Introduction

This application note describes slave transmission and reception implemented via the serial interface IICA. Using IICA, the single master system described here performs slave operation (address reception, and data transmission and reception).

Target Device

RL78/G13

When applying the sample program covered in this application note to another microcomputer, modify the program according to the specifications for the target microcomputer and conduct an extensive evaluation of the modified program.
Contents

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

2. Operation Check Conditions ................................................................................................. 18

3. Related Application Note ........................................................................................................ 18

4. Description of the Hardware ................................................................................................. 19
   4.1 Hardware Configuration Example .................................................................................... 19
   4.2 List of Pins to be Used ................................................................................................. 19

5. Description of the Software ................................................................................................... 20
   5.1 Operation Outline ......................................................................................................... 20
   5.2 List of Option Byte Settings ....................................................................................... 21
   5.3 List of Constants .......................................................................................................... 21
   5.4 List of Variables ............................................................................................................ 22
   5.5 List of Functions ........................................................................................................... 22
   5.6 Function Specifications ................................................................................................. 23
   5.7 Flowcharts .................................................................................................................... 25
      5.7.1 Initialization Function ............................................................................................ 25
      5.7.2 System Function .................................................................................................... 26
      5.7.3 I/O Port Setup ....................................................................................................... 27
      5.7.4 CPU Clock Setup ................................................................................................ 28
      5.7.5 Serial Interface IICA Setup .................................................................................. 29
      5.7.6 Main Processing ...................................................................................................... 35
      5.7.7 IICA0 Reception Preparation Function ................................................................ 37
      5.7.8 IICA0 Transmission Preparation Function .......................................................... 38
      5.7.9 IICA0 Interrupt Processing .................................................................................... 39
      5.7.10 IICA0 Slave Handler ............................................................................................ 40
      5.7.11 Error Flag Return Processing ............................................................................. 43

6. Sample Code .......................................................................................................................... 44

7. Documents for Reference ...................................................................................................... 44
1. Specifications

This application note describes how the single master system performs slave transmission and reception (address reception, and data transmission and reception) through the serial interface IICA.

Table 1.1 lists the peripheral function to be used and its use. Figure 1.1 presents an overview of IIC communication. Figures 1.2 through 1.8 show timing charts for explaining the IIC communication.

Table 1.1 Peripheral Function to be Used and Its Use

<table>
<thead>
<tr>
<th>Peripheral Function</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial interface IICA</td>
<td>IIC slave transmission/reception in a single master system (using the SCLA0 and SDAA0 pins)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RL78/G13 (master)</th>
<th>Address reception</th>
</tr>
</thead>
<tbody>
<tr>
<td>RL78/G13 (slave)</td>
<td>Data transmission/reception</td>
</tr>
</tbody>
</table>

Address reception: Each device connected to the IIC bus has a unique address. Each device receives the address of a transfer destination (slave) from the master. That is, it receives one byte of data consisting of 7 bits (indicating the address) and 1 bit (indicating the transfer direction). The slave generates an acknowledgement after receiving one byte of data.

Data transmission/reception: The slave sends/receives data to/from the master after receiving the address.

Figure 1.1 Overview of IIC Communication
(1) Master-to-slave communication 1 (start condition – address – data)

Note: The time between the falling edges of SDAA0 and SCLA0 is 4.0 \(\mu s\) or more for the standard mode and 0.6 \(\mu s\) or more for the fast mode.

Figure 1.2  IIC Communication Timing Chart (Master-to-Slave Communication Example) (1/4)
(1) The start condition trigger is set (STT0 = 1) on the master side. Then, the SDAA0 line falls, thereby generating a start condition. Later, when the start condition is detected (STD0 = 1), the master enters a master device communication state (MSTS0 = 1). The SCLA0 line falls at the end of the hold period. This completes preparations for communication.

(2) The values of the address and data direction bit W (transmission) are written to the IICA0 register on the master side. Then, the slave address is transmitted.

(3) If the received address and slave address match Note, the slave hardware sends ACK0 to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).

(4) When the ninth clock signal falls, an address transmission end interrupt (INTIICA0) occurs on the master side. If the addresses match, an address match interrupt (INTIICA0) occurs on the slave side. Both the master and the slave which has the matching address enter a wait state (SCLA0 line: Low) Note.

(5) The master writes transmit data to the IICA0 register and cancels the wait.

(6) The slave selects an 8-clock wait (WTIM0 = 0) because it receives data. When the slave cancels the wait (WREL0 = 1), the master starts transferring data to the slave.

Note: If the received address and local address do not match, the slave does not return ACK to the master (NACK). The INTIICA0 interrupt (address match interrupt) does not occur on the slave side and thus the slave does not enter a wait state. However, the INTIICA0 interrupt (address transmission end interrupt) occurs on the master side regardless of whether the master receives ACK or NACK.
(2) Master-to-slave communication 2 (address – data – data)

Figure 1.3  IIC Communication Timing Chart (Master-to-Slave Communication Example) (2/4)
(3) If the received address and slave address match \(^{\text{Note}}\), the slave hardware sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).

