SH7734 Group

Example of Ethernet Transmit Settings

R01AN0895EJ0100
Rev.1.00
Aug 30, 2012

Introduction

This application note presents a sample program for making Ethernet transmit settings on the SH7734.

Positioning of This Document

This application note is based on the sample program in SH7734 Group: SH7734 Example of Initialization (R01AN0665EJ) and presents a sample program for making settings for the Ethernet function. A description of the sample program for initial settings is omitted. Please refer to the application note SH7734 Group: SH7734 Example of Initialization (R01AN0665EJ).

Target Device

SH7734 Group

In order to use the sample program described in this application note for a microcontroller other than the above, make changes as appropriate to match the microcontroller to be used and perform careful evaluation.

Contents

1. Specifications

The sample program uses the media access control (MAC) function and Ethernet controller direct memory access controller (E-DMAC) function of the gigabit Ethernet controller (GETHER) module of the SH7734 to transmit 10 Ethernet frames (see figure 1.2) to another computer (the opposite host computer). With the exception of the preamble, SFD, and CRC, it is necessary for to user to prepare the transmit data.

The SH7734 supports GMII (Gigabit Media Independent Interface), MII (Media Independent Interface), and RMII (Reduced Media Independent Interface), but the sample program is targeted at an evaluation board that supports RMII. For information on the different settings for each of the above interfaces, see the Gigabit Ethernet Controller (GETHER) section in SH7734 User's Manual: Hardware (R01UH0233EJ).

The sample program makes settings for 10/100 Mbps transfer using the MAC and E-DMAC functions. It does not make use of the following functions of the GETHER:

- 1000 Mbps transfer function using GMII, 10/100 Mbps transfer function using MII
- TSU function
- CAM function
- Flow control
- Magic packet detection
- Checksum calculation function

Table 1.1 Peripheral Functions Used and Their Applications

Peripheral Function	Use
GETHER	Gigabit Ethernet controller: Connects to a physical layer chip (PHY-LSI) to generate and resolve Ethernet frames by using the MAC function and transfer data at high speed to and from the transmit and receive buffers in memory by using the E-DMAC function. For details, see the Gigabit Ethernet Controller (GETHER) section in SH7734 User's Manual: Hardware (R01UH0233EJ).

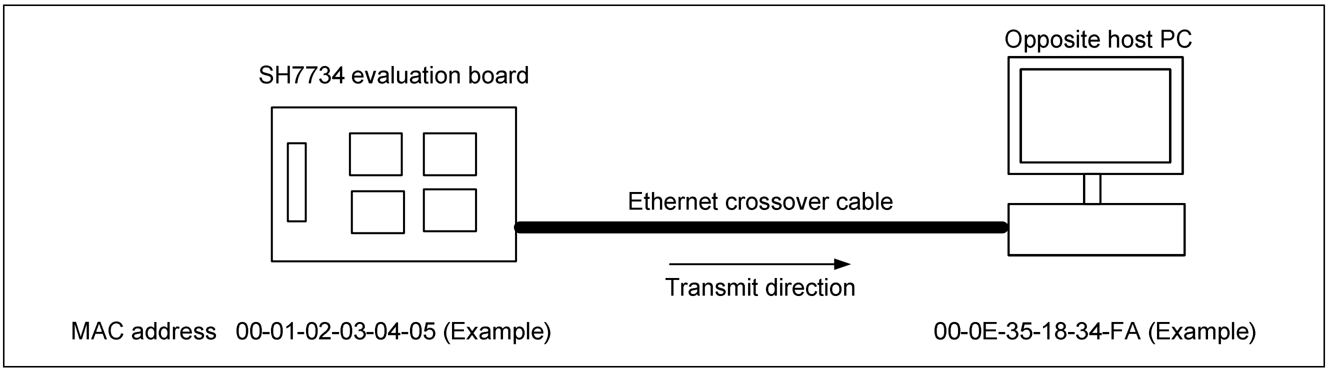
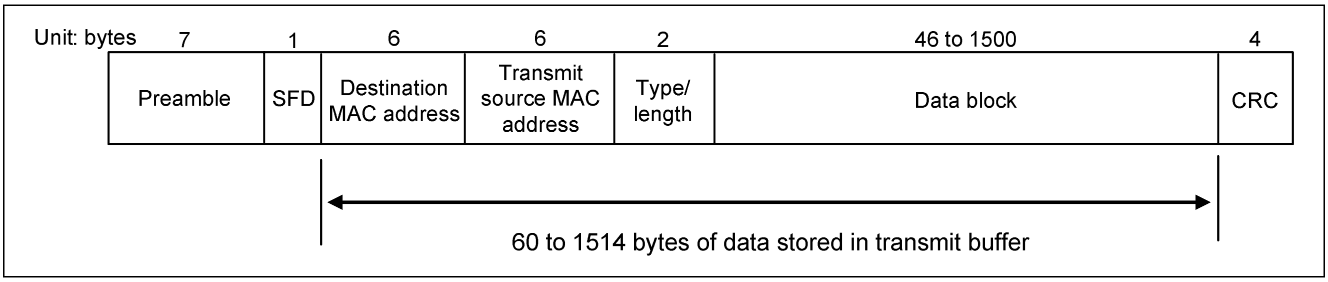


Figure 1.1 Operating Environment



Unit: bytes	7	1	6	6	2	46 to 1500	4
	Preamble	SFD	Destination MAC address	Transmit source MAC address	Type/ length	Data block	CRC

60 to 1514 bytes of data stored in transmit buffer

Figure 1.2 Ethernet Frame Format

2. Operation Confirmation Conditions

The sample code described in this application note has been confirmed to run normally under the operating conditions given below.

Table 2.1 Operating Conditions

Item	Description
Microcontroller used	SH7734 (R8A77343)
Operating frequency	EXTAL input frequency: 33.3333 MHz CPU clock (clki): 400 MHz SHwy clock (clks): 200 MHz SHwy clock (clks1): 100 MHz DDR clock (MCK0/MCK0#/MCK1/MCK1#): 200 MHz Bus clock (lkb): 50 MHz Peripheral clock (clkp): 50 MHz
Operating voltage	IO supply power (3.3 V) Core supply power (1.25 V)
Integrated development environment	Renesas Electronics High-performance Embedded Workshop (Version 4.08.00.011)
C compiler	Renesas Electronics C/C++ Compiler Package for SuperH Family (V.9.04 release00)
	Compiler options: -cpu=sh4a -endian=little -include="$(PROJDIR)¥inc" -change_message=warning -object="$(CONFIGDIR)¥$(FILELEAF).obj" -debug -optimize=0 -gbr=auto -chgincpath -errorpath -global_volatile=0 -opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1 -nologo
Version of the sample code	Ver. 1.00
Endian mode	Little endian
Processing mode	Operation in privileged mode only
Boot mode	CS0 boot mode
Address extension mode	29-bit
Memory management unit (MMU)	Disabled
Watchdog timer (WDT)	Disabled
Board used	Renesas Electronics Corporation SH7734 Evaluation Platform (R0P7734C00000RZ)

3. Related Application Notes

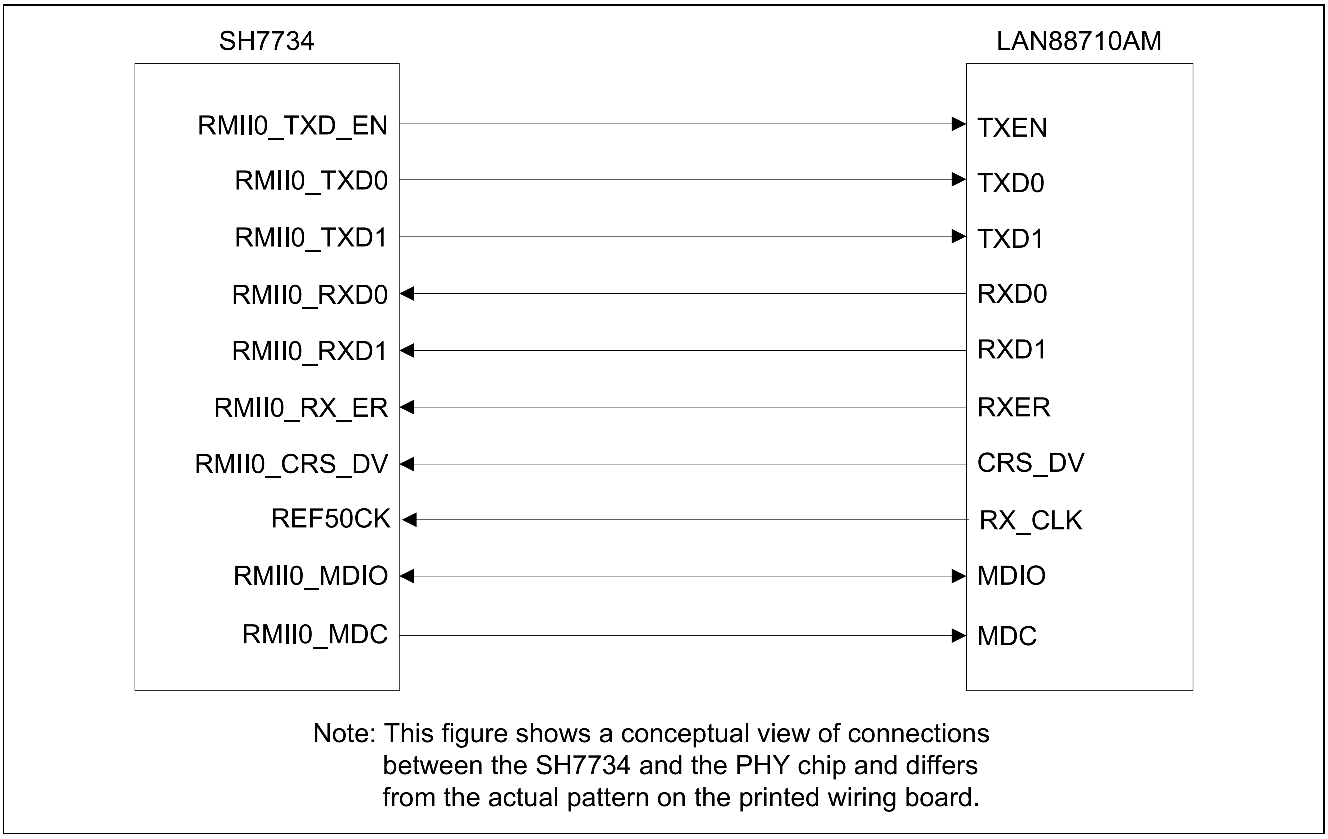
The following application notes are related to this document and should be referred to when using this application note.

