# RL78/G15

## Digital angle meter using I2C communication

### Introduction

This application note explains how to receive acceleration data from a gyro sensor (BMX055), convert it into angle data, and display the sensor's tilt information on an LCD (ACM1602NI-FLW-FBW-M01) using I2C communication.

### Target Device

RL78/G15

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.

## Content

# 1. Specifications

## 1.1 Basic specifications as a master on the I2C bus

The specifications for the I2C bus are as follows;

・Standard mode (transfer clock 85 kbps)

Target slaves (LCD display, gyro sensor)

・LCD display (ACM1602NI-FLW-FBW-M01) address : 0b1010000 (0x50)

・Gyro sensor (BMX055) address : 0b1101001 (0x69)

Table 1-1 shows the peripheral functions used and their use.

**Table 1-1 Peripheral functions used and their use**

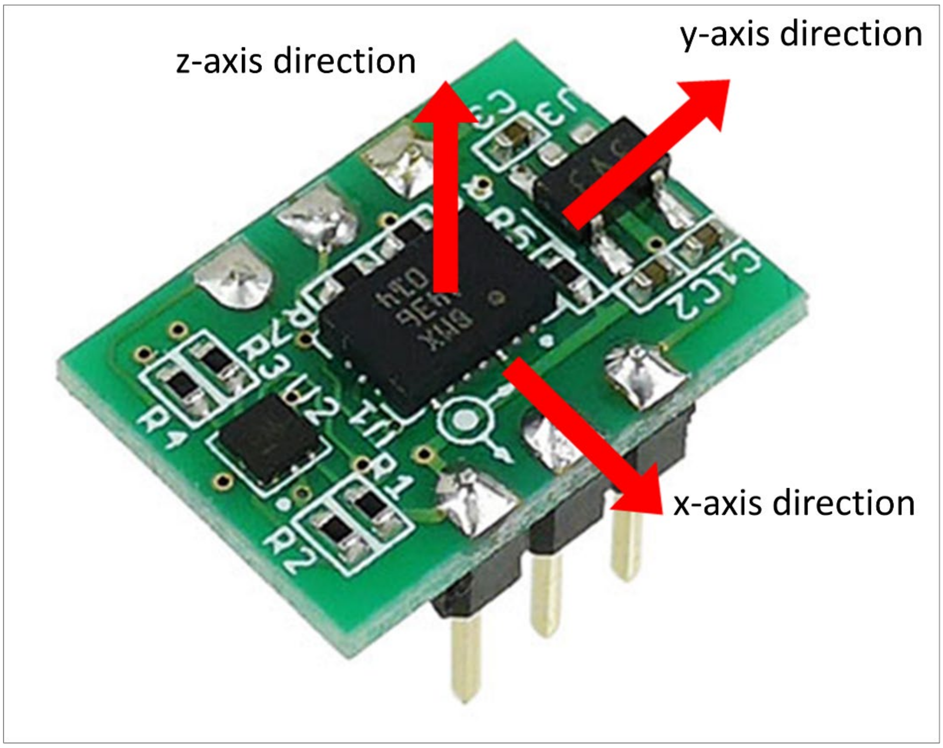
| Peripheral function | Use |
| --- | --- |
| IICA0 | Operation as a master function on the I2C bus |
| TAU0 | 1ms *interval timer interrupt |

## 1.2 Overview of Operation

Specify the registers of the gyro sensor (BMX055) and read the acceleration data one byte at a time. Convert that data into angles and send it to the LCD display (ACM1602NI-FLW-FBW-M01) to visualize the angles in the x and y directions.

Below, the axis directions (Figure 1-1) and the method for calculating angles for the gyro sensor (BMX055) are described.

**Figure 1-1 The axis direction of the gyro sensor (BMX055)**



<Method for calculating angles>

The data sent from the gyro sensor (BMX055) is 16-bit acceleration data, but it is initially configured with a full scale of ±125 (degree/s). To convert it to units of degrees/s, you can use the following calculation.

Angular Velocity (degree/s) = Gyro × 0.0038

*The LSB of the 16-bit data is approximately 0.0038 (since 250/65536 ≈ 0.0038).

Next, you can convert the angular velocity (degree/s) obtained from the above formula to angles (degree).

Since you are receiving data from the gyro sensor (BMX055) every 100 milliseconds, you can calculate the angle using the following formula.

Angle (degree) = ( (Angular Velocity (degree/s) ) + (Previous Angular Velocity (degree/s) ) ) × 0.1 (s) ÷ 2

Below are the initial settings for each function, the gyro sensor, and the LCD display.

(1) IICA0 initialization

＜Setting conditions＞

- Use the P06 pin as the SCLA0 pin.
- Use the P07 pin as the SDAA0 pin.
- Set the operation mode to standard mode and the transfer clock ($f_{SCL}$) to 85 kbps.
- For the interrupt priority level, select "Level 3 lowest priority (default)".

(2) TAU0 initialization

＜Setting conditions＞

- Set the operating clock to CK00.
- Set the clock source to $f_{CLK}$ (16000 kHz).
- Set the interval time to 1ms. (Do not generate INTTM0 interrupt at the start of counting.)
- For the interrupt priority level, select "Level 0 highest priority".

(3) LCD display (ACM1602NI-FLW-FBW-M01) initialization

＜Setting conditions＞

- The address is set to 0x50.
- Display the angle in the x-axis direction on the first line and the angle in the y-axis direction on the second line.

(4) Gyro sensor (BMX055) initialization

＜Setting conditions＞

- The address is set to 0x69.
- Set the measurement range to ±125 (degree/s).
- Set the output rate to 100 (Hz).
- Set the power mode to Normal mode (default).

## 2. Operation Evaluate Conditions

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

**Table 2-1 Operation Check Conditions**

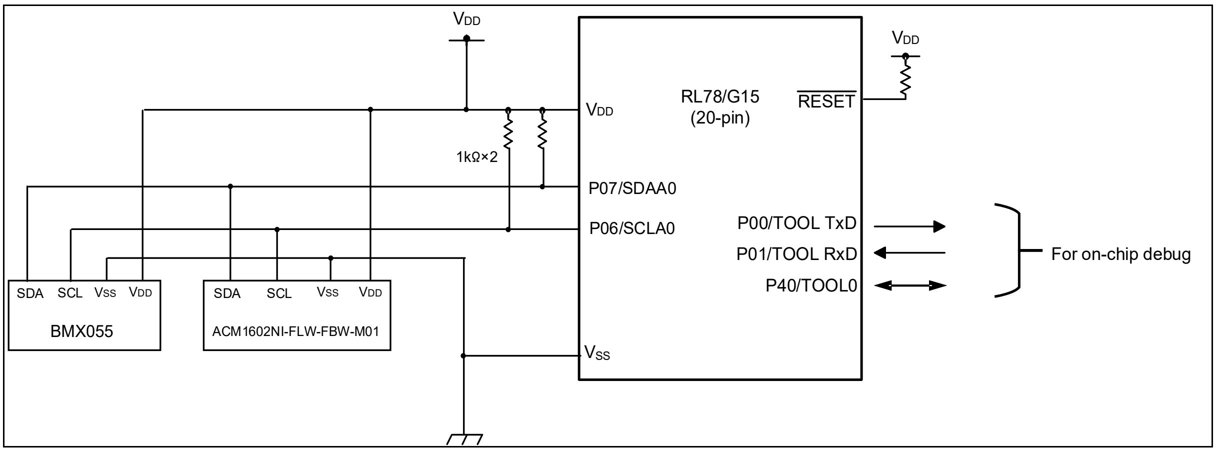
| Item | Description |
|---|---|
| Microcontroller used | RL78/G15 (R5F12608ASP) |
| Board to be used | RL78/G15-20p Fast Prototyping Board (RTK5RLG150C00000BJ) |
| Operating frequency | HOCO ( $f_{IH}$ ) : 16MHz |
| Operating voltage | 5.0V (Operation is possible over a voltage range of 2.4 to 5.5V.) |
| Integrated development environment( CS+ ) | CS+ for CC V8.09.00 from Renesas Electronics Corp. |
| Assembler ( CS+ ) | CC-RL V1.12.00 from Renesas Electronics Corp. |
| Integrated development environment ( e2studio ) | e2studio V2023-04( 23.4.0 ) from Renesas Electronics Corp. |
| Assembler( e2studio ) | CC-RL V1.12.00 from Renesas Electronics Corp. |
| Integrated development environment ( IAR) | IAR Embedded Workbench for Renesas RL78 V5.10.1 from IAR Systems. |
| Assembler( IAR ) | IAR C/C++ Compiler for Renesas RL78 V5.10.1 from IAR Systems. |
| Smart Configurator ( SC ) | V1.6.0 |
| Board Support Package( BSP ) | V1.60 |

# 3. Description of the Hardware

## 3.1 Hardware Configuration Example

Figure 3-1 shows an example hardware configuration used in this application note.

**Figure 3-1 Hardware Configuration**




Note. This simplified circuit diagram was created to show an overview of connections only. When actually designing your circuit, make sure the design includes appropriate pin handling and meets electrical characteristic requirements (connect each input-only port to $V_{DD}$ or $V_{SS}$ through a resistor).

## 3.2 List of Pins to be Used

Table 3-1 lists the pins to be used and their functions.

**Table 3-1 Pins to be Used and their Functions**

| Pin Name | I/O | Description |
|---|---|---|
| P06/SCLA0 | Output | Serial Clock |
| P07/SDAA0 | Output | Serial Data Bus |

Note. This application note only demonstrates the handling of the used pins. When creating an actual circuit, please ensure appropriate pin handling and design the circuit to meet electrical characteristics.

# 4. Description of the Software

## 4.1 List of Option Byte Settings

Table 4-1 summarizes the settings of the option bytes.

