RL78/F13, F14 Group

LIN Master Mode (RLIN3)

R01AN1841ED0101
Rev.1.01
May 29, 2015

Introduction

This document describes how to use the RLIN3 hardware in master mode.

Target Device

RL78 F13/F14 Group (R5F10PPJ)

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

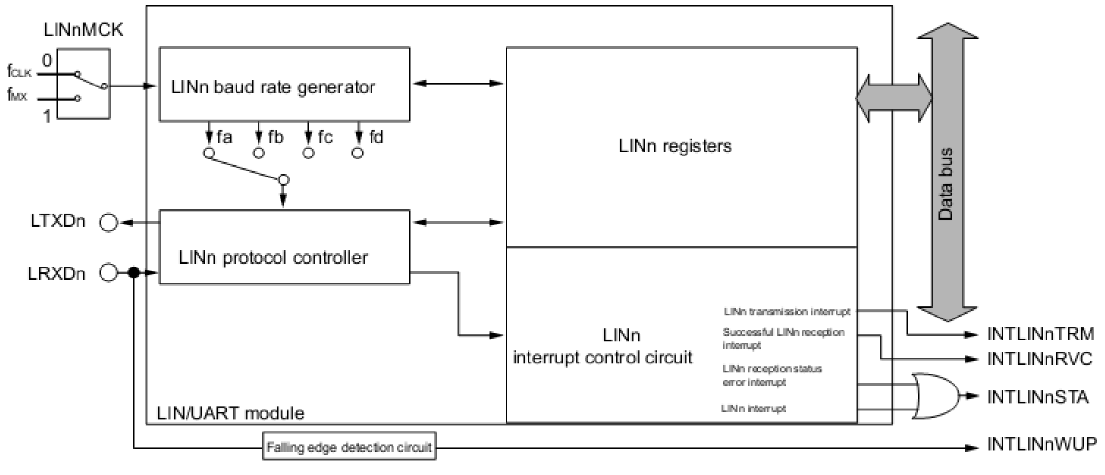
Development environment

IAR Embedded workbench for Renesas RL78 V1.30.3

Contents

1. RLIN3 hardware module specifications

The RLIN3 interface is a dedicated UART inface supporting LIN slave and master functionality.



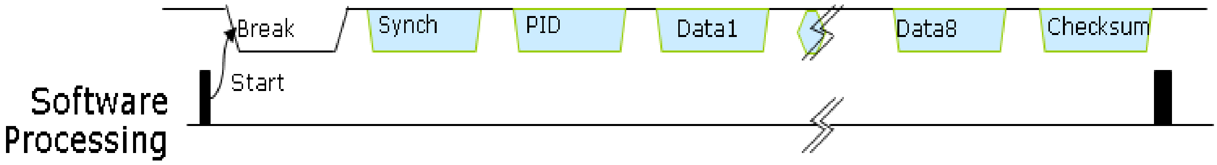
n = 0, 1

Figue1.1 RLIN3 Module Block Diagram

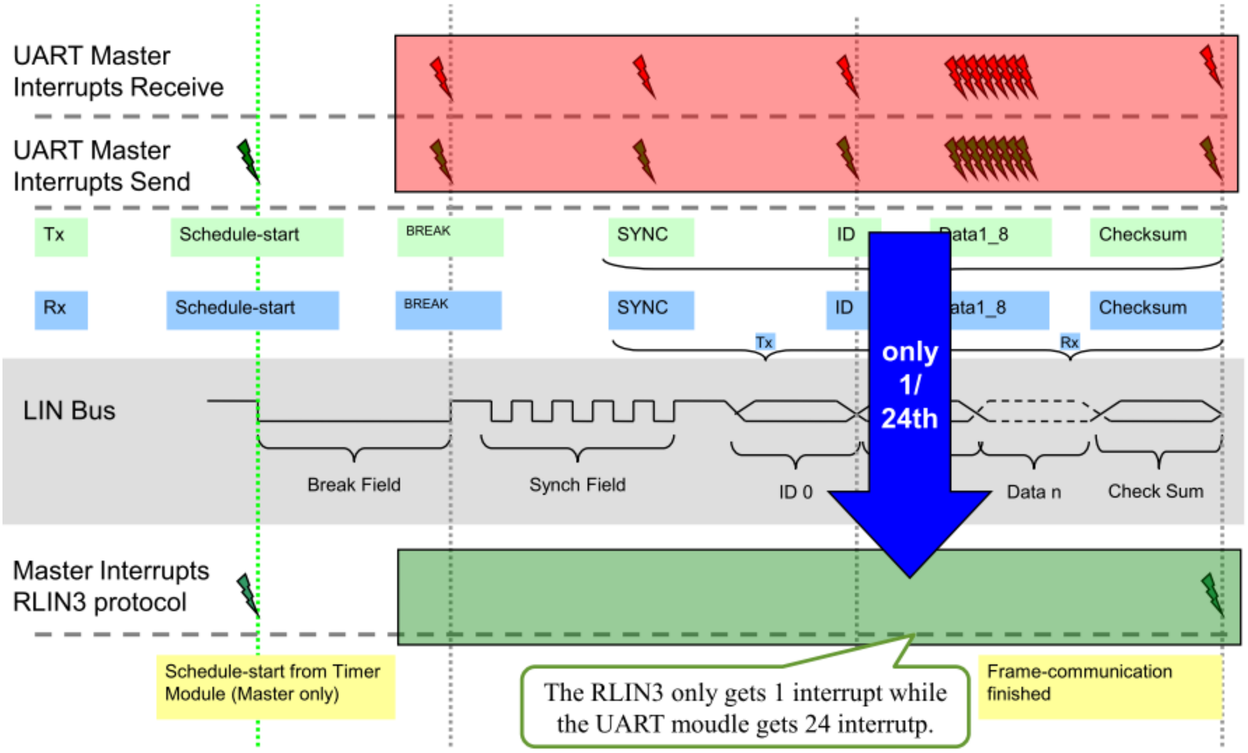
Features of the RLIN3 interface

- LIN Master mode support by extended hardware features
  - Inheritance of functionality from RLIN2 IP
  - Conform to LIN Specification Package Revision 1.3, 2.0, 2.1 and SAEJ2602
  - Frame combined operation (header + response processing in 1 step)
  - Frame separate operation (separate steps for header and response)
  - Wakeup transmission and reception
  - Automatic classic or enhanced checksum generation/verification
  - Automatic error detection
  - Automatic frame communication
    - Automatic Header transmission
    - Automatic response transmission/reception
- Advanced features for LIN
  - Extended response reception and transmission (extension to any data count by software)
  - LIN Wake Up mode
  - LIN Self-Test mode
  - Various settings for LIN frame timing (spacing, break/delimiter timing)
  - LIN Error detections
    - Bit errors (commonly, or in break/wakeup field [“physical bit error”])
    - Frame error (wrong STOP bit level)
    - Checksum error (received does not match internally calculated)
    - Timeout error (either frame or response, threshold automatically set)
    - Response preparation error (for LIN master – on response if not yet triggered)

- Software processing flow
  - During a complete LIN message only one interrupts are generated. The interrupt are generated when complete message response.



