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

R20TS1223EJ0100
Rev.1.00
Apr. 20, 2026

Outline

When using the following products, note the points below.

1. Precautions when using interface header definitions that use definitions related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_LEFT.
2. Precautions when using interface header definitions that use definitions related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_RIGHT, R_BSP_PRAGMA_UNPACK, and R_BSP_PRAGMA_PACKOPTION.

1. Precautions when using interface header definitions that use definitions related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_LEFT.

1.1 Applicable Products

(1) Graphic LCD Controller Module Using Firmware Integration Technology

All revisions from Rev.1.50 / R01AN1980EJ0150 to Rev.1.61 / R01AN1980EJ0161

(2) EPTPC Module Firmware Integration Technology

Rev.1.17 / R01AN1943EJ0117

(3) I^2C Bus Interface (RIIC) Module Using Firmware Integration Technology

All revisions from Rev.2.49 / R01AN1692EJ0249 to Rev.3.11 / R01AN1692EJ0311

(4) High-Speed I^2C Bus Interface (RIICHS) Module Using Firmware Integration Technology

All revisions from Rev.1.00 / R01AN5552EJ0100 to Rev.1.23 / R01AN5552EJ0123

(5) RSCI Simple I^2C Module Using Firmware Integration Technology

All revisions from Rev.1.00 / R01AN8028EJ0100 to Rev.1.01 / R01AN8028EJ0101

(6) Simple I^2C Module Using Firmware Integration Technology

All revisions from Rev.2.49 / R01AN1691EJ0249 to Rev.2.91 / R01AN1691EJ0291

(7) VBATT Module Firmware Integration Technology

All revisions from Rev.2.10 / R01AN2796EJ0210 to Rev.2.33 / R01AN2796EJ0233

(8) RX Driver Package

The above products (1) to (7) are also included in the following RX Driver Package.

All revisions from Rev.1.33 / R01AN6073EJ0133 to Rev.1.50 / R01AN8254EJ0150

1.2 Applicable Devices

RX family

1.3 Details

If the conditions in 1.4 Conditions are met, the definitions related to bit_order in the definitions related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_LEFT become invalid. This can cause the following issue when referencing or setting.

1.3.1 Issues caused by referencing

If a variable is passed as an argument to an API function, and that variable is referenced via a bit structure definition after the API function executes, the expected value cannot not be read because the reference target is incorrect.

1.3.2 Issues caused by setting

If a variable is set using a bit structure definition and passed as an argument to an API function, and the API function contains processing that references the bit structure definition, the reference target will be incorrect and the execution result will differ from expectations.

1.4 Conditions

This occurs when all of the following conditions are met:

1.4.1 Conditions that cause issues when referencing

- Create a CCRX and C++ project.
- Incorporate any of modules (3), (4), (5), (6), or (7) from 1.1 Applicable Products into the project.
- In the cpp file, declare variables using a structure definition that uses the definition related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_LEFT in the interface header, and execute the relevant API function.
- After executing the API function in the cpp file, reference the variable using the bit structure definition.

The following is an example of conditions that may trigger this issue in the relevant module:

(3) I²C Bus Interface (RIIC) Module Using Firmware Integration Technology

```
volatile riic_return_t ret;
riic_info_t iic_info_m;
riic_mcu_status_t riic_status;

iic_info_m.ch_no = 0;
ret = R_RIIC_GetStatus(&iic_info_m, &riic_status);
/* Referenced in the bit structure definition */
if( 0 == riic_status.BIT.SDAI )
{
   /* Processing when SDA is Low */
}
```

1.4.2 Conditions that cause issues when setting

- Create a CCRX and C++ project.
- Incorporate any of modules (1), (2), or (7) from 1.1 Applicable Products into the project.
- In the cpp file, declare variables using a structure definition that uses the definition related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_LEFT in the interface header.
- In the cpp file, set variables using the bit structure definition and execute the relevant API function.

The following is an example of conditions that may trigger this issue in the relevant module :

(1) Graphic LCD Controller Module Using Firmware Integration Technology

```
volatile glcdc_err_t ret_glcdc;
glcdc_color_t bg_color;
/* Set variables with bit structure definition */
bg_color.byte.r = 0x00h;
bg_color.byte.g = 0x00h;
bg_color.byte.b = 0xFFh;

ret_glcdc = R_GLCDC_Control(GLCDC_CMD_CHANGE_BG_COLOR, (void *)&bg_color);
```

1.5 Workaround

In the module interface header (*_if.h) for the products listed in 1.1 Applicable Products, add a preprocessor declaration for bit_order left directly to the structure definition that uses the definition related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_LEFT.

An example of a workaround is shown below. Please make the same modifications for the relevant modules.

