

## 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 I<sup>2</sup>C communication protocol during SNOOZE mode.

I<sup>2</sup>C 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

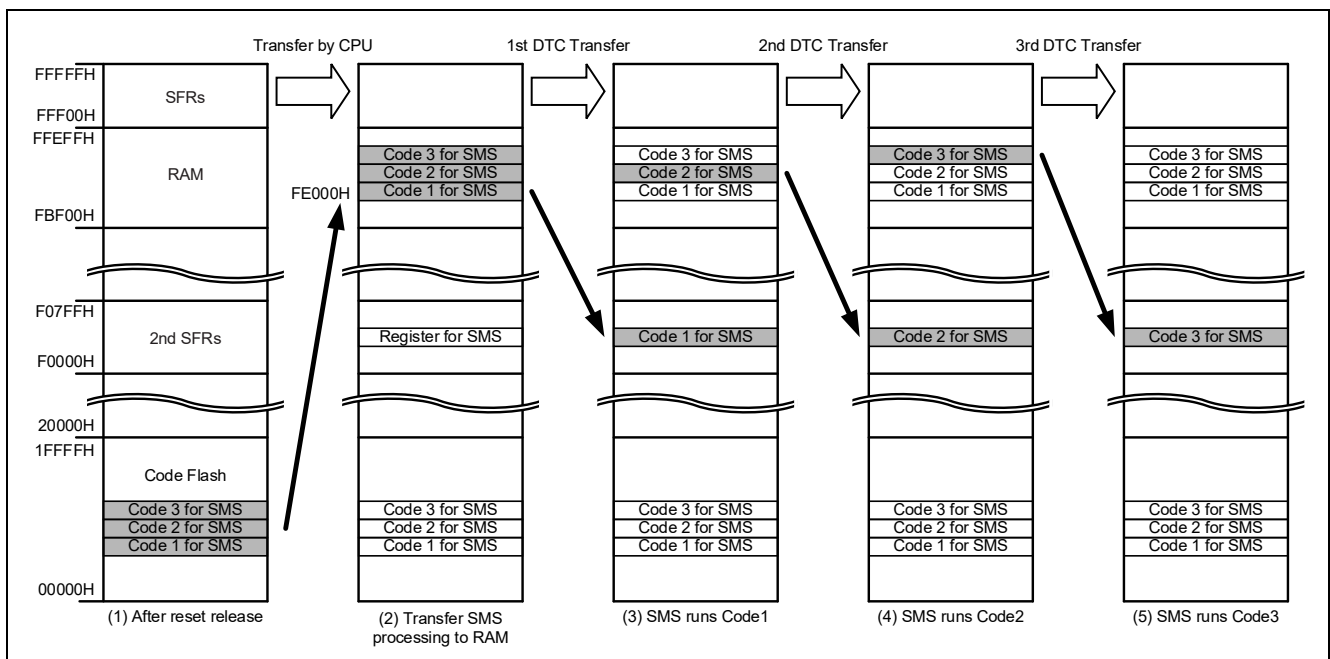
### 1. Specifications

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 I<sup>2</sup>C communication standard is 32 steps or more. Therefore, this application note shows how to perform I<sup>2</sup>C 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 I<sup>2</sup>C communication is set using the interval timer (TML32), and SMS is started by a TML32 interrupt (INTITL). After the SMS is started, the DMATRGR 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 I<sup>2</sup>C communication ((3) - (5) in Figure 1-1).

Figure 1-1 Memory map



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

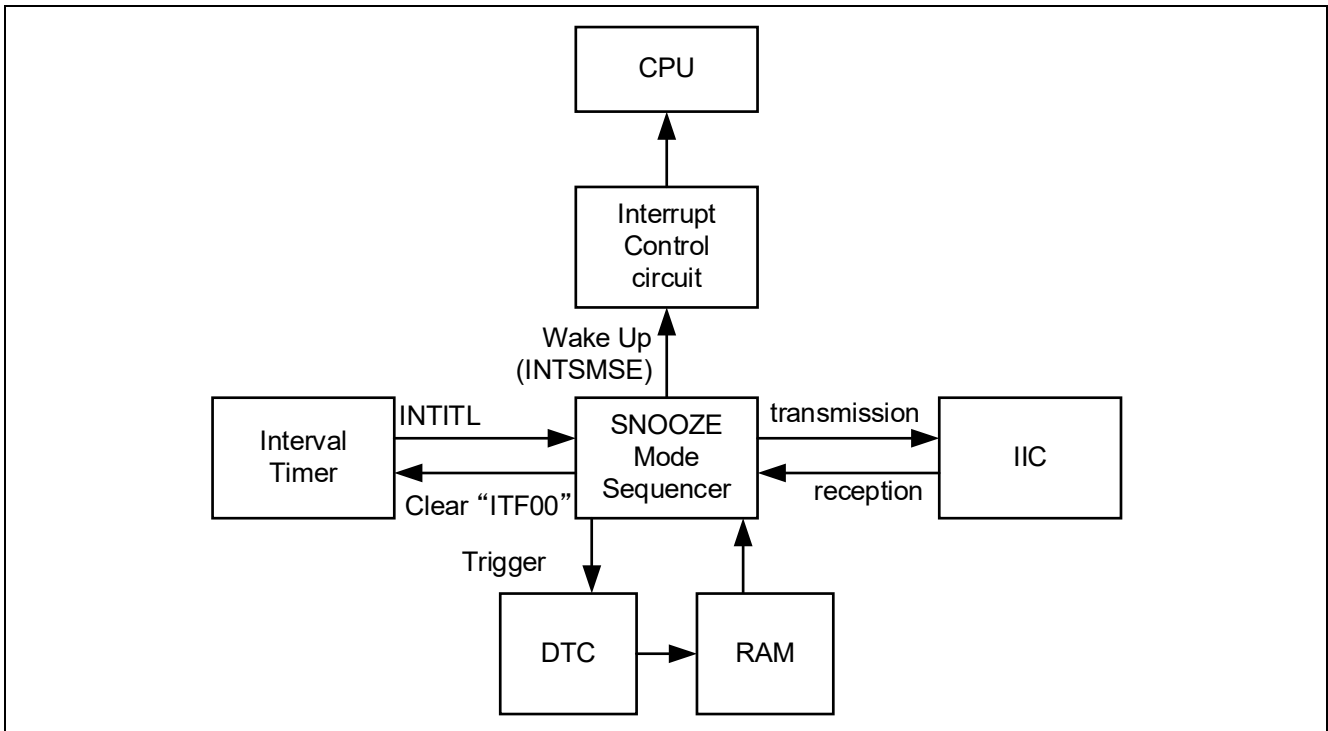
$$\text{Humidity}[\%RH] = \left( \frac{\text{Humidity}[13:0]}{2^{14} - 1} \right) * 100$$