  - Due to this enhanced LIN functionality the interrupt load will be drastically reduced compared to a standard UART.


UART Master Interrupts Receive
UART Master Interrupts Send
Tx
Schedule-start
BREAK
SYNC
ID
Data1_8
Checksum
Rx
Schedule-start
BREAK
SYNC
ID
Data1_8
Checksum
Tx
Rx
only 1/ 24th
LIN Bus
Break Field
Synch Field
ID 0
Data n
Check Sum
Master Interrupts RLIN3 protocol
Schedule-start from Timer Module (Master only)
The RLIN3 only gets 1 interrupt while the UART moudle gets 24 interrupt.
Frame-communication finished

2. Development environment

The sample code described in this application note runs under the conditions below.

Table 2.1 Development environment

Item	Contents
MCU	RL78/F14 R5F10PPJ (WS2.0)
Operating frequencies	Xin : 4MHz System clock: 32MHz (PLL) CPU clock: 32MHz
Operating voltage	5.0V for MCU, 12V for LIN transceiver
Integrated development environment	IAR Embedded workbench for Renesas RL78 V1.30.3
LIN protocol versions	V2.1
Evaluation board	See figure 2.1 below

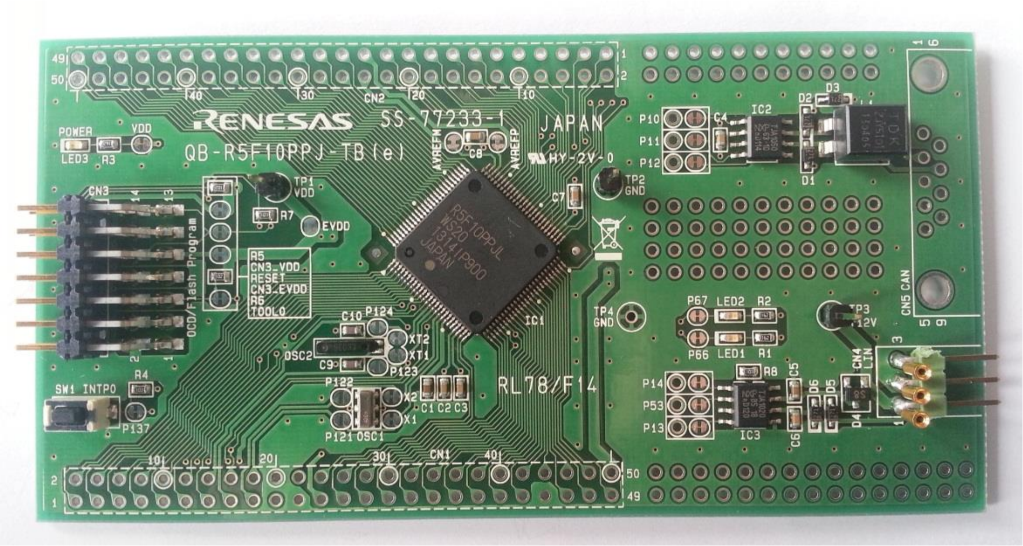


Figure 2.1 Evaluation board

3. Software

The sample code demonstrates the usage of the RLIN3 interface in LIN master mode. The sample code runs on the QB-R5F10PPJ-TB, which is a target board for the RL78F13,F14 microcontroller family including a LIN transceiver . In master mode, the RLIN3 waits for INTP0 falling edge, and it transmits a header frame to the LIN bus when the external interrupt generated. Otherwise, it prepares a response transmission or reception according to send IDs. LED1 and LED2 are showing that different IDs are contained in the header, and frames (headers and responses) were transmitted or received successfully.

3.1 Operation overview

Settings:

- Use channel of the RLIN3 to perform LIN communication in master mode.
- Use the P1.3/LTXD0 pin for the transmit data output.
- Use the P1.4/LRXD0 pin for the receive data input.
- Set the baud rate to 19200bps.
- Use the INTLIN0RVC interrupt; The INTLIN0RVC interrupt is generated after a LIN successful frame reception.
- Use the INTLIN0TRM interrupt; The INTLIN0TRM interrupt is generated after a LIN successful frame transmission, header transmission or wake up transmission.
- Use the INTLIN0 interrupt; TheINTLIN0 interrupt is generated when the LIN bus have any Error was detected. A complete error handling is not implemented.
- Communication direction and number of transmit/receive date at a response field are determined by the ID data received at the ID field.
- ID data store in the ID buffer register LIDB0.
- Load the data to the data buffer before starting the header transmission when master will transmit a response according to ID.

3.2 Functions

Function Name	Outline	Code size (bytes)
RLIN_Master_Init	Initial setting	98
RLIN_Master_HeaderTransmit	Header Transmit preparation	78
RLIN_Master_Transmit	Data transmission preparation	53
RLIN_Master_Receive	Data reception preparation	18
RLIN_Master_GetData	Get data from data buffer	18
Clear_DataBuffer	Setting all data buffer to 0	26
Get_response_RxData	Store data to variables array from Data buffer	41

Table 3.1 lists the Functions

3.3 Function Specifications

The following tables list the sample code function specifications

RLIN_Master_Init	
Outline	Initial setting of RLIN3's registers in master mode
Header	None
Declaration	void RLIN_Master_Init(void)
Description	Setting of channel, clock, baud rate, interrupts, header format.
Arguments	None
Returned value	None

Table 3.2 RLIN_Master_Init

RLIN_Master_HeaderTransmit		
Outline	Header transmit preparation	
Header	None	
Declaration	void RLIN_Master_HeaderTransmit(uint8_t ID)	
Description	Set RLIN3 to master mode, prepare response, setting header transmit start	
Arguments	uint8_t ID	Setting to ID buffer
Returned value	None	