(4) When the ninth clock signal falls, an address transmission end interrupt (INTIICA0) occurs on the master side. If the addresses match, an address match interrupt (INTIICA0) occurs on the slave side. Both the master and the slave which has the matching address enter a wait state (SCLA0 line: Low).

(5) The master writes transmit data to the IICA0 register and cancels the wait.

(6) The slave selects an 8-clock wait (WTIM0 = 0) because it receives data. When the slave cancels the wait (WREL0 = 1), the master starts transferring data to the slave.

(7) When the eighth clock signal falls after the data transfer, the slave hardware generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.

(8) When the slave reads the receive data and cancels the wait (WREL0 = 1), the slave sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).

(9) When the ninth clock signal falls, the master generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master side.

(10) The master writes transmit data to the IICA0 register and cancels the wait. Then, the master starts transferring data to the slave.

\(^{\text{Note}}\): If the received address and local address do not match, the slave does not return ACK to the master (NACK). The INTIICA0 interrupt (address match interrupt) does not occur on the slave side and thus the slave does not enter a wait state. However, the INTIICA0 interrupt (address transmission end interrupt) occurs on the master side regardless of whether the master receives ACK or NACK.
(3) Master-to-slave communication 3 (data – data – stop condition)

Note: The time from when SCLA0 rises after a stop condition is issued till when a stop condition is generated is 4.0 μs or more for the standard mode and 0.6 μs or more for the fast mode.

Figure 1.4  IIC Communication Timing Chart (Master-to-Slave Communication Example) (3/4)
(7) When the eighth clock signal falls after the data transfer, the slave hardware generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.

(8) When the slave reads the receive data and cancels the wait (WREL0 = 1), the slave sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).

(9) When the ninth clock signal falls, the master generates a wait (SCLA0 line: Low) and an address transmission end interrupt (INTIICA0) occurs on the master side.

(10) The master writes transmit data to the IICA0 register and cancels the wait. Then, the master starts transferring the data to the slave.

(11) When the eighth clock signal falls after the data transfer, the slave hardware generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.

(12) When the slave reads the receive data and cancels the wait (WREL0 = 1), the slave sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).

(13) When the ninth clock signal falls, the master generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master side.

(14) When the stop condition trigger is set (SPT0 = 1), the SDAA0 line falls and the SCLA0 line rises. Upon the elapse of the stop condition setup time, the SDAA0 line rises, thereby generating a stop condition.

(15) When the stop condition is generated, the slave detects it (SPD0 = 1) and a IICA0 interrupt (stop condition interrupt) occurs on the slave side.
(4) Master-to-slave communication 4 (data – restart condition – address)

Note: The time from when SCLA0 rises after a restart condition is issued till when a start condition is generated is 4.7 $\mu$s or more for the standard mode and 0.6 $\mu$s or more for the fast mode.

Figure 1.5  IIC Communication Timing Chart (Master-to-Slave Communication Example) (4/4)
(7) When the eighth clock signal falls after the data transfer, the slave hardware generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.

(8) The slave reads the receive data and cancels the wait (WREL0 = 1). Then, the slave sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).

(9) When the ninth clock signal falls, the master generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master side.

(10) The start condition trigger is set (STT0 = 1) on the master side again. Then, the SCLA0 line rises. Upon the elapse of the restart condition setup time, the SDAA0 line falls, thereby generating a start condition. Later, at the end of the hold period after the start condition is detected (STD0 = 1), the bus clock line falls, thereby completing preparations for communication.

(11) The master writes the slave address to the IICA0 register and starts transferring the address to the slave.
(5) Slave-to-master communication 1 (start condition – address – data)

Note: The time between the falling edges of SDAA0 and SCLA0 is 4.0 μs or more for the standard mode and 0.6 μs or more for the fast mode.

Figure 1.6  IIC Communication Timing Chart (Slave-to-Master Communication Example) (1/3)
(1) The start condition trigger is set (STT0 = 1) on the master side. Then, the SDAA0 line falls, thereby generating a start condition. Later, when the start condition is detected (STD0 = 1), the master enters a master device communication state (MSTS0 = 1). The SCLA0 line falls at the end of the hold period. This completes preparations for communication.

(2) The values of the address and data direction bit R (reception) are written to the IICA0 register on the master side. Then, the slave address is transmitted.

(3) If the received address and slave address match Note, the slave hardware sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKDO = 1).

(4) When the ninth clock signal falls, an address transmission end interrupt (INTIICA0) occurs on the master side. If the addresses match, an address match interrupt (INTIICA0) occurs on the slave side. Both the master and the slave which has the matching address enter a wait state (SCLA0 line: Low).

(5) The master selects an 8-clock wait (WTIM0 = 0) because it receives data.

(6) The slave writes transmit data to the IICA0 register and cancels the wait.

