RL78/G23
SMS HS300x Humidity sensor control by I2C communication

Introduction
This application note describes an example of controlling a humidity sensor (HS300x) with RL78/G23 to measure indoor air quality. It uses the SNOOZE mode sequencer (SMS), data transfer controller (DTC) and serial interface IICA (IICA) to control the HS300x with the I2C communication protocol during SNOOZE mode.

I2C communication via SMS allows communication with lower power consumption than when processed by the CPU.

Target Device
RL78/G23

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. Conditions for Operation Confirmation Test .............................................................................. 6

3. Related application note ............................................................................................................ 7

4. Hardware ................................................................................................................................... 8
   4.1 Example of Hardware Configuration ............................................................................................ 8
   4.2 Used Pins ................................................................................................................................................ 9

5. Software .................................................................................................................................. 10
   5.1 Overview of the sample program .......................................................................................................... 10
   5.2 Folder Configuration .............................................................................................................................. 11
   5.3 Option Byte Settings .............................................................................................................................. 12
   5.4 Constants .............................................................................................................................................. 12
   5.5 Variables ................................................................................................................................................ 13
   5.6 Functions ............................................................................................................................................... 13
   5.7 Function Specifications ......................................................................................................................... 13
   5.8 Flow Charts ........................................................................................................................................... 16
   5.8.1 SMS operation start processing .......................................................................................................... 17
   5.8.2 SMS status confirmation process ........................................................................................................ 18
   5.8.3 SMS interrupt process ......................................................................................................................... 18
   5.8.4 IICA0 status confirmation process ...................................................................................................... 18
   5.8.5 Data copy process ............................................................................................................................... 19
   5.9 SNOOZE Mode Sequencer settings ..................................................................................................... 20

6. Application example ................................................................................................................ 27
   6.1 r01an6065jj0110_sms_humidity_sensor.scfg ....................................................................................... 27
   6.1.1 Clocks .................................................................................................................................................. 29
   6.1.2 System ................................................................................................................................................. 29
   6.1.3 r_bsp.................................................................................................................................................... 29
   6.1.4 Config_LVD0 ....................................................................................................................................... 29
   6.1.5 Config_DTC ....................................................................................................................................... 29
   6.1.6 Config_IT000_ITL001 .......................................................................................................................... 29
   6.1.7 Config_PORT ...................................................................................................................................... 29
   6.1.8 Config_IICA0 ....................................................................................................................................... 29

7. Sample Code ................................................................................................................................ 30

8. Reference ................................................................................................................................ 30

Revision History .............................................................................................................................. 31
1. Specifications

The SMS sequentially executes up to 32 processes set in the sequencer instruction register SMSI0 to SMSI31 in advance. However, the processing according to the general I2C communication standard is 32 steps or more. Therefore, this application note shows how to perform I2C communication by dividing the processes to be executed in the SMS into three and sequentially updating the SMSI register during the SMS operation. The DTC is used to update the SMSI register.

Figure 1-1 shows the memory map. SMSI0 to SMSI31 are assigned to the extended special function register (2nd SFR) area.

The process to be used by the SMS is stored in RAM in advance ((1), (2) in Figure 1-1). The interval for I2C communication is set using the interval timer (TML32), and SMS is started by a TML32 interrupt (INTITL). After the SMS is started, the DMATRG instruction starts the DTC and transfers the data from RAM to the registers of the SMS. The SMS executes the transferred processes sequentially and performs I2C communication ((3) - (5) in Figure 1-1).

The humidity sensor (HS300x) used in this application note can determine the humidity from the received data (Humidity[13:0]) using the following equation.

\[
\text{Humidity[%RH]} = \left( \frac{\text{Humidity[13:0]}}{2^{14} - 1} \right) \times 100
\]

The sample code compares the value of Humidity[13:8] of the received data with the threshold value. The threshold value in the sample code is set to 20H, which corresponds to 50% humidity.

For details on HS300x, refer to the HS300x manual.
Figure 1-2 shows an example of the system configuration, and Figure 1-3 shows the flowchart of the entire system.

Figure 1-2 System Configuration
Figure 1-3 Entire Flowchart

**CPU**

1. main()
2. Initialize SMS
3. Copy the code for SMS from Code Flash to RAM
4. IICAMK0 = 1
5. DTC Start
6. Interval Timer Start
7. EI()
8. Waiting for the sequencer to start
9. Shifted to STOP mode
10. INTSMSE occurred
11. SMS Start
12. Wake up
13. Calculate the humidity
14. END

**SMS**

1. INTITL occurred SMS Start
2. 1st DTC transfer
3. IICAO generate start condition
4. Sending slave address
5. IICAO generate stop condition
6. Wait 62.5ms
7. 2nd DTC transfer
8. IICAO generate start condition
9. Receiving data
10. 3rd DTC transfer
11. IICAO generate stop condition
12. Above threshold?
13. Yes
   - Shifted to STOP mode
14. No
   - g_sms_wakeup_flag = 1
   - RETI
   - LED = ON
   - Calculate the humidity
   - END

**Notes**

- g_sms_wakeup_flag = 1
- LED = ON
- Calculate the humidity
- END
2. Conditions for Operation Confirmation Test

The sample code with this application note runs properly under the condition below.

Table 2-1 Operation Confirmation Conditions

<table>
<thead>
<tr>
<th>Items</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCU</td>
<td>RL78/G23 (R7F100GLG)</td>
</tr>
<tr>
<td>Operating frequencies</td>
<td>• High-speed on-chip oscillator clock: 32 MHz</td>
</tr>
<tr>
<td>Operating voltage</td>
<td>• CPU/peripheral hardware clock: 32 MHz</td>
</tr>
<tr>
<td>Operating voltage</td>
<td>• 3.3V</td>
</tr>
<tr>
<td>Integrated development environment (CS+)</td>
<td>CS+ for CC V 8.07.00 from Renesas Electronics Corp.</td>
</tr>
<tr>
<td>C compiler (CS+)</td>
<td>CC-RL V1.11 from Renesas Electronics Corp.</td>
</tr>
<tr>
<td>Integrated development environment (e² studio)</td>
<td>e² studio 2022-01 (22.1.0) from Renesas Electronics Corp.</td>
</tr>
<tr>
<td>C compiler (e² studio)</td>
<td>CC-RL V1.11 from Renesas Electronics Corp.</td>
</tr>
<tr>
<td>Integrated development environment (IAR)</td>
<td>IAR Embedded Workbench for Renesas RL78 v4.21.1 from IAR Systems</td>
</tr>
<tr>
<td>C compiler (IAR)</td>
<td>V.1.2.0</td>
</tr>
<tr>
<td>Smart Configurator</td>
<td>V.1.2.0</td>
</tr>
<tr>
<td>Board support package (r_bsp)</td>
<td>V.1.13</td>
</tr>
<tr>
<td>Emulator</td>
<td>CS+ , e² studio: COM port</td>
</tr>
<tr>
<td></td>
<td>IAR: E2 Emulator Lite</td>
</tr>
<tr>
<td>Board</td>
<td>RL78/G23 Fast Prototyping Board (RTK7RLG230CLG000BJ)</td>
</tr>
</tbody>
</table>
3. Related application note

The following application note is related to this application note.

