V850E2/MN4

CSIH Control

R01AN0924EJ0100
Rev.1.00
Feb 10, 2012

Introduction

This application note explains how to set up the CSIH (clocked three-wire serial interface) and also gives an outline of the operation and describes the procedures for using a sample program. The sample program transmits and receives data between the CSIH0 and CSIH3. The CSIH0 transmits data in master mode, while the CSIH3 receives data in slave mode. The sample program uses two memory modes: direct access modes and dual buffer mode.

Target Device

V850E2/MN4 Microcontrollers

Contents

1. Overview 2

2. Usage Environment 8

3. Software 9

4. Sample Application 10

1. Overview

This application note explains the following four operation modes of the CSIH as usage examples:

- Master dual-buffer transmit-only mode
- Slave dual-buffer receive-only mode
- Master direct-access transmit-only mode
- Slave direct-access receive-only mode

In master mode, the serial communication clock is generated by the internal baudrate generator (BRG) and supplied by signal CSIHnTSCK. In slave mode, another device is the communication master. The communication clock is supplied.

See section 4.1 “Flow Charts” for the details of the sample program.

The main points in master dual-buffer transmit-only mode are illustrated below.

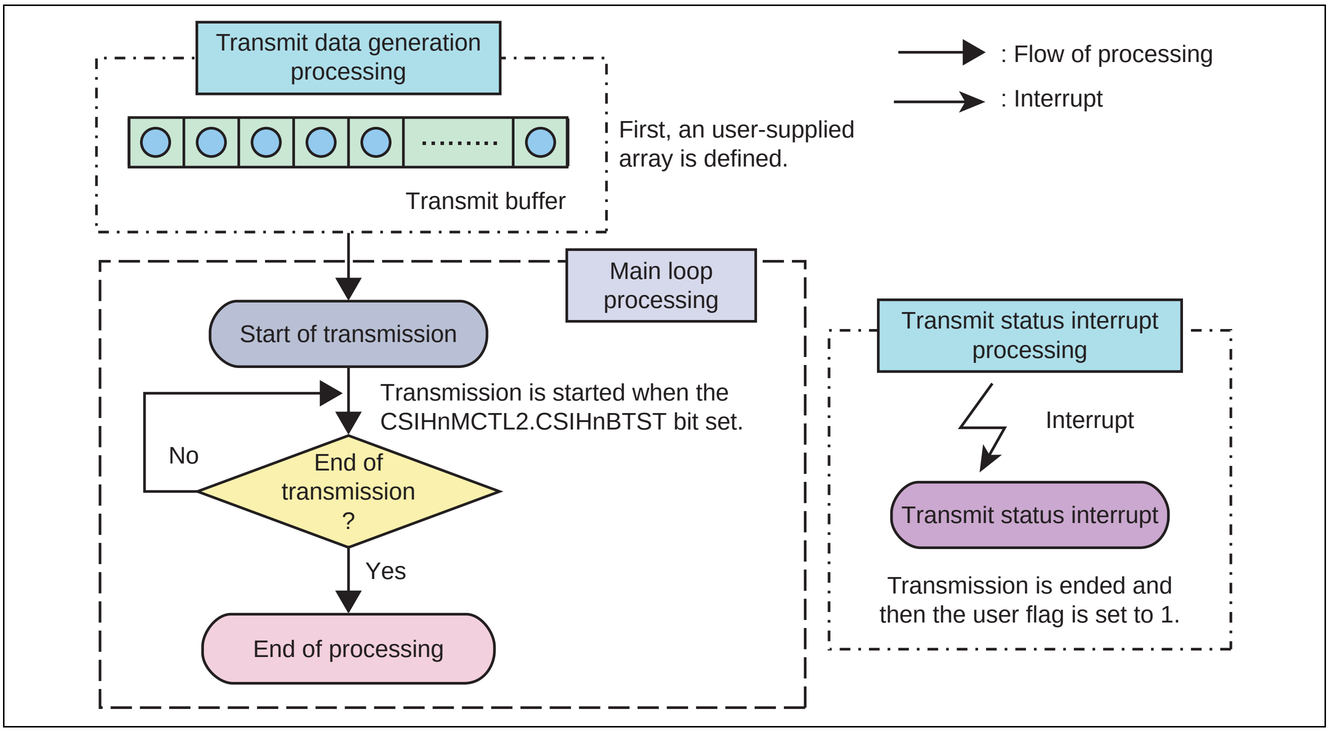

Transmit data generation processing
....
Transmit buffer
First, an user-supplied array is defined.
: Flow of processing
: Interrupt
Main loop processing
Start of transmission
Transmission is started when the CSIHnMCTL2.CSIHnBTST bit set.
No
End of transmission ?
Yes
End of processing
Transmit status interrupt processing
Interrupt
Transmit status interrupt
Transmission is ended and then the user flag is set to 1.

Figure 1.1 Master Dual-Buffer Transmit-Only Mode

The main points in slave dual-buffer receive-only mode are illustrated below.

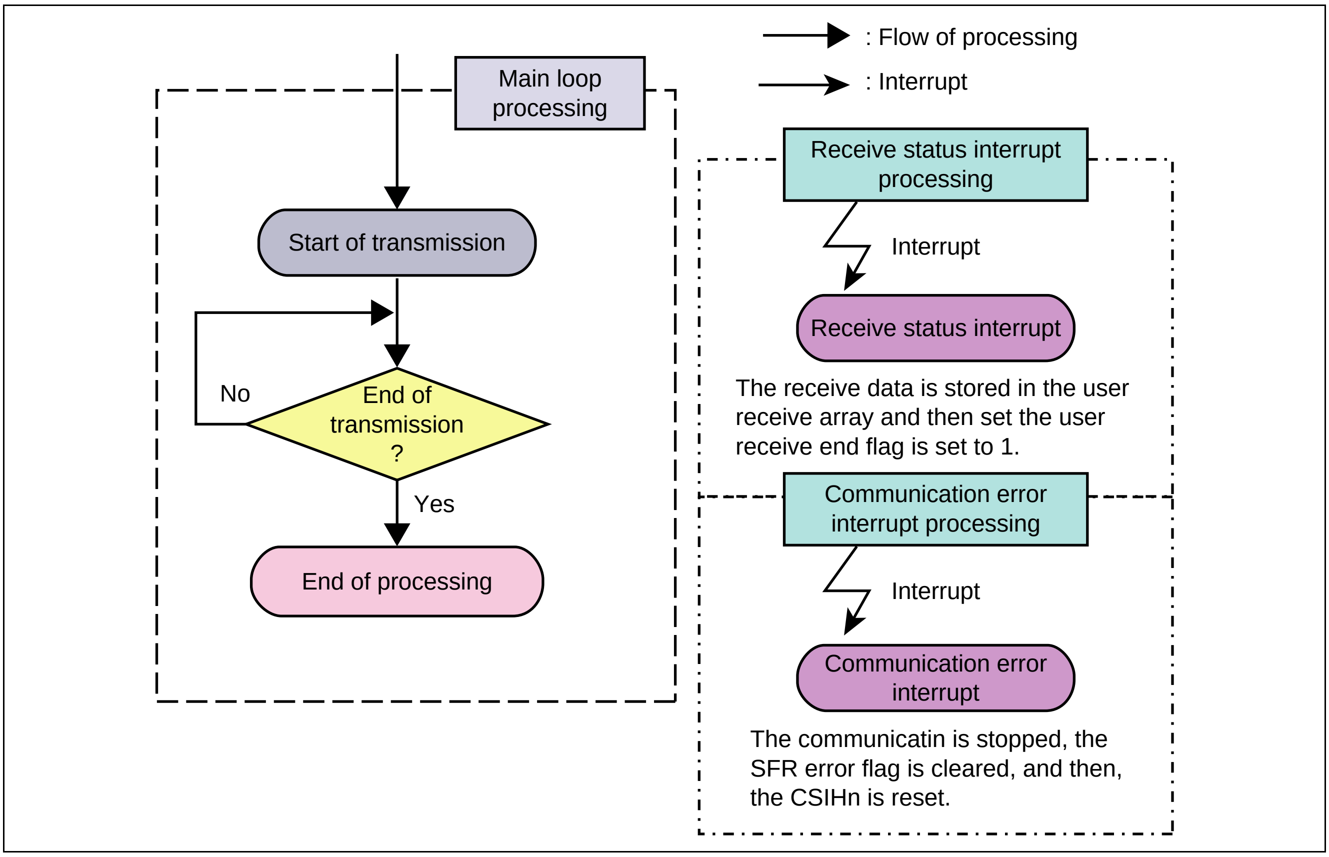

: Flow of processing
: Interrupt
Main loop processing
Start of transmission
End of transmission ?
No
Yes
End of processing
Receive status interrupt processing
Interrupt
Receive status interrupt
The receive data is stored in the user receive array and then set the user receive end flag is set to 1.
Communication error interrupt processing
Interrupt
Communication error interrupt
The communicatin is stopped, the SFR error flag is cleared, and then, the CSIHn is reset.

Figure 1.2 Slave Dual-Buffer Receive-Only Mode

The main points in master direct-access transmit-only mode (Job mode is enabled) are illustrated below.

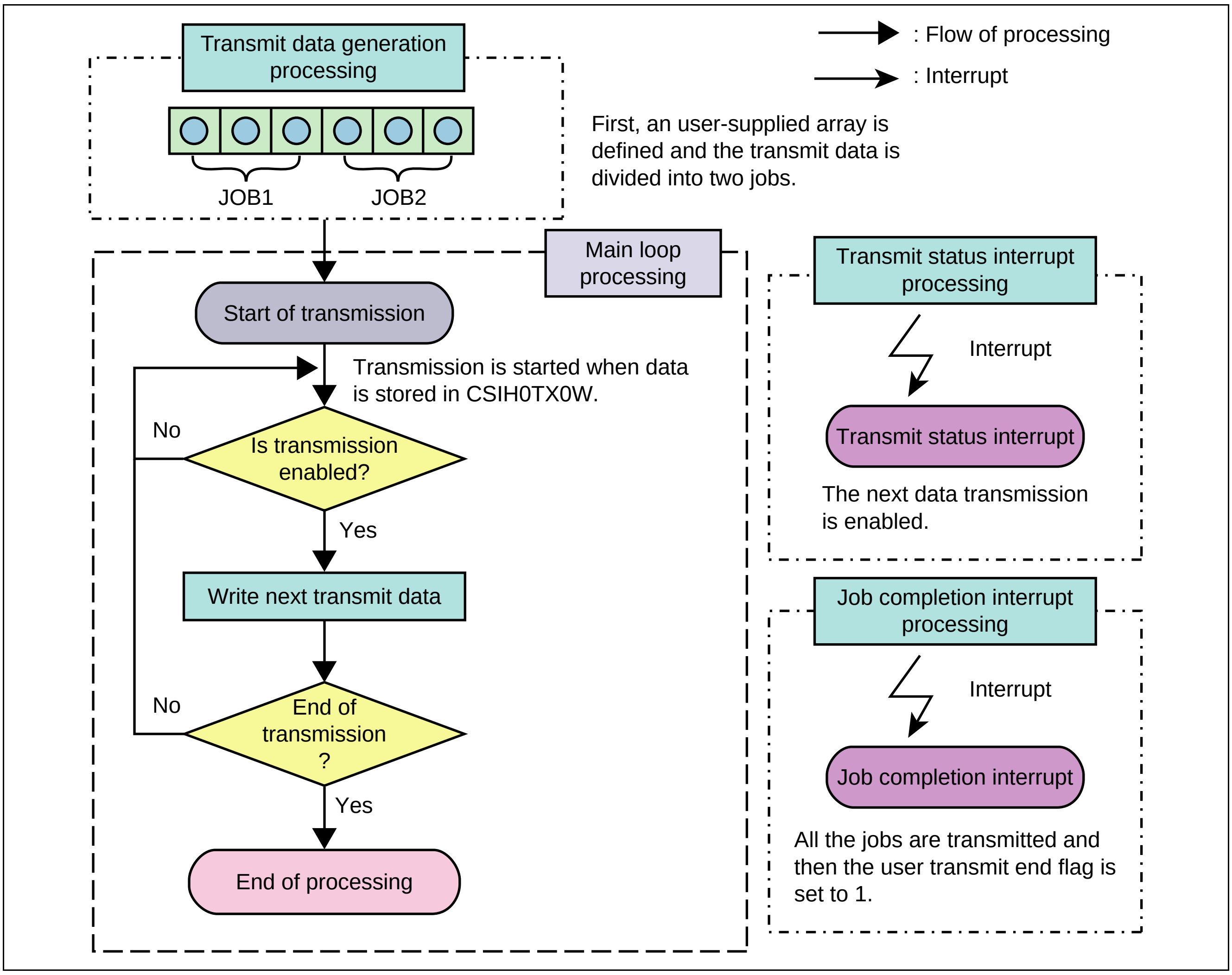

: Flow of processing
: Interrupt
Transmit data generation processing
JOB1
JOB2
First, an user-supplied array is defined and the transmit data is divided into two jobs.
Main loop processing
Start of transmission
Transmission is started when data is stored in CSIH0TX0W.
No
Is transmission enabled?
Yes
Write next transmit data
No
End of transmission ?
Yes
End of processing
Transmit status interrupt processing
Interrupt
Transmit status interrupt
The next data transmission is enabled.
Job completion interrupt processing
Interrupt
Job completion interrupt
All the jobs are transmitted and then the user transmit end flag is set to 1.

Figure 1.3 Master Direct-Access Transmit-Only Mode (Job Mode Is Enabled)

The main points in slave direct-access receive-only mode are illustrated below.

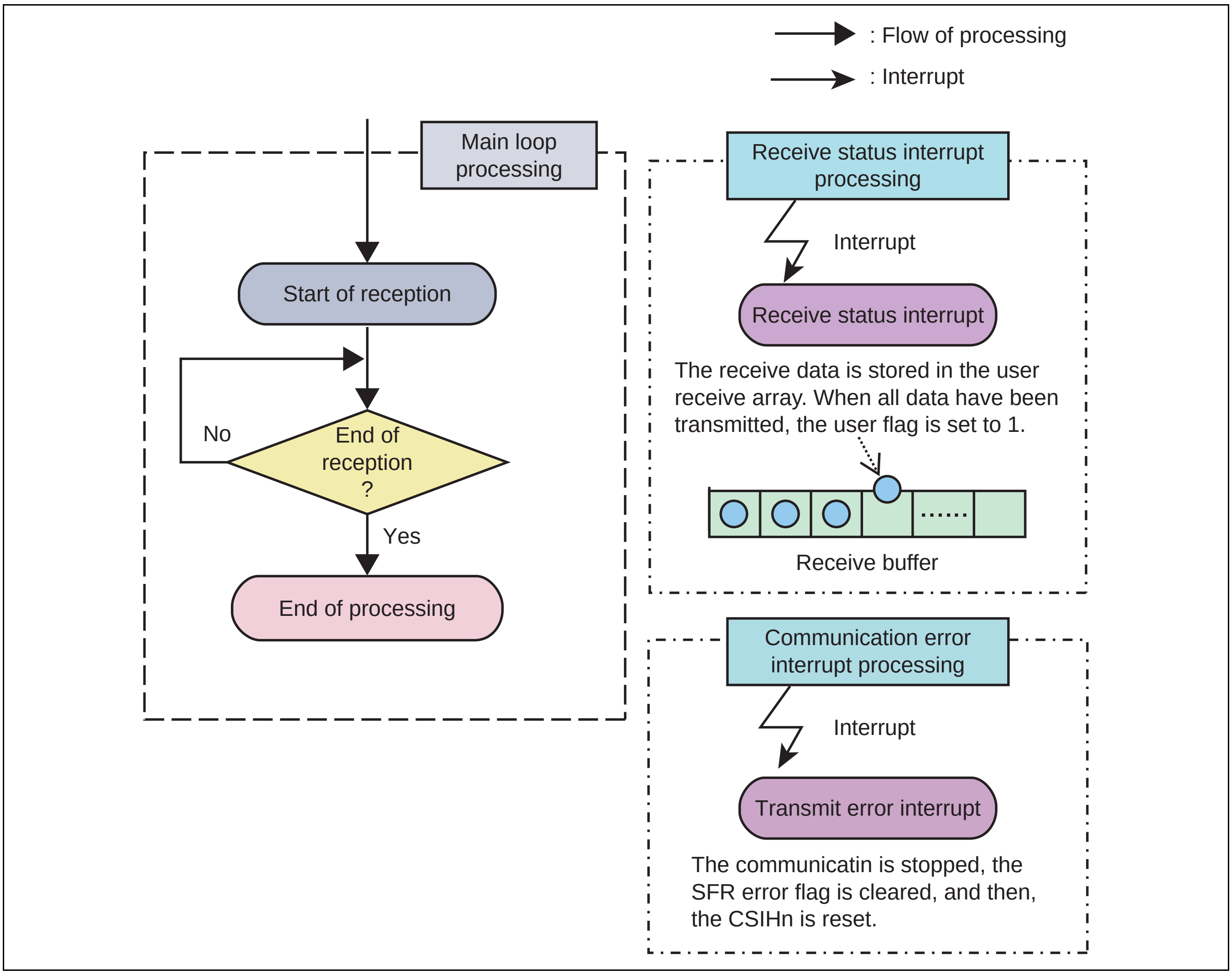

: Flow of processing
: Interrupt
Main loop processing
Start of reception
End of reception ?
No
Yes
End of processing
Receive status interrupt processing
Interrupt
Receive status interrupt
The receive data is stored in the user receive array. When all data have been transmitted, the user flag is set to 1.
......
Receive buffer
Communication error interrupt processing
Interrupt
Transmit error interrupt
The communicatin is stopped, the SFR error flag is cleared, and then, the CSIHn is reset.