(7) When the master cancels the wait (WREL0 = 1), the slave starts transferring data to the master.

Note: If the received address and local address do not match, the slave does not return ACK to the master (NACK). The INTIICA0 interrupt (address match interrupt) does not occur on the slave side and thus the slave does not enter a wait state. However, the INTIICA0 interrupt (address transmission end interrupt) occurs on the master side regardless of whether the master receives ACK or NACK.
(6) Slave-to-master communication 2 (address – data – data)

Figure 1.7  IIC Communication Timing Chart (Slave-to-Master Communication Example) (2/3)
(3) If the received address and slave address match Note, the slave hardware sends ACK to the master. When the ninth clock signal rises, the master detects ACK (ACKD0 = 1).

(4) When the ninth clock signal falls, an address transmission end interrupt (INTIICA0) occurs on the master side. If the addresses match, an address match interrupt (INTIICA0) occurs on the slave side. Both the master and the slave which has the matching address enter a wait state (SCLA0 line: Low).

(5) The master selects an 8-clock wait (WTIM0 = 0) because it receives data.

(6) The slave writes transmit data to the IICA0 register and cancels the wait.

(7) When the master cancels the wait (WREL0 = 1), the slave starts transferring data to the master.

(8) When the eighth clock signal falls, the master generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master side. The master hardware sends ACK to the slave.

(9) The master reads the receive data and cancels the wait (WREL0 = 1).

(10) When the ninth clock signal rises, the slave detects ACK (ACKD0 = 1).

(11) When the ninth clock signal falls, the slave generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.

(12) The slave writes transmit data to the IICA0 register and cancels the wait. Then, the slave starts transferring data to the master.

Note: If the received address and local address do not match, the slave does not return ACK to the master (NACK). The INTIICA0 interrupt (address match interrupt) does not occur on the slave side and thus the slave does not enter a wait state. However, the INTIICA0 interrupt (address transmission end interrupt) occurs on the master side regardless of whether the master receives ACK or NACK.
(7) Slave-to-master communication 3 (data – data – stop condition)

Note: The time from when SCLA0 rises after a stop condition is issued till when a stop condition is generated is 4.0 μs or more for the standard mode and 0.6 μs or more for the fast mode.

Figure 1.8  IIC Communication Timing Chart (Slave-to-Master Communication Example) (3/3)

Note: The time from when SCLA0 rises after a stop condition is issued till when a stop condition is generated is 4.0 μs or more for the standard mode and 0.6 μs or more for the fast mode.
(8) When the eighth clock signal falls, the master generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master side. The master hardware sends ACK to the slave.

(9) The master reads the receive data and cancels the wait (WREL0 = 1).

(10) When the ninth clock signal rises, the slave detects ACK (ACKD0 = 1).

(11) When the ninth clock signal falls, the slave generates a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the slave side.

(12) The slave writes transmit data to the IICA0 register and cancels the wait. Then, the slave starts transferring data to the master.

(13) When the eighth clock signal falls, a transfer end interrupt (INTIICA0) occurs on the master side and the master generates a wait (SCLA0 line: Low). The master hardware sends ACK to the slave.

(14) The master sets a NACK response (ACKE0 = 0) to inform the slave that the master has sent the last data (at the end of communication). Then, the master changes the wait time to 9 clock periods (WTIM0 = 1).

(15) After the master cancels the wait (WREL0 = 1), the slave detects NACK (ACKD0 = 0) at the rising edge of the ninth clock signal.

(16) When the ninth clock signal falls, the master and slave generate a wait (SCLA0 line: Low) and a transfer end interrupt (INTIICA0) occurs on the master and slave sides.

(17) When the master issues a stop condition (SPT0 = 1), the SDAA0 line falls, thereby canceling the wait on the master side. Later, the master waits until the SCLA0 line rises.

(18) The slave cancels the wait (WREL0 = 1) to terminate communication. Then, the SCLA0 line rises.

(19) The master confirms that the SCLA0 line has risen. Upon the elapse of the stop condition setup time after this confirmation, the master makes the SDAA0 line rise and issues a stop condition. When the stop condition is generated, the slave detects the stop condition (SPD0 = 1) and a stop condition interrupt (INTIICA0) occurs on the master and slave sides.
2. Operation Check Conditions

