SH7670 Group

Example of Setting for Automatic Negotiation by Ethernet PHY-LSI

R01AN0301EJ0101
Rev. 1.01
Oct. 15, 2010

Summary

This application note describes an example of settings for automatic negotiation with partners in communications when an Ethernet PHY-LSI chip has been connected to an SH7670, SH7671, SH7672 or SH7673.

Target Device

SH7670 MCU

Contents

1. Introduction

1.1 Specifications

- In this sample application, results of automatic negotiation when an Ethernet PHY-LSI chip has been connected to the SH7670 are acquired. The obtained connection mode (full-duplex and half-duplex modes) is specified as EtherC transfer method.
- An RTL8201 manufactured by Realtek Semiconductor Corp. is employed as the Ethernet PHY-LSI.
- Automatic negotiation function is used to establish the link with the Ethernet PHY-LSI.

1.2 Module Used

- Pin function controller (PFC)
- Ethernet controller (EtherC)

1.3 Applicable Conditions

MCU	SH7670
Operating Frequency	Internal clock: 200 MHz Bus clock: 66.6 MHz Peripheral clock: 33.3 MHz
Integrated Development Environment	Renesas Electronics High-performance Embedded Workshop Ver.4.03.00
C Compiler	Renesas Electronics SuperH RISC engine Family C/C++ compiler package Ver.9.01 Release 01
Compiler Options	Default setting in the High-performance Embedded Workshop (-cpu=sh2afpu -fpu=single -debug -gbr=auto -global_volatile=0 -opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1)

1.4 Related Application Notes

For more information, refer to the following application notes:

- SH7670 Group Example of Initialization
- SH7670 Group Example of Setting for Transmission of Ethernet Frames
- SH7670 Group Example of Setting for Reception of Ethernet Frames

2. Description of the Sample Application

This sample application employs an Ethernet PHY-LSI chip to perform automatic negotiation with a partner in communications. The result of automatic negotiation is read out via the PHY interface register (PIR) of the EtherC module.

2.1 Operational Overview of Module Used

Link processing on the physical layer is one function of an Ethernet PHY-LSI chip. The on-chip EtherC module of an SH7670 MCU can obtain the result of link processing by reading values from the Ethernet PHY chip. In this sample application, the PHY-LSI pins are set up to enable the function of automatic negotiation. For details on the functions of the Ethernet PHY-LSI, see the datasheet for the product.

The interface between the EtherC module and the Ethernet PHY-LSI is a Media Independence Interface (MII) compliant with IEEE802.3. Figure 1 shows an example of the connections between an SH7670 MCU and RTL8201CP.

The results of automatic negotiation are stored in the Ethernet PHY-LSI register, and are read out by using the serial interface between the MDC and MDIO pins. The SH7670 MCU can read or write both pins by using the PIR register. For the procedure to access to the registers of the PHY-LSI, see the next section 2.2, Procedure for Access to the MII Registers.

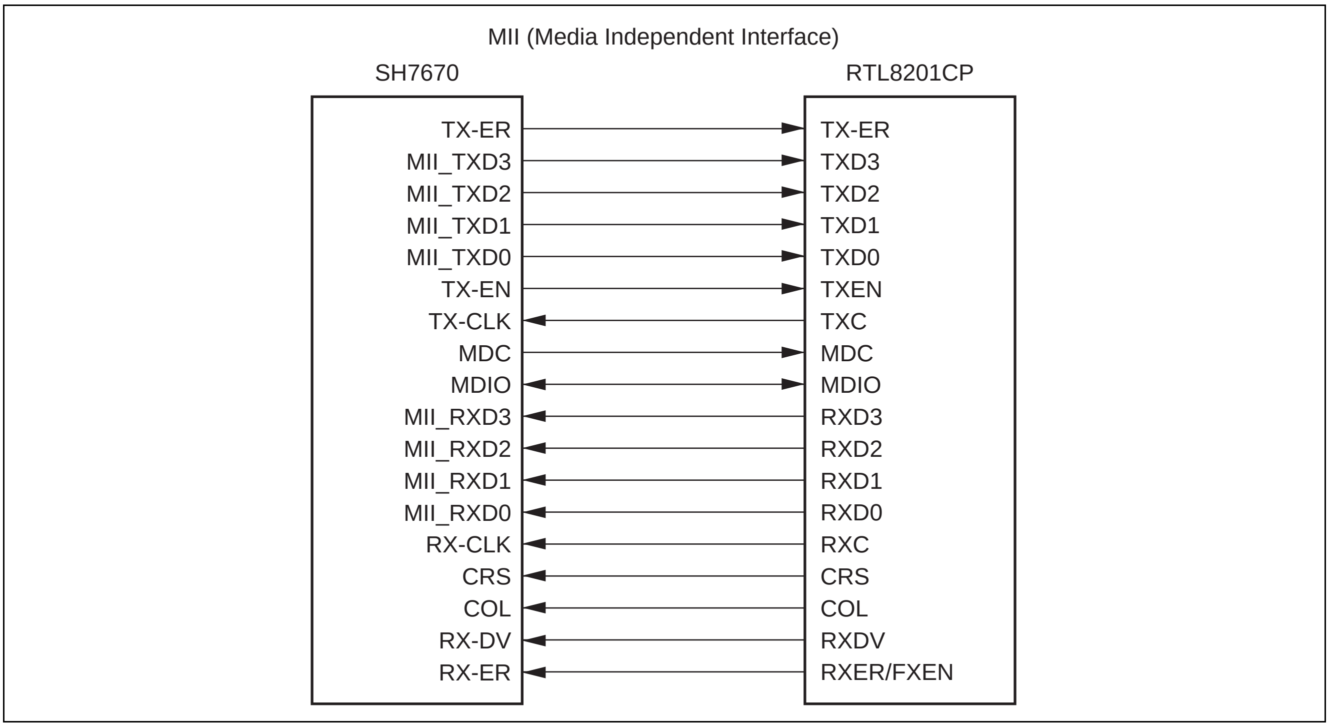


Figure 1 Example of Connecting an MCU to the RTL8201CP

2.2 Procedure for Access to the MII registers

This section describes the procedure for access to the MII registers of the Ethernet PHY-LSI register.

The MII operates through two pins, MDC and MDIO (both are the pin names on the EtherC side). The MDC pin is for the synchronizing clock signal, while data are input and output through the MDIO pin. The state of each pin can be referred or changed by the PIR register of EtherC. The MII must output data as specified in the standard format (the MII management frame) as it has no control pins. Figure 2 shows the MII management frame. In this sample program, Z is output over one bit period in the idle state. This is required because, while there is no reference to clock input in the IEEE802.3 standard, correct connection with some PHY-LSI circuits is not otherwise possible. This precautionary operation avoids such situations.

An MII management frame is input or output in one-bit units, in order from the preamble. Figures 3 to 5 depict the flow of input and output of the one-bit units. The timing of input and output on the MDC and MDIO pins must conform with the IEEE802.3 standard. Table 1 and Figure 6 show the standard's specifications for the timing of input and output.

Access Type	MII mangement frame							
Item	PRE	ST	OP	PHYAD	REGAD	TA	DATA	IDLE
Number of bits	32	2	2	5	5	2	16	1
Read	1..1	01	10	AAAAA	RRRRR	Z0	D..D	Z
Write	1..1	01	01	AAAAA	RRRRR	10	D..D	Z

[Legend]

PRE (preamble): 32 consecutive 1s are output for synchronization.

ST (start of frame): 01 is output to indicate the start of the frame.

OP (operation code): Output to specify reading or writing; 10 indicates reading, 01 indicates writing.

PHYAD (PHY address): The address to distinguish one amoung multiple PHY-LSIs.
The address is often specified by the levels on PHY-LSI pins.
This is output from the MSB.

REGAD (Register address): Specifies the number of MII registers. This is output from the MSB.

TA (turn around): Switches the transmission source of the MDIO pin to avoid collision of signals.
(a) Reading: The bus is released over one bit period (Z output).
As 0 is output from the PHY module, the notation is "Z0".
(b) Writing: 10 is output.

DATA (data): 16-bit value read from or to be written to the register. Writing or reading proceeds in order from the MSB.

