAC-SHA-1-96. It is assumed that our cryptographic accelerator is used in the PC-A which is a host PC in this network. In addition, we suppose that enough size of packets are stored in the IP-packet buffer of the PC-A to predict and prefetch the next configuration data during the execution of a certain task.

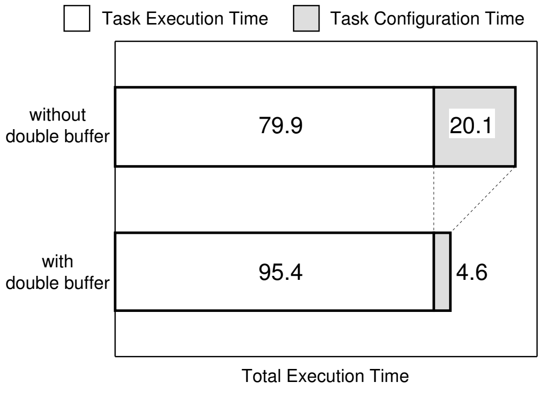


Figure 9. Run-Time Configuration Overhead and Impact of Double Buffering

Figure 9 shows the simulation result of the run-time configuration overhead. In this graph, the ratio of required configuration time of tasks in the case of with/without the double buffer is shown. If the configuration cannot be overlapped, the run-time configuration overhead is 20.1% of the total operation time. If the double buffer is available, the overhead decreases to 4.6% and it is possible to reduce 80.7% of the configuration time compared with the case that any configurations cannot be hidden.

The effects of double buffering are classified broadly into following two categories.

a) One is the effect of hiding configuration time behind a task execution.

b) The other effect is reduction of configuration frequency due to buffering of tasks in DRP Core. In this simulation, since 19.7% of tasks had been retained in the DRP Core, their configurations had been skipped.

From above analysis, we have demonstrated that configuration overhead can be reduced to 4.6% of total operation time by applying run-time configuration based on the double buffer. Consequently, it is also possible to implement a practical virtual hardware system with DRP Core.

Additionally, if more contexts (e.g. 64 contexts) are available, more tasks can be implemented and it is easy to reduce the overhead of run-time configurations. The hierarchical configuration cache can improve the bandwidth of the run-time configuration.

6 Related Work

Recently, FPGA-based cryptographic accelerators have been developed. Utilizing one or more FPGAs, high throughput and flexible systems have been implemented.

In [6], SLAAC-1V board that has three Xilinx's Virtex XCV1000 is proposed. High-throughput Triple-DES and AES are implemented in this accelerator. This board is a powerful and large scale cryptographic accelerator based on multiple FPGAs.

Task switching method by run-time FPGA configuration is introduced in [7]. In this literature, Adaptive Cryptographic Engine (ACE), which is an IPsec accelerator based on a single FPGA is proposed. In ACE, five private-key encryptions are implemented and ACE can switch them by run-time FPGA configuration. Besides, for the challenge to the configuration overhead, an efficient use of configuration memory by compressing configuration data is examined.

These FPGA-based systems achieve high throughput and flexibility, but require multiple high-end FPGAs which are connected to host PC by interface like PCI and USB. These systems are for acceleration of common PCs, and difficult to be integrated in embedded systems. And also, because of milli-second order configuration time of common FPGAs, the configuration overhead is extremely large.

In our cryptographic accelerator for IPsec, we adopted a coarse-grain dynamically reconfigurable processor, DRP whose configuration overhead is smaller than commercial FPGAs. Thus, in particularly embedded systems, our proposed system is advantageous compared with conventional approaches.

7 Summary

The implementation and the evaluation of the cryptographic accelerator for IPsec on NEC's Dynamically Reconfigurable Processor (DRP) are presented and discussed. This system accelerates multiple cryptographic tasks on DRP-1 and also can switch them on demand by a virtual hardware mechanism.

The evaluation result shows that all of cryptographic tasks can be implemented within 2 Tiles of DRP-1. The throughput of each cryptographic task outperformed the DSP and MIPS compatible processor. Furthermore, the simulation results show that our co-processing system eliminated a bottleneck of the software execution and achieved from 1.8 times to 4.1 times of performance improvement in the authentication application for IPsec.

In addition, we have analyzed the overhead of a run-time configuration on the virtual hardware. The evaluation result shows that the overhead is 20.1% of the total operation time. It is also clarified that about 80.7% of the run-time configuration time can be reduced using the double buffer method.

Acknowledgment

Throughout this work, the DRP-1 Device and its design and synthesis tools were provided from NEC/NEC Electronics. The authors would like to show our gratitude to all members of the DRP development group at NEC Electronics and NEC Laboratories for their design tool support and considerable amount of technical advice.

8. References

[1] Advanced Encryption Standard(AES). Federal Information Processing Standard (FIPS) Publication 197, 2001.

[2] The Keyed-Hash Message Authentication Code (HMAC). FIPS Pub 198, 2002.

[3] A. Alsolaim, J. Becker, M. Glesner, and J. Starzyk. Architecture and Application of a Dynamically Reconfigurable Hardware Array for Future Mobile Communication Systems. In *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM2000)*, pages 205–214, Apr. 2000.

[4] H. Amano, T. Inuo, H. Kami, T. Fujii, and M. Suzuki. Techniques for Virtual Hardware on a Dynamic Reconfigurable Processor -An approach to tough cases-. In *Proceedings of International Conference on Field Programmable Logic and Application (FPL2004)*, pages 464–473, Sept. 2004.

[5] K. Anjo, A. Okamura, and M. Motomura. Wrapper-based Bus Implementation Techniques for Performance Improvement and Cost Reduction. *IEEE Journal of Solid State Circuits*, 36(5):804–817, May 2004.

[6] P. Chodowiec, K. Gaj, P. Bellows, and B. Schott. Experimental Testing of the Gigabit IPSec-Compliant Implementations of Rijndael and Triple DES Using SLAAC-1V FPGA Accelerator Board. In *Proceedings of the 4th International Conference on Information Security (ISC2001)*, pages 220–234, Oct. 2001.

[7] A. Dandails, V. K. Prasanna, and J. D. P. Rolim. An Adaptive Cryptographic Engine for IPsec Architectures. In *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM2000)*, pages 132–141, Apr. 2000.

[8] Elixent. http://www.elixent.com/.

[9] IPFlex. http://www.ipflex.com/.

[10] X.-P. Ling and H. Amano. WASMII: A Data Driven Computer on a Virtual Hardware. In *Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines (FCCM1993)*, pages 33–42, Apr. 1993.

[11] M. Motomura. A Dynamically Reconfigurable Processor Architecture. *Microprocessor Forum*, Oct. 2002.

[12] PACT. http://www.pactcorp.com/.

[13] S.Kent and R. Atkinson. Security Architecture for the Internet Protocol. RFC 2401, 1998.

[14] G. J. M. Smit, P. J. M. Havinga, L. T. Smit, and P. M. Heysters. Dynamic Reconfiguration in Mobile Systems. In *Proceedings of International Conference on Field Programmable Logic and Application (FPL2002)*, pages 162–170, Aug. 2002.