Figure 1.4 Slave Direct-Access Receive-Only Mode

The basic communication specifications are shown below.

Memory mode	Direct access mode	Dual buffer mode
Receive I/F	CSIH3	
Transmit I/F	CSIH0	
Transfer direction	MSB first	
Parity bits during transmission/reception	No parity bit	
Data length	8 bits	
Baud rate	64 kbps	
FIFO size	None	64 bytes (each of transmit FIFO and receive FIFO
Communication data length	6 bytes	9 bytes
JOB (jobs)	2 jobs	None
EDL (extended data length)	None	
LBM (loop-back mode)	None	
SS (slave select)	None	

1.1 Initialization

The general registers and functional pins are initialized.

<Port setup>

- Port n function control expansion registers (PFCEn)
- Port n function control registers (PFCn)
- Port n mode control registers (PMCn)
- Port n mode registers (PMn)

1.2 CSIH Setup

The registers listed below are set up to control the operation of the CSIH. See section 4.2 for the details.

<CSIH control setup>

- CSIHn control register 0 (CSIHnCTL0)
- CSIHn control register 1 (CSIHnCTL1)
- CSIHn control register 2 (CSIHnCTL2)
- CSIHn memory control register 0 (CSIHnMCTL0)
- CSIHn memory control register 1 (CSIHnMCTL1)
- CSIHn memory control register 2 (CSIHnMCTL2)
- CSIHn configuration register x (CSIHnCFGx)

1.3 Interrupt Enabling

Interrupts are enabled by the EI instruction.

1.4 Main Loop Processing

- The operation of data transfer via the CSIH depends on the memory mode.

  In dual buffer mode, the transmitting CSIH0 transmits nine bytes of data and the receiving CSIH3 receives the nine bytes of data from the CSIH0. The internal RAM has nine bytes (one byte (one block data) × nine) of an area for each of the user transmit array to store transmit data and the user receive array to store receive data.

  In direct access mode, the transmitting CSIH0 prepares six-byte data, divides the data into three bytes as a job, and transmits the two jobs. The receiving CSIH3 receives six bytes of data and stores the data in the internal RAM. The internal RAM has the six-byte (one byte (one block data) × six) user transmit array to store transmit data and the six-byte user receive array to store receive data.

- The conditions under which data transfer via the CSIH starts depend on the memory mode.

  <1> Transmission in CSIH master mode:

  In direct access mode, data transmission is started by writing data to the CSIHnTX0W register after the setup of the CSIH ends.

  In dual buffer mode, data transmission is started by setting the CSIHnMCTL2.CSIHnBTST bit to 1.

  <2> Reception in CSIH slave mode:

  In direct access mode, data reception is started by detecting external clock CSIHnTSCK.

  In dual buffer mode, data reception is started by setting the CSIHnMCTL2.CSIHnBTST bit to 1.

- Interrupts occur at the timings described below.

  In direct access transmit mode, a transmit status interrupt (CSIHnTIC) occurs after each unit of data is transmitted. In job mode, a job completion interrupt (CSIHnTIJC) occurs after the transfer of the job that is enabled by setting the CSIH0CTL0.JOBE bit to 1 is completed.

  In direct access receive mode, a receive status interrupt (CSIHnTIR) occurs each time data is received.

  In dual buffer mode, a transmit status interrupt (CSIHnTIC) and a receive status interrupt (CSIHnTIR) occur after the specified amount of data is transferred.

  A communication status interrupt (CSIHnTIRE) is generated whenever a communication error occurs.

2. Usage Environment

This section provides the circuit diagram and operating environment of the hardware on which this sample program is to run.

2.1 Circuit Diagram

See “V850E2/MN4 Target Board User Manual: QB-V850E2MN4DUAL-TB (R20UT0683XJ)” for the details of the circuit diagram.

This sample program performs CSI communication between the CSIH0 and the CSIH3. The CSIH0 transmits data in master mode and the CSIH3 receives the data in slave mode. The P4_12 pin, the P4_13 pin, and the P4_11 pin are used for the SO0F pin, the SI0F pin, and the SCK0F pin for the CSIH0, respectively. The P4_3 pin, the P4_6 pin, and the P4_7 pin are used for the SO3F pin, the SI3F pin, and the SCK3F pin for the CSIH3, respectively. The SO0F pin is connected to the SI3F pin and the SCK0F pin is connected to the SCK3F pin.

LED1 and LED2 are connected to port 13. The P13_7 pin is used for LED1. The P13_6 pin is used for LED2.

2.2 Development Environment

It is necessary to install the tools that are listed below to run the sample program.

- CubeSuite+
  The integrated development environment CubeSuite+ from Renesas Electronics provides various software development tools that are necessary for the user to develop applications. The user can use these tools seamlessly and easily in various development stages including coding, assembly, compilation, debugging using an emulator or simulator, and flash programming.

- MINICUBE
  MINICUBE is a general-purpose in-circuit emulator from Renesas Electronics which adopts the JTAG interface system. It allows the user to debug an onboard real processor and provides highly transparent and stable emulation functionalities. An adapter is required to connect a TB board to MINICUBE.

- Multi
  Green Hills software, Inc. integrated development tool suit.

  IAR Embedded Workbench
  IAR Systems integrated development tool suit.

3. Software

This section describes the organization of the compressed files to be downloaded.

3.1 File Organization

The compressed files to be downloaded is summarized below.

File Name (Tool Structure)	Description	Common Source File	CubeSuite+ File	Multi File
crtE.s	Hardware initialization processing		●	
startup.s				●
V850E2MN4.dir	Link/directive file		●	
V850E2_MN4 CSIH.ld				●
vector.s	Vector table			●
csih.h	Variable and function declarations	●		
main.c	Main processing	●		
initial.c	Software initialization processing	●		
csih_transmit.c	Transmit processing	●		
csih_receive.c	Receive processing	●		
interrupt.c	Interrupt processing	●		

4. Sample Application

This section explains how to set up the CSIH functions.

4.1 Flow Charts

The flow charts of this sample program are given below. The pertinent subroutines are entered according to the operating mode that is set up by the user.

4.1.1 Main Processing

The main processing disables maskable interrupts first. After each setup ends, the maskable interrupts and transmit/receive status interrupts are enabled. The main loop processing repeatedly controls communication and awaits the end of communication.

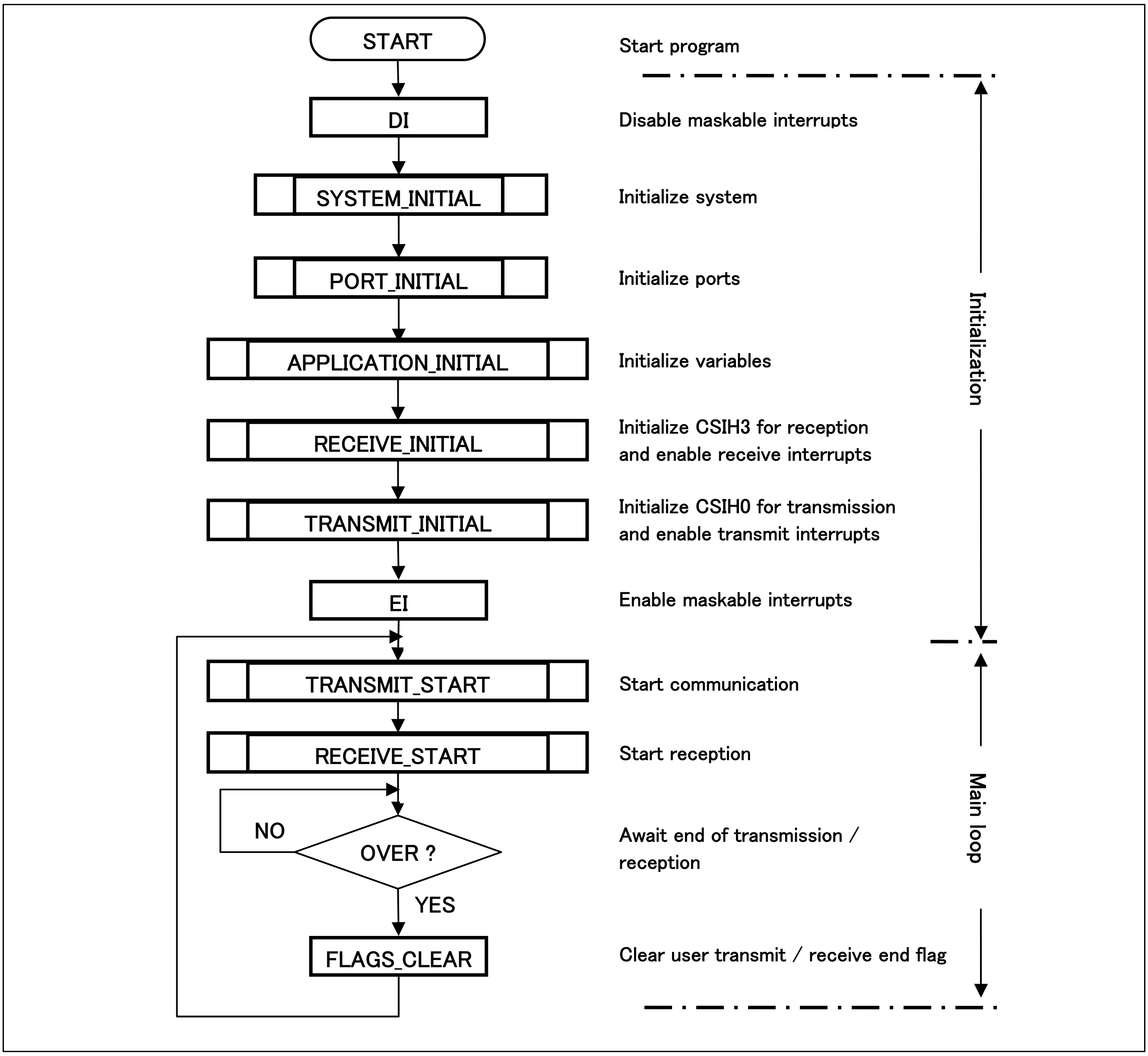
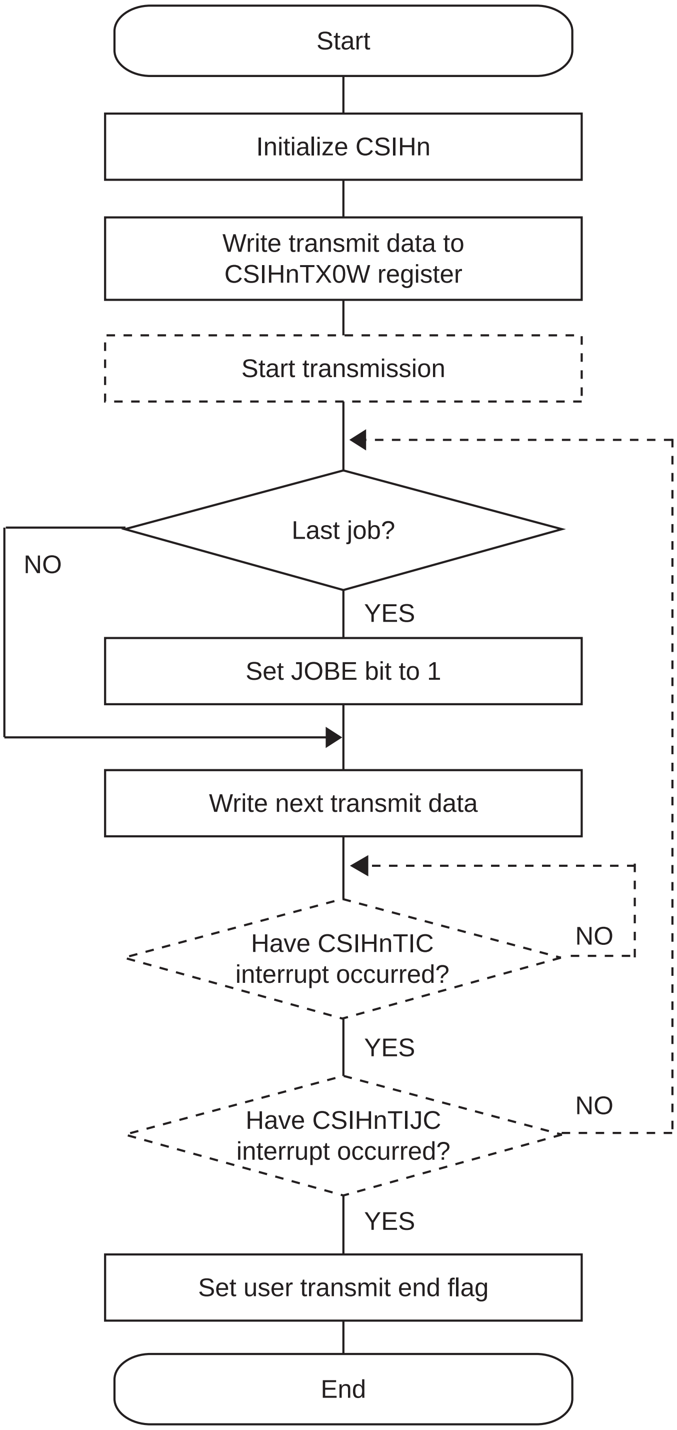


Figure 4.1 Main Processing Flowchart

4.1.2 Master Direct-Access Transmit-Only Mode

Master direct-access transmit-only mode is started by writing transmit data to the CSIHnTX0W register. Forty-eight bits (six bytes) of transmit data are divided into two jobs before transmission. The communication data length is eight bits.


Start
Initialize CSIHn
Write transmit data to CSIHnTX0W register
Start transmission
Last job?
NO
YES
Set JOBE bit to 1
Write next transmit data
Have CSIHnTIC interrupt occurred?
NO
YES
Have CSIHnTIJC interrupt occurred?
NO
YES
Set user transmit end flag
End

Remark 1. The broken lines indicate hardware processing.

Figure 4.2 Flowchart of Master Direct-Access Transmit-Only Mode

4.1.3 Slave Direct-Access Receive-Only Mode

Slave direct-access receive-only mode is started by detecting external clock CSIHTSCK.