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**





## 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**

Items	Contents
MCU	RL78/G23 (R7F100GLG)
Operating frequencies	<ul style="list-style-type: none"> <li>High-speed on-chip oscillator clock: 32 MHz</li> <li>CPU/peripheral hardware clock: 32 MHz</li> </ul>
Operating voltage	<ul style="list-style-type: none"> <li>3.3V</li> <li>LVD0 operations (<math>V_{LVD0}</math>) : Reset mode Rising edge TYP.1.90V (1.84V-1.95V) Falling edge TYP.1.86V (1.80V-1.91V)</li> </ul>
Integrated development environment (CS+)	CS+ for CC V 8.07.00 from Renesas Electronics Corp.
C compiler (CS+)	CC-RL V1.11 from Renesas Electronics Corp.
Integrated development environment (e <sup>2</sup> studio)	e <sup>2</sup> studio 2022-01 (22.1.0) from Renesas Electronics Corp.
C compiler (e <sup>2</sup> studio)	CC-RL V1.11 from Renesas Electronics Corp.
Integrated development environment (IAR)	IAR Embedded Workbench for Renesas RL78 v4.21.1 from IAR Systems
C compiler (IAR)	
Smart Configurator	V.1.2.0
Board support package (r_bsp)	V.1.13
Emulator	CS+, e <sup>2</sup> studio: COM port IAR: E2 Emulator Lite
Board	RL78/G23 Fast Prototyping Board (RTK7RLG230CLG000BJ)

### 3. Related application note

The following application note is related to this application note.

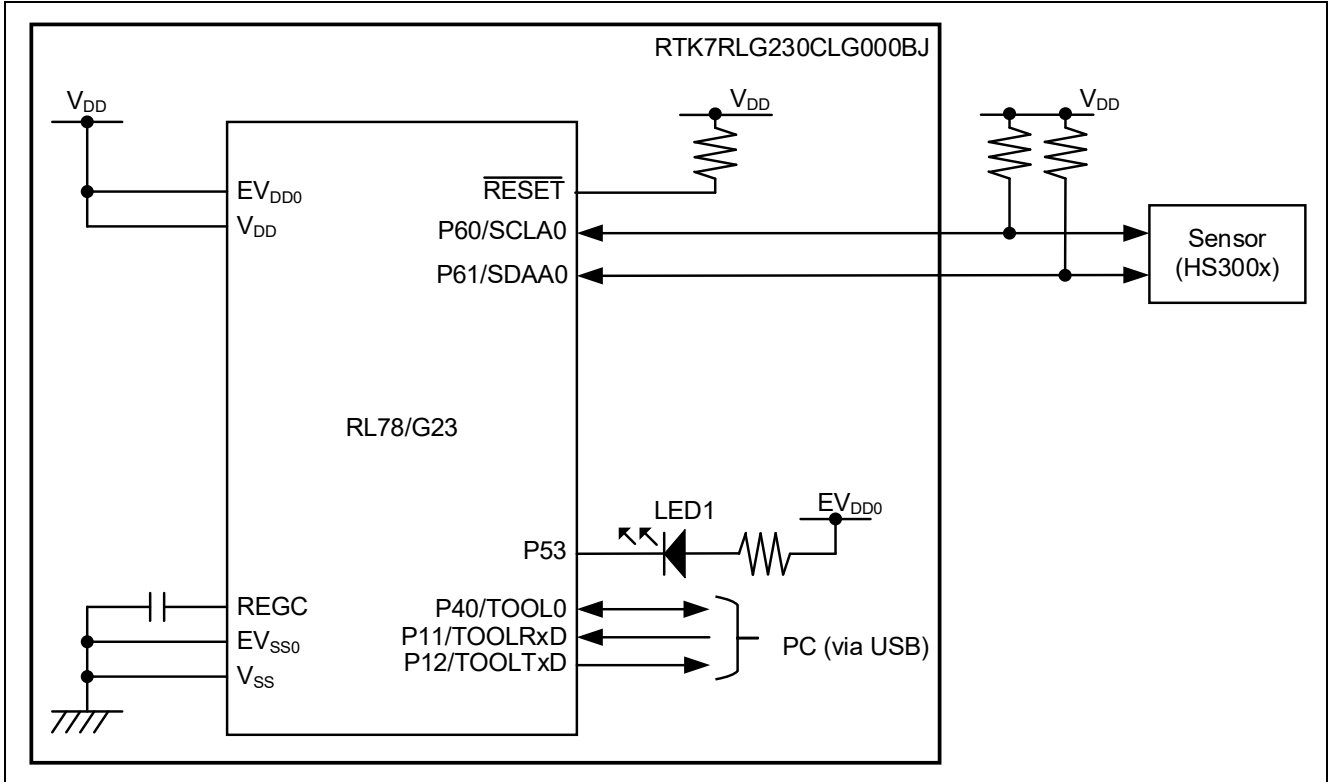
Please refer to them as well.

## 4. Hardware

### 4.1 Example of Hardware Configuration

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

Figure 4-1 Hardware Configuration



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 V<sub>DD</sub> or V<sub>SS</sub> through a resistor.)

Caution 2. Connect the EV<sub>SS</sub> pin to V<sub>SS</sub> and the EV<sub>DD</sub> pin to V<sub>DD</sub>.

Caution 3. V<sub>DD</sub> must be held at not lower than the reset release voltage (V<sub>LVD0</sub>) 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**

Pin Name	Input/Output	Function
P53	Output	LED1 lights (Low Active)
P60	Input/Output	Clock I/O pin of serial interfaces IICA0
P61	Input/Output	Serial data I/O pin of serial interfaces IICA0

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<sup>2</sup>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**

No.	Operation of each function		
	CPU	SMS	DTC
(1)	Stores code for SMS from Code Flash to RAM	-	-
(2)	Starts counting TML32	Waits for startup trigger	Waits for startup trigger
(3)	Shifts to STOP mode	↑	↑
(4)	During STOP	Starts SMS with TML32 compare match	↑
(5)	↑	Runs the DTC startup process	↑
(6)	↑	Waits for completion of DTC transfer	Transfers data from RAM to SMSIp register
(7)	↑	Sends MR command to the sensor and wait 62.5ms	Waits for startup trigger
(8)	↑	Runs the DTC startup process	↑
(9)	↑	Waits for completion of DTC transfer	Transfers data from RAM to SMSIp register
(10)	↑	Receives data from sensor and store in RAM	Waits for startup trigger
(11)	↑	Runs the DTC startup process	↑
(12)	↑	Waits for completion of DTC transfer	Transfers data from RAM to SMSIp register
(13)	↑	Branches to (14) if the received result does not exceed the threshold, else branches (15)	Waits for startup trigger
(14)	↑	SMS shifts to standby state and shifts to (5)	↑
(15)	↑	Starts the CPU	↑
(16)	Turns on LED1	-	-
(17)	Calculates humidity from data stored in RAM	-	-