**Table 4-1 Option Byte Settings**

| Address | Value | Description |
|---|---|---|
| 000C0H | 11101111B | Disables the watchdog timer. (Stops counting after the release from the reset state.) |
| 000C1H | 11110111B | At Rising: TYP. 2.90V<br>At Falling: TYP.2.84V |
| 000C2H | 11111001B | HOCO：16MHz |
| 000C3H | 10000101B | Enables the on-chip debugger |

## 4.2 List of functions

Table 4-2 shows the list of functions.

**Table 4-2 List of functions**

| Function Name | Overview |
|---|---|
| main() | Main processing |
| R_init_LCD() | LCD initialization function |
| R_init_sensor() | Gyro sensor initialization function |
| R_IICA0_start_condition() | Start condition generation function |
| R_IICA0_stop_condition() | Stop condition generation function |
| R_IICA0_transmit() | I2C transmission function |
| send_command_LCD() | Command transmission function (LCD) |
| send_data_LCD() | Data transmission function (LCD) |
| send_range_sensor() | Function for setting the measurement range for the gyro sensor |
| send_rate_sensor() | Function for setting the output rate for the gyro sensor |
| send_powermode_sensor() | Function for setting the power mode for the gyro sensor |
| send_register_select_sensor() | Function for setting registers in the gyro sensor |
| R_IICA0_receive_sensor_data() | I2C receive function |
| R_gyrodata_angle() | Angle calculation function |
| wait_3ms() | 3ms delay function |
| Gyro_modify | Gyro drift correction function |
| r_Config_TAU0_0_interrupt() | 1ms interval timer interrupt processing function |

### 4.3 Function Specifications

Below are the function specifications for the sample code.

| **main()** | |
|---|---|
| **Overview** | Main processing |
| **Header** | r_smc_entry.h |
| **Declaration** | void main(void) |
| **Explanation** | Initialize the LCD and gyro sensor, and perform periodic tasks. |
| **Arguments** | — |
| **Return value** | — |

| R_init_LCD() | |
|---|---|
| **Overview** | LCD initialization function |
| **Header** | r_smc_entry.h |
| **Declaration** | void R_init_LCD(void) |
| **Explanation** | Use the serial data bus to send initial data to the LCD. |
| **Arguments** | — |
| **Return value** | — |

| R_init_sensor() | |
|---|---|
| **Overview** | Gyro sensor initialization function |
| **Header** | r_smc_entry.h |
| **Declaration** | void R_init_sensor(void) |
| **Explanation** | Use the serial data bus to set the measurement range, output rate, and power mode of the gyro sensor. |
| **Arguments** | — |
| **Return value** | — |

| R_IICA0_start_condition() | |
|---|---|
| **Overview** | Start condition generation function |
| **Header** | r_smc_entry.h |
| **Declaration** | void R_IICA0_start_condition(uint8_t addr) |
| **Explanation** | Generate a start condition and send the address and address direction. |
| **Arguments** | uint8_t addr |
| **Return value** | — |

| R_IICA0_stop_condition() | |
|---|---|
| **Overview** | Stop condition generation function |
| **Header** | r_smc_entry.h |
| **Declaration** | void R_IICA0_stop_condition(void) |
| **Explanation** | Generating a stop condition. |
| **Arguments** | uint8_t addr |
| **Return value** | — |

| R_IICA0_transmit() | |
|---|---|
| **Overview** | I2C transmission function |
| **Header** | r_smc_entry.h |
| **Declaration** | void R_IICA0_transmit(uint8_t data) |
| **Explanation** | Assign the transmit data to the shift register and transmit the data. |
| **Arguments** | uint8_t data |
| **Return value** | — |

| send_command_LCD() | |
|---|---|
| **Overview** | Command transmission function (LCD) |
| **Header** | r_smc_entry.h |
| **Declaration** | void send_command_LCD(uint8_t comcode) |
| **Explanation** | Send the command value represented by the variable comcode to the LCD. |
| **Arguments** | uint8_t comcode |
| **Return value** | — |

| send_data_LCD() | |
|---|---|
| **Overview** | Data transmission function (LCD) |
| **Header** | r_smc_entry.h |
| **Declaration** | void send_data_LCD(uint8_t datacode) |
| **Explanation** | Send the data represented by the variable datacode to the LCD. |
| **Arguments** | uint8_t datacode |
| **Return value** | — |

| send_range_sensor() | |
|---|---|
| **Overview** | Function for setting the measurement range for the gyro sensor |
| **Header** | r_smc_entry.h |
| **Declaration** | void send_range_sensor(uint8_t datacode) |
| **Explanation** | Send the Measurement Range Setting Register address (0x0F) and the setting value represented by the variable datacode to the gyro sensor. |
| **Arguments** | uint8_t datacode |
| **Return value** | — |

| send_rate_sensor() | |
|---|---|
| **Overview** | Function for setting the output rate for the gyro sensor |
| **Header** | r_smc_entry.h |
| **Declaration** | void send_rate_sensor(uint8_t datacode) |
| **Explanation** | Send the Output Rate Setting Register address (0x10) and the setting value represented by the variable datacode to the gyro sensor. |
| **Arguments** | uint8_t datacode |
| **Return value** | — |

| send_powermode_sensor() | |
|---|---|
| **Overview** | Function for setting the power mode for the gyro sensor |
| **Header** | r_smc_entry.h |
| **Declaration** | void send_powermode_sensor(uint8_t datacode) |
| **Explanation** | Send the Power Mode Setting Register address (0x11) and the setting value represented by the variable datacode to the gyro sensor. |
| **Arguments** | uint8_t datacode |
| **Return value** | — |

| send_register_select_sensor() | |
|---|---|
| **Overview** | Function for setting registers in the gyro sensor |
| **Header** | r_smc_entry.h |
| **Declaration** | void send_register_select_sensor(uint8_t datacode) |
| **Explanation** | Send the register address value represented by the variable datacode to the gyro sensor. |
| **Arguments** | uint8_t datacode |
| **Return value** | — |

| R_IICA0_receive_sensor_data() | |
|---|---|
| **Overview** | I2C receive function |
| **Header** | r_smc_entry.h |
| **Declaration** | void R_IICA0_receive_sensor_data(void) |
| **Explanation** | Read data from the gyro sensor. |
| **Arguments** | — |
| **Return value** | — |

| R_gyrodata_angle() | |
|---|---|
| **Overview** | Angle calculation function |
| **Header** | r_smc_entry.h |
| **Declaration** | void R_gyrodata_angle(void) |
| **Explanation** | Convert the data received from the gyro sensor into angle data. |
| **Arguments** | — |
| **Return value** | — |

| wait_3ms() | |
|---|---|
| **Overview** | 3ms delay function |
| **Header** | r_smc_entry.h |
| **Declaration** | void wait_3ms(void) |
| **Explanation** | Count for 3 milliseconds. |
| **Arguments** | — |
| **Return value** | — |

| Gyro_modify() | |
|---|---|
| **Overview** | Gyro drift correction function |
| **Header** | r_smc_entry.h |
| **Declaration** | void Gyro_modify(void) |
| **Explanation** | Correct the angle by 1 degree for the X-axis every 12 seconds and for the Y-axis every 6.8 seconds. |
| **Arguments** | — |
| **Return value** | — |

**r_Config_TAU0_0_interrupt()**

| | |
|---|---|
| **Overview** | 1ms interval timer interrupt processing function |
| **Header** | r_cg_macrodriver.h、r_cg_userdefine.h、Config_TAU0_0.h |
| **Declaration** | static void __near r_Config_TAU0_0_interrupt(void) |
| **Explanation** | Perform a 1 ms interval timer interrupt processing. |
| **Arguments** | — |
| **Return value** | — |

## 4.4 List of constants

Table 4-3 lists the constants that are used in this sample program.

**Table 4-3 Constants for the Sample Program**

| Constant | Setting | Description |
|---|---|---|
| CLRDISP | 0x01 | Clearing the LCD display |
| LCD_Mode | 0x38 | Setting values for the operation mode |
| DISPON | 0x0C | Processing for turning on the display |
| ENTRY_Mode | 0x06 | Processing for the entry mode |
| slaveaddr_lcd | 0xA0 | Address value of the LCD |
| slaveaddr_sensor_W | 0xD2 | Gyro sensor address value + transfer direction (Write) |
| slaveaddr_sensor_R | 0xD3 | Gyro sensor address value + transfer direction (Read) |

## 4.5 List of variables

Table 4-4 lists the global variables that are used in this sample program.

**Table 4-4 Global Variables for the Sample Program**

| Type | Variable Name | Contents | Function Used |
|---|---|---|---|
| uint8_t | lcd_number_data[] | LCD display patterns for numbers 0 to 9 | main() |
| uint8_t | counter_3ms | 3ms counter | r_Config_TAU0_0_interrupt() |
| uint8_t | flag_3ms_start | 3ms count start flag | r_Config_TAU0_0_interrupt(), wait_3ms() |
| uint8_t | counter_100ms | 100ms counter | main()<br>r_Config_TAU0_0_interrupt() |
| uint8_t | flag_100ms | 100ms count flag | main()<br>r_Config_TAU0_0_interrupt() |
| uint8_t | counter_12s | 12s counter | main()<br>r_Config_TAU0_0_interrupt() |
| uint8_t | flag_12s | 12s count flag | main()<br>r_Config_TAU0_0_interrupt() |
| uint8_t | counter_6800ms | 6.8s counter | main()<br>r_Config_TAU0_0_interrupt() |
| uint8_t | flag_6800ms | 6.8s count flag | main()<br>r_Config_TAU0_0_interrupt() |
| float | xGyro | x-axis gyro data | R_gyrodata_angle() |
| float | yGyro | y-axis gyro data | R_gyrodata_angle() |
| float | xGyro_past | x-axis gyro past data | main(),<br>R_gyrodata_angle() |
| float | yGyro_past | y-axis gyro past data | main(),<br>R_gyrodata_angle() |
| float | xAngle | x-axis angle | R_gyrodata_angle() |
| int | xAngle_disp | x-axis angle (for LCD display) | main(),<br>R_gyrodata_angle() |
| float | yAngle | y-axis angle | R_gyrodata_angle() |
| int | yAngle_disp | y-axis angle (for LCD display) | main(),<br>R_gyrodata_angle() |
| uint8_t | data_receive_counter | Indicating the number of data receptions | main() |

## 4.6 Flowchart

### 4.6.1 Main Processing

Figure 4-1 through Figure 4-3 show the flowchart for the main processing.