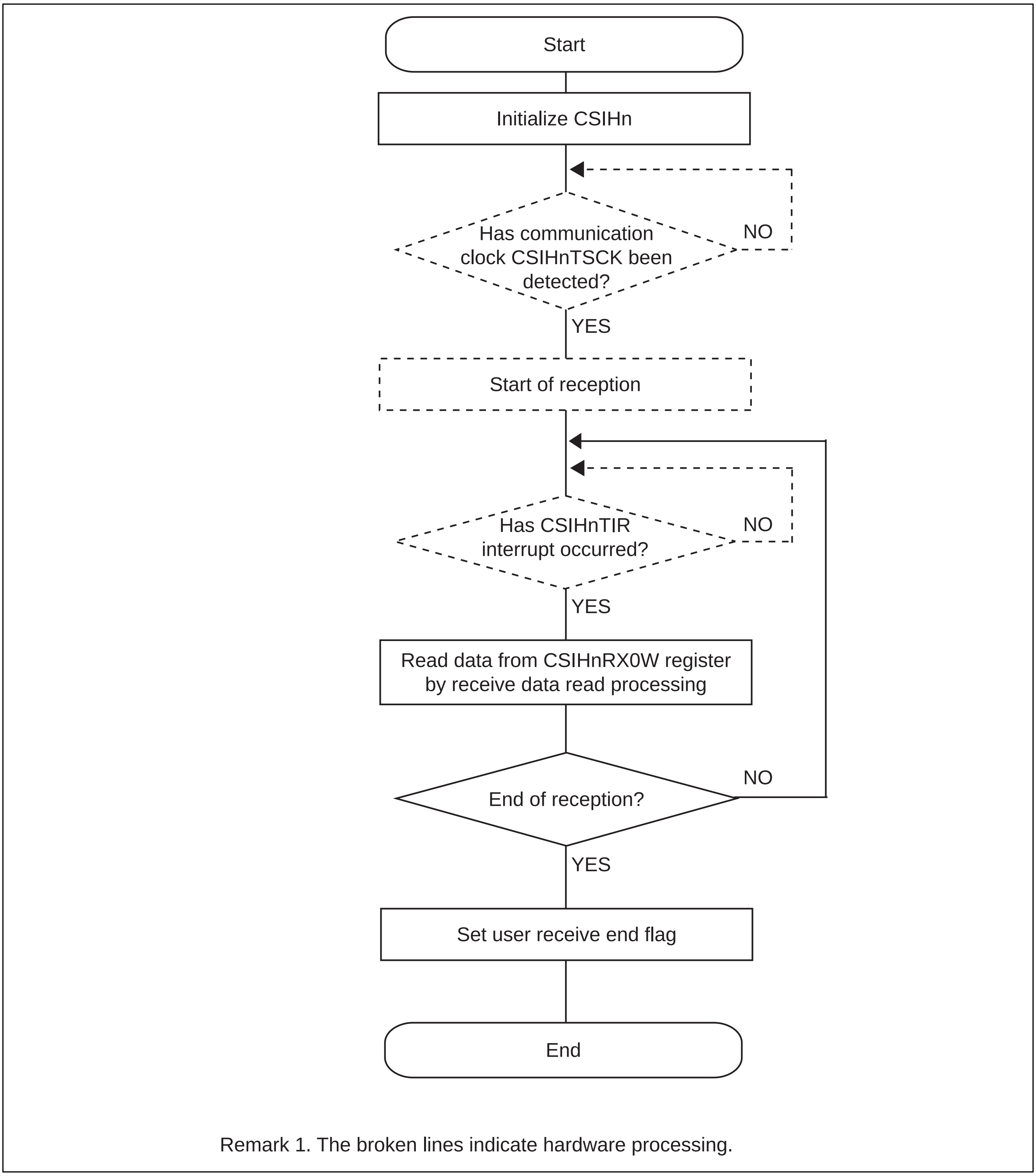
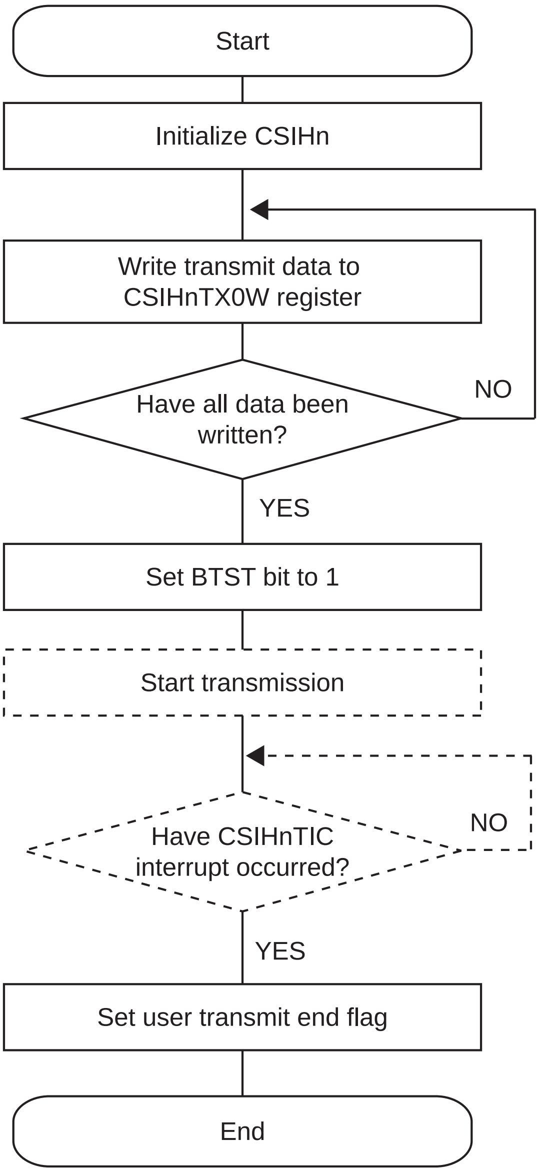

Start
Initialize CSIHn
Has communication clock CSIHnTSCK been detected?
NO
YES
Start of reception
Has CSIHnTIR interrupt occurred?
NO
YES
Read data from CSIHnRX0W register by receive data read processing
End of reception?
NO
YES
Set user receive end flag
End
Remark 1. The broken lines indicate hardware processing.

Figure 4.3 Flowchart of Slave Direct-Access Receive-Only Mode

4.1.4 Master Dual-Buffer Transmit-Only Mode

Master dual-buffer transmit-only mode is started by setting the CSIHnMCTL2.CSIHnBTST bit to 1.

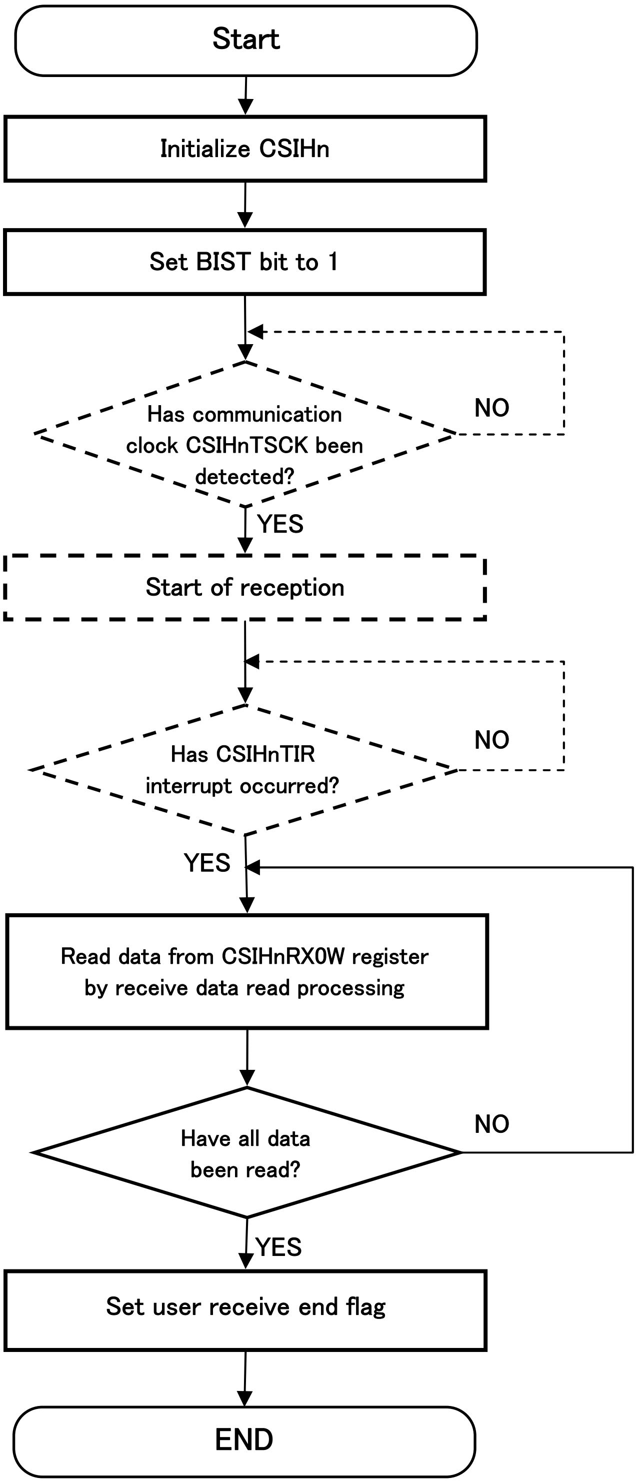

Start
Initialize CSIHn
Write transmit data to CSIHnTX0W register
Have all data been written?
NO
YES
Set BTST bit to 1
Start transmission
Have CSIHnTIC interrupt occurred?
NO
YES
Set user transmit end flag
End

Remark 1. The broken lines indicate hardware processing

Figure 4.4 Flowchart of Master Dual-Buffer Transmit-Only Mode

4.1.5 Slave Dual-Buffer Receive-Only Mode

Slave dual-buffer receive-only mode is started by setting the CSIHnMCTL2.CSIHnBTST bit to 1 and then detecting external clock CSIHTSCK.


Start
Initialize CSIHn
Set BIST bit to 1
Has communication clock CSIHnTSCK been detected?
NO
YES
Start of reception
Has CSIHnTIR interrupt occurred?
NO
YES
Read data from CSIHnRX0W register by receive data read processing
Have all data been read?
NO
YES
Set user receive end flag
END

Remark 1. The broken lines indicate hardware processing

Figure 4.5 Flowchart of Slave Dual-Buffer Receive-Only Mode

4.1.6 Communication Error Interrupt Processing

If a communication error occurs, a communication error interrupt is generated. Then, the communication error interrupt processing is executed. The communication is stopped and the SFR error flag is cleared. The CSIHn is reset at the same time.

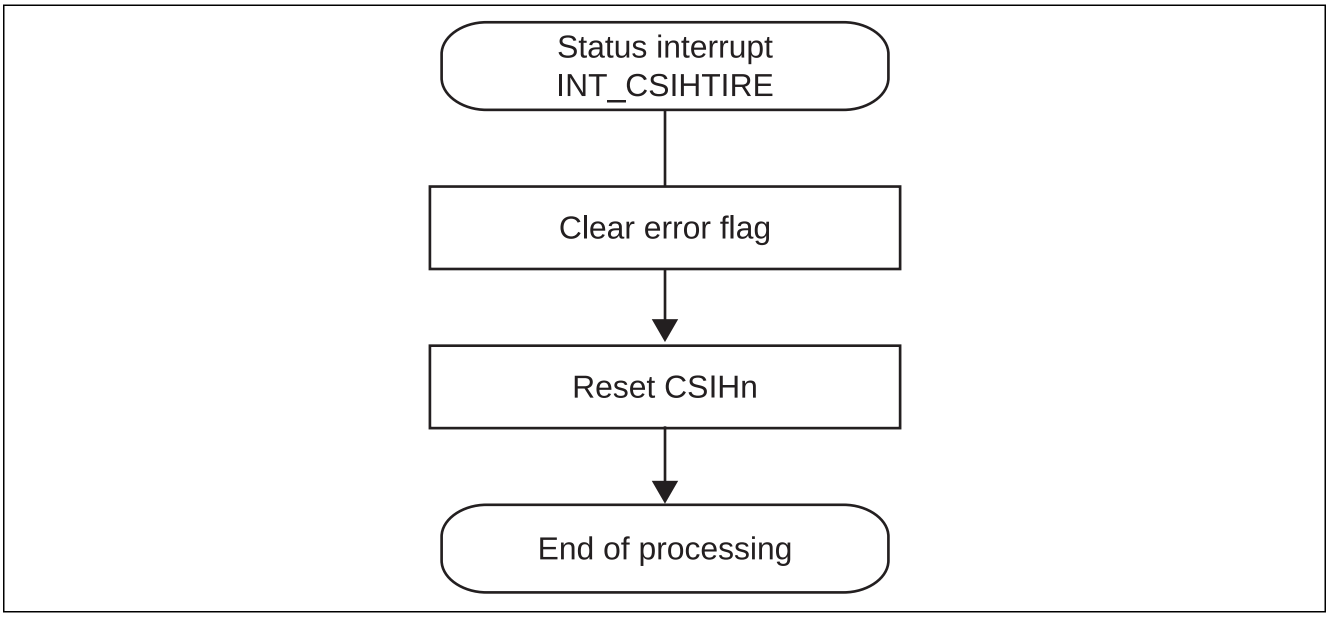


Figure 4.6 Communication Error Interrupt Processing

4.2 Register Setup

This section explains how to set up the relevant registers according to the flow charts shown in section 4.1. The registers described below must be configured to control the CSIH.

4.2.1 Port Setup

The program described in this application note executes serial transmission/reception by using two macros, the CSIH0 and the CSIH3. The relevant ports must be set up so that the pins for the CSIH0 and the CSIH3 are enabled.

The LEDs are connected to port 13. The P13_7 pin is used for LED1. The P13_6 pin is used for LED2.

Macro	Pin	PMC	PFCE	PFC	PM	Corresponding Function
CSIH0	SO0F	1	1	1	0	Alternative mode 4, output
	SI0F	1	1	1	1	Alternative mode 4, input
	SCK0F	1	1	1	0	Alternative mode 4, output
CSIH3	SO3F	1	1	1	0	Alternative mode 4, output
	SI3F	1	1	1	1	Alternative mode 4, input
	SCK3F	1	1	1	1	Alternative mode 4, input
PORT	P13_6	0	0	0	0	Port mode, output
	P13_7	0	0	0	0	Port mode, output

Setting examples

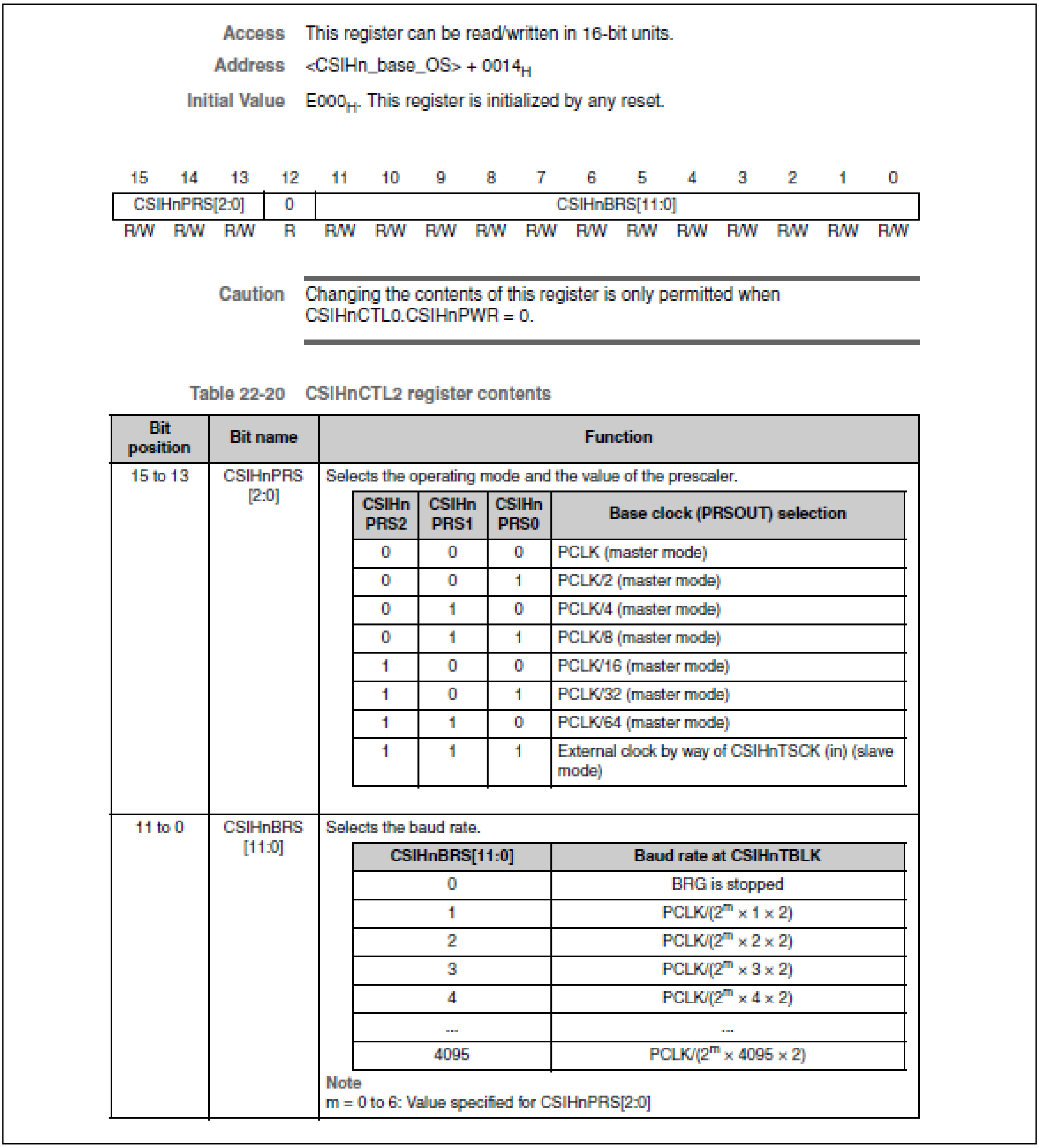
```
/* alternative mode 4 in that csih
  CSIH0:Master Mode,transmission
      P4_12: CSIHTA0SO
      P4_13: CSIHTA0SI
      P4_11: CSIHTA0SCO
  CSIH3:Slave Mode,receptiom
      P4_3: CSIHTA3SO
      P4_6: CSIHTA3SI
      P4_7: CSIHTA3SCI  */
PFCE4 = 0x38c8;
PFC4  = 0x38c8;
PMC4  = 0x38c8;
PM4   = 0x20c0;

/* P13_6,7: LEDs,IO,OUTPUT */
PFCE13 = 0x0000;
PFC13  = 0x0000;
PMC13  = 0x0000;
PM13   = 0x0000;
```

4.2.2 CSIH Control Register 2 (CSIHnCTL2)

The CSIHnCTL2 register selects the communication clock.