## 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**

Folder/File configuration	Outline	Created by Smart configurator
\r01an6065jj0110_sms_humidity_sensor<DIR>	Root folder of this sample code	
\src<DIR>	Folder for program source	
main.c	Sample code source file	
sms_p.c	Code storage file for SMS	
\SMS<DIR>	Folder for SMS program	
r_sms.c	Source file for SMS	
r_sms.h	Header file for SMS	
r_sms_ASM.smsasm	ASM source file for SMS	
\smc_gen<DIR> <sup>Note 2</sup>	Folder created by Smart Configurator	√
\Config_DTC <DIR>	Folder for DTC program	√
Config_DTC.c	Source file for DTC	√
Config_DTC.h	Header file for DTC	√
Config_DTC_user.c	Interrupt source file for DTC	√ <sup>Note 1</sup>
\Config_IICA0<DIR>	Folder for IICA0 program	√
Config_IICA0.c	Source file for IICA0	√
Config_IICA0.h	Header file for IICA0	√
Config_IICA0_user.c	Interrupt source file for IICA0	√ <sup>Note 1</sup>
\Config_ITL000_ITL001<DIR>	Folder for TML32 program	√
Config_ITL000_ITL001.c	Source file for TML32	√
Config_ITL000_ITL001.h	Header file for TML32	√
Config_ITL000_ITL001_user.c	Interrupt source file for TML32	√ <sup>Note 1</sup>
\Config_PORT<DIR>	Folder for PORT program	√
Config_PORT.c	Source file for PORT	√
Config_PORT.h	Header file for PORT	√
Config_PORT_user.c	Interrupt source file for PORT	√ <sup>Note 1</sup>
\general<DIR>	Folder for initialize or common program	√
\r_bsp<DIR>	Folder for BSP program	√
\r_config<DIR>	Folder for BSP_CFG program	√

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 5-3 Option Byte Settings**

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

### 5.4 Constants

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

**Table 5-4 Constants used in the sample code**

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

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 5-5 Global variables used in the sample code**

Type	Variable name	contents	Functions used in
float	g_result	Humidity calculation results	main.c
uint8_t	g_sms_wakeup_flag	SMS wakeup flag	r_sms_getstatus r_sms_start r_sms_interrupt
uint16_t	g_communication_status	IICA0 status flag	r_iica0_getstatus

## 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 5-6 Functions**

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

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

<b>Outline</b>	Main process
<b>Header</b>	r_smc_entry.h, string.h, r_sms.h
<b>Declaration</b>	void main (void);
<b>Description</b>	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.
<b>Arguments</b>	None
<b>Return value</b>	None
<b>Remarks</b>	None

[Function name] r\_sms\_create

---

<b>Outline</b>	Initialization process for SMS
<b>Header</b>	r_SMS.h, r_cg_macrodriver.h, r_cg_userdefine.h, r_SMS_ASM.h
<b>Declaration</b>	void r_sms_create (void)
<b>Description</b>	Initializes the SMS module before controlling it, including SMS configuration, copying SMS instructions, and copying SMS data.
<b>Arguments</b>	None
<b>Return value</b>	None
<b>Remarks</b>	None

[Function name] r\_sms\_start

---

<b>Outline</b>	Start SMS operation
<b>Header</b>	r_SMS.h, r_cg_macrodriver.h, r_cg_userdefine.h, r_SMS_ASM.h
<b>Declaration</b>	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)
<b>Description</b>	Sets the SMS data from the argument and starts the SMS module operation.
<b>Arguments</b>	addr_iic_sl, addr_iic_rx_s, addr_iic_rx_e, val_threshold
<b>Return value</b>	None
<b>Remarks</b>	None

[Function name] r\_sms\_getstatus

---

<b>Outline</b>	Check the status of SMS wakeup
<b>Header</b>	r_SMS.h, r_cg_macrodriver.h, r_cg_userdefine.h, r_SMS_ASM.h
<b>Declaration</b>	uint8_t r_sms_getstatus (void)
<b>Description</b>	Checks the status of SMS wakeup.
<b>Arguments</b>	None
<b>Return value</b>	None
<b>Remarks</b>	None

[Function name] r\_sms\_interrupt

---

<b>Outline</b>	INTSMSE Interrupt process
<b>Header</b>	r_SMS.h, r_cg_macrodriver.h, r_cg_userdefine.h, r_SMS_ASM.h
<b>Declaration</b>	#pragma interrupt r_sms_interrupt(vect=INTSMSE)
<b>Description</b>	Sets g_sms_wakeup_flag = 1.
<b>Arguments</b>	None
<b>Return value</b>	None
<b>Remarks</b>	None

[Function name] r\_iica0\_getstatus

---

<b>Outline</b>	IICA0 status confirmation process
<b>Header</b>	r_cg_macrodriver.h, r_cg_userdefine.h, Config_IICA0.h
<b>Declaration</b>	uint16_t r_iica0_getstatus (void) (void)
<b>Description</b>	Checks the status of IICA0.
<b>Arguments</b>	None
<b>Return value</b>	None
<b>Remarks</b>	None

