RL78 Family

RL78 I2C Multimaster

R01AN4411EU0100
Rev.1.00
August 31, 2018

Introduction

This application note describes the RL78 serial interface driver for IICA in the multi-master mode.

Target Device

The following is a list of devices that are currently supported by this API:

RL78/G13: R5F100LE

Contents

1. Introduction

The RL78 MCU family, IICA interface supports multimaster communication. Its arbitration-loss detection function makes multimaster communication possible in the I2C network. This application note describes the multimaster driver developed for RL78/G13 platform.

Note: I2C-BUS is a registered trademark of Royal Philips Electronics of the Netherlands.

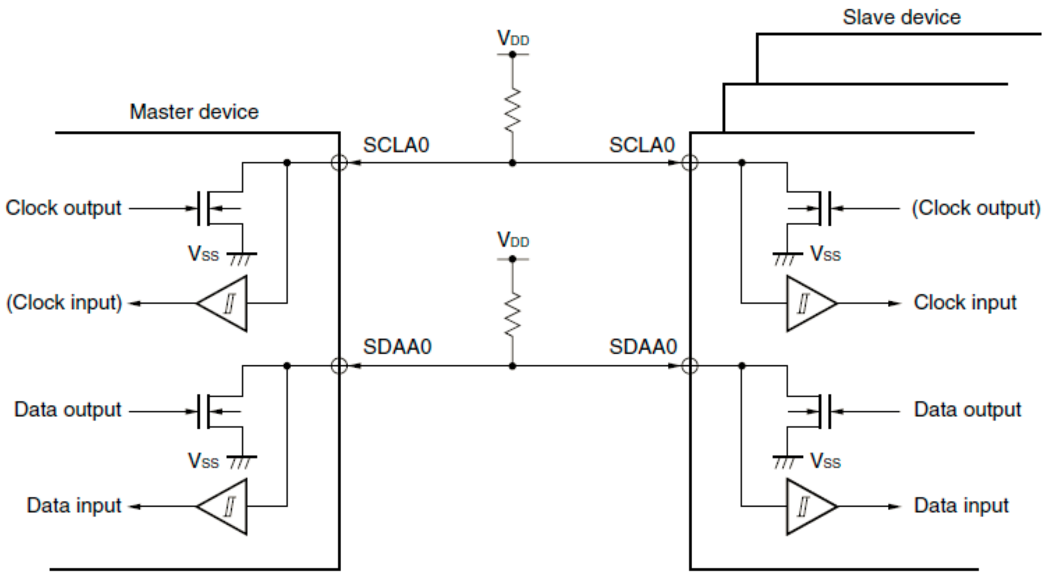
2. Overview

2.1 Hardware Interface

The I2C Multi-master driver interfaces with the IICA0 hardware level driver. For the correct data and clock lines/pins for your microcontroller please refer to the hardware manual.

It is important to note, to avoid floating grounds you must also connect all devices to the same ground line.

Since outputs from the serial clock line and the serial data bus line are N-ch open-drain outputs, an external pull-up resistor is required on both SCLA0 and SDAA0 line.



2.2 Timeout

The timer will begin countdown when you initiate either a master read or master write event. If the event does not complete (and issue a callback to the application level) before the timeout occurs then the driver will reset completely as if the Open command had just been called. Please note that this driver uses the 7th channel of timer 0, it is not possible to change this without directly modifying the driver source code.

3. Driver Control and interface

3.1 Configuration

Configurable options can be found in r_i2c_multimaster_cfg.h. To configure the driver, modify this file according to this section, referring to the hardware manual when necessary.

3.1.1 Interrupt priority

I2C_INT_PRIORITY is a configurable option for setting the interrupt priority of the I2C driver interrupt. This value ranges from 0 (highest) to 3 (lowest)

3.1.2 Main Clock Frequency

MAIN_CLKFREQ specifies the main clock frequency. The driver uses this to calculate bit rate.

3.1.3 Timeout

The configuration file provides a few example timeout values (100ms, 500ms, and 1000ms) which are used to initialize the timer. Select a value by defining R_I2C_USE_TIMEOUT_xxxx. The default for this is the 1000ms.

The driver allows you to define any value for a clock timer. If you wish to define a custom timeout value you must define the CK divisors as well as the counter value. To find the values you will need please reference the hardware manuals section on the Timer Array Unit. Once you have determined the divisors and counter value needed, set the CKxx_DIVISOR and TIMER_COUNTER_VALUE for your custom timeout configuration.

3.2 API Structures

3.2.1 i2c_baud_t

The baud selection allows for the selection of the I2C bitrate. The speed selections are either 100kbps, 400kbps, or 1000kbps. Each bitrate has a corresponding minimum requirement of the main clock to run effectively. The requirements are as follows

- I2C_BAUD_100K
  - minimum main clock value is 1MHz
- I2C_BAUD_400K
  - minimum main clock value is 3.5MHz
- I2C_BAUD_1000K
  - minimum main clock value is 10MHz

3.2.2 i2c_slave_callback_events_t

All possible events for the I2C slave callback.