In master mode, the transmission baud rate can be selected by the CSIHnPRS[2:0] bits and the CSIHnBRS[11:0] bits in the CSIHnCTL2 register. The maximum available baud rate is Pclk/4 in master mode and Pclk/6 in slave mode. The minimum available baud rate is Pclk/524160 in both modes.

In this sample program, the communication clock is set to 64 kbps, and the CSIHnPRS[2:0] bits are set to 1, and the CSIHnBRS[11:0] bits are set to 260.



Access	This register can be read/written in 16-bit units.
Address	<CSIHn_base_OS> + 0014_H
Initial Value	$E000_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnPRS[2:0]			0	CSIHnBRS[11:0]											
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Changing the contents of this register is only permitted when CSIHnCTL0.CSIHnPWR = 0.

Table 22-20 CSIHnCTL2 register contents

Bit position	Bit name	Function
15 to 13	CSIHnPRS [2:0]	Selects the operating mode and the value of the prescaler.
11 to 0	CSIHnBRS [11:0]	Selects the baud rate.

CSIHn PRS2	CSIHn PRS1	CSIHn PRS0	Base clock (PRSOUT) selection
0	0	0	PCLK (master mode)
0	0	1	PCLK/2 (master mode)
0	1	0	PCLK/4 (master mode)
0	1	1	PCLK/8 (master mode)
1	0	0	PCLK/16 (master mode)
1	0	1	PCLK/32 (master mode)
1	1	0	PCLK/64 (master mode)
1	1	1	External clock by way of CSIHnTSCK (in) (slave mode)

CSIHnBRS[11:0]	Baud rate at CSIHnTBLK
0	BRG is stopped
1	PCLK/(2^m × 1 × 2)
2	PCLK/(2^m × 2 × 2)
3	PCLK/(2^m × 3 × 2)
4	PCLK/(2^m × 4 × 2)
...	...
4095	PCLK/(2^m × 4095 × 2)

Note
m = 0 to 6: Value specified for CSIHnPRS[2:0]

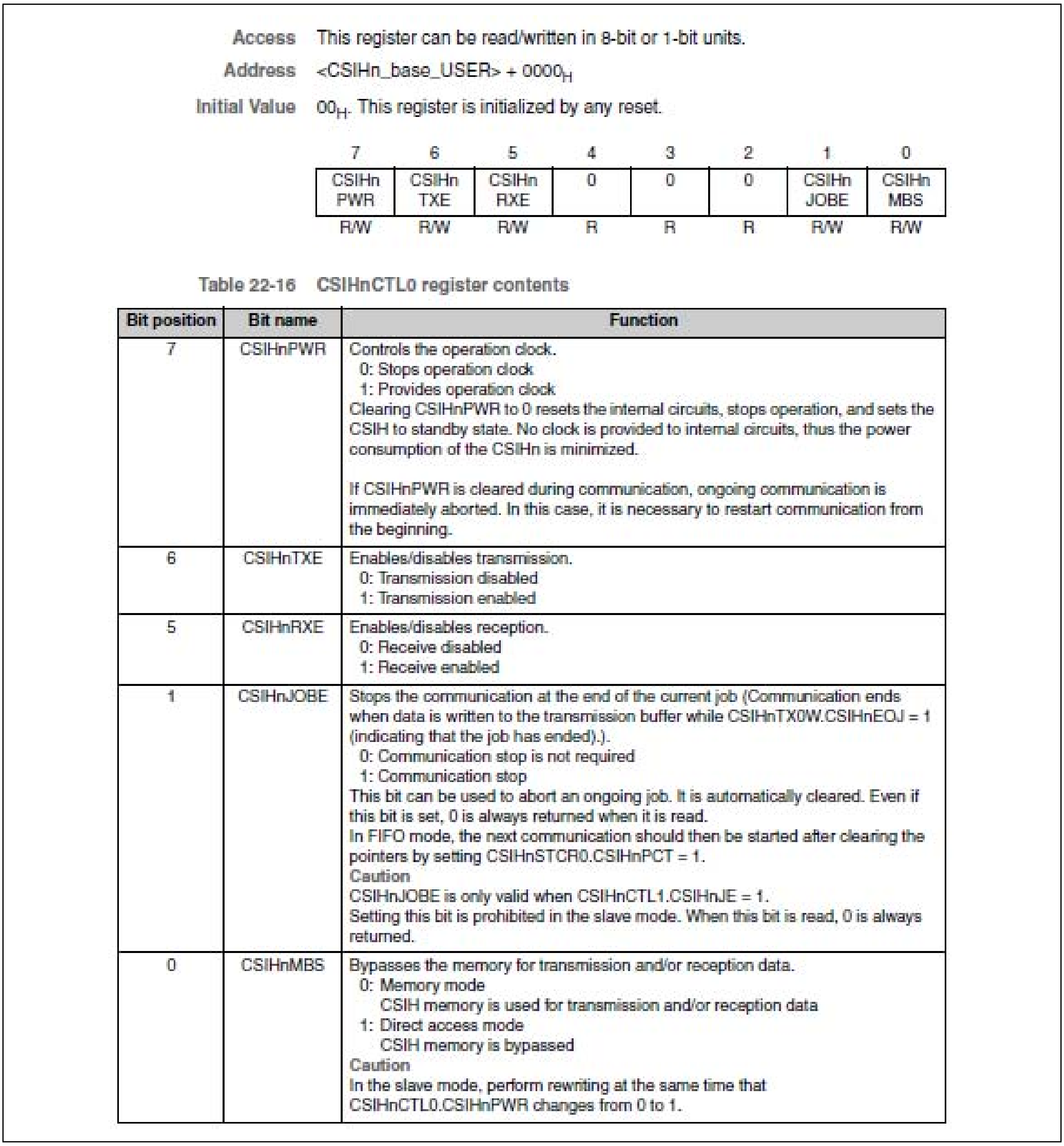
Figure 4.7 CSIHnCTL2 Register Format

Setting example

```
CSIHnCTL2 = 0x2104;                  /* master mode;Pclk/2^1*260*2 */
CSIHnCTL2 = 0xE000;                  /* slave mode */
```

4.2.3 CSIH Control Register 0 (CSIHnCTL0)

The CSIHnCTL0 register controls the operation clock, enables or disables transmission and reception, and specifies the use for the CSIH memory. It forces the stop of communication at the end of the current job.



Access This register can be read/written in 8-bit or 1-bit units.

Address <CSIHn_base_USER> + 0000_H

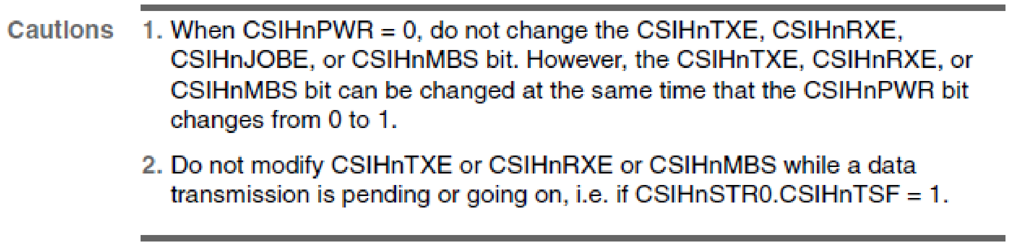
Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
CSIHn PWR	CSIHn TXE	CSIHn RXE	0	0	0	CSIHn JOBE	CSIHn MBS
R/W	R/W	R/W	R	R	R	R/W	R/W

Table 22-16 CSIHnCTL0 register contents

Bit position	Bit name	Function
7	CSIHnPWR	Controls the operation clock. 0: Stops operation clock 1: Provides operation clock Clearing CSIHnPWR to 0 resets the internal circuits, stops operation, and sets the CSIH to standby state. No clock is provided to internal circuits, thus the power consumption of the CSIHn is minimized. If CSIHnPWR is cleared during communication, ongoing communication is immediately aborted. In this case, it is necessary to restart communication from the beginning.
6	CSIHnTXE	Enables/disables transmission. 0: Transmission disabled 1: Transmission enabled
5	CSIHnRXE	Enables/disables reception. 0: Receive disabled 1: Receive enabled
1	CSIHnJOBE	Stops the communication at the end of the current job (Communication ends when data is written to the transmission buffer while CSIHnTX0W.CSIHnEOJ = 1 (indicating that the job has ended).). 0: Communication stop is not required 1: Communication stop This bit can be used to abort an ongoing job. It is automatically cleared. Even if this bit is set, 0 is always returned when it is read. In FIFO mode, the next communication should then be started after clearing the pointers by setting CSIHnSTCR0.CSIHnPCT = 1. Caution CSIHnJOBE is only valid when CSIHnCTL1.CSIHnJE = 1. Setting this bit is prohibited in the slave mode. When this bit is read, 0 is always returned.
0	CSIHnMBS	Bypasses the memory for transmission and/or reception data. 0: Memory mode CSIH memory is used for transmission and/or reception data 1: Direct access mode CSIH memory is bypassed Caution In the slave mode, perform rewriting at the same time that CSIHnCTL0.CSIHnPWR changes from 0 to 1.

Figure 4.8 CSIHnCTL0 Register Format (1/2)



Cautions

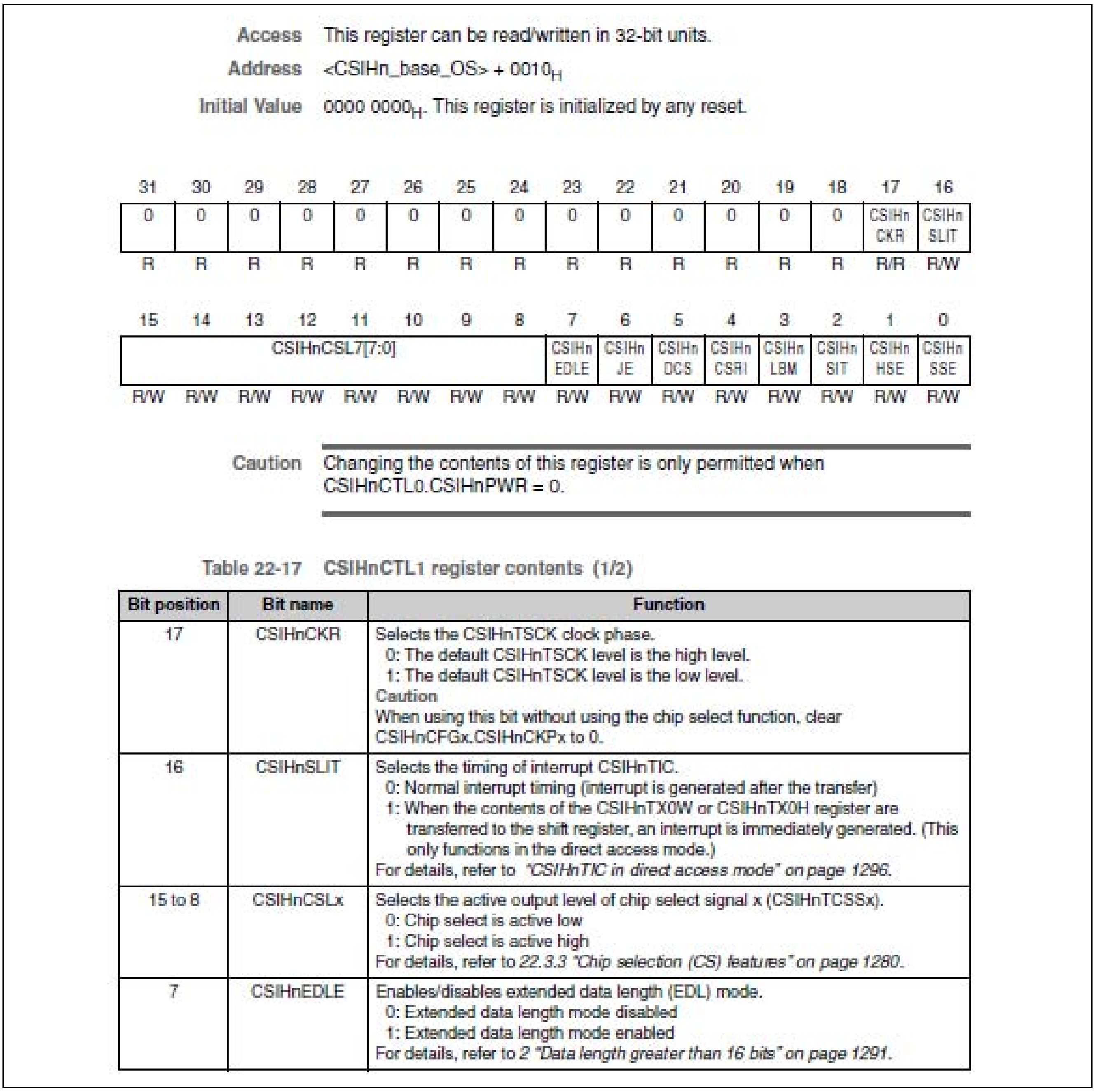
1. When CSIHnPWR = 0, do not change the CSIHnTXE, CSIHnRXE, CSIHnJOBE, or CSIHnMBS bit. However, the CSIHnTXE, CSIHnRXE, or CSIHnMBS bit can be changed at the same time that the CSIHnPWR bit changes from 0 to 1.
2. Do not modify CSIHnTXE or CSIHnRXE or CSIHnMBS while a data transmission is pending or going on, i.e. if CSIHnSTR0.CSIHnTSF = 1.

Figure 4.9 CSIHnCTL0 Register Format (2/2)

```
CSIHnCTL0 = 0x00;           /* stop CSIH0 */
CSIHnPWR = 1;               /* permit CSIHn */
CSIHnTXE = 1;               /* permit transmission */
CSIHnRXE = 1;               /* permit reception */
CSIHnMBS = 0;               /* memory mode */
CSIHnMBS = 1;               /* direct access mode */
CSIHnJOBE = 1;              /* stop communication after this JOB */
```

4.2.4 CSIH Control Register 1 (CSIHnCTL1)

The CSIHnCTL1 register controls the communication. It mainly specifies the interrupt timing and interrupt delay mode and selects the active output level of each chip select signal and the chip select signal operation to perform after the last data is transferred.



Access	This register can be read/written in 32-bit units.
Address	<CSIHn_base_OS> + 0010_H
Initial Value	$0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	CSIHn CKR	CSIHn SLIT
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/R	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnCSL7[7:0]								CSIHn EDLE	CSIHn JE	CSIHn DCS	CSIHn CSRI	CSIHn LBM	CSIHn SIT	CSIHn HSE	CSIHn SSE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Changing the contents of this register is only permitted when CSIHnCTL0.CSIHnPWR = 0.