Please refer to them as well.
4. Hardware

4.1 Example of Hardware Configuration

Figure 4-1 shows an example of the hardware configuration in this application.

Caution 1. This simplified circuit diagram was created to show an overview of connections only. When actually designing your circuit, make sure the design includes sufficient pin processing and meets electrical characteristic requirements. (Connect each input-only port to VDD or VSS through a resistor.)

Caution 2. Connect the EVSS pin to VSS and the EVDD pin to VDD.

Caution 3. VDD must be held at not lower than the reset release voltage (VLVD0) that is specified as LVD.

This application note uses a humidity sensor (HS300x). For details on HS300x, refer to the HS300x manual.
4.2 Used Pins

Table 4-1 shows list of used pins and assigned functions.

Table 4-1 List of Pins and Functions

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Input/Output</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>P53</td>
<td>Output</td>
<td>LED1 lights (Low Active)</td>
</tr>
<tr>
<td>P60</td>
<td>Input/Output</td>
<td>Clock I/O pin of serial interfaces IICA0</td>
</tr>
<tr>
<td>P61</td>
<td>Input/Output</td>
<td>Serial data I/O pin of serial interfaces IICA0</td>
</tr>
</tbody>
</table>

Caution. In this application note, only the used pins are processed. When actually designing your circuit, make sure the design includes sufficient pin processing and meets electrical characteristic requirements.
5. Software

5.1 Overview of the sample program

In this sample code, an interrupt request (INTITL) from the 32-bit interval timer (TML32) causes the device to shift from STOP mode to SNOOZE mode, and starts the DTC from the SNOOZE mode sequencer (SMS). The DTC transfers the code stored in RAM in advance to the sequencer instruction register p (SMSIp (p = 3-31)). The SMS executes the transferred code and performs I\(^2\)C communication to acquire data from the sensor and compare it with the threshold value. If the threshold value is exceeded, the SMS outputs an interrupt request signal (INTSMSE) to activate the CPU.

Table 5-1 shows an overview of the operation of this sample code.

Table 5-1 Operation Overview

<table>
<thead>
<tr>
<th>No.</th>
<th>Operation of each function</th>
<th>CPU</th>
<th>SMS</th>
<th>DTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>Stores code for SMS from Code Flash to RAM</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>(2)</td>
<td>Starts counting TML32</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Waits for startup trigger</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3)</td>
<td>Shifts to STOP mode</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Starts SMS with TML32 compare match</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(4)</td>
<td>During STOP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Runs the DTC startup process</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(5)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(6)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(7)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(8)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(9)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(11)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(12)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(13)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(14)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(15)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(16)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(17)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 5.2 Folder Configuration

Table 5-2 shows folder configuration of source file and header files using by sample code except the files generated by integrated development environment and the files in the bsp environment.

#### Table 5-2 Folder configuration

<table>
<thead>
<tr>
<th>Folder/File configuration</th>
<th>Outline</th>
<th>Created by Smart configurator</th>
</tr>
</thead>
<tbody>
<tr>
<td>\</td>
<td></td>
<td></td>
</tr><tr>
<td>01an6065jj0110_sms_humidity_sensor&lt;DIR&gt;</td>
<td>Root folder of this sample code</td>
<td></td>
</tr>
<tr>
<td>src&lt;DIR&gt;</td>
<td>Folder for program source</td>
<td></td>
</tr>
<tr>
<td>main.c</td>
<td>Sample code source file</td>
<td></td>
</tr>
<tr>
<td>sms_p.c</td>
<td>Code storage file for SMS</td>
<td></td>
</tr>
<tr>
<td>SMS&lt;DIR&gt;</td>
<td>Folder for SMS program</td>
<td></td>
</tr>
<tr>
<td>r_sms.c</td>
<td>Source file for SMS</td>
<td></td>
</tr>
<tr>
<td>r_sms.h</td>
<td>Header file for SMS</td>
<td></td>
</tr>
<tr>
<td>r_sms_ASM.smsasm</td>
<td>ASM source file for SMS</td>
<td></td>
</tr>
<tr>
<td>smc_gen&lt;DIR&gt;</td>
<td>Folder created by Smart Configurator</td>
<td>✓</td>
</tr>
<tr>
<td>Config_DTC&lt;DIR&gt;</td>
<td>Folder for DTC program</td>
<td>✓</td>
</tr>
<tr>
<td>Config_DTC.c</td>
<td>Source file for DTC</td>
<td>✓</td>
</tr>
<tr>
<td>Config_DTC.h</td>
<td>Header file for DTC</td>
<td>✓</td>
</tr>
<tr>
<td>Config_DTC_user.c</td>
<td>Interrupt source file for DTC</td>
<td>✓ Note 1</td>
</tr>
<tr>
<td>Config_IICA0&lt;DIR&gt;</td>
<td>Folder for IICA0 program</td>
<td>✓</td>
</tr>
<tr>
<td>Config_IICA0.c</td>
<td>Source file for IICA0</td>
<td>✓</td>
</tr>
<tr>
<td>Config_IICA0.h</td>
<td>Header file for IICA0</td>
<td>✓</td>
</tr>
<tr>
<td>Config_IICA0_user.c</td>
<td>Interrupt source file for IICA0</td>
<td>✓ Note 1</td>
</tr>
<tr>
<td>Config_ITL000_ITL001&lt;DIR&gt;</td>
<td>Folder for TML32 program</td>
<td>✓</td>
</tr>
<tr>
<td>Config_ITL000_ITL001.c</td>
<td>Source file for TML32</td>
<td>✓</td>
</tr>
<tr>
<td>Config_ITL000_ITL001.h</td>
<td>Header file for TML32</td>
<td>✓</td>
</tr>
<tr>
<td>Config_ITL000_ITL001_user.c</td>
<td>Interrupt source file for TML32</td>
<td>✓ Note 1</td>
</tr>
<tr>
<td>Config_PORT&lt;DIR&gt;</td>
<td>Folder for PORT program</td>
<td>✓</td>
</tr>
<tr>
<td>Config_PORT.c</td>
<td>Source file for PORT</td>
<td>✓</td>
</tr>
<tr>
<td>Config_PORT.h</td>
<td>Header file for PORT</td>
<td>✓</td>
</tr>
<tr>
<td>Config_PORT_user.c</td>
<td>Interrupt source file for PORT</td>
<td>✓ Note 1</td>
</tr>
<tr>
<td>general&lt;DIR&gt;</td>
<td>Folder for initialize or common program</td>
<td>✓</td>
</tr>
<tr>
<td>r_bsp&lt;DIR&gt;</td>
<td>Folder for BSP program</td>
<td>✓</td>
</tr>
<tr>
<td>r_config&lt;DIR&gt;</td>
<td>Folder for BSP_CFG program</td>
<td>✓</td>
</tr>
</tbody>
</table>

