To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

SH7763 Group

Example of Setting for Reception of Ethernet Frames

Introduction

This application note provides an example of setting for data reception through a port of the Ethernet function RMII (Reduced Media Independent Interface), which is incorporated in the SH7763 Group.

Target Device

SH7763

Contents

1. Preface

1.1 Specifications

- In this sample program, ten Ethernet frames are received. Every time an Ethernet frame is received, the frame received interrupt is used to initiate copying of the frame to a user buffer.

1.2 Modules Used

- Gigabit Ethernet controller (GETHER)
- Interrupt controller (INTC)
- General purpose I/O (GPIO)

1.3 Applicable Conditions

- Evaluation board: SH7763 Solution Engine (type no.: MS7763SE02) from Hitachi ULSI Systems Co., Ltd.
  Ethernet PHY: KSZ8721BL from Micrel
- MCU: SH7763 (R5S77631AY266BGV)
- Operating frequency:

CPU clock:	266.66 MHz
Local bus clock:	66.66 MHz
DDR-SDRAM I/F clock:	133.33 MHz
Peripheral bus clock 0:	66.67 MHz

- Toolchain: SuperH RISC engine Standard Toolchain Ver.9.3.0.0 from Renesas Technology
- Compiler options: Default settings of High-performance Embedded Workshop
  (-cpu=sh4a -include="$(PROJDIR)¥inc"
  -object="$(CONFIGDIR)¥$(FILELEAF).obj" -debug -gbr=auto –chgincpath
  -errorpath -global_volatile=0 -opt_range=all -infinite_loop=0
  -del_vacant_loop=0 -struct_alloc=1 -nologo)

1.4 Related Application Note

The operation of the reference program for this document was confirmed with the setting conditions described in the *SH7763 Group Application Note: SH7763 Example of Initialization* (REJ06B0934). Please refer to that document in combination with this one.

SH7763 Group Application Note: Example of Setting for Transmission of Ethernet Frames (REJ06B0936)

SH7670 Group Application Note: Example of Setting for Automatic Negotiation by Ethernet PHY-LSI (REJ06B0800)

2. Description of Sample Application

This sample program performs Ethernet reception through one (E-MAC-0) of the two MAC layer interface lines of the Gigabit Ethernet Controller (GETHER). The RMII is selected as the interface according to the specification of the evaluation board in order to perform 100BASE-T Ethernet communication.

During reception, the dedicated Direct Memory Access Controller (E-DMAC) incorporated in the GETHER is used to transfer Ethernet frames to a receive buffer in the memory.

2.1 Operational Overview of Module Used

The GETHER consists of the following three function units:

(1) DMA transfer controller (E-DMAC): DMA transfer between the transmit/receive buffer in the memory and the transmit/receive FIFO

Using its direct memory access (DMA) function, the E-DMAC performs DMA transfer of frame data between a user-specified Ethernet frame transmission/reception data storage destination (accessible memory space: transmit buffer/receive buffer) and the transmit/receive FIFO in the EDMAC.

To enable the E-DMAC to perform DMA transfer, information (data) including a transmit/receive data storage address and so forth, referred to as a descriptor, is required. The E-DMAC reads transmit data from the transmit buffer or writes receive data to the receive buffer according to the descriptor information. By arranging multiple descriptors as a descriptor row (list) (to be placed in a readable/writable memory space), multiple Ethernet frames can be transmitted or received continuously.

The E-DMAC consists of two lines: one for port 0 and the other for port 1, and both operate independently for transmission and reception.

(2) MAC controller (E-MAC): Transmission/reception processing between the transmit/receive FIFO and the GMII/MII/RMII

The E-MAC constructs an Ethernet frame using the data written to the transmit FIFO and transmits the frame to the GMII/MII/RMII. It also performs a CRC check of an Ethernet frame received from the GMII/MII/RMII and deconstructs the frame to write to the receive FIFO. The EMAC supports three formats MII, GMII and RMII for interface to the PHI-LSI connected externally to this LSI.

The E-MAC consists of two controllers: E-MAC0 for port 0 and E-MAC1 for port 1, which correspond to E-DMAC0 and E-DMAC1 respectively.

(3) Transfer Switching Unit (TSU): Transfer processing between port 0 and port 1, and CAM processing

The TSU performs Ethernet frame data transfer between the E-MAC0 and E-MAC1. The TSU, which is placed between the E-DMAC and E-MAC, references the CAM entry table to select one of the following tasks according to the Ethernet frame destination address (DA) input to the E-MAC.

- Receives frame and writes to the receive FIFO.
- Transfers frame and writes to the transfer FIFO.
- Receives frame and writes to the receive FIFO and transfer FIFO.
- Discards frame.

The TSU performs transfers from port 0 to port 1 and from port 1 to port 0 independently.

Table 1 lists the outline of the GETHER.

For details on the GETHER, please refer to the section on Gigabit Ethernet controller (GETHER) in the *SH7763 Group Hardware Manual* (REJ09B0256).

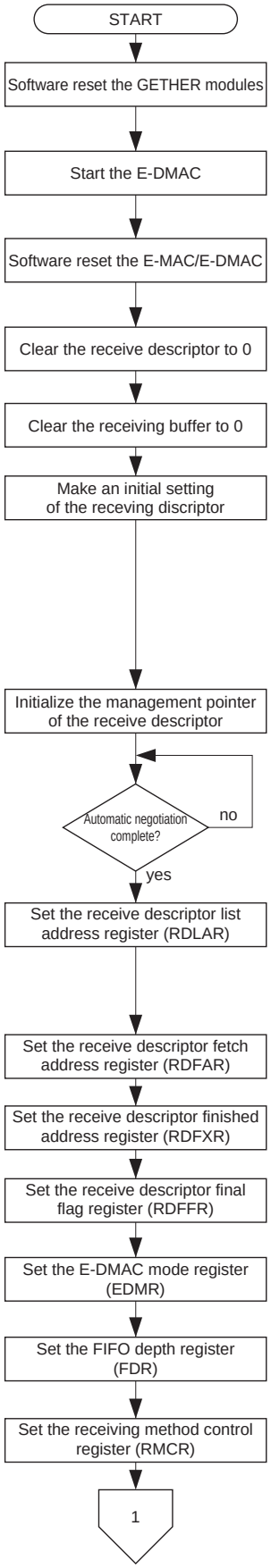
Table 1 Outline of the GETHER

Item	Description
E-MAC function	• Constructs/deconstructs data frames (frame format conforming to IEEE802.3, 2000 Edition) • Supports transfer at 10, 100, and 1000 Mbps • Supports full-duplex and half-duplex modes • Two channels (GETHER0 and GETHER1) • Flow control conforming to IEEE802.3x • Supports three PHY interfaces conforming to IEEE802.3 — GMII (Gigabit Media Independent Interface) — MII (Media Independent Interface) — RMII (Reduced Media Independent Interface) • Upward protocol support (checksum) function
E-DMAC function	• Data transfer between GETHER and external/internal memory • Four channels • 32-byte burst transfer • Supports single-frame/single-descriptor operation and single-frame/multi-descriptor (multibuffer) operation • Transfer data width: 32 bits • Transmit/receive FIFO (for transmission: 2 Kbytes, for reception: 8 Kbytes)
TSU function	• Switching unit for data transfer between channels (relay FIFO: 6 Kbytes)

2.2 Procedure for Setting Module Used

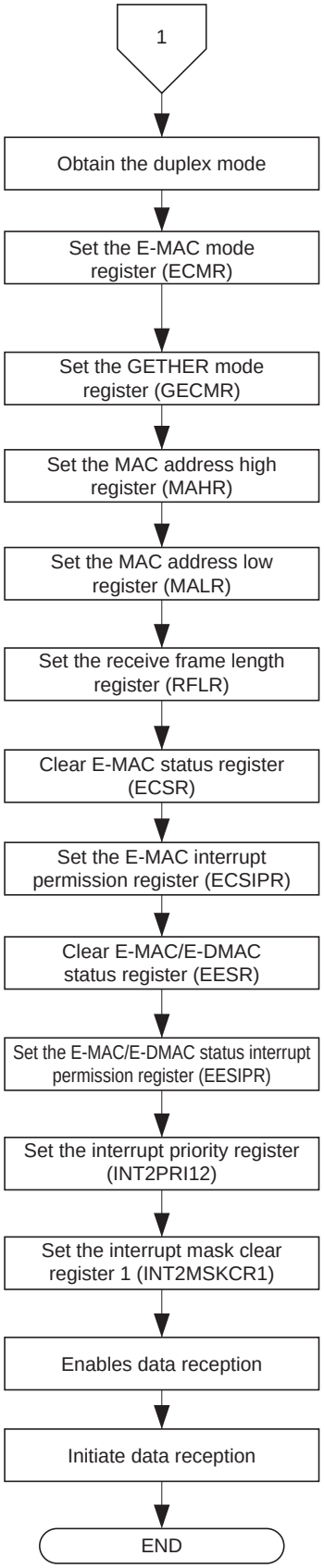
This section describes an example of fundamental settings for reception of the Ethernet frames.

Figures 1 and 2 shows an example of flowchart for setting the reception of Ethernet frames.



