

Microcontroller Technical Information

RX78K0R 78K0R Real-Time OS Usage Restrictions	Document No.	ZMT-F35-10-0013	1/1
	Date issued	March 30, 2011	
	Issued by	MCU Tool Product Marketing Department MCU Software Division MCU Business Unit Renesas Electronics Corporation	
Related documents None	Notification classification	√	Usage restriction
			Upgrade
			Document modification
			Other notification

1. Affected product

RX78K0R Ver. 4.30 package (kernel Ver. 4.30) and earlier versions

2. New restrictions

The following restriction related to the kernel has been added:

- No. 4 AND and OR specified together as the wait conditions for the same event flag during processing of the same task

3. Workarounds

The following workaround is available for the above restriction. See attachment 1 for details.

- No. 4 Do not specify AND and OR together as the wait conditions for the same event flag during processing of the same task.

4. Modification schedule

- No. 4 This will not be corrected, so regard it as a specification.

5. List of restrictions

A list of restrictions in the RX78K0R, including the revision history and detailed information, is described on the following page.

6. Revision history

Document Number	Issued on	Description
ZBG-CD-07-0043	July 11, 2007	1st edition
ZBG-CD-07-0065	September 19, 2007	Addition of new kernel restriction (No. 2)
ZBG-CD-09-0001	January 15, 2009	Addition of new kernel restriction (No. 3)
ZBG-CD-09-0035	June 18, 2009	Addition of new CubeSuite-related restriction (No. 1)
ZMT-F35-10-0013	March 30, 2011	Addition of new kernel restriction (No. 4)

Restrictions Related to RX78K0R Kernel

1. Product History

No.	Usage Restrictions	Kernel Version			
		4.00	4.11	4.20	4.30
1	Incorrect operation occurs if the <code>sysarea</code> segment is allocated to external RAM (an address out of address range F0000H to FFFFFH).	√			
2	A linker error is output if <code>rx78k0r</code> and <code>?CSEGUP</code> segments are allocated far from each other.	√			
3	Behavior is invalid after an interrupt handler has run during initial task startup.	√	√	√	
4	AND and OR specified together as the wait conditions for the same event flag during processing of the same task	√	√	√	√

√: Applicable

2. Restriction Details

No. 1 Incorrect operation occurs if the `sysarea` segment is allocated to external RAM (an address out of address range F0000H to FFFFFH).

Description:

If the `sysarea` segment (RAM area managed by RX) is allocated to external RAM (an address out of address range F0000H to FFFFFH), the incorrect operations shown below occur. The `sysarea` segment can be allocated to internal RAM (an address within address range F0000H to FFFFFH) without problem.

- When a task waiting for an event flag is to be released from the wait state due to establishment of conditions, the wait state is not released correctly.
- If a mailbox has been defined, the mailbox cannot be initialized correctly.
- When a task or cyclic handler, which has been waiting for timeout for two ticks or longer, times out, the task or cyclic handler does not operate correctly.

These occur because the ES register setting is mistakenly accessed during RX78K0R internal processing, and execution of such a code makes the address information, which is held in the `sysarea` segment and managed by the RX78K0R, invalid.

Workaround:

Allocate the `sysarea` segment to internal RAM (an address within address range F0000H to FFFFFH).

The following shows an example for coding a link directive file that allocates the `sysarea` segment to internal RAM.

Example:

```
MEMORY ROM : ( 0H, FFFFH )
MEMORY STK : ( 0FD700H, 600H )
MEMORY RAM : ( 0FDD00H, 2200H ) ; Defines internal RAM area
MERGE rx78k0r : = ROM
MERGE rxinf : = ROM
MERGE sit : = ROM
MERGE stkarea : = STK
MERGE sysarea : = RAM ; Allocates sysarea to internal RAM area
```

```
MERGE p0area : = RAM
```

Correction:

This issue has been corrected in Ver. 4.11.

No. 2 A linker error is output if `rx78k0r` and `?CSEGUP` segments are allocated far from each other.

Description:

The following linker error is output if addresses at which the `rx78k0r` segment and the `?CSEGUP` segment are respectively allocated are about 8000H away from each other.

```
RA78K0R error E3304: Operand out of range (file 'C:\...\librx.lib',
                    segment 'rx78k0r', symbol 'xxxxxx', address 84BFH,
                    type '$!addr20')
```

This restriction does not adversely affect to the code if linking is finished correctly.

