NOTICE：
There is a correction to the information on the version of C/C++ Compiler Package for RX Family in "4.3 Supported Tools" on page 3.
Incorrect: V.1.00 Release 00 or later Correct: V.1.01 Release 00 or later

RI600/4 V.1.01 Release 00

Release Notes

R20UT0797EJ0100
Rev.1.00
Oct 14, 2011

Contents

1. Packaged Tools

Agreement type and contents are different according to the product.

Product Name	Agreement Type	Contents
R0R5RX00TRW011	Evaluation License, Limited 1 host	A
R0R5RX00TRW01A	Evaluation License, Unlimited hosts	A
R0R5RX00TRW01K	Mass-production License, 3000 copies	A
R0R5RX00TRW01U	Mass-production License, Unlimited copies	A
R0R5RX00TRW01Z	Mass-production License, Unlimited copies, With source code	B

The following tools are provided.

Contents		Name
B	A	Kernel object
		Command-line Configurator “cfg600”
		GUI Configurator “GUI600”
		Table generation Utility “mkritbl”
		Kernel source code

2. Tool News

Tool News provides information on our products so that customers can use the products more efficiently.

The Tool News pages are available on our Web site.

URL : http://tool-support.renesas.com/eng/toolnews/index.htm

Get the latest information about new products, upgraded versions and precautions from Tool News, and take advantage of it in your development projects. Since the release notes do not include information issued after the release of the product, be sure to check the latest issue of Tool News.

3. Target Devices

The following devices are supported by the product.

- RX600 series MCU
- RX200 series MCU

4. Operating Environment

Below is described the operating environment for using the product.

4.1 Hardware Environment

- Memory capacity: 256 MB or more recommended. Minimum requirement is 128 MB or more
- Display: Resolution at least 800 x 600

4.2 Software Environment

The following software environments are supported.

- Windows XP (32bit)
- Windows Vista (32bit, 64bit)
- Windows 7 (32bit, 64bit)

Remark: For any of these, we recommend having the latest service pack installed.

4.3 Supported Tools

The following tools are supported.

Tool Name	Version
C/C++ Compiler Package for RX Family	~~V.1.00 Release 00 or later~~ V.1.01 Release 00 or later

Note, the RX family C/C++ Compiler Package V.1.02 Release 00 or later is required to generate RI600/PX project by using High-performance Embedded Workshop. The build-setting of the project generated with the version before this is improper.

5. Installation and Uninstallation

Windows administrator privileges are required to install and uninstall the software.

To install the software, start “setup.exe” in the root directory of the CD, and then follow the instructions displayed on the screen. When installation, close all applications.

To uninstall the software, select [RI600/4 V.1.01 Release 00] from [Add/Remove Program] of the Control-Panel.

6. Timer Template File

The relation between template file provided by RI600/4 and corresponded MCUs is shown as follows.

The timer template file is specified to “clock.template” in the system configuration file.

Template File	Corresponded MCUs
rx610.tpl	RX610 group
rx62t.tpl	RX62T group
rx62n.tpl	RX62N group
rx630.tpl	RX630 group
rx210.tpl	RX210 group

7. How to Build Kernel Source Code[1]

The kernel source code is stored in "< installation directory >\src600". It moves to the source code installation directory to build the kernel, and "nmake.exe"[2] is executed. The environment variable settings are needed by compiler when building the kernel.

Example:

```
C:\RI600-4\v101r00\src600> nmake(RET)
```

After the building the kernel, the kernel library is generated to the following directories.

Kernel Library Name	Contents
product\big\debug\ri600big.lib	Big endian library with debugging information
product\big\release\ri600big.lib	Big endian library without debugging information
product\little\debug\ri600lit.lib	Little endian library with debugging information
product\little\ release\ri600lit.lib	Little endian library without debugging information

Please copy "src600" directory to the writable directory if you don't have the write-access permission to the product installation directory. After the build, copy the generated library to the "lib600" directory under the product installation directory by the user who has write-access permission to the product installation directory.

1 The source code is only attached to R0R5RX00TRW01Z.

2 "nmake.exe" is a tool to build the project provided by Microsoft Corporation in United States. "nmake.exe" is included in Microsoft Visual Studio 2008 etc.