- The GETHER modules are reset by software.
  Write 1 to the ARST bit in the software reset register (ARSTR).
  Prohibit access to the registers of all blocks in the GETHER from during the issuance of a software reset (256 cycles of the external bus clock Bck).
- Start the E-DMAC transmitter and receiver.
- The E-MAC and E-DMAC modules are reset by software.
  Write 1 to the SWRT and SWRR bits in the E-DMAC mode register (EDMR).
  Prohibit access to the registers of all modules related to Ethernet communication on a channel that has been reset, until the SWRT and SWRR bits are returned to 0.
- Areas of receive descriptors on memory is cleared.
- Areas of receiving buffers on memory is cleared.
- The entire receive-descriptor list is initialized.
  RD0: Bit RACT is set to 1 (valid).
  Bit RDLE is set to 1 in the last descriptor (and 0 in the others).
  Setting is not required for bit RFP because this item is written back by the E-DMAC.
  RD1: Bit RBL is set to specify the receive buffer maximum transfer byte length on a 32-byte boundary basis.
  Setting is not required for bit RDL because this item is written back by the E-DMAC.
  RD2: The start address of the receive buffer for each descriptor is specified on a 32-byte boundary.
  Padding area: This area is not used by the E-DMAC, but is freely available to the user.
- Initializes pointer variable that manages the current descriptor.
  The start address of the receive descriptor list is set as the inital value.
- When automatic negotiation is enabled, wait until it is completed.
- Set the start address of the receive descriptor list.
  Lower-order bits are set as follows according to the specified descriptor length.
  16-byte boundary: RDLA[3:0] = 0000
  32-byte boundary: RDLA[4:0] = 00000
  64-byte boundary: RDLA[5:0] = 000000
  Actual memory areas are also allocated on corresponding boundaries.
- Set the receive descriptor fetch address register.
  In the initial setting, set the start address of the receive descriptor.
- Set the receive descriptor finished address register.
  In the initial setting, set the address of the receive descriptor immediately before the descriptor that is pointed to by the address in RDFAR.
- Set the receive descriptor final flag register.
  Set this register to 0x00000001 if the RDFXR indicates the end of the receive descriptor list.
- Slects whether or not the endian format is converted on data transfer by the E-DMAC.
- Decriptor length is set.
- Sets the capacity of the transmit FIFO and receive FIFO.
  Set this register to 0x0000071F when selecting 2 KB and 8 KB as the maximum sizes of transmit and receive buffers, respectively.
- This setting is made to indicate whether frame reception is continued or not.
  When continuous reception after receiving one frame is desired, the setting is 1.
  When continuous reception after receiving one frame is not desired, the setting is 0.

Figure 1 Example of a Flowchart for Ethernet Settings (1)



- Obtain information on the method of transmission, such as full duplex/half duplex and 10 Mbps/100 Mbps, from the result of automatic negotiation by the PHY-LSI.
- Set whether CRC error frame is received as an error or not.
- Magic Packet detection is enebled if required.
- Internal/external loopback is specified if requied.
- Specify full duplex or half duplex mode
  The result of automatic negotiation by the PHY-LSI is reflected.
- Select the transmission mode from 10 Mbps and 100 Mbps
  The result of automatic negotiation by the PHY-LSI is reflected.
- The highest-order 32 bits of the 48-bit MAC address are set.
  ex.) When the MAC address 01-23-45-67-89-AB (in hexadecimal) → 0x01234567
- The lowest-order 16 bits of the 48-bit MAC address are set.
  ex.) When the MAC address is 01-23-45-67-89-AB (in hexadecimal) → 0x000089AB
- The maximum frame length is set.
  This setting must be made if frames with lengths greater 1518 bytes, including the CRC, are to be received.
- The register is cleared to 0 by writing 1 to all of its bits.
- Settings are made to the following bits: the link signal change interrupt enable, magic packet detection interrupt enable, and illegal carrier detection interrupt enable. Set ECIIP in EESIPR to 1 when this interrupt is used.
- The register is cleared to 0 by writing 1 to all of its bits.
- Interrupts to all the bits in the E-MAC/E-DMAC status register (EESR) are enabled.
- Priority level of the GETHER-related interrupt is set.
  Bits 15 to 12 in the IPR12 are set. The setting 0x0 indicates the priority level 0 (requested mask), and 0xF indicates the priority level 15 (the maximum level).
- The GETHER-related interrupt mask is cleared.
- Reception of data is enabled by setting the RE bit in the ECMR to 1.
- Reception of data is enabled by setting the RR bit in the EDRRR to 1.

Figure 2 Example of a Flowchart for Ethernet Settings (2)

2.3 Operation of the Sample Program

This sample program employs the E-MAC-0 and the E-DMAC0 modules to receive 10 Ethernet frames from the host personal computer at the other end. In this sample program, there are four receive descriptors, and four areas of the receiving buffer each with 1,520 bytes. The receive enable control (RNC) bit in the receiving method control register (RMCR) is set to 1 to enable continuous reception operations. Every time an interrupt related to reception such as frame reception (FR), etc. is generated, the RFE bit (bit 27 in the RD0) of the receive descriptor is checked, and if no errors are found (i.e. RFE = 0) the single frame of data in the receiving buffer is copied to the user buffer. The corresponding descriptor is then initialized in readiness for its next round of reception. If an error is found (i.e. RFE = 1), data in the receiving buffer are not copied to the user buffer but the corresponding descriptor is initialized.

Additionally, data other than the preamble, SFD, and CRC in the Ethernet frame are transferred to the receiving buffer.

Figure 3 shows operating environment of the sample program, and figure 4 shows a format of the Ethernet frame.

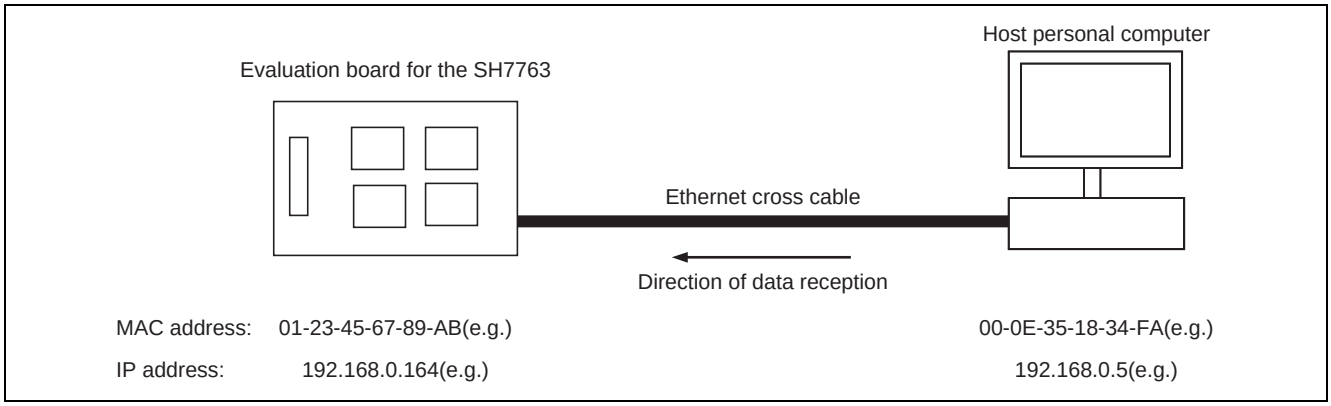


Figure 3 Operating Environment of the Sample Program

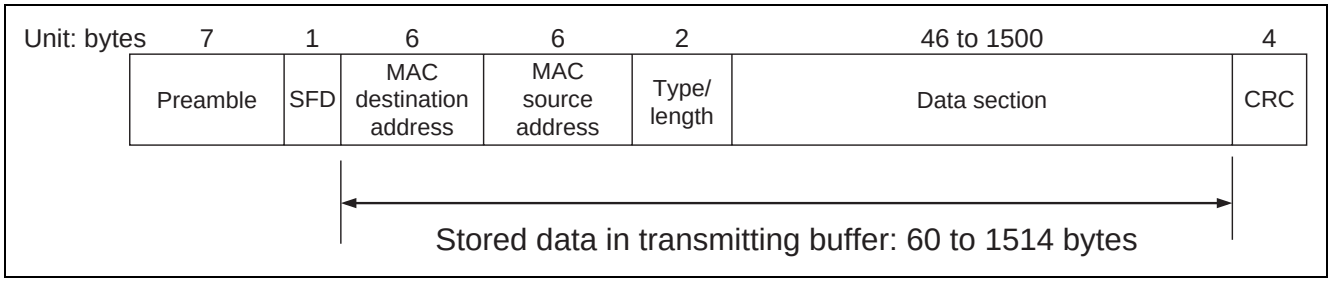


Figure 4 Ethernet Frame Format

2.4 Definition of Descriptors Used in the Sample Program

The E-DMAC does not use the padding area of a descriptor; this area is freely available to the user. In this sample program, this area is used to specify the address where the next descriptor starts, and this in conjunction with software is used to arrange the descriptors in a ring structure.

Figure 5 shows the definition of the transmit-descriptor structure in the sample program and an example of how the array of transmit descriptors is used.

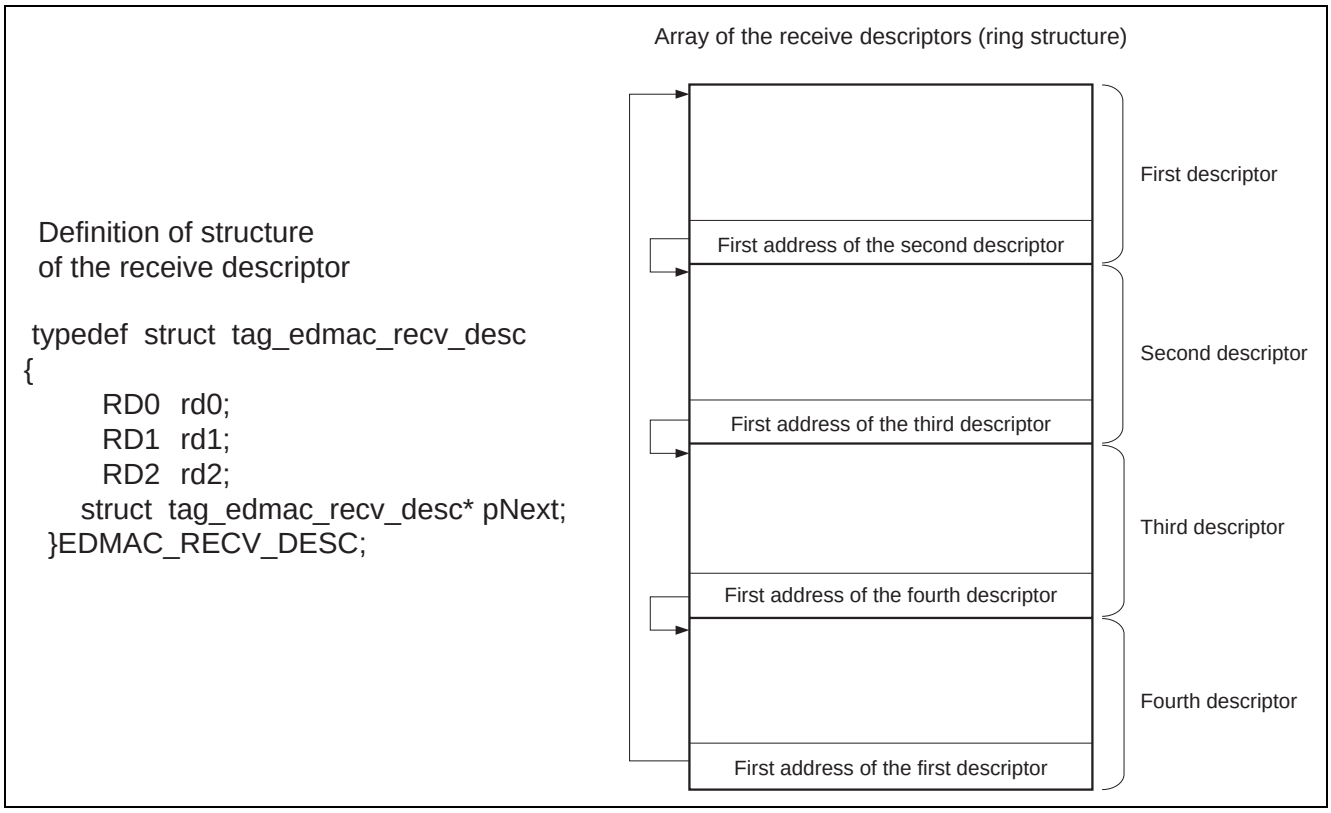


Figure 5 Definition of Receive Descriptor and Usage Example of Receive Descriptor Array

2.5 Sequence of Processing by the Sample Program

Figures 6 to10 show the flow of processing in the sample program. Although descriptors and the various registers of the E-MAC and E-DMAC modules are initially set up for transmission, processing for transmission is not performed.