Table 22-17 CSIHnCTL1 register contents (1/2)

Bit position	Bit name	Function
17	CSIHnCKR	Selects the CSIHnTSCK clock phase. 0: The default CSIHnTSCK level is the high level. 1: The default CSIHnTSCK level is the low level. Caution When using this bit without using the chip select function, clear CSIHnCFGx.CSIHnCKPx to 0.
16	CSIHnSLIT	Selects the timing of interrupt CSIHnTIC. 0: Normal interrupt timing (interrupt is generated after the transfer) 1: When the contents of the CSIHnTX0W or CSIHnTX0H register are transferred to the shift register, an interrupt is immediately generated. (This only functions in the direct access mode.) For details, refer to *"CSIHnTIC in direct access mode" on page 1296.*
15 to 8	CSIHnCSLx	Selects the active output level of chip select signal x (CSIHnTCSSx). 0: Chip select is active low 1: Chip select is active high For details, refer to *22.3.3 "Chip selection (CS) features" on page 1280.*
7	CSIHnEDLE	Enables/disables extended data length (EDL) mode. 0: Extended data length mode disabled 1: Extended data length mode enabled For details, refer to *2 "Data length greater than 16 bits" on page 1291.*

Figure 4.10 CSIHnCTL1 Register Format (1/3)

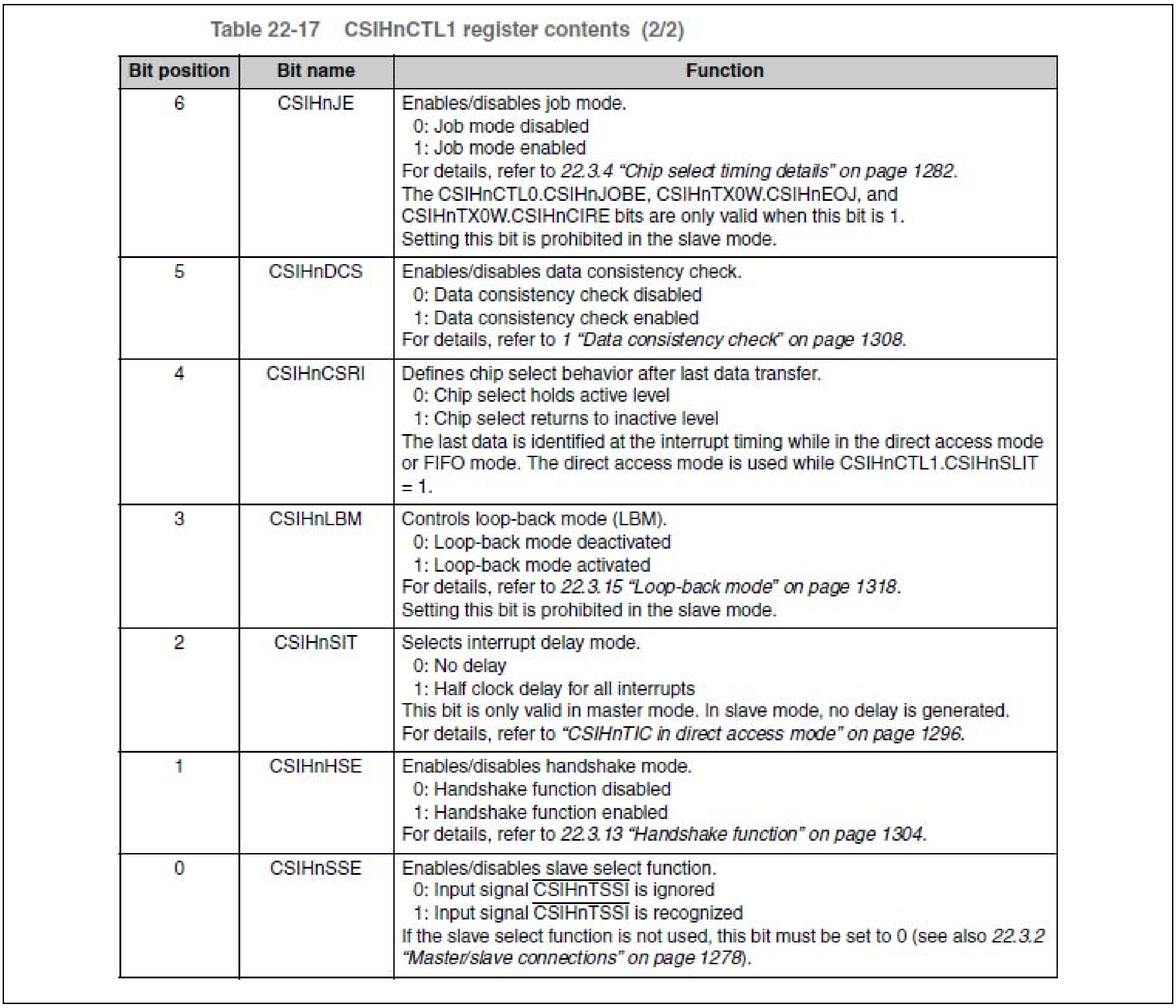
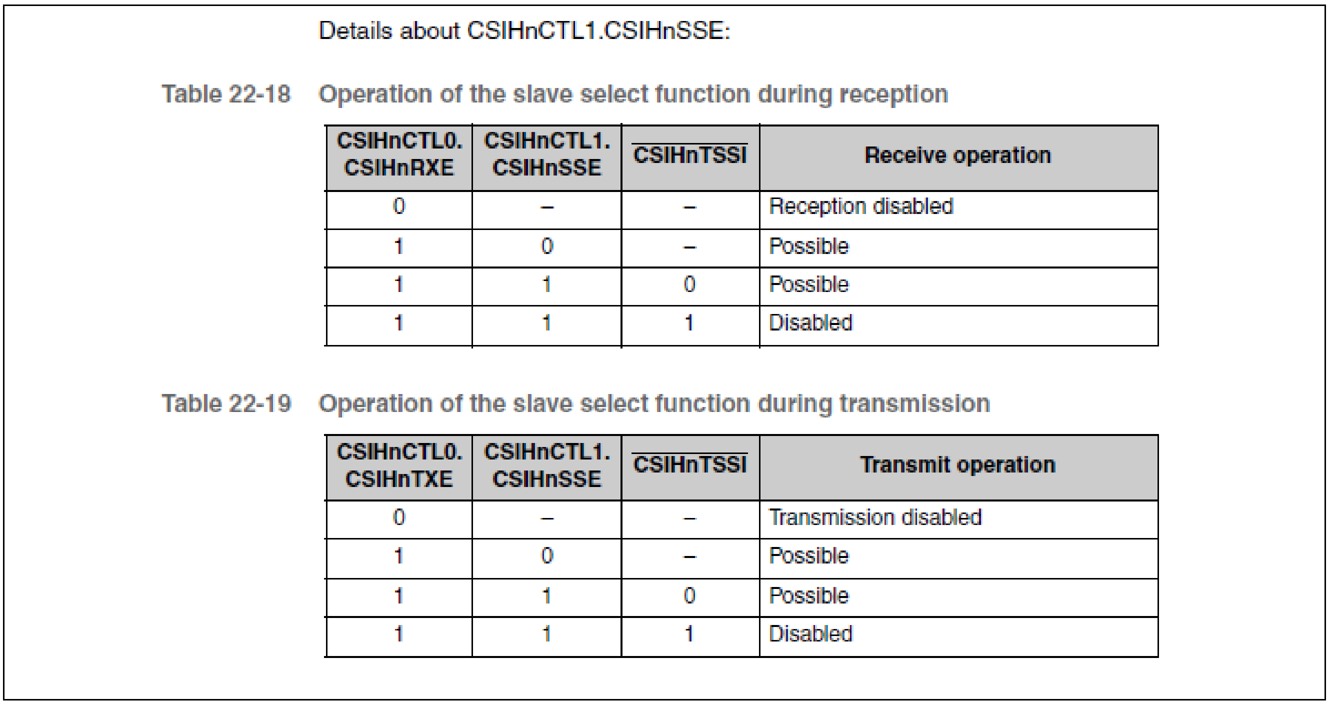


Table 22-17 CSIHnCTL1 register contents (2/2)

Bit position	Bit name	Function
6	CSIHnJE	Enables/disables job mode. 0: Job mode disabled 1: Job mode enabled For details, refer to *22.3.4 "Chip select timing details" on page 1282.* The CSIHnCTL0.CSIHnJOBE, CSIHnTX0W.CSIHnEOJ, and CSIHnTX0W.CSIHnCIRE bits are only valid when this bit is 1. Setting this bit is prohibited in the slave mode.
5	CSIHnDCS	Enables/disables data consistency check. 0: Data consistency check disabled 1: Data consistency check enabled For details, refer to *1 "Data consistency check" on page 1308.*
4	CSIHnCSRI	Defines chip select behavior after last data transfer. 0: Chip select holds active level 1: Chip select returns to inactive level The last data is identified at the interrupt timing while in the direct access mode or FIFO mode. The direct access mode is used while CSIHnCTL1.CSIHnSLIT = 1.
3	CSIHnLBM	Controls loop-back mode (LBM). 0: Loop-back mode deactivated 1: Loop-back mode activated For details, refer to *22.3.15 "Loop-back mode" on page 1318.* Setting this bit is prohibited in the slave mode.
2	CSIHnSIT	Selects interrupt delay mode. 0: No delay 1: Half clock delay for all interrupts This bit is only valid in master mode. In slave mode, no delay is generated. For details, refer to *"CSIHnTIC in direct access mode" on page 1296.*
1	CSIHnHSE	Enables/disables handshake mode. 0: Handshake function disabled 1: Handshake function enabled For details, refer to *22.3.13 "Handshake function" on page 1304.*
0	CSIHnSSE	Enables/disables slave select function. 0: Input signal $\overline{\text{CSIHnTSSI}}$ is ignored 1: Input signal $\overline{\text{CSIHnTSSI}}$ is recognized If the slave select function is not used, this bit must be set to 0 (see also *22.3.2 "Master/slave connections" on page 1278*).

Figure 4.11 CSIHnCTL1 Register Format (2/3)



Details about CSIHnCTL1.CSIHnSSE:

Table 22-18 Operation of the slave select function during reception

CSIHnCTL0. CSIHnRXE	CSIHnCTL1. CSIHnSSE	$\overline{\text{CSIHnTSSI}}$	Receive operation
0	–	–	Reception disabled
1	0	–	Possible
1	1	0	Possible
1	1	1	Disabled

Table 22-19 Operation of the slave select function during transmission

CSIHnCTL0. CSIHnTXE	CSIHnCTL1. CSIHnSSE	$\overline{\text{CSIHnTSSI}}$	Transmit operation
0	–	–	Transmission disabled
1	0	–	Possible
1	1	0	Possible
1	1	1	Disabled

Figure 4.12 CSIHnCTL1 Register Format (3/3)

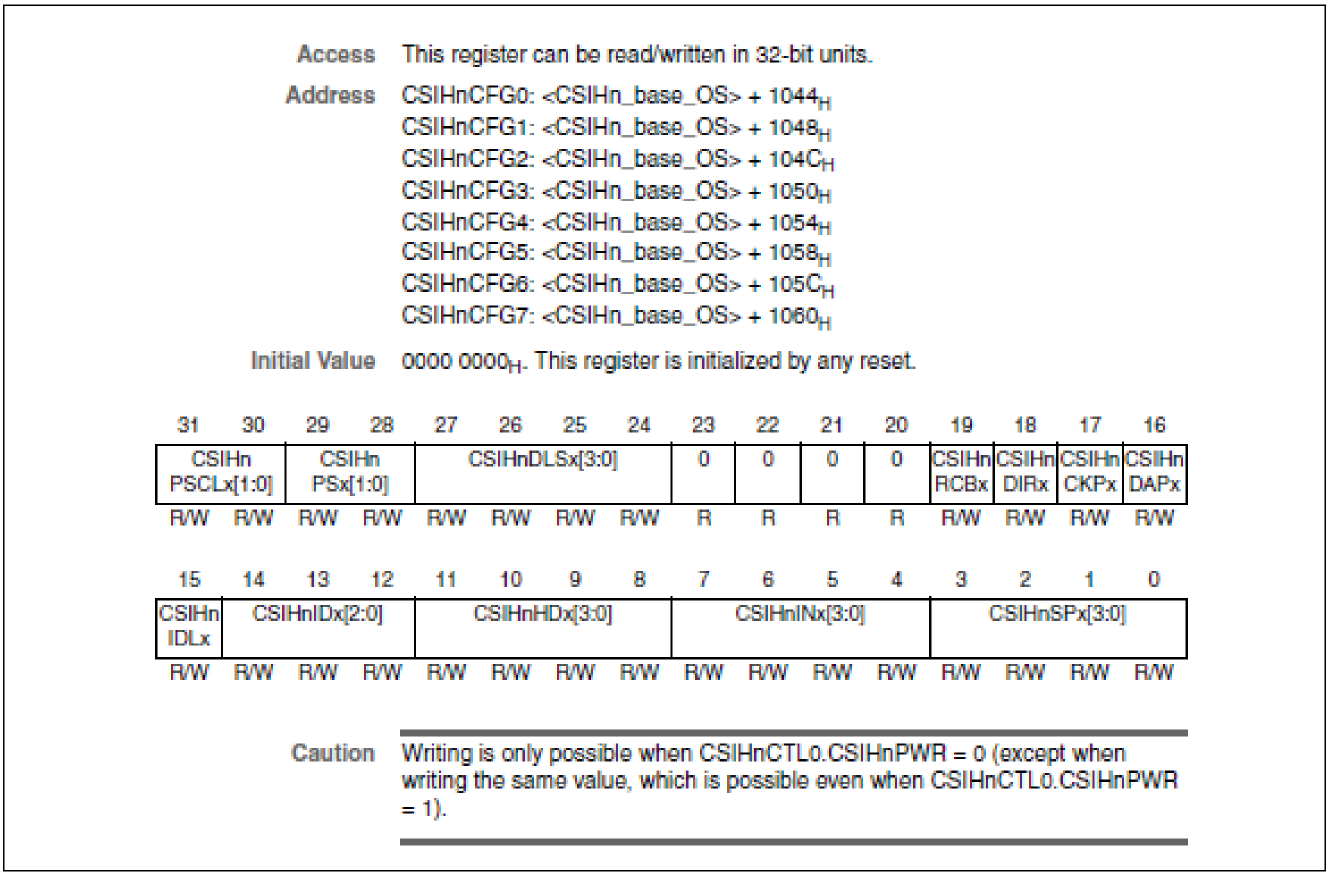
```
CSIHnCTL1 = 0x00010040;        /* TIC at start;CS0 inactive;JOB mode enable */
                               /* Output initial CSIHTSCO value at high level */
                               /* Transmit status interrupt request is generated at beginning of
                               transmission */
                               /* Set chip select signal CS0 to active low*/
                               /* Disable extended data length mode */
                               /* Enable job mode */
                               /* Disable data consistency check */
                               /* Chip select signals retain active level */
                               /* Set loopback mode inactive */
                               /* No interrupt delay mode */
                               /* Disable handshake function */
                               /* Disable slave selection (SS) */
CSIHnCTL1 = 0x00000000;        /* Normal interrupt timing */
                               /* CS0 inactive */
                               /* Disable job mode */
CSIHnCTL1 = 0x00000000;        /* Set chip select signals to active low */
                               /* hand shaking disable */
```

4.2.5 CSIH Configuration Register x (CSIHnCFGx)

The CSIHnCFGx registers specify the prescaler, the parity, the data length, the recessive configuration for broadcasting, the serial data direction, the clock phase and the data phase, the idle enforcement configuration, the idle timing, the hold timing, the inter-data timing, and the setup timing for each chip select signal CSIHCSSx.

In master mode, one or more chip select signals can be used. If several slaves are connected to the master, the chip select signals can be used to address one or more of the slaves. Only a selected slave is then enabled for communication.

A value must be set in the bit for each chip select signal according to the baud rate. In this sample program, these bits are set to initial values.



Access This register can be read/written in 32-bit units.