The sample code contained in this application note has been checked under the conditions listed in the table below.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microcontroller used</td>
<td>RL78/G13 (R5F100LEA)</td>
</tr>
<tr>
<td>Operating frequency</td>
<td>• High-speed on-chip oscillator (HOCO) clock: 32 MHz</td>
</tr>
<tr>
<td></td>
<td>• CPU/peripheral hardware clock: 32 MHz</td>
</tr>
<tr>
<td>Operating voltage</td>
<td>5.0 V (Operation is possible over a voltage range of 2.9 V to 5.5 V.)</td>
</tr>
<tr>
<td></td>
<td>LVD operation ((V_{LVD})): Reset mode which uses 2.81 V (2.76 V to 2.87 V)</td>
</tr>
<tr>
<td>Integrated development environment (CS+)</td>
<td>CS+ V3.01.00 from Renesas Electronics Corp.</td>
</tr>
<tr>
<td>C compiler (CS+)</td>
<td>CC-RL V1.01.00 from Renesas Electronics Corp.</td>
</tr>
<tr>
<td>Integrated development environment (e² studio)</td>
<td>e² studio V4.0.0.26 from Renesas Electronics Corp.</td>
</tr>
<tr>
<td>C compiler (e² studio)</td>
<td>CC-RL V1.01.00 from Renesas Electronics Corp.</td>
</tr>
</tbody>
</table>

3. Related Application Note

The application notes that are related to this application note are listed below for reference.

- RL78/G13 Initialization (R01AN2575EJ0100) Application Note
- RL78/G13 Serial Interface IICA (for Master Transmission/Reception) (R01AN2759EJ0100) Application Note
4. Description of the Hardware

4.1 Hardware Configuration Example

Figure 4.1 shows an example of hardware configuration that is used for this application note.

Cautions: 1. The purpose of this circuit is only to provide the connection outline and the circuit is simplified accordingly. When designing and implementing an actual circuit, provide proper pin treatment and make sure that the hardware's electrical specifications are met (connect the input-only ports separately to V_{DD} or V_{SS} via a resistor).
2. Connect any pins whose name begins with EV_{SS} to V_{SS} and any pins whose name begins with EV_{DD} to V_{DD}, respectively.
3. V_{DD} must be held at not lower than the reset release voltage (V_{LVD}) that is specified as LVD.

4.2 List of Pins to be Used

Table 4.1 lists the pins to be used and their functions.

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>I/O</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>P60/SCLA0</td>
<td>Input/Output</td>
<td>IICA0 serial clock I/O pin</td>
</tr>
<tr>
<td>P61/SDAA0</td>
<td>Input/Output</td>
<td>IICA0 serial data transmission/reception pin</td>
</tr>
</tbody>
</table>
5. Description of the Software

5.1 Operation Outline

The sample program covered in this application note provides IICA slave transmission and reception (address reception, and data transmission and reception) through the serial interface IICA.

(1) Initialize serial interface IICA.

<Conditions for setting>
- Select the fast mode as the operation mode.
- Set the transfer clock frequency to 400 kHz.
- Set the local address to 0xA0.
- Turn the digital filter on.
- Generate an interrupt in response to the ninth clock signal.
- Disable stop condition interrupts.
- Use the P60/SCLA0 pin for transfer clock input and the P61/SDAA0 pin for data transmission/reception.

(2) Get the communication buffer (16 bytes) ready for use.

(3) After initialization is completed, use the wakeup function of IICA to reduce the power consumption. Execute a STOP instruction and wait for an interrupt (INTIICA0) to occur indicating the reception of the local address or extension code.

(4) When the local address or extension code is received, perform wakeup operation and start data communication.

(5) Receive data from the master side. This data (16 bytes) is stored in the communication buffer sequentially. After the communication is completed, store transmit data (16 bytes) in the communication buffer in preparation for the next transmission.

(6) Use the wakeup function of IICA. Execute a STOP instruction and wait for an interrupt (INTIICA0) to occur indicating the reception of the local address or extension code.

(7) After the wakeup, transmit data (16 bytes) to the master sequentially.

(8) Repeat steps (2) to (7).

Caution: This sample code is related to RL78/G13 Serial Interface IICA (for Master Transmission/Reception) (R01AN2759EJ0100) Application Note only. The conditions for completion of communication are as follows: detection of a stop condition during data reception, completion of transmission/reception of 16-byte data, or NACK detection. When communication is completed, the next processing is started (for example, when reception is completed, transmission is started).
5.2 List of Option Byte Settings

Table 5.1 summarizes the settings of the option bytes.

**Table 5.1   Option Byte Settings**

<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000C0H/010C0H</td>
<td>01101110B</td>
<td>Disables the watchdog timer. (Stops counting after the release from the reset state.)</td>
</tr>
<tr>
<td>000C1H/010C1H</td>
<td>01111111B</td>
<td>LVD reset mode, 2.81 V (2.76 V to 2.87 V)</td>
</tr>
<tr>
<td>000C2H/010C2H</td>
<td>11101000B</td>
<td>HS mode, HOCO: 32 MHz</td>
</tr>
<tr>
<td>000C3H/010C3H</td>
<td>10000101B</td>
<td>Enables the on-chip debugger.</td>
</tr>
</tbody>
</table>

5.3 List of Constants

Table 5.2 lists the constants that are used in this sample program.