The programs other than main.c and ether.c are the same as those used in the example of setting for transmission of Ethernet frames.

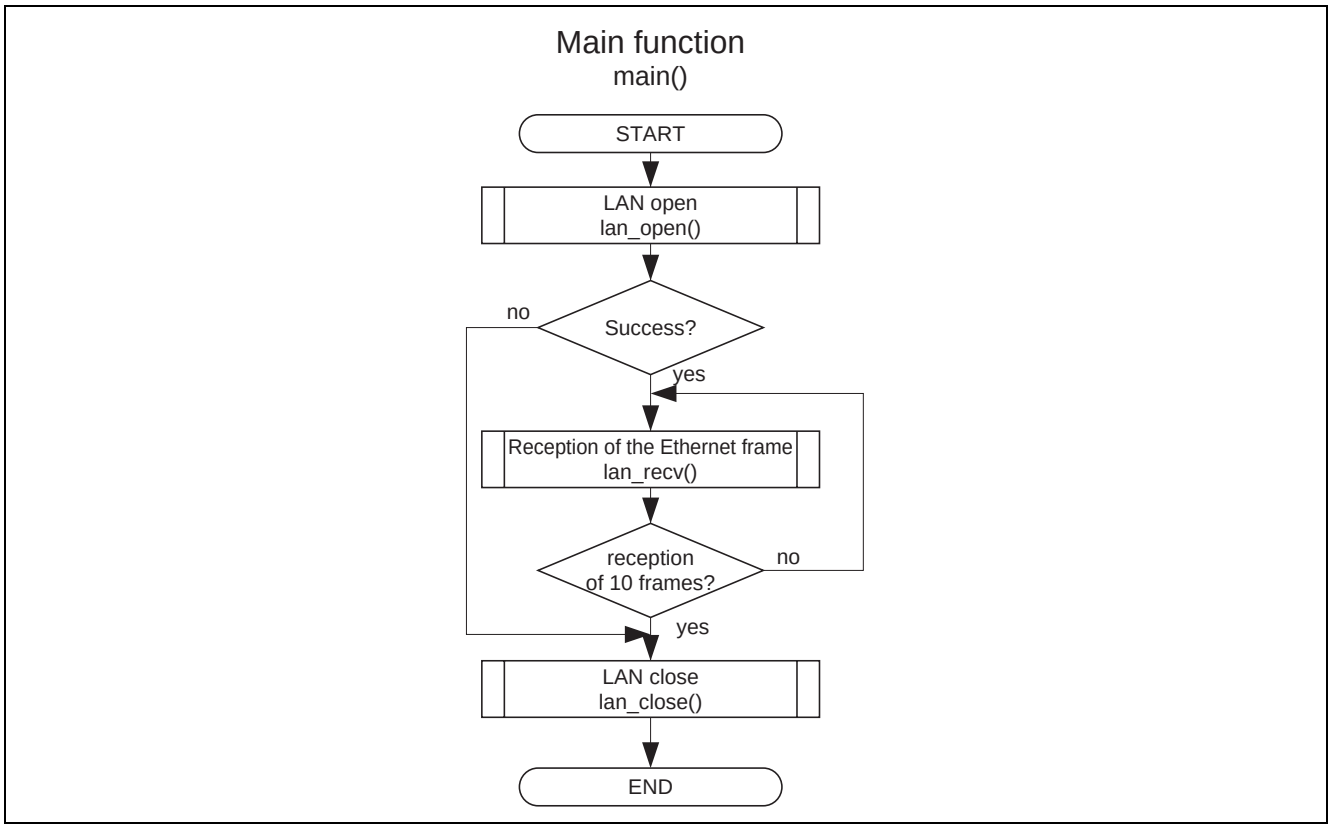


Figure 6 Flow of Handling in the Sample Program (1)

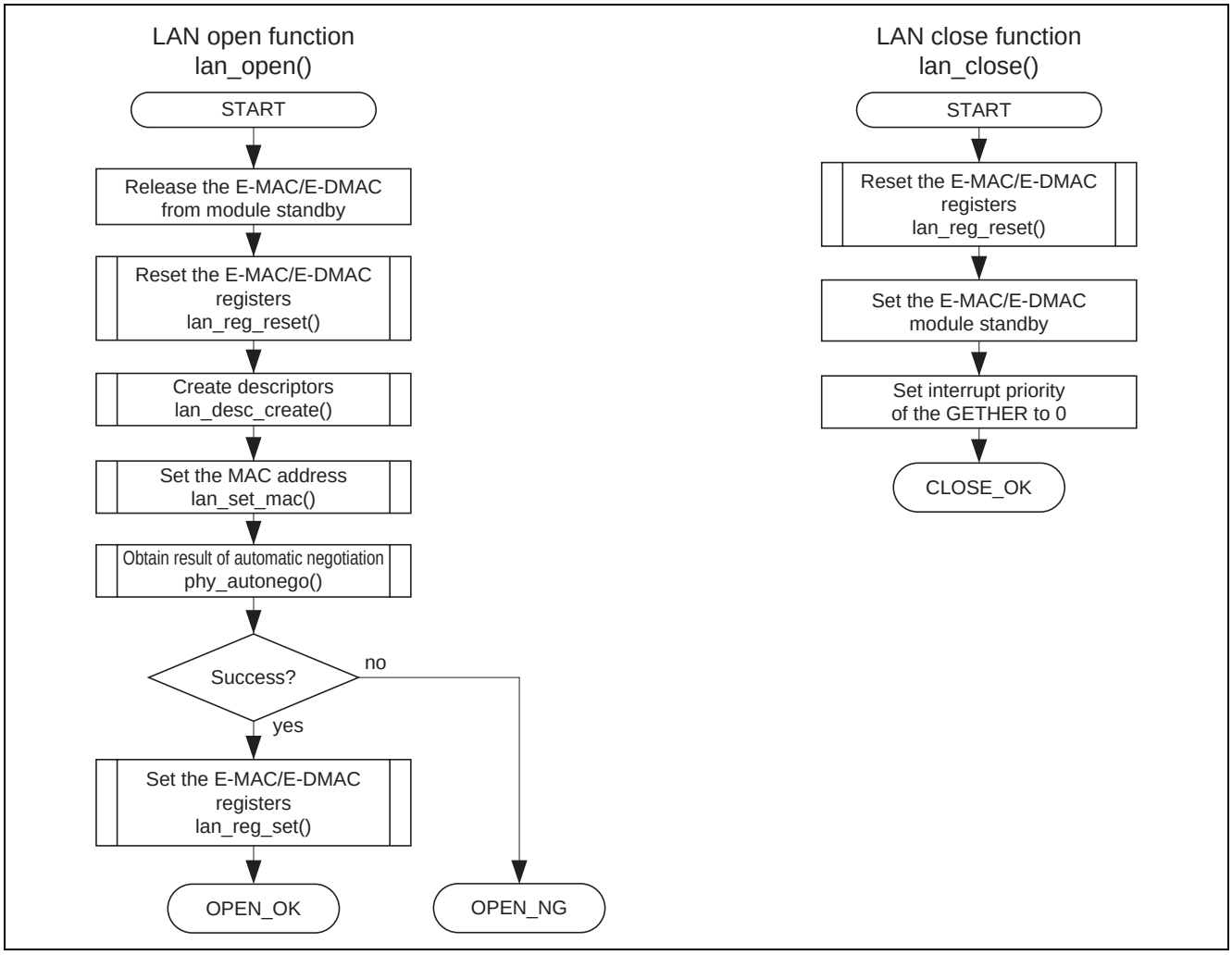


Figure 7 Flow of Handling in the Sample Program (2)