Address
CSIHnCFG0: <CSIHn_base_OS> + 1044_H
CSIHnCFG1: <CSIHn_base_OS> + 1048_H
CSIHnCFG2: <CSIHn_base_OS> + $104C_H$
CSIHnCFG3: <CSIHn_base_OS> + 1050_H
CSIHnCFG4: <CSIHn_base_OS> + 1054_H
CSIHnCFG5: <CSIHn_base_OS> + 1058_H
CSIHnCFG6: <CSIHn_base_OS> + $105C_H$
CSIHnCFG7: <CSIHn_base_OS> + 1060_H

Initial Value 0000 0000_H. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CSIHn PSCLx[1:0]		CSIHn PSx[1:0]		CSIHnDLSx[3:0]				0	0	0	0	CSIHn RCBx	CSIHn DIRx	CSIHn CKPx	CSIHn DAPx
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHn IDLx	CSIHnIDx[2:0]			CSIHnHDx[3:0]				CSIHnINx[3:0]				CSIHnSPx[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Writing is only possible when CSIHnCTL0.CSIHnPWR = 0 (except when writing the same value, which is possible even when CSIHnCTL0.CSIHnPWR = 1).

Figure 4.13 CSIHnCFGx Register Format (1/5)

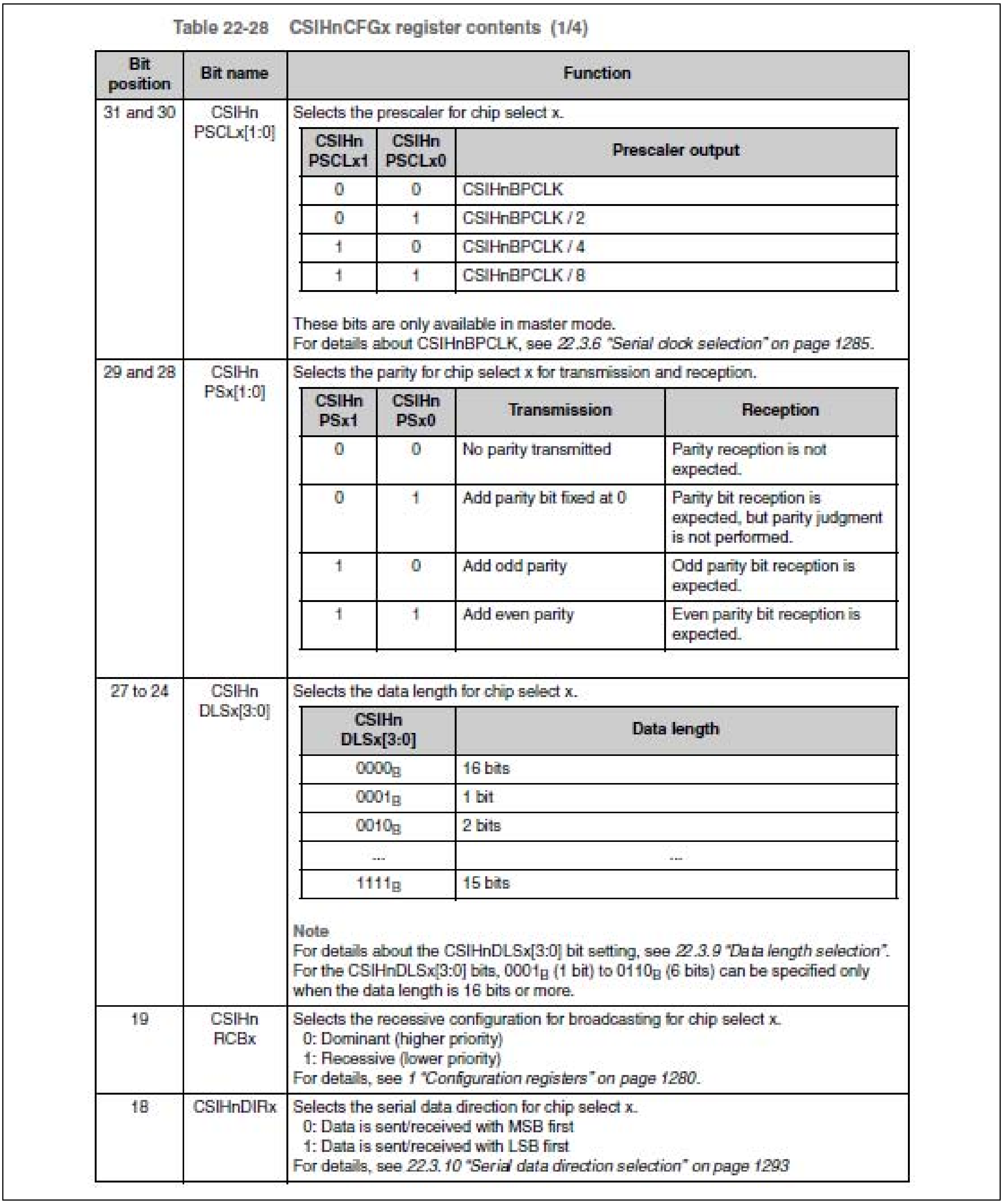


Table 22-28 CSIHnCFGx register contents (1/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>31 and 30</td><td>CSIHn PSCLx[1:0]</td><td>Selects the prescaler for chip select x.
<table>
<tr><th>CSIHn PSCLx1</th><th>CSIHn PSCLx0</th><th>Prescaler output</th></tr>
<tr><td>0</td><td>0</td><td>CSIHnBPCLK</td></tr>
<tr><td>0</td><td>1</td><td>CSIHnBPCLK / 2</td></tr>
<tr><td>1</td><td>0</td><td>CSIHnBPCLK / 4</td></tr>
<tr><td>1</td><td>1</td><td>CSIHnBPCLK / 8</td></tr>
</table>
These bits are only available in master mode.
For details about CSIHnBPCLK, see <i>22.3.6 "Serial clock selection" on page 1285.</i></td></tr>
<tr><td>29 and 28</td><td>CSIHn PSx[1:0]</td><td>Selects the parity for chip select x for transmission and reception.
<table>
<tr><th>CSIHn PSx1</th><th>CSIHn PSx0</th><th>Transmission</th><th>Reception</th></tr>
<tr><td>0</td><td>0</td><td>No parity transmitted</td><td>Parity reception is not expected.</td></tr>
<tr><td>0</td><td>1</td><td>Add parity bit fixed at 0</td><td>Parity bit reception is expected, but parity judgment is not performed.</td></tr>
<tr><td>1</td><td>0</td><td>Add odd parity</td><td>Odd parity bit reception is expected.</td></tr>
<tr><td>1</td><td>1</td><td>Add even parity</td><td>Even parity bit reception is expected.</td></tr>
</table></td></tr>
<tr><td>27 to 24</td><td>CSIHn DLSx[3:0]</td><td>Selects the data length for chip select x.
<table>
<tr><th>CSIHn DLSx[3:0]</th><th>Data length</th></tr>
<tr><td>0000_B</td><td>16 bits</td></tr>
<tr><td>0001_B</td><td>1 bit</td></tr>
<tr><td>0010_B</td><td>2 bits</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1111_B</td><td>15 bits</td></tr>
</table>
<b>Note</b>
For details about the CSIHnDLSx[3:0] bit setting, see <i>22.3.9 "Data length selection"</i>.
For the CSIHnDLSx[3:0] bits, 0001_B (1 bit) to 0110_B (6 bits) can be specified only when the data length is 16 bits or more.</td></tr>
<tr><td>19</td><td>CSIHn RCBx</td><td>Selects the recessive configuration for broadcasting for chip select x.
0: Dominant (higher priority)
1: Recessive (lower priority)
For details, see <i>1 "Configuration registers" on page 1280.</i></td></tr>
<tr><td>18</td><td>CSIHnDIRx</td><td>Selects the serial data direction for chip select x.
0: Data is sent/received with MSB first
1: Data is sent/received with LSB first
For details, see <i>22.3.10 "Serial data direction selection" on page 1293</i></td></tr>
</table>

Figure 4.14 CSIHnCFGx Register Format (2/5)

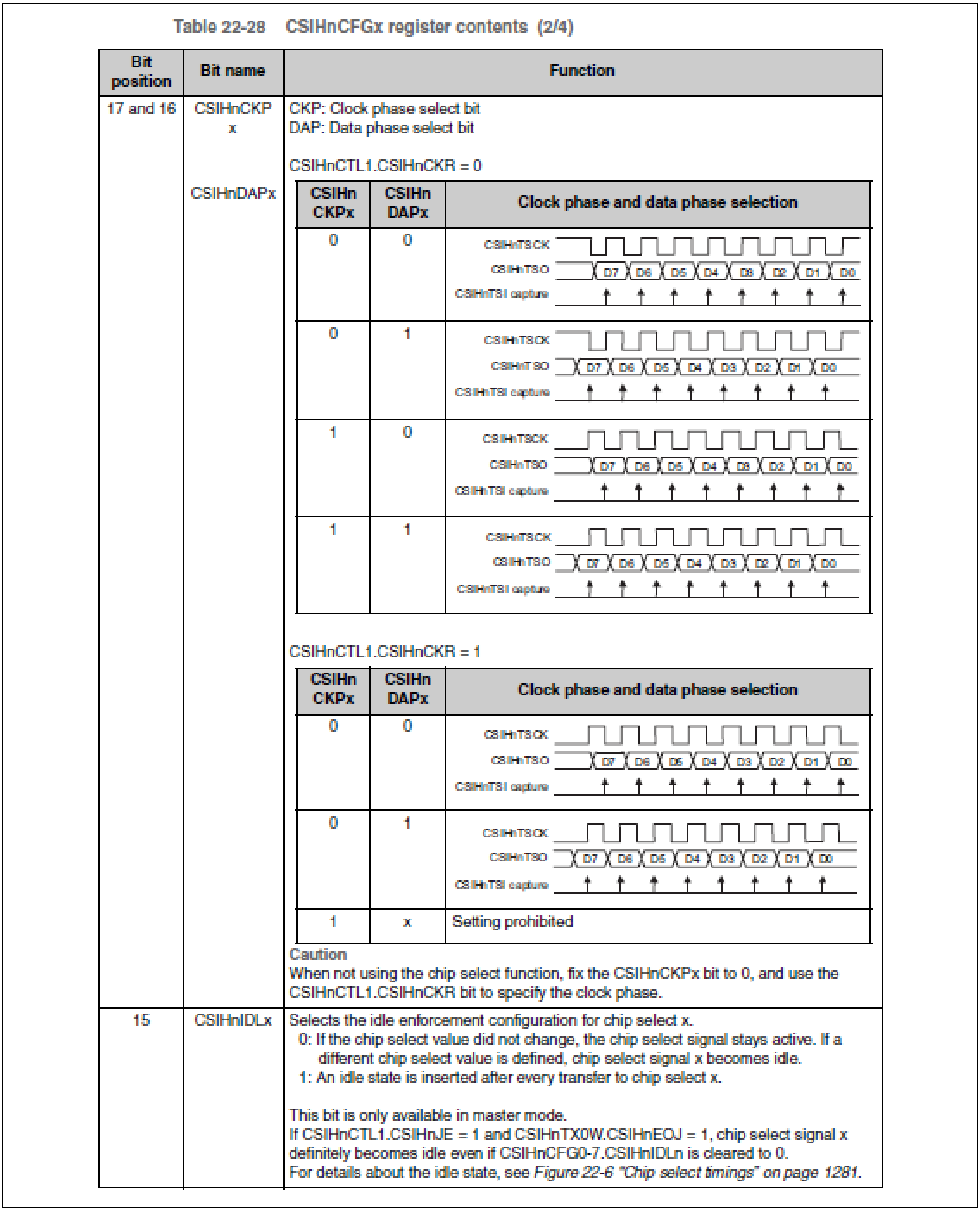


Table 22-28 CSIHnCFGx register contents (2/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>17 and 16</td><td>CSIHnCKPx
CSIHnDAPx</td><td>CKP: Clock phase select bit
DAP: Data phase select bit
CSIHnCTL1.CSIHnCKR = 0
<table>
<tr><th>CSIHn CKPx</th><th>CSIHn DAPx</th><th>Clock phase and data phase selection</th></tr>
<tr><td>0</td><td>0</td><td>

CSIHnTSCK
CSIHnTSO
D7 D6 D5 D4 D3 D2 D1 D0
CSIHnTSI capture

</td></tr>
<tr><td>0</td><td>1</td><td>

CSIHnTSCK
CSIHnTSO
D7 D6 D5 D4 D3 D2 D1 D0
CSIHnTSI capture

</td></tr>
<tr><td>1</td><td>0</td><td>

CSIHnTSCK
CSIHnTSO
D7 D6 D5 D4 D3 D2 D1 D0
CSIHnTSI capture

</td></tr>
<tr><td>1</td><td>1</td><td>

CSIHnTSCK
CSIHnTSO
D7 D6 D5 D4 D3 D2 D1 D0
CSIHnTSI capture

</td></tr>
</table>
CSIHnCTL1.CSIHnCKR = 1
<table>
<tr><th>CSIHn CKPx</th><th>CSIHn DAPx</th><th>Clock phase and data phase selection</th></tr>
<tr><td>0</td><td>0</td><td>

CSIHnTSCK
CSIHnTSO
D7 D6 D5 D4 D3 D2 D1 D0
CSIHnTSI capture

</td></tr>
<tr><td>0</td><td>1</td><td>

CSIHnTSCK
CSIHnTSO
D7 D6 D5 D4 D3 D2 D1 D0
CSIHnTSI capture

</td></tr>
<tr><td>1</td><td>x</td><td>Setting prohibited</td></tr>
</table>
Caution
When not using the chip select function, fix the CSIHnCKPx bit to 0, and use the CSIHnCTL1.CSIHnCKR bit to specify the clock phase.</td></tr>
<tr><td>15</td><td>CSIHnIDLx</td><td>Selects the idle enforcement configuration for chip select x.
0: If the chip select value did not change, the chip select signal stays active. If a different chip select value is defined, chip select signal x becomes idle.
1: An idle state is inserted after every transfer to chip select x.
This bit is only available in master mode.
If CSIHnCTL1.CSIHnJE = 1 and CSIHnTX0W.CSIHnEOJ = 1, chip select signal x definitely becomes idle even if CSIHnCFG0-7.CSIHnIDLn is cleared to 0.
For details about the idle state, see <i>Figure 22-6 "Chip select timings" on page 1281</i>.</td></tr>
</table>

Figure 4.15 CSIHnCFGx Register Format (3/5)