Table 3.3 RLIN_Master_HeaderTransmit

RLIN_Master_Transmit		
Outline	Data transmission preparation	
Header	None	
Declaration	void RLIN_Master_Transmit(uint8_t * databuf, uint8_t data_length)	
Description	Setting data buffer and response transmission start, it was called by RLIN_Master_HeaderTransmit.	
Arguments	uint8_t * databuf	Transmit data
	uint8_t data_length	Transmission data length
Returned value	None	

Table 3.4 RLIN_Master_Transmit

RLIN_Master_Receive		
Outline	Data reception preparation	
Header	None	
Declaration	void RLIN_Master_Receive(uint8_t data_length)	
Description	Clear data buffer, setting reception format, response reception start	
Arguments	Uint8_t data_length	Receive data length
Returned value	None	

Table 3.5 RLIN_Master_Receive

RLIN_Master_GetData	
Outline	Get data from data buffer
Header	None
Declaration	void RLIN_Master_GetData(void)
Description	This function is get data from data buffer according to ID during the response reception
Arguments	None
Returned value	None

Table 3.6 RLIN_Master_GetData

Clear_DataBuffer	
Outline	Clear all data buffer to 0
Header	None
Declaration	void Clear_DataBuffer(void)
Description	Clear the complete data buffer
Arguments	None
Returned value	None

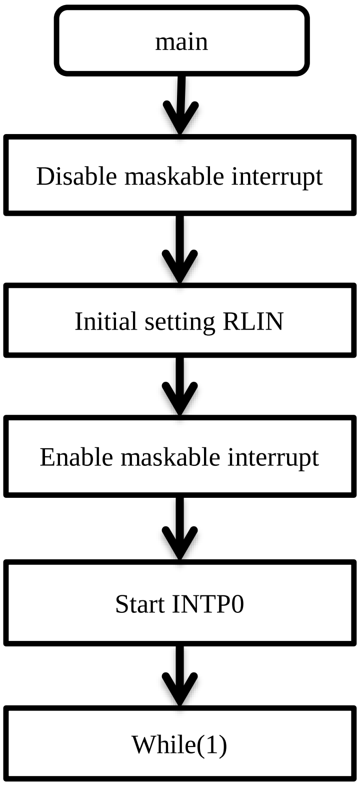
Table 3.7 Clear_DataBuffer

Get_response_RxData		
Outline	Store data to variable array from ID buffer	
Header	None	
Declaration	uint8_t Get_response_RxData(uint8_t * RxData)	
Description	Get reception data to variable array	
Arguments	Uint8_t * RxData	Data variable array
Returned value	RxData[1]	

Table 3.8 Get_response_RxData

3.4 Flowcharts

3.4.1 Main Flowchart

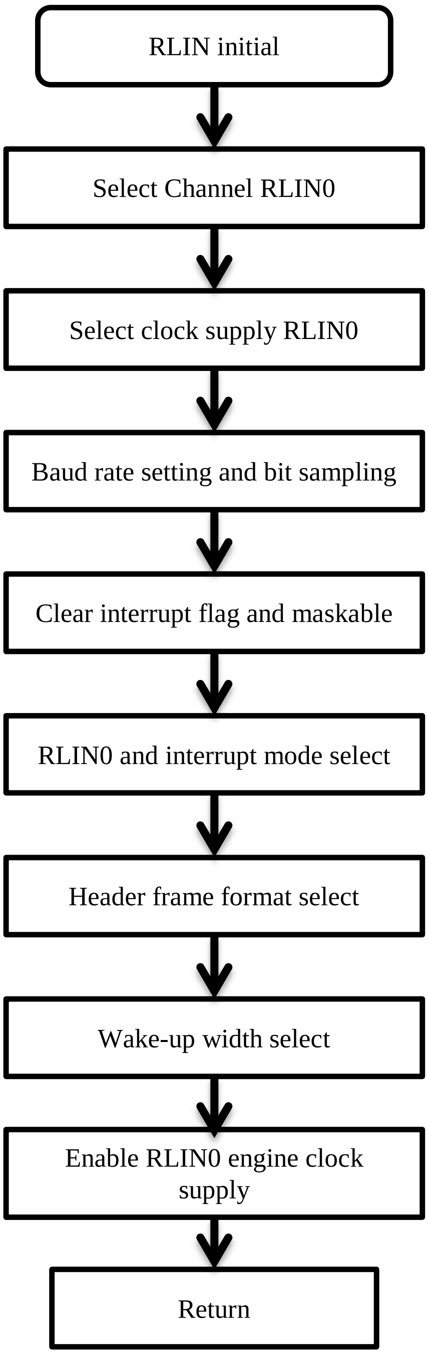


See figure 3.2 master initial flowcharts

Enable intp0 interrupt, wait external falling edge for transmit header frame.

Figure 3.5 show the main processing

3.4.2 Initial RLIN flowchart


RLIN initial
Select Channel RLIN0
Select clock supply RLIN0
Baud rate setting and bit sampling
Clear interrupt flag and maskable
RLIN0 and interrupt mode select
Header frame format select
Wake-up width select
Enable RLIN0 engine clock supply
Return

0x00 → LCHSEL selects RLIN0

Enable clock by PER2

Select Fclk=32MHz to RLIN0

0x01 → LWBR0, prescaler clock to 32MHz, 16 bit sampling.

1 0 →LMD[1:0] LIN master mode

LPRP00=0x67, fa=19230bps

LIOS=1; Transmission interrupt, successful reception interrupt, and reception status interrupt are used.

BLT=5, transmit break width 18 Tbit ; BDT=1, 1Tbit Delimiter.

0x11 → LSC0,setting 1Tbit for response space ,1Tbit for inter-Byte space

0x30 → LWUP0,wake up transmission low width is 4Tbit

LIN0MCKE=1, enable LIN engine clock supply.