(3) I²C Bus Interface (RIIC) Module Using Firmware Integration Technology

r_riic_rx_if.h excerpt

Before modification

```
typedef union
{
    uint32_t LONG;
    R_BSP_ATTRIB_STRUCT_BIT_ORDER_LEFT_21
    (
        uint32_t rsv :12, /* reserve */
        uint32_t AAS2 :1, /* Slave2 address detection flag */
        uint32_t AAS1 :1, /* Slave1 address detection flag */
        uint32_t AAS0 :1, /* Slave0 address detection flag */
        uint32_t GCA :1, /* Generalcall address detection flag */
        uint32_t DID :1, /* DeviceID address detection flag */
        uint32_t HOA :1, /* Host address detection flag */
        uint32_t MST :1, /* Master mode / Slave mode flag */
        uint32_t TMO :1, /* Time out flag */
        uint32_t AL :1, /* Arbitration lost detection flag */
        uint32_t SP :1, /* Stop condition detection flag */
        uint32_t ST :1, /* Start condition detection flag */
        uint32_t RBUF :1, /* Receive buffer status flag */
        uint32_t SBUF :1, /* Send buffer status flag */
        uint32_t SCLO :1, /* SCL pin output control status */
        uint32_t SDAO :1, /* SDA pin output control status */
```

```
        uint32_t SCLI :1, /* SCL pin level */
        uint32_t SDAI :1, /* SDA pin level */
        uint32_t NACK :1, /* NACK detection flag */
        uint32_t TRS :1, /* Send mode / Receive mode flag */
        uint32_t BSY :1 /* Bus status flag */
    ) BIT;
} riic_mcu_status_t;
```

After correction (red part added)

```
#pragma bit_order left
typedef union
{
    uint32_t LONG;
    R_BSP_ATTRIB_STRUCT_BIT_ORDER_LEFT_21
    (
        uint32_t rsv :12, /* reserve */
        uint32_t AAS2 :1, /* Slave2 address detection flag */
        uint32_t AAS1 :1, /* Slave1 address detection flag */
        uint32_t AAS0 :1, /* Slave0 address detection flag */
        uint32_t GCA :1, /* Generalcall address detection flag */
        uint32_t DID :1, /* DeviceID address detection flag */
        uint32_t HOA :1, /* Host address detection flag */
        uint32_t MST :1, /* Master mode / Slave mode flag */
        uint32_t TMO :1, /* Time out flag */
        uint32_t AL :1, /* Arbitration lost detection flag */
        uint32_t SP :1, /* Stop condition detection flag */
        uint32_t ST :1, /* Start condition detection flag */
        uint32_t RBUF :1, /* Receive buffer status flag */
        uint32_t SBUF :1, /* Send buffer status flag */
        uint32_t SCLO :1, /* SCL pin output control status */
        uint32_t SDAO :1, /* SDA pin output control status */
        uint32_t SCLI :1, /* SCL pin level */
        uint32_t SDAI :1, /* SDA pin level */
        uint32_t NACK :1, /* NACK detection flag */
        uint32_t TRS :1, /* Send mode / Receive mode flag */
        uint32_t BSY :1 /* Bus status flag */
    ) BIT;
} riic_mcu_status_t;
#endif bit_order
```

1.6 Schedule for Fixing the Problem

Except for (2) in 1.1 Applicable Products these issues are scheduled to be fixed in future versions.

2. Precautions when using interface header definitions that use definitions related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_RIGHT, R_BSP_PRAGMA_UNPACK, and R_BSP_PRAGMA_PACKOPTION.

2.1 Applicable Products

(1) Graphic LCD Controller Module Using Firmware Integration Technology

All revisions from Rev.1.50 / R01AN1980EJ0150 to Rev.1.61 / R01AN1980EJ0161

(2) I²C Bus Interface (RIIC) Module Using Firmware Integration Technology

All revisions from Rev.2.49 / R01AN1692EJ0249 to Rev.3.11 / R01AN1692EJ0311

(3) High-Speed I²C Bus Interface (RIICHS) Module Using Firmware Integration Technology

All revisions from Rev.1.00 / R01AN5552EJ0100 to Rev.1.23 / R01AN5552EJ0123

(4) RSCI Simple I²C Module Using Firmware Integration Technology

All revisions from Rev.1.00 / R01AN8028EJ0100 to Rev.1.01 / R01AN8028EJ0101

(5) RSPI Module Using Firmware Integration Technology

All revisions from Rev.3.03 / R01AN1827EJ0303 to Rev.3.81 / R01AN1827EJ0381

(6) RSPIA Module Using Firmware Integration Technology

All revisions from Rev.1.10 / R01AN5684EJ0110 to Rev.1.63 / R01AN5684EJ0163

(7) Simple I²C Module Using Firmware Integration Technology

All revisions from Rev.2.49 / R01AN1691EJ0249 to Rev.2.91 / R01AN1691EJ0291

(8) USB Host Communications Device Class Driver (HCDC) using Firmware Integration Technology

All revisions from Rev.1.31 / R01AN2027EJ0131 to Rev.1.45 / R01AN2027EJ0145

(9) USB Host Communications Device Class Driver for USB Mini Firmware Using Firmware Integration Technology

All revisions from Rev.1.20 / R01AN2167EJ0131 to Rev.1.32 / R01AN2167EJ0132

(10) USB Peripheral Communications Device Class Driver (PCDC) Firmware Integration Technology

All revisions from Rev.1.31 / R01AN2030EJ0131 to Rev.1.45 / R01AN2030EJ0145

(11) USB Peripheral Communications Device Class Driver for USB Mini Firmware Using Firmware Integration Technology

All revisions from Rev.1.20 / R01AN2170EJ0131 to Rev.1.32 / R01AN2170EJ0132

(12) VBATT Module Firmware Integration Technology

All revisions from Rev.2.10 / R01AN2796EJ0210 to Rev.2.33 / R01AN2796EJ0233

(13) RX Driver Package

The above products (1) to (12) are also included in the following RX Driver Package.