---

[Function name] memcpy\_ff

---

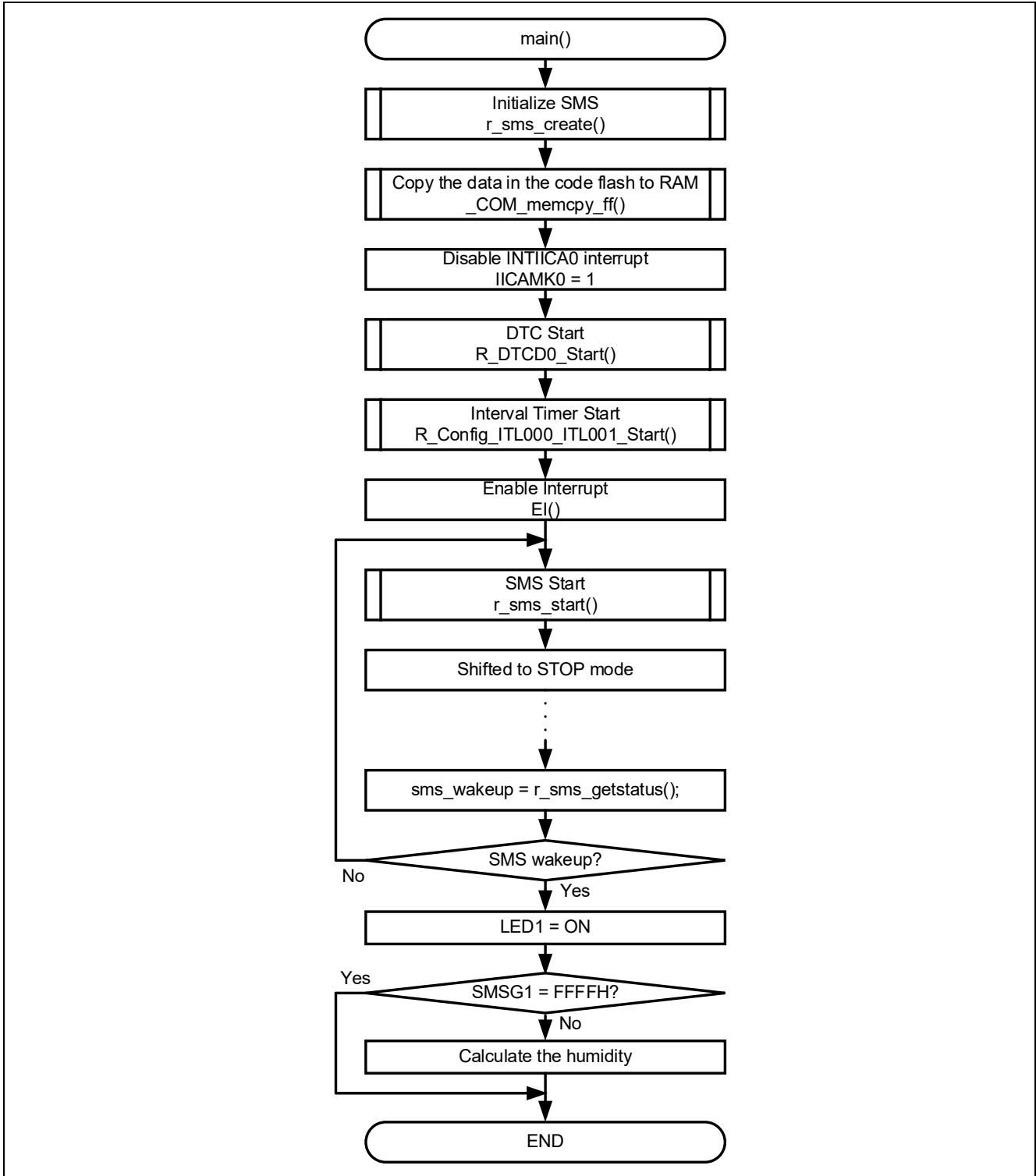
**Outline** Copy Data  
**Header** r\_smc\_entry.h, r\_SMS.h  
**Declaration** \_\_far\_func void \_\_far \* memcpy\_ff(void \_\_near \* \_Restrict s1, const void \_\_far  
\*\_Restrict s2, uint32\_t n)  
**Description** Copies the data in s2 to s1 for n bytes.  
**Arguments** s1, s2, n  
**Return value** None  
**Remarks** None

### 5.8 Flow Charts

Main Process

Figure 5-1 shows flowchart of main process.

Figure 5-1 Main process

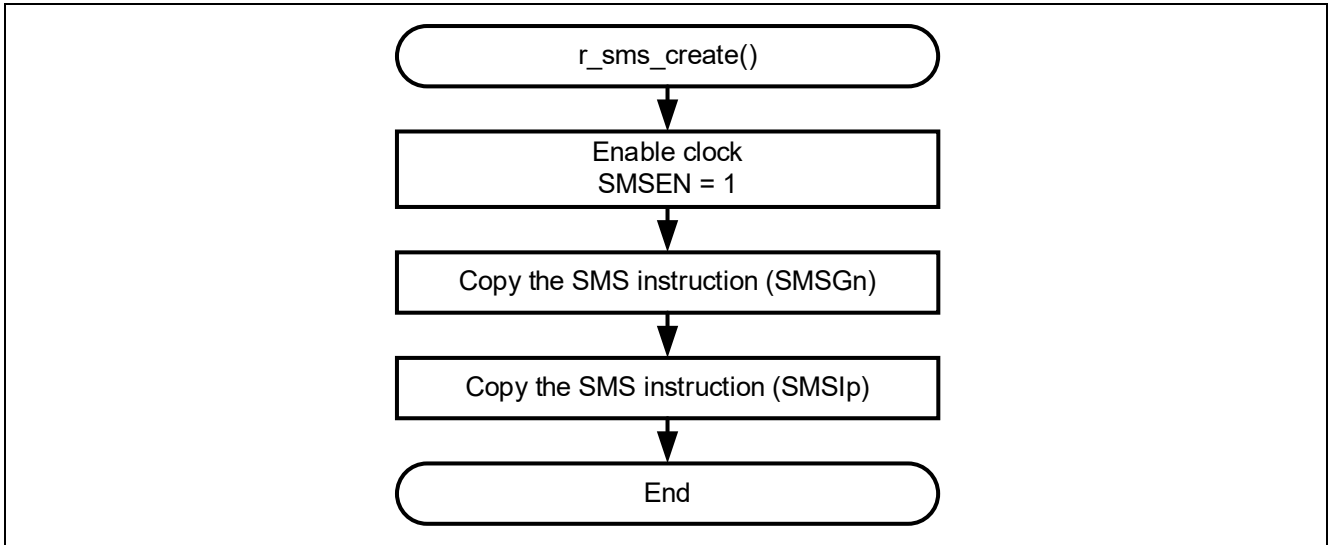




SMS initialization process

Figure 5-2 shows flowchart of SMS initialization process.

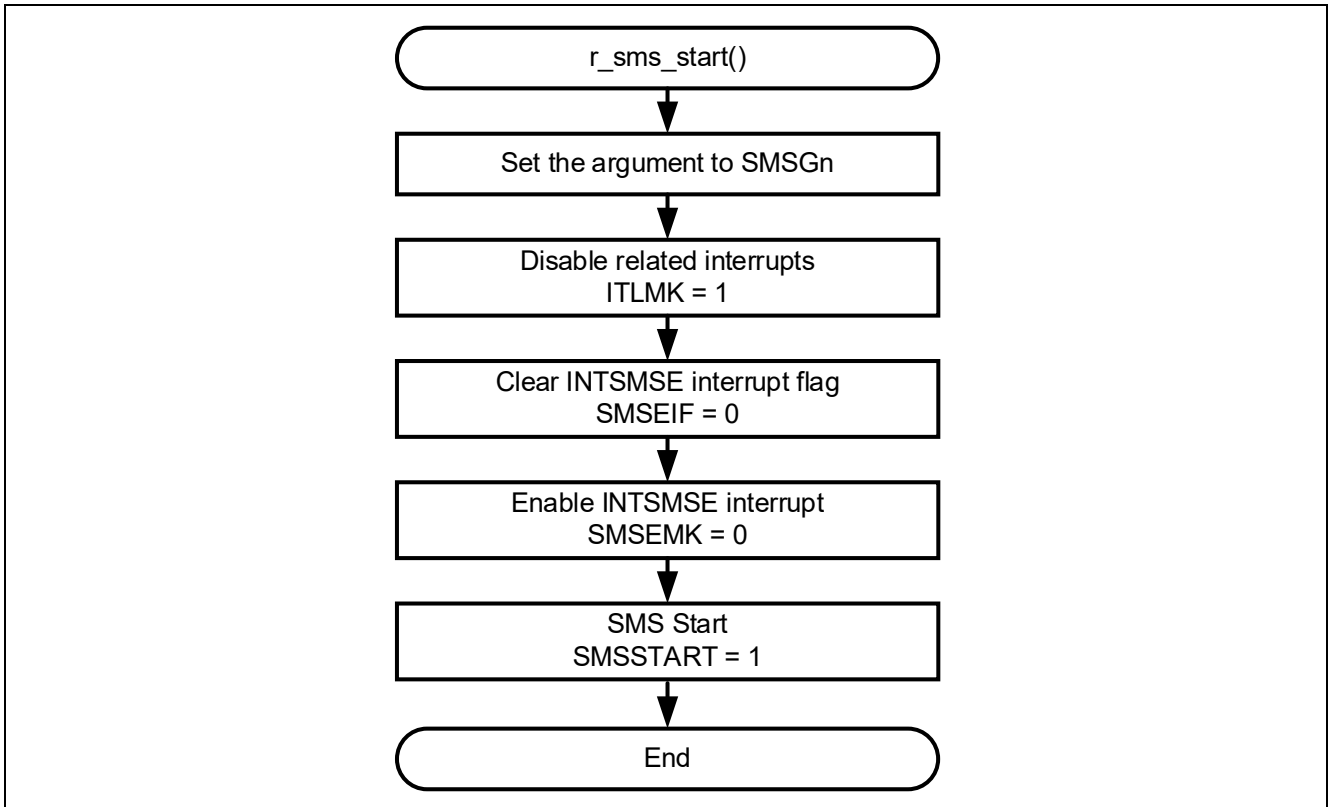
**Figure 5-2 SMS initialization process**