Note. <DIR> means directory.
Note 1. Not used in this sample code.
Note 2. The sample code of the IAR version has a different configuration. Check the sample code of the IAR version for details. In addition, stores r01an6065jj0110_sms_humidity_sensor.ipcf. For details, refer to "RL78 Smart Configurator User’s Guide: IAREW (R20AN0581)".
5.3 Option Byte Settings

Table 5-3 shows the option byte settings.

<table>
<thead>
<tr>
<th>Address</th>
<th>Setting Value</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>000C0H/040C0H</td>
<td>1110 1111B (EFH)</td>
<td>Operation of Watchdog timer is stopped (counting is stopped after reset)</td>
</tr>
<tr>
<td>000C1H/040C1H</td>
<td>1111 1110B (FEH)</td>
<td>LVD operating mode: reset mode Rising edge 1.90V (1.84V-1.95V)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Falling edge 1.86V (1.80V-1.91V)</td>
</tr>
<tr>
<td>000C2H/040C2H</td>
<td>1110 1000B (E8H)</td>
<td>Flash operating mode: HS mode High-speed on-chip oscillator clock: 32MHz</td>
</tr>
<tr>
<td>000C3H/040C3H</td>
<td>1000 0101B (85H)</td>
<td>On-chip debugging is enabled</td>
</tr>
</tbody>
</table>

5.4 Constants

Table 5-4 shows the constants that are used in this sample code.

<table>
<thead>
<tr>
<th>Constant Name</th>
<th>Setting Value</th>
<th>Contents</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td>LED1</td>
<td>P5_bit.no3</td>
<td>P53</td>
<td>main.c</td>
</tr>
<tr>
<td>LED_ON</td>
<td>0</td>
<td>Setting value for turning on the LED</td>
<td>main.c</td>
</tr>
<tr>
<td>THRESHOLD</td>
<td>20H</td>
<td>Humidity threshold (Upper 2 bytes)</td>
<td>main.c</td>
</tr>
<tr>
<td>DATA_BYTE</td>
<td>27<em>2</em>3</td>
<td>Number of bytes of code to be transferred to SMS (27*2byte)*3times = 162</td>
<td>main.c</td>
</tr>
<tr>
<td>NUMBER_OF_DATA</td>
<td>4</td>
<td>Sensor data (bytes)</td>
<td>main.c</td>
</tr>
<tr>
<td>ERROR</td>
<td>FFFFH</td>
<td>Value at communication error</td>
<td>main.c</td>
</tr>
<tr>
<td>NO_DATA</td>
<td>-1</td>
<td>Value at communication error</td>
<td>main.c</td>
</tr>
<tr>
<td>sensor_add[]</td>
<td>88H</td>
<td>Address of the sensor (for sending)</td>
<td>main.c</td>
</tr>
<tr>
<td></td>
<td>89H</td>
<td>Address of the sensor (for receiving)</td>
<td>main.c</td>
</tr>
<tr>
<td>g_dtc_data[]</td>
<td>Note</td>
<td>Code to transfer to the SMS</td>
<td>sms_p.c</td>
</tr>
</tbody>
</table>

Note. For details, refer to Table 5-7 - Table 5-10.
5.5 Variables
Table 5-5 shows the global variables used in this sample code.

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable name</th>
<th>contents</th>
<th>Functions used in</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>g_result</td>
<td>Humidity calculation results</td>
<td>main.c</td>
</tr>
<tr>
<td>uint8_t</td>
<td>g_sms_wakeup_flag</td>
<td>SMS wakeup flag</td>
<td>r_sms_getstatus</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>r_sms_start</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>r_sms_interrupt</td>
</tr>
<tr>
<td>uint16_t</td>
<td>g_communication_status</td>
<td>IICA0 status flag</td>
<td>r_iica0_getstatus</td>
</tr>
</tbody>
</table>

5.6 Functions
Table 5-6 shows the functions used in the sample code. However, the unchanged functions generated by the Smart Configurator are excluded.

<table>
<thead>
<tr>
<th>Function name</th>
<th>Outline</th>
<th>Source file</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>Main process</td>
<td>main.c</td>
</tr>
<tr>
<td>r_sms_create</td>
<td>SMS initialization process</td>
<td>r_SMS.c</td>
</tr>
<tr>
<td>r_sms_start</td>
<td>SMS operation start processing</td>
<td>r_SMS.c</td>
</tr>
<tr>
<td>r_sms_getstatus</td>
<td>SMS status confirmation process</td>
<td>r_SMS.c</td>
</tr>
<tr>
<td>r_sms_interrupt</td>
<td>SMS interrupt process</td>
<td>r_SMS.c</td>
</tr>
<tr>
<td>r_iica0_getstatus</td>
<td>IICA0 status confirmation process</td>
<td>Config_IICA0_user.c</td>
</tr>
<tr>
<td>memcpy_ff</td>
<td>Data copy process</td>
<td>main.c</td>
</tr>
</tbody>
</table>

Note: IAR version only. CC-RL version uses the standard function “_COM_memcpy_ff”.