Workaround:

Allocate the `rx78k0r` and `?CSEGUP` segments close to each other, using a link directive file. A code example is shown below.

Example:

```
MEMORY ?CSEGUP: AT( 07000H )
MERGE rx78k0r: AT( 07100H )
```

Correction:

This issue has been corrected in Ver. 4.11.

No. 3 Behavior is invalid after an interrupt handler has run during initial task startup.

Description:

The following two phenomena occur if an interrupt handler (managed by the OS) operates while a task for which startup interrupt status is authorized (`TA_ENAINT` attribute) is performing OS processing during initial startup^{Note 1}.

1. The interrupt handler operates with an invalid stack.

Although the specification calls for the system stack to be used during interrupt handler processing, the task stack is used erroneously. The task stack is also used even if more than one interrupt handler runs in this state.

Using the task stack for interrupt-handler processing does not pose a problem as long as there is enough stack space, but if there is not enough stack space, the stack will become corrupt and behavior will be undefined.

2. Task scheduling is delayed.

The specifications state that if a schedule is created while an interrupt handler is running, then task scheduling will be performed after the interrupt handler returns.

If an interrupt handler was accepted during the task's initial startup, however, then if a schedule is created while the handler is running, task scheduling is not performed after the handler returns.

Instead, it is delayed until the next time task scheduling is performed^{Note 2}.

Note 1 *Initial task startup* refers to the initial start of task processing after a task's status is changed from Dormant to Running.

Note 2 Task scheduling is performed after a service call capable of scheduling returns, and after an interrupt handler returns.

Workaround:

There is no workaround.

Correction:

This issue has been corrected in Ver. 4.30.

No. 4 AND and OR specified together as the wait conditions for the same event flag during processing of the same task

Description:

When the service call `wai_flg` or `twai_flg` is issued multiple times with both OR and AND specified at least once as the bit pattern checking condition for the same event flag during processing of the same task, if a subsequent service call is issued with AND specified, the condition might be handled as OR.

The operation in question might occur under the following conditions:

1. During task A processing, `wai_flg` or `twai_flg` is issued for event flag X, with OR specified as the bit pattern checking condition.

Task A then enters the state in which the operation in question might occur.

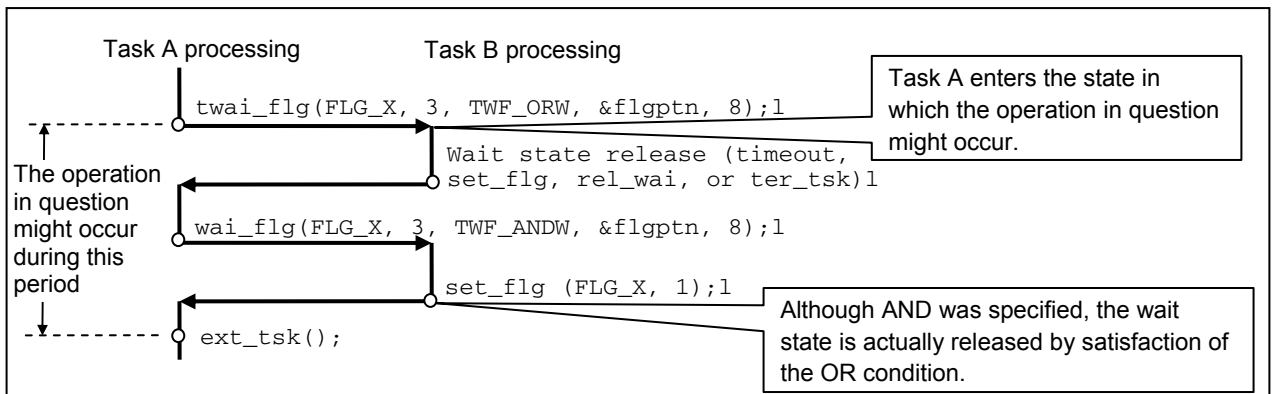
2. During task A processing, `wai_flg` or `twai_flg` is issued for event flag X, with AND specified as the bit pattern checking condition.

Task A then enters the event flag wait state.

