# For M16C Series, R8C Family C compiler Package \$V.5.45\$ Release 01

# Guidebook

# (Rev. 2.00)

# **Renesas Solutions Corporation**

Jun 1, 2010

# Abstract

This document provides a guide to the introduction of M16C Series, R8C Family C compiler Package V.5.45 Release 01. When you install this software package or create a project or want to know about the compiler, please refer to this guidebook.

A. Guide	book for V.5.45 Release 01	2
A.1. Poi	nts to be noted when upgrading V.5.30 Release $02$ or earlier versions of the compiler to V.5.45 $\pm$	Release 01 2
A.1.1.	Modify startup file	
A.1.2.	Change size of size_t, ptrdiff_t	
A.1.3.	Interrupt vector table	
A.1.4.	Special page table	
A.2. Poi	nts to be noted when creating a new project	
A.2.1.	Select CPU	
A.2.2.	To create a new workspace with a microcomputer that is not listed in CPU Group	
A.2.3.	When using an assembler startup	
B. AGui	de to Porting Projects Created with TM to High-performance Embedded Workshop V.4	
B.1. Sur	nmary	
B.2. Por	ting Procedure	
B.3. Usa	age Notices	
B.3.1.	TM-to-High-performance Embedded Workshop Portable and Non-Portable Information	
B.3.2.	Cross Tools	
B.3.3.	High-performance Embedded Workshop Versions	
B.3.4.	Generated Project Workspace	
B.3.5.	Load Module Converter	
B.3.6.	Other Tools	
B.3.7.	Linkage order	
B.3.8.	Placing the Start Up program at the top of Linkage Order	



# A. Guidebook for V.5.45 Release 01

This section describes the precautions to be taken when you upgrade the projects you created with old versions of the compiler to V.5.45 Release 01, and the points to be noted when you create new projects with V.5.45 Release 01.

A.1. Points to be noted when upgrading V.5.30 Release 02 or earlier versions of the compiler to V.5.45 Release 01

#### A.1.1. Modify startup file

• \_init0 function

Beginning with V.5.40 Release 00(A), the name of the library function init() has been changed to \_init().

Therefore, if you attempt to build without modification, an error message <u>*init' value is undefined*</u> may be generated during a link process. This error occurs in the following cases:

- If you created the project with an old version prior to V.5.40 Release 00, select the check box (1)
- When init function calls are enabled by altering ncrt0.a30 directly

New Project-2/5-Setting the Contents of Files to be Generated 🛛 💽 🔀	
What kind of initialization routine would you like to create?         RDM       128K         Use Standard I/O Library (UART1)         Use Heap Memory         Heap Size:       0x300         Generate main() Function         C source file       •         Use OnChip Debugging Emulator         None       •         Firmware Address:       •         Size:       •         VorkRAM Address:       •         Size:       •         Kack       Next >	(1)

If this error occurs, alter a part of ncrt0.a30 that is shown below



Before modification	
; Initialize standard I/O	
; .ifSTANDARD_IO = 	= 1
After modification	
; Initialize standard I/O	
, .ifSTANDARD_IO_ = 	= 1

[When you are using the startup file (ncrt0.a30) supplied with the compiler]

[When you are using the startup file (crt0mr.a30) supplied with the Real Time OS]





1

Ι

# For M3T-MR30

\* In several versions including the latest version (V.3.30 Release 2), the jump process shown below is 'commented out.'

Before m	odifica	tion
; Initialize	stand	ard I/O
, .{ j:	glb sr.a	_init _init
After mod	dificatio	on
; Initialize	stand	ard I/O
, .{ j;	glb sr.a	init init



# A.1.2. Change size of size\_t, ptrdiff\_t

Beginning with this version, size\_t and ptrdiff\_t have been changed in size from 16 bits to 32 bits. If you need to use size\_t and ptrdiff\_t in 16 bits because you are using a size\_t and ptrdiff\_t type based user library created with an old version, for example, make the following settings.

• Set the compile options -fsize\_t16 (-fS16) and -fptrdiff\_t (-fP16).

• Change the libraries to be linked from nc30lib.lib and r8clib.lib to nc30s16.lib and r8cs16.lib when you're using NC30WA, or from nc308lib.lib and nc382lib.lib to nc308\_16.lib and nc382\_16.lib when you're using NC308WA.