5.7 Function Specifications
This part describes function specifications of the sample code.

[Function name] main

Outline: Main process
Header: r_smc_entry.h, string.h, r_sms.h
Declaration: void main (void);
Description: Initializes the SMS and transfers the code (g_dtc_data[]) to be transferred by the DTC at SMS startup from the code flash area to RAM. Starts the operation of the TML32, puts the DTC in the trigger waiting state, and shifts to the STOP mode. When returning from standby mode, the process shifts to checking the CPU boot request flag. If the CPU boot request flag (g_sms_wakeup_flag) is not set, the mode shifts to STOP mode again, else if the CPU boot request flag is set, LED1 lights up.

Arguments: None
Return value: None
Remarks: None
**[Function name] r_sms_create**

**Outline**
Initialization process for SMS

**Header**
r_SMS.h, r_cg_macrodriver.h, r_cg_userdefine.h, r_SMS_ASM.h

**Declaration**
void r_sms_create (void)

**Description**
Initializes the SMS module before controlling it, including SMS configuration, copying SMS instructions, and copying SMS data.

**Arguments**
None

**Return value**
None

**Remarks**
None

---

**[Function name] r_sms_start**

**Outline**
Start SMS operation

**Header**
r_SMS.h, r_cg_macrodriver.h, r_cg_userdefine.h, r_SMS_ASM.h

**Declaration**
void r_sms_start (uint16_t addr_iic_sl, uint16_t addr_iic_rx_s, uint16_t addr_iic_rx_e, uint16_t val_threshold)

**Description**
Sets the SMS data from the argument and starts the SMS module operation.

**Arguments**
addr_iic_sl, addr_iic_rx_s, addr_iic_rx_e, val_threshold

**Return value**
None

**Remarks**
None

---

**[Function name] r_sms_getstatus**

**Outline**
Check the status of SMS wakeup

**Header**
r_SMS.h, r_cg_macrodriver.h, r_cg_userdefine.h, r_SMS_ASM.h

**Declaration**
uint8_t r_sms_getstatus (void)

**Description**
Checks the status of SMS wakeup.

**Arguments**
None

**Return value**
None

**Remarks**
None

---

**[Function name] r_sms_interrupt**

**Outline**
INTSMSE Interrupt process

**Header**
r_SMS.h, r_cg_macrodriver.h, r_cg_userdefine.h, r_SMS_ASM.h

**Declaration**
#pragma interrupt r_sms_interrupt(vect=INTSMSE)

**Description**
Sets g_sms_wakeup_flag = 1.

**Arguments**
None

**Return value**
None

**Remarks**
None

---

**[Function name] r_iica0_getstatus**

**Outline**
IICA0 status confirmation process

**Header**
r_cg_macrodriver.h, r_cg_userdefine.h, Config_IICA0.h

**Declaration**
uint16_t r_iica0_getstatus (void) (void)

**Description**
Checks the status of IICA0.

**Arguments**
None

**Return value**
None

**Remarks**
None
### Function: memcpy_ff

<table>
<thead>
<tr>
<th>Outline</th>
<th>Copy Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td><code>r_smc_entry.h, r_SMS.h</code></td>
</tr>
<tr>
<td>Declaration</td>
<td><code>__far_func void __far * memcpy_ff(void __near * __Restrict s1, const void __far * __Restrict s2, uint32_t n)</code></td>
</tr>
<tr>
<td>Description</td>
<td>Copies the data in s2 to s1 for n bytes.</td>
</tr>
<tr>
<td>Arguments</td>
<td>s1, s2, n</td>
</tr>
<tr>
<td>Return value</td>
<td>None</td>
</tr>
<tr>
<td>Remarks</td>
<td>None</td>
</tr>
</tbody>
</table>
5.8 Flow Charts

Main Process

Figure 5-1 shows flowchart of main process.

Figure 5-1 Main process
SMS initialization process

Figure 5-2 shows flowchart of SMS initialization process.

5.8.1 SMS operation start processing

Figure 5-3 shows flowchart of SMS operation start processing.
5.8.2 SMS status confirmation process
Figure 5-4 shows flowchart of SMS status confirmation process.

Figure 5-4 SMS status confirmation process

```
<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>r_sms_getstatus()</td>
</tr>
<tr>
<td>Return g_sms_wakeup_flag</td>
</tr>
<tr>
<td>End</td>
</tr>
</tbody>
</table>
```

5.8.3 SMS interrupt process
Figure 5-5 shows flowchart of SMS interrupt process.

Figure 5-5 SMS interrupt process

```
<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>r_sms_interrupt()</td>
</tr>
<tr>
<td>g_sms_wakeup_flag = 1</td>
</tr>
<tr>
<td>End</td>
</tr>
</tbody>
</table>
```

5.8.4 IICA0 status confirmation process
Figure 5-6 shows flowchart of IICA0 status confirmation process.

Figure 5-6 IICA0 status confirmation process

```
<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>r_iica0_getstatus()</td>
</tr>
<tr>
<td>Return g_communication_status</td>
</tr>
<tr>
<td>End</td>
</tr>
</tbody>
</table>
```
5.8.5 Data copy process

Figure 5-7 shows flowchart of Data copy process.

Figure 5-7 Data copy process

```
memcpy_ff()

Copy the Data from s2 to s1

End
```
5.9 SNOOZE Mode Sequencer settings

When the event set in the start trigger occurs, SMS executes the processing commands stored in the sequencer instruction register (SMSI0-31) in order. When executing a processing command, the Sequencer general-purpose register (SMSG0-15) is used to store the source address, destination address, calculated data, and so on.

SMSI0-31 and SMSG0-15 are set by writing the SMS program (.SMSASM file) in assembly language. The created SMS program is converted to a C language file by the SMS assembler and incorporated into the program.

The specifications of SMS processing executed by the sample code are shown below.

<table>
<thead>
<tr>
<th>Outline</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS process</td>
<td></td>
</tr>
<tr>
<td>The SMS is activated by the TML32 interrupt, sends MR commands to the sensor, and waits for 62.5ms for the sensor to stabilize its operation. After waiting for 4 seconds, it obtains the measurement results from the sensor and stores them in RAM. If the humidity data stored in RAM is greater than the threshold value, the CPU is activated. If it is less than the threshold value, the SMS process is terminated and the device shifts to STOP mode again. The instructions to process I2C communication are stored in RAM in advance, and then transferred and executed in three separate times by DTC.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Arguments Note1</th>
</tr>
</thead>
<tbody>
<tr>
<td>addr_iic_sl: The start address of the buffer that stores the slave address of the sensor</td>
</tr>
<tr>
<td>addr_iic_rx_s: Start address of data storage area</td>
</tr>
<tr>
<td>addr_iic_rx_e: End address of data storage area</td>
</tr>
<tr>
<td>val_threshold: Threshold value of humidity data</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return value</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Note1. Argument to be specified in the r_sms_start function setting.