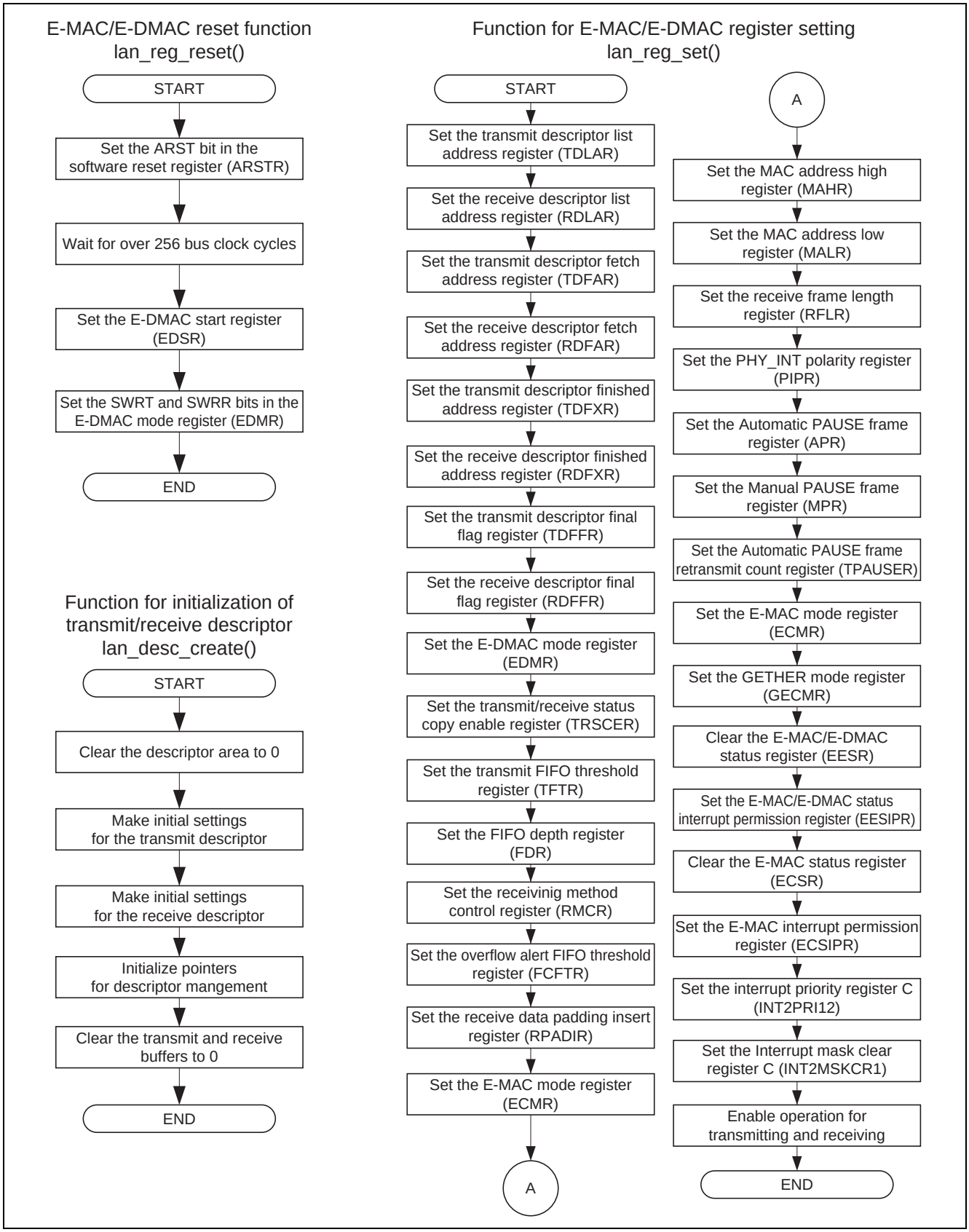


Figure 8 Flow of Handling in the Sample Program (3)

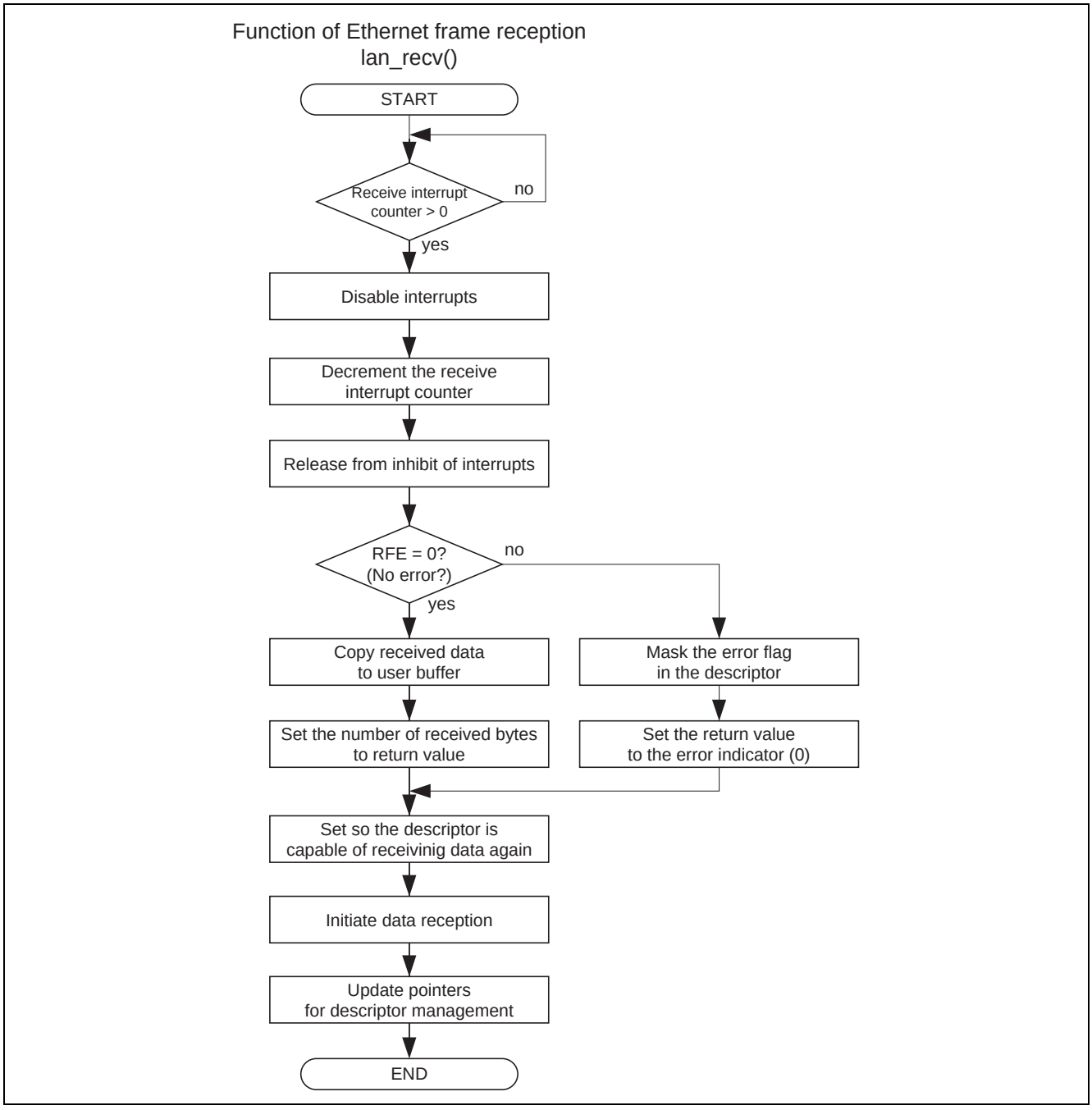


Figure 9 Flow of Handling in the Sample Program (4)

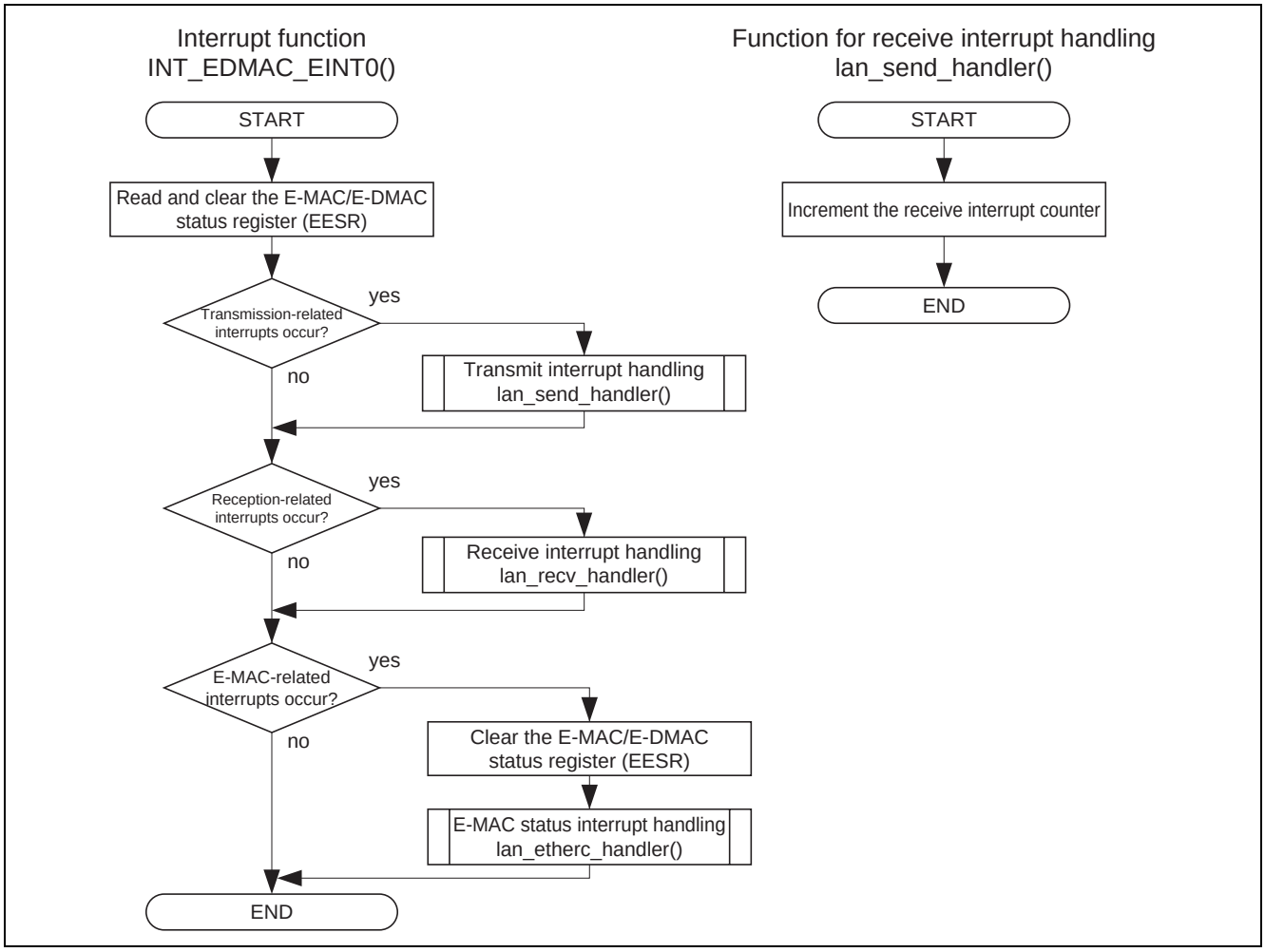


Figure 10 Flow of Handling in the Sample Program (5)

2.6 Allocation of Sections

In this sample program, the areas of the buffers and the transmit/receive descriptors are determined by allocating sections. Table 2 lists the allocation of sections.