IDLE (IDLE condition): Wait for further input in the MII management format. The bus is released (Z output).

Figure 2 MII Management Frame Format

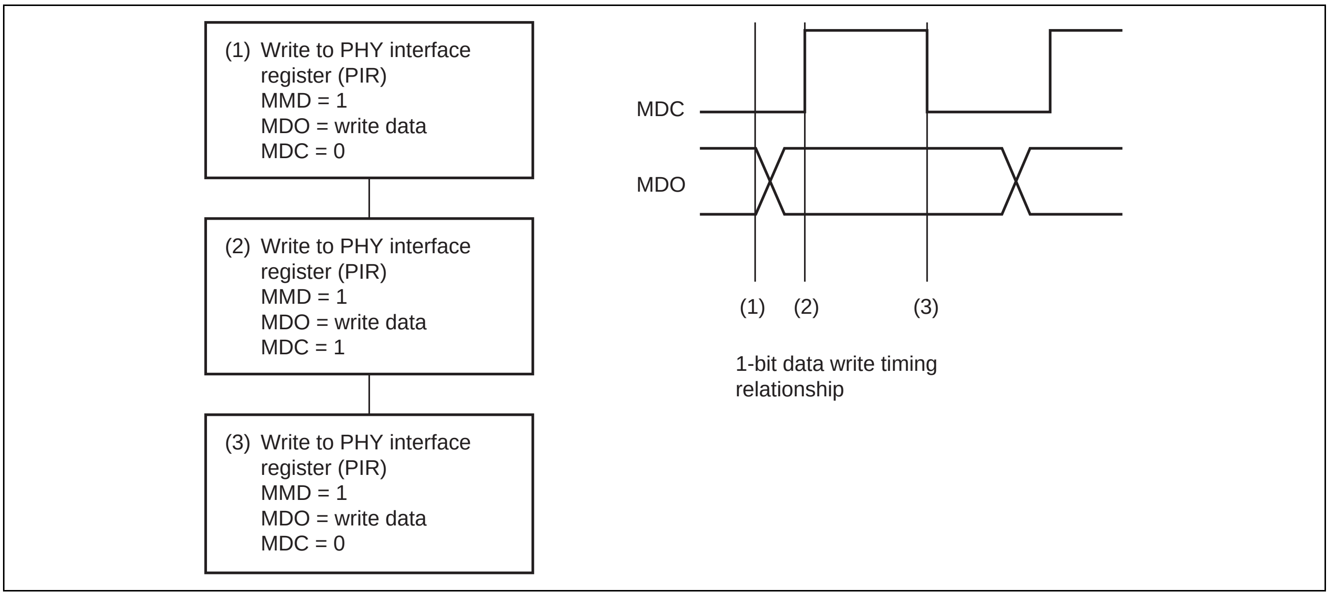

(1) Write to PHY interface register (PIR)
MMD = 1
MDO = write data
MDC = 0
(2) Write to PHY interface register (PIR)
MMD = 1
MDO = write data
MDC = 1
(3) Write to PHY interface register (PIR)
MMD = 1
MDO = write data
MDC = 0
MDC
MDO
(1)
(2)
(3)
1-bit data write timing relationship

Figure 3 Flow for Writing of One Bit

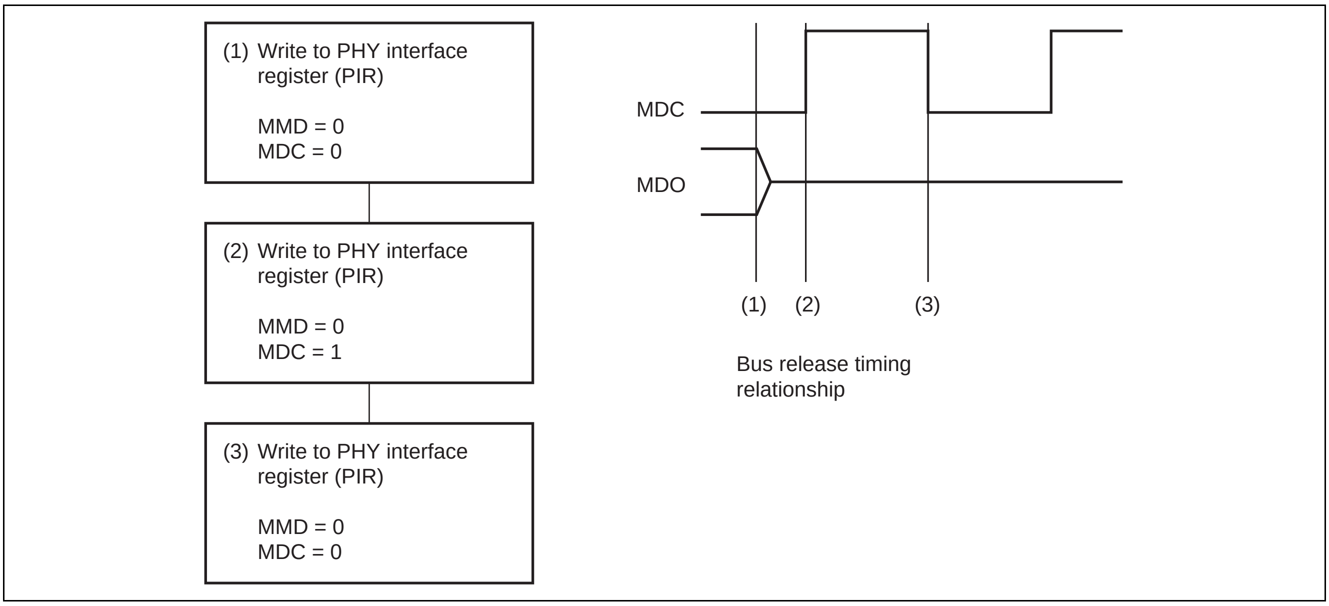

(1) Write to PHY interface register (PIR)
MMD = 0
MDC = 0
(2) Write to PHY interface register (PIR)
MMD = 0
MDC = 1
(3) Write to PHY interface register (PIR)
MMD = 0
MDC = 0
MDC
MDO
(1)
(2)
(3)
Bus release timing relationship

Figure 4 Flow of Bus Release (Z-output)

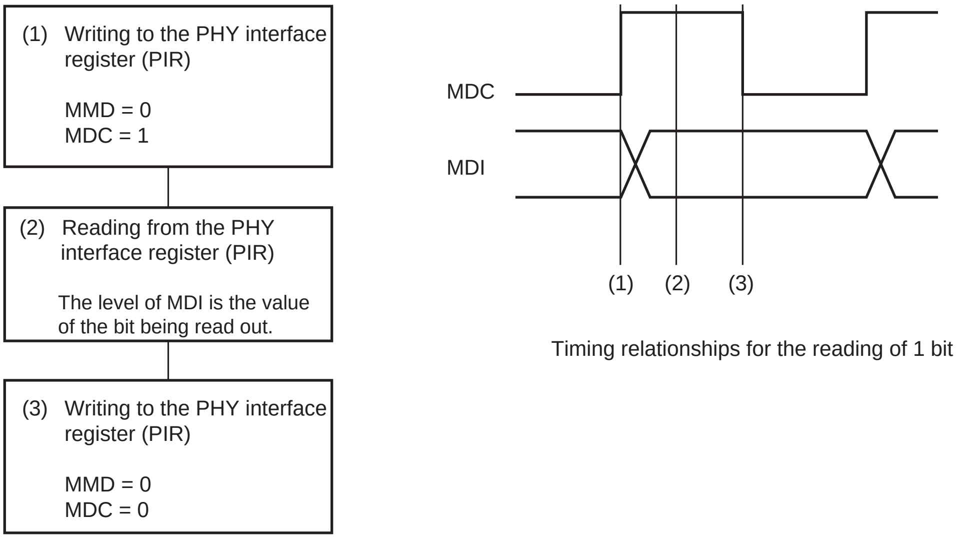

(1) Writing to the PHY interface register (PIR)
MMD = 0
MDC = 1
(2) Reading from the PHY interface register (PIR)
The level of MDI is the value of the bit being read out.
(3) Writing to the PHY interface register (PIR)
MMD = 0
MDC = 0
MDC
MDI
(1)
(2)
(3)
Timing relationships for the reading of 1 bit