8. Stack Consumption

8.1 Stack Consumption of System Clock Interrupt Handler (ε1, ε2, ε3)

The value of ε1, ε2 and ε3 described in the RI600/4 User's Manual paragraph 12.4 are as follows.

- ε1 = 104
- ε2 = 104
- ε3 = 192

8.2 Stack Consumption of Service Calls (α)

In the service call, the stack is used as follows.

(1) Called from the task context

The stack in the task context execution is a user stack. The service call is using following.

(a) User stack (Former call stack)

(b) System stack

(2) Called from the non-task context

The stack in the non-task context execution is a system stack. The service call is using following.

(a) System stack (Former call stack)

The use size of former stack ((a), (c)) which the service call uses is displayed by Call Walker.

Moreover, to calculate consumption of the system stack described in manual 12.4, the size of (b) and (c) is needed. (Paragraph 12.4 has described as α.)The size of (a), (b) and (c) of each service call is shown as follows.

	Service call	The use size of User stack(a)	The use size of System stack ((b),(c))	Note
Task Management Function				
1	act_tsk	0	24	
2	iact_tsk	0	24	
3	can_act	0	24	
4	ican_act	0	24	
5	sta_tsk	0	24	
6	ista_tsk	0	24	
7	ext_tsk	0	64	ext_tsk is called at the return from the task beginning function.
8	ter_tsk	0	128	
9	chg_pri	0	36	
10	ichg_pri	0	52	
11	get_pri	0	24	
12	iget_pri	0	24	
13	ref_tsk	0	28	
14	iref_tsk	0	28	
15	ref_tst	0	24	
16	iref_tst	0	24	
Task Dependent Synchronization Function				
17	slp_tsk	0	24	
18	tslp_tsk	0	24	
19	wup_tsk	0	40	
20	iwup_tsk	0	52	

	Service call	The use size of User stack(a)	The use size of System stack ((b),(c))	Note
21	can_wup	0	24	
22	ican_wup	0	24	
23	rel_wai	0	112	
24	irel_wai	0	128	
25	sus_tsk	0	24	
26	isus_tsk	0	24	
27	rsm_tsk	0	24	
28	irsm_tsk	0	24	
29	frsm_tsk	0	24	
30	ifrsm_tsk	0	24	
31	dly_tsk	0	24	
Semaphore				
32	sig_sem	0	44	
33	isig_sem	0	60	
34	wai_sem	0	32	
35	pol_sem	0	24	
36	ipol_sem	0	24	
37	twai_sem	0	36	
38	ref_sem	0	24	
39	iref_sem	0	24	
Eventflag				
40	set_flg	0	48	
41	iset_flg	0	64	
42	clr_flg	0	24	
43	iclr_flg	0	24	
44	wai_flg	0	44	
45	pol_flg	0	24	
46	ipol_flg	0	24	
47	twai_flg	0	48	
48	ref_flg	0	24	
49	iref_flg	0	24	
Data Queue				
50	snd_dtq	0	36	
51	psnd_dtq	0	32	
52	ipsnd_dtq	0	52	
53	tsnd_dtq	0	40	
54	fsnd_dtq	0	32	
55	ifsnd_dtq	0	52	
56	rcv_dtq	0	32	
57	prcv_dtq	0	32	
58	iprcv_dtq	0	52	
59	trcv_dtq	0	32	
60	ref_dtq	0	24	
61	iref_dtq	0	24	
Mailbox				
62	snd_mbx	0	40	
63	isnd_mbx	0	56	
64	rcv_mbx	0	32	
65	prcv_mbx	0	28	
66	iprcv_mbx	0	28	
67	trcv_mbx	0	36	
68	ref_mbx	0	24	
69	iref_mbx	0	24	
Mutex				
70	loc_mtx	0	40	
71	ploc_mtx	0	24	
72	tloc_mtx	0	44	

