Abstract

In I²C communications, the slave device may fix the SDA pin to “L” due to a noise and others, which disturbs communications. For recovery, inputting clock to the slave using the measure other than I²C communication is required. This application note gives information on how to input clock to the slave using the clocked synchronous serial format of the I²C bus interface 3 (hereinafter called IIC3).

Products

SH7263/SH7203 (hereinafter called “SH7263”)

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.
Contents

1. Specifications .................................................................................................................................... 3

2. Operation Confirmation Conditions ................................................................................................... 4

3. Reference Application Note .............................................................................................................. 4

4. Peripheral Functions .......................................................................................................................... 5
   4.1 Clocked Synchronous Serial Format ........................................................................................... 5
   4.2 Note on Master Receive Mode .................................................................................................... 6
   4.3 Note on Setting ACKBT in Master Receive Mode ....................................................................... 6
   4.4 Note on Issuing A Stop Condition or Re-transmitting A Start Condition in Master Receive Mode
   6
   4.5 Note on Using the IICRST ........................................................................................................... 6
   4.6 Note on Issuing A Stop Condition in Master Transmit Mode ....................................................... 6

5. Hardware........................................................................................................................................... 7
   5.1 Used Pins and the Pin Functions ................................................................................................. 7
   5.2 Hardware Configuration............................................................................................................... 7

6. Software ............................................................................................................................................ 8
   6.1 Operation Overview ..................................................................................................................... 8
       6.1.1 Sequential Read Operation .................................................................................................. 8
       6.1.2 Detection of Bus Occupation ................................................................................................. 8
       6.1.3 Procedure of Bus Cancellation ............................................................................................. 8
   6.2 File Composition .......................................................................................................................... 9
   6.3 Constants..................................................................................................................................... 9
   6.4 Variables .................................................................................................................................... 10
   6.5 Functions ................................................................................................................................... 10
   6.6 Function Specifications .............................................................................................................. 10
   6.7 Flowchart.................................................................................................................................... 13
       6.7.1 Main Function ..................................................................................................................... 13
       6.7.2 IIC3 Module Initialization..................................................................................................... 14
       6.7.3 Function for Data Read from the EEPROM ........................................................................ 15
       6.7.4 Function for Data Read ....................................................................................................... 16
       6.7.5 Function for Transmitting Slave Device Address .................................................................. 18
       6.7.6 Function for Data Transmission ....................................................................................... 19
       6.7.7 Function for Recovery from Bus Occupation State ............................................................ 20

7. Sample Code................................................................................................................................... 21

8. Reference Documents....................................................................................................................... 21
1. Specifications

Operate the SH7263 in IIC3 master mode. Connect the EEPROM to the slave, then perform a 10-byte data read. When a hang-up is caused to the IIC3 by EEPROM bus occupation, transmit a dummy clock to recover from the failure and restart the data read.

When an 8-bit dummy clock is input to the EEPROM that is occupying the bus, the EEPROM will transit to the state that received the NACK from the master device, and releases the bus. In the sample code, the same process is followed; an 8-bit clock is transmitted as a dummy clock.

For the dummy clock transmission, use the IIC3 clocked synchronous serial format since the PB6/SDA pin on the SH7263 cannot be set to the output port.

Table 1.1 shows the peripheral function and its application. Figure 1.1 shows an example of the flow till a bus occupation failure is caused by the slave.

<table>
<thead>
<tr>
<th>Peripheral Function</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>I2C bus interface 3</td>
<td>• I2C bus format</td>
</tr>
<tr>
<td></td>
<td>Access to the EEPROM</td>
</tr>
<tr>
<td></td>
<td>• Clocked synchronous serial format</td>
</tr>
<tr>
<td></td>
<td>Dummy clock transmission</td>
</tr>
</tbody>
</table>

Table 1.1 Peripheral Functions and Their Applications

Figure 1.1 Flow till Bus Occupation Failure Occurred by the Slave (Example)
2. Operation Confirmation Conditions

The sample code accompanying this application note has been run and confirmed under the conditions below.