**5.8.1 SMS operation start processing**

Figure 5-3 shows flowchart of SMS operation start processing.

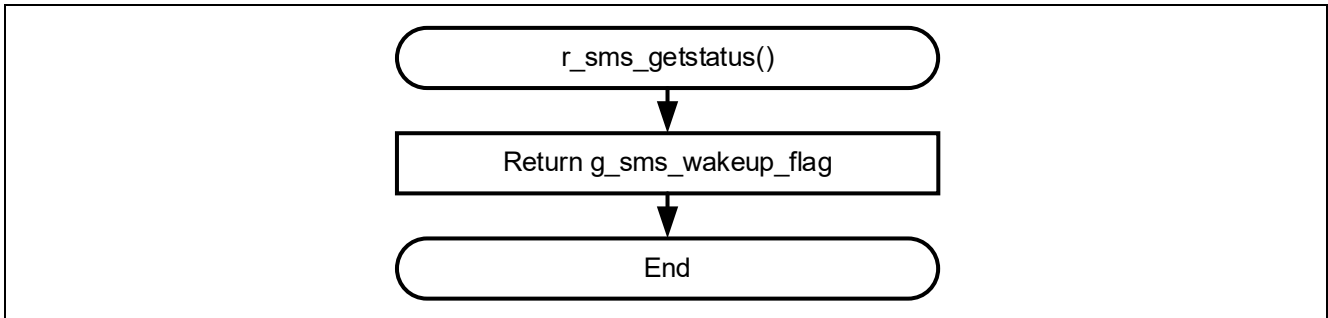
**Figure 5-3 SMS operation start processing**



**5.8.2 SMS status confirmation process**

Figure 5-4 shows flowchart of SMS status confirmation process.

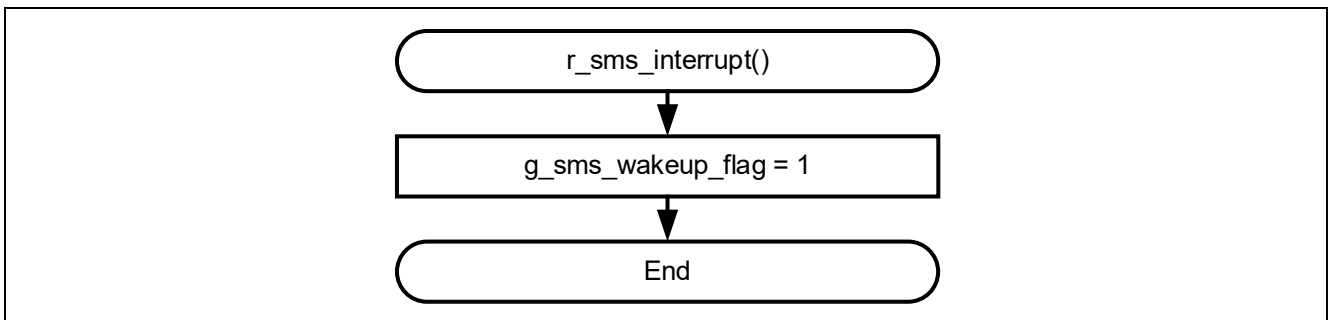
**Figure 5-4 SMS status confirmation process**



**5.8.3 SMS interrupt process**

Figure 5-5 shows flowchart of SMS interrupt process.