**Table 5.2   Constants for the Sample Program**

<table>
<thead>
<tr>
<th>Constant</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA_LENGTH</td>
<td>0x10</td>
<td>IIC transmit/receive data length</td>
</tr>
<tr>
<td>_80_IICA_STATUS_MASTER</td>
<td>0x80</td>
<td>Constant for determining the IICS0 value (mask for reading the master state check flag value)</td>
</tr>
<tr>
<td>_80_IICA_ADDRESS_COMPLETE</td>
<td>0x80</td>
<td>Variable g_iica0_slave_status_flag setting (address transmission complete state)</td>
</tr>
<tr>
<td>MD_NACK</td>
<td>0x02U</td>
<td>NACK response</td>
</tr>
<tr>
<td>MD_ERROR</td>
<td>0x80U</td>
<td>Addresses do not match</td>
</tr>
</tbody>
</table>
5.4 List of Variables

Table 5.3 lists the global variables that are used in this sample program.

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable Name</th>
<th>Contents</th>
<th>Function Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>uint8_t</td>
<td>g_iica0_slave_status_flag</td>
<td>IICA0 slave flag</td>
<td>R_IICA0_Slave_Send()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>R_IICA0_Slave_Receive()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>iica0_slave_handler()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>r_iica0_callback_slave_error()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>main()</td>
</tr>
<tr>
<td>uint8_t*</td>
<td>gp_iica0_rx_address</td>
<td>IICA0 receive buffer address</td>
<td>R_IICA0_Slave_Receive()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>iica0_slave_handler()</td>
</tr>
<tr>
<td>uint16_t</td>
<td>g_iica0_rx_len</td>
<td>IICA0 receive data length</td>
<td>R_IICA0_Slave_Receive()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>iica0_slave_handler()</td>
</tr>
<tr>
<td>uint16_t</td>
<td>g_iica0RxCnt</td>
<td>IICA0 receive data count</td>
<td>R_IICA0_Slave_Receive()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>iica0_slave_handler()</td>
</tr>
<tr>
<td>uint8_t*</td>
<td>g_plica0TxAddress</td>
<td>IICA0 transmit buffer address</td>
<td>R_IICA0_Slave_Send()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>iica0_slave_handler()</td>
</tr>
<tr>
<td>uint16_t</td>
<td>g_plica0TxCnt</td>
<td>IICA0 transmit data count</td>
<td>R_IICA0_Slave_Send()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>iica0_slave_handler()</td>
</tr>
<tr>
<td>static uint8_t</td>
<td>rx_data[DATA_LENGTH]</td>
<td>Data receive buffer</td>
<td>R_IICA0_Slave_Receive()</td>
</tr>
<tr>
<td>static uint8_t</td>
<td>direction</td>
<td>Transmission direction flag</td>
<td>main()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>R_RxPreparation()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>R_TxPreparation()</td>
</tr>
</tbody>
</table>

5.5 List of Functions

Table 5.4 summarizes the functions that are used in this sample program.

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_RxPreparation</td>
<td>Reception preparation</td>
</tr>
<tr>
<td>R_TxPreparation</td>
<td>Transmission preparation</td>
</tr>
<tr>
<td>R_IICA0_Slave_Receive</td>
<td>Slave reception setting</td>
</tr>
<tr>
<td>R_IICA0_Slave_Send</td>
<td>Slave transmission setting</td>
</tr>
<tr>
<td>r_iica0_interrupt</td>
<td>IICA0 interrupt handler</td>
</tr>
<tr>
<td>iica0_slave_handler</td>
<td>Slave operation during an interrupt</td>
</tr>
<tr>
<td>r_iica0_callback_slave_error</td>
<td>Transmission/reception error processing</td>
</tr>
</tbody>
</table>
5.6 Function Specifications
This section describes the specifications for the functions that are used in this sample program.

### [Function Name] R_RxPreparation

**Synopsis**  
Reception preparation

**Header**  
-

**Declaration**  
void R_RxPreparation(void)

**Explanation**  
Prepares for data reception, and clears the IICA0 slave flag.

**Arguments**  
None

**Return value**  
None

**Remarks**  
None

### [Function Name] R_TxPreparation

**Synopsis**  
Transmission preparation

**Header**  
-

**Declaration**  
void R_TxPreparation(void)

**Explanation**  
Prepares for data transmission, and clears the IICA0 slave flag.

**Arguments**  
None

**Return value**  
None

**Remarks**  
None

### [Function Name] R_IICA0_Slave_Receive

**Synopsis**  
Slave reception setting

**Header**  
r_cg_serial.h

**Declaration**  
void R_IICA0_Slave_Receive(uint8_t * const rx_buf and uint16_t rx_num)

**Explanation**  
Specifies the slave reception mode.

**Arguments**  
rx_buf  
Receive data buffer address

rx_num  
Receive data length

**Return value**  
None

**Remarks**  
None

### [Function Name] R_IICA0_Slave_Send

**Synopsis**  
Slave transmission setting

**Header**  
r_cg_serial.h

**Declaration**  
void R_IICA0_Slave_Send(uint8_t * const tx_buf and uint16_t tx_num)

**Explanation**  
Specifies the slave transmission mode.

**Arguments**  
rx_buf  
Transmit data buffer address

rx_num  
Transmit data length

**Return value**  
None

**Remarks**  
None
**Synopsis**
IICA0 interrupt handler

**Header**
- r_iica0_interrupt

**Declaration**
static void __near r_iica0_interrupt(void)

**Explanation**
Interrupt handler for IICA0 interrupt.

**Arguments**
None

**Return value**
None

**Remarks**
In this sample code, in order to meet the specifications, a statement to disable the wakeup function is added to the beginning of this function, which is generated by the code generator. If the wakeup function is enabled, reading from the IICS0 register is prohibited.