[Procedure for setup in HEW]

Set the compile options -fsize\_t16 (-fS16) and -fptrdiff\_t (-fP16).

From the Build menu of HEW, select [Renesas M16C Standard Toolchain] -> C tab.

Renesas M16C Standard Toolchain	<u>? x </u>		
Configuration : Debug All Loaded Projects Hesttest5 C source file Hester Assembly source file	C Assembly Link Librarian Lmc Ofg MKMRTBL		(1)
	Options C: -D_STACKSIZE_=0X80 -D_ISTACKSIZE_=0X80 - DVECTOR ADR=0x0fedc -D_E8D_WORK_RAM_=0x100 -c -finfo -dir *\$(CONFIGDIR)* -fS16 -fP16	`	(2)

(1)For Category, select Other

(2)For User-Defined Options, enter -fsize\_t16 (or -fS16) and -fptrdiff\_t16 (or -fP16).



# Change the libraries to be linked.

From the Build menu of HEW, select [Renesas M16C Standard Toolchain] -> Link tab.

Renesas M16C Standard Toolchain	<u>? ×</u>
Renesas M16C Standard Toolchain         Configuration :         Debug         Debug         Image: Standard Toolchain         Configuration :         Image: Standard Toolchain         Configuration :         Image: Standard Toolchain         Image: Standard Toolchain <td>? ×         embly Link Librarian Lmc Cfg MKMRTBL         Input         s For :         recifies directory of library and library         Add         Insert         Remove         recifies start address of         module :         (:         '`- G - MS - O *\$(CONFIGDIR)         TNAME):x30" - ORDER         Obbss SEdata SObss SO,data NEbss NE,data NO,b</td>	? ×         embly Link Librarian Lmc Cfg MKMRTBL         Input         s For :         recifies directory of library and library         Add         Insert         Remove         recifies start address of         module :         (:         '`- G - MS - O *\$(CONFIGDIR)         TNAME):x30" - ORDER         Obbss SEdata SObss SO,data NEbss NE,data NO,b
	 OK キャンセル

(3) For Category, select Input

For Show Entries-For select Library files.

(4) Click the Remove button to remove nc30lib.lib temporarily. Remove r8clib.lib when you're using R8C/Tiny. Remove nc308lib.lib when you're using M16C/8X Remove nc382lib.lib when you're using M32C/8X

Renesas M16C Standard Toolchain	<u>?</u> ×	
Configuration : Debug  All Loaded Projects  All Loaded Projects  C source file  Assembly source file	C Assembly Link Librarian Lmc Cfg MKMRTBL  Category: Input Show Entries For: Library files [-L -LD] Specifies directory of library and library file to be referenced: Path File Path File Down	— (5)
	□ LFEJ Spectries start address of absolute module : Options Link : □ G -MS -0 "\$(CONFIGDIR)¥\$(PROJECTNAME)x30" -ORDER ▲ data SE=0400.bss SE.data SO.bss SO.data NE.bss NE.data NO.b ss_NO.stack.istack.heap_NE.rom_NE.rom_NO.data_FE=010000.bss ▼ OK キャンセル	

(5) Click the Add button to select [Library files].



(6) Enter a library name usable for size\_t and ptrdiff\_t in 16-bit size. Input nc30s16.lib when you're using M16C
Input r8cs16.lib when you're using R8C/Tiny. Input nc308\_16.lib when you're using M16C/8X
Input nc382\_16.lib when you're using M32C/8X
Click the OK button to finish.

[When you're using the makefiles generated by the configurator of the real-time OS] Correct the following part of statements in the makefile. The following shows an example for the case where nc308\_16.lib is linked.

#### Before modification

# Use the following macro when you use C-libraries for M32C/80 series. #NEWLIB = -1 nc382lib

#### After modification

# Use the following macro when you use C-libraries for M32C/80 series. #NEWLIB = -1 nc308\_16.lib

[when you're using M3T-MR30]

Since the "LIBS" macro in the makefile is rewritten to "nc30lib.lib" by the configurator, the problem cannot be solved by correcting the "LIBS" macro.