	Service call	The use size of User stack(a)	The use size of System stack ((b),(c))	Note
73	unl_mtx	0	56	
74	ref_mtx	0	24	
Message Buffer				
75	snd_mbf	0	44	
76	psnd_mbf	0	44	
77	ipsnd_mbf	0	60	
78	tsnd_mbf	0	44	
79	rcv_mbf	0	56	
80	prcv_mbf	0	56	
81	trcv_mbf	0	56	
82	ref_mbf	0	24	
83	iref_mbf	0	24	
Fixed-sized Memory Pool				
84	get_mpf	0	48	
85	pget_mpf	0	36	
86	ipget_mpf	0	36	
87	tget_mpf	0	48	
88	rel_mpf	20	36	
89	irel_mpf	0	56	
90	ref_mpf	0	24	
91	iref_mpf	0	24	
Variable Size Memory Pool				
92	get_mpl	36	92	
93	pget_mpl	0	108	
94	ipget_mpl	0	108	
95	tget_mpl	36	92	
96	rel_mpl	0	104	
97	ref_mpl	0	24	
98	iref_mpl	0	24	
Time Management Function				
99	set_tim	0	24	
100	iset_tim	0	24	
101	get_tim	0	24	
102	iget_tim	0	24	
Cyclic Handler				
103	sta_cyc	0	24	
104	ista_cyc	0	24	
105	stp_cyc	0	24	
106	istp_cyc	0	24	
107	ref_cyc	0	24	
108	iref_cyc	0	24	
Alarm Handler				
109	sta_alm	0	24	
110	ista_alm	0	24	
111	stp_alm	0	24	
112	istp_alm	0	24	
113	ref_alm	0	24	
114	iref_alm	0	24	
System State Management Function				
115	rot_rdq	0	24	
116	irot_rdq	0	24	
117	get_tid	0	24	
118	iget_tid	0	24	
119	loc_cpu	0	24	
120	iloc_cpu	0	24	
121	unl_cpu	0	24	
122	iunl_cpu	0	24	

	Service call	The use size of User stack(a)	The use size of System stack ((b),(c))	Note
123	dis_dsp	0	24	
124	ena_dsp	0	24	
125	sns_ctx	0	24	
126	sns_loc	0	24	
127	sns_dsp	0	24	
128	sns_dpn	0	24	
129	vsta_knl	0	64	After the system stack pointer is initialized, it uses it.
130	ivsta_knl	0	64	
131	vsys_dwn	0	16	
132	ivsys_dwn	0	16	
Interrupt Management Function				
133	chg_ims	0	28	
134	ichg_ims	0	16	
135	get_ims	4	0	
136	iget_ims	0	4	
137	ret_int	0	32	
System Configuration Management Function				
138	ref_ver	0	24	
139	iref_ver	0	24	
Object Reset Function				
140	vrst_dtq	0	48	
141	vrst_mbx	0	24	
142	vrst_mbf	0	48	
143	vrst_mpf	0	48	
144	vrst_mpl	0	68	

8.3 When the Kernel Library is Built

Please note that the stack consumption might change when a version and/or an optional setting of the compiler are changed and the kernel library is built.

9. Changes from Previous Version

This chapter explains changes from the previous version (V.1.00 Release 02).

9.1 Release of Restriction

The following restriction have been removed. For details of restriction, refer to Tool News.

(1) Problem in loc_mtx, tloc_mtx and chg_pri
Tool News URL : http://tool-support.renesas.com/eng/toolnews/date2011.htm#d1111101

(2)Problem in irel_mpf
Tool News URL : http://tool-support.renesas.com/eng/toolnews/date2011.htm#d1111101

(3)Problem of dispatching disabled state
Tool News URL : http://tool-support.renesas.com/eng/toolnews/date2011.htm#d1111101

(4)Problem of the GUI configurator
Tool News URL : http://tool-support.renesas.com/eng/toolnews/date2011.htm#d1111101

9.2 isus_tsk

When the RUNNING task is specified by isus_tsk and the system is in the dispatching disabled state, E_OBJ error is returned.

9.3 Correspond to Real-Time OS Aware Debugging

It became possible for OS trace to use. The OS trace is the function provided by the Real-Time OS aware debugging of the High-performance Embedded Workshop.