Figure 5-8 shows the flow chart of SMS process.

Table 5-7 to Table 5-10 show the register settings that control the SNOOZE mode sequencer.

The SMS is controlled according to the HS300x protocol using I2C communication. For details on the HS300x protocol, refer to the HS300x manual.
Figure 5-8 SMS process

1. Start
2. 1st DTC transfer
3. Flag clear
4. Initialize SMSG10, SMSG11
5. Generate start condition
6. Sending slave address (MR Command)
7. ACK detection?
   - No
   - Yes
     - Generate stop condition
     - Wait 62.5ms

DTCTRG
Clear ITF00
IICCTL00.STT0 = 1,
Wait for start condition
IICA0 = Slave address (for sending)
Wait IF1L.IICAIF0 = 1
IICCTL00.SPT0 = 1
Figure 5-9 SMS process

- **DTCTRG**
  - SMSG11 = Data table start address
  - IICCTL00.STT0 = 1, Wait for start condition
  - IICA0 = Slave address (for receiving), Wait IF1L.IICAIF0 = 1

- **IICCTL00.ACKE0 = 1, IICCTL00.WTIM0 = 0**

- **IICCTL00.WREL0 = 1, Wait IF1L.IICAIF0 = 1, SMSG10 = IICA0**

- **SMSG11 = SMSG10**

- **SMSG11 = SMSG11 + 1**

- **SMSG11 >= Data table end address**

- **IICCTL00.ACKE0 = 0, IICCTL00.WTIM0 = 1, IICCTL00.WREL0 = 1**

- **IICCTL00.SPT0 = 1, WAIT**
### Table 5-7 Sequencer general-purpose registers 0-15

<table>
<thead>
<tr>
<th>Register Symbol</th>
<th>Setting</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMSG0</td>
<td>0000H</td>
<td>fixed value: 0000H</td>
</tr>
<tr>
<td>SMSG1</td>
<td>&amp;ITLS0</td>
<td>ITLS0 address</td>
</tr>
<tr>
<td>SMSG2</td>
<td>&amp;IICCTL00</td>
<td>IICCTL00 address</td>
</tr>
<tr>
<td>SMSG3</td>
<td>&amp;IF1L</td>
<td>IF1L address</td>
</tr>
<tr>
<td>SMSG4</td>
<td>&amp;IICA0</td>
<td>IICA0 address</td>
</tr>
<tr>
<td>SMSG5</td>
<td>0000H</td>
<td>Slave address of sensor</td>
</tr>
<tr>
<td>SMSG6</td>
<td>0000H</td>
<td>Data table start address</td>
</tr>
<tr>
<td>SMSG7</td>
<td>0000H</td>
<td>Data table end address</td>
</tr>
<tr>
<td>SMSG8</td>
<td>1</td>
<td>fixed value: 1</td>
</tr>
<tr>
<td>SMSG9</td>
<td>0000H</td>
<td>Humidity threshold</td>
</tr>
<tr>
<td>SMSG10</td>
<td>0000H</td>
<td>for calculation</td>
</tr>
<tr>
<td>SMSG11</td>
<td>0000H</td>
<td>for calculation</td>
</tr>
<tr>
<td>SMSG12</td>
<td>&amp;SMSG1</td>
<td>SMSG1 address</td>
</tr>
<tr>
<td>SMSG13</td>
<td>&amp;SMSI28</td>
<td>SMSI28 address</td>
</tr>
<tr>
<td>SMSG14</td>
<td>&amp;SMSG11</td>
<td>SMSG11 address</td>
</tr>
<tr>
<td>SMSG15</td>
<td>FFFFH</td>
<td>fixed value: FFFFH</td>
</tr>
<tr>
<td>Register Symbol</td>
<td>Setting</td>
<td>Contents</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SMSI0</td>
<td>2DF2H</td>
<td>MOVW [MSG13+2], MSG15</td>
</tr>
<tr>
<td>SMSI1</td>
<td>F002H</td>
<td>DTCTRГ</td>
</tr>
<tr>
<td>SMSI2</td>
<td>AD02H</td>
<td>WHILE1 [MSG13+2].0</td>
</tr>
<tr>
<td>SMSI3</td>
<td>0100H</td>
<td>MOV [MSG1+0], MSG0</td>
</tr>
<tr>
<td>SMSI4</td>
<td>30A0H</td>
<td>MOVW MSG10, [MSG0+0]</td>
</tr>
<tr>
<td>SMSI5</td>
<td>30B0H</td>
<td>MOVW MSG11, [MSG0+0]</td>
</tr>
<tr>
<td>SMSI6</td>
<td>4210H</td>
<td>SET1 [MSG2+0].1</td>
</tr>
<tr>
<td>SMSI7</td>
<td>9640H</td>
<td>WAIT 100, 0</td>
</tr>
<tr>
<td>SMSI8</td>
<td>15A0H</td>
<td>MOV MSG10, [MSG5+0]</td>
</tr>
<tr>
<td>SMSI9</td>
<td>04A0H</td>
<td>MOV [MSG4+0], MSG10</td>
</tr>
<tr>
<td>SMSI10</td>
<td>B330H</td>
<td>WHILE0 [MSG3+0].3</td>
</tr>
<tr>
<td>SMSI11</td>
<td>5330H</td>
<td>CLR1 [MSG3+0].3</td>
</tr>
<tr>
<td>SMSI12</td>
<td>6421H</td>
<td>MOV1 SCY, [MSG4+1].2</td>
</tr>
<tr>
<td>SMSI13</td>
<td>8051H</td>
<td>BNC 5</td>
</tr>
<tr>
<td>SMSI14</td>
<td>4200H</td>
<td>SET1 [MSG2+0].0</td>
</tr>
<tr>
<td>SMSI15</td>
<td>9804H</td>
<td>WAIT 128, 4</td>
</tr>
<tr>
<td>SMSI16</td>
<td>7FF2H</td>
<td>CMPW MSG15, MSG15</td>
</tr>
<tr>
<td>SMSI17</td>
<td>8EF2H</td>
<td>BZ -17</td>
</tr>
<tr>
<td>SMSI18</td>
<td>2CF0H</td>
<td>MOVW [MSG12+0], MSG15</td>
</tr>
<tr>
<td>SMSI19</td>
<td>F001H</td>
<td>WAKEUP</td>
</tr>
<tr>
<td>SMSI20</td>
<td>0000H</td>
<td>Unused</td>
</tr>
<tr>
<td>SMSI21</td>
<td>0000H</td>
<td>Unused</td>
</tr>
<tr>
<td>SMSI22</td>
<td>0000H</td>
<td>Unused</td>
</tr>
<tr>
<td>SMSI23</td>
<td>0000H</td>
<td>Unused</td>
</tr>
<tr>
<td>SMSI24</td>
<td>0000H</td>
<td>Unused</td>
</tr>
<tr>
<td>SMSI25</td>
<td>0000H</td>
<td>Unused</td>
</tr>
<tr>
<td>SMSI26</td>
<td>0000H</td>
<td>Unused</td>
</tr>
<tr>
<td>SMSI27</td>
<td>0000H</td>
<td>Unused</td>
</tr>
<tr>
<td>SMSI28</td>
<td>0000H</td>
<td>Unused</td>
</tr>
<tr>
<td>SMSI29</td>
<td>0000H</td>
<td>Unused</td>
</tr>
<tr>
<td>SMSI30</td>
<td>0000H</td>
<td>Unused</td>
</tr>
<tr>
<td>SMSI31</td>
<td>0000H</td>
<td>Unused</td>
</tr>
</tbody>
</table>