Figure 3.1 show the RLIN initial processing

3.4.3 Master Header frame transmit flowchart

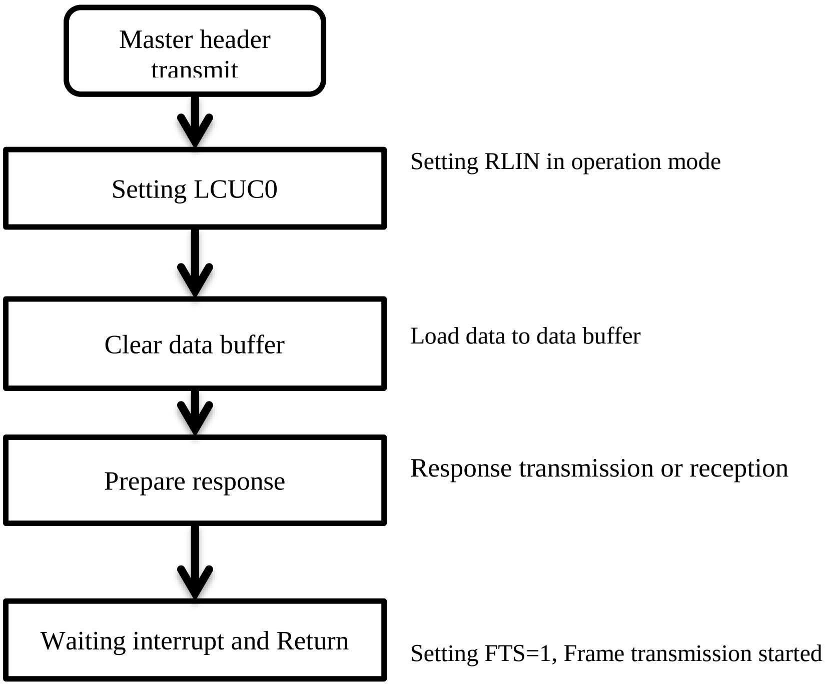

Master header transmit
Setting LCUC0
Clear data buffer
Prepare response
Waiting interrupt and Return
Setting RLIN in operation mode
Load data to data buffer
Response transmission or reception
Setting FTS=1, Frame transmission started

Figure 3.2 show the master transmit processing

3.4.4 Master transmit flowchart

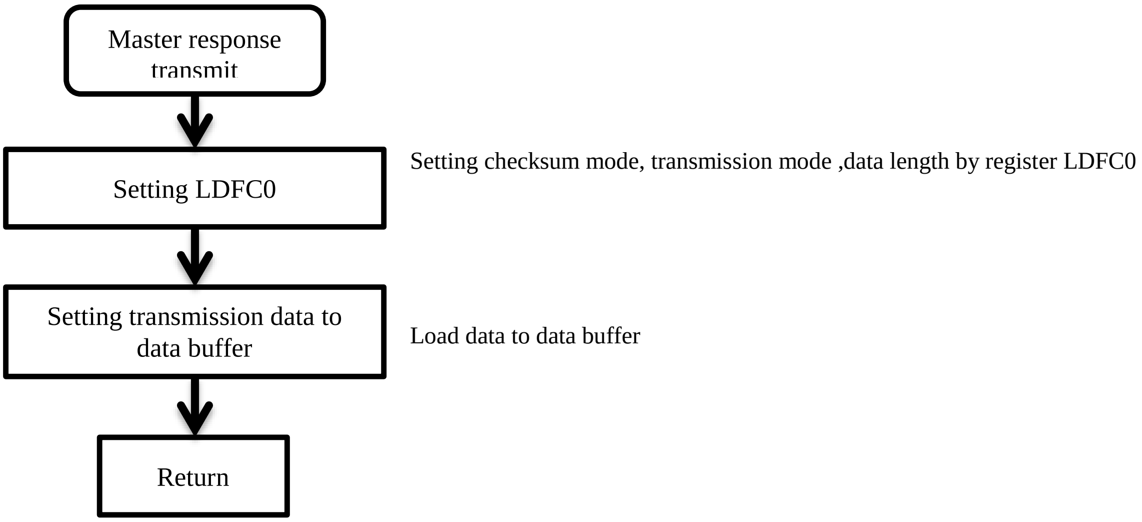

Master response transmit
Setting LDFC0
Setting transmission data to data buffer
Return
Setting checksum mode, transmission mode ,data length by register LDFC0
Load data to data buffer

Figure 3.3 show the master transmit processing

3.4.5 Master receive flowchart

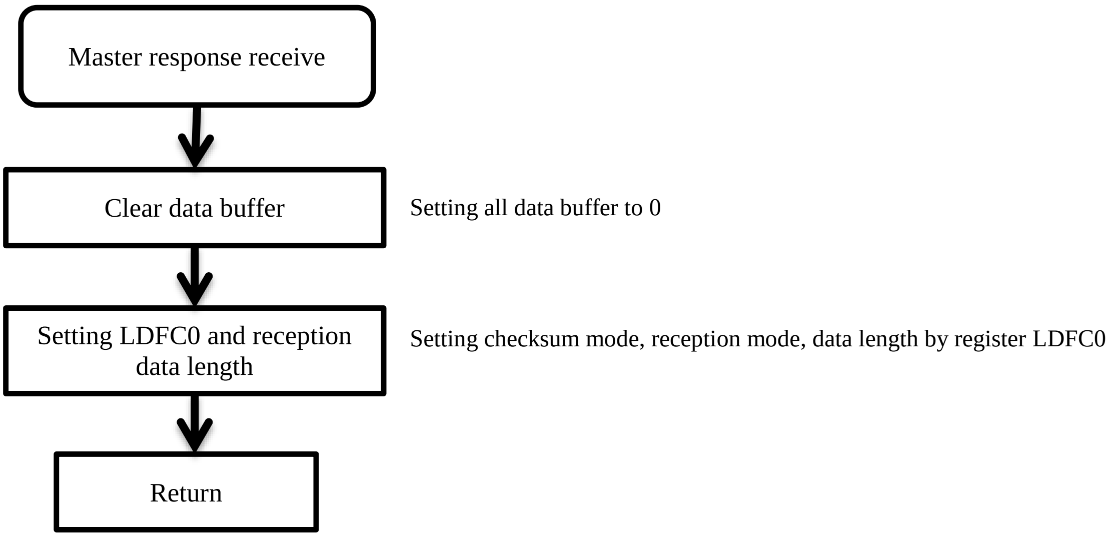

Master response receive
Clear data buffer
Setting all data buffer to 0
Setting LDFC0 and reception data length
Setting checksum mode, reception mode, data length by register LDFC0
Return