Note, “pragma_switch=H” is required at definition of an interrupt handler to trace an interrupt handler. For details, refer to “12(12) “8.4.15 (4) Switch passed to PRAGMA extension function (pragma_switch)”” and “12(13) “8.4.16 Fixed Vector Definition (interrupt_fvector[])””.

Refer to the following for the Real-Time OS aware debugging.

http://www.renesas.com/products/tools/os_middleware/os_aware_debugging/ecxos/osaware_hew_mid_level_landing.jsp

The RX family C/C++ Compiler Package V.1.02 Release 00 or later is required to use this function.

9.4 Guarantee of ACC Register by Interrupt Handlers

“interrupt_vector[].pragma_switch” and “interrupt_fvector[].pragma_switch” came to be able to specify the following.

- ACC : The ”acc” switch that guarantees the ACC register in the interrupt handler is passed.
- NOACC : The ”no_acc” switch that does not guarantees the ACC register in the interrupt handler is passed.

The RX family C/C++ Compiler Package V.1.01 Release 00 or later is required to use this function.

9.5 Fixed Vector Definition (interrupt_fvector[])

(1) Numeric value can be specified for "entry_address"

In the former version, only function name can be specified for "entry_address". In this version, numeric value can also be specified for "entry_address". As a result, it became easy to set register in the fixed vector area (address: 0xFFFFFF80 – 0xFFFFFFFF).

(2) When definition is omitted

- Vector 0
  In a part of MCU, the endian select register is assigned to vector 0 (address : 0xFFFFFF80). When definition of "interrupt_vector[0]" is omitted, the endian select register is set in order to endian option for compiler.
- Vector 1 to 15
  In the former version, when definition of these vector are omitted, the vector contents are the start address of the kernel's undefined interrupt processing.. In this version, the vector contents are 0xFFFFFFFF.

9.6 Addition of Timer Template Files

"rx62t.tpl", "rx62n.tpl", "rx630.tpl" and "rx210.tpl" are added. Foe details, refer to "6 Timer Template File".

9.7 Version Information

Item	Before	After
TKERNEL_MAKER, T_RVER.maker, which is returned by the ref_ver and iref_ver	0x0115	0x011B
TKERNEL_PRID, T_RVER.prid, which is returned by the ref_ver and iref_ver	0x0015	0x0003
TKERNEL_PRVER, T_RVER.prver, which is returned by the ref_ver and iref_ver	0x0102	0x0110

10. Cautions

10.1 Cautions When Using RX610 Group

The value specified as follows should be less then 8 because the PSW.IPL is configured in 3-bit widths.

- Interrupt mask specified in chg_ims and ichg_ims
- “system.system_IPL” in cfg file
- “clock.IPL” in cfg file

10.2 Cautions for High-performance Embedded Workshop

When RI600/4 project is generated by using High-performance Embedded Workshop, the following statements in the generated cfg file are always set for RX610 group. When you use MCU other than RX610 group, please change these appropriately.

- clock.template : “rx610.tpl” is set.
- clock.timer_clock : “25MHz” is set.

10.3 Cautions Concerning Address 0

Please do not put the following in address 0.

- Fixed-sized memory pool section (memorypool[].section)
- Variable-sized memory pool section (variable_memorypool[].mpl_section)
- Message sent to the mailbox

11. Restrictions

There are no restrictions for this product.

12. Changes in User's Manual

The user's manual (The document number: REJ10J2052-0100) is corrected as follows.

(1) "5.6.8 Refers to Task Status (ref_tsk, iref_tsk)"

"◆ tskpri" and "◆ tskbpri" are corrected as follows.

◆ tskpri *
Current priority of the task. If the task is in the DORMANT state, this return value is indeterminate.

◆ tskbpri *
Base priority of the task. If the task is in the DORMANT state, this return value is indeterminate

(2) "5.7.5 Suspends Task (sus_tsk, isus_tsk)" Error Code

Correct:

E_OBJ Object state error

(1) Task indicated by tskid is in the DORMANT state.
(2) Task indicated by tskid is in the RUNNING state, when isus_tsk is called in the dispatching disabled state.