Table 2 Allocation of Sections

Section Name	Application of Section	Area	Allocation Address (Virtual Address)	
P	Program area (in the case of none specified)	ROM	0x00002000	Area P0 (caching is enabled, MMU addresses can be translated)
C	Constant area	ROM		
C$BSEC	Address structure for non-initialized data area	ROM		
C$DSEC	Address structure for initialized data area	ROM		
D	Initialized data (initial value)	ROM		
RINTTBL	Initialized data area	RAM	0x08000000	
B	Non-initialized data area	RAM		
R	Initialized data area	RAM		
RP	Program transfer area	RAM		
RC	Constant transfer area	RAM		
S	Stack area	RAM	0x0FFFF9F0	
INTHandler	Exception/interrupt handler	ROM	0x80000800	Area P1 (caching is enabled, MMU addresses cannot be translated)
VECTTBL	Reset vector table Interrupt vector table	ROM		
INTTBL	Interrupt mask table	ROM		
PIntPRG	Interrupt function	ROM		
SP_S	Stack area for handler of TLB misses	RAM	0x8FFFFDF0	
RSTHandler	Reset handler	ROM	0xA0000000	Area P2 (caching is disabled, MMU addresses cannot be translated)
PResetPRG	Reset program	ROM		
DINTTBL	Initialized data area	ROM		
PnonCACHE	Program area (non-cacheable access)	ROM		
BETH_DESC	Descriptor area	RAM	0xAF000000	
BETH_BUFF	Buffer area	RAM	0xAF001000	

2.7 Setting for Automatic Negotiation by PHY-LSI

Figures 11 to 15 show the processing flows of the sample programs for obtaining the result of automatic negotiation with the PHY-LSI.

The MII register in the PHY-LSI is accessed via the PHY interface register (PIR) to obtain the result of automatic negotiation in the physical layer.

This sample application uses the PHY-LSI of the RMII interface, but the basic idea is the same as that of the MII interface.

For details, please refer to the *SH7670 Group Application Note: Example of Setting for Automatic Negotiation by Ethernet PHY-LSI* (REJ06B0800) and PHY-LSI data sheet.

The sample code of ether.h, intprg.c, phy.c, and phy.h is the same as that used in the example of setting for transmission of Ethernet frames.

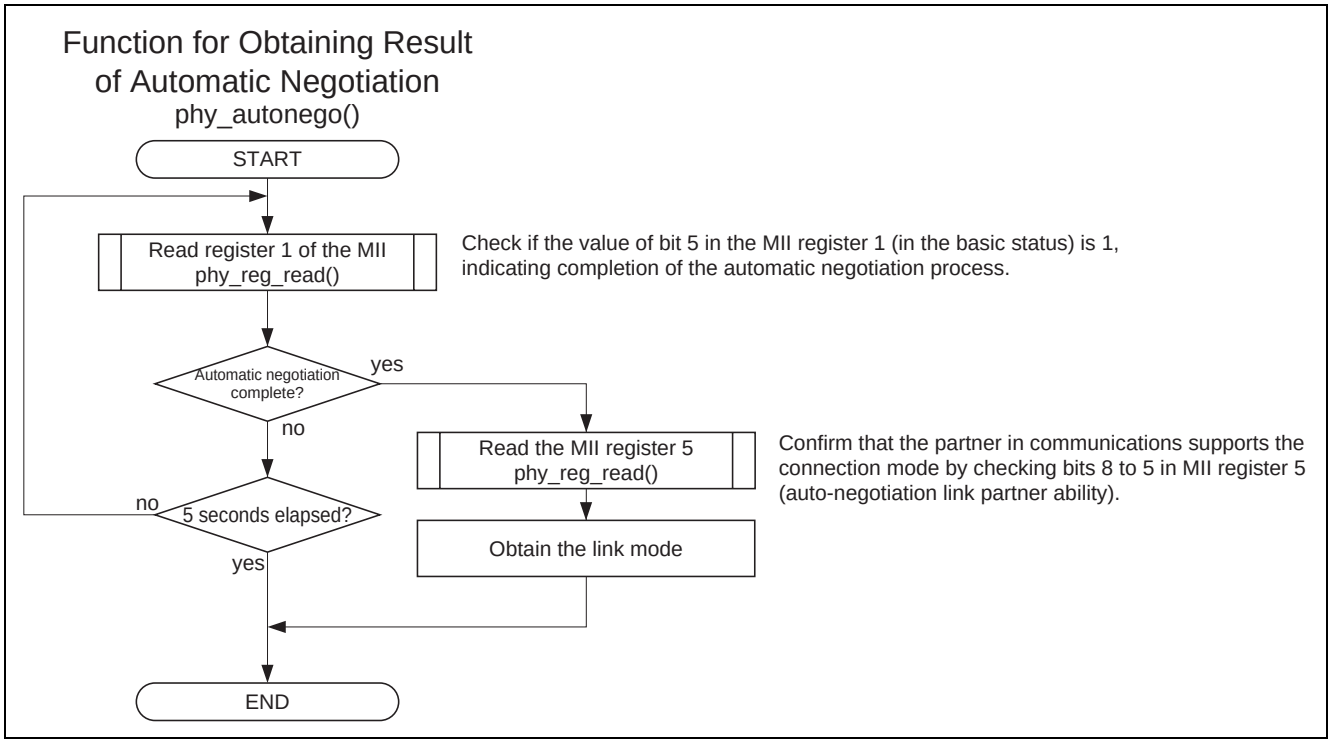


Figure 11 Flow of Handling in the Sample Program (6)

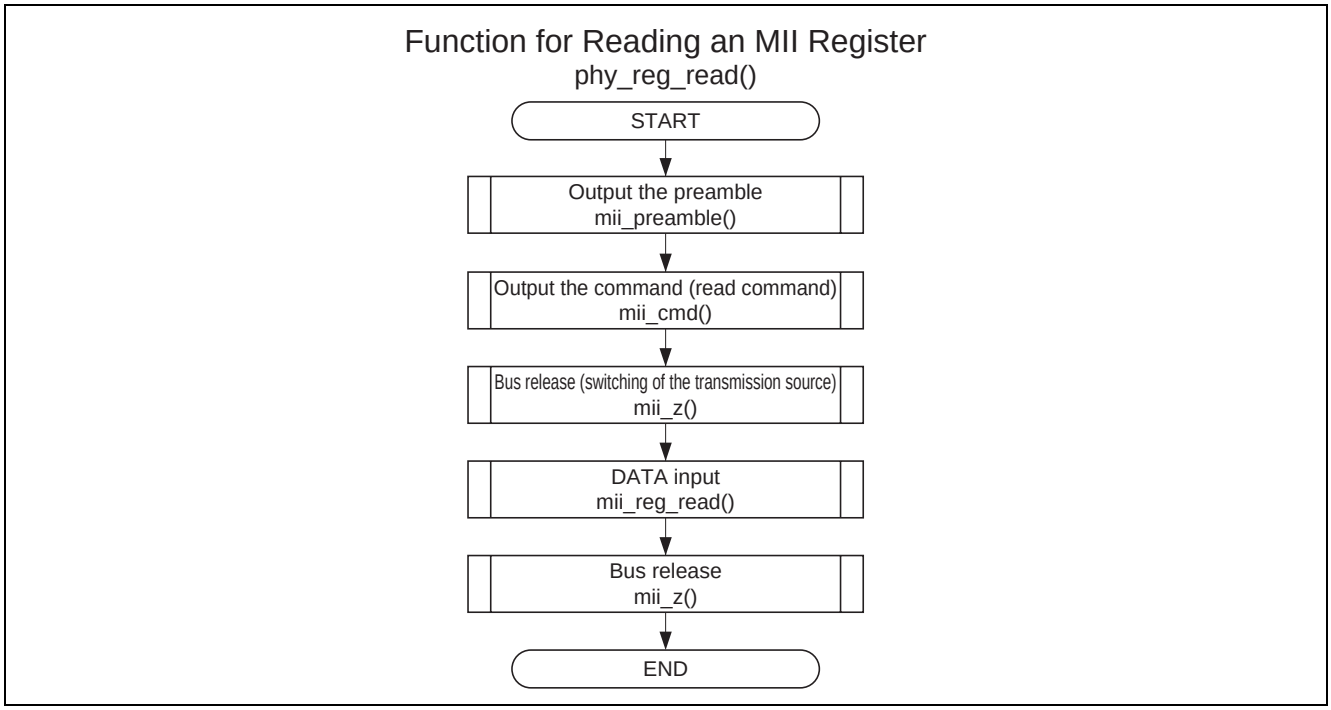


Figure 12 Flow of Handling in the Sample Program (7)