Figure 3.4 show the master receive processing

3.4.6 Master interrupt flowchart

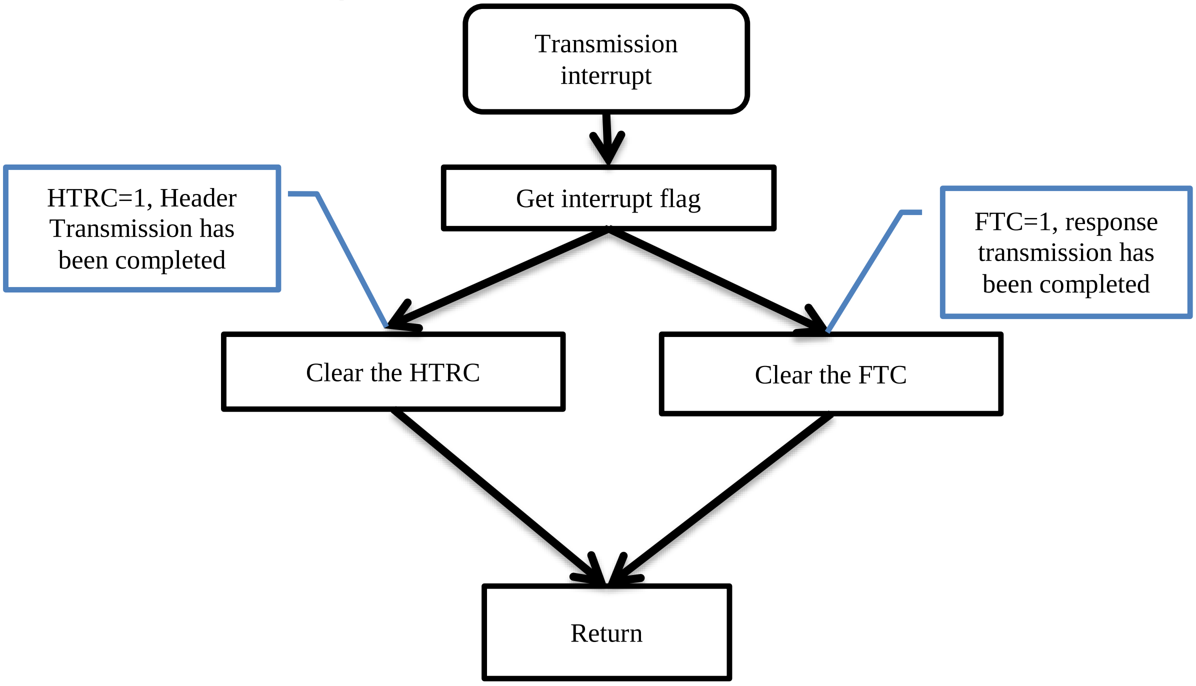

Transmission interrupt
Get interrupt flag
HTRC=1, Header Transmission has been completed
FTC=1, response transmission has been completed
Clear the HTRC
Clear the FTC
Return

Figure 3.5 show the transmit interrupt processing

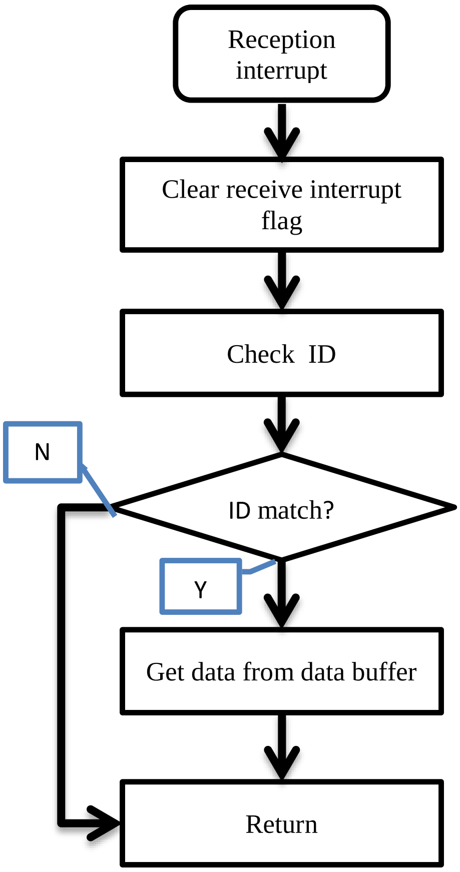


Figure 3.6 show the reception interrupt processing

4. Demo system

The below pictures shows the demo system consists out of two RL78 F14 target boards. One board is runningn in master mode and the second one in slave mode. The software from the master mode is part of this application note, where the slave mode is described in a separate document. Both boards are connected via the LIN interface. The master is indicating proper data communication via the two LEDs mounted on the target boards