---

**Synopsis**
Slave operation during interruption

**Header**
r_cg_serial.h

**Declaration**
static void iica0_slave_handler(void)

**Explanation**
Performs slave transmission and reception during IICA0 interrupt handling.

**Arguments**
None

**Return value**
None

**Remarks**
In this sample code, the statement to disable acknowledgement of the interrupt that occurs with the eighth clock signal during the reception of the last data (ACKE0 = 0) is deleted from this function, which is generated by the code generator, so that ACK is output after all the data is received.

---

**Synopsis**
Transmission/reception error processing

**Header**
r_cg_serial.h

**Declaration**
static void r_iica0_callback_slave_error(MD_STATUS flag)

**Explanation**
Handles, during IICA0 interrupt handling, errors that occur when addresses do not match or NACK is received.

**Arguments**
flag Error status

**Return value**
None

**Remarks**
None
5.7 Flowcharts

Figure 5.1 shows the overall flow of the sample program described in this application note.

![Overall Flow](image1)

5.7.1 Initialization Function

Figure 5.2 shows the flowchart for the initialization function.

![Initialization Function](image2)
5.7.2 System Function

Figure 5.3 shows the flowchart for the system function.

![Flowchart for System Function]

Figure 5.3 System Function
5.7.3 I/O Port Setup

Figure 5.4 shows the flowchart for I/O port setup.

Figure 5.4 I/O Port Setup

Note: Refer to the section entitled "Flowcharts" in RL78/G13 Initialization Application Note (R01AN2575EJ0100) for the configuration of the unused ports.

Caution: Provide proper treatment for unused pins so that their electrical specifications are observed. Connect each of any unused input-only ports to \( V_{DD} \) or \( V_{SS} \) via separate resistors.
5.7.4 CPU Clock Setup

Figure 5.5 shows the flowchart for setting up the CPU clock.

![Flowchart for CPU Clock Setup](image)

**cmi**: CPU Clock Setup

- **R_CGC_Create()**
  - Set up high-speed system clock/subsystem clock
  - Select CPU/peripheral hardware clock (fCLK)

- **CMC register**: 00H: Does not use high-speed system clock and subsystem clock.
- **MSTOP bit**: 1
- **XTSTOP bit**: 1
- **MCM0 bit**: 0: Select high-speed OCO clock (fOCO) as main system clock (fMAIN).
- **CSS bit**: 0: Selects main system clock (fMAIN) as CPU/peripheral hardware clock (fCLK).

Caution: For details on the procedure for setting up the CPU clock (R_CGC_Create()), refer to the section entitled "Flowcharts" in RL78/G13 Initialization Application Note (R01AN2575EJ0100).
5.7.5 Serial Interface IICA Setup

Figures 5.6 shows the flowchart for serial interface IICA setup.

---

**Figure 5.6 Serial Interface IICA Setup**

- **R_IICA0_Create**
  - Supply clock signals to IICA0
  - IICA0EN bit ← 1
  - IICE0 bit ← 0
  - IICAMK0 bit ← 1
  - IICAIF0 bit ← 0
  - IICAPR10 bit ← 1
  - IICAPR00 bit ← 1
  - SMC0 bit ← 1
  - IICWL0 register ← 15H
  - IICWH0 register ← 14H
  - DFC0 bit ← 1
  - PRS0 bit ← 1
  - SVA0 register ← A0H
  - STCEN0 bit ← 1
  - IICRSV0 bit ← 1
  - SPIE0 bit ← 0
  - WTIM0 bit ← 1
  - ACKE0 bit ← 1
  - IICAMK0 bit ← 0
  - IICE0 bit ← 1
  - LREL0 bit ← 1
  - P6 register ← 00H
  - PM6 register ← F0H

- Deactivate IICA0
- Disable IICA0 interrupts
- Clear IICA0 interrupt request flag
- Set IICA0 interrupt priority level to 3
- Set up IICA0
  - Fast mode
  - Set IICA low-level width
  - Set IICA high-level width
  - Turn digital filter on
  - Select f_{CLK}/2 as operation clock
  - Set local address to A0H
  - Allow generation of a start condition without detecting stop condition after enabling operation (IICE0 = 1)
  - Disable communication reservation function
  - Prevent interrupt request from being generated by stop condition detection
  - Generate interrupt request upon falling edge of 9th clock signal
  - Enable acknowledgement
  - Pull data line low during duration of 9th clock signal

- Enable IICA0 interrupts
- Enable IICA0
- Release data line and clock line
- Set up IICA0 pin
- return

---
Starting clock signal supply to serial interface IICA0

- Peripheral enable register 0 (PER0)
  Start supplying clock signals to IICA0 by using IICAEN.