<table>
<thead>
<tr>
<th>Item</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCU used</td>
<td>SH7263</td>
</tr>
<tr>
<td>Device used</td>
<td>Renesas Electronics Corporation</td>
</tr>
<tr>
<td></td>
<td>EEPROM</td>
</tr>
<tr>
<td></td>
<td>Model: R1EX24128ASA00A</td>
</tr>
<tr>
<td>Operating frequency</td>
<td>• CPU internal clock (Iφ): 200MHz</td>
</tr>
<tr>
<td></td>
<td>• Internal clock (Bφ): 66.66MHz</td>
</tr>
<tr>
<td></td>
<td>• Peripheral clock (Pφ): 33.33MHz</td>
</tr>
<tr>
<td>Operating voltage</td>
<td>• Source power (I/O): 3.3V</td>
</tr>
<tr>
<td></td>
<td>• Source power (internal): 1.2V</td>
</tr>
<tr>
<td>Integrated development</td>
<td>Renesas Electronics Corporation</td>
</tr>
<tr>
<td>environment</td>
<td>High-performance Embedded Workshop Ver.4.03.00</td>
</tr>
<tr>
<td>C compiler</td>
<td>Renesas Electronics Corporation</td>
</tr>
<tr>
<td></td>
<td>SuperH RISC engine FamilyC/C++ Compiler Package Ver.9.01 Release01</td>
</tr>
<tr>
<td>Complier option</td>
<td>-cpu=sh2afpu -fpu=single -include=&quot;$(WORKSPDIR)\inc&quot;</td>
</tr>
<tr>
<td></td>
<td>-object=&quot;$(CONFIGDIR)$(FILELEAF).obj&quot; -debug -gbr=auto</td>
</tr>
<tr>
<td></td>
<td>-chgincpath -errorpath -global_volatile=0 -opt_range=all</td>
</tr>
<tr>
<td></td>
<td>-infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1 -nologo</td>
</tr>
<tr>
<td>Sample code version</td>
<td>2.00</td>
</tr>
</tbody>
</table>

3. Reference Application Note

For additional information associated with this document, refer to the following application note.

- SH7263/SH7203 Group Reception by the I2C Bus Interface 3 Module in Single-Master Operation (EEPROM Reading) (document No.: REJ06B0838)
4. Peripheral Functions

This section gives the precautions for the IIC3 and the supplemental information on the receive setting for the clocked synchronous serial format used in the sample code. For the basic information, refer to the SH7263 Group Hardware Manual.

4.1 Clocked Synchronous Serial Format

Figure 4.1 shows the setting for the clocked synchronous serial format in master receive mode.
4.2  Note on Master Receive Mode
Reading the I2C bus receive data register (ICDRR) around the falling edge of the 8th clock may fail to fetch the receive data.

In addition, when the receive disable bit (RCVD) in the I2C bus controller register is set to 1 around the falling edge of the 8th clock and the receive buffer is full, a stop condition may not be issued. Use either 1 or 2 below against the situations above.

1. In master receive mode, read the ICDRR before the rising edge of the 8th clock.
2. In master receive mode, set the RCVD bit to so that transfer proceeds in byte unit.

In the sample program, the RCVD is set to 1 and communication is performed in byte unit.

4.3  Note on Setting ACKBT in Master Receive Mode
In master receive mode operation, set ACKBT bit before the falling edge of the 8th SCL cycle of the final data on consecutive data transfer. Otherwise, an overrun may occur on the slave transmit device.

In the sample program, the RCVD is set to 1 and the communication is performed in byte unit, which does not apply to the above described case.

4.4  Note on Issuing A Stop Condition or Re-transmitting A Start Condition in Master Receive Mode
When the timing for issuing a stop condition or a retransmit start condition overlaps with the timing of the fall of the 9th clock of SCL, an extra one-clock SCL will be output after the 9th clock. Issue a stop condition or retransmit start condition after confirming the fall of the 9th clock of the SCL.

The following is how to confirm the fall of the 9th clock.
- After reading that the RDRF bit (receive data register full flag) in the ICSR register becomes 1, read the SCLO bit (SCL monitor flag) in the ICCR2 register becomes 0 (the SCL pin is “L”).

For the details on this item, refer to Renesas Technical Update (No.: TN-MC*-A020A/E).

4.5  Note on Using the IICRST
While the I2C bus is operating writing 0 to the ICE bit in the ICCR1 register or writing 1 to the IICRST bit in the ICCR2 register leads the BBSY bit in the ICCR2 register and the STOP bit in the ICSR register undefined.

For the details on this item, refer to Renesas Technical Update (No.: TN-MC*-A022A/E).