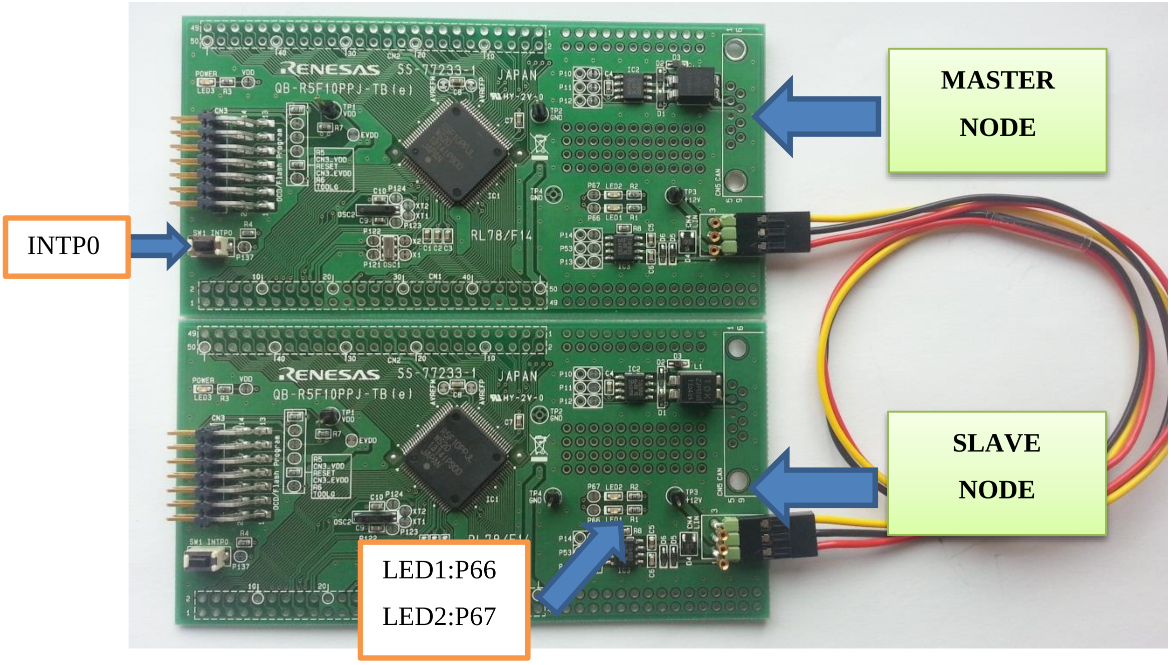
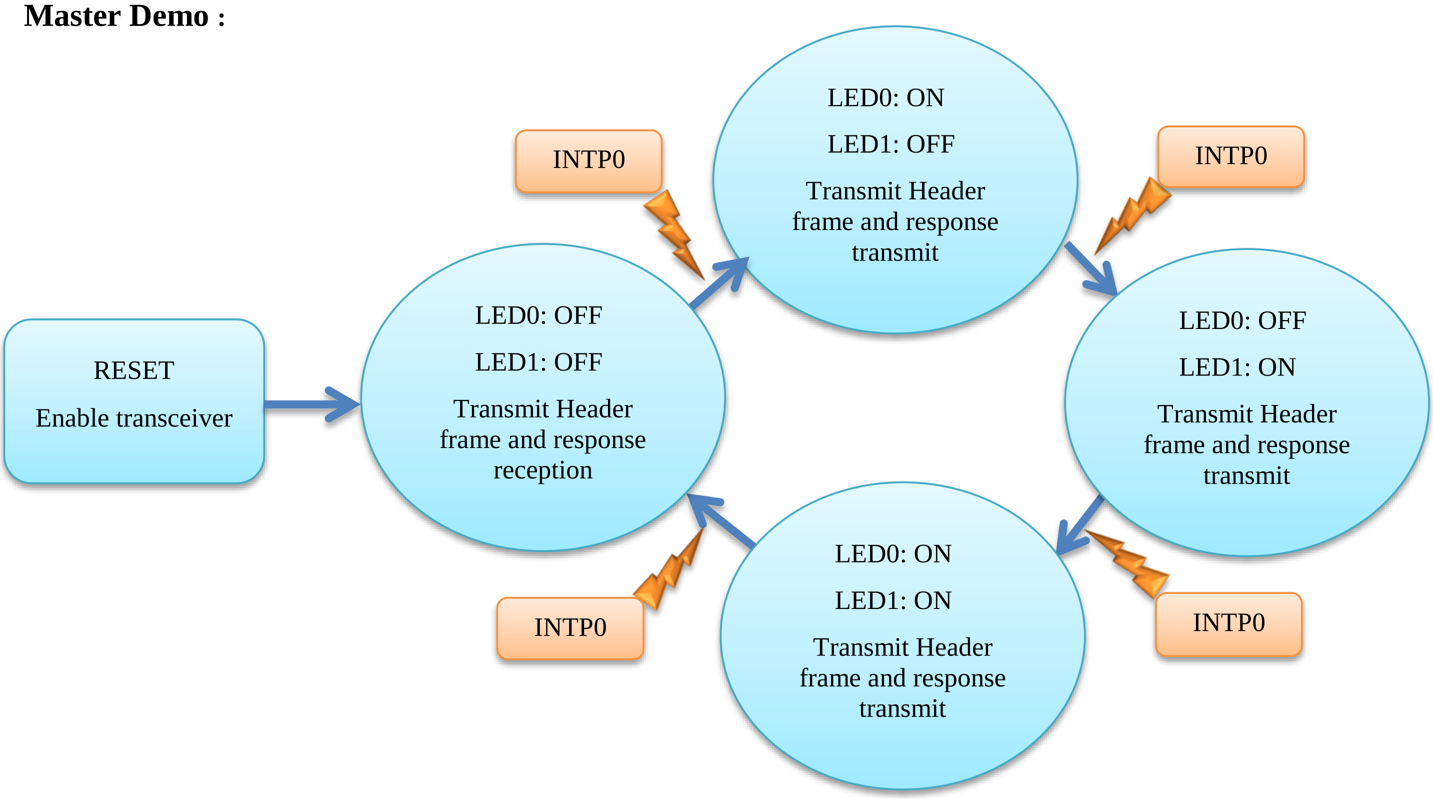


Figure 4.1 Picture of the demo system

In the below state diagram of the master you will find the different internal states of the master demo with the corresponding LED.

Master Demo :


RESET
Enable transceiver
LED0: OFF
LED1: OFF
Transmit Header frame and response reception
INTP0
LED0: ON
LED1: OFF
Transmit Header frame and response transmit
INTP0
LED0: OFF
LED1: ON
Transmit Header frame and response transmit
INTP0
LED0: ON
LED1: ON
Transmit Header frame and response transmit
INTP0