**Figure 4-1 Main Processing (1/3)**

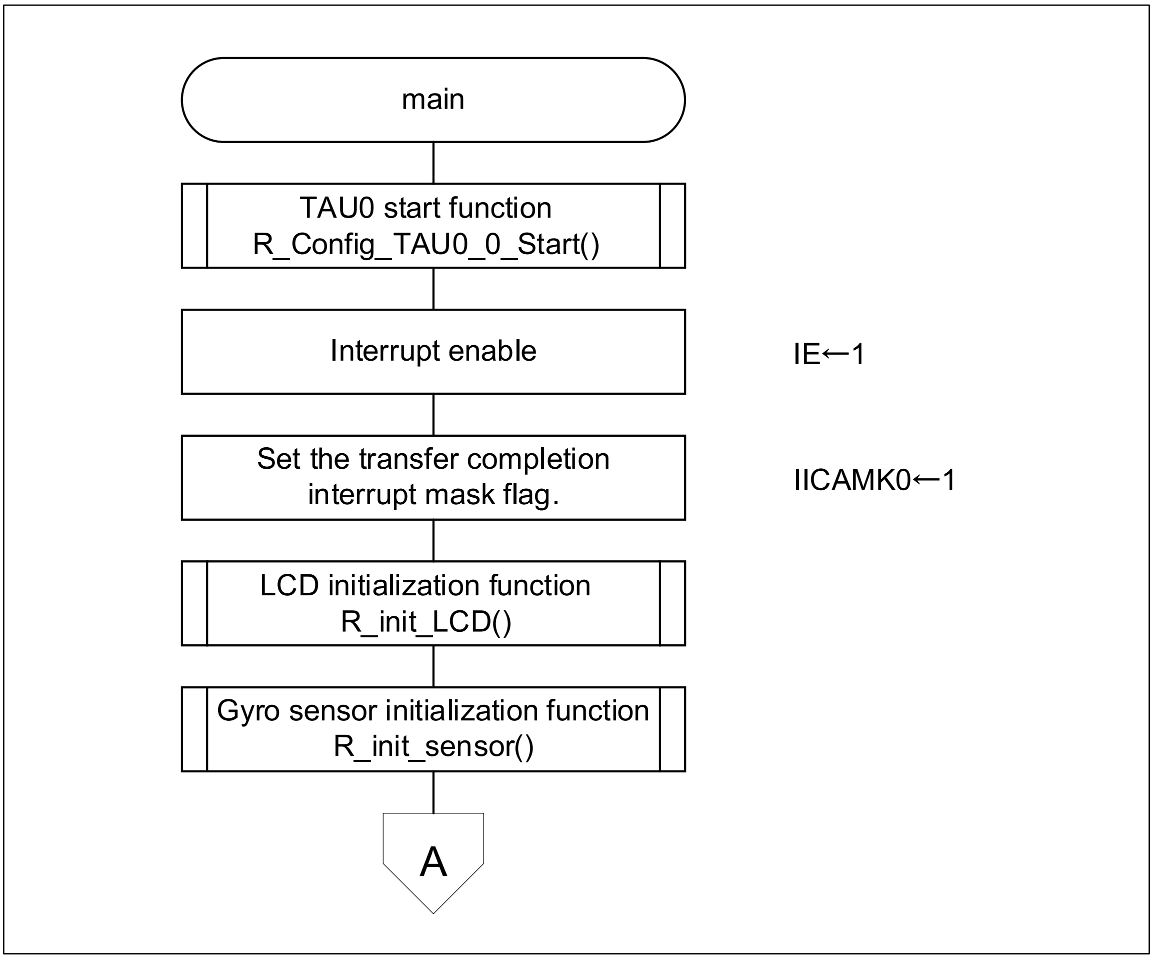


**Figure 4-2 Main Processing (2/3)**

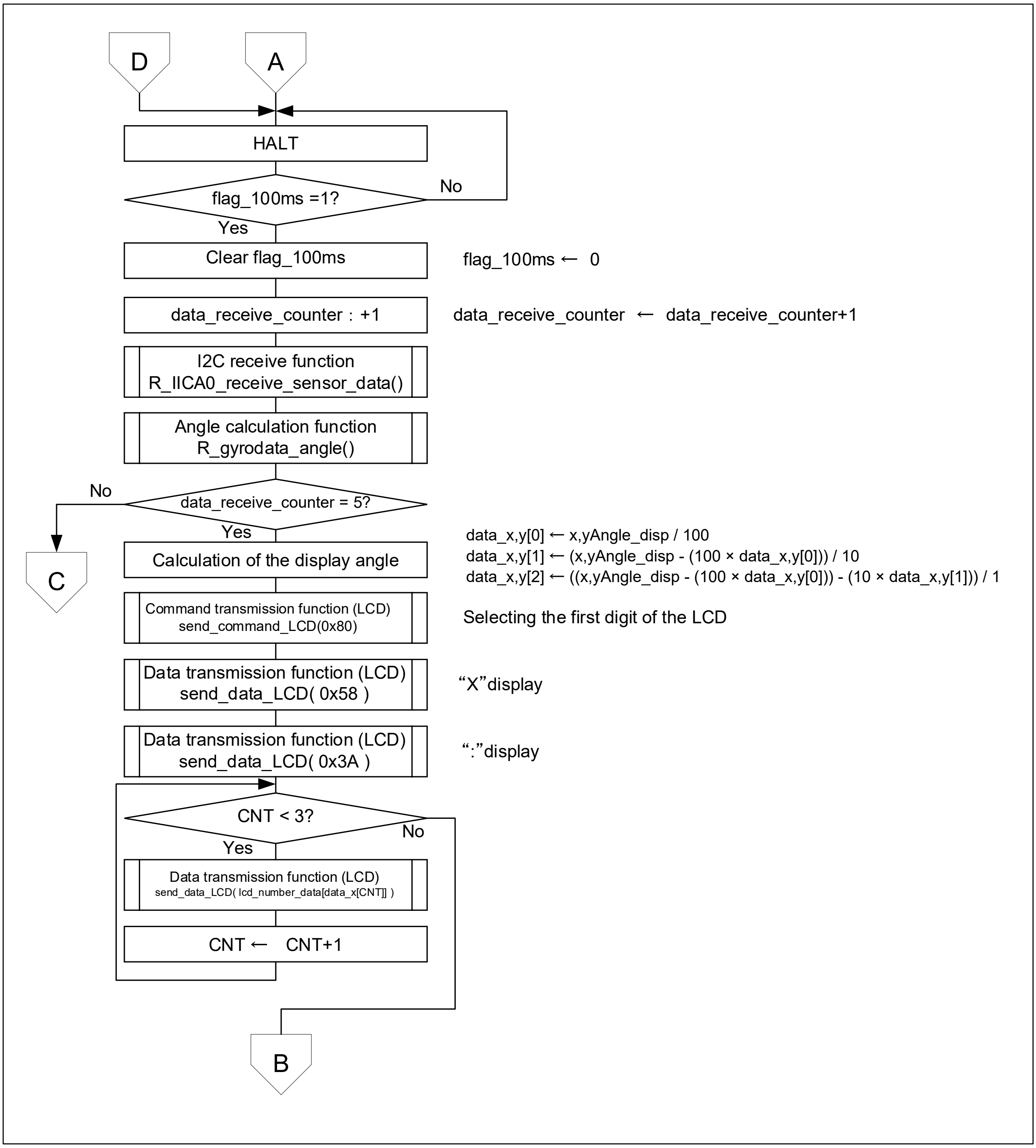
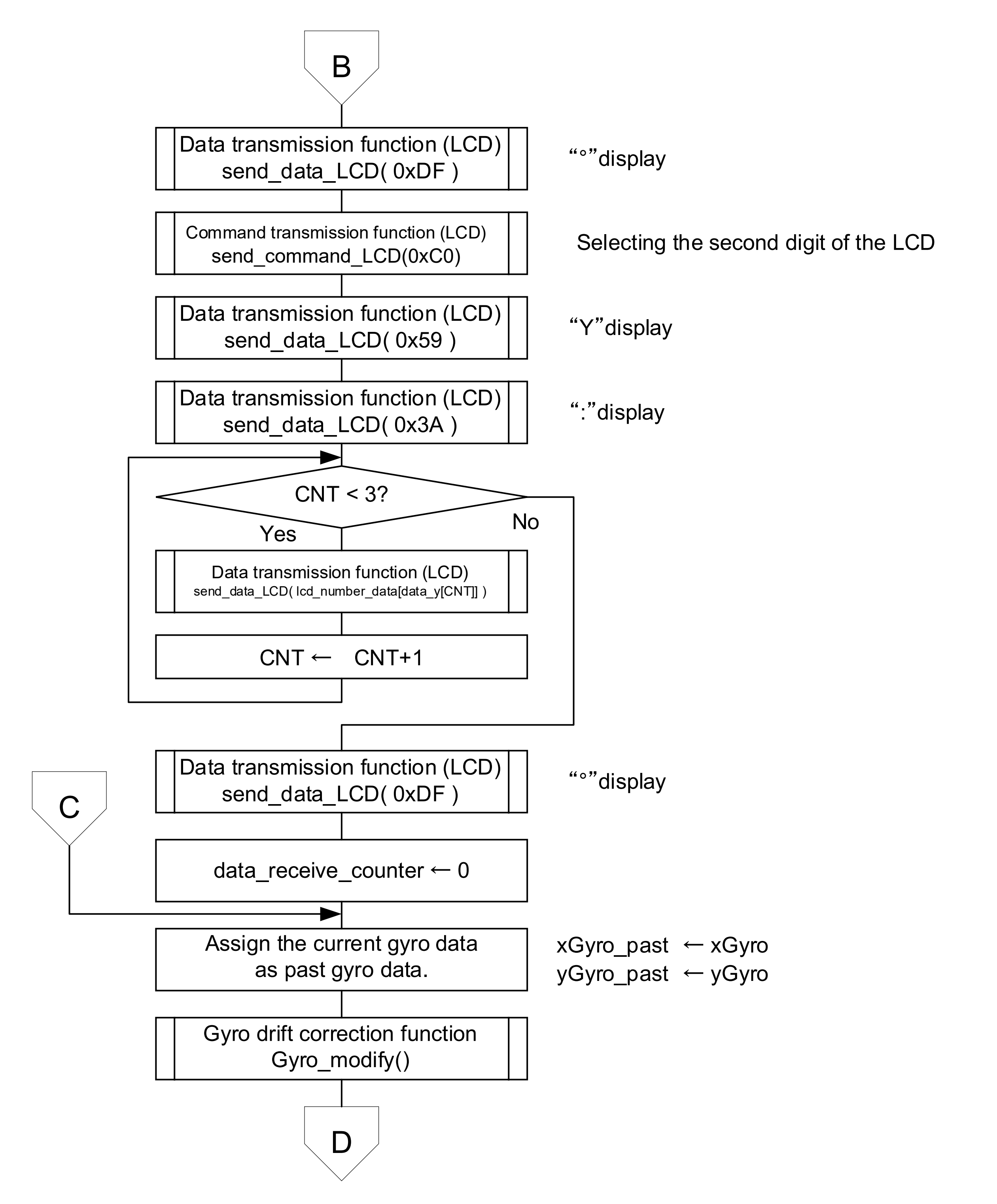