4.6  Note on Issuing A Stop Condition in Master Transmit Mode
When issuing a stop condition in master transmit mode with the ACKE bit =1 in the I2C bus interrupt enable register (ICIER), the stop condition may not be output normally depending on the timing of issuance.

For the details on this item, refer to Renesas Technical Update (No.: TN-MC*-A023A/E).
5. Hardware

5.1 Used Pins and the Pin Functions

Table 5.1 lists the pins used and the pin functions.

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Input/output</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>PB6/SCL3</td>
<td>Output</td>
<td>Clock output in the I^2C communications</td>
</tr>
<tr>
<td>PB7/SDA3</td>
<td>Output/Input</td>
<td>Data input/output in the I^2C communications</td>
</tr>
</tbody>
</table>

5.2 Hardware Configuration

Figure 5.1 shows the configuration diagram of SH7263 connecting the EEPROM.
6. Software

6.1 Operation Overview

In the sample code, IIC3 is set to master mode to execute the sequential read to the EEPROM for 10 bytes. For data transmission and reception, the I²C bus format is used. When the I²C communication is disturbed by the bus occupation in the slave device, switch the IIC3 to the clocked synchronous format to transmit a dummy clock until the bus is released. After the bus is released, the sequential read is carried out again using the I²C bus format.

6.1.1 Sequential Read Operation

Figure 6.1 shows the EEPROM sequential read.

For the device code and the device address, refer to the EEPROM data sheet. In the sample code, "B'1010" is employed as the device code, and "B'000" as the device address.

The memory address indicates the start address for reading the EEPROM. The address is incremented on the EEPROM side each time the data is read.

![Figure 6.1 EEPROM Sequential Read](image)

Notes:
1. Don’t care bits for 128k and 256k
2. Don’t care bit for 128k

6.1.2 Detection of Bus Occupation

In the sample code, the bus occupation by the slave device is monitored during the I²C communications. The bus occupation is determined after a certain period of standby for a cancellation in the following two events.

- Wait for completing transmission in master transmit mode (wait for the TEND bit setting).
- Wait for completing an issuance of a stop condition in master transmit mode or master receive mode (wait for the STOP bit setting).

6.1.3 Procedure of Bus Cancellation

Use the master receive mode in the clocked synchronous serial format. Set the MST bit and the master receive mode, and the clock is transmitted automatically. When the slave device releases the bus, the SDA pin becomes H. When detecting that the SDA pin becomes H, stop the clock. For details on the procedure, refer to Figure 6.9 Function for Recovery from Bus Occupation State.

In the sample code, the sequential read is restarted after issuing a stop condition. For the recovery measure, refer to the specification of the slave device.
6.2 File Composition

Table 6.1 lists the file used in the sample code. Files generated by the integrated development environment should not be listed in this table.

<table>
<thead>
<tr>
<th>File Name</th>
<th>Outline</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>main.c</td>
<td>Main processing function</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EEPROM operation function</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IIC3 control function</td>
<td></td>
</tr>
</tbody>
</table>

6.3 Constants

Table 6.2 lists the constants used in the sample code.

<table>
<thead>
<tr>
<th>Constant Name</th>
<th>Setting Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EEPROM_MEM_ADDR</td>
<td>0x0000</td>
<td>Start address of the EEPROM</td>
</tr>
<tr>
<td>DEVICE_CODE</td>
<td>0xA0</td>
<td>Device code of the EEPROM</td>
</tr>
<tr>
<td>DEVICE_ADDR</td>
<td>0x00</td>
<td>Device address of the EEPROM</td>
</tr>
<tr>
<td>IIC_DATA_WR</td>
<td>0x00</td>
<td>Data write code</td>
</tr>
<tr>
<td>IIC_DATA_RD</td>
<td>0x01</td>
<td>Data read code</td>
</tr>
<tr>
<td>IIC3_DATA</td>
<td>10</td>
<td>Data transfer size in bytes</td>
</tr>
<tr>
<td>E_OK</td>
<td>0</td>
<td>Normal end</td>
</tr>
<tr>
<td>E_ERR_ACK</td>
<td>-1</td>
<td>NAK reception error</td>
</tr>
<tr>
<td>E_ERR_BUS</td>
<td>-2</td>
<td>Detects bus occupation failures</td>
</tr>
<tr>
<td>E_ERR_FATAL</td>
<td>-3</td>
<td>Detects fatal errors</td>
</tr>
</tbody>
</table>
6.4 Variables
Table 6.3 lists the global variable used in the sample code.