5. Sample code

5.1 RLIN_driver.c

```
/************************************************************************************************
* File Name    : RLIN_driver.c
* Device(s)    : R5F10PPJ
* Tool-Chain   : IAR Systems iccrl78
* Description  : This file implements device driver for PORT module.
* Creation Date: 15.07.2013
************************************************************************************************/
/************************************************************************************************
Includes
***********************************************************************************************/
#include "RLIN_macrodriver.h"
#include "RLIN_driver.h"
#include "RLIN_userdefine.h"

uint8_t Master_TxData1[]={0x08,0x80};   /*Transmission data store array*/
uint8_t Master_TxData2[]={0x49,0x40};   /*Transmission data store array*/
uint8_t Master_TxData3[]={0xCA,0x00};   /*Transmission data store array*/

uint8_t Master_RxData1[8];
uint8_t Master_RxData2[8];
uint8_t Master_RxData3[8];

/************************************************************************************************
* Function Name: RLIN_Master_Init(void)
* Description  : This function initializes the RLIN Slave node, setting clock supply,baud rate,ect.
* Arguments    : None
* Return Value : None
***********************************************************************************************/
void RLIN_Master_Init(void)
{
  LCHSEL = 0x00;         /*  Selects RLIN0 */
  PER2  |= 0x04;         /*  Enable input clock supply RLIN0*/
  LINCKSEL=0x00;         /*  selects the  fclk=32MHz  clock to RLIN0.*/
  LWBR0  = 0x01;         /*  b0=1, LIN2.0 or 2.1;  Prescaler Clock Selcet 1/1;  bit sampling count select 0000 : 16
sampling. */
  LBRP00 = 0x67;         /*   fa:0X67=103D, Baud rate= 32M/ (103+1)*16= 19230 bps  fb:9615bps    fc:2403bps */
  LBRP01 = 0x5F;         /*   fd:10416bps*/
```

```
   LIN0RVCIF = 0U;       /*  Clear Reception interrupt request signal */
  LIN0TRMIF = 0U;       /*  Clear Transmission interrupt request signal */
  LIN0WUPIF = 0U;       /*  Clear Wake up interrupt request signal */
  LIN0IF    = 0U;       /*  Clear LIN or LIN Status interrupt */

  LIN0RVCMK = 0U;       /*  interrupt reception servicing enable */
  LIN0TRMMK = 0U;       /*  interrupt transmission servicing enable */
  LIN0WUPMK = 0U;       /*  interrupt wake up servicing enable */
  LIN0MK    = 0U;       /*  interrupt Status servicing enable */
  LIE0  |=  0x0F;       /*  Enable successful response/wake-up reception interrupt, enable all interrupt*/
  LEDE0 |=  0x8F;       /*  Enable error detection */

   /*  Header format setting*/
  LMD0  =   0x10;        /*   b0b1=00: LIN master mode ; b3b2=00: fa=LIN sysclock;  b4=1:transmission
interrupt,sucessful reception interrupt...; b5=0: The noise filter is enable.*/
  LBFC0 =   0x15;        /*  b3-b0=0101: transmission break width 18Tbits; b5b4=01: break delimiter 2Tbit*/
  LSC0  =   0x11;        /*  b2-b0=001:inter-byte space 1bit or Response space 4bit; b5b4=01: inter-byte space 1Tbit;*/
  LWUP0 =   0x30;        /*  b7-b4=0100: Wake-up Transmission low width 4 bits.*/
  LIDB0 &=  0x00;        /*  Clear the ID buffer */

  ISC   =   0x00;        /*  INTP11 pin input signal is set as external interrupt input*/
  LINCKSEL|=0x10;        /*  Enable RLIN0 engine clock supply,*/

}

/****************************************************************************************************
* Function Name: RLIN_Master_HeaderTransmit(uint8_t ID)
* Description  : This function is setting in slave mode, enable header reception is started.
* Arguments    : None
* Return Value : None
****************************************************************************************************/
void RLIN_Master_HeaderTransmit(uint8_t ID)
{
  LCUC0 = 0x03;          /*  01: RLIN rest mode is canceled; 03:RLIN operation mode */
  LIDB0 = 0x00;          /*  clear ID buffer */
  LIDB0 = ID;            /*  ID  load to ID buffer */
  Clear_DataBuffer();    /* clear all data buffer */

  switch(ID)
     {
```

```
    case 0x08: RLIN_Master_Transmit(Master_TxData1,2);     /* ready for response transmit*/
          break;
    case 0x49: RLIN_Master_Transmit(Master_TxData2,2);     /* ready for response transmit*/
          break;
    case 0xCA: RLIN_Master_Transmit(Master_TxData3,2);     /* ready for response transmit*/
          break;
    case 0x8B: RLIN_Master_Receive(2);                     /* ready for response receive*/
          break;
    default:   break;
    }

  LTRC0=0x01;
}

/**************************************************************************************************
* Function Name: RLIN_Master_Transmit(void)
* Description  : This function seting data buffer for response transmission start
* Arguments    : uint8_t* databuf    : variable array data.
           uint8_t Data_length : transmit data length.
* Return Value : None
*************************************************************************************************/
void RLIN_Master_Transmit(uint8_t * databuf,uint8_t Data_length)
{
 uint8_t i;
 uint16_t  Databuf_adr;
 LDFC0=0x30;
 LDFC0|=Data_length;    /*  MSB=0011, RCDS=1:Transmission, LCS=1: Enhanced checksum mode;  LSB=0100: response data lengh select 4 byte*/
 Databuf_adr=RLIN_DateBuffer;
 for(i=0;i<Data_length;i++)
 {
  *((uint8_t *)(Databuf_adr+i))=databuf[i];
 }
}
/**************************************************************************************************
* Function Name: RLIN_Master_Receive(void)
* Description  : This function clear data buffer ready for response reception start
* Arguments    : uint8_t Data_length : receive data length.
* Return Value : None
**************************************************************************************************/
```

```
void RLIN_Master_Receive(uint8_t Data_length)
{
  Clear_DataBuffer();
  LDFC0=0x20;
  LDFC0|=Data_length;   /* MSB=0011, RCDS=1:Transmission, LCS=1: Enhanced checksum mode;  LSB=0100:
response data lengh select 4 byte*/
}

/**********************************************************************************************
* Function Name: RLIN_Master_GetData(void)
* Description  : This function is get data from data buffr according to ID data during the response reception
* Arguments    : uint8_t Data_length : receive data length.
* Return Value : None
**********************************************************************************************/
void RLIN_Master_GetData(void)
{
    switch(LIDB0)
    {
    case 0x8B: Get_reponse_RxData(Master_RxData1);
               P6=Master_RxData1[1];
             break;
    case 0x4c: Get_reponse_RxData(Master_RxData2);     /*no used*/
             break;
    case 0x0D: Get_reponse_RxData(Master_RxData3);     /*no used*/
             break;
    default: break;
      }
}

/**********************************************************************************************
* Function Name: Clear_DataBuffer
* Description  : This function setting all data buffer to some value
* Arguments    : uint8_t x : setting data buff value
* Return Value : None
**********************************************************************************************/

void Clear_DataBuffer()
{
  uint8_t i;
  uint16_t Databuf_adr;
  Databuf_adr=RLIN_DateBuffer;
  for(i=0;i<8;i++)
  {
   *((uint8_t *)(Databuf_adr+i))=0U;
  }
}


/*************************************************************************************************
* Function Name: Get_reponse_RxData
* Description  : This function get data buffer value to a variable array
* Arguments    : uint8_t * RxData : a avriable array for store Data
* Return Value : None
*************************************************************************************************/
uint_8  Get_reponse_RxData(uint8_t * RxData)
{
  uint8_t i,k;
  uint16_t Databuf_adr;
  k=LDFC0&0x0F;
  Databuf_adr=RLIN_DateBuffer;
  for(i=0;i<k;i++)
  {
    RxData[i]=(*((uint8_t *)(Databuf_adr+i)));
  }
Return RxData[1];
}
```

