[Notes]
RX Family
SCI Module Firmware Integration Technology,
RX Driver Package

R20TS0752ES0100
Rev.1.00
Sep. 16, 2021

Outline

When using the products in the title, note the following point.

1. Notes on possible junk byte being transmitted while using asynchronous mode

1. Notes on possible junk byte being sent out while using asynchronous mode

1.1 Applicable Products

(1) SCI module Firmware Integration Technology (SCI FIT module)

The applicable revision numbers and document numbers are as follows.

Table 1.1 SCI FIT module applicable products

Revision number of the SCI FIT module	Document number
Rev.3.70	R01AN1815EJ0370
Rev.3.60	R01AN1815EJ0360
Rev.3.50	R01AN1815EJ0350
Rev.3.40	R01AN1815EJ0340
Rev.3.30	R01AN1815EJ0330
Rev.3.21	R01AN1815EJ0321
Rev.3.20	R01AN1815EJ0320
Rev.3.10	R01AN1815EJ0310
Rev.3.00	R01AN1815EJ0300
Rev.2.20	R01AN1815EJ0220
Rev.2.11	R01AN1815EJ0211
Rev.2.10	R01AN1815EJ0210
Rev.2.01	R01AN1815EJ0201
Rev.2.00	R01AN1815EJ0200
Rev.1.90	R01AN1815EJ0190
Rev.1.80	R01AN1815EJ0180
Rev.1.70	R01AN1815EU0170

(2) RX Driver Package

The SCI FIT module in (1) is also included in the RX Driver Package.
The product names and revision numbers of the applicable RX Driver Package and the revision numbers of the SCI FIT module are as follows.

Table 1.2 Products which include the SCI FIT module

RX Driver Package product name	RX Driver Package revision number	Document number	Revision number of the included SCI FIT module
RX Family RX Driver Package Ver.1.30	Rev.1.30	R01AN5882xx0130	Rev.3.70
RX Family RX Driver Package Ver.1.29	Rev.1.29	R01AN5826xx0129	Rev.3.70

RX Family RX Driver Package Ver.1.27	Rev.1.27	R01AN5600xx0127	Rev.3.70
RX Family RX Driver Package Ver.1.26	Rev.1.26	R01AN5401xx0126	Rev.3.50
RX Family RX Driver Package Ver.1.25	Rev.1.25	R01AN5371xx0125	Rev.3.40
RX Family RX Driver Package Ver.1.24	Rev.1.24	R01AN5267xx0124	Rev.3.30
RX Family RX Driver Package Ver.1.23	Rev.1.23	R01AN4976xx0123	Rev.3.21
RX Family RX Driver Package Ver.1.22	Rev.1.22	R01AN4873xx0122	Rev.3.20
RX Family RX Driver Package Ver.1.20	Rev.1.20	R01AN4794EJ0120	Rev.3.00
RX Family RX Driver Package Ver.1.19	Rev.1.19	R01AN4677EJ0119	Rev.2.20
RX Family RX Driver Package Ver.1.18	Rev.1.18	R01AN4659EJ0118	Rev.2.11
RX Family RX Driver Package Ver.1.16	Rev.1.16	R01AN4471EJ0116	Rev.2.10
RX Family RX Driver Package Ver.1.15	Rev.1.15	R01AN4372EJ0115	Rev.2.01
RX Family RX Driver Package Ver.1.14	Rev.1.14	R01AN4191EJ0114	Rev.2.01
RX Family RX Driver Package Ver.1.13	Rev.1.13	R01AN3859EJ0113	Rev.2.00
RX Family RX Driver Package Ver.1.12	Rev.1.12	R01AN3651EJ0112	Rev.1.90
RX Family RX Driver Package Ver.1.11	Rev.1.11	R01AN3467EJ0111	Rev.1.80
RX Family RX Driver Package Ver.1.10	Rev.1.10	R01AN3345EJ0110	Rev.1.70
RX Family RX Driver Package Ver.1.04	Rev.1.04	R01AN2606EJ0104	Rev.1.70
RX Family RX Driver Package Ver.1.03	Rev.1.03	R01AN3233EJ0103	Rev.1.70
RX Family RX Driver Package Ver.1.02	Rev.1.02	R01AN3159EJ0102	Rev.1.70
RX Family RX Driver Package Ver.1.01	Rev.1.01	R01AN2670EJ0101	Rev.1.70