Therefore, correct a process during "\$(LINKLIST)" generation to solve the problem.

Correct the following part of statements in the makefile.

#### Before modification

\$(LINKLST): makefile @mrecho "-o \$(PROGRAM)" \$(LINKLST) @mrecho -a "-ld \$(LIB30)" \$(LINKLST) @mrecho -a "-l \$(LIBS)" \$(LINKLST)

#### After modification

\$(LINKLST): makefile @mrecho "-o \$(PROGRAM)" \$(LINKLST) @mrecho -a "-ld \$(LIB30)" \$(LINKLST) @mrecho -a "-l nc30s16.lib -l \$(LIBS)" \$(LINKLST)



#### A.1.3. Interrupt vector table

V.5.40 Release 00 and later versions of the compiler automatically generates a vector table when an interrupt function is declared with specification of a vector number.

Thus it is no longer necessary to specify the -fmake\_vector\_table (-fMVT) option. Since the vector table is now managed in section **vector**, setting of section **\_\_NC\_rvector** is also not required.

If the "Can't generate automatically the variable interrupt vector table" error is reported at the time of linkage, modify the variable-vector-table setting as shown below.

Before modif	ication			
	if	MVT_	_==0	
,	variable	vector se	ction	
•				
		.section .org	vector,ROMDATA ; variable VECTOR_ADR	e vector table
	if	M60TYF	PE == 1	
		.lword	dummy_int	; vector 0 (BRK)
		.lword	dummy_int	; vector 1
		:		
		(omi	tted)	
		:		
		.lword	dummy_int	; vector 62
		.lword	dummy_int	; vector 63
	else;N	AVT	-	
		section	NC ryector ROMDATA	
		org	VECTOR ADR	
		.01g	VECTOR_ADIC	
	endif;	_MVT		
After modifi	cation			
		.section .org	vector,ROMDATA ; variable VECTOR_ADR	e vector table



#### A.1.4. Special page table

V.5.40 Release 00 and later versions of the compiler automatically generates a vector table when **#pragma SPECIAL** is declared with specification of a vector number.

Thus it is no longer necessary to specify the -fmake\_special\_table (-fMST) option. Since the special table is now managed in section **svector** that is automatically created at the time of assembly, setting of section **\_\_NC\_svector** and specification of a vector number with macro definition **SPECIAL** are also not required (these are now prohibited).

If the "Can't generate automatically the variable interrupt vector table" error is reported at the time of linkage, modify the special-page-table setting as shown below.

.if ;======	MST==0	
; fixed v	ector section	
, .if	SVECTOR_ADR < 0fffdcH .section svector,ROMDATA .org SVECTOR_ADR	; specialpage vector table
.endif ·		
,; special	page defination	
; ; ; ;	macro is defined in ncrt0.a30 Format: SPECIAL number	
; ; ;	SPECIAL 255 SPECIAL 254	
	(omitted)	
; ;	SPECIAL 19 SPECIAL 18	
.else .if	(SVECTOR_ADR < OfffdcH) section NC_svector BOMDATA	
.endif .endif;_	.org SVECTOR_ADR	

All of the code shown above must be deleted.



#### A.2. Points to be noted when creating a new project

## A.2.1. Select CPU

When creating a new project you can select the type of microcomputer in CPU Group. However, the selection of microcomputer types is enabled when

- ${\boldsymbol{\cdot}}$  Registration of the  ${\boldsymbol{\mathsf{sfr}}}$  header file
- Registering the variable vector interrupt table entry function registration file (intprg.c) to the workspace
- Link address settings

#### A.2.2. To create a new workspace with a microcomputer that is not listed in CPU Group

#### [In the case of R8C/Tiny series]

(1) Choose R8C/Tiny from CPU Series.

(2) Choose Other from CPU Group.

When you create a new project using V.5.45 Release 01, please check whether the compile option and the library file suit the ROM size of the microcomputer type you use on Renesas M16C Standard Toolchain Dialog Box of HEW.

From the Build menu of HEW, select [Renesas M16C Standard Toolchain]

ROM size	compile option	library
Less than 64 Kbytes	-R8C	r8clib.lib
64 Kbytes or more	-R8CE	nc30lib.lib

When you create a new project using V.5.43 Release 00 or earlier, please make a change shown below.