- I2C_CALLBACK_EVENT_SLAVE_ADDRESS_RECEIVED
  - the slave address of this device has been received
- I2C_CALLBACK_EVENT_SLAVE_BYTE_RECEIVED

  - a byte of data has been received
- I2C_CALLBACK_EVENT_SLAVE_TRANSMIT_REQUEST
  - a request to transmit data has been received
- I2C_CALLBACK_EVENT_SLAVE_STOP_RECEIVED
  - a stop bit has been detected

3.2.3 i2c_master_callback_events_t

All possible events for the I2C slave callback.

- I2C_CALLBACK_EVENT_MASTER_WRITE_COMPLETE
  - master write completed
- I2C_CALLBACK_EVENT_MASTER_READ_COMPLETE
  - master read completed
- I2C_CALLBACK_EVENT_MASTER_TIMEOUT
  - the last command has timed out
- I2C_CALLBACK_EVENT_MASTER_ARBITRATION_LOST
  - arbitration has been lost
- I2C_CALLBACK_EVENT_MASTER_EXIT_CODE_RECEIVED
  - Exit code received
- I2C_CALLBACK_EVENT_MASTER_NAK_RECEIVED
  - NAK received

3.2.4 i2c_err_t

All possible error messages.

- I2C_SUCCESS
  - No error occurred
- I2C_ERR_ILL_PARAMETER
  - argument contained an illegal value
- I2C_ERR_NOT_INITIALIZED,
  - operation attempted without initialization
- I2C_ERR_ALREADY_INITIALIZED,
  - driver already initialized
- I2C_ERR_BUSY
  - another operation is still in progress
- I2C_ERR_ILLEGAL_MAINCLOCK
  - main clock cannot support baud rate selected
- I2C_ERR_LOST_ARBITRATION
  - arbitration was lost
- I2C_ERR_FAILED_START

    - arbitration resulted in neither master nor slave operation
- I2C_ERR_NAK
    - NAK was received
- I2C_ERR_RECEIVED_EXITCODE
    - Exit code was received

3.2.5 i2c_command_t

Defines the commands for the Control function.

- I2C_CHANGE_SLAVE_ADDRESS
    - allows the driver to modify its slave address at any given time
    - the data argument is the new address (1 byte)
- I2C_SET_SLAVE_ACK
    - Configure this device to ACK when another master on the line reads or writes to this address
    - data is not used in this instance
- I2C_SET_SLAVE_NAK
    - Configure this device to NAK when another master on the line reads or writes to this address
    - data is not used in this instance

3.2.6 i2c_config_t

Elements needed for the Open command.

- i2c_baud_t baud;
    - selects the baud rate
- uint8_t slave_address;
    - the address this device responds to
- void (*slave_callback)(i2c_slave_callback_events_t event, uint8_t *p_data);
    - pointer to the slave callback function
- void (*master_callback)(i2c_master_callback_events_t event, uint8_t *p_data);
    - pointer to the master callback function

3.3 API Functions

3.3.1 Open

```
i2c_err_t R_I2C_Multimaster_Open(i2c_config_t *param);
```

Initializes performs all functions necessary to run the I2C driver. Including initializing the timer driver.

Parameters:

- i2c_config_t *param
    - Structure for all parameter values, please review section 3.2.6i2c_config_t for information regarding this data structure

Error codes:

- I2C_SUCCESS
- I2C_ERR_ALREADY_INITIALIZED
- I2C_ERR_ILL_PARAMETER
- I2C_ERR_ILLEGAL_MAINCLOCK

Example:

```
    i2c_err_t err;

    param.baud = I2C_BAUD_100K;
    param.slave_callback = &i2c_slv_callback;
    param.master_callback = &i2c_mst_callback;
    param.slave_address = myAddress;


    err = R_I2C_Multimaster_Open(&param);
```

3.3.2 Master Write

```
i2c_err_t R_I2C_Multimaster_Master_Write(uint8_t addr,
                uint8_t *p_buffer,
                uint16_t size);
```

This function transmits “size” bytes of data from “p_buffer” to the device located at “addr”. **Warning**: The data located at “p_buffer” should not be modified until transmission has completed.

Parameters:

- uint8_t addr
  - Address to send message to
- uint8_t *p_buffer
  - Pointer to a buffer containing the message to be sent
- uint16_t size
  - Size of the message to be sent

Error codes:

- I2C_SUCCESS
- I2C_ERR_NOT_INITIALIZED
- I2C_ERR_BUSY
- I2C_ERR_ILL_PARAMETER
- I2C_ERR_FAILED_START

Example:

```
    i2c_err_t err;

    uint8_t remoteAddress = DEVICE_ADDRESS;
    uint8_t txbuf[MSG_SIZE] = {'h', 'e', 'l', 'l', 'o', 0};

    err = R_I2C_Multimaster_Master_Write(remoteAddress, txbuf, MSG_SIZE);
```

3.3.3 Master Read

```
i2c_err_t R_I2C_Multimaster_Master_Read(uint8_t addr,
                  uint8_t *p_buffer,
                  uint16_t size);
```

This function stores “size” bytes of data from “addr” into the buffer pointed to by “p_buffer”. Warning: Contents of the buffer are not considered to be valid until a receive-complete event occurs.

