RENA-CHIP Software Configuration

Kenji Kawai†

Abstract

NTT has developed software called RENA-ware to control the RENA-CHIP, a large-scale integration (LSI) chip used to offload packet processing from the CPU (central processing unit) of a home gateway.

1. Introduction

NTT's RENA-CHIP is a large-scale integration (LSI) chip that performs network-adapter processing in hardware [1]. It performs a series of network-adapter processes—from the receiving to the sending of packets—all in hardware. This approach prevents the drop in speed that results when these processes are performed by software and makes it possible to achieve a full wire rate of 1 Gbit/s in each direction. It also results in significantly less power consumption because packet processing by dedicated hardware is more efficient than that by software. All in all, this chip achieves a high level of performance that is difficult to achieve with software while also achieving low power consumption, which is suitable for network adapters. The chip itself, however, does not perform all of the packet processing performed by a network adapter because exception-packet processing is still performed by software. This approach prevents the circuit scale of the chip from becoming too large and keeps manufacturing costs down.

This approach of converting a series of processes from software to hardware implies that software must be extensively revised. In contrast, the approach used for conventional network processors of offloading individual functions, while making it difficult to achieve high speeds, does have the advantage of requiring no major changes to existing software. In our chip, tables like the routing table and classifier table that have traditionally been managed by basic software are also implemented in hardware, which means that their contents must be updated by basic software. This suggests that basic software must be revised—despite the difficulty involved—in order to control the chip. After giving this problem much thought, NTT developers of network adapters incorporating the RENA-CHIP developed RENA-ware to control the chip without having to make extensive changes to basic software.

2. RENA-ware overview

The RENA-CHIP achieves the functionality of a network adapter not by performing all network-adapter functions by itself but by linking with software processing where needed. Implementing exception-packet processing, for example, in hardware would have various detrimental effects. It would not only hamper high-speed processing in the network adapter, but also increase the circuit scale and be less flexible to changes in specifications. To keep the circuit scale in check and maintain flexibility with respect to changes in specifications, it was decided that exception-packet processing would be performed by software. The types of exception packets to be handled by software are listed in **Table 1**.

Furthermore, while a network adapter has a function for dynamically updating tables, the RENA-CHIP has no function for automatically rewriting tables. But the number of users and connected terminals is not large in the case of a network adapter, so the updating frequency of NAPT (network address port translation), routing, and classifier tables is low. Accordingly, implementing such a function in hardware would not contribute to higher speeds in the net-

† NTT Cyber Solutions Laboratories
Yokosuka-shi, 239-0847 Japan
E-mail: kawai.kenji@lab.ntt.co.jp

Table 1. RENA-ware exception-packet processing.

Packet conditions	Processing
Header check sum, packet length, or other parameter is abnormal.	Return ICMP packet indicating “packet abnormality” to originator.
TTL is less than 1.	Return ICMP packet indicating “time exceeded” to originator.
Packet length exceeds MTU value.	Divide packet if fragmentation is allowed. Otherwise, return ICMP packet indicating “destination unreachable” to originator.
Origin and destination addresses are in the same network.	Return ICMP packet indicating “redirect” to originator.
Partial packet generated by IP fragmentation is an NAPT object.	Since partial packets generated by IP fragmentation cannot be NAPT processed by the RENA-CHIP, the pre-fragmentation packet is reconstructed by RENA-ware, enabling NAPT processing.

ICMP: Internet control message protocol
TTL: time to live
MTU: maximum transfer unit
IP: Internet protocol

work adapter. A typical example of rewriting tables is ARP (address resolution protocol) operations. ARP renews the MAC (media access control) address database at intervals of several minutes. When the number of connected terminals is less than 50, the routing table in the chip is rewritten at intervals of several seconds. It was therefore decided to perform dynamic table management in the RENA-CHIP by software, the same as for exception-packet processing. On the other hand, it was decided to implement packet-fragment tracking in hardware because it was thought that if this function were performed by software, it would not be fast enough.

A software-based network adapter normally manages tables like the routing table and classifier table by basic software. In the RENA-CHIP, these tables are built-in, but they can still be managed dynamically, the same as in software processing, if they match up with tables on the software side. This function for ensuring that our chip’s built-in tables are consistent with software-side tables can be added to existing basic software in the form of RENA-ware.

RENA-ware must keep tables up to date in the same way as conventional software processing. But in contrast to the conventional method of processing all packets by software, most packets in a RENA-CHIP-equipped network adapter are transferred within the chip without any software intervention. To keep tables up to date, software processing must therefore be initiated in some way. This can be accomplished by sending a packet that is qualified as a table updating occasion (such as a connection-initiate packet when dynamic filtering is being performed) to the CPU (central processing unit) without any transfer processing within the RENA-CHIP. Table usage conditions may also be checked. However, during a table update that deletes unused entries, it will not be sufficient to check only entry usage conditions on the software side: in addition to those conditions, it will also be necessary to check the conditions of entries within the RENA-CHIP.