1.2 Applicable Devices

RX110, RX111, RX113, and RX130, RX13T groups

RX230, RX231, RX23E-A, RX23W, RX23T, RX24T, and RX24U groups

RX64M, RX65N, RX66N, and RX66T groups

RX71M, RX72T, RX72M, and RX72N groups

1.3 Details and Conditions

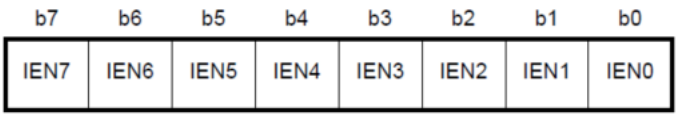
We have identified a potential error where asynchronous R_SCI_Send() and R_SCI_Receive() are called in a program. Under certain condition, an additional garbage byte will be transmitted

We have also identified the root cause as the IENx bit is corrupted due to non-atomic macros DISABLE_RXI_INT and ENABLE_RXI_INT; which are called in an asynchronous receive routine.

We use RX65N TXI5/RXI5 as example below:

Interrupt request enable bit for RXI5 and TXI5 are IER0A.IEN4 and IER0A.IEN5 respectively. Located in the same register means potential of corruption (due to read-modify-write) if the operation on the register is not atomic

IER0A register (RX65N):



b7	b6	b5	b4	b3	b2	b1	b0
IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IEN0

IER0A.IEN4 is used to enable or disable RXI5

IER0A.IEN5 is used to enable or disable TXI5

In asyncrhonous R_SCI_Receive() routine, the following macros will be called:

DISABLE_RXI_INT: IEN4 will be cleared to ‘0’

ENABLE_RXI_INT: IEN4 will be set to ‘1’

Note that these two macros are not atomic (verified with disassembly code)

```
static sci_err_t sci_receive_async_data(sci_hdl_t const    hdl, uint8_t    *p_dst, uint16_t const    length)
{
      …
      DISABLE_RXI_INT;        // this is not atomic! 1
      byteq_err = R_BYTEQ_Get(hdl->u_rx_data.que, p_dst++);
      ENABLE_RXI_INT;        // this is not atomic 2
      …
}
```

In asynchronous R_SCI_Send() routine, the following macro will be called when the last byte is copied to TDR register:

DISABLE_TXI_INT: IEN5 will be cleared to ‘0’

```
void txi_handler(sci_hdl_t const hdl) 3
{
      …
      DISABLE_TXI_INT; // this will be called when last byte of data is copied to TDR
                              // Clear IEN5 to ‘0'
      …
}
```

The following explains how IEN5 can be corrupted, and how it results to additional garbage data being transmitted:

```
void main(void)
{
    …
    R_SCI_Send(…);    // txi_handler() will be triggerred n times if n bytes are to be sent
    R_SCI_Receive(…); // txi_handler() could be triggered while R_SCI_Receive() is in progress
    …
}
```

- When all n bytes are transmitted, expected value of IEN5 is '0'
- But if either ❶ or ❷ is interrupted by the last (n^{th}) txi_handler() (❸), IEN5 could unexpectedly become '1' due to *read-modify-write* at the end of non-atomic ❶❷

  When this happens, txi_handler() will be triggered once more, even though all data has been sent. This results in an additional garbage byte being transmitted

1.4 Workaround

Make the following macros atomic (note that RX65N is used as example):

Before modification

```
#define ENABLE_RXI_INT      (*hdl->rom->icu_rxi |= hdl->rom->rxi_en_mask)
#define DISABLE_RXI_INT     (*hdl->rom->icu_rxi &= (uint8_t)~hdl->rom->rxi_en_mask)
```

After modification

```
#define ENABLE_RXI_INT      (R_BSP_BIT_SET(hdl->rom->icu_rxi, hdl->rom->rxi_bit_num))
#define DISABLE_RXI_INT     (R_BSP_BIT_CLEAR(hdl->rom->icu_rxi, hdl->rom->rxi_bit_num))
```

1.5 Schedule for Fixing the Problem

This problem will be fixed in Rev.3.80.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Sep.16.21	-	First edition issued

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

TS Colophon 4.2