- SH7734 Group: SH7734 Example of Initialization (R01AN0665EJ)
- SH7734 Group: SH7734 Example of Ethernet Receive Settings (R01AN0898EJ)

4. Hardware

4.1 Reference Circuit

Figure 4.1 is a connection diagram to an Ethernet PHY-LSI for the RMII interface used by the sample program, using the LAN88710AM manufactured by SMSC as an example. For details of connections to other peripheral circuits, etc., see the technical documentation of the SH7734 Evaluation Platform (R0P7734C00000RZ).



Note: This figure shows a conceptual view of connections between the SH7734 and the PHY chip and differs from the actual pattern on the printed wiring board.

Figure 4.1 PHY-LSI Connection Example (RMII)

4.2 List of Pins Used

Table 4.1 lists the pins used by the sample program and their functions.

Table 4.1 Pins Used and Their Functions

Pin Name	I/O	Description
RMII0_MDC	Output	RMII management data clock
RMII0_MDIO	I/O	RMII management data I/O
RMII0_CRS_DV	Input	RMII carrier detect
RMII0_RX_ER	Input	RMII receive error
RMII0_RXD0	Input	RMII receive data
RMII0_RXD1	Input	RMII receive data
RMII0_TXD_EN	Output	RMII transmit enable
RMII0_TXD0	Output	RMII transmit data
RMII0_TXD1	Output	RMII transmit data
REF50CK	Input	50 MHz reference clock

5. Software

5.1 Operation Overview

Figure 5.1 is a sequence diagram showing an overview of the operation of the sample program.

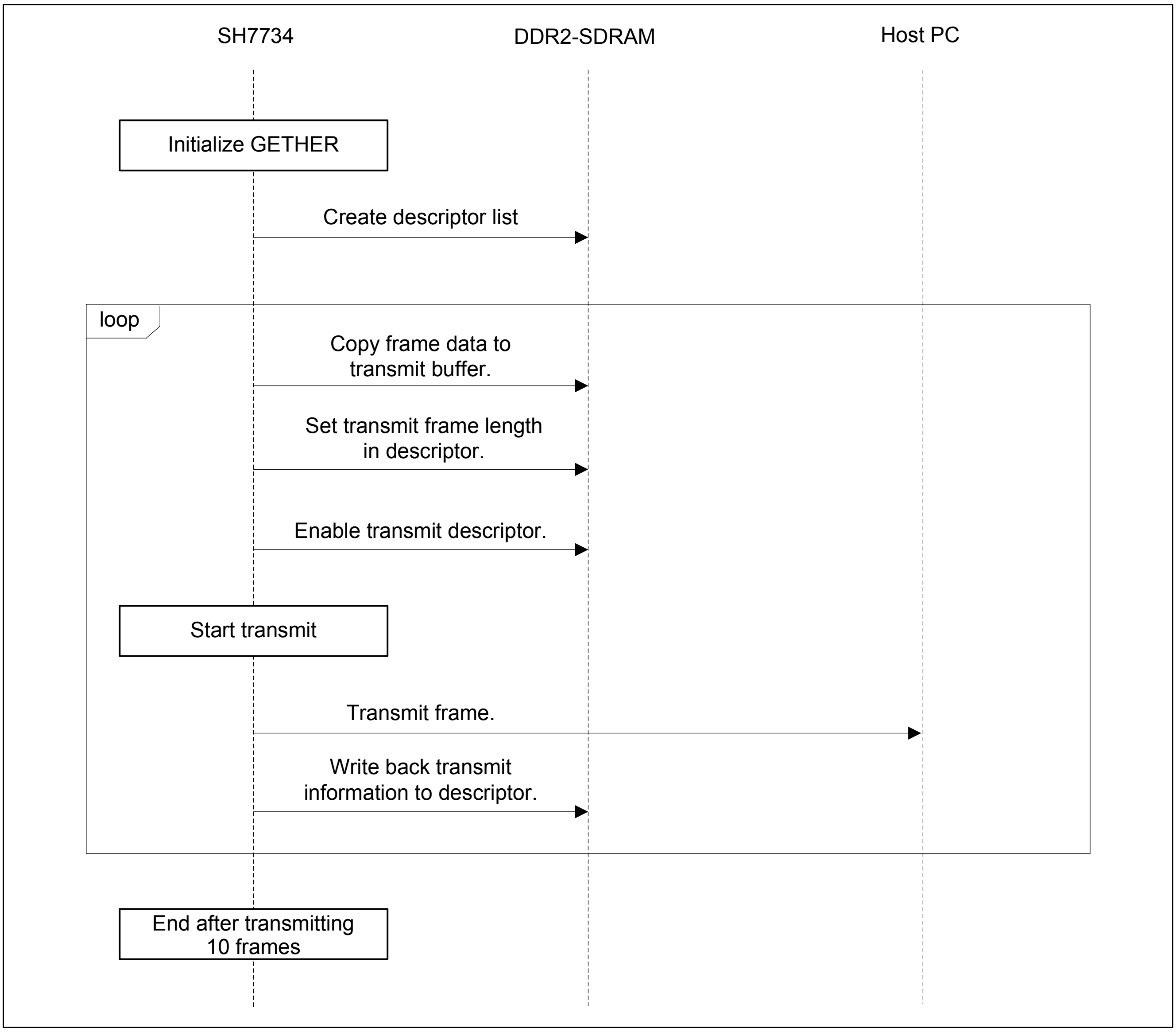

SH7734
DDR2-SDRAM
Host PC
Initialize GETHER
Create descriptor list
loop
Copy frame data to transmit buffer.
Set transmit frame length in descriptor.
Enable transmit descriptor.
Start transmit
Transmit frame.
Write back transmit information to descriptor.
End after transmitting 10 frames

Figure 5.1 Sample Program Operation Overview Sequence Diagram

5.2 File Structure

Table 5.1 lists the files used for the sample code. Note that files generated automatically by the integrated development environment and files from *SH7734 Group: SH7734 Example of Initialization* (R01AN0665EJ) that are used without modification are omitted.

Table 5.1 File Structure

File Name	Overview	Remarks
sh7734_main.c	Ethernet transmit main processing module	
r_ether.c	Ethernet transmit/receive setting module	
r_phy.c	PHY-LSI automatic negotiation processing module	
dbsct.c	Memory initialization	
intprg.c	Definition of Ethernet transmit/receive interrupt handler	
r_ether.h	Include headers for externally referencing Ethernet transmit/receive setting module	
r_phy.h	Include headers for externally referencing PHY-LSI automatic negotiation processing module	
typedefine.h	Declaration headers for variable type names	
vecttbl.src	Exception (reset, general exception, interrupt) function table, interrupt level setting table during exception handling	

5.3 List of Constants

Table 5.2 lists the constants used in the sample code.

Table 5.2 Constants Used in Sample Code

Constant	Setting Value	Description
NUM_OF_USER_BUFFER	10	User data area
NUM_OF_TX_DESCRIPTOR	8	Transmit descriptor count
NUM_OF_RX_DESCRIPTOR	8	Receive descriptor count
NUM_OF_TX_BUFFER	8	Transmit buffer count
NUM_OF_RX_BUFFER	8	Receive buffer count
SIZE_OF_BUFFER	1600	Buffer size
MIN_FRAME_SIZE	60	Min. frame size
MAX_FRAME_SIZE	1514	Max. frame size
LOOP_100us	6700	100 μs software wait
EDMAC_EESIPR_INI_SEND	H'2428 0700	GETHER EESIPR transmit setting Used in interrupt handler to identify transmit interrupt.
EDMAC_EESIPR_INI_RECV	H'0205 001F	GETHER EESIPR receive setting Used in interrupt handler to identify receive interrupt.
EDMAC_EESIPR_INI_EtherC	H'0040 0000	GETHER EESIPR E-MAC status interrupt enable Used in interrupt handler to identify E-MAC interrupt.
EtherC_ECSIPR_INI	H'0000 0004	GETHER ECSIPR setting

Constants related to register addresses, constants generated automatically by the integrated development environment, and constants described in SH7734 Group: SH7734 Example of Initialization (R01AN0665EJ) are not listed in this document.

5.4 List of Structures and Unions

Figure 5.2 lists the structures and unions used in the sample code.

```
/* ==== Transmit descriptor ==== */
typedef union
{
    uint32_t LONG;
    struct{
        uint32_t TACT:1;            /* Transmit descriptor enabled */
        uint32_t TDLE:1;            /* End of transmit descriptor */
        uint32_t TFP :2;            /* Location 1, 0 within transmit frame */
        uint32_t TFE :1;            /* Transmit frame error */
        uint32_t TWBI :1;           /* Write-back completion interrupt notification */
        uint32_t reserved1 :16;     /* Reserved */
        uint32_t TFS9:1;            /* Transmit FIFO underflow (TCU bit in EESR) */
        uint32_t TFS8:1;            /* Transmit abort detect (TABT bit in EESR) */
        uint32_t reserved2 :8;      /* Reserved */
    }BIT;
} td0_t;

typedef struct
{
#if defined(_BIG)
    uint16_t    TDL;                /* Transmit buffer data length (Big endian) */
    uint16_t    reserved;
#else
    uint16_t    reserved;
    uint16_t    TDL;                /* Transmit buffer data length (Little endian) */
#endif
} td1_t;

typedef struct
{
    uint8_t    *TBA;                /* Address of transmit buffer */
} td2_t;

typedef struct tag_edmac_send_desc
{
    td0_t    td0;
    td1_t    td1;
    td2_t    td2;
    struct    tag_edmac_send_desc *pNext;
} edmac_send_desc_t;

/* ==== Receive descriptor ==== */
typedef union
{
    uint32_t LONG;
    struct{
        uint32_t RACT:1;        /* Receive descriptor enabled */
        uint32_t RDLE:1;        /* End of receive descriptor */
        uint32_t RFP :2;        /* Location 1,0 within receive frame */
        uint32_t RFE :1;        /* Receive frame error */
        uint32_t PV  :1;        /* Padding insertion */
        uint32_t reserved1:16;  /* Reserved */
        uint32_t RFS9:1;        /* Receive FIFO overflow (RFOF bit in EESR) */
        uint32_t RFS8:1;        /* Receive abort detect (RABT bit in EESR) */
```