<table>
<thead>
<tr>
<th>Table 6.3 Global Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

6.5 Functions
Table 6.4 lists the functions used in the sample code.

<table>
<thead>
<tr>
<th>Table 6.4 Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
</tr>
<tr>
<td>main</td>
</tr>
<tr>
<td>io_iic3_init</td>
</tr>
<tr>
<td>io_iic3_eeprom_read</td>
</tr>
<tr>
<td>io_iic3_data_receive</td>
</tr>
<tr>
<td>io_iic3_address_send</td>
</tr>
<tr>
<td>io_iic3_data_send</td>
</tr>
<tr>
<td>io_iic3_mst_send_end</td>
</tr>
<tr>
<td>io_iic3_bus_recovery</td>
</tr>
</tbody>
</table>

6.6 Function Specifications
The following tables list the function specifications in the sample code.

main

<table>
<thead>
<tr>
<th>Outline</th>
<th>Main processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>None</td>
</tr>
<tr>
<td>Declaration</td>
<td>void main(void);</td>
</tr>
<tr>
<td>Description</td>
<td>Initializes the IIC3 and reads data for 10 bytes from the EEPROM. In case that the bus occupation failure occurs, issues a dummy clock to recover from the failure. Repeats data read till the processing ends normally.</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Returned value</td>
<td>None</td>
</tr>
</tbody>
</table>

io_iic3_init

<table>
<thead>
<tr>
<th>Outline</th>
<th>Initializes the IIC3 module</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>None</td>
</tr>
<tr>
<td>Declaration</td>
<td>void io_iic3_init(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Initializes the IIC3 channel 3.</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Returned value</td>
<td>None</td>
</tr>
</tbody>
</table>
io_iic3_eeprom_read
Outline  Reads data from the EEPROM
Header   None
Declaration int io_iic3_eeprom_read( unsigned char d_code, unsigned char d_adr,
               unsigned short r_adr, unsigned int r_size, unsigned char *r_buf)
Description Reads data for r_size byte from the EEPROM specified by the device code d_code
               and the device address d_adr. Stores the read data to the area specified by r_buf.
               The memory address for the EEPROM is specified by r_adr.
Argument  unsigned char d_code : Device code
               unsigned char d_adr : Device address
               unsigned short r_adr : Address at read destination
               unsigned int r_size : Byte count to read
               unsigned char *r_buf : Storage address for the read data
Returned value  E_OK : Normal end
               E_ERR_ACK: NAK reception error
               E_ERR_BUS: Detects a bus occupation failure

io_iic3_data_receive
Outline  Master receive mode
Header   None
Declaration int io_iic3_data_receive(unsigned char *r_buf, unsigned int r_size)
Description Sets to master receive mode, receives for the byte counts specified by r_size, and
               stores the received data to r_buf. After receiving the specified byte counts, switches
               the mode to slave receive mode.
Argument  unsigned int r_size : Storage for the read data
               unsigned char *r_buf : Data size for the read data
Returned value  E_OK : Normal end
               E_ERR_BUS: Detects a bus occupation failure

io_iic3_address_send
Outline  Transmits the slave device address
Header   None
Declaration int io_iic3_address_send(unsigned char *data)
Description Transmits the one-byte slave device address and 2-byte memory address specified
               by data.
Argument  unsigned char *data : Address for transmit data
Returned value  E_OK : Normal end
               E_ERR_ACK: NAK reception error
               E_ERR_BUS: Detects a bus occupation failure
**io_iic3_data_send**

**Outline**
Transmits data for one byte

**Header**
None

**Declaration**
int io_iic3_data_send(unsigned char data)

**Description**
Transmits data on the procedure below.
1. Waits for ICDRT empty
2. Sets the transmit data
3. Confirms transmission completed
4. Confirms ACK response

**Argument**
unsigned char data : Transmit data

**Returned value**
E_OK : Normal end
E_ERR_ACK: NAK reception error
E_ERR_BUS: Detects a bus occupation failure

**io_iic3_mst_send_end**

**Outline**
Issues a stop condition

**Header**
None

**Declaration**
int io_iic3_mst_send_end(void)

**Description**
Issues a stop condition, and switches to the slave receive mode.

**Argument**
None

**Returned value**
E_OK : Normal end
E_ERR_ACK: NAK reception error
E_ERR_BUS: Detects a bus occupation failure

**io_iic3_bus_recovery**

**Outline**
Recovers from the bus occupation state

**Header**
None

**Declaration**
int io_iic3_bus_recovery(void)

**Description**
Transmits the 8-bit dummy clock in the master receive mode with the clocked synchronous serial format. After transmitting the dummy clock, and if the SDA pin H is not detected, returns an error.