Figure 5 Flow for Reading of One Bit

Table 1 MDC/MDIO Input/Output Timing

Item	Symbol	Min	Max	Unit
MDC high-level pulse width	t_1	160		ns
MDC low-level pulse width	t_2	160		ns
MDC cycle time	t_3	400		ns
MDIO setup time	t_4	10		ns
MDIO hold time	t_5	10		ns
MDIO output delay time	t_6	0	300	ns

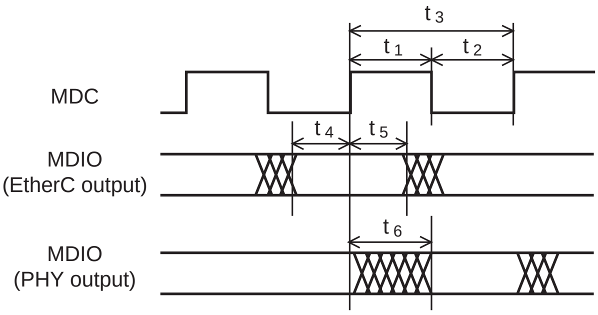


Figure 6 MDC/MDIO Input/Output Timing

2.3 Description of Settings in the Sample Program

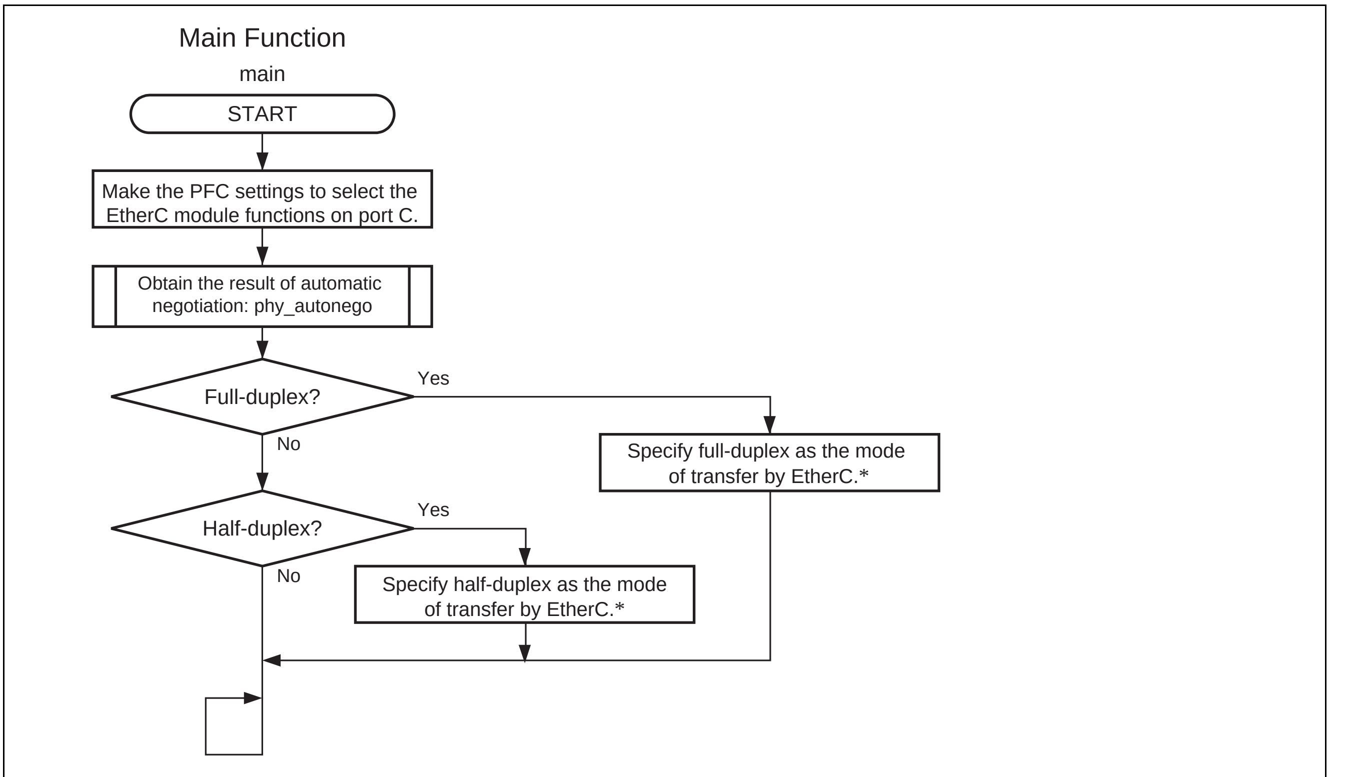
This sample program is in two files of source code, main.c and phy.c, and the files for initialization created in the application note “Example of Initialization” for the SH7670 (REJ06B0799).

- main.c

This contains the definition of the main function, and obtains the result of automatic negotiation. Figure 7 shows the flow of processing by the main function.

- phy.c

This contains the definition of the function for obtaining the result of automatic negotiation (function phy_autonego). Figure 8 shows the flow of processing by the phy_autonego function. Figures 9 to 12 show the flows of processing by the MII register-read function (function phy_reg_read) that is called by the phy_autonego function, and of functions lower in the hierarchy of calls.


Main Function
main
START
Make the PFC settings to select the EtherC module functions on port C.
Obtain the result of automatic negotiation: phy_autonego
Full-duplex?
Yes
No
Specify full-duplex as the mode of transfer by EtherC.*
Half-duplex?
Yes
No
Specify half-duplex as the mode of transfer by EtherC.*

Note: Specification of full- or half-duplex as the mode of transfer by the EtherC module is made in the DM bit of the EtherC mode register (ECMR).

Figure 7 Flow of Processing by the main Function

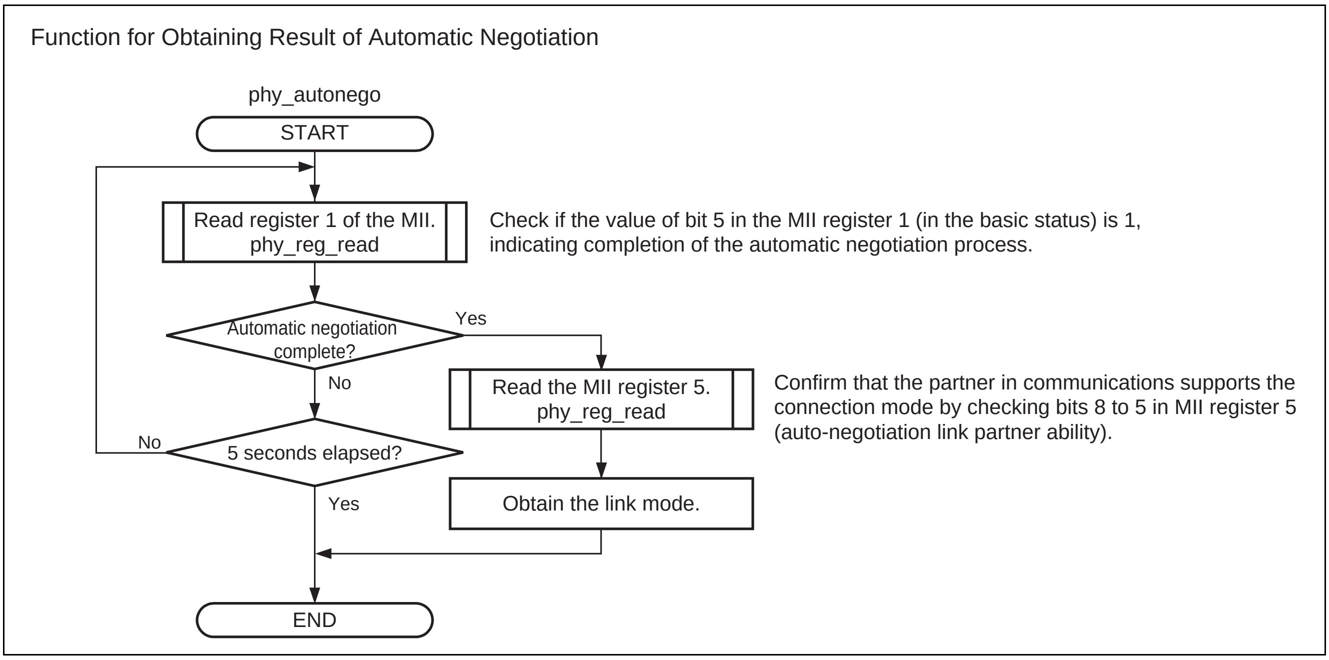

Function for Obtaining Result of Automatic Negotiation
phy_autonego
START
Read register 1 of the MII.
phy_reg_read
Check if the value of bit 5 in the MII register 1 (in the basic status) is 1, indicating completion of the automatic negotiation process.
Automatic negotiation complete?
Yes
No
Read the MII register 5.
phy_reg_read
Confirm that the partner in communications supports the connection mode by checking bits 8 to 5 in MII register 5 (auto-negotiation link partner ability).
No
5 seconds elapsed?
Yes
Obtain the link mode.
END