Note: Two versions of the structure are denoted, one for each endian setting, to enable the order of the members in the structure to be reversed to match the endian mode.

```
        uint32_t RFS7:1;        /* Receive multicast frames (RMAF bit in EESR) */
        uint32_t RFS6:1;        /* Carrier extension error (CEEF bit in EESR) */
        uint32_t RFS5:1;        /* Carrier extension loss (CELF bit in EESR) */
        uint32_t RFS4:1;        /* Residual bits frame receive error (RRF bit in EESR) */
        uint32_t RFS3:1;        /* Long frame receive error (RTLE bit in EESR) */
        uint32_t RFS2:1;        /* Short frame receive error (RTSF bit in EESR) */
        uint32_t RFS1:1;        /* PHY-LSI receive error (PRE bit in EESR) */
        uint32_t RFS0:1;        /* Receive frame CRC error detected (CERF bit in EESR) */
    }BIT;
} rd0_t;

typedef struct
{
#if defined(_BIG)
    uint16_t    RBL;            /* Receive buffer length (Big endian) */
    uint16_t    RDL;            /* Receive data length (Big endian) */
#else
    uint16_t    RDL;            /* Receive data length (Little endian) */
    uint16_t    RBL;            /* Receive buffer length (Little endian) */
#endif
} rd1_t;

typedef struct
{
    uint8_t *RBA;                   /* Receive buffer address */
} rd2_t;

typedef struct tag_edmac_recv_desc
{
    rd0_t    rd0;
    rd1_t    rd1;
    rd2_t    rd2;
    struct tag_edmac_recv_desc *pNext;
} edmac_recv_desc_t;

/* ==== The whole transmit/receive descriptors (must be allocated in 16-byte boundaries) ==== */
typedef struct
{
    edmac_send_desc_t send[NUM_OF_TX_DESCRIPTOR];
    edmac_recv_desc_t recv[NUM_OF_RX_DESCRIPTOR];
    edmac_send_desc_t *pSend_top;           /* Registration location of transmit descriptors */
    edmac_recv_desc_t *pRecv_end;           /* Registration location and reception end of transmit descriptors */
} txrx_descriptor_set_t;

/* ==== Transmit/receive buffers (must be allocated in 32-byte boundaries) ==== */
/* ---- Definition of all transmit/receive buffer areas ---- */
typedef struct
{
    uint8_t send[NUM_OF_TX_BUFFER][SIZE_OF_BUFFER];
    uint8_t recv[NUM_OF_RX_BUFFER][SIZE_OF_BUFFER];
} txrx_buffer_set_t;
```

Note: Two versions of the structure are denoted, one for each endian setting, to enable the order of the members in the structure to be reversed to match the endian mode.

Figure 5.2 Structures and Unions Used in Sample Code

5.5 List of Variables

Table 5.3 lists the static variables.

Table 5.3 Static Variables

Type	Variable Name	Description	Used by Function
static uint8_t	s_frame	Transmit frame data	R_Ether_Write
static uint8_t	mac_addr	MAC address	R_Ether_Open
static volatile txrx_descriptor_set_t	eth_desc	Descriptor area	R_Ether_Write lan_desc_create
static volatile txrx_buffer_set_t	eth_buf	Transmit buffer area	lan_desc_create

5.6 List of Functions

Table 5.4 lists the functions.

Table 5.4 FunctionsTable

Function Name	Description
R_Ether_Open	GETHER open function
R_Ether_Write	GETHER frame transmit function
R_Ether_Close	GETHER close function

5.7 Function Specifications

The specifications of the functions of the sample code are listed below.

R_Ether_Open

Overview	Initializes the GETHER module.	
Header	r_ether.h	
Declaration	int R_Ether_Open(uint32_t ch, uint8_t mac_addr[])	
Description	Initializes the GETHER module.	
Arguments	• uint32_t ch • uint8_t mac_addr[]	E-MAC channel number E-MAC MAC address
Return values	• [R_ETHER_OK(0)]: Open successful • [R_ETHER_ERROR(-1)]: Open failed	
Notes	This function initializes the GETHER module, using the MAC address specified as an argument. When 0 is specified as the MAC address, an address is acquired from the system in EEPROM, etc. The implementation should match the usage conditions. Note that the SH7734 provides a single channel as an Ethernet port, so the argument specifying the E-MAC channel number should be set to 0.	

R_Ether_Write

Overview	Performs Ethernet frame transmit processing.	
Header	r_ether.h	
Declaration	int32_t R_Ether_Write (uint32_t ch, void * buf, uint32_t len)	
Description	This function sets the transmit data in the transmit buffer and updates the descriptor information. The data set in the transmit buffer is transmitted by the E-MAC. If no free transmit descriptors are available, the function does not wait but returns an error (R_ETHER_ERROR) and ends. If the transmit data does not constitute 60 bytes, padding is added.	
Arguments	• uint32_t ch • void *buf • uint32_t len	E-MAC channel number Transmit data pointer Ethernet frame length
Return values	• [R_ETHER_OK(0)]: Normal end • [R_ETHER_ERROR(-1)]: Error generated	
Notes	The SH7734 provides a single channel as an Ethernet port, so the argument specifying the E-MAC channel number should be set to 0.	

R_Ether_Close

Overview	Resets and stops the GETHER module.	
Header	r_ether.h	
Declaration	int R_Ether_Close(uint32_t ch)	
Description	Resets and stops the GETHER module.	
Arguments	uint32_t ch	E-MAC channel number
Return values	• [R_ETHER_OK(0)]: Close successful • [R_ETHER_ERROR(-1)]: Close failed	
Notes	Note that the Ethernet driver of the sample program does not use the R_ETHER_ERROR(-1) return value. Also, the SH7734 provides a single channel as an Ethernet port, so the argument specifying the E-MAC channel number should be set to 0.	

5.8 Flowcharts

5.8.1 Main Processing

Figure 5.3 is a flowchart of the main processing routine.

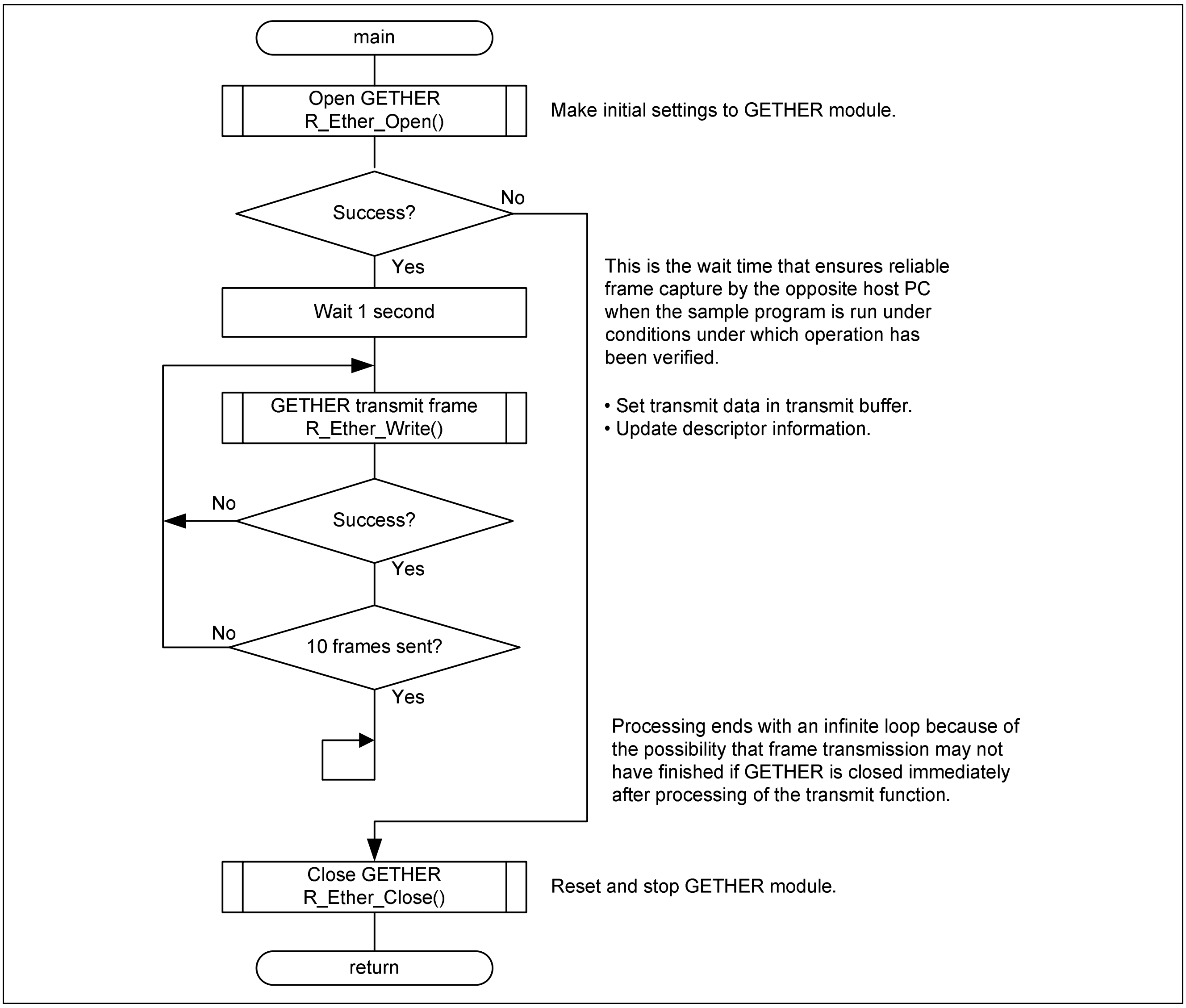

main
Open GETHER
R_Ether_Open()
Make initial settings to GETHER module.
Success?
No
Yes
Wait 1 second
This is the wait time that ensures reliable frame capture by the opposite host PC when the sample program is run under conditions under which operation has been verified.
GETHER transmit frame
R_Ether_Write()
• Set transmit data in transmit buffer.
• Update descriptor information.
No
Success?
Yes
No
10 frames sent?
Yes
Processing ends with an infinite loop because of the possibility that frame transmission may not have finished if GETHER is closed immediately after processing of the transmit function.
Close GETHER
R_Ether_Close()
Reset and stop GETHER module.
return

Figure 5.3 Main Processing

5.8.2 GETHER Open Processing

Figure 5.4 is a flowchart of the GETHER open processing.