Symbol: PER0

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTCEN</td>
<td>IICA1EN</td>
<td>ADCEN</td>
<td>IICA0EN</td>
<td>SAU1EN</td>
<td>SAU0EN</td>
<td>TAU1EN</td>
<td>TAU0EN</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

Bit 4

<table>
<thead>
<tr>
<th>IICA0EN</th>
<th>Serial interface IICA0 input clock control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Stops supply of input clock.</td>
</tr>
<tr>
<td>1</td>
<td>Enables supply of input clock.</td>
</tr>
</tbody>
</table>

Setting up the IICA0 operation mode

- IICA control register 01 (IICCTL01)
  Select an operation clock frequency.
  Turn the digital filter on.
  Select the fast mode.
  Disable the wakeup function.

Symbol: IICCTL01

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>WUP0</td>
<td>0</td>
<td>CLD0</td>
<td>DAD0</td>
<td>SMC0</td>
<td>DFC0</td>
<td>0</td>
<td>PRS0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Bit 7

<table>
<thead>
<tr>
<th>WUP0</th>
<th>Address match wakeup control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Disable the address match wakeup function in STOP mode.</td>
</tr>
<tr>
<td>1</td>
<td>Enable the address match wakeup function in STOP mode.</td>
</tr>
</tbody>
</table>

Bit 3

<table>
<thead>
<tr>
<th>SMC0</th>
<th>Operation mode selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Standard mode</td>
</tr>
<tr>
<td>1</td>
<td>Fast mode</td>
</tr>
</tbody>
</table>

Bit 2

<table>
<thead>
<tr>
<th>DFC0</th>
<th>Digital filter operation control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Turns the digital filter off.</td>
</tr>
<tr>
<td>1</td>
<td>Turns the digital filter on.</td>
</tr>
</tbody>
</table>

Bit 0

<table>
<thead>
<tr>
<th>PRS0</th>
<th>Operation clock frequency selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Selects fCLK as the operation clock frequency.</td>
</tr>
<tr>
<td>1</td>
<td>Selects fCLK/2 as the operation clock frequency.</td>
</tr>
</tbody>
</table>

Configuring the transfer clock

- IICA low-level width setting register 0 (IICWL0)
- IICA high-level width setting register 0 (IICWH0)
  
  Set the low-level width and high-level width of the SCLA0 pin signal.

Symbol: IICWL0

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Symbol: IICWH0

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>


Setting the local address

- Slave address register 0 (SVA0)
  
  Set the local address.

Symbol: SVA0

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Setting up the IICA operation

- IICA control register 00 (IICCTL00)
  - Enable I2C operation.
  - Disable stop condition interrupts.
  - Set the wait and interrupt request generation timing.
  - Enable acknowledgement output.

Symbol: IICCTL00

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>IICE0</th>
<th>Enabling/disabling of I2C operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x</td>
<td>Stops operation. Resets the IICA status register 0 (IICS0). Also stops internal operation.</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Enables operation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 6</th>
<th>LREL0</th>
<th>Transition from the communication state</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x</td>
<td>Normal operation</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Makes a transition from the current communication state to the standby state. Automatically cleared to 0 after the transition.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 4</th>
<th>SPIE0</th>
<th>Enabling/disabling generation of interrupt requests due to stop condition detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Disabled</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Enabled</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 3</th>
<th>WTIM0</th>
<th>Wait/interrupt request control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x</td>
<td>Interrupt request is generated at the falling edge of the eighth clock signal. Waits with the clock output remaining at low level, after eight clock signals are output.</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Interrupt request is generated at the falling edge of the ninth clock signal. Waits with the clock output remaining at low level, after nine clock signals are output.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 2</th>
<th>ACKE0</th>
<th>Acknowledgement control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x</td>
<td>Disables acknowledgements.</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Enables acknowledgements.</td>
</tr>
</tbody>
</table>

Setting up the IICA pins
- Port register 6 (P6)
- Port mode register 6 (PM6)
  Use P60 for SCLA0 and P61 for SDAA0 in output mode.