3. During another process, `set_flg` is issued for event flag X, with OR specified to release the wait state.

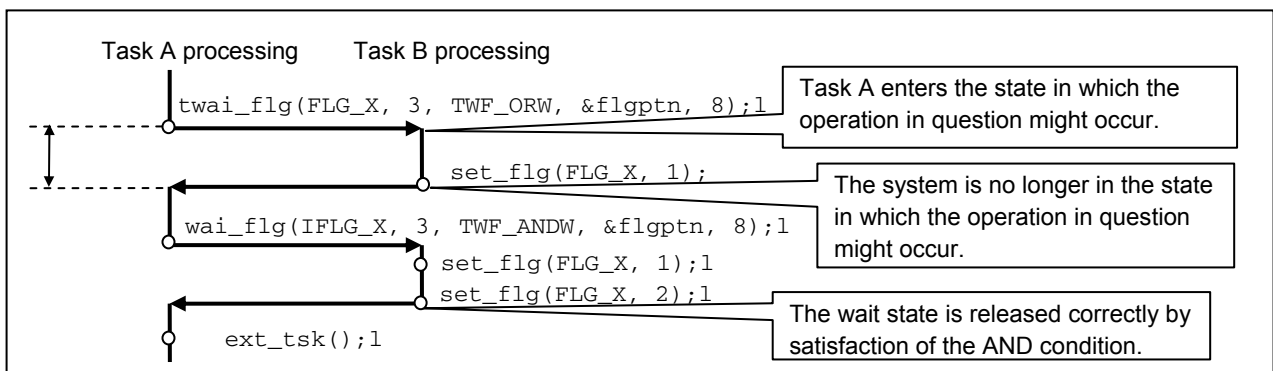
Although AND is specified as the condition for releasing the wait state in 2, the wait state is actually released by satisfaction of the OR condition.

Example 1:



To recover from the state in which the operation in question might occur, issue `ext_tsk` or `ter_tsk` for the relevant task, or initialize the operating system. For example, this operation does not occur if the wait state is released by issuing `set_flg` as follows:

Example 2:



Workaround:

Do not specify AND and OR together as the wait conditions for the same event flag during processing of the same task.

Correction:

This will not be corrected, so regard it as a specification.

CubeSuite-Related Usage Restrictions

1. Product History

No.	Usage Restrictions	Package Version	
		4.20 or earlier	4.30
1	Restriction that the Help menu cannot be opened from a message by using the F1 key		√

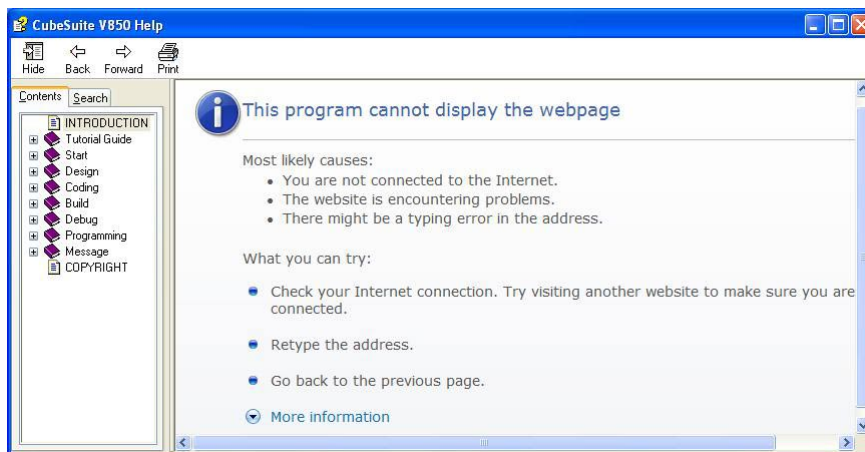
√: Applicable

2. Restriction Details

No. 1 Restriction that the **Help** menu cannot be opened from a message by using the **F1** key

Description:

If the **F1** key is pressed while the cursor is pointing to a message output by the build tool in an RX78K0R project, the **Help** menu does not open with an appropriate explanation. Instead, the following **Help** message is displayed:



Workaround:

On the menu bar, click **Help** and then **Help for CubeSuite** to open the **CubeSuite Help** dialog box.

Search for the appropriate **Help** message in this dialog box.

Correction:

This issue has been corrected in Realtime OS Build Tool Plug-in (Common) Ver. 1.01.