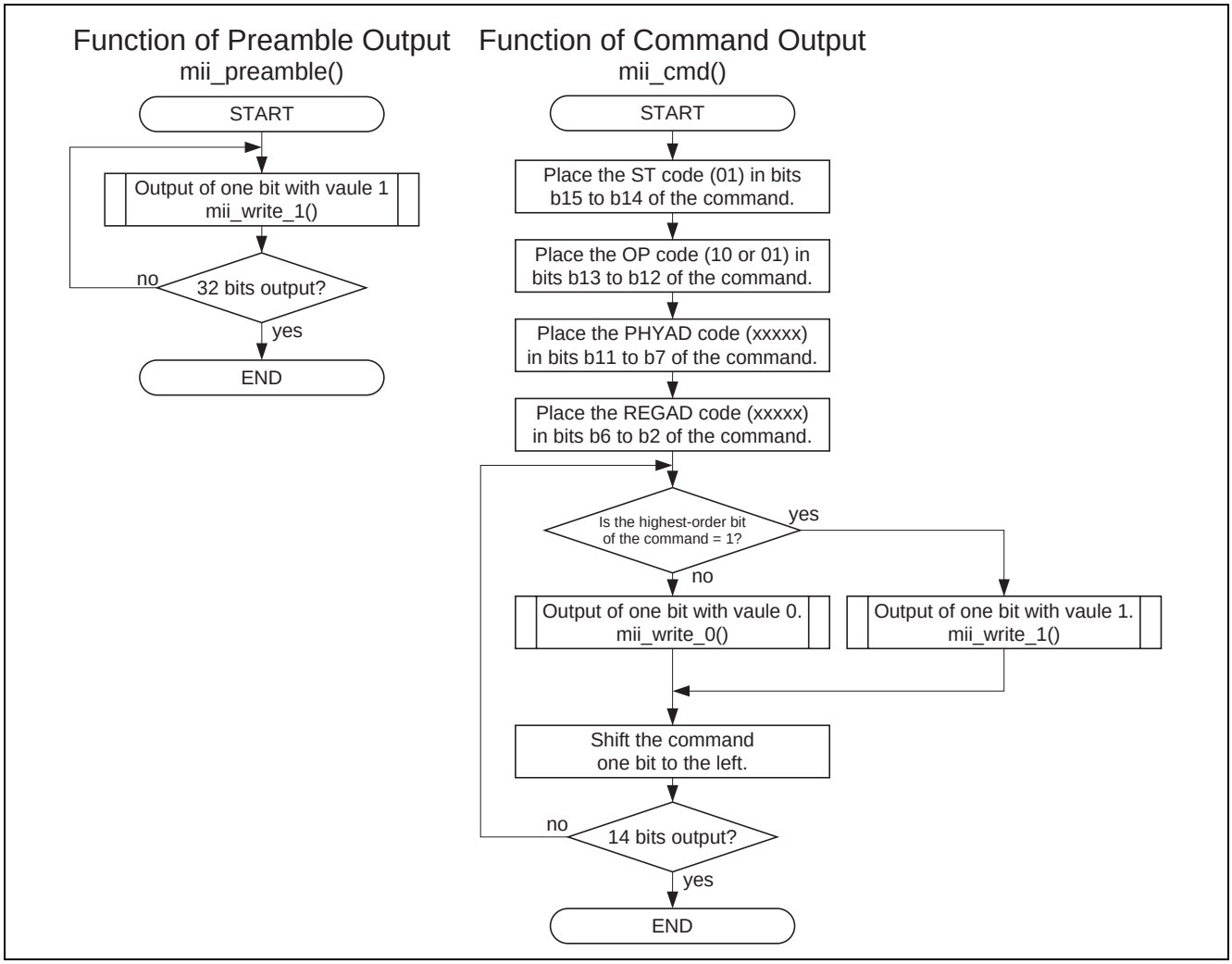


Figure 13 Flow of Handling in the Sample Program (8)

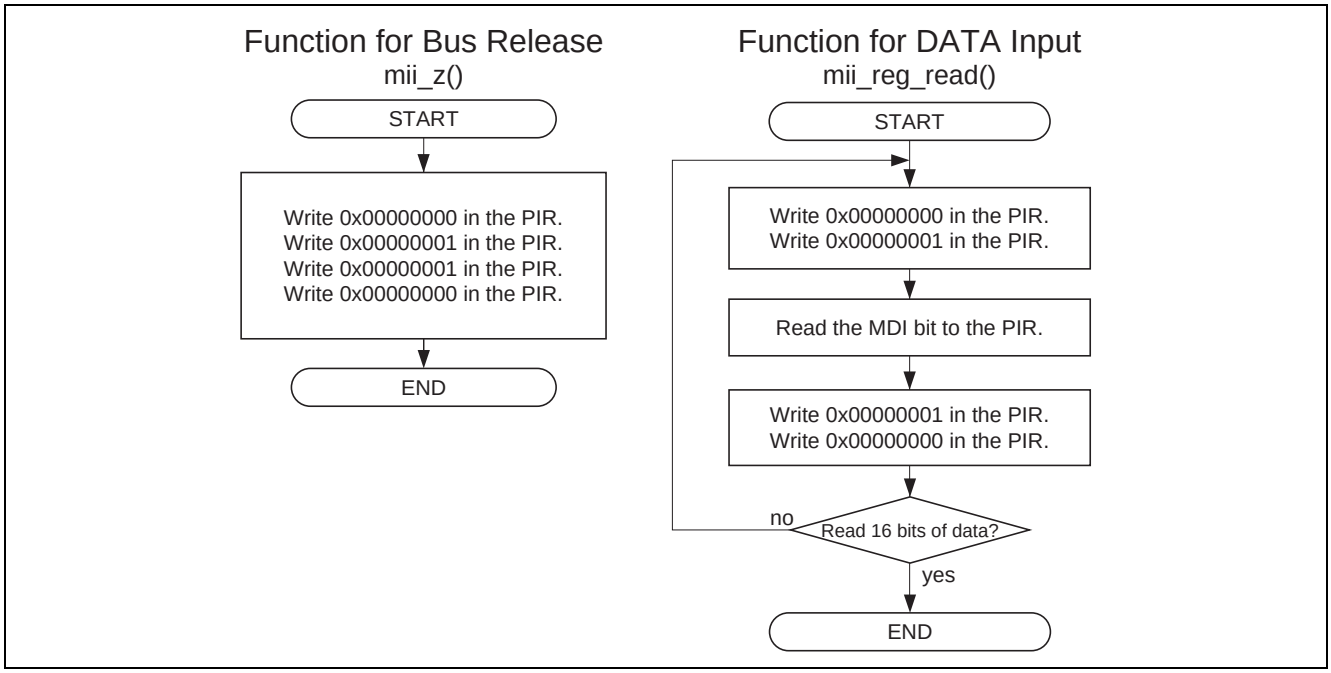


Figure 14 Flow of Handling in the Sample Program (9)

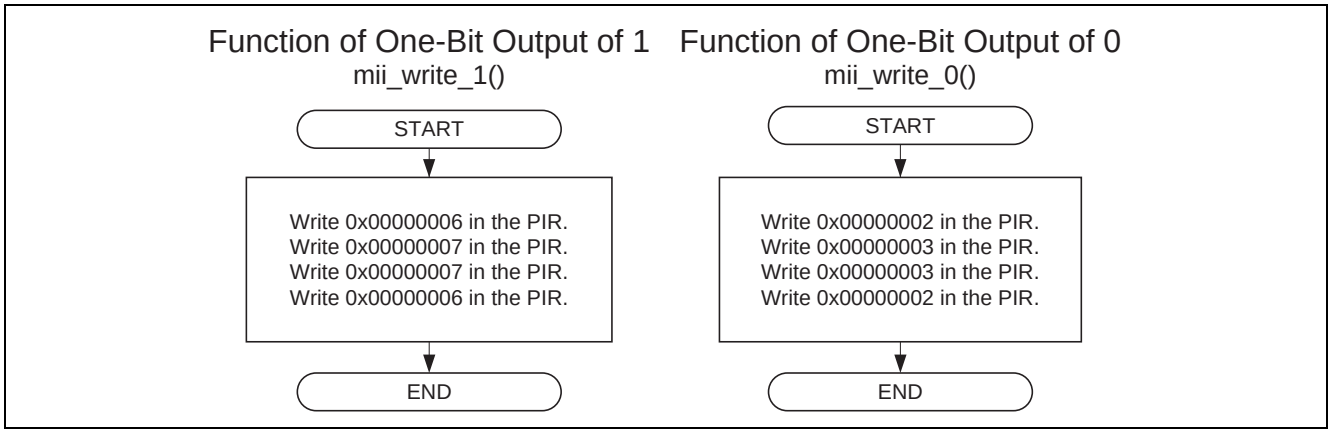


Figure 15 Flow of Handling in the Sample Program (10)

3. Listing of the Sample Program

3.1 Sample Program Listing: "main.c"(1)

```
1     /***********************************************************************
2     * DISCLAIMER
3
4     * This software is supplied by Renesas Technology Corp. and is only
5     * intended for use with Renesas products. No other uses are authorized.
6
7     * This software is owned by Renesas Technology Corp. and is protected under
8     * all applicable laws, including copyright laws.
9
10    * THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    * REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    * INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    * PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    * DISCLAIMED.
15
16    * TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    * TECHNOLOGY CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    * FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    * FOR ANY REASON RELATED TO THE THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    * AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21
22    * Renesas reserves the right, without notice, to make changes to this
23    * software and to discontinue the availability of this software.
24    * By using this software, you agree to the additional terms and
25    * conditions found by accessing the following link:
26    * http://www.renesas.com/disclaimer
27    ***********************************************************************/
28    /* Copyright (C) 2009. Renesas Technology Corp., All Rights Reserved.         */
29    /*""FILE COMMENT""*********** Technical reference data ***************
30    * System Name  : SH7763 Sample Program
31    * File Name    : main.c
32    * Abstract     : Sample Program for Reception of Ethernet Frames
33    * Version      : Ver 1.00
34    * Device       : SH7763
35    * Tool-Chain   : High-performance Embedded Workshop (Version 4.05.01.001)
36    *              : C/C++ Compiler Package for SuperH Family (V.9.03 release00)
37    * OS           : None
38    * H/W Platform : MS7763SE02
39    * Description  : Sample Program for Reception of Ethernet Frames
40    *              :
41    * Operation    :
42    * Limitation   :
43    *              :
44    ********************************************************************
45    * History      : 31.July.2009 Ver. 1.00 First Release
46    *""FILE COMMENT END""*************************************************/
47
48    #include "iodefine.h"
49    #include "ether.h"
50
```

3.2 Sample Program Listing: "main.c"(2)

```
/* **** Prototype declaration **** */
void main(void);

/* **** Variable Declaration **** */
#pragma section ETH_BUFF                    /* Allocated to SDRAM because of its large capacity */
typedef struct{
    unsigned char frame[SIZE_OF_BUFFER];
    int len;
    unsigned char wk[12];
}USER_BUFFER;
static USER_BUFFER recv[10];
#pragma section

/*""FUNC COMMENT""************************************************
* ID                    :
* Outline               : main function
* Include               : #include "iodefine.h"
* Declaration           : void main(void)
* Description           : Receive Ethernet frames.
*                       :
* Argument              : none
* Return Value          : none
* Calling Functions     :
*""FUNC COMMENT END""*****************************************/
void main(void)
{
    int i,j;
    int ret;

    /* ==== Ethernet initial setting ==== */
    ret = lan_open();
    if( ret == OPEN_OK ){
      /* ==== Start reception of 10 frames ==== */
      for(i=0; i<10; i++){
          /* ---- Reception ---- */
          recv[i].len = lan_recv( recv[i].frame );
          if( recv[i].len == 0 ){
              i--;
          }
      }
    }
    /* ==== Ethernet transmission/reception halted ==== */
    lan_close();
}
/* End of file */
```