Incorrect:

E_OBJ Object state error (task indicated by tskid is in the DORMANT state)

(3) "5.12.3 Refers to Mutex Status (ref_mtx)

Remove following description in "Error Code E_CTX"

~~Note : The E_CTX is not detected in the following cases.~~
~~(1) Invocation of ref_mtx from non-task context~~

(4) Table 5.19

Correct:

Table 5.19 Service Calls for Message Buffer Function

No.	Service Calls *1		...
1	snd_mbf		...
2	psnd_mbf		...
3	ipsnd_mbf		...
4	tsnd_mbf		...
5	rcv_mbf		...
6	prcv_mbf		...
7	trcv_mbf		...
8	ref_mbf		...
9	iref_mbf		...

Incorrect:

Table 5.19 Service Calls for Message Buffer Function

No.	Service Calls *1		...
1	snd_mbf	[R]	...
2	psnd_mbf	[R]	...
3	ipsnd_mbf		...
4	tsnd_mbf	[R]	...
5	rcv_mbf	[R]	...
6	prcv_mbf	[R]	...
7	trcv_mbf	[R]	...
8	ref_mbf	[R]	...
9	iref_mbf		...

(5) “5.21.1 (1) maker”

The description is corrected as follows.

Represents the manufacturer who created the kernel. For the kernel described herein, maker = 0x011B which means Renesas Electronics Corporation.

(6) “5.21.1 (2) prid”

The description is corrected as follows.

Represents the number that identifies the kernel and the type of VLSI. For the kernel described herein, 0x0003 is returned.

(7) Table 5.34

Correct:

Classification	Macro	Definition	Where	Description
...	...	...	...	...
Kernel configuration ...	VTMAX_TSK	Number of "task[]"s	kernel_id.h	Maximum task ID
	...	...	...	..
	TKERNEL_MAKER	0x011B	kernel.h	Kernel maker code
	TKERNEL_PRID	0x0003	kernel.h	Identification number of the kernel
	...	...	...	..
	TKERNEL_PRVER	0x0110	kernel.h	Version number of the kernel
	...	...	...	..
	VTMAX_AREASIZE	0x10000000	kernel.h	Maximum size of various areas
	...	...	...	..
Error codes	E_NOSPT	-9	itron.h	Unsupported function
...	...	...	...	..

Incorrect:

Classification	Macro	Definition	Where	Description
...	...	...	...	...
Kernel configuration ...	VTMAX TSK	Number of "task[]"s	kernel_id.h	Maximum task ID
	...	...	...	..
	TKERNEL_MAKER	0x0115	kernel.h	Kernel maker code
	TKERNEL_PRID	0x0015	kernel.h	Identification number of the kernel
	...	...	...	..
	TKERNEL_PRVER	0x0100	kernel.h	Version number of the kernel
	...	...	...	..
	VTMAX_AREASIZE	0x20000000	kernel.h	Maximum size of various areas
	...	...	...	..
Error codes	E_NOSPT	-11	itron.h	Unsupported function
...	...	...	...	..

(8) “6.3.3 CPU state at Start”

Correct:

Since tasks are executed in user mode, privileged instructions cannot be used. In the assembler, however, there is a helpful facility (-chkpm option) that produces a warning when privileged instructions are used.

Incorrect:

Since tasks are executed in user mode, privileged instructions cannot be used. For example, some facilities of the built-in functions supplied by the compiler, such as those provided in the header smachine.h, require caution because they use privileged instructions. In the assembler, however, there is a helpful facility (-chkpm option) that produces a warning when privileged instructions are used.

(9) “6.8 (2) Handlers”

The description is corrected as follows.

(2) Interrupt handlers

If the application contains any tasks or handlers that use the above-mentioned instructions, it is necessary that all of the interrupt handlers guarantee the ACC register. There are the following three method.

(a) Use “save_acc” compiler option.

(b) Specify “pragma_switch=ACC” for all “interrupt_vector[]” in the cfg file.

(c) Interrupt handlers explicitly guarantee the ACC register

Figure 6.7 in the user's manual shows an example of how to write a handler that guarantees the ACC register.

(10) Figure 6.7

The last line but one is corrected as follows.