If the ROM space of the microcomputer type you use exceeds the 64 Kbyte boundary, make the following settings.

(3) Change the compile option from -R8C to -R8CE.

(4) Change the library to link from r8clib.lib to nc30lib.lib.



Change the compile option from -R8C to -R8CE.
---

From the Build menu of HEW, select [Renesas M16C Standard Toolchain] -> CPU tab.

Renesas M16C Standard Toolcha	in 🔹 🤋 🔀	
Configuration Debug  All Loaded Projects  Sample1  C source file  Assembly source file	Assembly Link Librarian Lmc RTOS CPU Toc  CPU Type : Create code for R8C Family (ROM >= 64KB) With no specification Create code for R8C Family (ROM < 64KB) Create code for R8C Family (ROM >= 64KB)	(1)
	OK Cancel	

(1) From the CPU Type pulldown menu, select Create code for R8C Family ( $ROM \ge 64KB$ ).

#### Change the linked library from r8clib.lib to nc30lib.lib

From the Build menu of HEW, select [Renesas M16C Standard Toolchain] -> Link tab.

Renesas M16C Standard Toolchain	
Configuration : Debug Configuration : Debug All Loaded Projects Configuration tes5 Con	C Assembly Link Librarian Lmc RTOS CPU Tc (2) Category: Frout Show Entries For: Library files [-L -LD] Specifies directory of library and library file to be referenced : Path File r8clib (3)
	C [-E] Specifies start address of absolute module : Options Link : -L "r8clib" -G -MS -O "\$(CONFIGDIR)¥\$(PROJECTNAME).x 30" -ORDER data_SE=0400 bss_SE,data_SO,bss_SO,data_NE,bss_NE,data_NO,bss_NO,stack,is マ OK キャンセル

# (2) For Category, select Input

For Show Entries-For select Library files.

(3) Click the Remove button to remove r8clib.lib temporarily.

Renesas M16C Standard Toolchain	
Configuration :       C       Assembly       Link       Librarian       Lmc       Cfg       MKMRTBL         Debug       Category :       Input       Show Entries For :       Show Entries For :       Library files       Image: Control of the property of the	
Path       File         Remove         Image: Construction of the start address of absolute module :         Options Link :         -G -MS -0 "\$(CONFIGDIR)*\$(PROJECTNAME).x30" - ORDER address of data SE=0400,bss SE,data SO,bss SO,data NE,bss NE,data NO,b ss_NO,stack,istack,heap_NE,rom_NO,data_FE=010000,bss Image: Construction of the start address of the start adddtest address of the start address of the star	(4)
OKキャンセル	

(4) Click the Add button to select [Library files].



(5)Input nc30lib.lib and click the OK button to finish.

[To create a new workspace using microcomputers other than the R8C/Tiny]

- Choose the relevant CPU series from CPU Series.
- Choose Other from CPU Group.

In addition to the above selections, the following requires caution.

• No sfr header files are registered.

Acquire the sfr header file corresponding to the microcomputer you use from the Web site, or create one if necessary.

• The order of sections is inaccurate.

Change the address of each section according to the ROM and RAM spaces of the microcomputer you use.

• No variable vector interrupt entry functions (intprg.c) are registered.

The interrupt functions should be created in a user file.

Change the section's address at linking

From the Build menu of HEW, select [Renesas M16C Standard Toolchain] -> Link tab.

(2) For Category, select Section-Order Click the Edit button

Section			
Address	Section	ок	
0×00000400	data_SE		
	bss_SE	Cancel	
	data_SO		
	bss_SO		
	data_NE		
	bss_NE	Had	
	data_NO	Modify	(4)
	DSS_NU	<u>m</u> ouny	(1)
	stack		
	ISTACK		
0-000000000		<u>R</u> emove	
0x00000000			
	data SET		
	data SOI		
	data NEI	<u>Up</u> Down	
	data NOI		
	switch table		(3)
	program		
	interrupt		
0x0000FEDC	vector		

(3) select address (4) click the Modify botton



(6)input the address



# A.2.3. When using an assembler startup

Instead of using a C language startup program, if you want to use the startups written in assembler ncrt0.a30, sect30.inc (for NC308WA, sect308.inc) or nc\_define.inc to create a project and the following applies, you need to make corrections.