All revisions from Rev.1.33 / R01AN6073EJ0133 to Rev.1.50 / R01AN8254EJ0150

2.2 Applicable Devices

RX family

2.3 Details

If the conditions in 2.4 Conditions are met, the definition related to bit_order in the definition related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_RIGHT will be invalidated. In addition, R_BSP_PRAGMA_UNPACK and R_BSP_PRAGMA_PACKOPTION may also become invalid. As a result, even if pragma declarations are specified in the source file, they may be invalidated after compilation, leading to a mismatch between the declared state and the actual state.

2.4 Conditions

This occurs when all of the following conditions are met:

- Create a CCRX and C++ project.
- Incorporate any modules from 2.1 Applicable Products into the project.
- In the cpp file, declare and use variables using the structure definition that uses the definitions related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_RIGHT and the definition of R_BSP_PRAGMA_UNPACK in the interface header.

2.5 Workaround

2.5.1 Workaround for modules that use definitions related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_RIGHT

In the module interface header (*_if.h) for the products listed in 2.1 Applicable Products, add a preprocessor declaration for bit_order left directly to the structure definition that uses the definition related to R_BSP_ATTRIB_STRUCT_BIT_ORDER_RIGHT.

An example of a workaround is shown below. Please make the same modifications for the relevant modules.

(5) RSPI Module Using Firmware Integration Technology

r_rspi_rx_if.h excerpt

Before modification

```
typedef union rspi_command_word_s
{
   R_BSP_ATTRIB_STRUCT_BIT_ORDER_RIGHT_11 (
       rspi_spcmd_cpha_t        cpha         :1,
       rspi_spcmd_cpol_t        cpol         :1,
       rspi_spcmd_br_div_t      br_div       :2,
       rspi_spcmd_ssl_assert_t   ssl_assert    :3,
       rspi_spcmd_ssl_negation_t ssl_negate    :1,
       rspi_spcmd_bit_length_t   bit_length    :4,
       rspi_spcmd_bit_order_t    bit_order     :1,
       rspi_spcmd_spnden_t      next_delay    :1,
       rspi_spcmd_slnden_t      ssl_neg_delay :1,
       rspi_spcmd_sckden_t      clock_delay   :1,
       rspi_spcmd_dummy_t       dummy        :16
);
   uint16_t word[2];
```

```
} rspi_command_word_t;
```

After correction (red part added)

```
#pragma bit_order right
typedef union rspi_command_word_s
{
   R_BSP_ATTRIB_STRUCT_BIT_ORDER_RIGHT_11 (
       rspi_spcmd_cpha_t        cpha         :1,
       rspi_spcmd_cpol_t        cpol         :1,
       rspi_spcmd_br_div_t      br_div       :2,
       rspi_spcmd_ssl_assert_t   ssl_assert    :3,
       rspi_spcmd_ssl_negation_t ssl_negate    :1,
       rspi_spcmd_bit_length_t   bit_length    :4,
       rspi_spcmd_bit_order_t    bit_order     :1,
       rspi_spcmd_spnden_t      next_delay    :1,
       rspi_spcmd_slnden_t      ssl_neg_delay :1,
       rspi_spcmd_sckden_t      clock_delay   :1,
       rspi_spcmd_dummy_t       dummy        :16
);
   uint16_t word[2];
} rspi_command_word_t;
#endif bit_order
```

2.5.2 Workaround for modules that use R_BSP_PRAGMA_UNPACK and R_BSP_PRAGMA_PACKOPTION definitions

In the module interface header (*_if.h) for the products listed in 2.1 Applicable Products, add a preprocessor declaration for bit_order left directly to the structure definition that uses the definition related to R_BSP_PRAGMA__UNPACK. Add the preprocessor declaration for packoption directly to the definition of R_BSP_PRAGMA_PACKOPTION.

Before modification

```
R_BSP_PRAGMA_UNPACK
```

```
R_BSP_PRAGMA_PACKOPTION
```

After correction (red part added)

```
R_BSP_PRAGMA_UNPACK
#pragma unpack
```

```
R_BSP_PRAGMA_PACKOPTION
#pragma packoption
```

2.6 Schedule for Fixing the Problem

This will be fixed in a future version.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Apr.20.26	-	First edition issued

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.

The past news contents have been based on information at the time of publication. Now changed or invalid information may be included.

The URLs in the Tool News also may be subject to change or become invalid without prior notice.

Corporate Headquarters

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

Trademarks

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/

TS Colophon 4.3