Symbol: P6

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Bit 1

<table>
<thead>
<tr>
<th>P61</th>
<th>Output data control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Output 0</td>
</tr>
<tr>
<td>1</td>
<td>Output 1</td>
</tr>
</tbody>
</table>

Bit 0

<table>
<thead>
<tr>
<th>P60</th>
<th>Output data control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Output 0</td>
</tr>
<tr>
<td>1</td>
<td>Output 1</td>
</tr>
</tbody>
</table>

Symbol: PM6

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Bit 1

<table>
<thead>
<tr>
<th>PM61</th>
<th>P61 I/O mode selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Output mode (output buffer on)</td>
</tr>
<tr>
<td>1</td>
<td>Input mode (output buffer off)</td>
</tr>
</tbody>
</table>

Bit 0

<table>
<thead>
<tr>
<th>PM60</th>
<th>P60 I/O mode selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Output mode (output buffer on)</td>
</tr>
<tr>
<td>1</td>
<td>Input mode (output buffer off)</td>
</tr>
</tbody>
</table>

5.7.6 Main Processing

Figure 5.7 shows the flowchart for main processing.

![Flowchart for main processing](image)

Figure 5.7 Main Processing
Setting up the wakeup function

- IICA control register 01 (IICTL01)
  Enable the wakeup function.

Symbol: IICTL01

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>WUP0</td>
<td>CLD0</td>
<td>DAD0</td>
<td>SMC0</td>
<td>DFC0</td>
<td>0</td>
<td>PRS0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Bit 7

<table>
<thead>
<tr>
<th>WUP0</th>
<th>Address match wakeup control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Disables the address match wakeup function in STOP mode</td>
</tr>
<tr>
<td>1</td>
<td>Enables the address match wakeup function in STOP mode</td>
</tr>
</tbody>
</table>


Communication termination setting

- IICA control register 00 (IICTL00)
  Terminate the current communication, and make the system enter a communication standby state

Symbol: IICTL00

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>IICE0</td>
<td>LREL0</td>
<td>WREL0</td>
<td>SPIE0</td>
<td>WTIM0</td>
<td>ACKE0</td>
<td>STT0</td>
<td>SPT0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>x</td>
<td>0/1</td>
<td>0/1</td>
<td>0/1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Bit 6

<table>
<thead>
<tr>
<th>LREL0</th>
<th>Transition from the communication state</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Normal operation</td>
</tr>
<tr>
<td>1</td>
<td>Makes a transition from the current communication state to the standby state</td>
</tr>
</tbody>
</table>

5.7.7 IICA0 Reception Preparation Function

Figure 5.8 shows the flowchart for the IICA0 reception preparation function.

- **R_RxPreparation()**
  - Set transfer direction flag to indicate reception
  - Configure slave reception settings
    - R_IICA0_Slave_Receive()
  - Set receive data counter to 16.
  - Clear receive data counter to 0.
  - Set receive buffer address.
  - Set IIC slave status to 0 (for address transfer).

```
Figure 5.8  IICA0 Reception Preparation Function
```
5.7.8 IICA0 Transmission Preparation Function

Figure 5.9 shows the flowchart for the IICA0 transmission preparation function.

```
R_TxPreparation()

Set transfer direction flag to indicate transmission

Add 16 to each element of receive buffer array

Configure slave transmission settings R_IICA0_Slave_Send()

Set transmit data counter to 16.
Set transmit buffer address.
Set IIC slave status to 0 (for address transfer).

return
```

Figure 5.9 IICA0 Transmission Preparation Function
5.7.9 IICA0 Interrupt Processing

Figure 5.10 shows the flowchart for IICA0 interrupt processing.

![Flowchart for IICA0 Interrupt Processing]

Figure 5.10 IICA0 Interrupt Processing
5.7.10 IICA0 Slave Handler

Figures 5.11 through 5.13 show the flowcharts for the IICA0 slave handler.

Figure 5.11 IICA0 Slave Handler (1/3)
Figure 5.12   IICA0 Slave Handler (2/3)

- Transmission mode?
  - No
  - Communication ended?
    - No: Check if there is still transmit data remaining without ACK being received from master.
    - Yes: Write transmit data to IICA0 register and decrement transmit data counter by 1.
  - Yes: Return error flag `r_iica0_callback_slave_error()`
- Transmit data found?
  - Yes: Cancel wait
  - No: Write transmit data

---

WREL0 bit ← 1
Figure 5.13   IICA0 Slave Handler (3/3)
5.7.11 Error Flag Return Processing

Figure 5.14 shows the flowchart for error flag return processing.

![Flowchart for error flag return processing]

Figure 5.14 Error Flag Return Processing
6. Sample Code

The sample code is available on the Renesas Electronics Website.

7. Documents for Reference

User’s Manual:
- RL78/G13 User's Manual: Hardware (R01UH0146EJ)
  The latest version can be downloaded from the Renesas Electronics website.

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

Website and Support

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

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>Apr. 16, 2015</td>
<td>First edition issued</td>
<td>—</td>
<td></td>
</tr>
</tbody>
</table>
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 document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins
   Handle unused pins in accordance 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 a product with a different part number, confirm that the change will not lead to problems.
   — The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.
Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical equipment; and industrial robots etc.)

9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or contents and conditions set forth in this document, Renesas Electronics assumes no liability for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.

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

11. 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.

© 2015 Renesas Electronics Corporation. All rights reserved.

Colophon 5.0