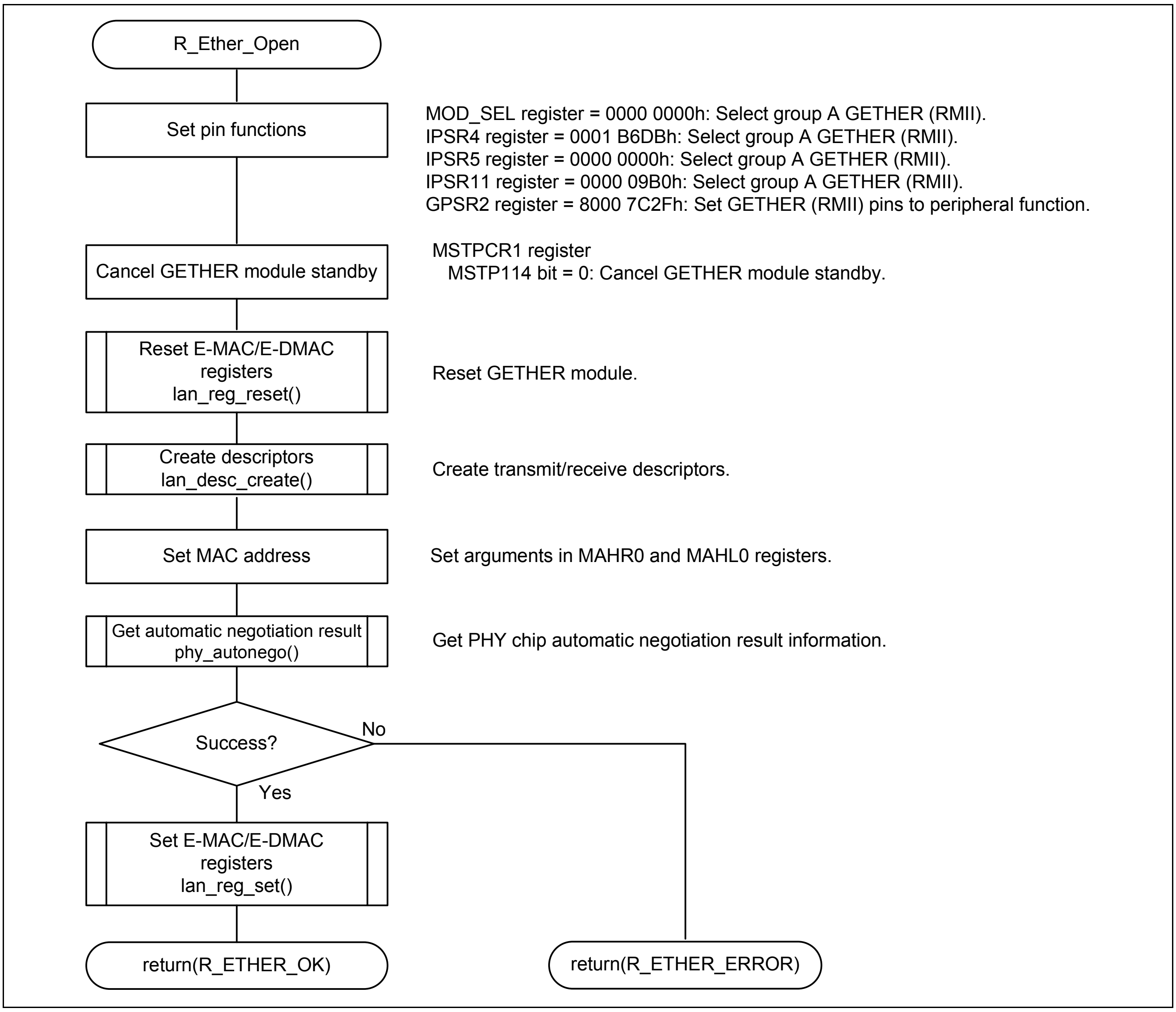

R_Ether_Open
Set pin functions
MOD_SEL register = 0000 0000h: Select group A GETHER (RMII).
IPSR4 register = 0001 B6DBh: Select group A GETHER (RMII).
IPSR5 register = 0000 0000h: Select group A GETHER (RMII).
IPSR11 register = 0000 09B0h: Select group A GETHER (RMII).
GPSR2 register = 8000 7C2Fh: Set GETHER (RMII) pins to peripheral function.
Cancel GETHER module standby
MSTPCR1 register
MSTP114 bit = 0: Cancel GETHER module standby.
Reset E-MAC/E-DMAC registers lan_reg_reset()
Reset GETHER module.
Create descriptors lan_desc_create()
Create transmit/receive descriptors.
Set MAC address
Set arguments in MAHR0 and MAHL0 registers.
Get automatic negotiation result phy_autonego()
Get PHY chip automatic negotiation result information.
Success?
No
Yes
Set E-MAC/E-DMAC registers lan_reg_set()
return(R_ETHER_OK)
return(R_ETHER_ERROR)

Figure 5.4 GETHER Open Processing

5.8.3 GETHER Close Processing

Figure 5.5 is a flowchart of the GETHER close processing routine.

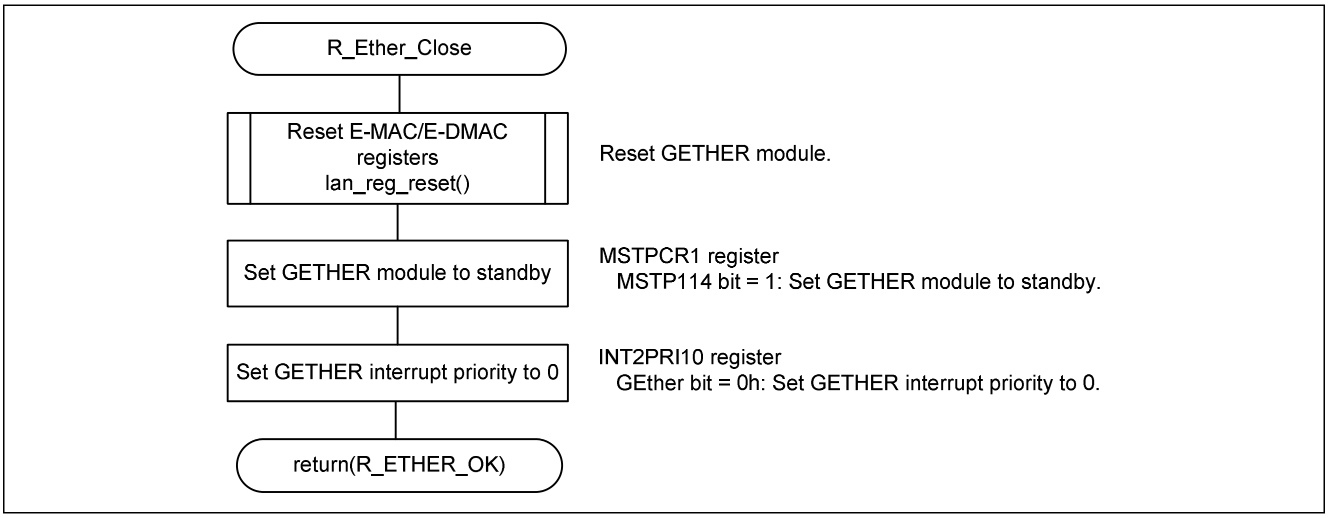

R_Ether_Close
Reset E-MAC/E-DMAC registers lan_reg_reset()
Reset GETHER module.
Set GETHER module to standby
MSTPCR1 register
MSTP114 bit = 1: Set GETHER module to standby.
Set GETHER interrupt priority to 0
INT2PRI10 register
GEther bit = 0h: Set GETHER interrupt priority to 0.
return(R_ETHER_OK)

Figure 5.5 GETHER Close Processing

5.8.4 GETHER Frame Transmit Processing

Figure 5.6 is a flowchart of the GETHER frame transmit processing routine.