3.3 Sample Program Listing: "ether.c"(1)

```
1    /***********************************************************************
2    * DISCLAIMER
3
4    * This software is supplied by Renesas Technology Corp. and is only
5    * intended for use with Renesas products. No other uses are authorized.
6
7    * This software is owned by Renesas Technology Corp. and is protected under
8    * all applicable laws, including copyright laws.
9
10   * THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11   * REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12   * INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13   * PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14   * DISCLAIMED.
15
16   * TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17   * TECHNOLOGY CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18   * FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19   * FOR ANY REASON RELATED TO THE THIS SOFTWARE, EVEN IF RENESAS OR ITS
20   * AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21
22   * Renesas reserves the right, without notice, to make changes to this
23   * software and to discontinue the availability of this software.
24   * By using this software, you agree to the additional terms and
25   * conditions found by accessing the following link:
26   * http://www.renesas.com/disclaimer
27   ***********************************************************************/
28   /* Copyright (C) 2009. Renesas Technology Corp., All Rights Reserved.         */
29   /*""FILE COMMENT""************ Technical reference data ***************
30   * System Name  : SH7763 Sample Program
31   * File Name    : ether.c
32   * Abstract     : Sample Program for Reception of Ethernet Frames
33   * Version      : Ver 1.00
34   * Device       : SH7763
35   * Tool-Chain   : High-performance Embedded Workshop (Version 4.05.01.001)
36   *              : C/C++ Compiler Package for SuperH Family (V.9.03 release00)
37   * OS           : None
38   * H/W Platform : MS7763SE02
39   * Description  : Sample Program for Reception of Ethernet Frames
40   *              :
41   * Operation    :
42   * Limitation   :
43   *              :
44   ***********************************************************************
45   * History      : 31.July.2009 Ver. 1.00 First Release
46   *""FILE COMMENT END""*************************************************/
47
48   #include "machine.h"
49   #include "string.h"
50   #include "iodefine.h"
```

3.4 Sample Program Listing: "ether.c"(2)

```
51    #include "phy.h"
52    #include "ether.h"
53
54    /* **** Macro definition **** */
55    #define DEFAULT_MAC_H    0x00010203            /* For debugging */
56    #define DEFAULT_MAC_L    0x00000405
57    #define MACSET_OK        0
58    #define MACSET_NG        -1
59    #define FPGA_ETON        (*(volatile unsigned short *)0xBB00001A)  /* FPGA Ether ON address */
60
61    /* **** Prototype declaration **** */
62    void main(void);
63    void lan_send_handler( unsigned long status );
64    static void lan_desc_create( void );
65    static void lan_reg_reset( void );
66    static void lan_reg_set( int link );
67
68    /* **** Variable Declaration **** */
69    /* ---- Descriptor ---- */
70    #pragma section ETH_DESC                    /* Allocated to a 16-byte boundary */
71    static volatile TXRX_DESCRIPTOR_SET desc;      /* Descriptor area */
72    #pragma section
73    /* ---- Buffer ---- */
74    #pragma section ETH_BUFF                    /* Allocated to a 16-byte boundary */
75    static volatile TXRX_BUFFER_SET  buf;          /* Area for transmission/reception buffer */
76    #pragma section
77    /* ---- MAC address ---- */
78    static unsigned long my_macaddr_h;
79    static unsigned long my_macaddr_l;
80    /* ---- Other ---- */
81    static volatile int c_recv = 0;                /* Received frame counter */
82
83    /*""FUNC COMMENT""************************************************
84    * ID                      :
85    * Outline                 : Ethernet open function
86    * Include                 : #include "iodefine.h"
87    *                         : #include "phy.h"
88    *                         : #include "ether.h"
89    * Declaration             : int lan_open(void)
90    * Description             : Initializes E-DMAC, E-MAC, PHY, and buffer memory.
91    *                         :
92    *                         :
93    * Argument                : none
94    * Return Value            : OPEN_OK(0) :Success in opening
95    *                         : OPEN_NG(-1):Failure in opening
96    * Calling Functions       :
97    *""FUNC COMMENT END""*****************************************/
98    int lan_open(void)
99    {
100       int link;
```

3.5 Sample Program Listing: "ether.c"(3)

```
101
102       /* ==== PFC setting ==== */
103       GPIO.PSEL2.BIT.PTSEL23 = 0;
104       GPIO.PSEL3.BIT.PTSEL33 = 0;
105       GPIO.PSEL4.BIT.PTSEL4A = 3;
106       GPIO.PSEL4.BIT.PTSEL43 = 2;
107       GPIO.PJCR.WORD = 0xFFFC;
108       GPIO.PMCR.WORD = 0x0000;
109       GPIO.POCR.WORD = 0x0000;
110       /* ==== FPGA setting ==== */
111       FPGA_ETON = 0x0001;                     /* FPGA: Ether port usable */
112       /* ==== E-MAC,E-DMAC halted === */
113       lan_reg_reset();
114       /* ==== Buffer initialization ==== */
115       lan_desc_create();
116       /* ==== E-MAC,E-DMAC setting ==== */
117       link = phy_autonego();                  /* Check duplex mode */
118       if( link == NEGO_FAIL ){
119         return OPEN_NG;                   /* OPEN failed */
120       }
121       else{
122         lan_reg_set(link);
123       }
124       return OPEN_OK;
125   }
126
127   /*""FUNC COMMENT""*************************************************
128   * ID                        :
129   * Outline                   : Ethernet close function
130   * Include                   : #include "iodefine.h"
131   *                           : #include "ether.h"
132   * Declaration               : int lan_close(void)
133   * Description               : E-DMAC/E-MAC halted.
134   *                           :
135   * Argument                  : none
136   * Return Value              : none
137   * Calling Functions         :
138   *""FUNC COMMENT END""*********************************************/
139   int lan_close( void )
140   {
141       int i;
142
143       /* ==== Reset E-MAC,E-DMAC === */
144       lan_reg_reset();
145       /* ==== E-DMAC-related interrupts are disabled === */
146       INTC.INT2PRI12.BIT._GETHER = 0;
147
148       return CLOSE_OK;
149   }
150
```

3.6 Sample Program Listing: "ether.c"(4)

```
151  /*""FUNC COMMENT""************************************************
152  * ID                    :
153  * Outline               : Frame reception function
154  * Include               : #include "iodefine.h"
155  *                       : #include "ether.h"
156  * Declaration           : int lan_recv( unsigned char *addr )
157  * Description           : Copies a received frame to the specified buffer.
158  *                       : If there is no received frame, a loop is set up to wait for one.
159  *                       :
160  * Argument              : unsigned char *addr
161  * Return Value          : int : Number of bytes in the received frame (or 0 for error in reception)
162  * Calling Functions     :
163  *""FUNC COMMENT END""*******************************************/
164  int lan_recv( unsigned char *addr )
165  {
166      int i;
167      int pri;
168      int ret = 0;
169      EDMAC_RECV_DESC *p;
170
171      /* ==== Wait for reception ==== */
172      while( c_recv <= 0 ){
173        ;/* wait */
174      }
175      /* ==== Decrement the interrupt count ==== */
176      pri = INTC.INT2PRI12.BIT._GETHER;              /* Exclusive control (interrupt disabled) */
177      INTC.INT2PRI12.BIT._GETHER = 0;
178      --c_recv;
179      INTC.INT2PRI12.BIT._GETHER = pri;
180
181      /* ==== Copy the received frame ==== */
182      p = desc.pRecv_end;
183      if( p->rd0.BIT.RFE == 0 ){
184        memcpy(addr, p->rd2.RBA, p->rd1.RDL);
185        ret = p->rd1.RDL;
186      }
187      /* ---- Receive error ---- */
188      else{
189        p->rd0.LONG &= 0x70000000;               /* Processing for the error flags */
190        ret = 0;                                 /* 0 for error in reception */
191      }
192      /* ==== Restore the descriptor to the state where reception is possible ==== */
193      p->rd0.BIT.RACT = 1;
194      /* ---- Initiate data reception ---- */
195      if( EDMAC0.EDRRR.BIT.RR == 0 ){               /* 0 must be read before writing 1 */
196        EDMAC0.EDRRR.BIT.RR = 1;
197      }
198      /* ==== Update the current pointer value ==== */
199      desc.pRecv_end = p->pNext;
200
```

3.7 Sample Program Listing: "ether.c"(5)

```
201       return ret;
202   }
203
204   /*""FUNC COMMENT""************************************************
205   * ID                    :
206   * Outline               : Descriptor configuration function
207   * Include               : #include "ether.h"
208   * Declaration           : static void lan_desc_create( void )
209   * Description           : Initialize transmit/receive buffer required for Ethernet and
210   *                       : initialize descriptor. One frame/one buffer is assumed.
211   *                       :
212   * Argument              : none
213   * Return Value          : none
214   * Calling Functions     :
215   *""FUNC COMMENT END""*******************************************/
216   static void lan_desc_create( void )
217   {
218       int i;
219       /* ==== Descriptor area configuration ==== */
220       /* ---- Memory clear ---- */
221       memset(&desc, 0, sizeof(desc) );
222       /* ---- Transmit descriptor ---- */
223       for(i=0; i<NUM_OF_TX_DESCRIPTOR; i++){
224         desc.send[i].td2.TBA = buf.send[i];       /* TD2 */
225         desc.send[i].td1.TDL = 0;             /* TD1 */
226         desc.send[i].td0.LONG= 0x30000000;    /* TD0:1frame/1buf, transmission disabled*/
227         if( i != (NUM_OF_TX_DESCRIPTOR-1) ){  /* pNext */
228             desc.send[i].pNext   = &desc.send[i+1];
229         }
230       }
231       desc.send[i-1].td0.BIT.TDLE = 1;
232       desc.send[i-1].pNext        = &desc.send[0];
233       /* ---- Receive descriptor ---- */
234       for(i=0; i<NUM_OF_RX_DESCRIPTOR; i++){
235         desc.recv[i].rd2.RBA = buf.recv[i];       /* RD2 */
236         desc.recv[i].rd1.RBL = SIZE_OF_BUFFER;    /* RD1 */
237         desc.recv[i].rd0.LONG= 0xb0000000;    /* RD0:1frame/1buf, reception enabled*/
238         if( i != (NUM_OF_RX_DESCRIPTOR-1) ){  /* pNext */
239             desc.recv[i].pNext   = &desc.recv[i+1];
240         }
241       }
242       desc.recv[i-1].rd0.BIT.RDLE = 1;              /* Set the last descriptor */
243       desc.recv[i-1].pNext        = &desc.recv[0];
244
245       /* ---- Initialize descriptor management information ---- */
246       desc.pSend_top = &desc.send[0];
247       desc.pRecv_end = &desc.recv[0];
248
249       /* ==== Buffer area configuration ==== */
250       /* ---- Clear the area ---- */
```