Correct:

```
set_acc(&st_acc); // Restores the ACC register
```

Incorrect:

```
set_acc(st_acc); // Restores the ACC register
```

(11) “7.6 Processor Mode”

Correct:

Tasks are executed in user mode (PSW.PM=1). Handlers are executed in supervisor mode (PSW.PM=0). The CPU detects privileged instruction exception when a privileged instruction is executed in user mode. In the program executed in user mode, please do not use a privileged instruction.

The assembler supports "-chkpm" option which detects privileged instruction as warning, and uses this option if necessary.

Incorrect:

Tasks are executed in user mode (PSW.PM=1). Handlers are executed in supervisor mode (PSW.PM=0). The CPU detects privileged instruction exception when a privileged instruction is executed in user mode. In the program executed in user mode, please do not use a privileged instruction.

Privileged instructions are used in the following cases.

- Uses intrinsic functions which are provided by "smachine.h"
- Writes privileged instructions in assembly source programs

The assembler supports "-chkpm" option which detects privileged instruction as warning, and uses this option if necessary.

(12) "8.4.15 (4) Switch passed to PRAGMA extension function (pragma_switch)"

The following are added to "Definition range".

- ACC : The "acc" switch that guarantees the ACC register in the interrupt handler is passed.
- NOACC : The "no_acc" switch that does not guarantee the ACC register in the interrupt handler is passed.
- H : The interrupt handler is traced. (OS trace of Real-Time OS aware debugging)

And following description is added.

Refer to the following for the guarantee of the ACC register.

Setting of pragma_switch	"save_acc" compiler option	
	Not specified	Specified
Neither "ACC" nor "NOACC" is not specified.	Neither "acc" nor "no_acc" switch is not passed. The ACC register is not guaranteed.	Neither "acc" nor "no_acc" switch is not passed. The ACC register is guaranteed.
"ACC" is specified.	The "acc" switch is passed. The ACC register is guaranteed.	
"NOACC" is specified.	The "no_acc" switch is passed. The ACC register is not guaranteed.	

(13) "8.4.16 Fixed Vector Definition (interrupt_fvector[])"

The description is corrected as follows.

The definition item interrupt_fvector[] is used to define the interrupt handlers for fixed vectors. The causes of fixed-vector interrupts are all handled as non-kernel interrupts.

Although the causes of fixed-vector interrupts in microcomputer specifications are not assigned vector numbers, the vector addresses in the RI600/4 each are assigned a vector number, as shown in Table 8.8. Table 8.8 also shows behavior when an vector is not defined.

Table 8.8 Fixed vector number

Vector Address	Vector Number	Factor *1	Behavior when an vector is not defined
0xFFFFFF80	0	Endian select register	The following is set according to "endian" option for compiler. endian=little : 0xFFFFFFFF endian=big : 0xFFFFFFF8
0x FFFFFF84	1	(Reserved)	0xFFFFFFFF
0x FFFFFF88	2	Option function select register 1	
0x FFFFFF8C	3	Option function select register 0	
0x FFFFFF90	4	(Reserved)	
0x FFFFFF94	5	(Reserved)	
0x FFFFFF98	6	(Reserved)	
0x FFFFFF9C	7	ROM code protection (Flash memory)	
0x FFFFFFA0	8	ID code protection on connection of the on-chip debugger (Flash memory)	
0x FFFFFFA4	9		
0x FFFFFFA8	10		
0x FFFFFFAC	11		
0x FFFFFFB0	12	(Reserved)	
0x FFFFFFB4	13	(Reserved)	
0x FFFFFFB8	14	(Reserved)	
0x FFFFFFBC	15	(Reserved)	
0x FFFFFFC0	16	(Reserved)	System down
0x FFFFFFC4	17	(Reserved)	
0x FFFFFFC8	18	(Reserved)	
0x FFFFFFCC	19	(Reserved)	
0x FFFFFFD0	20	Privileged instruction exception	
0x FFFFFFD4	21	Access exception	
0x FFFFFFD8	22	(Reserved)	
0x FFFFFFDC	23	Undefined instruction exception	
0x FFFFFFE0	24	(Reserved)	
0x FFFFFFE4	25	Floating-point exception	
0x FFFFFFE8	26	(Reserved)	
0x FFFFFFEC	27	(Reserved)	
0x FFFFFFF0	28	(Reserved)	
0x FFFFFFF4	29	(Reserved)	
0x FFFFFFF8	30	Non-maskable interrupt	
0x FFFFFFFC	31	Reset	PowerON_Reset_PC()

Note: The factors are different according to MCU.