An example of RENA-CHIP control by RENA-ware is shown in **Fig. 1**. The various types of table management functions are listed in **Table 2**. The only revision made to basic software here was to add a call to a chip’s control application programming interface (API); actual table changes are performed by new middleware. This configuration decreases the man-hours required to revise basic software. This, in turn, reduces the probability of inadvertently introducing defective code as a result of making revisions while also reducing the man-hours required to deploy new versions of basic software or add new functions. It also has the effect of simplifying the maintenance and

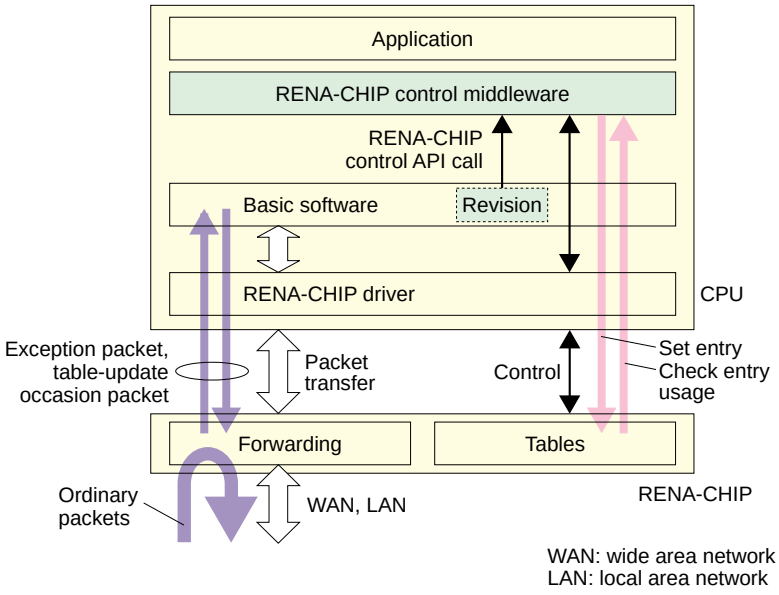

Fig. 1. RENA-ware configuration.

Table 2. RENA-ware table management functions.

Function	Operation	Occasion	Information needed for management
PPPoE	Hook session information and register entry in logical interface table.	Initiate session (NCP connection)	Non-IP packet received in PPPoE discovery stage or session stage
	Delete corresponding entry from logical interface table.	Terminate session (terminate LCP)	Same as above
IPsec SA	Hook IPsec SA parameters and register entry in IPsec SA table.	Generate IPsec SA	IKE packet (self-addressed)
	Update corresponding entry in IPsec SA table.	Update entry	Same as above
	Issue IPsec SA update request.	Interrupt RENA-CHIP	ESP sequence no., total number of packets for each entry of IPsec SA table
	Delete corresponding entry from IPsec SA table.	Delete IPsec SA	IKE packet (self-addressed)
ARP/ NDP	Hook ARP/NDP information and register entry in routing table.	Register entry	Address-unresolved packet, ARP reply, NDP packet
	Delete corresponding entry from routing table.	Delete entry	—
	Determine that corresponding entry in routing table has timed out due to nonuse.	Timeout	Number of times each entry in routing table is used
Routing	Hook routing-table entry information and register entry in routing table.	Register entry	IP packet not matching routing table
	Delete corresponding entry from routing table.	Delete entry	—
	Determine that corresponding entry in routing table has timed out due to nonuse.	Timeout	Number of times each entry in routing table is used
Dynamic NAPT	Hook NAPT information and register entry in NAPT table.	Register entry	IP packet not matching NAPT table
	Delete corresponding entry from NAPT table.	Delete entry	—
	Determine that corresponding entry in NAPT table has timed out due to nonuse.	Timeout	Number of times each entry in NAPT table is used
Dynamic filtering	Hook connection information and register entry in classifier table.	Establish connection	TCP/UDP/ICMP packet for establishing connection
	Delete corresponding entry from classifier table.	Terminate connection	TCP/UDP/ICMP packet for terminating connection
	Determine that corresponding entry in classifier table has timed out due to nonuse.	Timeout	Number of times each entry in classifier table is used

NCP: network control protocol
LCP: link control protocol
IPsec: Internet protocol security
SA: security association
IKE: Internet key exchange
ESP: encapsulating security payload
NDP: neighbour discovery protocol

support of RENA-ware itself.