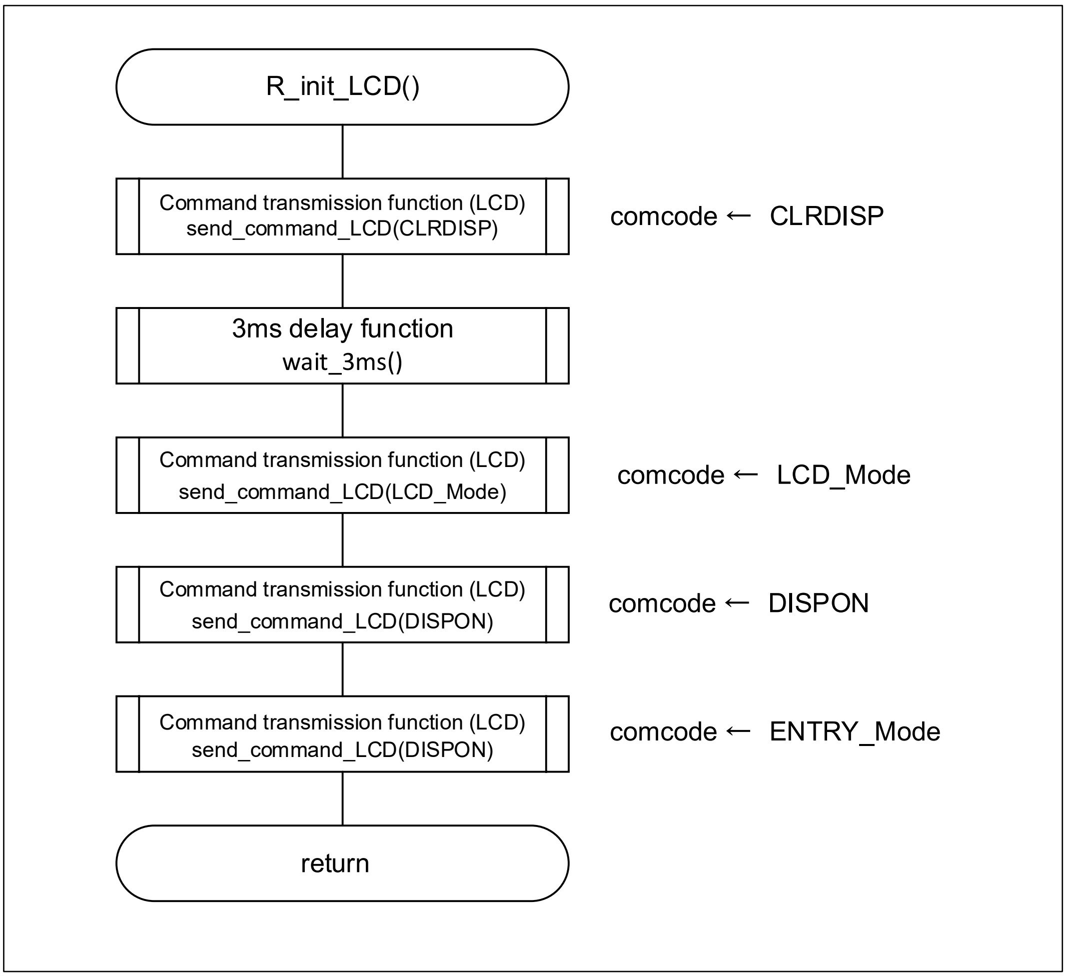
**Figure 4-3 Main Processing (3/3)**



### 4.6.2 LCD initialization function

Figure 4-4 show the flowchart for LCD initialization function.

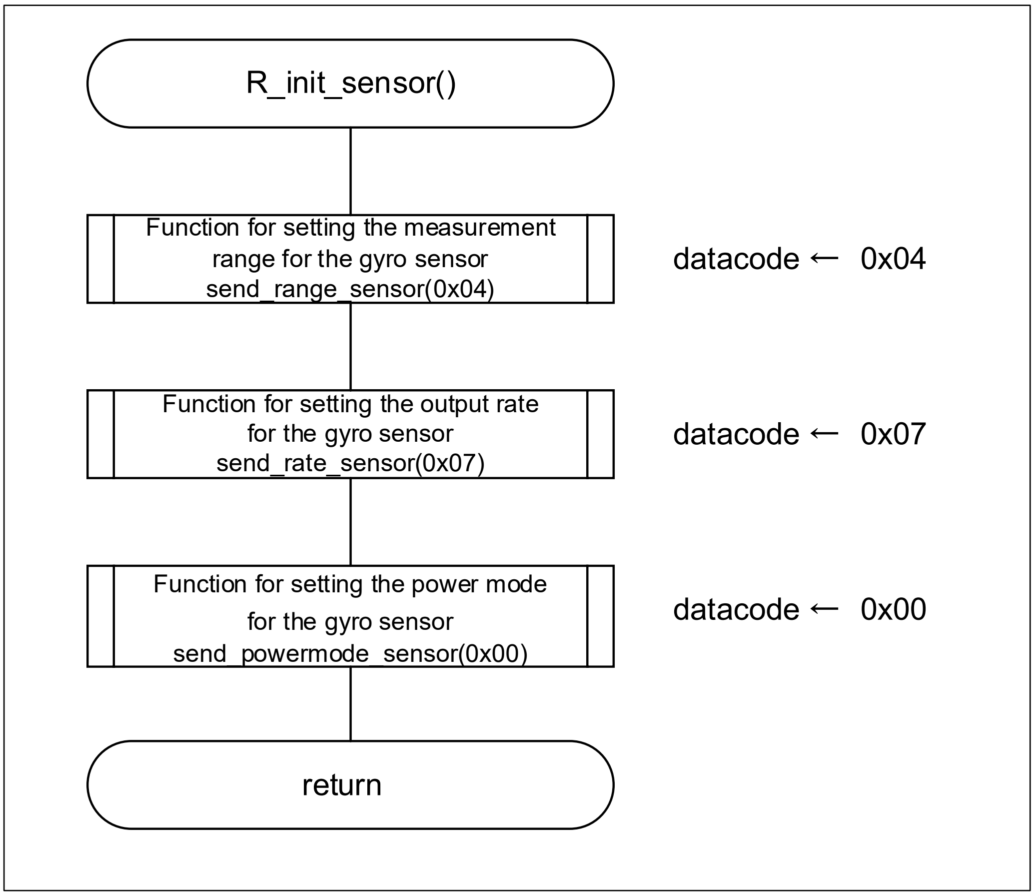
**Figure 4-4 LCD initialization function**



### 4.6.3 Gyro sensor initialization function

Figure 4-5 show the flowchart for Gyro sensor initialization function.