Parameters:

- uint8_t addr
  - Address to retrieve data from
- uint8_t *p_buffer
  - Pointer to a buffer for the data to be stored in
- uint16_t size
  - Size of the message to be retrieved

Error codes:

- I2C_SUCCESS
- I2C_ERR_NOT_INITIALIZED
- I2C_ERR_BUSY
- I2C_ERR_ILL_PARAMETER
- I2C_ERR_FAILED_START

Example:

```
i2c_err_t err;

uint8_t remoteAddress = DEVICE_ADDRESS;
uint8_t rxbuf[MSG_SIZE];

err = R_I2C_Multimaster_Master_Read(remoteAddress, rxbuf, MSG_SIZE);
```

3.3.4 Control

```
i2c_err_t R_I2C_Multimaster_Control(i2c_command_t function, void *data);
```

This function is used for special driver operations at runtime. The data parameters are discussed in more detail in Section 3.2.5.

Parameters:

- i2c_command_t function
  - Function to be performed by the control method
- void *data
  - Parameter data to be used by the command function
  - Please refer to 3.2.5i2c_command_t for information on the format for each individual command

Error codes:

- I2C_SUCCESS
- I2C_ERR_NOT_INITIALIZED
- I2C_ERR_ILL_PARAMETER

Examples:

```
//Change address example
uint8_t address = 0x80;
R_I2C_Multimaster_Control(I2C_CHANGE_SLAVE_ADDRESS, (void *) &address);


//Set ACK example
R_I2C_Multimaster_Control(I2C_SET_SLAVE_ACK, NULL);


//Set NAK example
R_I2C_Multimaster_Control(I2C_SET_SLAVE_NAK, NULL);
```

3.3.5 Close

```
i2c_err_t R_I2C_Multimaster_Close(void);
```

Parameters:

- None

Error codes:

- I2C_SUCCESS
- I2C_ERR_NOT_INITIALIZED

Closes operation of the driver. To use the driver again it must re-opened.

3.4 Callbacks

3.4.1 Slave Callback

The slave callback function is used to notify the application layer of an event which has occurred on the slave side of the Multimaster protocol so that special processing (if desired) may be done.

Example:

```
void i2c_slv_callback(i2c_slave_callback_events_t event, uint8_t *p_data)
{
    switch (event)
    {
        case I2C_CALLBACK_EVENT_SLAVE_ADDRESS_RECEIVED:
            //Notifies the Application layer that
            //our address has been received on the I2C bus
            break;
        case I2C_CALLBACK_EVENT_SLAVE_BYTE_RECEIVED:
            //Notifies the Application layer that
            //we have received a byte of data
            //p_data will hold the value of the byte received (Only 1 byte)
            break;
        case I2C_CALLBACK_EVENT_SLAVE_TRANSMIT_REQUEST:
            //Notifies the Application layer that
            //there is a request to transmit the next byte
            //pass into p_data the value to send to the master (Only 1 byte)
            break;
        case I2C_CALLBACK_EVENT_SLAVE_STOP_RECEIVED:
            //Notifies the Application layer that
            //there has been a stop signal on the bus
            break;
    }
}
```

3.4.2 Master Callback

The master callback function is used to notify the application layer of an event which has occurred on the master side of the Multimaster protocol.

Example:

```
void i2c_mst_callback(i2c_master_callback_events_t event, uint8_t *p_data)
{
    switch (event)
    {
    case I2C_CALLBACK_EVENT_MASTER_WRITE_COMPLETE:
        //Notifies the Application layer that
        //the write request has been completed
        break;
    case I2C_CALLBACK_EVENT_MASTER_READ_COMPLETE:
        //Notifies the Application layer that
        //the read request has been completed
        break;
    case I2C_CALLBACK_EVENT_MASTER_TIMEOUT:
        //Notifies the Application layer that
        //a timeout has occurred
        break;
    case I2C_CALLBACK_EVENT_MASTER_ARBITRATION_LOST:
        //Notifies the Application layer that
        //arbitration has been lost, retry last operation
        break;
    case I2C_CALLBACK_EVENT_MASTER_EXIT_CODE_RECIEVED:
        //Notifies the Application layer that
        //Exit code received, retry last operation
        break;
    case I2C_CALLBACK_EVENT_MASTER_NAK_RECEIVED:
        //Notifies the Application layer that
        //a NAK was received on the line
        break;
    }
}
```

Website and Support

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

Inquiries
http://www.renesas.com/contact/

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	08/30/2018		Initial Release

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. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

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 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.
5. 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.
6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

RENESAS

SALES OFFICES http://www.renesas.com

Renesas Electronics Corporation

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics Corporation
TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

Renesas Electronics America Inc.
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 7.2