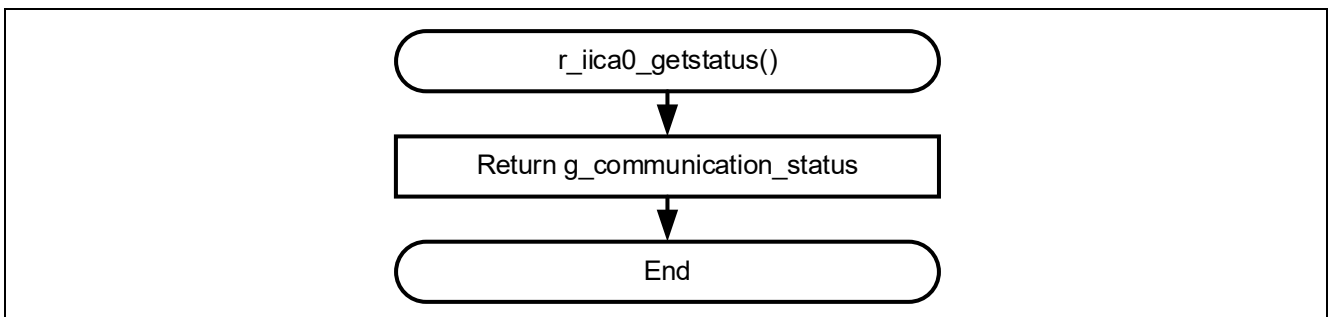
**Figure 5-5 SMS interrupt process**



**5.8.4 IICA0 status confirmation process**

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

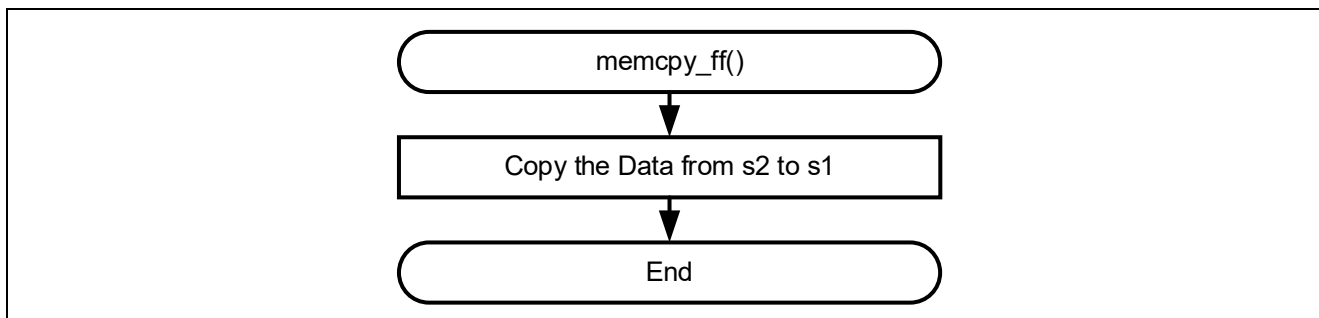
**Figure 5-6 IICA0 status confirmation process**



### 5.8.5 Data copy process

Figure 5-7 shows flowchart of Data copy process.

Figure 5-7 Data copy process



## 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.

<b>Outline</b>	SMS process
<b>Description</b>	<p>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.</p> <p>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.</p> <p>The instructions to process I<sup>2</sup>C communication are stored in RAM in advance, and then transferred and executed in three separate times by DTC.</p>
<b>Arguments</b> <sup>Note1</sup>	<p>addr_iic_sl: The start address of the buffer that stores the slave address of the sensor</p> <p>addr_iic_rx_s: Start address of data storage area</p> <p>addr_iic_rx_e: End address of data storage area</p> <p>val_threshold: Threshold value of humidity data</p>
<b>Return value</b>	None
<b>Remarks</b>	None

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 I<sup>2</sup>C communication. For details on the HS300x protocol, refer to the HS300x manual.