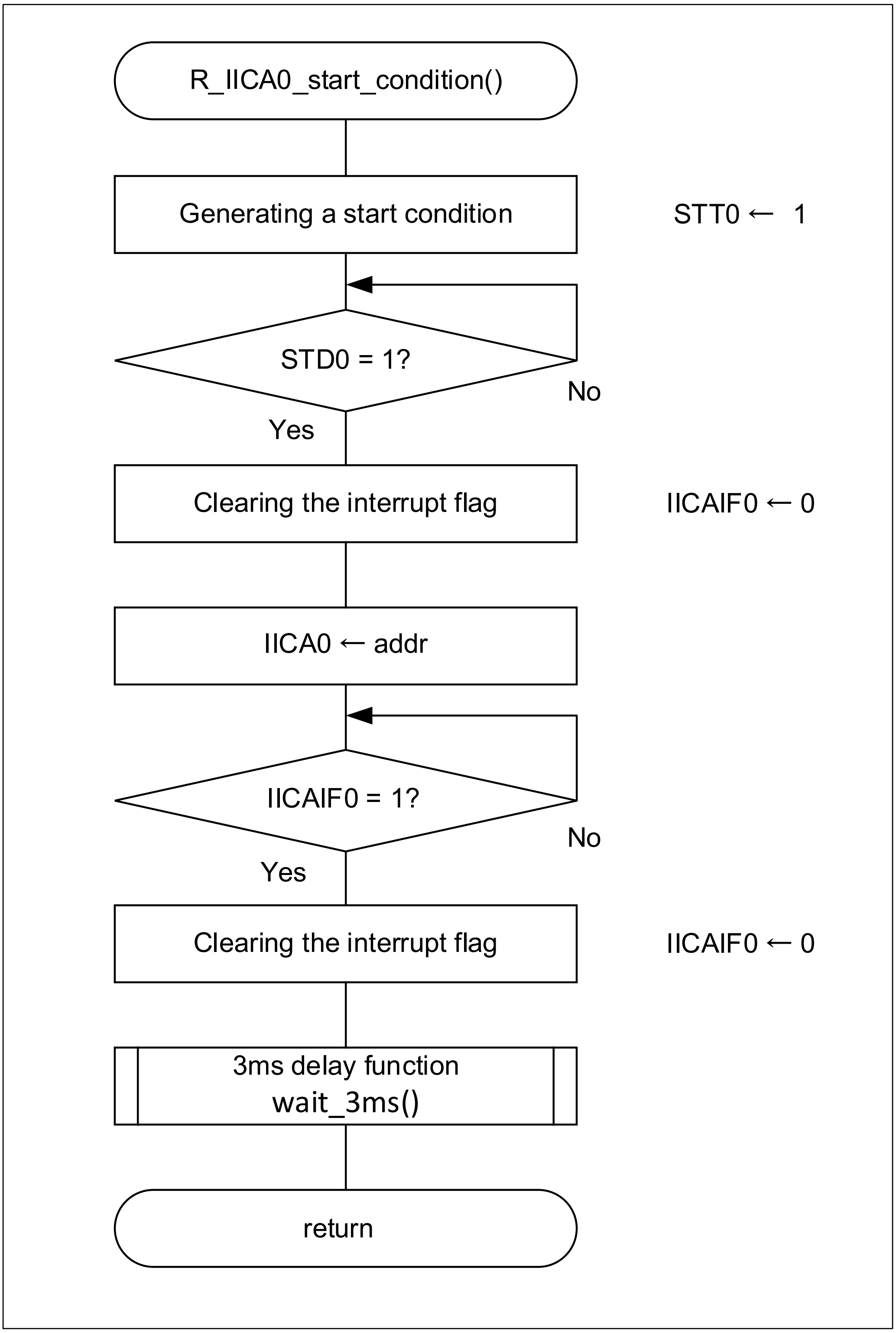
**Figure 4-5 Gyro sensor initialization function**



### 4.6.4 Start condition generation function

Figure 4-6 show the flowchart for Start condition generation function.

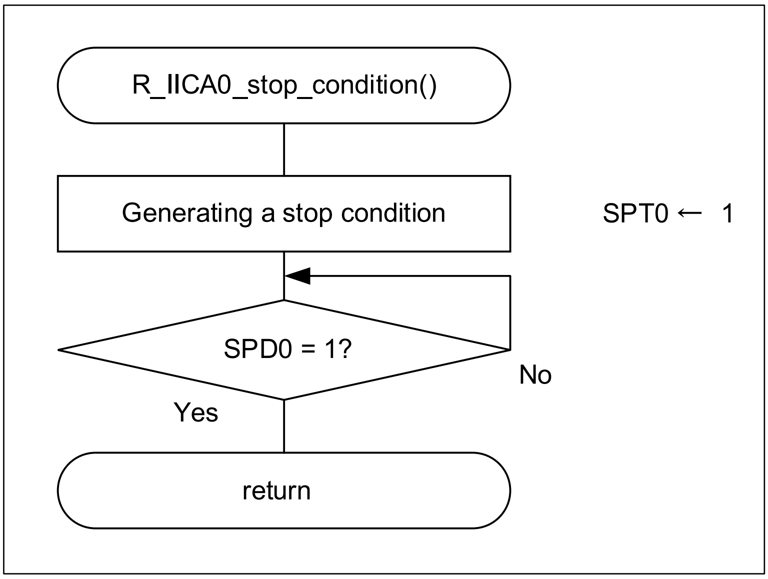
**Figure 4-6 Start condition generation function**



### 4.6.5 Stop condition generation function

Figure 4-7 show the flowchart for Stop condition generation function.

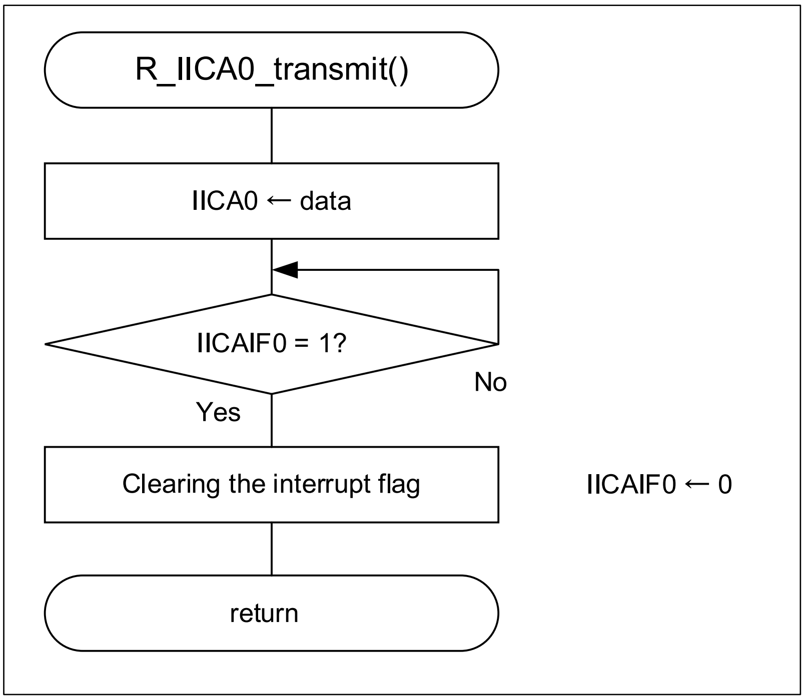
**Figure 4-7 Stop condition generation function**




### 4.6.6 I2C transmission function

Figure 4-8 show the flowchart for I2C transmission function.

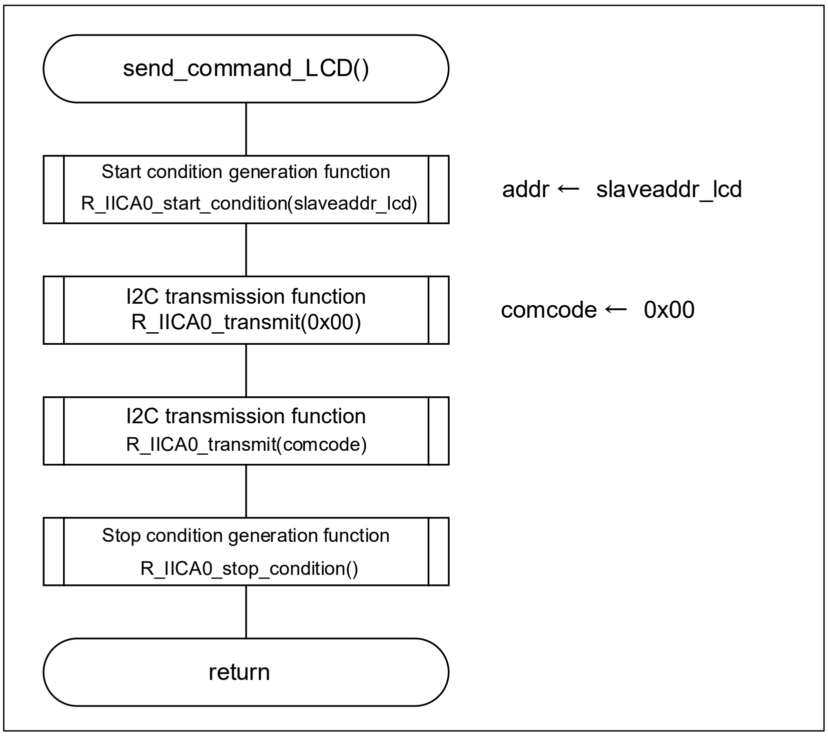
**Figure 4-8 I2C transmission function**



### 4.6.7 Command transmission function (LCD)

Figure 4-9 show the flowchart for Command transmission function (LCD).

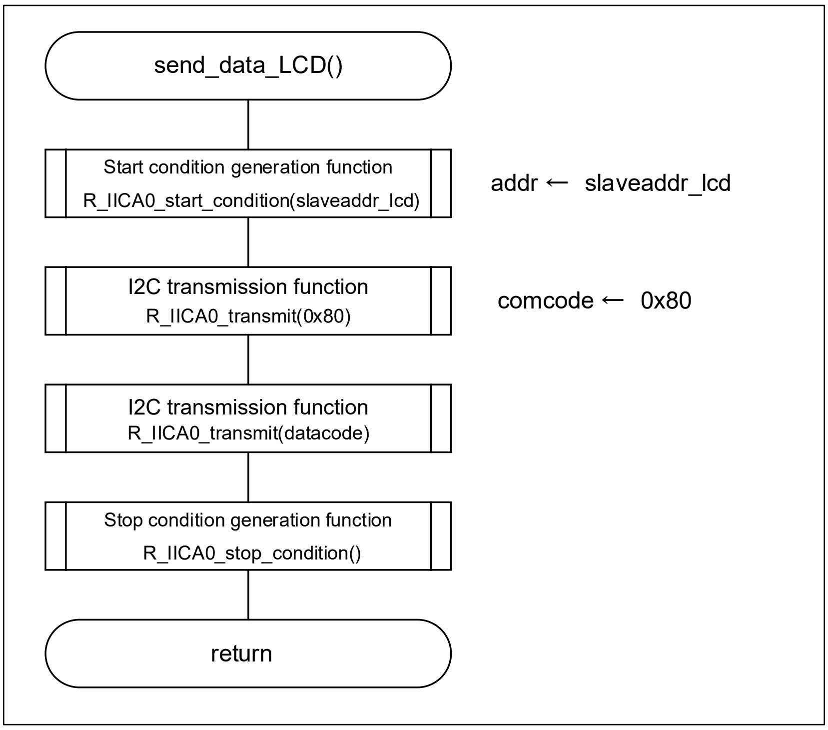
**Figure 4-9 Command transmission function (LCD)**




### 4.6.8 Data transmission function (LCD)

Figure 4-10 show the flowchart for Data transmission function (LCD).