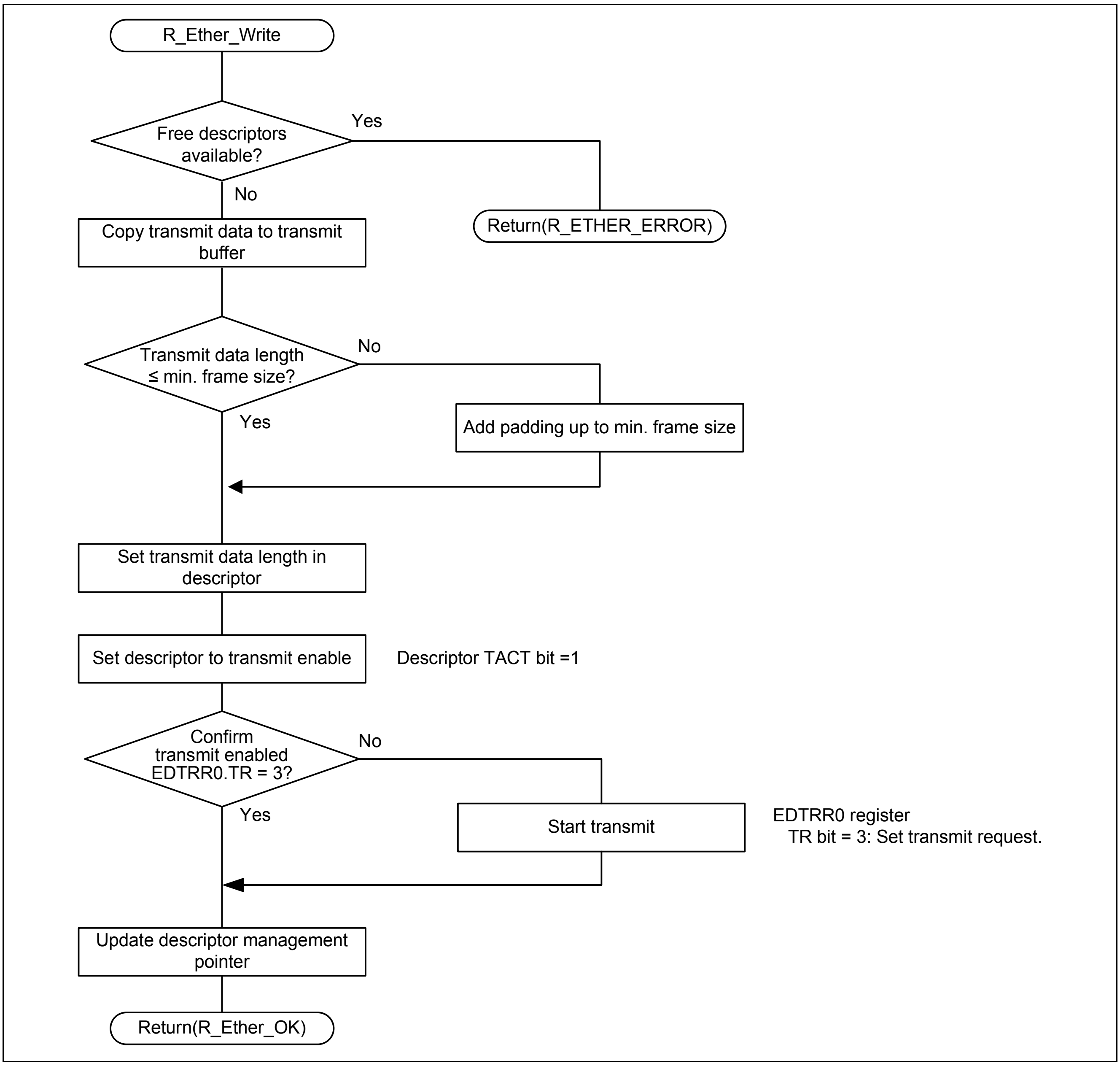

R_Ether_Write
Free descriptors available?
Yes
No
Return(R_ETHER_ERROR)
Copy transmit data to transmit buffer
Transmit data length ≤ min. frame size?
No
Yes
Add padding up to min. frame size
Set transmit data length in descriptor
Set descriptor to transmit enable
Descriptor TACT bit =1
Confirm transmit enabled EDTRR0.TR = 3?
No
Yes
Start transmit
EDTRR0 register
TR bit = 3: Set transmit request.
Update descriptor management pointer
Return(R_Ether_OK)

Figure 5.6 GETHER Frame Transmit Processing

5.8.5 E-MAC/EDMAC Reset Function

Figure 5.7 is a flowchart of the E-MAC/EDMAC reset function.

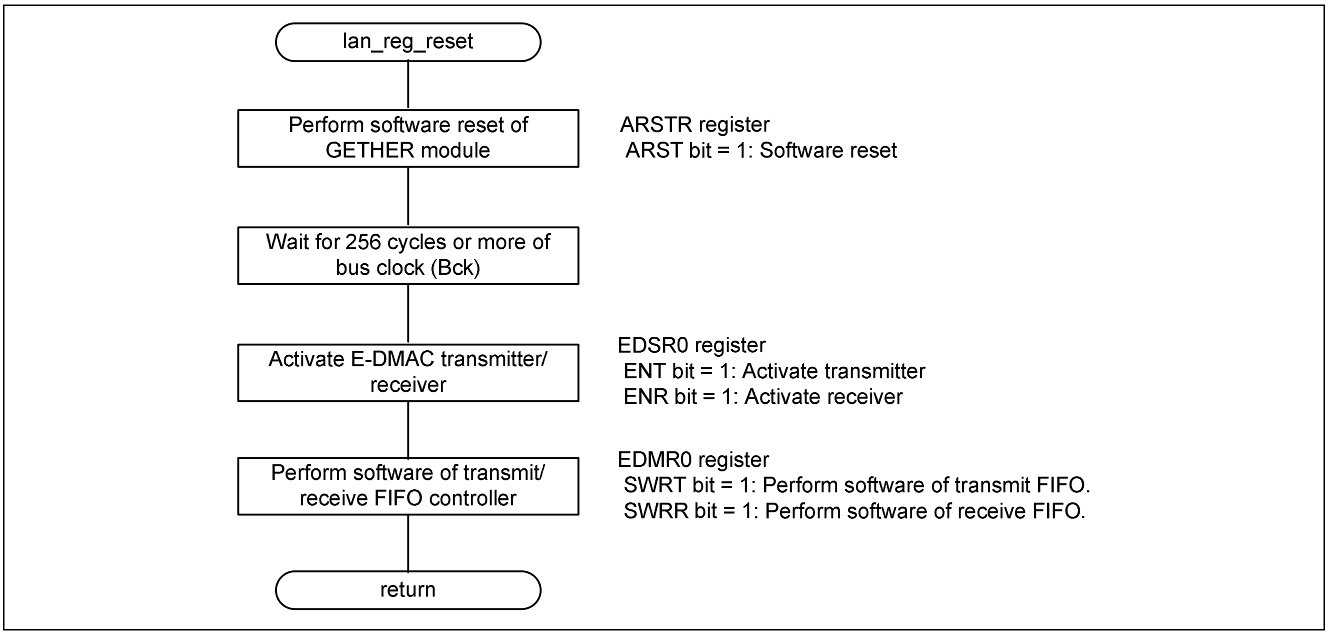

lan_reg_reset
Perform software reset of GETHER module
ARSTR register
ARST bit = 1: Software reset
Wait for 256 cycles or more of bus clock (Bck)
Activate E-DMAC transmitter/ receiver
EDSR0 register
ENT bit = 1: Activate transmitter
ENR bit = 1: Activate receiver
Perform software of transmit/ receive FIFO controller
EDMR0 register
SWRT bit = 1: Perform software of transmit FIFO.
SWRR bit = 1: Perform software of receive FIFO.
return

Figure 5.7 E-MAC/EDMAC Reset Function

5.8.6 Transmit/Receive Descriptor Initialization Function

Figure 5.8 is a flowchart of the transmit/receive descriptor initialization function.

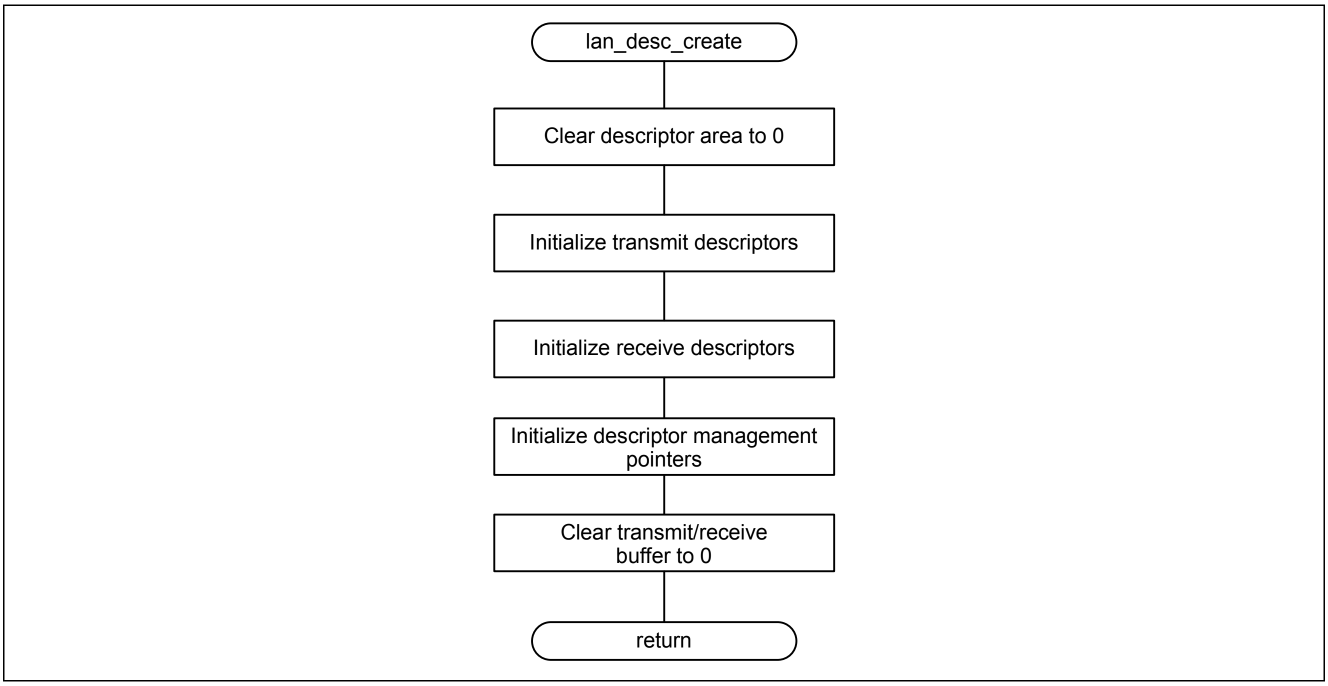

lan_desc_create
Clear descriptor area to 0
Initialize transmit descriptors
Initialize receive descriptors
Initialize descriptor management pointers
Clear transmit/receive buffer to 0
return

Figure 5.8 Transmit/Receive Descriptor Initialization Function

5.8.7 E-MAC/E-DMAC Register Setting Function

Figure 5.9 is a flowchart of the E-MAC/E-DMAC register setting function.