Figure 8 Flow of Processing by the Function for Obtaining the Result of Automatic Negotiation

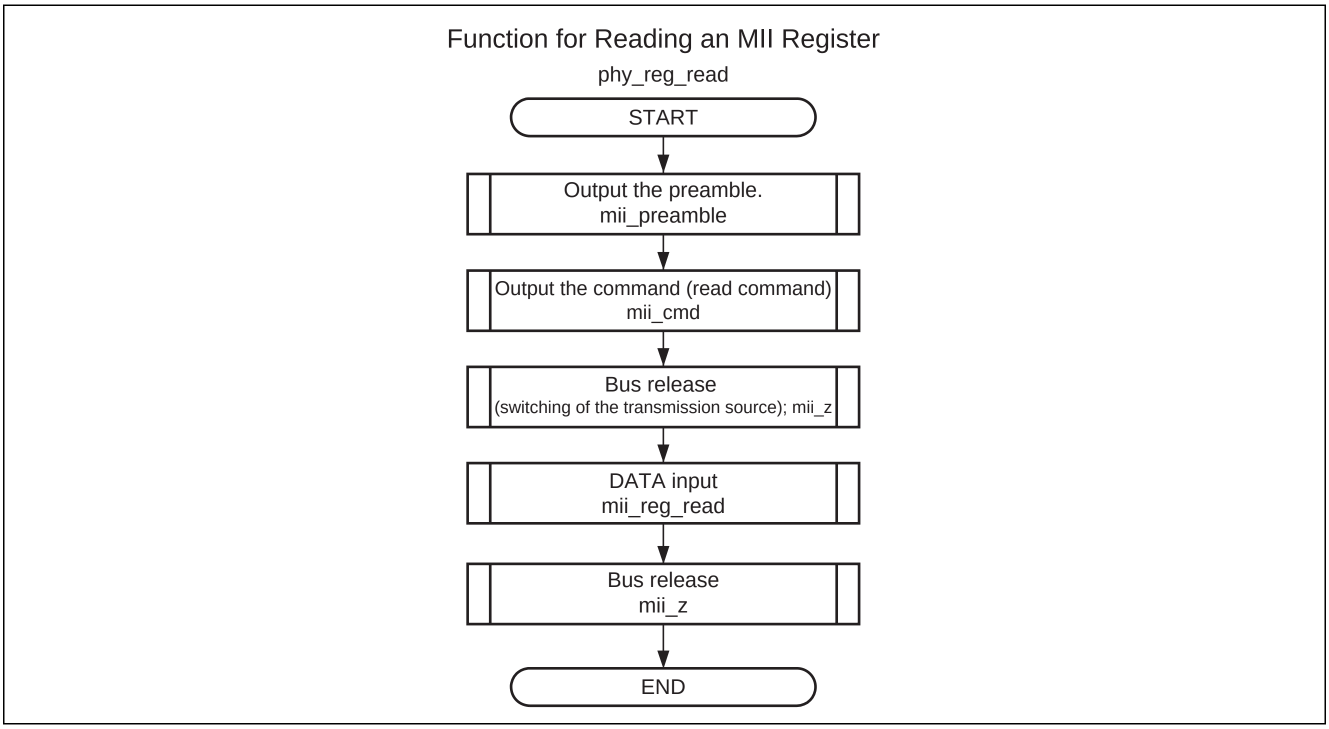

Function for Reading an MII Register
phy_reg_read
START
Output the preamble.
mii_preamble
Output the command (read command)
mii_cmd
Bus release
(switching of the transmission source); mii_z
DATA input
mii_reg_read
Bus release
mii_z
END

Figure 9 Flow of Processing to Access the MII Registers (1)