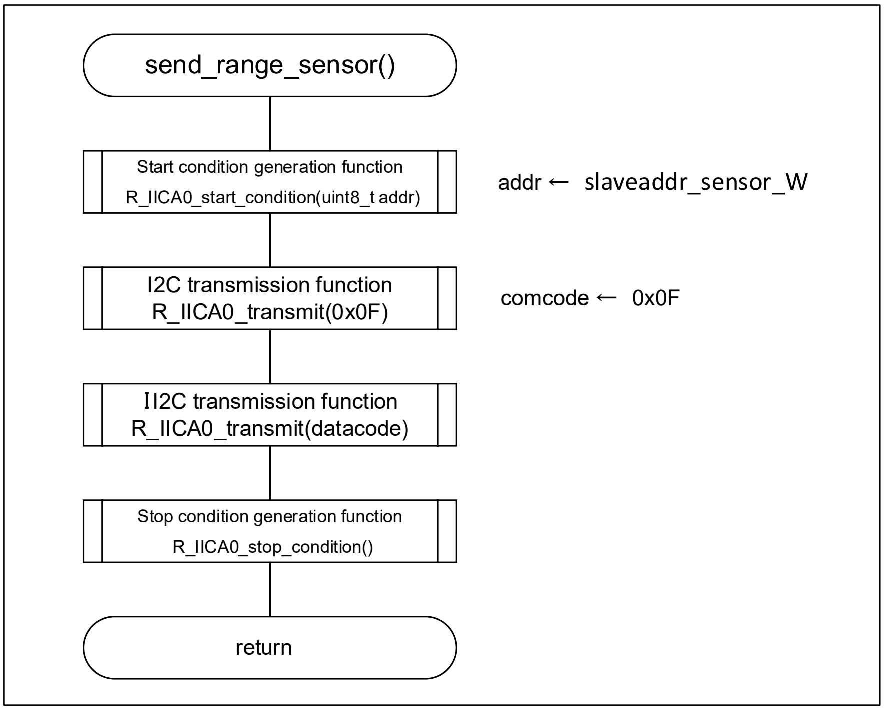
**Figure 4-10 Data transmission function (LCD)**



### 4.6.9 Function for setting the measurement range for the gyro sensor

Figure 4-11 show the flowchart for Function for setting the measurement range for the gyro sensor.

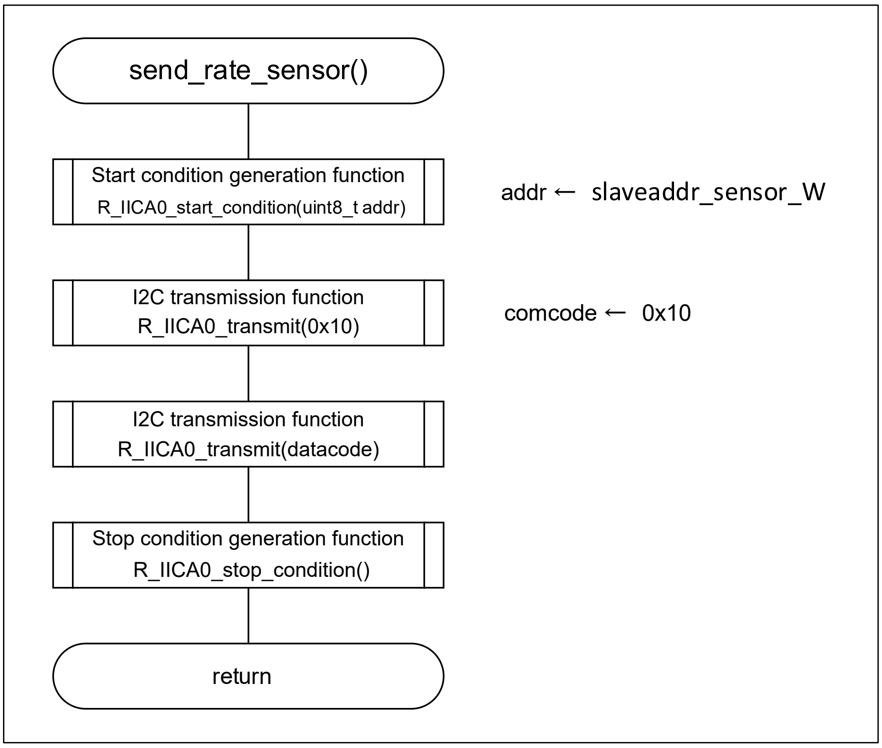
**Figure 4-11 Function for setting the measurement range for the gyro sensor**




### 4.6.10 Function for setting the output rate for the gyro sensor

Figure 4-12 show the flowchart for Function for Function for setting the output rate for the gyro sensor.

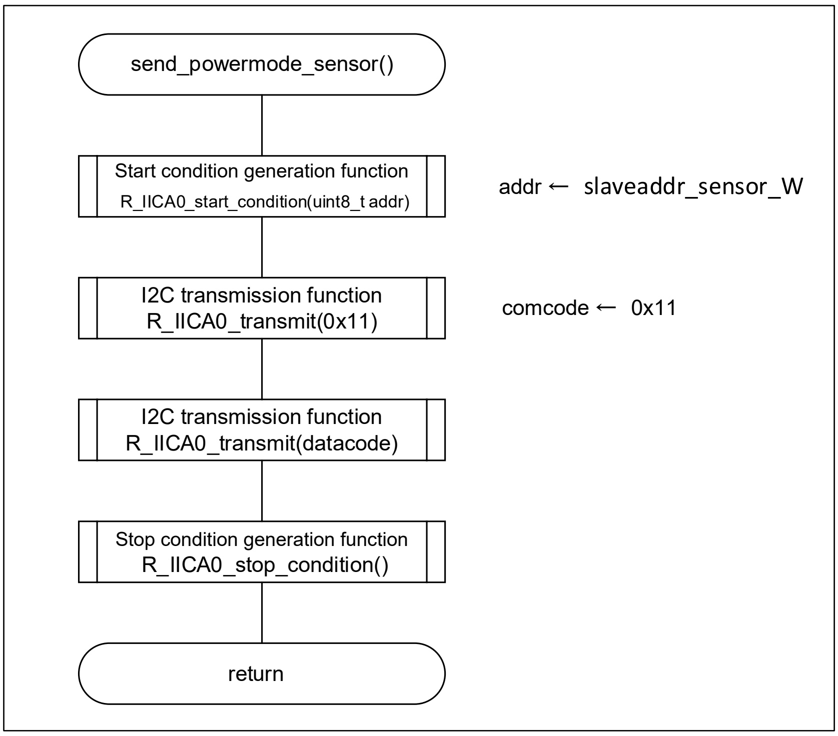
**Figure 4-12 Function for setting the output rate for the gyro sensor**



### 4.6.11 Function for setting the power mode for the gyro sensor

Figure 4-13 show the flowchart for Function for Function for setting the power mode for the gyro sensor.

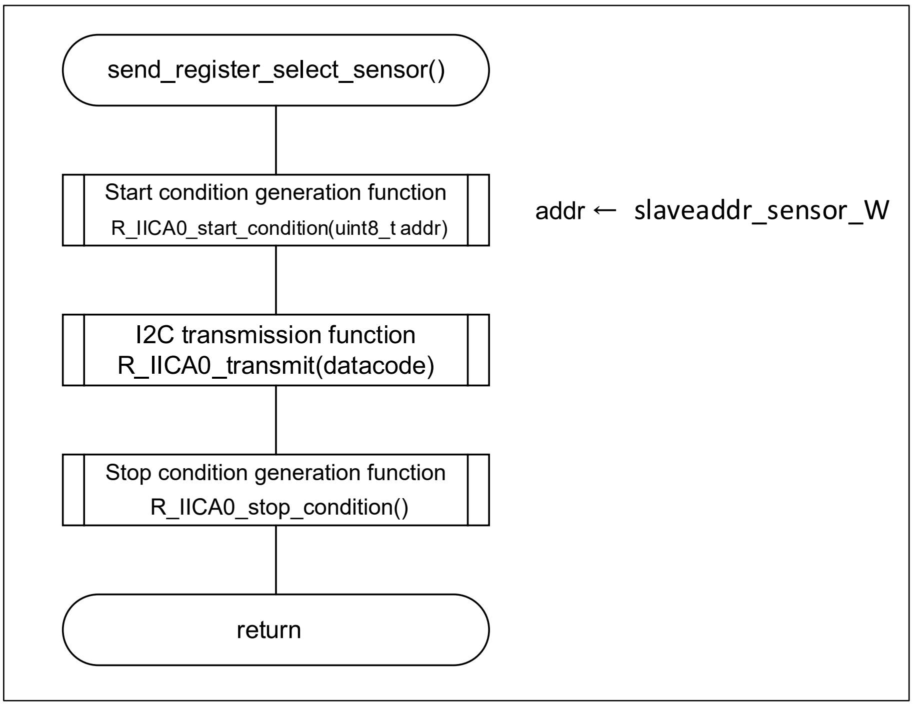
**Figure 4-13 Function for setting the power mode for the gyro sensor**




### 4.6.12 Function for setting registers in the gyro sensor

Figure 4-14 show the flowchart for Function for Function for setting the power mode for the gyro sensor.