Note. The setting values for SMSI0 to SMSI2 are set in r_SMS_ASM.smsasm.
The data of SMSI3 to SMSI29 transferred by DTC is stored in g_dtc_data.
Table 5-9 Sequencer instruction registers 0-31 (After the second DTC transfer)

<table>
<thead>
<tr>
<th>Register Symbol</th>
<th>Setting</th>
<th>Contents</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMSI0</td>
<td>2DF2H</td>
<td>MOVW [SMGS13+2], SMSG15</td>
<td>SMSI29 &lt;- FFFFH</td>
</tr>
<tr>
<td>SMSI1</td>
<td>F002H</td>
<td>DTCTRGR</td>
<td>DTC startup</td>
</tr>
<tr>
<td>SMSI2</td>
<td>AD02H</td>
<td>WHILE1 [SMGS13+2], 0</td>
<td>Waiting for DTC transfer completion</td>
</tr>
<tr>
<td>SMSI3</td>
<td>2E60H</td>
<td>MOVW [SMGS14+0], SMSG6</td>
<td>SMSG11 &lt;- Value of SMSG6</td>
</tr>
<tr>
<td>SMSI4</td>
<td>4210H</td>
<td>SET1 [SMGS2+0], 1</td>
<td>IICCTL00.STT0 &lt;- 1</td>
</tr>
<tr>
<td>SMSI5</td>
<td>9640H</td>
<td>WAIT 100, 0</td>
<td>Waiting for start condition</td>
</tr>
<tr>
<td>SMSI6</td>
<td>15A1H</td>
<td>MOV SMSG10, [SMGS5+1]</td>
<td>SMSG10 &lt;- For sending slave address</td>
</tr>
<tr>
<td>SMSI7</td>
<td>04A0H</td>
<td>MOV [SMGS4+0], SMSG10</td>
<td>IICAn &lt;- SMSG10</td>
</tr>
<tr>
<td>SMSI8</td>
<td>B330H</td>
<td>WHILE0 [SMGS3+0], 3</td>
<td>Wait until IFxx.IICAIFn = 1</td>
</tr>
<tr>
<td>SMSI9</td>
<td>5330H</td>
<td>CLR1 [SMGS3+0], 3</td>
<td>IF1L.IICAIF0 &lt;- 0</td>
</tr>
<tr>
<td>SMSI10</td>
<td>6421H</td>
<td>MOV1 SCY, [SMGS4+1], 2</td>
<td>SCY &lt;- IICS0.ACKD0</td>
</tr>
<tr>
<td>SMSI11</td>
<td>8101H</td>
<td>BNC 16</td>
<td>If ACK=0, branch to SMSI27</td>
</tr>
<tr>
<td>SMSI12</td>
<td>4220H</td>
<td>SET1 [SMGS2+0], 2</td>
<td>IICCTL00.ACKE0 &lt;- 1</td>
</tr>
<tr>
<td>SMSI13</td>
<td>5230H</td>
<td>CLR1 [SMGS2+0], 3</td>
<td>IICCTL00.WTIM0 &lt;- 0</td>
</tr>
<tr>
<td>SMSI14</td>
<td>4250H</td>
<td>SET1 [SMGS2+0], 5</td>
<td>IICCTL00.WREL0 &lt;- 1</td>
</tr>
<tr>
<td>SMSI15</td>
<td>B330H</td>
<td>WHILE0 [SMGS3+0], 3</td>
<td>Wait until IFxx.IICAIFn = 1</td>
</tr>
<tr>
<td>SMSI16</td>
<td>5330H</td>
<td>CLR1 [SMGS3+0], 3</td>
<td>IF1L.IICAIF0 &lt;- 0</td>
</tr>
<tr>
<td>SMSI17</td>
<td>14A0H</td>
<td>MOV SMSG10, [SMGS4+0]</td>
<td>SMSG10 &lt;- IICAn</td>
</tr>
<tr>
<td>SMSI18</td>
<td>0B0H</td>
<td>MOV [SMGS11+0], SMSG10</td>
<td>Stored Address &lt;- SMSG10</td>
</tr>
<tr>
<td>SMSI19</td>
<td>7B80H</td>
<td>ADDW SMSG11, SMSG8</td>
<td>SMSG11 &lt;- SMSG11 + 1</td>
</tr>
<tr>
<td>SMSI20</td>
<td>7B72H</td>
<td>CMPW SMSG11, SMSG7</td>
<td>When SMSG11 &lt; SMSG7</td>
</tr>
<tr>
<td>SMSI21</td>
<td>8F90H</td>
<td>BC -7</td>
<td>Branch to SMSI14</td>
</tr>
<tr>
<td>SMSI22</td>
<td>5220H</td>
<td>CLR1 [SMGS2+0], 2</td>
<td>IICCTL00.ACKE0 &lt;- 0</td>
</tr>
<tr>
<td>SMSI23</td>
<td>4230H</td>
<td>SET1 [SMGS2+0], 3</td>
<td>IICCTL00.WTIM0 &lt;- 1</td>
</tr>
<tr>
<td>SMSI24</td>
<td>4250H</td>
<td>SET1 [SMGS2+0], 5</td>
<td>IICCTL00.WREL0 &lt;- 1</td>
</tr>
<tr>
<td>SMSI25</td>
<td>7FF2H</td>
<td>CMPW SMSG15, SMSG15</td>
<td>Comparison for branching to SMSI0</td>
</tr>
<tr>
<td>SMSI26</td>
<td>8E62H</td>
<td>BZ -26</td>
<td>Branch to SMSI0</td>
</tr>
<tr>
<td>SMSI27</td>
<td>2CF0H</td>
<td>MOVW [SMGS12+0], SMSG15</td>
<td>SMSG11 &lt;- FFFFH</td>
</tr>
<tr>
<td>SMSI28</td>
<td>F001H</td>
<td>WAKEUP</td>
<td>WAKEUP</td>
</tr>
<tr>
<td>SMSI29</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI30</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI31</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
</tbody>
</table>