**Argument**
None

**Returned value**
E_OK : Normal end
E_ERR_FATAL: Fatal error
6.7 Flowchart
This section describes the processing procedure of the major functions used in the sample code. The words in boldface in the figures indicate the processing related to the bus occupation failure and recovery from the failure.

6.7.1 Main Function
Figure 6.2 shows the procedure of the main function.

![Main Function Flowchart]

Figure 6.2 Main Function
6.7.2 IIC3 Module Initialization

Figure 6.3 shows the procedure of the function for initializing the IIC3 module.

```
IIC3 module initial setting
(i0_iic3_init)

START

Set standby control register 5
(STBCR5)

- Enables clock supply to IIC3 ch3 (STBCR5)
- Clears the MSTP54 (module stop 54) bit to 0
  [Function] Supplies clock to IIC3 ch3

Set port B control register 2
(PBCRL2)

- Sets the port B pin (PBCRL2)
- Sets the PB7MD (PB7 mode) bit to 1
  [Function] Sets SDA3 pin
- Sets the PB6MD (PB6 mode) bit to 1
  [Function] Sets SCL3 pin

Set I2C bus control register 1
(ICCR1_3)

- Sets the I2C bus control register 1 (ICCR1_3)
- Sets the ICE (I2C bus interface 3 enable) bit to 1
  [Function] Enables module function to operate
- Sets the RCVD (reception disable) bit to 0
  [Function] Continues the next reception
- Sets the CKS (transfer clock select) bit to 4
  [Function] P15 / 84

Set I2C bus mode register
(ICMR_3)

- Sets the I2C bus mode register (ICMR_3)
- Sets the MLS (MSB-first/LSB-first selectable) bit to 0
  [Function] MSB-first
- Sets the BCWP (BC write protect) bit to 0
  [Function] Sets the value to BC[2:0] when writing
- Sets the BC (bit counter) bit to 0
  [Function] 9 bits

END
```

Figure 6.3 IIC3 Module Initialization
6.7.3 Function for Data Read from the EEPROM

Figure 6.4 shows the procedure of the function for data read from the EEPROM.

Data read from the EEPROM (io_iic3_eeprom_read)

1. Place in master transmit mode
2. Issue the start condition
3. Transmit the slave device address
   io_iic3_address_send function
   - 1st byte: device code, device address, write code
   - 2nd byte: memory address (upper byte)
   - 3rd byte: memory address (lower byte)
4. Received the ACK response?
   - Yes: Transmission successful
   - No: Issue the start condition
5. Transmit the slave device address
   io_iic3_address_send function
6. Received the ACK response?
   - Yes: Transmission successful
   - No: Issue the start condition
7. Transmit data
   io_iic3_data_send function
   - 4th byte: device code, device address, read code
8. Received the ACK response?
   - Yes: Transmission successful
   - No: Issue the start condition
9. Receive data
   io_iic3_data_receive function
10. Bus occupation failure occurs?
    - No: Data read from EEPROM
    - Yes:
      - NAK responded?
        - Yes: Issue a stop condition in slave receive mode
          io_iic3_mst_send_end function
        - No: E_OK
      - NAK responded?
        - Yes: Issue a stop condition in slave receive mode
          io_iic3_mst_send_end function
        - No: E.ERR_ACK

   - Determines that a bus occupation failure occurred when the standby for a transmit completion or for issuing a stop condition has not completed.
6.7.4 Function for Data Read

Figure 6.5 and Figure 6.6 show the procedure of the function for data read.