Note that the cfg600 does not generate code to initialize the interrupt control registers (e.g., IPL), the causes of interrupts, etc. for the interrupts defined here. These initialization routines need to be created in the startup file or in any way deemed appropriate for the application developed by the user.

❑Attention

Please do not define interrupt handlers for MCU's reserved vector.

❑Format

```
interrupt_fvector[(1) Vector number]{
        entry_address       = (2) Handler entry address;
        pragma_switch       = (3) Switch passed to PRAGMA extension function;
};
```

❑Contents

(1) Vector number

Description: Define the vector number of each interrupt referring to Table 8.8.
Definition format: Numeric value
Definition range: 0 - 31
When omitting: Cannot be omitted (error assumed)

(2) Handler entry address (entry_address)

Description: Define the entry address of the interrupt handler starting from which it is executed, or value for the fixed vector.
Definition format: Function name or numeric value
Definition range: 0 - 0xFFFFFFFF, when numeric value is specified
When omitting: Cannot be omitted (error assumed)

(3) Switch passed to PRAGMA extension function (pragma_switch)

Description: For the function specified by entry_address, the cfg600 outputs a #pragma interrupt directive to kernel_id.h as interrupt function. Specify the switch to be passed to this pragma directive. For details about program specifications, see the compiler's manual.
Note, #pragma interrupt is not generated, when entry_address is specified by numeric value or the vector number is 31 (reset).
Definition format: Symbol
Definition range: One of the following can be specified. To specify multiple choices, separate each with a comma.
Note, both ACC and NOACC cannot be specified at the same time.

- S: The "save" switch that limits the number of registers used by the interrupt handler is passed.
- ACC : The "acc" switch that the interrupt handler guarantees the ACC register is passed.
- NOACC : The "no_acc" switch that the interrupt handler does not guarantees the ACC register is passed.
- H : The interrupt handler is traced. (OS trace of Real-Time OS aware debugging)

Refer to the following for the guarantee of the ACC register.

Setting of pragma_switch	“save_acc” compiler option	
	Not specified	Specified
Neither “ACC” nor “NOACC” is not specified.	Neither “acc” nor “no_acc” switch is not passed. The ACC register is not guaranteed.	Neither “acc” nor “no_acc” switch is not passed. The ACC register is guaranteed.
“ACC” is specified.	The “acc” switch is passed. The ACC register is guaranteed.	
“NOACC” is specified.	The “no_acc” switch is passed. The ACC register is not guaranteed.	

When omitting: No switches are passed.

(14) “11.1 Overview”

Correct:

Table 11.1 Functions in the Sample Program

Function	Type	ID number	Task priority	Description
task1()	Task	1	1	Outputs "task1 running"
task2()	Task	2	2	Outputs "task2 running"
cyh1()	Cyclic handler	1	-	Wakes up task1

Incorrect:

Table 11.1 Functions in the Sample Program

Function	Type	ID number	Task priority	Description
main()	Task	1	1	Activates task1 and task2 and cyh1
task1()	Task	2	2	Outputs "task1 running"
task2()	Task	3	3	Outputs "task2 running"
cyh1()	Cyclic handler	1	-	Wakes up task1

Correct:

The content of processing is described below.

- The task1 operates in order to following.

Incorrect:

The content of processing is described below.

- The main task activates task1, task2, and cyh1, and then terminates itself.
- The task1 operates in order to following.

Website and Support

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

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

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. 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.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. 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.
5. When exporting the 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. You should not use Renesas Electronics products or the 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. 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.
6. 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.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. 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 categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. 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 an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
   - "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.
   - "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
   - "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. 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.
9. 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 system manufactured by you.
10. 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.
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

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

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

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

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: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

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

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, 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 Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.1