Figure 5-8 SMS process

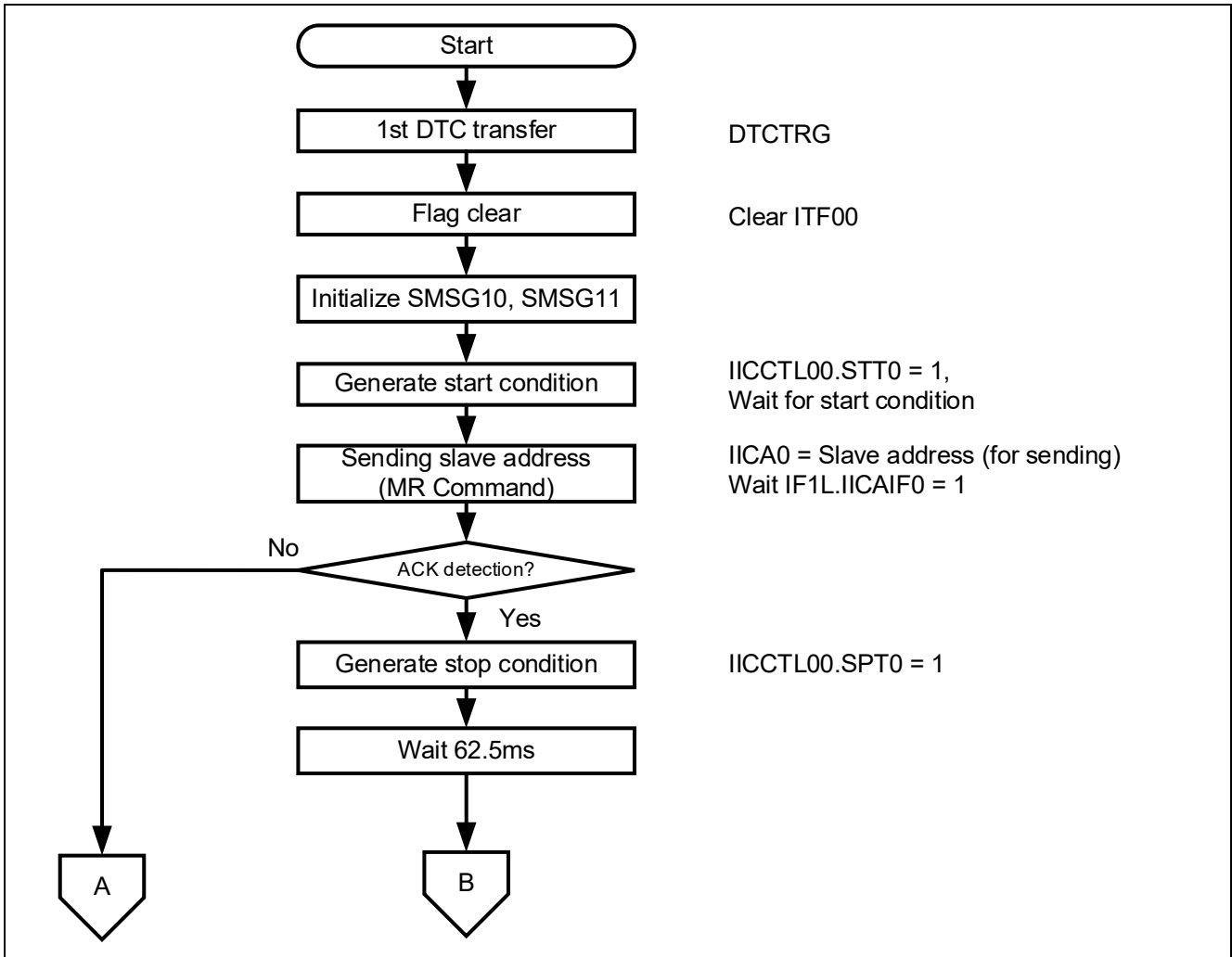


Figure 5-9 SMS process

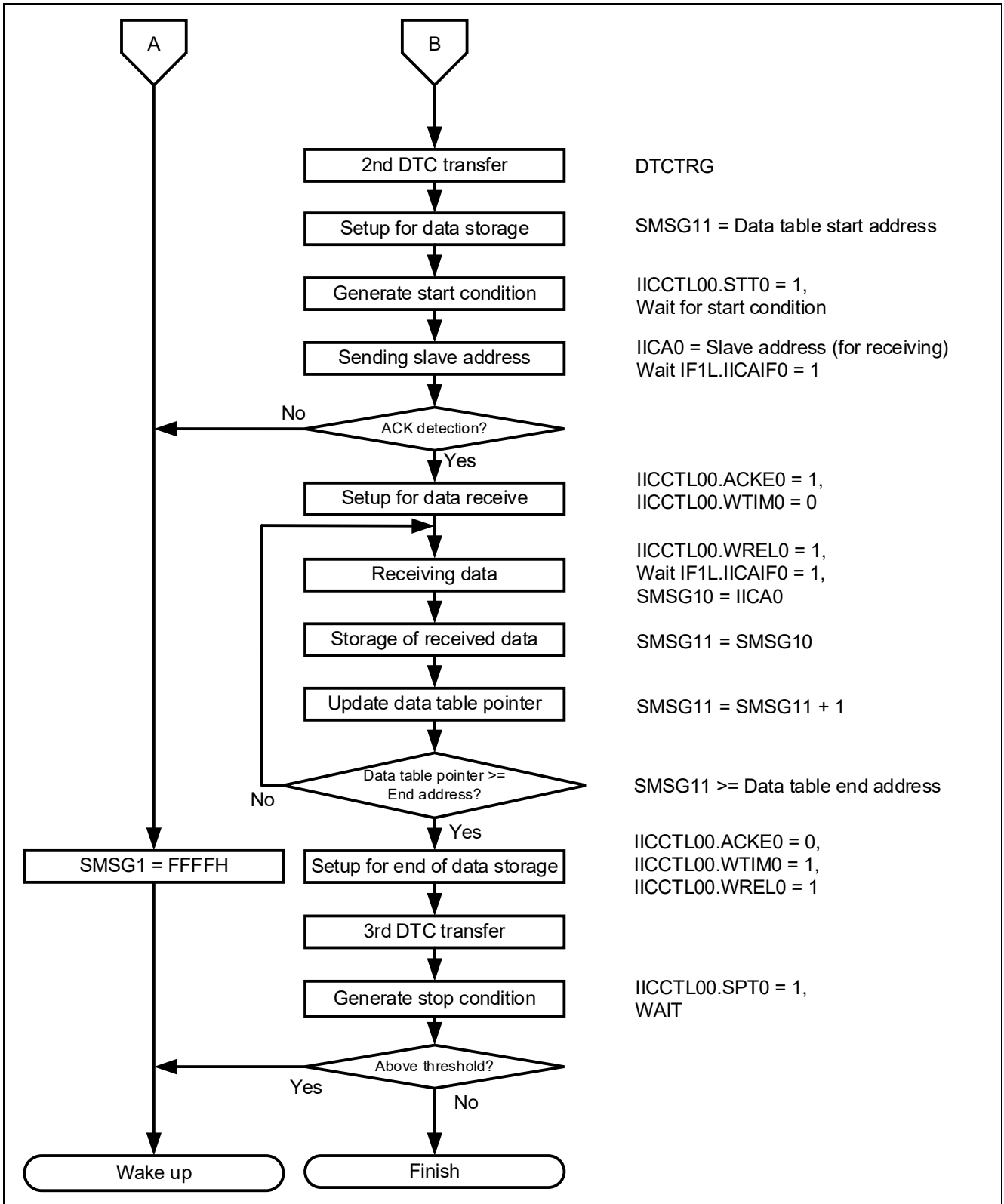


Table 5-7 Sequencer general-purpose registers 0-15

Register Symbol	Setting	Remark
SMSG0	0000H	fixed value: 0000H
SMSG1	&ITLS0	ITLS0 address
SMSG2	&IICCTL00	IICCTL00 address
SMSG3	&IF1L	IF1L address
SMSG4	&IICA0	IICA0 address
SMSG5	0000H	Slave address of sensor
SMSG6	0000H	Data table start address
SMSG7	0000H	Data table end address
SMSG8	1	fixed value: 1
SMSG9	0000H	Humidity threshold
SMSG10	0000H	for calculation
SMSG11	0000H	for calculation
SMSG12	&SMSG1	SMSG1 address
SMSG13	&SMSI28	SMSI28 address
SMSG14	&SMSG11	SMSG11 address
SMSG15	FFFFH	fixed value: FFFFH

Table 5-8 Sequencer instruction registers 0-31 (After the first DTC transfer)

Register Symbol	Setting	Contents	Remark
SMSI0	2DF2H	MOVW [SMSG13+2], SMSG15	SMSI29 <- FFFFH
SMSI1	F002H	DTCTRG	DTC startup
SMSI2	AD02H	WHILE1 [SMSG13+2].0	Waiting for DTC transfer completion
SMSI3	0100H	MOV [SMSG1+0], SMSG0	ITLS0 <- 00H
SMSI4	30A0H	MOVW SMSG10, [SMSG0+0]	Initialization of SMSG10
SMSI5	30B0H	MOVW SMSG11, [SMSG0+0]	Initialization of SMSG11
SMSI6	4210H	SET1 [SMSG2+0].1	IICCTL00.STT0 <- 1
SMSI7	9640H	WAIT 100, 0	Waiting for start condition
SMSI8	15A0H	MOV SMSG10, [SMSG5+0]	SMSG10 <- For sending slave address
SMSI9	04A0H	MOV [SMSG4+0], SMSG10	IICAn <- SMSG10
SMSI10	B330H	WHILE0 [SMSG3+0].3	Wait until IFxx.IICAIFn = 1
SMSI11	5330H	CLR1 [SMSG3+0].3	IF1L.IICAIF0 <- 0
SMSI12	6421H	MOV1 SCY, [SMSG4+1].2	SCY <- IICS0.ACKD0
SMSI13	8051H	BNC 5	If ACK=0, branch to SMSI25
SMSI14	4200H	SET1 [SMSG2+0].0	IICCTLn0.SPT0 <- 1
SMSI15	9804H	WAIT 128, 4	Wait 62.5ms
SMSI16	7FF2H	CMPW SMSG15, SMSG15	Comparison for branching to SMSI0
SMSI17	8EF2H	BZ -17	Branch to SMSI0
SMSI18	2CF0H	MOVW [SMSG12+0], SMSG15	SMSG1 <- FFFFH
SMSI19	F001H	WAKEUP	WAKEUP
SMSI20	0000H	Unused	-
SMSI21	0000H	Unused	-
SMSI22	0000H	Unused	-
SMSI23	0000H	Unused	-
SMSI24	0000H	Unused	-
SMSI25	0000H	Unused	-
SMSI26	0000H	Unused	-
SMSI27	0000H	Unused	-
SMSI28	0000H	Unused	-
SMSI29	0000H	Unused	-
SMSI30	0000H	Unused	-
SMSI31	0000H	Unused	-

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)