![Flowchart of Data Read](Image)

**Figure 6.5 Function for Data Read**
**Figure 6.6 Function for Data Read**

1. Received 1-byte data?
   - Yes: Issue a stop condition
   - No: SCL="L"?
     - Yes: Loop for more than a certain period?
     - No: Stop bit is 1?
       - Yes: Read data
       - No: Clear the STOP bit

   - Wait for the falling of 9th clock of the SCK before issuing a stop condition. (Refer to Technical Update [TN-MC*-A020A/E])

2. E_OK

- Receive data (final byte)
- Set the slave receive mode
- Clear the RCVD bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Issue a stop condition
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
- Yes: Read data
- No: Clear the STOP bit
- Loop for more than a certain period?
- Stop bit is 1?
6.7.5 Function for Transmitting Slave Device Address

Figure 6.7 shows the procedure of the function for transmitting the slave device address.

![Function Diagram](image)

- **Transmit a slave device address (io_iic3_address_send)**

- **Start**

- Transmit data
  - io_iic3_data_send function

- Received ACK response?
  - Yes
  - Transmit data
    - io_iic3_data_send function
  -Received ACK response?
    - Yes
    - Transmit data
      - io_iic3_data_send function
      - 1st byte: device code, device address, write code
    - No
    - Received ACK response?
      - Yes
      - Transmit data
        - io_iic3_data_send function
        - 2nd byte: memory address (upper byte)
      - No
      - Received ACK response?
        - Yes
        - Transmit data
          - io_iic3_data_send function
          - 3rd byte: memory address (lower byte)
        - No
        - Bus failure occurred?
          - Yes
          - E_ERR_BUS
          - No
          - Received ACK response?
            - Yes
            - E_OK
            - No
            - E_ERR_ACK

**Figure 6.7 Function for Slave Device Address Transmission**
6.7.6 Function for Data Transmission

Figure 6.8 shows the procedure of the functions for data transmission.

![Flowchart of Data Transmission Functions]

- **Transmit data** (io_iic3_data_send)
  - **START**
  - Transmit data register is empty?
    - No
    - Yes: Set transmit data
  - Arbitration lost occurred?
    - No
    - Yes: E_ERR_BUS
  - Completed transmission?
    - No
    - Yes: Received ACK response?
      - No
        - E_ERR_ACK
      - Yes: E_OK

- **Issue a stop condition in the slave receive mode** (io_iic3_mst_send_end)
  - **START**
  - Clear the TEND bit
  - Clear the STOP bit
  - Issue a stop condition
  - Loop more than a certain period?
    - No
    - Yes: STOP = 1?
      - No
        - E_ERR_BUS
      - Yes: Set slave receive mode
        - Clear the TDRE bit
  - Completed transmission?
    - No
    - Yes: E_OK

*After confirming the transmit data register is empty, set the transmit data.*

*When occurring a bus occupation failure while transmitting data, the transmission is terminated by an arbitration lost.*

*When a bus occupation failure occurs, a stop condition cannot be issued once a bus occupation failure occurs.*

---

**Figure 6.8 Functions for Data Transmission**
### 6.7.7 Function for Recovery from Bus Occupation State

Figure 6.9 shows the procedure of the function for recovering from the bus occupation state.

<table>
<thead>
<tr>
<th>Recover from bus occupation failure (io_iic3_bus_recovery)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>START</strong></td>
</tr>
</tbody>
</table>

**Set I^2^C bus control register 1 (ICCR1_3)**
- Sets the I^2^C bus control register 1 (ICCR1_3)
- Sets the ICE (I^2^C bus interface 3 enable) bit to 0
- **Function**: Module Function halted
- Sets the RCVD (reception disable) bit to 0
- **Function**: Continues the following reception
- Sets the MST (master/slave selectable) bit to 0
- Sets the TRS (transmission/reception selectable) bit to 0
- **Function**: Slave receive mode
- Sets the CKS (transfer clock select) bit to 4
- **Function**: P_ψ/84

**Reset the IIC control part**

**Clear the status bit**

**Place in the clocked synchronous serial format**

**Enable module function**

**Start clock output**

**Wait for completing transfer for one bit**

**Disable the next reception**
- Sets the RCVD bit to 1

**Wait for completing reception**
- Waits till the RDRF bit becomes 1

**SDA pin is H?**

- Yes
  - **E_ERR_FATAL**
  - Port B port register (PBPRL)
  - Reads PB7PR bit (SDA3)

- No
  - Sets the FS bit to 0
  - **E_OK**
  - Sets the MST bit to 0

**Return to IIC format**

**Place in the slave receive mode**

---

**Figure 6.9 Function for Recovery from Bus Occupation State**
7. Sample Code
Sample code can be downloaded from the Renesas Electronics website.