3.8 Sample Program Listing: "ether.c"(6)

```
251       memset(&buf, 0, sizeof(buf) );
252   }
253
254   /*""FUNC COMMENT""************************************************
255   * ID                    :
256   * Outline               : E-MAC,E-DMAC registers initialization function
257   * Include               : #include "iodefine.h"
258   * Declaration           : static void lan_reg_reset( void )
259   * Description           : Reset E-MAC and E-DMAC registers
260   *                       :
261   * Argument              : none
262   * Return Value          : none
263   * Calling Functions     :
264   *""FUNC COMMENT END""*******************************************/
265   static void lan_reg_reset( void )
266   {
267       volatile int j = 200;              /* Wait for Bφ256 cycles */
268
269       /* ---- GETHER software reset ---- */
270       GETHER.ARSTR.BIT.ARST =  1;     /* E-DMAC software reset */
271       while(j--){
272         /* Wait for Bφ256 cycles */
273       }
274
275       /* ---- E-DMAC software reset ---- */
276       EDMAC0.EDSR       =  0x00000003;    /* Initiating E-DMAC */
277       EDMAC0.EDMR.LONG =  0x00000003;     /* E-DMAC software reset */
278
279       /* ---- Check clear software reset ---- */
280       while(EDMAC0.EDMR.LONG !=  0x00000000){
281         nop();
282         nop();
283       }
284   }
285
286   /*""FUNC COMMENT""************************************************
287   * ID                    :
288   * Outline               : Setting E-MAC,E-DMAC registers
289   * Include               : #include "iodefine.h"
290   *                       : #include "ether.h"
291   *                       : #include "PHY.h"
292   * Declaration           : void lan_reg_set(int link)
293   * Description           : E-DMAC, E-MAC initialization
294   *                       :
295   * Argument              : int link
296   * Return Value          : none
297   * Calling Functions     :
298   *""FUNC COMMENT END""*******************************************/
299   static void lan_reg_set( int link )
300   {
```

3.9 Sample Program Listing: "ether.c"(7)

```
301       /* ==== EDMAC ==== */
302       EDMAC0.TDLAR      = &desc.send[0];/* Transmit descriptor start            */
303       EDMAC0.RDLAR      = &desc.recv[0];/* Receive descriptor start         */
304       EDMAC0.TDFAR      = &desc.send[0];/* Transmit descriptor fetch address register    */
305       EDMAC0.RDFAR      = &desc.recv[0];/* Receive descriptor fetch address register */
306       EDMAC0.TDFXR      = &desc.send[3];/* Transmit descriptor finished address register */
307       EDMAC0.RDFXR      = &desc.recv[3];/* Receive descriptor finished address register  */
308       EDMAC0.TDFFR      = 0x00000001;   /* Transmit descriptor final flag register   */
309       EDMAC0.RDFFR      = 0x00000001;   /* Receive descriptor final flag register    */
310       EDMAC0.EDMR.LONG  = 0x00000000;   /* Endian not changed (big endian)      */
311                                         /* descriptor length is 16 bytes        */
312       EDMAC0.TRSCER.LONG    = 0x00000000;   /* Copy all status to descriptor */
313       EDMAC0.TFTR.LONG  = 0x00000000;   /* Transmit FIFO threshold: store&forward */
314       EDMAC0.FDR.BIT.TFD  = 0x07;           /* Transmit FIFO capacity of 2048 bytes         */
315       EDMAC0.FDR.BIT.RFD    = 0x1F;         /* Receive FIFO capacity of 8192 bytes          */
316       EDMAC0.RMCR.BIT.RNC   = 1;        /* Continuous reception enabled */
317       EDMAC0.FCFTR.LONG = 0x00170007;   /* Flow control threshold setting, disabled by E-MAC */
318       EDMAC0.RPADIR.LONG    = 0x00000000;   /* No padding insertion                 */
319       /* ==== E-MAC ==== */
320       MAC0.ECMR.LONG        = 0x00000000;   /* Counter clear mode               */
321                                         /* Checksum is not calculated               */
322                                         /* Padding is added to short frame      */
323                                         /* 0TIMEPAUSE frame reception disabled          */
324                                         /* PAUSE frame is not relayed           */
325                                         /* Lost carrier error is checked        */
326                                         /* PAUSE frame is not relayed           */
327                                         /* Flow control disabled                    */
328                                         /* Multi-cast frame other than CAM entry is received */
329                                         /* Magic Packet detection is disabled       */
330                                         /* Reception disabled                           */
331                                         /* Transmission disabled                        */
332                                         /* No internal loopback         */
333                                         /* No external loopback         */
334                                         /* Duplex mode (half-duplex mode)       */
335                                         /* No promiscuous-mode operation    */
336       MAC0.MAHR     = DEFAULT_MAC_H;/* MAC address setting                  */
337       MAC0.MALR     = DEFAULT_MAC_L;
338       MAC0.RFLR.LONG        = 0x00000;      /* Maximum receive frame length of 1518 bytes    */
339       MAC0.PIPR.BIT.PHYIP = 0;          /* ET_PHY-INT pin is low-active         */
340       MAC0.APR.BIT.AP       = 0x0000;       /* TIME parameter value of an automatic PAUSE frame: Flow control is disabled  */
341       MAC0.MPR.BIT.MP       = 0x0000;       /* TIME parameter value of a manual PAUSE frame: Flow control is disabled          */
342       MAC0.TPAUSER      = 0x00000000;   /* Automatic PAUSE frame retransmission count is unlimited       */
343       if( link == FULL_TX || link == FULL_10M ){
344         MAC0.ECMR.BIT.DM = 1;           /* Set to full-duplex mode                  */
345       }
346       if( link == FULL_TX || link == HALF_TX ){
347         MAC0.GECMR.LONG  = 0x00000004;  /* Set to 100 Mbps                      */
348       }
349       else{
350         MAC0.GECMR.LONG  = 0x00000000;  /* Set to 10 Mbps                       */
```

3.10 Sample Program Listing: "ether.c"(8)

```
351       }
352       MAC0.BCULR.BIT.BSTLMT=    0x000;      /* Burst cycle upper-limit is 256 cycles            */
353       /* ==== Interrupt-related ==== */
354       EDMAC0.EESR           = 0xFF3F07FF;   /* Clear all status ( clear by writing 1)   */
355       EDMAC0.EESIPR.LONG    = EDMAC_EESIPR_INI_SEND | EDMAC_EESIPR_INI_RECV | EDMAC_EESIPR_INI_EtherC;
356                                     /* Transmit/receive and E-MAC interrupts enabled        */
357       MAC0.ECSR.LONG    = 0x00000017;   /* Clear all status (clear by writing 1)*/
358       MAC0.ECSIPR.LONG  = EtherC_ECSIPR_INI;/* Enable interrupts                */
359       INTC.INT2PRI12.BIT._GETHER    = 5;        /* Assign the fifth priority level to the E-DMAC interrupt (EINT0)    */
360       INTC.INT2MSKCR1.BIT._GETHER   = 1;        /* GETHER interrupt mask clear  */
361
362       /* ==== Enable transmission/reception ==== */
363       /* ==== Enable E-MAC  ==== */
364       MAC0.ECMR.BIT.RE  = 1;        /* Reception enabled                        */
365       MAC0.ECMR.BIT.TE  = 1;        /* Transmission enabled                         */
366       /* ==== Enable E-DMAC ==== */
367         if(EDMAC0.EDRRR.BIT.RR == 0){
368         EDMAC0.EDRRR.BIT.RR = 1;    /* Initiate data reception                              */
369       }
370   }
371
372   /*""FUNC COMMENT""************************************************
373   * ID                    :
374   * Outline               : Transmit interrupt function
375   * Include               : #include "iodefine.h"
376   *                       : #include "ether.h"
377   * Declaration           : void lan_send_handler( unsigned long status )
378   * Description           : Interrupt handler related to transmission regarding E-DMAC(EESR)
379   *                       :
380   * Argument              : unsigned long status
381   * Return Value          : none
382   * Calling Functions     :
383   *""FUNC COMMENT END""*********************************************/
384   void lan_send_handler( unsigned long status )
385   {
386   }
387
388   /*""FUNC COMMENT""************************************************
389   * ID                    :
390   * Outline               : Receive interrupt function
391   * Include               : #include "iodefine.h"
392   *                       : #include "ether.h"
393   * Declaration           : void lan_recv_handler( unsigned long status )
394   * Description           : Interrupt handler related to reception regarding E-DMAC (EESSR)
395   *                       :
396   * Argument              : unsigned long status
397   * Return Value          : none
398   * Calling Functions     :
399   *""FUNC COMMENT END""*********************************************/
400   void lan_recv_handler( unsigned long status )
```

3.11 Sample Program Listing: "ether.c"(9)

```
401   {
402         c_recv++;                       /* Increment the counter for the number of reception interrupts */
403   }
404
405   /*""FUNC COMMENT""************************************************
406   * ID                        :
407   * Outline                   : E-MAC interrupt function
408   * Include                   : #include "iodefine.h"
409   *                           : #include "ether.h"
410   * Declaration               : void lan_etherc_handler( unsigned long status )
411   * Description               : Interrupt handler regarding E-MAC(ECSR)
412   *                           :
413   * Argument                  : unsigned long status
414   * Return Value              : none
415   * Calling Functions         :
416   *""FUNC COMMENT END""*******************************************/
417   void lan_etherc_handler( unsigned long status )
418   {
419   }
420
421
422   /* End of file */
```

4. Documents for Reference

- Software Manual
  SH-4A Software Manual (REJ09B0003)
  (The most up-to-date versions of the documents are available on the Renesas Technology Website.)

- Hardware Manual
  SH7763 Group Hardware Manual (REJ09B0256)
  (The most up-to-date versions of the documents are available on the Renesas Technology Website.)

Website and Support

Renesas Technology Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/inquiry
csc@renesas.com

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Dec.03.09	—	First edition issued

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
   (1) artificial life support devices or systems
   (2) surgical implantations
   (3) healthcare intervention (e.g., excision, administration of medication, etc.)
   (4) any other purposes that pose a direct threat to human life

   Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.