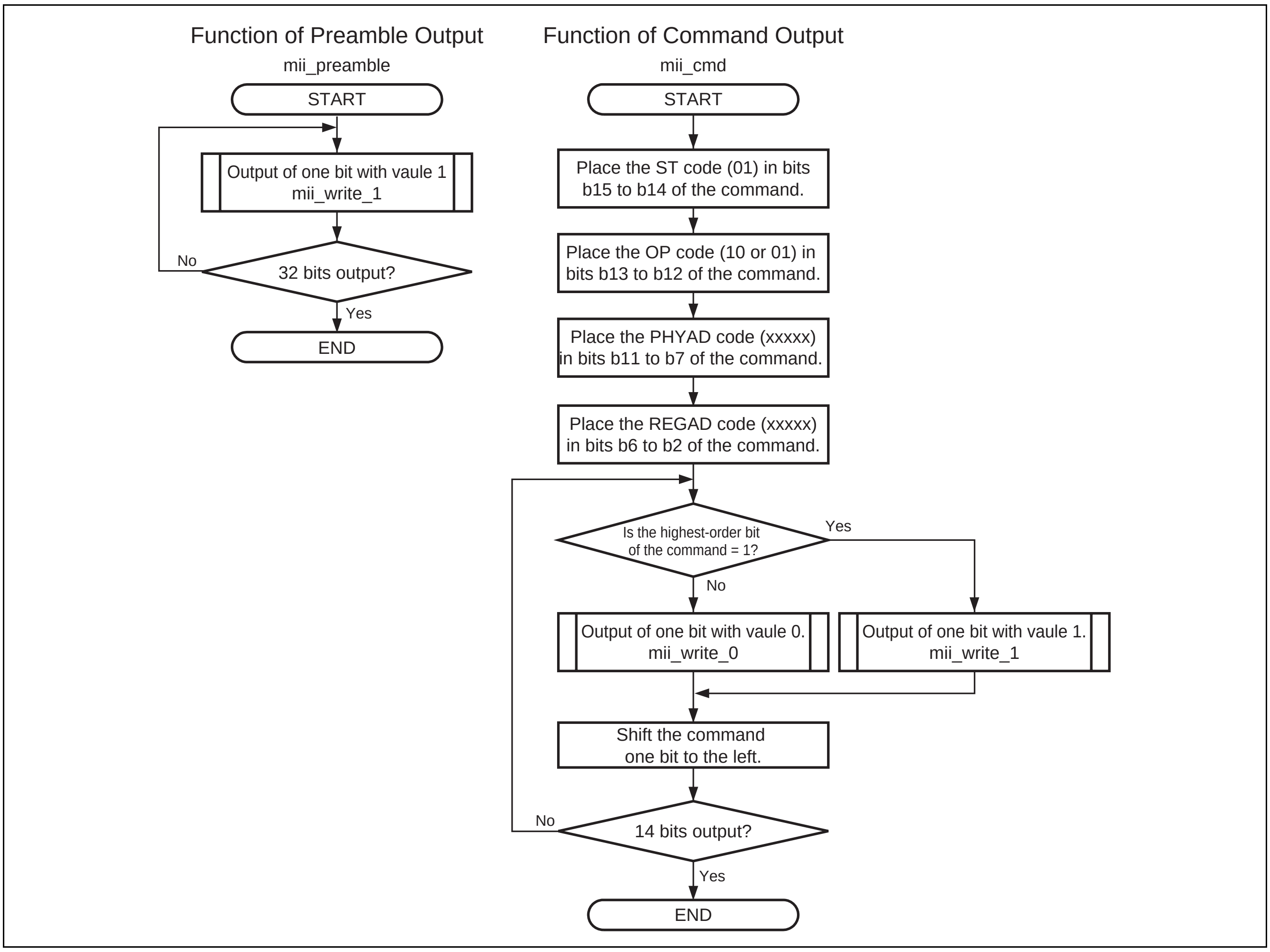
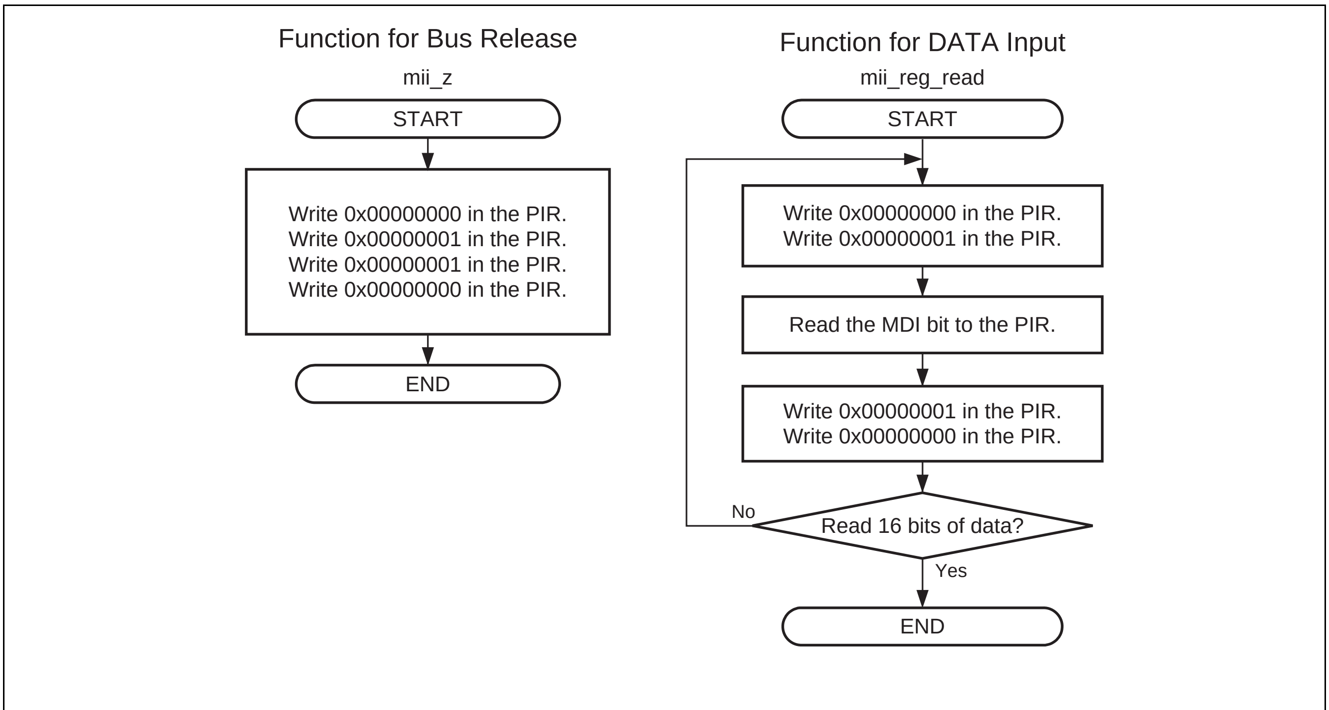
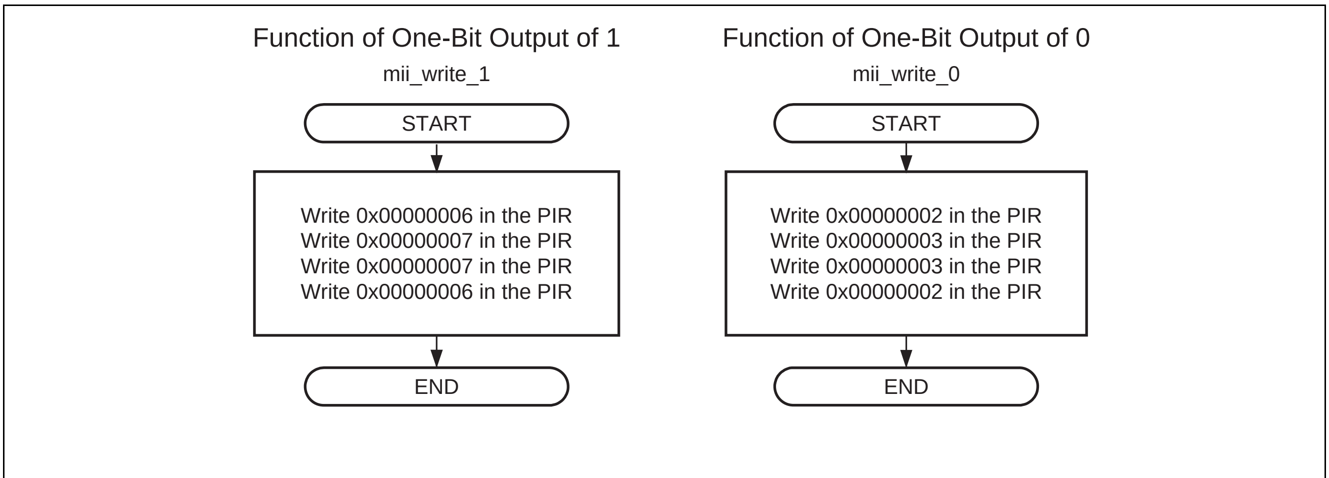

Function of Preamble Output
mii_preamble
START
Output of one bit with vaule 1
mii_write_1
32 bits output?
No
Yes
END
Function of Command Output
mii_cmd
START
Place the ST code (01) in bits b15 to b14 of the command.
Place the OP code (10 or 01) in bits b13 to b12 of the command.
Place the PHYAD code (xxxxx) in bits b11 to b7 of the command.
Place the REGAD code (xxxxx) in bits b6 to b2 of the command.
Is the highest-order bit of the command = 1?
Yes
No
Output of one bit with vaule 0.
mii_write_0
Output of one bit with vaule 1.
mii_write_1
Shift the command one bit to the left.
14 bits output?
No
Yes
END

Figure 10 Flow of Processing to Access the MII Registers (2)


Function for Bus Release
mii_z
START
Write 0x00000000 in the PIR.
Write 0x00000001 in the PIR.
Write 0x00000001 in the PIR.
Write 0x00000000 in the PIR.
END
Function for DATA Input
mii_reg_read
START
Write 0x00000000 in the PIR.
Write 0x00000001 in the PIR.
Read the MDI bit to the PIR.
Write 0x00000001 in the PIR.
Write 0x00000000 in the PIR.
No
Read 16 bits of data?
Yes
END

Note: The above processing is executed to meet the requirements for the timing of input and output through the MDC and MDIO pins.

Figure 11 Flow of Processing to Access the MII Registers (3)


Function of One-Bit Output of 1
mii_write_1
START
Write 0x00000006 in the PIR
Write 0x00000007 in the PIR
Write 0x00000007 in the PIR
Write 0x00000006 in the PIR
END
Function of One-Bit Output of 0
mii_write_0
START
Write 0x00000002 in the PIR
Write 0x00000003 in the PIR
Write 0x00000003 in the PIR
Write 0x00000002 in the PIR
END

Note: The processing is executed to meet the input/output timing of the MDC/MDIO pins.

Figure 12 Flow of Processing to Access the MII Registers (4)

2.4 Description of Settings in the Sample Program

Table 2 is a list of the settings in the sample program.

Table 2 Settings in the Sample Program

Item	Description
PHY used	RTL8201CP manufactured by Realtek Semiconductor Corp.
Link mode	100 Mbps (full- and half-duplex modes) and 10 Mbps (full- and half-duplex modes)
Link determination method	Automatic negotiation (auto-negotiation)
MII registers used	Basic mode register (address: 01h) Auto-negotiation link partner ability (address: 05h)

2.5 Notes on Using the Sample Program

- The automatic negotiation mode is assumed to be the method by which linking of the PHY chip is detected in this sample program.
- When the partner is also in the automatic negotiation mode, the link is established in accord with the order of priority shown in table 3.
- The completion of automatic negotiation usually takes a couple of seconds, but the phy_autonego function allows up to 5 seconds for the completion of negotiations in the sample program.