A photograph of the evaluation board used as a development environment for RENA-ware is shown in **Fig. 2**, and a block diagram of this board is shown in **Fig. 3**. The board uses a 200-MHz embedded CPU and incorporates two 512-Mbit SDR-SDRAMs (single data rate synchronous dynamic random access memories) as CPU main memory, two 128-Mbit DDR-SDRAMs (double data rate SDRAMs) as a RENA-CHIP packet buffer, four 64-Mbit flash ROMs (read-only memories) for storing RENA-firmware, and a 4-port layer 2 switch (L2-SW) for the local area network (LAN) side. On the board, a memory bus connects the RENA-CHIP with the CPU main memory to achieve high-speed transfer of packet data between the RENA-CHIP and CPU where the RENA-CHIP is used as the bus master. The VxWorks* realtime operating system is used as basic software for the evaluation board.

3. RENA-ware process flow

The processing involved in adding an entry to the ARP table is described below as a typical example of an operation sequence performed by RENA-ware.

* VxWorks is a registered trademark of Wind River Systems, Inc.

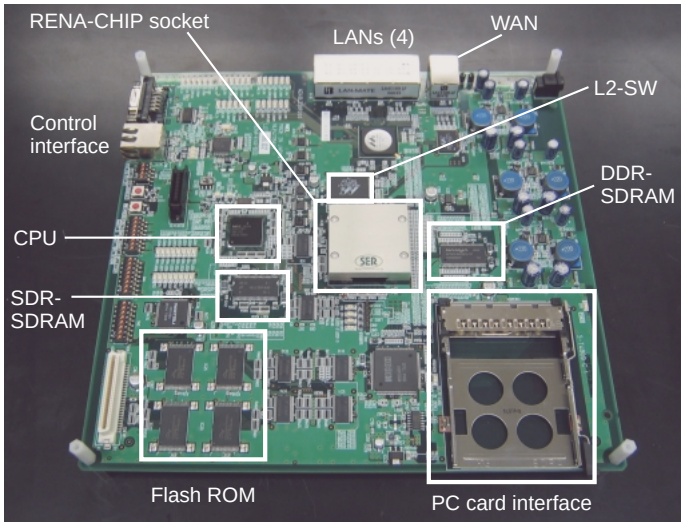

Fig. 2. RENA-CHIP evaluation board.

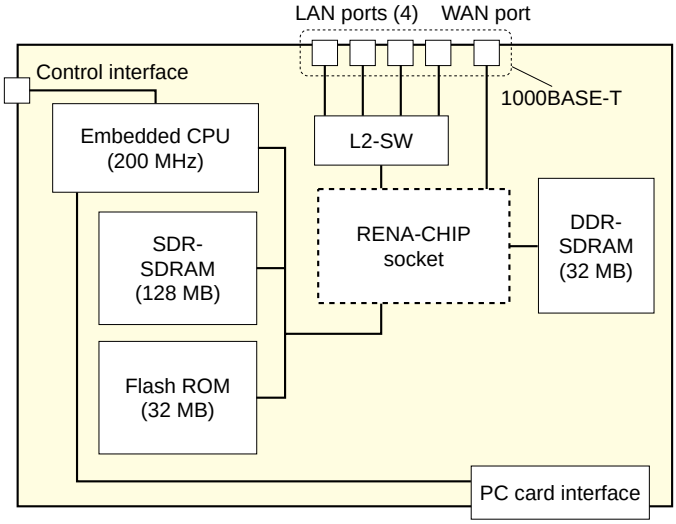

Fig. 3. Configuration of RENA-CHIP evaluation board.

Figure 4 is a diagram showing the sequence of operations that occur when the network adapter receives at the WAN (wide area network) port an IP packet destined for a terminal whose MAC address is unknown. First, the RENA-CHIP inputs the IP packet at its WAN port and proceeds to search the routing table (1). The search returns an entry for which the address-unresolved flag is ON, so the chip transfers the IP packet to the CPU (2). At this time, RENA-ware learns that the address is unresolved after searching the software-managed routing table and ARP table. It therefore decides to hold off on transferring the IP packet and broadcasts through one of the LAN ports an ARP request packet that asks for the MAC address corresponding to the packet's IP address (3). Next, upon receiving an ARP reply from the terminal in question, the chip adds a resolved IP-address/MAC-address entry to its routing table and RENA-ware does the same to its ARP table (4). The IP packet whose transfer had been postponed can now be output from a LAN port using the MAC address so obtained as the destination address (5). From here on, whenever an IP packet with the same IP address is received, a search of the chip's routing table will return the entry with the resolved MAC address so that the packet can be transferred from the WAN port to a LAN port within the chip based on the instructions given by that entry.