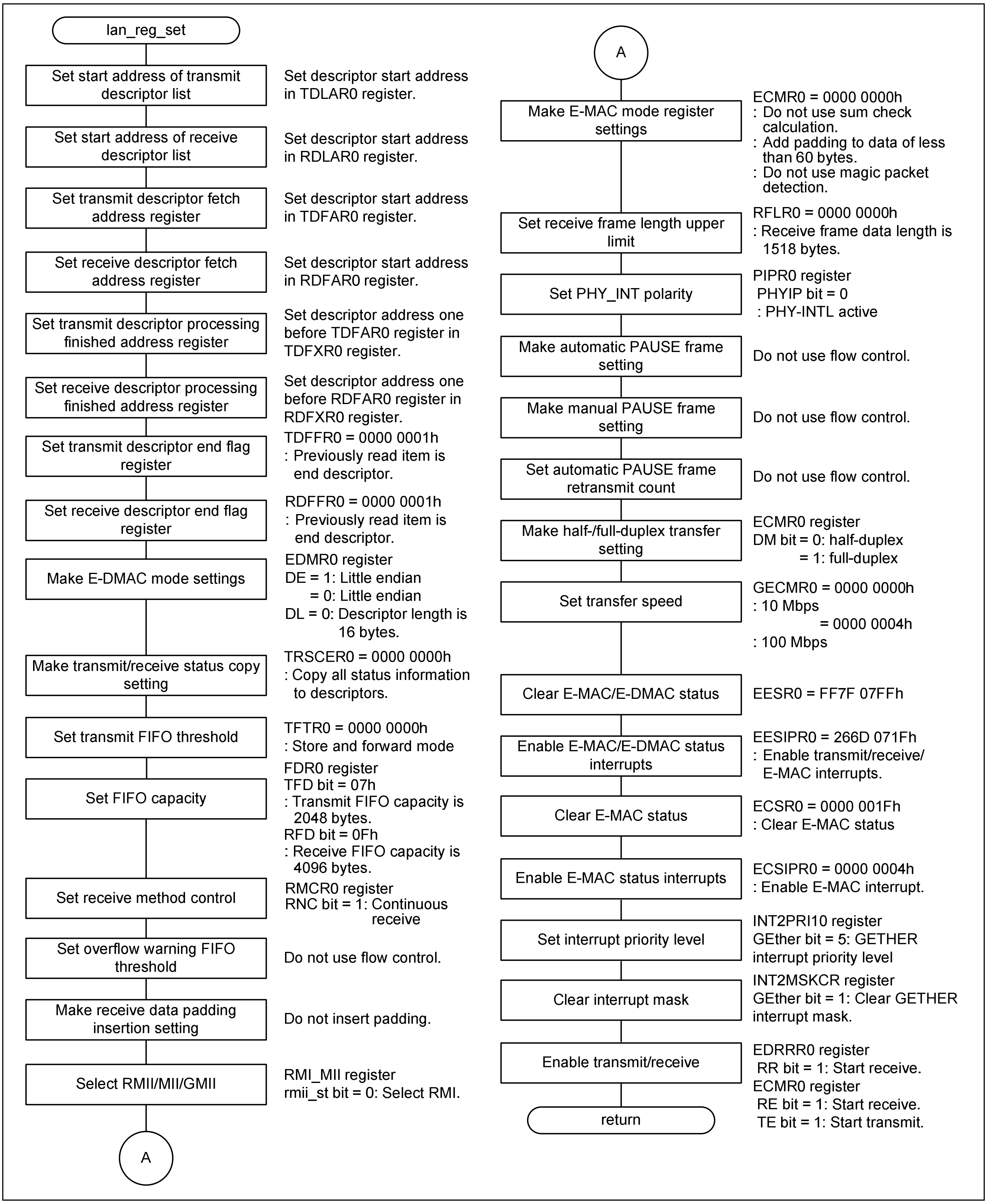

lan_reg_set
Set start address of transmit descriptor list
Set descriptor start address in TDLAR0 register.
Set start address of receive descriptor list
Set descriptor start address in RDLAR0 register.
Set transmit descriptor fetch address register
Set descriptor start address in TDFAR0 register.
Set receive descriptor fetch address register
Set descriptor start address in RDFAR0 register.
Set transmit descriptor processing finished address register
Set descriptor address one before TDFAR0 register in TDFXR0 register.
Set receive descriptor processing finished address register
Set descriptor address one before RDFAR0 register in RDFXR0 register.
Set transmit descriptor end flag register
TDFFR0 = 0000 0001h
: Previously read item is end descriptor.
Set receive descriptor end flag register
RDFFR0 = 0000 0001h
: Previously read item is end descriptor.
Make E-DMAC mode settings
EDMR0 register
DE = 1: Little endian
= 0: Little endian
DL = 0: Descriptor length is 16 bytes.
Make transmit/receive status copy setting
TRSCER0 = 0000 0000h
: Copy all status information to descriptors.
Set transmit FIFO threshold
TFTR0 = 0000 0000h
: Store and forward mode
Set FIFO capacity
FDR0 register
TFD bit = 07h
: Transmit FIFO capacity is 2048 bytes.
RFD bit = 0Fh
: Receive FIFO capacity is 4096 bytes.
Set receive method control
RMCR0 register
RNC bit = 1: Continuous receive
Set overflow warning FIFO threshold
Do not use flow control.
Make receive data padding insertion setting
Do not insert padding.
Select RMII/MII/GMII
RMI_MII register
rmii_st bit = 0: Select RMI.
A
A
Make E-MAC mode register settings
ECMR0 = 0000 0000h
: Do not use sum check calculation.
: Add padding to data of less than 60 bytes.
: Do not use magic packet detection.
Set receive frame length upper limit
RFLR0 = 0000 0000h
: Receive frame data length is 1518 bytes.
Set PHY_INT polarity
PIPR0 register
PHYIP bit = 0
: PHY-INTL active
Make automatic PAUSE frame setting
Do not use flow control.
Make manual PAUSE frame setting
Do not use flow control.
Set automatic PAUSE frame retransmit count
Do not use flow control.
Make half-/full-duplex transfer setting
ECMR0 register
DM bit = 0: half-duplex
= 1: full-duplex
Set transfer speed
GECMR0 = 0000 0000h
: 10 Mbps
= 0000 0004h
: 100 Mbps
Clear E-MAC/E-DMAC status
EESR0 = FF7F 07FFh
Enable E-MAC/E-DMAC status interrupts
EESIPR0 = 266D 071Fh
: Enable transmit/receive/ E-MAC interrupts.
Clear E-MAC status
ECSR0 = 0000 001Fh
: Clear E-MAC status
Enable E-MAC status interrupts
ECSIPR0 = 0000 0004h
: Enable E-MAC interrupt.
Set interrupt priority level
INT2PRI10 register
GEther bit = 5: GETHER interrupt priority level
Clear interrupt mask
INT2MSKCR register
GEther bit = 1: Clear GETHER interrupt mask.
Enable transmit/receive
EDRRR0 register
RR bit = 1: Start receive.
ECMR0 register
RE bit = 1: Start receive.
TE bit = 1: Start transmit.
return

Figure 5.9 E-MAC/E-DMAC Register Setting Function

5.8.8 GETHER Interrupt Handler

Figure 5.10 is a flowchart of the GETHER interrupt handler.

In the sample program presented here, no particular processing is performed.

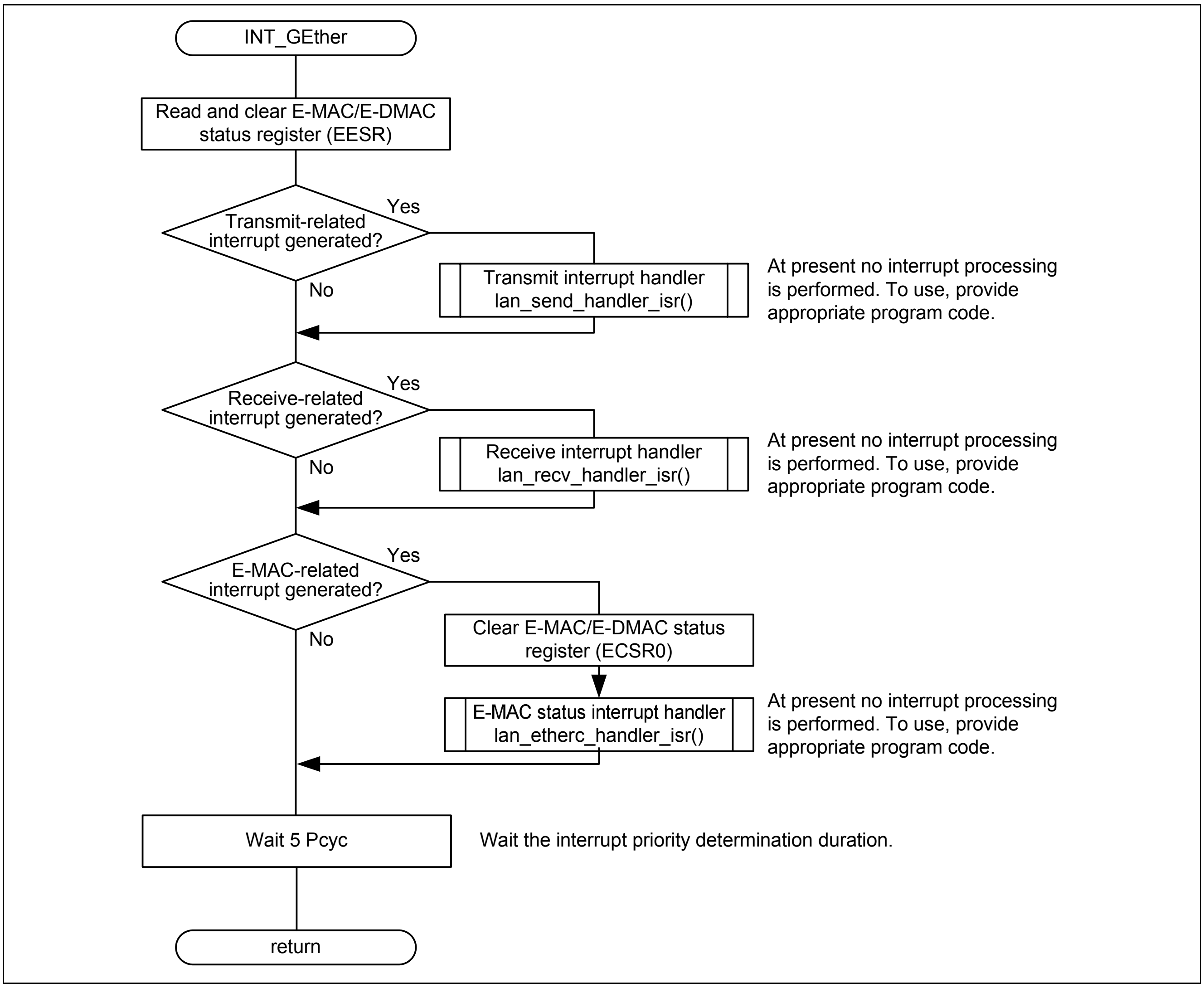

INT_GEther
Read and clear E-MAC/E-DMAC status register (EESR)
Transmit-related interrupt generated?
Yes
No
Transmit interrupt handler lan_send_handler_isr()
At present no interrupt processing is performed. To use, provide appropriate program code.
Receive-related interrupt generated?
Yes
No
Receive interrupt handler lan_recv_handler_isr()
At present no interrupt processing is performed. To use, provide appropriate program code.
E-MAC-related interrupt generated?
Yes
No
Clear E-MAC/E-DMAC status register (ECSR0)
E-MAC status interrupt handler lan_etherc_handler_isr()
At present no interrupt processing is performed. To use, provide appropriate program code.
Wait 5 Pcyc
Wait the interrupt priority determination duration.
return

Figure 5.10 GETHER Interrupt Handler

5.8.9 PHY-LSI Automatic Negotiation Result Acquisition Processing

Figure 5.11 is a flowchart of the PHY-LSI automatic negotiation result acquisition processing routine.