Table 3 Link Type Priorities

Order of Priority		Link
High	1	Full-duplex mode, 100 Mbps
	2	Half-duplex mode, 100 Mbps
	3	Full-duplex mode, 10 Mbps
Low	4	Half-duplex mode, 10 Mbps

3. Sample Program Listing

3.1 Sample program list "main.c" (1)

```
/******************************************************************************
*   DISCLAIMER
*
*   This software is supplied by Renesas Electronics Corporation and is only
*   intended for use with Renesas products. No other uses are authorized.
*
*   This software is owned by Renesas Electronics Corporation and is protected under
*   all applicable laws, including copyright laws.
*
*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
*   DISCLAIMED.
*
*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
*   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
*
*   Renesas reserves the right, without notice, to make changes to this
*   software and to discontinue the availability of this software.
*   By using this software, you agree to the additional terms and
*   conditions found by accessing the following link:
*   http://www.renesas.com/disclaimer
********************************************************************************
* Copyright (C) 2007(2010) Renesas Electronics Corporation. All rights reserved.
*""FILE COMMENT""*********** Technical reference data **************************
*   System Name : SH7671 Sample Program
*   File Name   : main.c
*   Abstract    : Acquiring Ethernet PHY automatic negotiation results
*   Version     : 1.01.01
*   Device      : SH7671
*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.03.00).
*               : C/C++ compiler package for the SuperH RISC engine family
*               :                             (Ver.9.01 Release01).
*   OS          : None
*   H/W Platform: M3A-HS71(CPU board)
*   Description :
********************************************************************************
*   History     : Jul.04,2007 ver.1.00.00
*               : Sep.18,2007 ver.1.01.00 support for revised board DK30686-A
*               : May 12,2010 ver.1.01.01 Changed the company name and device name
*""FILE COMMENT END""**********************************************************/
#include "iodefine.h"
#include "defs.h"
#include "phy.h"

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

3.2 Sample program list "main.c" (2)

```

/*""FUNC COMMENT""**************************************************************
 * ID          :
 * Outline     : Main function of acquiring PHY-LSI automatic negotiation results
 *------------------------------------------------------------------------------
 * Include     : #include "iodefine.h"
 *------------------------------------------------------------------------------
 * Declaration : void main(void)
 *------------------------------------------------------------------------------
 * Function    : PHY link mode is determined by PHY-LSI automatic negotiation function.
 *             : This sample reads out the results.
 *             : Normally the automatic negotiation is compleded in 1200 ms, and
 *             : uatomatic negotiation acquisition function checks negotiation completion for 5 s max.
 *             : The results of full-duplex or half-duplex is set in EtherC.
 *------------------------------------------------------------------------------
 * Argument    : void
 *------------------------------------------------------------------------------
 * ReturnValue : void
 *------------------------------------------------------------------------------
 * Notice      :
 *""FUNC COMMENT END""**********************************************************/
void main(void)
{
  int link;

  /* ==== PFC setting ==== */
//PORT.PBCRL1.BIT.PB6MD = 1;      /* Setting when DK30686 board is used */
  PORT.PCCRH1.WORD = 0x0155;      /* EtherC function  */
  PORT.PCCRL1.WORD = 0x5555;      /* PHY-LSI operates independently; this is for PHY register access */
  PORT.PCCRL2.WORD = 0x5555;      /* Necessary for using MII control interface  */

  /* ==== Acquire PHY negotiation results ==== */
  link=phy_autonego();

  /* ==== EtherC duplex mode setting ==== */
  if( link == FULL_TX || link == FULL_10M ){
    EtherC.ECMR.BIT.DM = 1;                   /* Full duplex */
  }
  else if(link == HALF_TX || link == HALF_10M){
    EtherC.ECMR.BIT.DM = 0;                   /* Half duplex */
  }
  else{
    ;/* Link failed */
  }

  /* ==== Main loop ==== */
  while(1){
    ;
  }
}

/* End of file */
```

3.3 Sample program list "phy.c" (1)

```
/******************************************************************************
*   DISCLAIMER
*
*   This software is supplied by Renesas Electronics Corporation and is only
*   intended for use with Renesas products. No other uses are authorized.
*
*   This software is owned by Renesas Electronics Corporation and is protected under
*   all applicable laws, including copyright laws.
*
*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
*   DISCLAIMED.
*
*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
*   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
*
*   Renesas reserves the right, without notice, to make changes to this
*   software and to discontinue the availability of this software.
*   By using this software, you agree to the additional terms and
*   conditions found by accessing the following link:
*   http://www.renesas.com/disclaimer
********************************************************************************
* Copyright (C) 2007(2010) Renesas Electronics Corporation. All rights reserved.
*""FILE COMMENT""*********** Technical reference data **************************
*   System Name : SH7671 Sample Program
*   File Name   : phy.c
*   Abstract    : Obtaining the result of automatic negotiation by the Ethernet PHY
*   Version     : 1.00.01
*   Device      : SH7671
*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.03.00).
*               : C/C++ compiler package for the SuperH RISC engine family
*               :                             (Ver.9.01 Release01).
*   OS          : None
*   H/W Platform: M3A-HS71(CPU board)
*   Description :
********************************************************************************
*   History     : Jul.13,2007 ver.1.00.00
*               : May 12,2010 ver.1.00.01 Changed the company name and device name
*""FILE COMMENT END""**********************************************************/
#include "iodefine.h"
#include "defs.h"
#include "phy.h"

/* **** Prototype declaration **** */
static unsigned short phy_reg_read (unsigned short reg_addr);
static void phy_reg_write (unsigned short reg_addr, unsigned short data);
static void mii_preamble (void);
static void mii_cmd (unsigned short reg_addr, int option);
static void mii_ta10 (void);
```

3.4 Sample program list "phy.c" (2)

```
static void mii_reg_read (unsigned short *data);
static void mii_reg_write (unsigned short data);
static void mii_write_1 (void);
static void mii_write_0 (void);
static void mii_z (void);


/* **** Macro definition **** */
/* PHY register */
#define BASIC_MODE_CONTROL_REG 0
#define BASIC_MODE_STATUS_REG 1
#define PHY_IDENTIFIER1_REG 2
#define PHY_IDENTIFIER2_REG 3
#define AN_ADVERTISEMENT_REG 4
#define AN_LINK_PARTNER_ABILITY_REG 5
#define AN_EXPANSION_REG 6
/* PHY address */
#define PHY_ADDR 1 /* Confirm the pin connection of the PHY-LSI */
/* For accessing the MII management interface */
#define PHY_ST 1
#define PHY_WRITE 1
#define PHY_READ 2
#define MDC_WAIT 3 /* 400 ns/4 < (Pφ*2)*MDC_WAIT */


/*""FUNC COMMENT""**************************************************************
 * ID             :
 * Outline        : Detection of the result of automatic negotiation of the PHY link
 *------------------------------------------------------------------------------
 * Include        : "iodefine.h", "defs.h"
 *------------------------------------------------------------------------------
 * Declaration    : int phy_autonego(void);
 *------------------------------------------------------------------------------
 * Description    : The result of automatic negotiation is read out by using the MII
 *                : management interface, and indicated in the return value.
 *                : The PHY chip used with this sample program supports all of the listed
 *                : transfer modes, including full-duplex connection at 100 Mbps.
 *                : If the target of the link supports automatic negotiation,
 *                : it chooses its top-performance link mode for connection.
 *                : If it does not, the parallel detection function detects the link speed;
 *                : in that case, half-duplex mode is specified. Although automatic
 *                : negotiation is usually completed in about 1,200 ms,
 *                : this function allows up to 5 seconds for completion of the negotiation.
 *------------------------------------------------------------------------------
 * Argument       : void
 *------------------------------------------------------------------------------
 * Return Value   : 4(FULL_TX) :100 Mbps in full-duplex mode
 *                : 3(HALF_TX) :100 Mbps in half-duplex mode
 *                : 2(FULL_10M) :10 Mbps in full-duplex mode
 *                : 1(HALF_10M) :10 Mbps in half-duplex mode
 *                : 0(NEGO_FAIL) :Failure in negotiation
 *------------------------------------------------------------------------------
 * Note          : The parallel detection function is not considered in this sample program.
 *""FUNC COMMENT END""**********************************************************/
```

3.5 Sample program list "phy.c" (3)