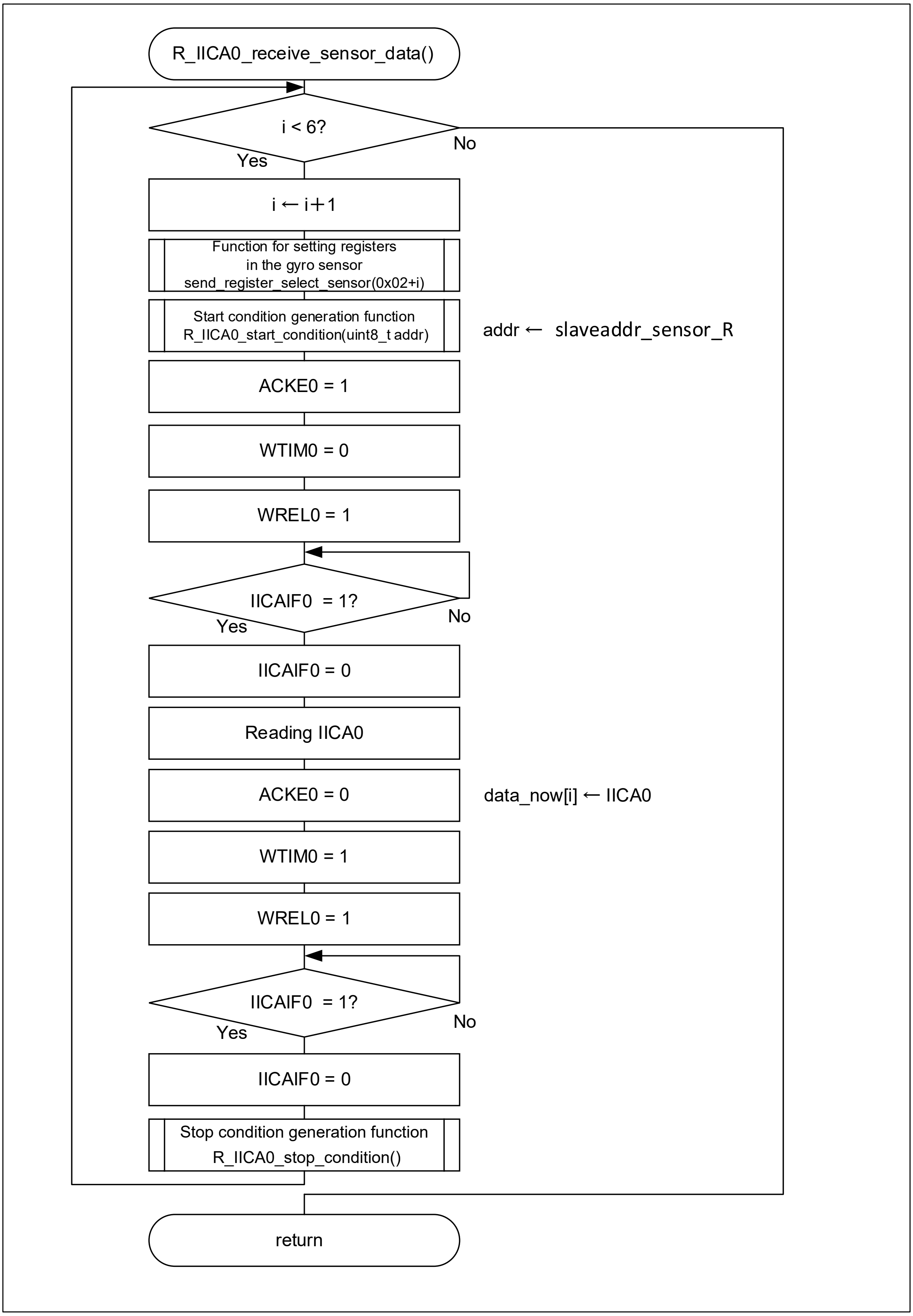
**Figure 4-14 Function for setting registers in the gyro sensor**



### 4.6.13 I2C receive function

Figure 4-15 show the flowchart for I2C receive function.

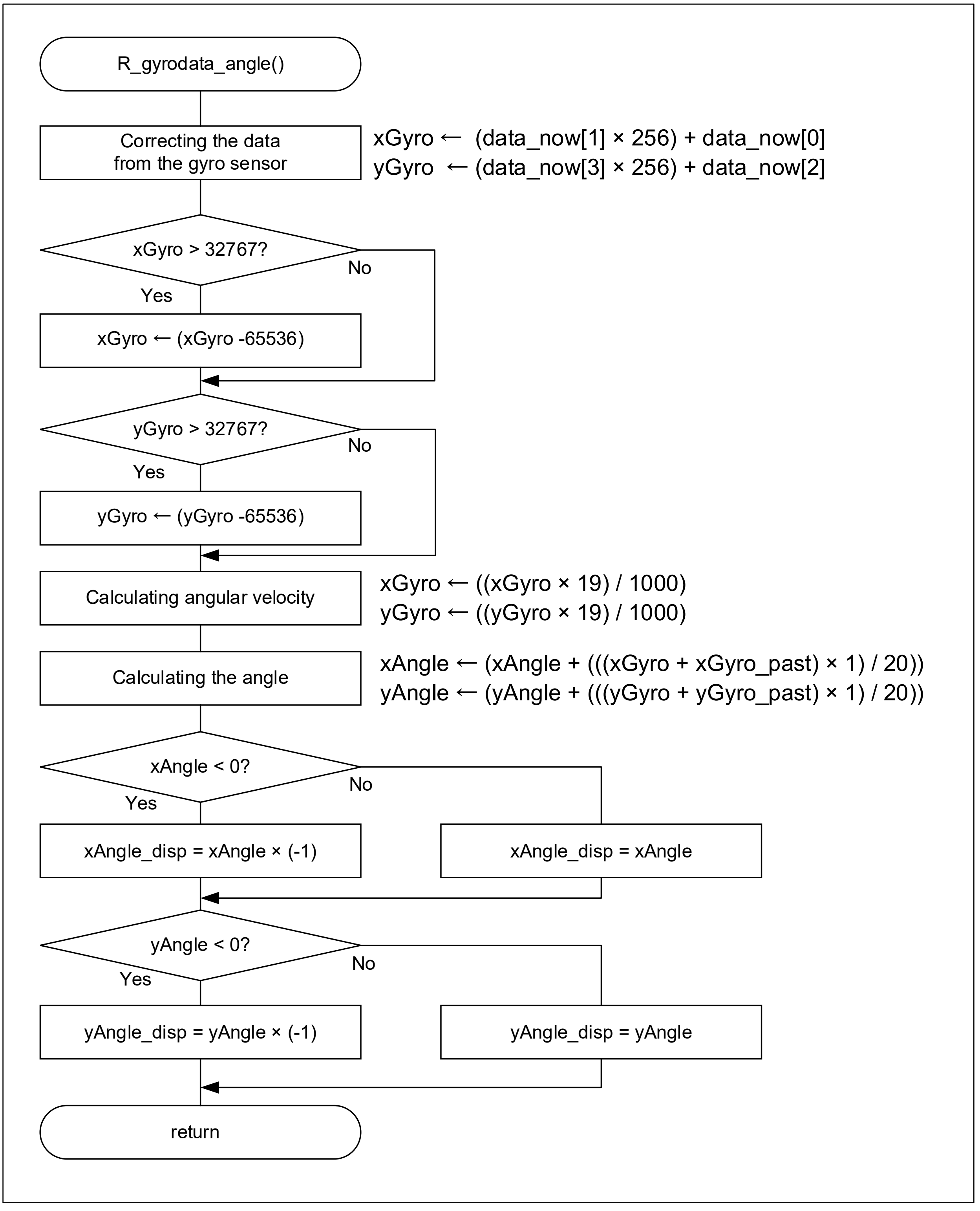
**Figure 4-15 I2C receive function**



### 4.6.14 Angle calculation function

Figure 4-16 show the flowchart for Angle calculation function.

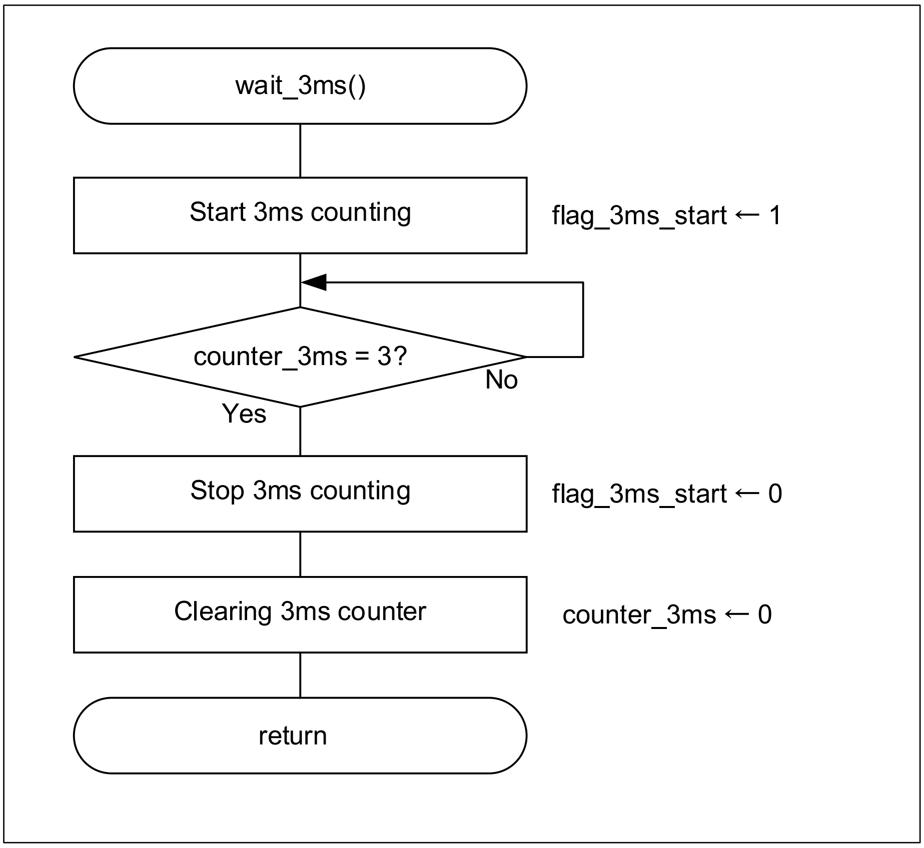
**Figure 4-16 Angle calculation function**



### 4.6.15 3ms delay function

Figure 4-17 show the flowchart for 3ms delay function.