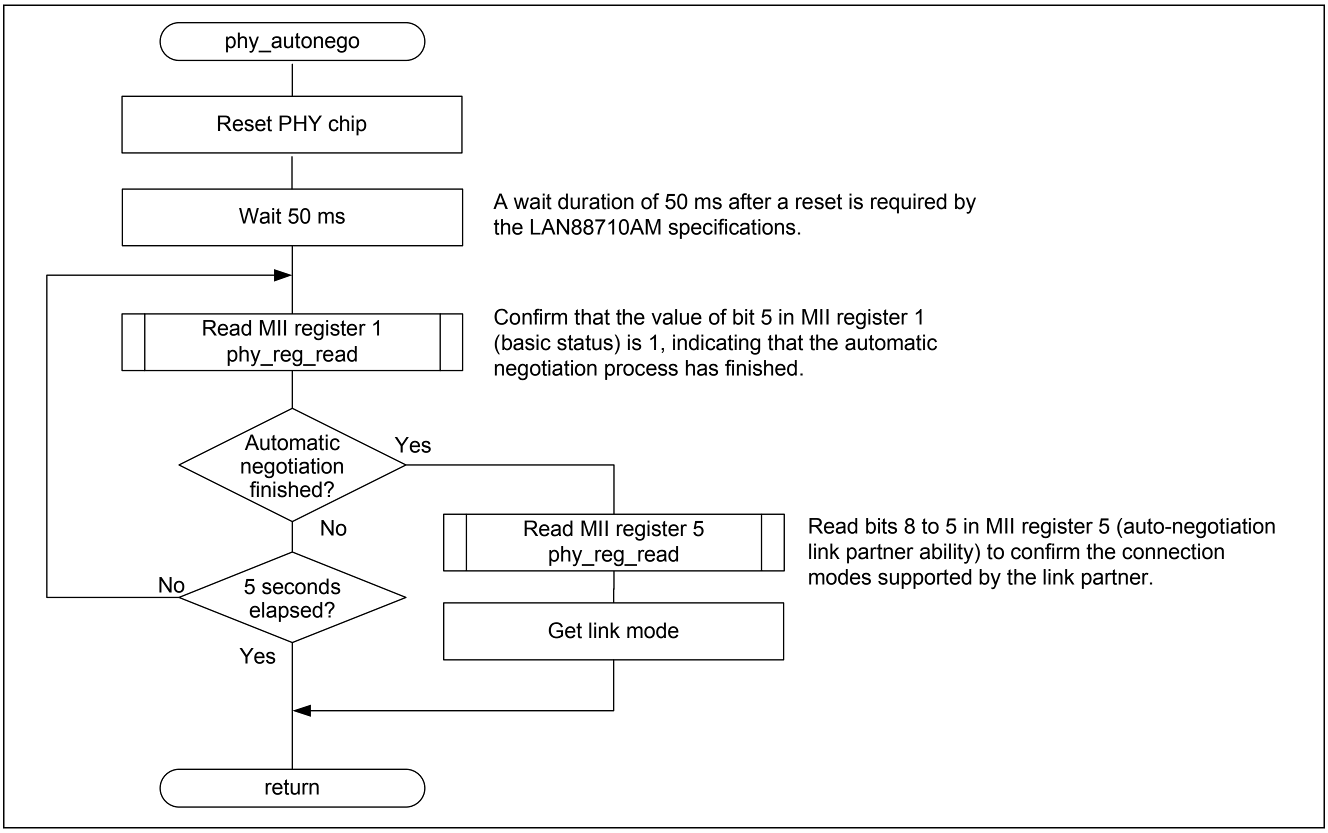

phy_autonego
Reset PHY chip
Wait 50 ms
A wait duration of 50 ms after a reset is required by the LAN88710AM specifications.
Read MII register 1
phy_reg_read
Confirm that the value of bit 5 in MII register 1 (basic status) is 1, indicating that the automatic negotiation process has finished.
Automatic negotiation finished?
Yes
No
Read MII register 5
phy_reg_read
Read bits 8 to 5 in MII register 5 (auto-negotiation link partner ability) to confirm the connection modes supported by the link partner.
No
5 seconds elapsed?
Yes
Get link mode
return

Figure 5.11 PHY-LSI Automatic Negotiation Result Acquisition Processing

5.8.10 Functions Related to PHY-LSI Automatic Negotiation Result Acquisition Processing

Figures 5.12 to 5.15 are flowcharts of the functions that perform PHY-LSI automatic negotiation result acquisition processing.

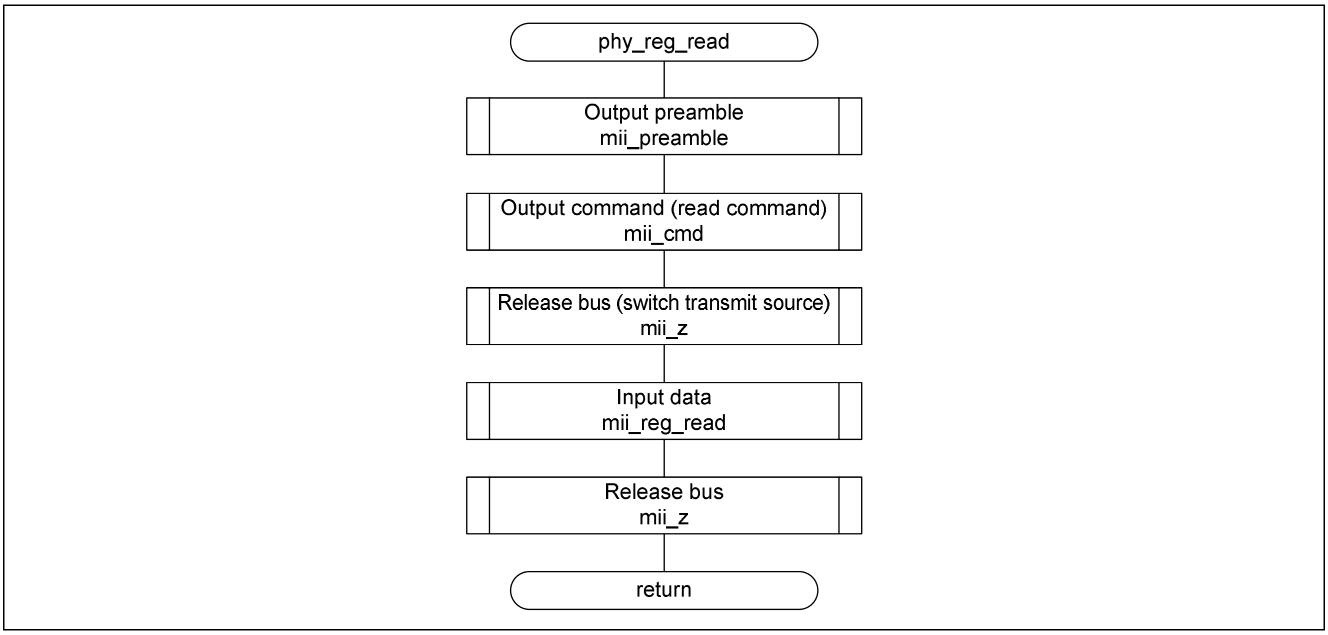


Figure 5.12 Functions Related to PHY-LSI Automatic Negotiation Result Acquisition Processing (1)