```

int phy_autonego (void)
{
  unsigned short data;
  int link = NEGO_FAIL;
  volatile int t;
  int i;

  /*==== Loop to wait for completion of automatic negotiation (for up to 5 seconds)==== */
  for(i=0; i<500; i++){
    /* ==== Wait for 10 ms to set up a benchmark for measurement of elapsed-time ==== */
    t = LOOP_100us*100;
    while( --t){
        ;
    }
    /* ==== Confirm completion of automatic negotiation==== */
    data = phy_reg_read(BASIC_MODE_STATUS_REG);
    if( data & 0x0020 ){   /* PHY register 1 :Basic status *
                            * bit 5 :(1) completion of automatic negotiation process *
                            *       :(0) non-completion of automatic negotiation process */
    /* ---- The capability of the partner as determined
    on completion of automatic negotiation. ---- */
    data = phy_reg_read(AN_LINK_PARTNER_ABILITY_REG);
                           /* PHY register 5 :Auto-nego link partner ability *
                            * bit 8 :(1)100 Mbps/full-duplex mode is possible *
                            * bit 7 :(1)100 Mbps/half-duplex mode is possible *
                            * bit 6 :(1) 10 Mbps/full-duplex mode is possible *
                            * bit 5 :(1) 10 Mbps/half-duplex mode is possible */
    /* ---- Break on completion of negotiation following detection of results ---- */
    if( data&0x0100 ){
        link = FULL_TX;
    }
    else if (data&0x0080){
        link = HALF_TX;
    }
    else if (data&0x0040){
        link = FULL_10M;
    }
    else if (data&0x0020){
        link = HALF_10M;
    }
    else{
        link = NEGO_FAIL;
    }
    /* ---- Processing to detect the result of automatic negotiation is complete. ---- */
    break;
    }
  }
  return link;
}
```

3.6 Sample program list "phy.c" (4)

```
/*""FUNC COMMENT""**************************************************************
 * ID            :
 * Outline       : Reading of a PHY module register
 *------------------------------------------------------------------------------
 * Include       :
 *------------------------------------------------------------------------------
 * Declaration   : static unsigned short phy_reg_read (unsigned short reg_addr);
 *------------------------------------------------------------------------------
 * Description   : Obtains the value from a register of the PHY module.
 *------------------------------------------------------------------------------
 * Argument      : unsigned short reg_addr ; I : Address of the PHY register
 *               :                               from which value is to be read.
 *------------------------------------------------------------------------------
 * Return Value  : The obtained register value
 *------------------------------------------------------------------------------
 * Note          :
 *""FUNC COMMENT END""**********************************************************/
static unsigned short phy_reg_read (unsigned short reg_addr)
{
  unsigned short data;

  mii_preamble();
  mii_cmd (reg_addr, PHY_READ);
  mii_z();
  mii_reg_read (&data);
  mii_z();

  return data;
}
```

3.7 Sample program list "phy.c" (5)

```
/*""FUNC COMMENT""**************************************************************
 * ID             :
 * Outline        : Writing to a PHY module register
 *-------------------------------------------------------------------------------
 * Include        :
 *-------------------------------------------------------------------------------
 * Declaration    : static void phy_reg_write (unsigned short reg_addr,
 *                :                                          unsigned short data);
 *-------------------------------------------------------------------------------
 * Description    : Set the value in the PHY module register
 *-------------------------------------------------------------------------------
 * Argument       : unsigned short reg_addr ; I : Address of the PHY register to which
 *                :                              the value is to be written
 *                : unsigned short data ; I : Value to be set in the PHY register
 *-------------------------------------------------------------------------------
 * Return Value   : void
 *-------------------------------------------------------------------------------
 * Note           :
 *""FUNC COMMENT END""***********************************************************/static
void phy_reg_write (unsigned short reg_addr, unsigned short data)
{
  mii_preamble();
  mii_cmd (reg_addr, PHY_WRITE);
  mii_ta10();
  mii_reg_write (data);
  mii_z();

}
```

3.8 Sample program list "phy.c" (6)

```
/*""FUNC COMMENT""**************************************************************
 * ID              :
 * Outline         : Preparation to access a PHY module register
 *-----------------------------------------------------------------------------
 * Include         :
 *-----------------------------------------------------------------------------
 * Declaration     : static void mii_preamble (void);
 *-----------------------------------------------------------------------------
 * Description     : As preliminary preparation for access to the PHY module register,
 *                 : 1 is output via the MII management interface.
 *-----------------------------------------------------------------------------
 * Argument        : void
 *-----------------------------------------------------------------------------
 * Return Value    : void
 *-----------------------------------------------------------------------------
 * Note            :
 *""FUNC COMMENT END""**********************************************************/
static void mii_preamble (void)
{
  short i;
  i = 32;

  while( i > 0 ) {
    /* 1 is output via the MII (Media Independent Interface) block */
    mii_write_1();
    i--;
  }
}
```

3.9 Sample program list "phy.c" (7)

```
/*""FUNC COMMENT""**************************************************************
 * ID              :
 * Outline         : Setting of the register mode of the PHY module
 *------------------------------------------------------------------------------
 * Include         :
 *------------------------------------------------------------------------------
 * Declaration     : static void mii_cmd (unsigned short reg_addr, int option );
 *------------------------------------------------------------------------------
 * Description     : Placing the PHY module register in read or write mode
 *------------------------------------------------------------------------------
 * Argument        : unsigned short reg_addr ; I : Address of the PHY register
 *                 : int option ; I : Specification of reading or writing
 *------------------------------------------------------------------------------
 * Return Value    : void
 *------------------------------------------------------------------------------
 * Note            :
 *""FUNC COMMENT END""**********************************************************/
static void mii_cmd (unsigned short reg_addr, int option)
{
  int i;
  unsigned short data;

  data = 0;
  data = (PHY_ST << 14); /* ST code */
  if (option == PHY_READ) {
    data |= (PHY_READ << 12); /* OP code(RD) */
  }
  else {
    data |= (PHY_WRITE << 12); /* OP code(WT) */
  }
  data |= (PHY_ADDR << 7); /* PHY Address */
  data |= (reg_addr << 2); /* Reg Address */

  for(i=14; i>0; i--){
    if ((data & 0x8000) == 0) {
        mii_write_0();
    }
    else {
        mii_write_1();
    }
    data <<= 1;
  }
}
```

3.10 Sample program list "phy.c" (8)

```
/*""FUNC COMMENT""**************************************************************
 * ID               :
 * Outline          : Obtaining the register value from the PHY module
 *------------------------------------------------------------------------------
 * Include          :
 *------------------------------------------------------------------------------
 * Declaration      : static void mii_reg_read (unsigned short *data);
 *------------------------------------------------------------------------------
 * Description      : Obtains the value of the PHY module register bit by bit.
 *                  : Signals are input/output in accord with the following conditions;
 *                  : MDC-high pulse width:160 ns (min)
 *                  : MDC-low pulse width:160 ns (min)
 *                  : MDC-cycle time: 400 ns (min)
 *                  : MDIO-output delay time (from PHY): 300 ns (max).
 *------------------------------------------------------------------------------
 * Argument        : unsigned short *data ; O : Address where obtained values are to be stored
 *------------------------------------------------------------------------------
 * Return Value     : void
 *------------------------------------------------------------------------------
 * Note             : The wait time may require adjustment to suit the system in use.
 *""FUNC COMMENT END""**********************************************************/
static void mii_reg_read (unsigned short *data)
{
  int i,j;
  unsigned short reg_data;

  /* Data are read in one bit at a time */
  reg_data = 0;
  for (i=16; i>0; i--){
    for(j=MDC_WAIT; j>0; j--){
        EtherC.PIR.LONG = 0x00000000;
    }
    for(j=MDC_WAIT; j>0; j--){
        EtherC.PIR.LONG = 0x00000001;
    }
    reg_data <<= 1;
    reg_data |= (EtherC.PIR.LONG & 0x00000008) >> 3; /* MDI read */

    for(j=MDC_WAIT; j>0; j--){
        EtherC.PIR.LONG = 0x00000001;
    }
    for(j=MDC_WAIT; j>0; j--){
        EtherC.PIR.LONG = 0x00000000;
    }
  }
  *data = reg_data;
}
```

3.11 Sample program list "phy.c" (9)