Note. The setting values for SMSI0 to SMSI2 are set in r_SMS_ASM.smsasm.
The data of SMSI3 to SMSI29 transferred by DTC is stored in g_dtc_data.
### Table 5-10 Sequencer instruction registers 0-31 (After the third DTC transfer)

<table>
<thead>
<tr>
<th>Register Symbol</th>
<th>Setting</th>
<th>Contents</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMSI0</td>
<td>2DF2H</td>
<td>MOVW [SM13+2], SMSG15</td>
<td>SMSI29 &lt;- FFFFH</td>
</tr>
<tr>
<td>SMSI1</td>
<td>F002H</td>
<td>DTCTRAG</td>
<td>DTC startup</td>
</tr>
<tr>
<td>SMSI2</td>
<td>AD02H</td>
<td>WHILE1 [SM13+2].0</td>
<td>Waiting for DTC transfer completion</td>
</tr>
<tr>
<td>SMSI3</td>
<td>B330H</td>
<td>WHILE0 [SM13+0].3</td>
<td>Wait until IFxx.IIFCAFn = 1</td>
</tr>
<tr>
<td>SMSI4</td>
<td>5330H</td>
<td>CLR1 [SM3+0].3</td>
<td>IF1L.IIFCAF0 &lt;- 0</td>
</tr>
<tr>
<td>SMSI5</td>
<td>4200H</td>
<td>SET1 [SM2+0].0</td>
<td>IICCTLn0.SPT0 &lt;- 1</td>
</tr>
<tr>
<td>SMSI6</td>
<td>9A10H</td>
<td>WAIT 161, 0</td>
<td>Waiting for stop condition</td>
</tr>
<tr>
<td>SMSI7</td>
<td>16A0H</td>
<td>MOV SMSG10, [SM6+0]</td>
<td>SMSG10 &lt;- humi_h</td>
</tr>
<tr>
<td>SMSI8</td>
<td>79A2H</td>
<td>CMPW SMSG9, SMSG10</td>
<td>When threshold &lt; humi_h</td>
</tr>
<tr>
<td>SMSI9</td>
<td>8020H</td>
<td>BC 2</td>
<td>Branch to SMSI11</td>
</tr>
<tr>
<td>SMSI10</td>
<td>F000H</td>
<td>FINISH</td>
<td>FINISH</td>
</tr>
<tr>
<td>SMSI11</td>
<td>F001H</td>
<td>WAKEUP</td>
<td>WAKEUP</td>
</tr>
<tr>
<td>SMSI12</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI13</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI14</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI15</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI16</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI17</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI18</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI19</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI20</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI21</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI22</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI23</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI24</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI25</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI26</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI27</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI28</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI29</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI30</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
<tr>
<td>SMSI31</td>
<td>0000H</td>
<td>Unused</td>
<td>-</td>
</tr>
</tbody>
</table>

Note. The setting values for SMSI0 to SMSI2 are set in r_SMS_ASM.smsasm.
The data of SMSI3 to SMSI29 transferred by DTC is stored in g_dtc_data.
6. Application example

In addition to the sample code, this application note stores the following Smart Configurator configuration files.

r01an6065jj0110_sms_humidity_sensor.scfg

The following is a description of the file and setting examples and precautions for use.

6.1 r01an6065jj0110_sms_humidity_sensor.scfg

This is the Smart Configurator configuration file used in the sample code. It contains all the features configured in the Smart Configurator. The sample code settings are as follows.

Table 6-1 Parameters of Smart Configurator

<table>
<thead>
<tr>
<th>Tag name</th>
<th>Components</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clocks</td>
<td>-</td>
<td>Operation mode: High-speed main mode 2.4 (V) ~ 5.5 (V) EVDD setting: 1.8V ≦ EVDD &lt; 5.5V High-speed on-chip oscillator: 32MHz fIH: 32MHz fCLK: 32MHz (High-speed on-chip oscillator) fSXP: 32.768kHz (Low-speed on-chip oscillator)</td>
</tr>
<tr>
<td>Components</td>
<td>r_bsp</td>
<td>Start up select : Enable (use BSP startup) Control of invalid memory access detection : Disable RAM guard space (GRAM0-1) : Disabled Guard of control registers of port function (GPORT) : Disabled Guard of registers of interrupt function (GINT) : Disabled Guard of control registers of clock control function, voltage detector, and RAM parity error detection function (GCSC) : Disabled Data flash access control (DFLEN) : Disables Initialization of peripheral functions by Code Generator/Smart Configurator : Enable API functions disable : Enable Parameter check enable : Enable Setting for starting the high-speed on-chip oscillator at the times of release from STOP mode and of transitions to SNOOZE mode : High-speed Enable user warm start callback (PRE) : Unused Enable user warm start callback (POST) : Unused Watchdog Timer refresh enable : Unused</td>
</tr>
<tr>
<td>Config_LVD0</td>
<td></td>
<td>Operation mode setting: Reset mode Voltage detection setting: Reset generation level (V_LVD0): 1.86 (V)</td>
</tr>
</tbody>
</table>
### Table 6-2 Parameters of Smart Configurator