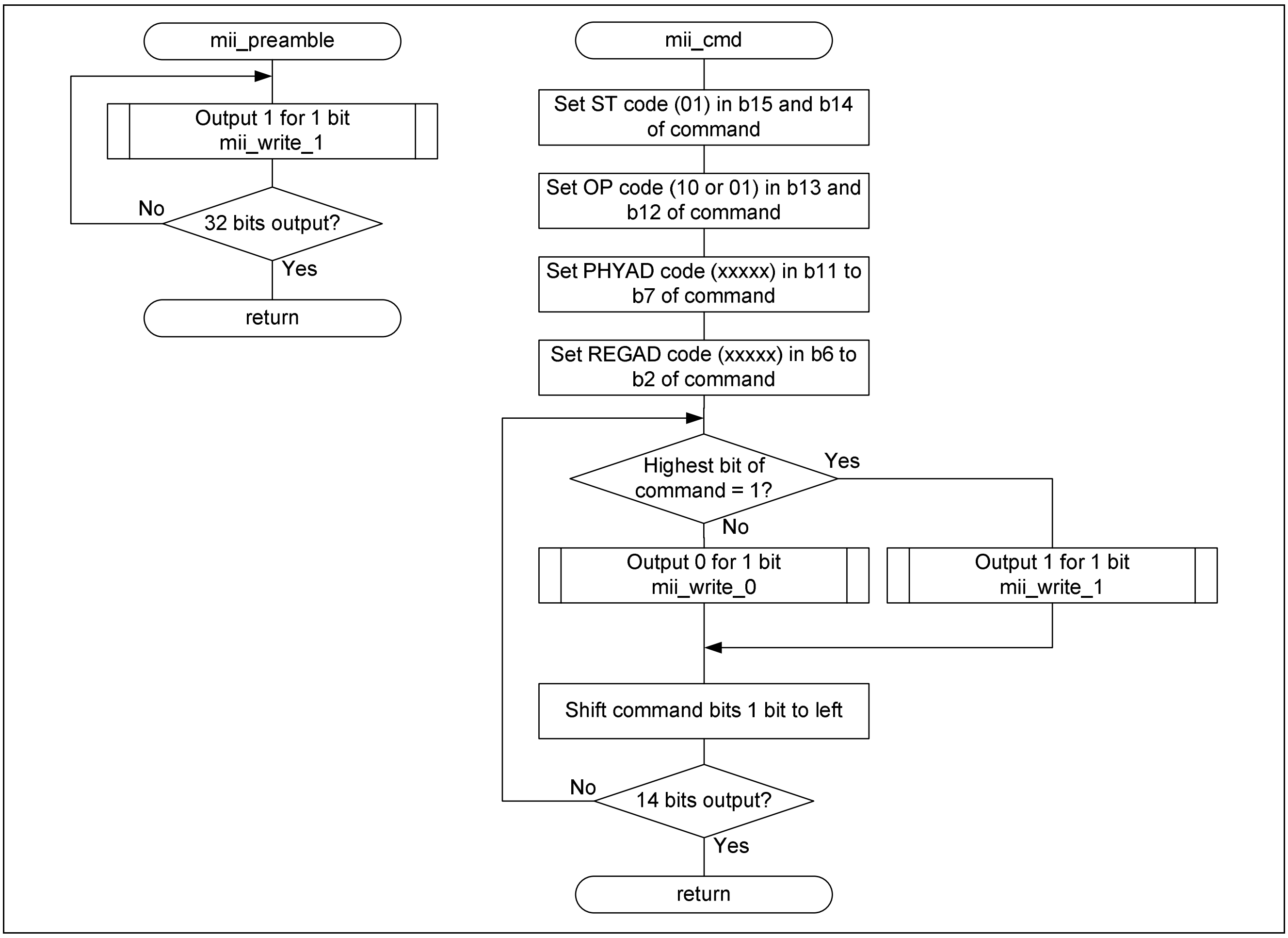

mii_preamble
Output 1 for 1 bit
mii_write_1
No
32 bits output?
Yes
return
mii_cmd
Set ST code (01) in b15 and b14 of command
Set OP code (10 or 01) in b13 and b12 of command
Set PHYAD code (xxxxx) in b11 to b7 of command
Set REGAD code (xxxxx) in b6 to b2 of command
Highest bit of command = 1?
Yes
No
Output 0 for 1 bit
mii_write_0
Output 1 for 1 bit
mii_write_1
Shift command bits 1 bit to left
No
14 bits output?
Yes
return

Figure 5.13 Functions Related to PHY-LSI Automatic Negotiation Result Acquisition Processing (2)

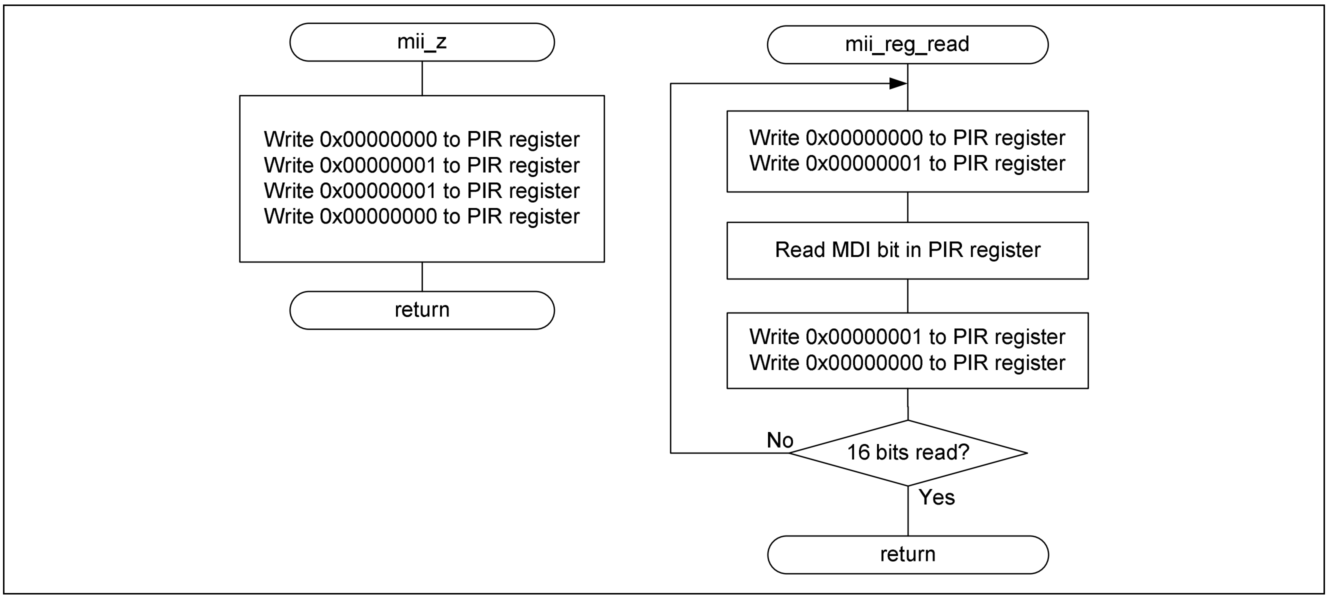

mii_z
Write 0x00000000 to PIR register
Write 0x00000001 to PIR register
Write 0x00000001 to PIR register
Write 0x00000000 to PIR register
return
mii_reg_read
Write 0x00000000 to PIR register
Write 0x00000001 to PIR register
Read MDI bit in PIR register
Write 0x00000001 to PIR register
Write 0x00000000 to PIR register
No
16 bits read?
Yes
return

Figure 5.14 Functions Related to PHY-LSI Automatic Negotiation Result Acquisition Processing (3)

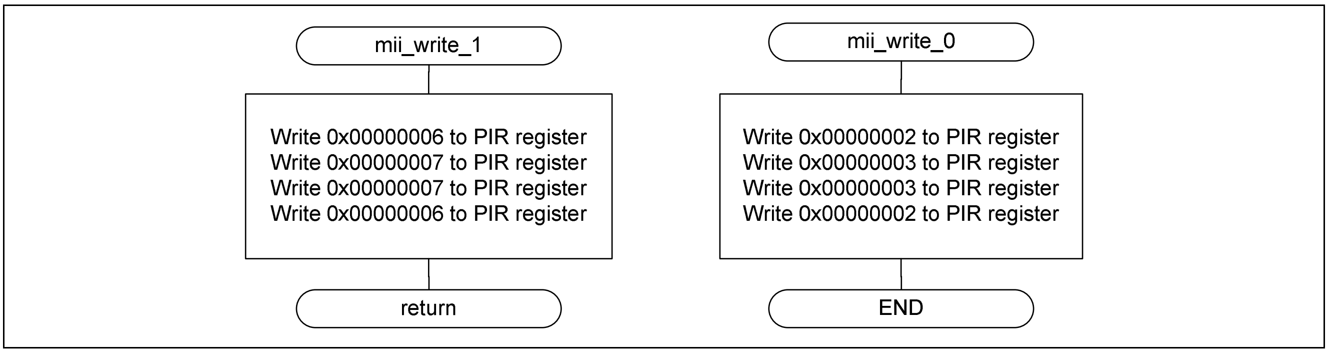

mii_write_1
Write 0x00000006 to PIR register
Write 0x00000007 to PIR register
Write 0x00000007 to PIR register
Write 0x00000006 to PIR register
return
mii_write_0
Write 0x00000002 to PIR register
Write 0x00000003 to PIR register
Write 0x00000003 to PIR register
Write 0x00000002 to PIR register
END

Figure 5.15 Functions Related to PHY-LSI Automatic Negotiation Result Acquisition Processing (4)

5.9 Section Assignments

Table 5.5 lists the assignments of the sections.

Table 5.5 Section Assignments

Section	Application	Area	Address assignment (virtual address)	
P	Program area (when not otherwise specified)	ROM	H'00003000	P0 area (cacheable, MMU address conversion supported)
C	Constant area	ROM		
P$PSEC	Section initialization program area	ROM		
C$BSEC	Address structure for uninitialized data	ROM		
C$DSEC	Address structure for initialized data	ROM		
D	Initialized data (initial value)	ROM		
B	Uninitialized data area	RAM	H'0C000000	
R	Initialized data area	RAM		
PRAM	Target area for copying program (P) from ROM	RAM		
S	Stack area	RAM	0x0FFFF9F0	
PINTHandler	Exception/interrupt handler	ROM	H'80000800	P1 area (cacheable, MMU address conversion not supported)
VECTTBL	Reset vector table	ROM		
INTTBL	Interrupt vector table Interrupt mask table	ROM		
PIntPRG	Interrupt handler	ROM		
SP_S	Dedicated stack area for TLB miss handler	RAM	H'8FFFFDF0	
RSTHandler	Reset handler	ROM	H'A0000000	P2 area (not cacheable, MMU address conversion not supported)
PResetPRG	Reset program	ROM		
P_LBSC_ROM	ROM program area (for LBSC)	ROM		
P_DBSC3_ROM	ROM program area (for DBSC3)	ROM		
PnonCache	Program area (cache-disabled access)	ROM		
BETH_DESC	Ethernet descriptor area	RAM	H'AD000000	
BETH_BUFF	Ethernet buffer area	RAM	H'AD001000	
INTTBL_OL	Interrupt mask table copy area	RAM	H'E500E000	OL memory
PINTHandler_IL	Exception/interrupt handler copy area	RAM	H'E5200000	IL memory
PIntPRG_IL	Interrupt handler copy area	RAM		
P_LBSC_IL	ROM program copy area (for LBSC)	RAM		

Note: For information on the reasons for providing special sections, section copying specifications, etc., see SH7734 Group: SH7734 Example of Initialization (R01AN0665EJ).

6. Sample Code

The sample code is available for download from the Renesas Electronics Web site.

7. Reference Documents

- SH7734 Group User’s Manual: Hardware, (R01UH0233EJ) Rev.1.00
  (The latest version can be downloaded from the Renesas Electronics Web site.)
- Technical Updates/Technical News
  (The latest information can be downloaded from the Renesas Electronics Web site.)
- Integrated Development Environment User’s Manual
  Super H C/C++ Compiler Package V.9.04 User’s Manual Rev.1.00
  (The latest version can be downloaded from the Renesas Electronics Web site.)

Website and Support

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

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

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Aug.30.12	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins
   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.
   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.
5. Differences between Products
   Before changing from one product to another, i.e. to one with a different type number, confirm that the change will not lead to problems.
   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

Notice

1. 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.
2. 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.
3. 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.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.
   "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 etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). 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 for which it is not intended. 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 which the product is not intended by Renesas Electronics.
6. 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.
7. 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 systems manufactured by you.
8. 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.
9. 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. You should not use Renesas Electronics products or 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. When exporting the Renesas Electronics 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.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
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.

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 2.2