```
/*""FUNC COMMENT""**************************************************************
 * ID            :
 * Outline       : Setting of a value in a PHY module register
 *-------------------------------------------------------------------------------
 * Include       :
 *-------------------------------------------------------------------------------
 * Declaration   : static void mii_reg_write (unsigned short data );
 *-------------------------------------------------------------------------------
 * Description   : The value of the PHY module register is set one bit at a time.
 *-------------------------------------------------------------------------------
 * Argument      : unsigned short data ; I : The value to be set in the register
 *-------------------------------------------------------------------------------
 * Return Value  : void
 *-------------------------------------------------------------------------------
 * Note          : The wait time may require adjustment to suit the system in use.
 *""FUNC COMMENT END""**********************************************************/
static void mii_reg_write (unsigned short data)
{
  int i;

  /* Data are written one bit at a time.*/
  for(i=16; i>0; i--){
    if( (data & 0x8000) == 0 ) {
        mii_write_0();
    }
    else {
        mii_write_1();
    }
    data <<= 1;
  }
}
```

3.12 Sample program list "phy.c" (10)

```
/*""FUNC COMMENT""**************************************************************
 * ID            :
 * Outline       : Bus release for access to the register of the PHY module
 *------------------------------------------------------------------------------
 * Include       :
 *------------------------------------------------------------------------------
 * Declaration   : static void mii_z (void);
 *------------------------------------------------------------------------------
 * Description   : Reading is selected as the direction of access to the PHY module.
 *               : Signals are input/output in accord with the following conditions;
 *               : MDC-high pulse width: 160 ns (min)
 *               : MDC-low pulse width: 160 ns (min)
 *               : MDC-cycle time: 400 ns (min)
 *               : MDIO-setup time: 10 ns (min)
 *               : MDIO-hold time: 10 ns (min).
 *------------------------------------------------------------------------------
 * Argument      : void
 *------------------------------------------------------------------------------
 * Return Value  : void
 *------------------------------------------------------------------------------
 * Note          : The wait time may require adjustment to suit the system in use.
 *""FUNC COMMENT END""**********************************************************/
static void mii_z (void)
{
  int j;
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000000;
  }
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000001;
  }
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000001;
  }
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000000;
  }
}
```

3.13 Sample program list "phy.c" (11)

```
/*""FUNC COMMENT""**************************************************************
 * ID             :
 * Outline        : Output of the TA(10) bits for access to the register of the PHY module
 *------------------------------------------------------------------------------
 * Include        :
 *------------------------------------------------------------------------------
 * Declaration    : static void mii_ta10 (void);
 *------------------------------------------------------------------------------
 * Description    : Outputs 1 or 0 to the MII management interface of the PHY module.
 *------------------------------------------------------------------------------
 * Argument       : void
 *------------------------------------------------------------------------------
 * Return Value   : void
 *------------------------------------------------------------------------------
 * Note           :
 *""FUNC COMMENT END""**********************************************************/
static void mii_ta10 (void)
{
  mii_write_1();
  mii_write_0();
}
```

3.14 Sample program list "phy.c" (12)

```
/*""FUNC COMMENT""**************************************************************
 * ID               :
 * Outline          : Output of one bit (1) during access to the register of the PHY module
 *------------------------------------------------------------------------------
 * Include          :
 *------------------------------------------------------------------------------
 * Declaration      : static void mii_write_1 (void);
 *------------------------------------------------------------------------------
 * Description      : 1 is output to the MII management interface of the PHY module.
 *                  : Signals are output in accord with the following conditions;
 *                  : MDC-high pulse width: 160 ns (min)
 *                  : MDC-low pulse width:160 ns (min)
 *                  : MDC-cycle time: 400 ns (min)
 *                  : MDIO-setup time:10 ns (min)
 *                  : MDIO-hold time:10 ns (min)
 *------------------------------------------------------------------------------
 * Argument         : void
 *------------------------------------------------------------------------------
 * Return Value     : void
 *------------------------------------------------------------------------------
 * Note             : The wait time may require adjustment to suit the system in use.
 *""FUNC COMMENT END""**********************************************************/
static void mii_write_1 (void)
{
  int j;
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000006;
  }
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000007;
  }
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000007;
  }
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000006;
  }
}
```

3.15 Sample program list "phy.c" (13)

```
/*""FUNC COMMENT""**************************************************************
 * ID              :
 * Outline         : Output of one bit (0) during access to the register of the PHY module
 *------------------------------------------------------------------------------
 * Include         :
 *------------------------------------------------------------------------------
 * Declaration     : static void mii_write_0 (void);
 *------------------------------------------------------------------------------
 * Description     : 0 is output to the MII management interface of the PHY module.
 *                 : Signals are output in accord with the following conditions;
 *                 : MDC-high pulse width: 160 ns (min)
 *                 : MDC-low pulse width:160 ns (min)
 *                 : MDC-cycle time: 400 ns (min)
 *                 : MDIO-setup time:10 ns (min)
 *                 : MDIO-hold time:10 ns (min)
 *------------------------------------------------------------------------------
 * Argument        : void
 *------------------------------------------------------------------------------
 * Return Value    : void
 *------------------------------------------------------------------------------
 * Note            : The wait time may require adjustment to suit the system in use.
 *""FUNC COMMENT END""**********************************************************/
static void mii_write_0 (void)
{
  int j;
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000002;
  }
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000003;
  }
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000003;
  }
  for(j=MDC_WAIT; j>0; j--){
    EtherC.PIR.LONG = 0x00000002;
  }
}
/* End of File */
```

3.16 Sample program list "phy.h" (1)

```
/******************************************************************************
*   DISCLAIMER
*
*   This software is supplied by Renesas Electronics Corporation and is only
*   intended for use with Renesas products. No other uses are authorized.
*
*   This software is owned by Renesas Electronics Corporation and is protected under
*   all applicable laws, including copyright laws.
*
*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
*   DISCLAIMED.
*
*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
*   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
*
*   Renesas reserves the right, without notice, to make changes to this
*   software and to discontinue the availability of this software.
*   By using this software, you agree to the additional terms and
*   conditions found by accessing the following link:
*   http://www.renesas.com/disclaimer
********************************************************************************
* Copyright (C) 2007(2010) Renesas Electronics Corporation. All rights reserved.
*""FILE COMMENT""*********** Technical reference data **************************
*   System Name : SH7671 Sample Program
*   File Name   : phy.h
*   Abstract    : Example of Setting for Automatic Negotiation by Ethernet PHY-LSI
*   Version     : 1.00.01
*   Device      : SH7671
*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.03.00).
*               : C/C++ compiler package for the SuperH RISC engine family
*               :                             (Ver.9.01 Release01).
*   OS          : None
*   H/W Platform: M3A-HS71(CPU board)
*   Description :
********************************************************************************
*   History     : Jul.04,2007 ver.1.00.00
*               : May 12,2010 ver.1.00.01 Changed the company name and device name
*""FILE COMMENT END""**********************************************************/
```

3.17 Sample program list "phy.h" (2)

```
#ifndef _PHY_H
#define _PHY_H

#define NEGO_FAIL         0
#define HALF_10M          1
#define FULL_10M          2
#define HALF_TX           3
#define FULL_TX           4

int  phy_autonego( void );

#endif

/* End of File */
```

4. References

- Software Manual
  SH-2A/SH2A-FPU Software Manual Rev. 3.00
  The latest version of the software manual can be downloaded from the Renesas Electronics website.

- Hardware Manual
  SH7670 Group Hardware Manual Rev. 2.00
  The latest version of the hardware user's manual can be downloaded from the Renesas Electronics website.

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	Dec.24.08	—	First edition issued
1.01	Oct.15.10	—	Changed the sample program (AC Switching Characteristics are removed)

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. 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.

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: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, 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.
7F, No. 363 Fu Shing North Road Taipei, Taiwan, R.O.C.
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

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 1.0