5.2 RLIN_driver_user.c

```
/*************************************************************************************************
* File Name    : RLIN_driver_user.c
* Device(s)    : R5F10PPJ
* Tool-Chain   : IAR Systems iccrl78
* Description  : This file implements device driver for Interrupt module.
* Creation Date: 02.08.2013
*************************************************************************************************/
#include "RLIN_macrodriver.h"
#include "RLIN_driver.h"
#include "RLIN_userdefine.h"
```

```
uint8_t GetIDbuffer;

/***********************************************************************************************
* Function Name: RLIN0_Transmission_interrupt
* Description  : This function is RLIN0 Transmission interrupt service routine.
* Arguments    : None
* Return Value : None
***********************************************************************************************/
#pragma vector = INTLIN0TRM_vect
__interrupt static void RLIN0_Transmission_interrupt(void)
{

  uint8_t transmit_header_flag;
  uint8_t transmit_response_flag;
  transmit_header_flag=LST0 & 0x80;
  transmit_response_flag=LST0 & 0x01;

    if(transmit_header_flag)
  {
    LST0&=0x7F;    /*clear successful header reception flag */
    }

  if(transmit_response_flag)
  {
    LST0&=0xFE;
  }
}
/***********************************************************************************************
* Function Name: RLIN0_Reception_interrupt
* Description  : This function is RLIN0 Reception interrupt service routine.
* Arguments    : None
* Return Value : None
**********************************************************************************************/
#pragma vector = INTLIN0RVC_vect
__interrupt static void RLIN0_Reception_interrupt(void)
{


LST0&=0xFD;    /*clear response reception successful flag*/
RLIN_Master_GetData();  /*get the reception data*/
}

/***********************************************************************************************
* Function Name: RLIN0_Status_interrupt
* Description  : This function is RLIN0 Status interrupt service routine.
* Arguments    : None
* Return Value : None
***********************************************************************************************/

#pragma vector = INTLIN0_vect
__interrupt static void RLIN0_Status_interrupt(void)
{
while(1U)
{
  ;
}
}

/***********************************************************************************************
* Function Name: RLIN0_Wakeup_interrupt
* Description  : This function is RLIN0 Wakeup interrupt service routine.
* Arguments    : None
* Return Value : None
***********************************************************************************************/
#pragma vector = INTLIN0WUP_vect
__interrupt static void RLIN0_Wakeup_interrupt(void)
{
 LCUC0=0x03;
 LED1=ON;
 LED2=ON;
}
```

5.3 RLIN _driver.h

```
/***********************************************************************************************
```

```
* File Name    : RLIN_Driver.h
* Device(s)    : R5F10PPJ
* Tool-Chain   : IAR Systems iccrl78
* Description  : This file implements device driver for PORT module.
* Creation Date: 15.07.2013
***************************************************************************************************/
#include "RLIN_userdefine.h"
void RLIN_Master_Init(void);     /* init Master RLIN0*/
void RLIN_Master_HeaderTransmit(uint8_t ID);
void RLIN_Master_Receive(uint8_t Data_length);
void RLIN_Master_Transmit(uint8_t * databuf,uint8_t Data_length);
void RLIN_Master_GetData(void);
void Clear_DataBuffer(void);
uint_8 Get_reponse_RxData(uint8_t * RxData);
```

5.4 RLIN_main.c

```
/****************************************************************************************************
************************
* File Name    : RLIN_main.c
* Device(s)    : R5F10PPJ
* Tool-Chain   : IAR Systems iccrl78
* Description  : This file implements main function.
* Creation Date: 02.08.2013
****************************************************************************************************/
/****************************************************************************************************
Includes
****************************************************************************************************/
#include "RLIN_macrodriver.h"
#include "RLIN_cgc.h"
#include "RLIN_port.h"
#include "RLIN_intc.h"
#include "RLIN_timer.h"
#include "RLIN_wdt.h"
#include "RLIN_driver.h"
#include "RLIN_userdefine.h"

/****************************************************************************************************
Global variables and functions
***************************************************************************************************/
```

```
/* Set option bytes */
#pragma location = "OPTBYTE"
__root const uint8_t opbyte0 = 0x7AU;
#pragma location = "OPTBYTE"
__root const uint8_t opbyte1 = 0xFFU;
#pragma location = "OPTBYTE"
__root const uint8_t opbyte2 = 0xE8U;
#pragma location = "OPTBYTE"
__root const uint8_t opbyte3 = 0x84U;

/* Set security ID */
#pragma location = "SECUID"
__root const uint8_t secuid[10] =
    {0x00U, 0x00U, 0x00U, 0x00U, 0x00U, 0x00U, 0x00U, 0x00U, 0x00U, 0x00U};

/* Secure trace RAM area */
__no_init __root unsigned char ocdtraceram[512] @ 0xFE300U;

/* Secure hot plug-in RAM area */
__no_init __root unsigned char hotpluginram[48] @ 0xFE500U;

void R_MAIN_UserInit(void);

/***********************************************************************************************
* Function Name: main
* Description  : This function implements main function.
* Arguments    : None
* Return Value : None
***********************************************************************************************/
void main(void)
{
     R_MAIN_UserInit();
     RLIN_Enable=TRUE;
     LED1=OFF;
     LED2=OFF;
     R_INTC0_Start();       /*waitting interrupt send header frame*/

      while (1U)
      {
        R_WDT_Restart();
       }

}



/************************************************************************************************
* Function Name: R_MAIN_UserInit
* Description  : This function adds user code before implementing main function.
* Arguments    : None
* Return Value : None
************************************************************************************************/
void R_MAIN_UserInit(void)
{
  RLIN_Master_Init();
   EI();
}
```

Website and Support

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

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

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

Revision History of RL78/F13, F14 Group, LIN Master Mode (RLIN3)

Rev.	Date	Description	
		Page	Summary
1.00	25.Sep.2013		First edition issued
1.01	29.May 2015		1st revision, source code changed on page 21, control of LIE0 register removed.

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

   Handle unused pins in accord 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 type number, confirm that the change will not lead to problems.

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

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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of 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. 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 should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended 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; and industrial robots etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate 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. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be 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, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

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

Renesas Electronics Corporation

SALES OFFICES

http://www.renesas.com

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

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

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: +44-1628-585-100, Fax: +44-1628-585-900

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, P. R. China 200333
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, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 5.0