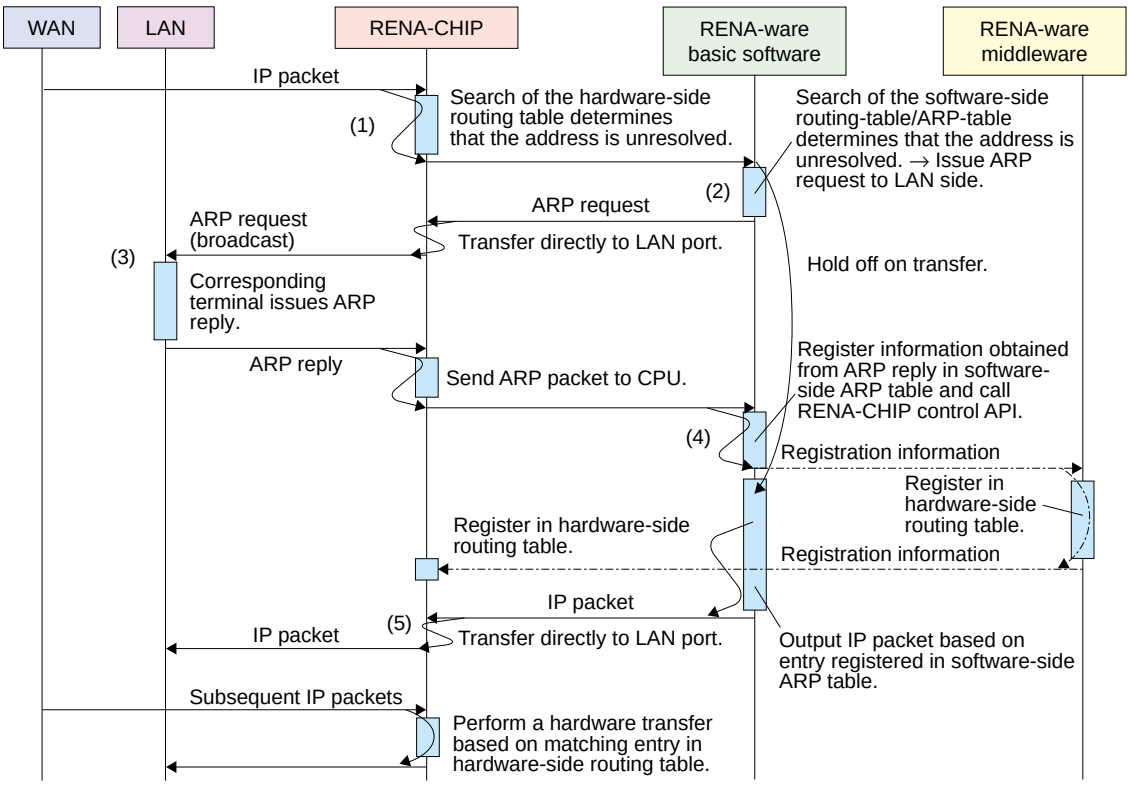

Fig. 4. Example of RENA-ware operation (ARP table registration).

In Fig. 4, the only RENA-ware operation that differs from those of existing basic software is operation (4). Here, as a function that is activated whenever an entry is added to the ARP table managed by existing basic software, we added one for hooking the information in that entry and calling the control API to update the chip's routing table. And in conjunction with this function, we added control middleware for actually adding the entry in the chip's routing table based on that API call.

In short, RENA-ware has been made by revising the existing basic software and adding middleware. The revision involves the addition of a function for hooking entry information and calling the control API whenever an entry addition, deletion, or timeout occurs within basic software. The additional middleware performs various processes based on that API call such as adding and deleting table entries and evaluating their usage status.

4. Future developments

Topics for future research include the development of RENA-ware using Linux as the basic software in addition to VxWorks. It will also be necessary to create a business model for Linux-version RENA-ware and construct a support system; these are expected to be difficult tasks, but they are essential to bring the RENA-CHIP to the general market.

Reference

[1] K. Koike, "Technical Trends of Network Processors and the RENA-CHIP," NTT Technical Review, Vol. 4, No. 9, pp. 12-16, 2006 (this issue).

Kenji Kawai
Senior Research Engineer, First Promotion Project, NTT Cyber Solutions Laboratories.
He received the B.E. and M.E. degrees in electronic communication engineering from Waseda University, Tokyo, in 1989 and 1991, respectively. He joined NTT LSI Laboratories, Kanagawa, in 1991. He has been engaged in research on the design of high-speed LSIs.