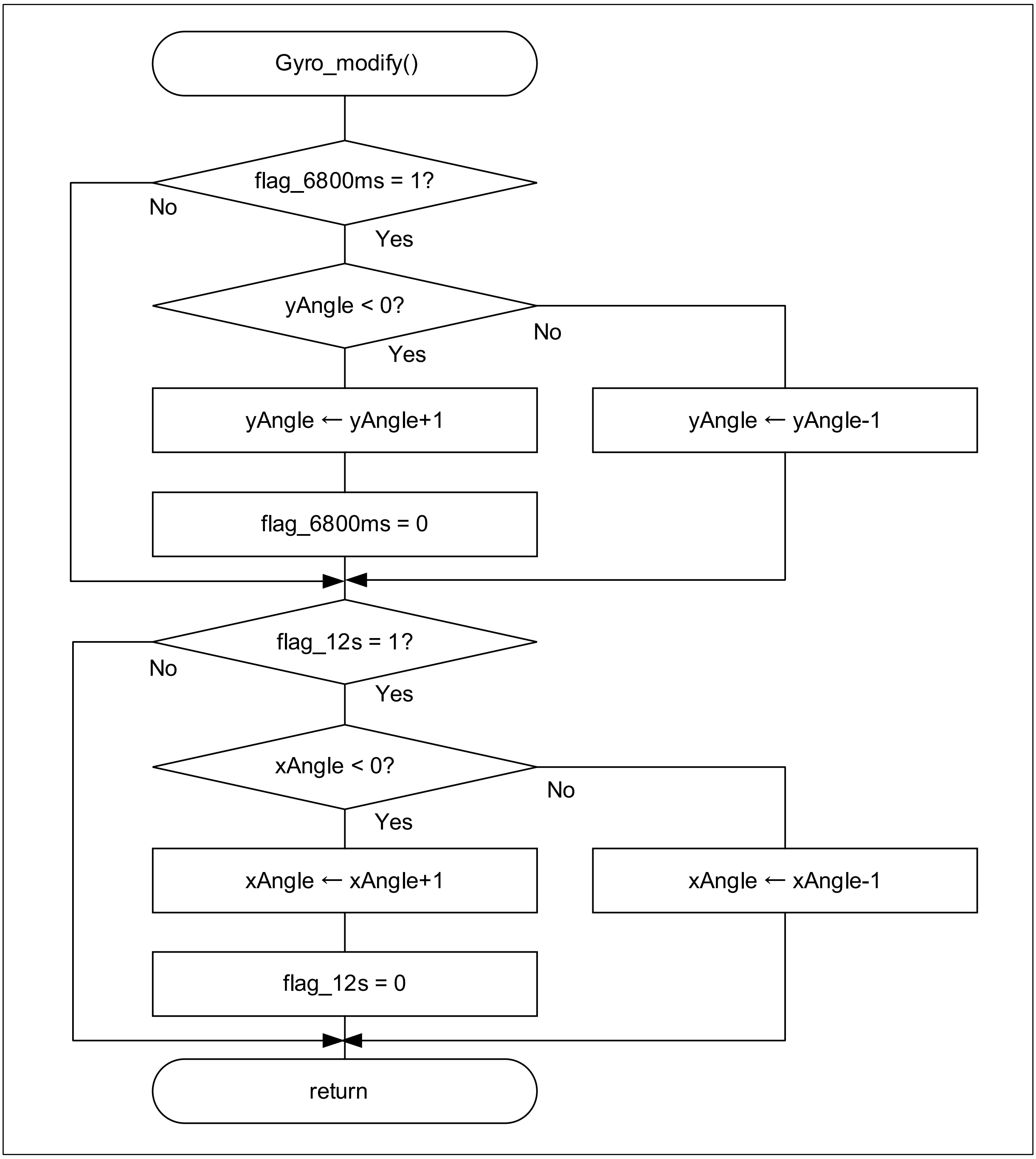
**Figure 4-17 3ms delay function**



### 4.6.16 Gyro drift correction function

Figure 4-18 show the flowchart for Gyro drift correction function.

**Figure 4-18 Gyro drift correction function**

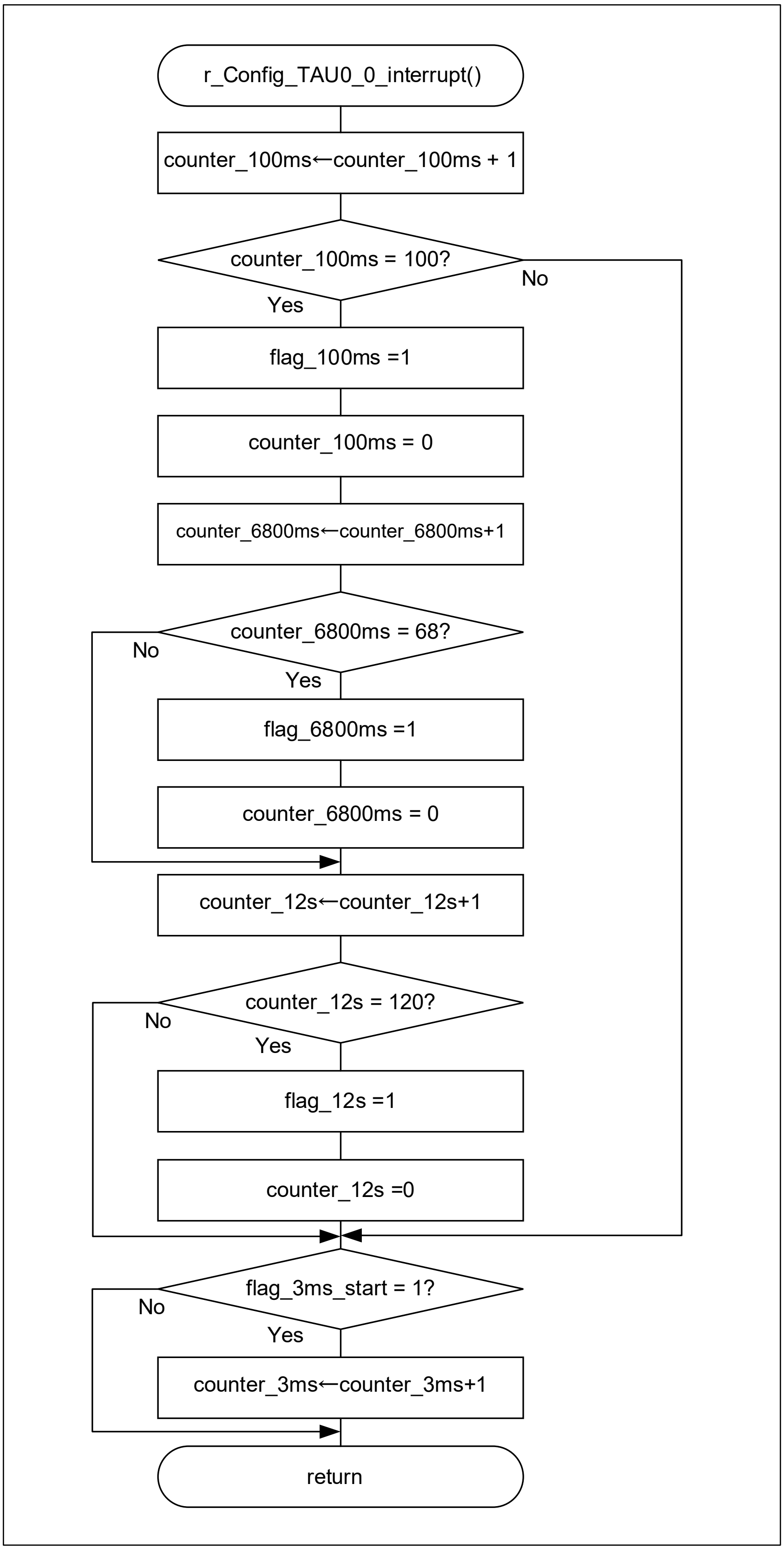



Note. Gyro drift characteristics may vary between individual sensors, so please perform calibration according to the gyro drift of your specific sensor.

### 4.6.17 1ms interval timer interrupt processing function

Figure 4-19 show the flowchart for 1ms interval timer interrupt processing function.

**Figure 4-19 1ms interval timer interrupt processing function**



## 5. Sample Code

Please obtain the sample code from the Renesas Electronics website.

## 6. Documents for Reference

User's Manual:

RL78/G15 User's Manual: Hardware ( R01UH0959EJ )

RL78 Family User's Manual: Software( R01US0015JJ )

The latest version can be downloaded from the Renesas Electronics website.

Technical Updates/Technical News

The latest information can be downloaded from the Renesas Electronics website.

## Revision History

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | | — | First edition issued |
| | | | |

# 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.

## Corporate Headquarters


TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com


## 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/.

## Trademarks