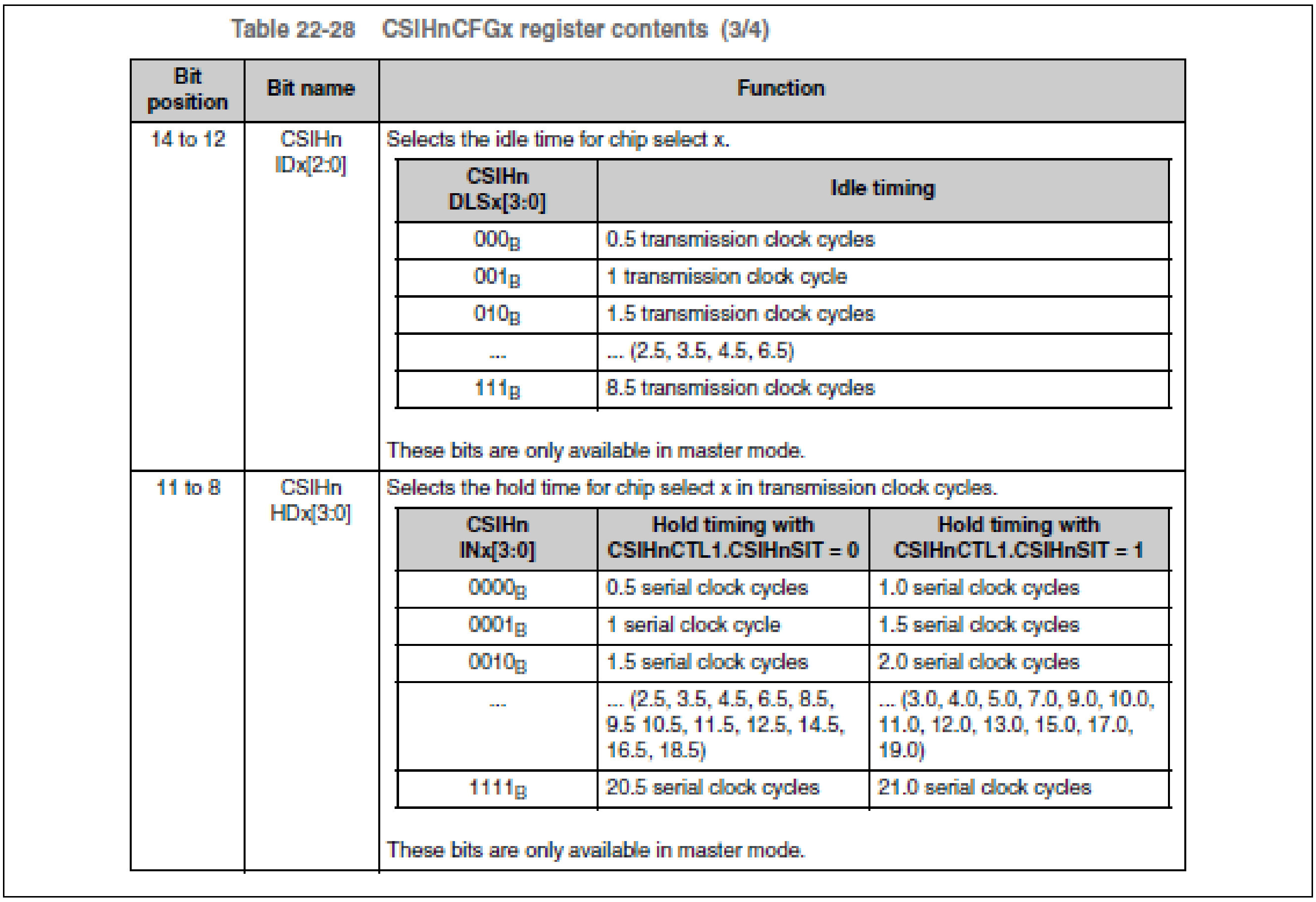


Table 22-28 CSIHnCFGx register contents (3/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>14 to 12</td><td>CSIHn IDx[2:0]</td><td>Selects the idle time for chip select x.
<table>
<tr><th>CSIHn DLSx[3:0]</th><th>Idle timing</th></tr>
<tr><td>000_B</td><td>0.5 transmission clock cycles</td></tr>
<tr><td>001_B</td><td>1 transmission clock cycle</td></tr>
<tr><td>010_B</td><td>1.5 transmission clock cycles</td></tr>
<tr><td>...</td><td>... (2.5, 3.5, 4.5, 6.5)</td></tr>
<tr><td>111_B</td><td>8.5 transmission clock cycles</td></tr>
</table>
These bits are only available in master mode.</td></tr>
<tr><td>11 to 8</td><td>CSIHn HDx[3:0]</td><td>Selects the hold time for chip select x in transmission clock cycles.
<table>
<tr><th>CSIHn INx[3:0]</th><th>Hold timing with CSIHnCTL1.CSIHnSIT = 0</th><th>Hold timing with CSIHnCTL1.CSIHnSIT = 1</th></tr>
<tr><td>0000_B</td><td>0.5 serial clock cycles</td><td>1.0 serial clock cycles</td></tr>
<tr><td>0001_B</td><td>1 serial clock cycle</td><td>1.5 serial clock cycles</td></tr>
<tr><td>0010_B</td><td>1.5 serial clock cycles</td><td>2.0 serial clock cycles</td></tr>
<tr><td>...</td><td>... (2.5, 3.5, 4.5, 6.5, 8.5, 9.5 10.5, 11.5, 12.5, 14.5, 16.5, 18.5)</td><td>... (3.0, 4.0, 5.0, 7.0, 9.0, 10.0, 11.0, 12.0, 13.0, 15.0, 17.0, 19.0)</td></tr>
<tr><td>1111_B</td><td>20.5 serial clock cycles</td><td>21.0 serial clock cycles</td></tr>
</table>
These bits are only available in master mode.</td></tr>
</table>

Figure 4.16 CSIHnCFGx Register Format (4/5)

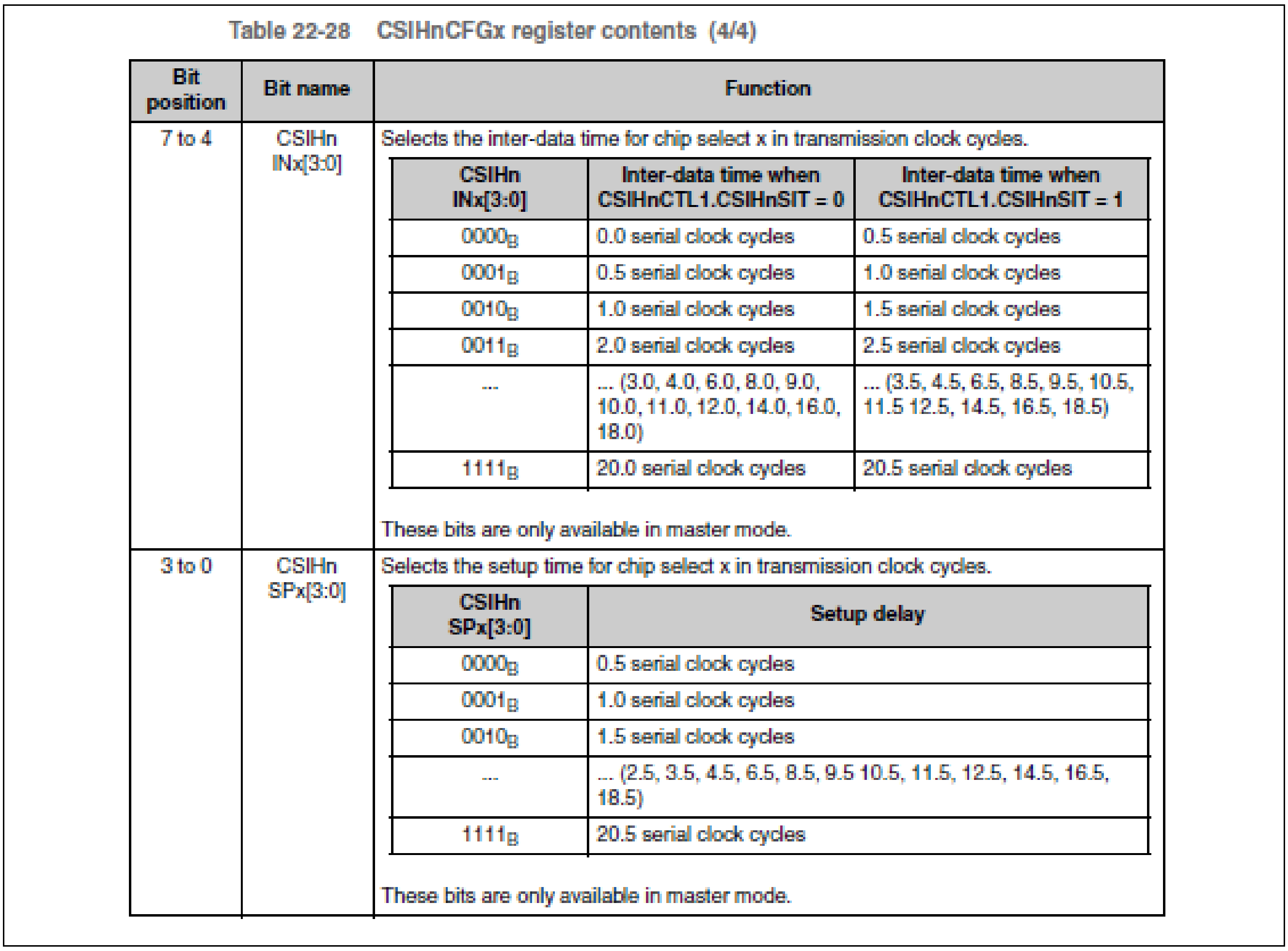


Table 22-28 CSIHnCFGx register contents (4/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7 to 4</td><td>CSIHn INx[3:0]</td><td>Selects the inter-data time for chip select x in transmission clock cycles.
<table>
<tr><th>CSIHn INx[3:0]</th><th>Inter-data time when CSIHnCTL1.CSIHnSIT = 0</th><th>Inter-data time when CSIHnCTL1.CSIHnSIT = 1</th></tr>
<tr><td>0000_B</td><td>0.0 serial clock cycles</td><td>0.5 serial clock cycles</td></tr>
<tr><td>0001_B</td><td>0.5 serial clock cycles</td><td>1.0 serial clock cycles</td></tr>
<tr><td>0010_B</td><td>1.0 serial clock cycles</td><td>1.5 serial clock cycles</td></tr>
<tr><td>0011_B</td><td>2.0 serial clock cycles</td><td>2.5 serial clock cycles</td></tr>
<tr><td>...</td><td>... (3.0, 4.0, 6.0, 8.0, 9.0, 10.0, 11.0, 12.0, 14.0, 16.0, 18.0)</td><td>... (3.5, 4.5, 6.5, 8.5, 9.5, 10.5, 11.5 12.5, 14.5, 16.5, 18.5)</td></tr>
<tr><td>1111_B</td><td>20.0 serial clock cycles</td><td>20.5 serial clock cycles</td></tr>
</table>
These bits are only available in master mode.</td></tr>
<tr><td>3 to 0</td><td>CSIHn SPx[3:0]</td><td>Selects the setup time for chip select x in transmission clock cycles.
<table>
<tr><th>CSIHn SPx[3:0]</th><th>Setup delay</th></tr>
<tr><td>0000_B</td><td>0.5 serial clock cycles</td></tr>
<tr><td>0001_B</td><td>1.0 serial clock cycles</td></tr>
<tr><td>0010_B</td><td>1.5 serial clock cycles</td></tr>
<tr><td>...</td><td>... (2.5, 3.5, 4.5, 6.5, 8.5, 9.5 10.5, 11.5, 12.5, 14.5, 16.5, 18.5)</td></tr>
<tr><td>1111_B</td><td>20.5 serial clock cycles</td></tr>
</table>
These bits are only available in master mode.</td></tr>
</table>

Figure 4.17 CSIHnCFGx Register Format (5/5)

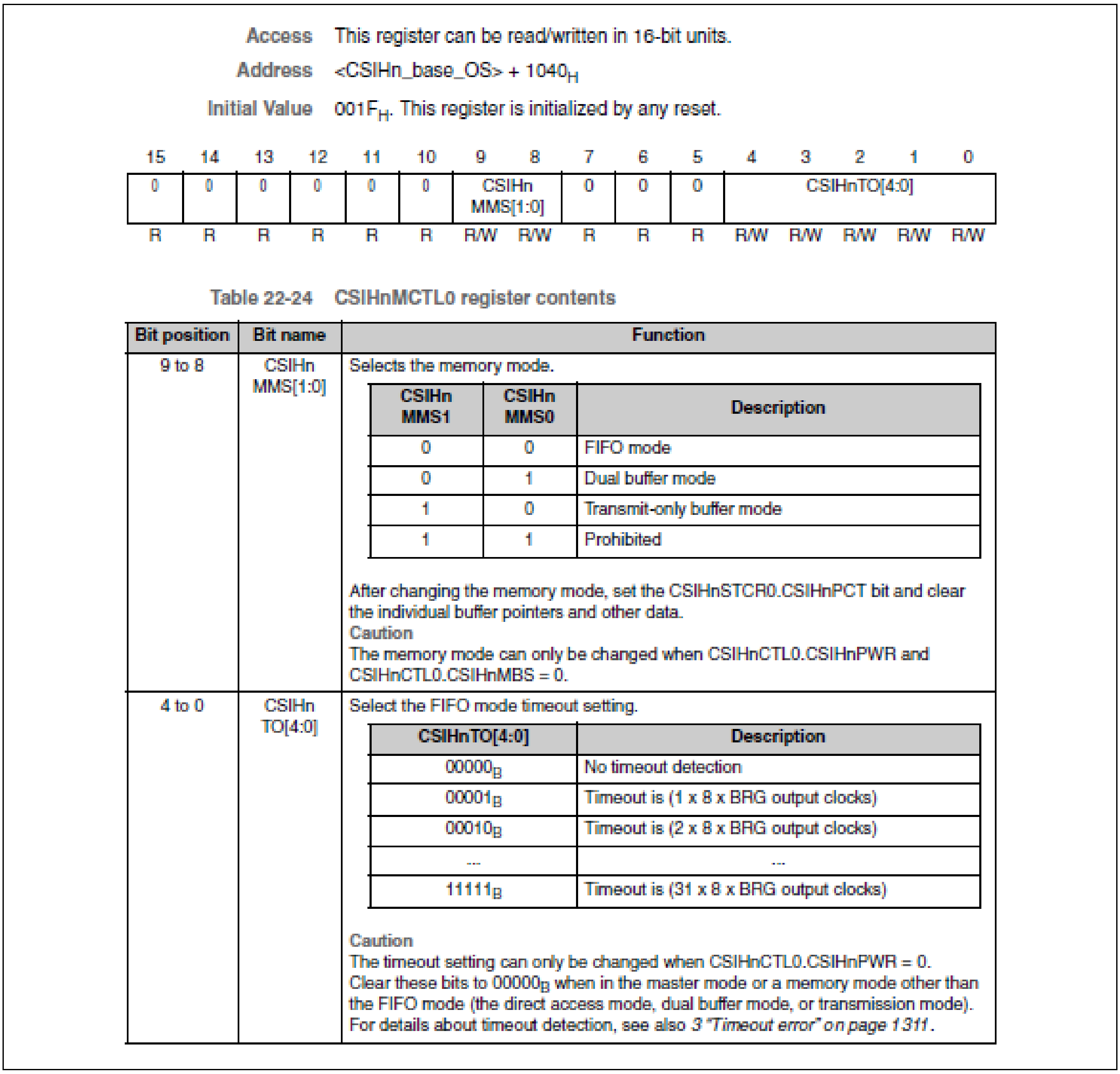
```
CSIHnCFG0 = 0x08000000;         /* No parity */
                                /* data length 8 bits */
                                /* Recessive configuration: Dominant (higher priority) */
                                /* MSB first */
```

4.2.6 CSIH Memory Control Register 0 (CSIHnMCTL0)

The CSIHnMCTL0 register selects the memory mode and timeout setting.

FIFO mode, dual buffer mode, transmit-only buffer mode, and direct access mode can be set in the CSIH as memory modes.

The sample program uses only dual buffer mode. It does not detect timeout time.



Access This register can be read/written in 16-bit units.