<table>
<thead>
<tr>
<th>Tag name</th>
<th>Components</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Config_DTC</td>
<td><strong>Components</strong></td>
<td><strong>Contents</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DTC base address: 0xFFD00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Control data0 (DTCD0): Event output from the SNOOZE mode sequencer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[DTCD0]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transfer mode setting: Repeat mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transfer data size setting: 16 bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Repeat mode interrupt setting: Disable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Repeat area setting: Transfer source</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transfer source address: 0xE000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transfer destination address: 0x0386 Address fixed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transfer count: 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transfer block size: 54</td>
</tr>
<tr>
<td>Config_ITL000 _ITL001</td>
<td><strong>Components</strong>: Interval timer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Operation mode: 16 bit count mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Resource: ITL000_ITL001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Operation clock: fSXP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clock source: fITL0/128</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Interval value: 5000 ms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Interrupt setting: unused</td>
</tr>
<tr>
<td>Config_PORT</td>
<td><strong>Components</strong>: Port</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Port selection: PORT5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>P53: Out (Output 1)</td>
</tr>
<tr>
<td>Config_IICA0</td>
<td><strong>Clock mode setting</strong>: fCLK/2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Address: 16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Operation mode setting: Standard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transfer clock (fSCL): 100000 (bps)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Communication end interrupt priority: Level 3 (low)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Callback function setting: Master transmission end, Master reception end</td>
</tr>
</tbody>
</table>

Note 1. When using IAR, use the following settings.
- On-chip debug operation setting: Use emulator
- Emulator setting: E2 Emulator Lite
6.1.1 Clocks
Set the clock used in the sample code.

6.1.2 System
Set the on-chip debug of the sample code.
"Control of on-chip debug operation" and "Security ID authentication failure setting" affect "On-chip debugging is enabled" in "Table 5-3 Option Byte Settings". Note that changing the settings.

6.1.3 r_bsp
Set the startup of the sample code.

6.1.4 Config_LVD0
Set the power management of the sample code.
Affects "Setting of LVD0" in "Table 5-3 Option Byte Settings". Note that changing the settings.

6.1.5 Config_DTC
Set the DTC used in the sample code.
The sample code uses control data 0 to transfer the SMS process, which was stored in RAM in advance (Transfer source address: FE000H (set value: 0xE000)), to the SMS register (Transfer destination address: F0386H (set value: 0x0386)). main.c will store the data in RAM.

6.1.6 Config_IT000_ITL001
Initialize the interval timer for the sample code.
The interval timer interrupt (INTITL) is used to start the SMS in the sample code. Therefore, "Interrupt setting" is set to "Not used". "Interrupt Settings" can also be changed to "Use".
Since INTITL is masked by the R_Config_SMS_Start function, the CPU will not start even if INTITL is generated during STOP or SNOOZE mode. After returning from STOP mode and SNOOZE mode, INTITL is in a masked state, so unmask INTITL if necessary.

6.1.7 Config_PORT
Set the port of the sample code.
In the sample code, P53 is used to control LED1 and P52 is used to control LED2.

6.1.8 Config_IICA0
Set IICA0 of the sample code.
7. Sample Code
Sample code can be downloaded from the Renesas Electronics website.

8. Reference
RL78/G23 User’s Manual: Hardware (R01UH0896E)
RL78 Family User’s Manual: Software (R01US0015E)
SMS assembler User’s Manual (R20UT4792J)
RL78 Smart Configurator User's Guide: CS+ (R20AN0580E)
RL78 Smart Configurator User's Guide: e² studio (R20AN0579E)
RL78 Smart Configurator User's Guide: IAREW (R20AN0581E)
HS300x Datasheet
(The latest version can be downloaded from the Renesas Electronics website.)

Technical Update / Technical News
(The latest version can be downloaded from the Renesas Electronics website.)

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

<table>
<thead>
<tr>
<th>Rev.</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>Sep.30.21</td>
<td>First edition</td>
</tr>
<tr>
<td>1.10</td>
<td>Feb.15.22</td>
<td>The threshold value in the sample code is set to 2CH, which corresponds to 70% humidity. -&gt; The threshold value in the sample code is set to 20H, which corresponds to 50% humidity.</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>Figure 1-3 Entire Flowchart Wait 4s -&gt; Wait 62.5ms</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>Updated tool version</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Table 2-1 Operation Confirmation Conditions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Integrated development environment (CS+) : CS+ for CC V8.06.00 -&gt; V8.07.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C compiler (CS+) : CC-RL V1.10 -&gt; V1.11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Integrated development environment (e2 studio) : e2 studio 2021-07 (21.7.0) -&gt; 2022-01 (22.1.0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C compiler (e2 studio) : CC-RL V1.10 -&gt; V1.11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Smart Configurator : V.1.0.1 -&gt; V.1.2.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Board support package (r_bsp) : V.1.11 -&gt; V.1.13</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>Table 5-1 Operation Overview 4 seconds -&gt; 62.5ms</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>Update folder name</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>Table 5-4 Constants used in the sample code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>THRESHOLD : 2CH -&gt; 20H</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Removed dtc_init[]</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>[Function name] main float.h -&gt; r_sms.h</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td>[Function name] r_sms_start Removed addr_dtc_init[]</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>waits for 4 seconds -&gt; waits for 62.5ms</td>
</tr>
<tr>
<td>21</td>
<td></td>
<td>Figure 5-8 SMS process</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Initialize SMSG11, SMSG12 -&gt; Initialize SMSG10, SMSG11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Wait 4s -&gt; 62.5ms</td>
</tr>
<tr>
<td>22</td>
<td></td>
<td>Figure 5-9 SMS process</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SMSG12 -&gt; SMSG11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SMSG11 -&gt; SMSG10</td>
</tr>
<tr>
<td>23</td>
<td></td>
<td>Table 5-7 Sequencer general-purpose registers 0-15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Update with sample program update</td>
</tr>
<tr>
<td>24 - 26</td>
<td></td>
<td>Table 5-8 Sequencer instruction registers 0-31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Table 5-1 Sequencer instruction registers 0-31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Table 5-2 Sequencer instruction registers 0-31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Update with sample program update</td>
</tr>
<tr>
<td>27</td>
<td></td>
<td>Update file name</td>
</tr>
</tbody>
</table>
General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit 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. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.

2. Processing at power-on
   The state of the product is undefined at the time 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 time 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 time 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 time when power is supplied until the power reaches the level at which resetting is specified.

3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.

4. 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 the 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.

5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is 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. Additionally, 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.

6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between VIL (Max.) and VIH (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between VIL (Max.) and VIH (Min.).

7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.

8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of 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.
1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.

2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.

3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.

4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.

5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.

6. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

   - "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   - "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space systems; underwater repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user's manual or other Renesas Electronics document.

7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION ("Vulnerability Issues"). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.

8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, "General Notes for Handling and Using Semiconductors Devices") in the reliability handbook, etc., and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.

9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or damage to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.

10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.

11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.

12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.

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

14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

© 2022 Renesas Electronics Corporation. All rights reserved.