8. Reference Documents

Hardware Manual
SH7263 Group Hardware Manual Rev.3.00
SH7203 Group Hardware Manual Rev.3.00
The latest version can be downloaded from the Renesas Electronics website.

Technical Update/Technical News
The latest information can be downloaded from the Renesas Electronics website.

Development Tool Manual
SuperH C/C++ Compiler Package V.9.04 User’s Manual Rev.1.01
The latest version can be downloaded from the Renesas Electronics website.

SuperH Family E10A-USB Emulator User’s Manual Rev. 9.00
The latest version can be downloaded from the Renesas Electronics website.

Website and Support
Renesas Electronics website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/
<table>
<thead>
<tr>
<th>Rev.</th>
<th>Date</th>
<th>Description</th>
<th>Page</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>Jan. 23, 2012</td>
<td>—</td>
<td>—</td>
<td>First edition issued</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>1. Handling of Unused Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.</td>
</tr>
<tr>
<td>— 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.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2. Processing at Power-on</th>
</tr>
</thead>
<tbody>
<tr>
<td>The state of the product is undefined at the moment when power is supplied.</td>
</tr>
<tr>
<td>— 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.</td>
</tr>
<tr>
<td>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.</td>
</tr>
<tr>
<td>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.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3. Prohibition of Access to Reserved Addresses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access to reserved addresses is prohibited.</td>
</tr>
<tr>
<td>— 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.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4. Clock Signals</th>
</tr>
</thead>
<tbody>
<tr>
<td>After applying a reset, only release the reset line after the operating clock signal has become stable.</td>
</tr>
<tr>
<td>— 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.</td>
</tr>
<tr>
<td>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.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5. Differences between Products</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.</td>
</tr>
<tr>
<td>— 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.</td>
</tr>
</tbody>
</table>
Notice

1. All information in this document is current as of the date this document is issued. Such information, however, is subject to change without prior notice. Before purchasing or using any Renesas Electronics products listed herein, please consult the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional instructions and information to be issued by Renesas Electronics such as that disclosed through our websites.

2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by using or allowing 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 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 only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the interpretation 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 operating the products or technology described in this document, you should comply with all applicable import and export 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 or incorporated into any products or systems whose manufacturer, user, or resale is prohibited under any applicable export control laws or regulations.

6. Renesas Electronics is not responsible for any damages necessarily caused in preparing the information included in this document, but Renesas Electronics does not warrant that the information included herein is error-free. Renesas Electronics assumes no liability whatsoever for any damages necessarily caused by errors or omissions to 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 depend 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 any application categorized as "Specific" or for the use of the product is not prescribed where you have been told to obtain the prior written consent of Renesas Electronics.

8. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheet or data book, etc.


10. "High Quality": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-theft systems, and other safety equipment, and medical equipment not specifically designed for use in automobiles.

11. "Specific": Aircraft, aerospace equipment, semiconductor testers, nuclear reactor control systems, medical equipment or systems for the support (e.g., artificial life support systems), surgical equipment, or medical instruments (e.g., X-ray machines, etc.), and any other applications or purposes that are directly related to human life.

12. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics and comply with any applicable standards and regulations. Operating outside the range, even under conditions that appear to be safe, may not be safe. Renesas Electronics shall not be liable for any failures or damages arising from the use of Renesas Electronics products beyond their specifications.

13. Although Renesas Electronics endeavors to improve the quality and reliability of its products, some products contain specific characteristics such as a certain occurrence rate of failures or a certain occurrence rate of failures under certain conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard against the possibility of physical injury, loss or damage caused by the use of the Renesas Electronics products, such as safety design for hardware and software integration and safe and reliable implementation. Safety measures are very difficult, please consider the use of the line production system manufactured by others.

14. Please contact Renesas Electronics sales office for details about the environmental matters such as the environmental compatibility or the Renesas Electronics product. Please use Renesas Electronics products in compliance with applicable laws and regulations that regulate the indication or use of controlled substances, including without limitation, REACH Directive. Renesas Electronics assumes no liability to damages or losses occurring as a result of your compliance with applicable laws and regulations.

15. This document may not be reproduced or distributed, in any form, in whole or in part, without prior written consent of Renesas Electronics.

16. Please contact a Renesas Electronics sales office if you have any questions regarding the information included in this document or Renesas Electronics product, or if you have any other inquiries.

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

https://www.renesas.com