Address <CSIHn_base_OS> + 1040_H

Initial Value $001F_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	CSIHn MMS[1:0]		0	0	0	CSIHnTO[4:0]				
R	R	R	R	R	R	R/W	R/W	R	R	R	R/W	R/W	R/W	R/W	R/W

Table 22-24 CSIHnMCTL0 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>9 to 8</td><td>CSIHn MMS[1:0]</td><td>Selects the memory mode.
<table>
<tr><th>CSIHn MMS1</th><th>CSIHn MMS0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>FIFO mode</td></tr>
<tr><td>0</td><td>1</td><td>Dual buffer mode</td></tr>
<tr><td>1</td><td>0</td><td>Transmit-only buffer mode</td></tr>
<tr><td>1</td><td>1</td><td>Prohibited</td></tr>
</table>
After changing the memory mode, set the CSIHnSTCR0.CSIHnPCT bit and clear the individual buffer pointers and other data.
Caution
The memory mode can only be changed when CSIHnCTL0.CSIHnPWR and CSIHnCTL0.CSIHnMBS = 0.</td></tr>
<tr><td>4 to 0</td><td>CSIHn TO[4:0]</td><td>Select the FIFO mode timeout setting.
<table>
<tr><th>CSIHnTO[4:0]</th><th>Description</th></tr>
<tr><td>00000_B</td><td>No timeout detection</td></tr>
<tr><td>00001_B</td><td>Timeout is (1 x 8 x BRG output clocks)</td></tr>
<tr><td>00010_B</td><td>Timeout is (2 x 8 x BRG output clocks)</td></tr>
<tr><td>---</td><td>---</td></tr>
<tr><td>11111_B</td><td>Timeout is (31 x 8 x BRG output clocks)</td></tr>
</table>
Caution
The timeout setting can only be changed when CSIHnCTL0.CSIHnPWR = 0.
Clear these bits to 00000_B when in the master mode or a memory mode other than the FIFO mode (the direct access mode, dual buffer mode, or transmission mode).
For details about timeout detection, see also <i>3 "Timeout error" on page 1311</i>.</td></tr>
</table>

Figure 4.18 CSIHnMCTL0 Register Format

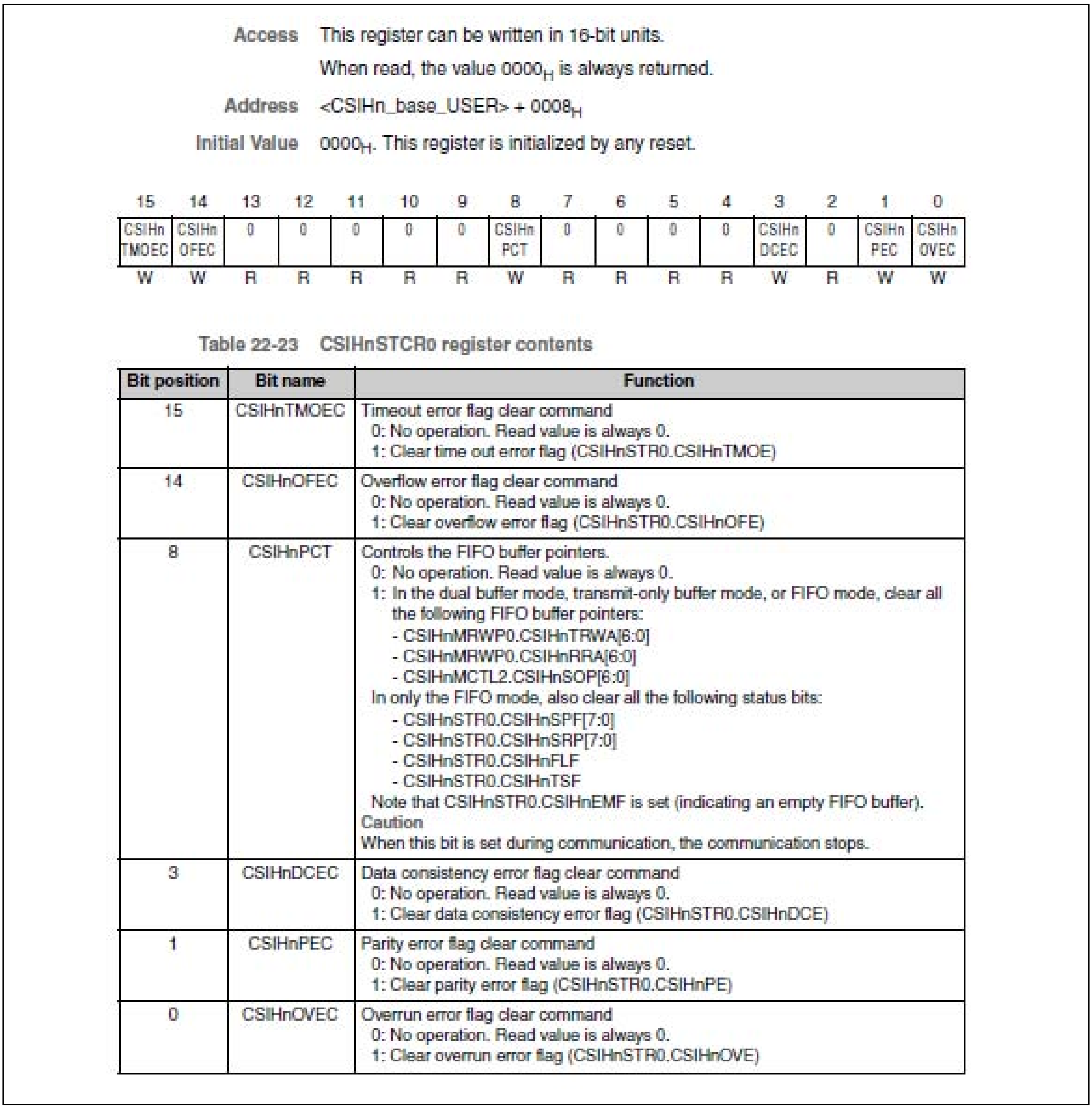
```
CSIH0MCTL0 = 0x0100;                    /* dual buffer mode; no timeout detection */
```

4.2.7 CSIH Status Clear Register 0 (CSIHnSTCR0)

The CSIH can detect five errors: data consistency error, parity error, overrun error, timeout error, and overflow error.

The parity error, data consistency error, and timeout error can be individually enabled or disabled by the CSIHnSTCR0 register. When any of these errors is detected, receive error interrupt CSIHTIRE is generated.

In this sample program, when receive error interrupt CSIHTIRE is detected, the relevant error flags are cleared by setting each bit in the status clear register to 1.



Access This register can be written in 16-bit units.
When read, the value 0000_H is always returned.

Address <CSIHn_base_USER> + 0008_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHn TMOEC	CSIHn OFEC	0	0	0	0	0	CSIHn PCT	0	0	0	0	CSIHn DCEC	0	CSIHn PEC	CSIHn OVEC
W	W	R	R	R	R	R	W	R	R	R	R	W	R	W	W

Table 22-23 CSIHnSTCR0 register contents

Bit position	Bit name	Function
15	CSIHnTMOEC	Timeout error flag clear command 0: No operation. Read value is always 0. 1: Clear time out error flag (CSIHnSTR0.CSIHnTMOE)
14	CSIHnOFEC	Overflow error flag clear command 0: No operation. Read value is always 0. 1: Clear overflow error flag (CSIHnSTR0.CSIHnOFE)
8	CSIHnPCT	Controls the FIFO buffer pointers. 0: No operation. Read value is always 0. 1: In the dual buffer mode, transmit-only buffer mode, or FIFO mode, clear all the following FIFO buffer pointers: - CSIHnMRWP0.CSIHnTRWA[6:0] - CSIHnMRWP0.CSIHnRRA[6:0] - CSIHnMCTL2.CSIHnSOP[6:0] In only the FIFO mode, also clear all the following status bits: - CSIHnSTR0.CSIHnSPF[7:0] - CSIHnSTR0.CSIHnSRP[7:0] - CSIHnSTR0.CSIHnFLF - CSIHnSTR0.CSIHnTSF Note that CSIHnSTR0.CSIHnEMF is set (indicating an empty FIFO buffer). Caution When this bit is set during communication, the communication stops.
3	CSIHnDCEC	Data consistency error flag clear command 0: No operation. Read value is always 0. 1: Clear data consistency error flag (CSIHnSTR0.CSIHnDCE)
1	CSIHnPEC	Parity error flag clear command 0: No operation. Read value is always 0. 1: Clear parity error flag (CSIHnSTR0.CSIHnPE)
0	CSIHnOVEC	Overrun error flag clear command 0: No operation. Read value is always 0. 1: Clear overrun error flag (CSIHnSTR0.CSIHnOVE)

Figure 4.19 CSIHnSTCR0 Register Format

```
/* Clear status flags to 0 */
CSIHnTMOEC = 1;                  /* Clear timeout error flag */
CSIHnOFEC = 1;                   /* Clear overflow error flag */
CSIHnPEC = 1;                    /* Clear parity error flag */
CSIHnOVEC = 1;                   /* Clear overrun error flag */
CSIHnPCT = 1;                    /* Clear FIFO buffer pointer */
CSIHnDCEC = 1;                   /* Clear data consistency error flag */
```

4.3 Memory Modes

The CSIH supports FIFO mode, dual buffer mode, transmit-only buffer mode, and direct access mode as memory modes. The memory mode can be changed by resetting CSIHnMCTL0.CSIHnMMS[1:0]. The conditions for starting CSIH data transfer and the interrupt timing depends on the memory mode, the operation mode, and the transfer mode.

Table 4.1 Start of Data Transfer

Memory Mode and Operating Mode		**Transfer Mode**	
		Transmit-Only and Transmit/Receive	**Receive**
FIFO mode Direct access mode	Master	Writes to the CSIHnTX0 register	Writes to the CSIHnTX0 register
	Slave	Writes to the CSIHnTX0 register and starts the master clock	Receives a clock from the master
Transmit-only buffer mode	Master	Sets CSIHnMCTL2.BTST to 1	Sets CSIHnMCTL2.BTST to 1
	Slave	Sets CSIHnMCTL2.BTST to 1 and starts the master clock	Receives a clock from the master
Dual buffer mode	Master	Sets CSIHnMCTL2.BTST to 1	Sets CSIHnMCTL2.BTST to 1
	Slave	Sets CSIHnMCTL2.BTST to 1 and starts the master clock	Receives a clock from the master

4.4 Function Specifications

This section describes the specifications for the functions that are used by the sample program.

4.4.1 Main Processing (main.c)

[Function Name]	main ()
[Function]	Calls necessary initialization functions before entering an infinite loop.
[Arguments]	None
[Return Value]	None
[Startup Method]	Enters the main function after hardware initialization.
[SFRs Used]	None
[Calling Function]	None
[Variables]	flag_mode, flag_transmit_over, flag_receive_over, flag_error
[File Name]	main.c
[Notes]	None

4.4.2 Software Initialization Processing (initial.c)

[Function Name]	port_initial()
[Function]	Sets up ports and their mode.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	PFCE4, PFC4, PMC4, PM4, PFCE13, PFC13, PMC13, PM13
[Calling Function]	main()
[Variables]	None
[File Name]	initial.c
[Notes]	None

[Function Name]	cg_initial()
[Function]	Initializes the special clock frequency control register.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	SFRCTL3
[Calling Function]	main()
[Variables]	None
[File Name]	initial.c
[Notes]	None

[Function Name]	hbus_initial()
[Function]	Initializes the AHB bus.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	ETARCFG0, ETARADRS0, ETARMASK0
[Calling Function]	main()
[Variables]	None
[File Name]	initial.c
[Notes]	None

[Function Name]	board_initial()
[Function]	Sets up the initial state of the LEDs.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	P13
[Calling Function]	main()
[Variables]	None
[File Name]	initial.c
[Notes]	None

[Function Name]	ram_initial()
[Function]	Sets up the initial states of the receive buffer and flags.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	None
[Calling Function]	main()
[Variables]	buf_receive[], flag_transmit_over, flag_receive_over, flag_job_transmit, flag_error, flag_fifo_error, count, LED, point_receive, point_transmit
[File Name]	initial.c
[Notes]	None

[Function Name]	wait()
[Function]	Waits for a certain number of steps.
[Arguments]	int number
[Return Value]	None
[Startup Method]	Call according to the an argument setting.
[SFRs Used]	None
[Calling Function]	main(), csih_transmit_1_start()
[Variables]	None
[File Name]	initial.c
[Notes]	None

[Function Name]	display()
[Function]	Controls the LEDs according to the state of the relevant flags.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	P13
[Calling Function]	main()
[Variables]	flag_transmit_over, flag_receive_over, flag_error
[File Name]	initial.c
[Notes]	None

[Function Name]	clear_receive_buffer ()
[Function]	Clears receive buffer to 0.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	None
[Calling Function]	csih_receive_start ()
[Variables]	point_receive, buf_receive
[File Name]	initial.c
[Notes]	None

4.4.3 Receive Processing (csih_receive.c)

[Function Name]	csih_receive_initial()
[Function]	Selects the subroutine according to the communication mode flags.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	None
[Calling Function]	main()
[Variables]	flag_mode
[File Name]	csih_receive.c
[Notes]	None

[Function Name]	csih_receive_1_initial()
[Function]	The CSIH3 macro performs initialization in direct-access receive-only mode.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	CSIH3CTL0, CSIH3CTL1, CSIH3CTL2, CSIH3CFG0, ICCSIH3IR
[Calling Function]	csih_receive_initial()
[Variables]	None
[File Name]	csih_receive.c
[Notes]	None

[Function Name]	csih_receive_2_initial()
[Function]	The CSIH3 macro performs initialization in dual-buffer receive-only mode.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	CSIH3CTL0, CSIH3CTL1, CSIH3CTL2, CSIH3CFG0, ICCSIH3IR
[Calling Function]	csih_receive_initial()
[Variables]	None
[File Name]	csih_receive.c
[Notes]	None

[Function Name]	csih_receive_start ()
[Function]	The CSIH3 macro performs reception in dual-buffer receive-only mode.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	CSIH3PWR, CSIH3MCTL2
[Calling Function]	main()
[Variables]	None
[File Name]	csih_receive.c
[Notes]	None

4.4.4 Transmit Processing (csih_transmit.c)

[Function Name]	csih_transmit_initial()
[Function]	Selects the subroutine according to the communication mode flags.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	None
[Calling Function]	main()
[Variables]	flag_mode
[File Name]	csih_transmit.c
[Notes]	None

[Function Name]	csih_transmit_1_initial()
[Function]	The CSIH0 macro performs initialization in direct-access transmit-only mode.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	CSIH0CTL0, CSIH0CTL1, CSIH0CTL2, CSIH0CFG0, ICCSIH0IC, ICCSIH0IJC
[Calling Function]	csih_transmit_initial()
[Variables]	None
[File Name]	csih_transmit.c
[Notes]	None

[Function Name]	csih_transmit_2_initial()
[Function]	The CSIH0 macro performs initialization in dual-buffer transmit-only mode.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	CSIH0CTL0, CSIH0CTL1, CSIH0CTL2, CSIH0CFG0, ICCSIH0IC, CSIH0MCTL0, CSIH0MCTL2
[Calling Function]	csih_transmit_initial()
[Variables]	None
[File Name]	csih_transmit.c
[Notes]	None

[Function Name]	csih_transmit_start()
[Function]	Selects the subroutine according to the communication mode flags.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	None
[Calling Function]	main()
[Variables]	flag_mode
[File Name]	csih_transmit.c
[Notes]	None

[Function Name]	csih_transmit_1_start()
[Function]	The CSIH0 macro performs transmission in direct-access transmit-only mode.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	CSIH0CTL0, CSIH3CTL0, CSIH0TX0W
[Calling Function]	csih_transmit_start()
[Variables]	flag_job_transmit
[File Name]	csih_transmit.c
[Notes]	None

[Function Name]	csih_transmit_2_start()
[Function]	The CSIH0 macro performs transmission in dual-buffer transmit-only mode.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	CSIH0CTL0, CSIH3CTL0, CSIH0TX0W, CSIH0TX0W, CSIH0MCTL2
[Calling Function]	csih_transmit_start()
[Variables]	point_transmit
[File Name]	csih_transmit.c
[Notes]	None

4.4.5 Interrupt Processing (interrupt.c)

[Function Name]	int_csih0ic()
[Function]	Processes CSIH0 macro transmit status interrupt.
[Arguments]	None
[Return Value]	None
[Startup Method]	Request CSIH0TIC is present in an unmasked state.
[SFRs Used]	CSIH0CTL0
[Calling Function]	None
[Variables]	flag_mode, flag_job_transmit, flag_transmit_over
[File Name]	interrupt.c
[Notes]	None

[Function Name]	I int_csih0ijc()
[Function]	Processes CSIH0 macro job completion interrupt.
[Arguments]	None
[Return Value]	None
[Startup Method]	Request CSIH0TIJC is present in an unmasked state.
[SFRs Used]	CSIH0CTL0
[Calling Function]	None
[Variables]	flag_mode, flag_transmit_over
[File Name]	interrupt.c
[Notes]	None

[Function Name]	int_csih0ire()
[Function]	Processes CSIH0 macro communication error interrupt.
[Arguments]	None
[Return Value]	None
[Startup Method]	Request CSIH0TIRE is present in an unmasked state.
[SFRs Used]	CSIH0STCR0, CSIH0CTL0
[Calling Function]	None
[Variables]	flag_error, point_receive, point_transmit
[File Name]	interrupt.c
[Notes]	None

[Function Name]	int_csih3ir()
[Function]	Processes CSIH3 macro receive status interrupt.
[Arguments]	None
[Return Value]	None
[Startup Method]	Request CSIH3TIR is present in an unmasked state.
[SFRs Used]	CSIH3RX0W, CSIH3CTL0
[Calling Function]	None
[Variables]	flag_mode, point_receive, count, flag_receive_over
[File Name]	interrupt.c
[Notes]	None

[Function Name]	int_csih3ire()
[Function]	Processes CSIH3 macro communication error interrupt.
[Arguments]	None
[Return Value]	None
[Startup Method]	Request CSIH3TIRE is present in an unmasked state.
[SFRs Used]	CSIH3STCR0, CSIH3CTL0
[Calling Function]	None
[Variables]	flag_error, point_receive, point_transmit, count
[File Name]	interrupt.c
[Notes]	None

Website and Support

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

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

All trademarks and registered trademarks are the property of their respective owners.

Revision Record

		Description	
Rev.	Date	Page	Summary
1.00	Feb 10, 2012	—	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. 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: +49-211-65030, 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.
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