Register Symbol	Setting	Contents	Remark
SMSI0	2DF2H	MOVW [MSG13+2], MSG15	SMSI29 <- FFFFH
SMSI1	F002H	DTCTRG	DTC startup
SMSI2	AD02H	WHILE1 [MSG13+2].0	Waiting for DTC transfer completion
SMSI3	2E60H	MOVW [MSG14+0], MSG6	MSG11 <- Value of MSG6
SMSI4	4210H	SET1 [MSG2+0]. 1	IICCTL00.STT0 <- 1
SMSI5	9640H	WAIT 100, 0	Waiting for start condition
SMSI6	15A1H	MOV MSG10, [MSG5+1]	MSG10 <- For sending slave address
SMSI7	04A0H	MOV [MSG4+0], MSG10	IICAn <- MSG10
SMSI8	B330H	WHILE0 [MSG3+0]. 3	Wait until IFxx.IICAIFn = 1
SMSI9	5330H	CLR1 [MSG3+0].3	IF1L.IICAIF0 <- 0
SMSI10	6421H	MOV1 SCY, [MSG4+1].2	SCY <- IICS0.ACKD0
SMSI11	8101H	BNC 16	If ACK=0, branch to SMSI27
SMSI12	4220H	SET1 [MSG2+0].2	IICCTL00.ACKE0 <- 1
SMSI13	5230H	CLR1 [MSG2+0].3	IICCTL00.WTIM0 <- 0
SMSI14	4250H	SET1 [MSG2+0].5	IICCTL00.WRELO <- 1
SMSI15	B330H	WHILE0 [MSG3+0].3	Wait until IFxx.IICAIFn = 1
SMSI16	5330H	CLR1 [MSG3+0].3	IF1L.IICAIF0 <- 0
SMSI17	14A0H	MOV MSG10, [MSG4+0]	MSG10 <- IICAn
SMSI18	0BA0H	MOV [MSG11+0], MSG10	Stored Address <- MSG10
SMSI19	7B80H	ADDW MSG11, MSG8	MSG11 <- MSG11 + 1
SMSI20	7B72H	CMPW MSG11, MSG7	When MSG11 < MSG7
SMSI21	8F90H	BC -7	Branch to SMSI14
SMSI22	5220H	CLR1 [MSG2+0].2	IICCTL00.ACKE0 <- 0
SMSI23	4230H	SET1 [MSG2+0].3	IICCTL00.WTIM0 <- 1
SMSI24	4250H	SET1 [MSG2+0].5	IICCTL00.WRELO <- 1
SMSI25	7FF2H	CMPW MSG15, MSG15	Comparison for branching to SMSI0
SMSI26	8E62H	BZ -26	Branch to SMSI0
SMSI27	2CF0H	MOVW [MSG12+0], MSG15	MSG1 <- FFFFH
SMSI28	F001H	WAKEUP	WAKEUP
SMSI29	0000H	Unused	-
SMSI30	0000H	Unused	-
SMSI31	0000H	Unused	-

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)

Register Symbol	Setting	Contents	Remark
SMSI0	2DF2H	MOVW [MSG13+2], MSG15	SMSI29 <- FFFFH
SMSI1	F002H	DTCTRG	DTC startup
SMSI2	AD02H	WHILE1 [MSG13+2].0	Waiting for DTC transfer completion
SMSI3	B330H	WHILE0 [MSG3+0].3	Wait until IFxx.IICAIFn = 1
SMSI4	5330H	CLR1 [MSG3+0].3	IF1L.IICAIF0 <- 0
SMSI5	4200H	SET1 [MSG2+0].0	IICCTLn0.SPT0 <- 1
SMSI6	9A10H	WAIT 161, 0	Waiting for stop condition
SMSI7	16A0H	MOV MSG10, [MSG6+0]	MSG10 <- humi_h
SMSI8	79A2H	CMPW MSG9, MSG10	When threshold < humi_h
SMSI9	8020H	BC 2	Branch to SMSI11
SMSI10	F000H	FINISH	FINISH
SMSI11	F001H	WAKEUP	WAKEUP
SMSI12	0000H	Unused	-
SMSI13	0000H	Unused	-
SMSI14	0000H	Unused	-
SMSI15	0000H	Unused	-
SMSI16	0000H	Unused	-
SMSI17	0000H	Unused	-
SMSI18	0000H	Unused	-
SMSI19	0000H	Unused	-
SMSI20	0000H	Unused	-
SMSI21	0000H	Unused	-
SMSI22	0000H	Unused	-
SMSI23	0000H	Unused	-
SMSI24	0000H	Unused	-
SMSI25	0000H	Unused	-
SMSI26	0000H	Unused	-
SMSI27	0000H	Unused	-
SMSI28	0000H	Unused	-
SMSI29	0000H	Unused	-
SMSI30	0000H	Unused	-
SMSI31	0000H	Unused	-

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**

Tag name	Components	Contents
Clocks	-	Operation mod: High-speed main mode 2.4 (V) ~ 5.5 (V) EV <sub>DD</sub> setting: $1.8V \leq EV_{DD0} < 5.5V$ High-speed on-chip oscillator: 32MHz f <sub>IHP</sub> : 32MHz f <sub>CLK</sub> : 32MHz (High-speed on-chip oscillator) f <sub>SXP</sub> : 32.768kHz (Low-speed on-chip oscillator)
System	-	On-chip debug operation setting: COM port <sup>Note 1</sup> Pseudo-RRM/DMM function setting: Used Start/Stop function setting: Unused Trace function setting: Used Security ID setting: Use security ID Security ID : 0x00000000000000000000 Security ID authentication failure setting: Do not erase flash memory data
Components	r_bsp	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
	Config_LVDD0	Operation mode setting: Reset mode Voltage detection setting: Reset generation level (V <sub>LVDD0</sub> ): 1.86 (V)

Table 6-2 Parameters of Smart Configurator

Tag name	Components	Contents
Components	Config_DTC	DTC base address: 0xFFD00 Control data0 (DTCD0): Event output from the SNOOZE mode sequencer [DTCD0] Transfer mode setting: Repeat mode Transfer data size setting: 16 bits Repeat mode interrupt setting: Disable Repeat area setting: Transfer source Transfer source address: 0xE000 Transfer destination address: 0x0386 Address fixed Transfer count: 3 Transfer block size: 54
	Config_ITL000_ITL001	Components: Interval timer Operation mode: 16 bit count mode Resource: ITL000_ITL001 Operation clock: $f_{SXP}$ Clock source: $f_{ITL0}/128$ Interval value: 5000 ms Interrupt setting: unused
	Config_PORT	Components: Port Port selection: PORT5 P53: Out (Output 1)
	Config_IICA0	Clock mode setting: $f_{CLK}/2$ Address: 16 Operation mode setting: Standard Transfer clock ( $f_{SCL}$ ): 100000 (bps) Communication end interrupt priority: Level 3 (low) Callback function setting: Master transmission end, Master reception end

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: e<sup>2</sup> 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**

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

## 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  $V_{IL}$  (Max.) and  $V_{IH}$  (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  $V_{IL}$  (Max.) and  $V_{IH}$  (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.



## Notice

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 system; undersea 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 Semiconductor 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 danger 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.

(Rev.5.0-1 October 2020)

## Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,  
Koto-ku, Tokyo 135-0061, Japan  
[www.renesas.com](http://www.renesas.com)

## Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

## Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:  
[www.renesas.com/contact/](http://www.renesas.com/contact/).