• No vector numbers are specified when interrupt functions are declared.

If you build a project without making this correction, an error "Can't generate automatically the variable interrupt vector table" may be output when linking.

Conter	nt of correct	ion		
When u	using NC30	WA		
[sect30.	.inc]			
.if	MVT_	_==0		
·				
, ; variab	le vector se	ction		
;	.section .org	vector,ROMDATA VECTOR_ADR	; variable vector table	
.if 0	lword	dummy int	; vector 0 (BRK)	Deletes .if 0 to enable .lword.
	.lword	dummy_int	; vector 1	
	.lword	dummy_int	; vector 63	
.endif	•			Deletes .endit
When ı	using NC30	8WA		
[sect30	8.inc: near t	he 428th line]		
;				
; variab	ole vector se	ction		
,	.section	vector,ROMDATA	; variable vector table	)
	.org	VECTOR_ADR		
.if 0	larond	dummer int	· DDV (astructor int 0	Deletes .if 0 to enable .lword.
	.tword	dummy_int	; BKK (software lift 0	)
	.10010	aammy_mt	,	
	.lword	dummy_int	; software int 63	
.endif				Deletes .if 0 to enable .lword.



# B. A Guide to Porting Projects Created with TM to High-performance Embedded Workshop V.4

This document explains how to port projects created with TM V.2.xx or V.3.xx into High-performance Embedded Workshop V.4.

#### B.1. Summary

To port projects created using TM V.2.xx or V.3.xx into High-performance Embedded Workshop V.4, the Import Makefile function of High-performance Embedded Workshop is used. This function can create projects from such items of information as source files and build options described in the specified makefile files. In TM, project files are created in the makefile format executable in GNU make format. When project files created

with TM are selected as makefile files using High-performance Embedded Workshop Import Makefile function, they are converted to files that can run in High-performance Embedded Workshop. In addition to TM project files, the Import Makefile function can also convert files in the makefile formats for hmake, nmake, and gmake to High-performance Embedded Workshop projects.

# B.2. Porting Procedure

To port projects created using TM into High-performance Embedded Workshop, perform the following steps:

- 1. Open the File menu and select the New Workspace command.
- 2. The New Project Workspace dialog box opens.

New Project Workspace	?>
New Project Workspace Projects Projects Application Project Markefile Library	Vorkspace Name:  Project Name:  Directory:  G:\Program Files\Renesas\Hew Browse
	CPU family:         M16C         Tool chain:         Renesas M16C Standard
Properties	(6)
	OK Cancel

Figure 1 New Project Workspace Dialog Box

- Select the type of CPU used in the TM project from the Type of CPU drop-down list.
- Select the tool chain (cross tool) used for the TM project from the Toolchain drop-down list. The names of tool chains and corresponding cross tools are shown in Table 1.



Tool Chain	Cross Tool
Renesas M16C Standard	NC30WA
Renesas R8C Standard	NC8C
Renesas M32C Standard	NC308WA
Renesas M32R Standard	CC32R

Table 17	lool Chains	and Corresp	onding Cr	oss Tools
----------	-------------	-------------	-----------	-----------

- Select Import Makefile from the Project list.
- Type the directory path in the Directory text box.
- Type the workspace name in the Workspace Name text box. The same name will be automatically entered as the project name in the Project Name text box.
- Click **OK**.
- 3. You should now be able to see the New Project-1/4-Import Makefile wizard.

New Project-1/4-Import Makefile		? 🗙
(2) (1)	Makefile path :	Start (3) Add Remove
000000000000000000000000000000000000000	🔲 Show file path	
(4)	Next > Finish	Cancel

Figure 2 New Project-1/4-Import Makefile Wizard

- Select the Import options check box; this will enable information on build options (compiling and assembling options etc.) to be used to create High-performance Embedded Workshop projects. If you clear the Import options check box, the above information is neglected and not used in High-performance Embedded Workshop.
- Type the name of the TM project file (with extension .tmk) in the Makefile path text box. As soon the name is input, the specified file is analyzed, and upon analysis completion, the analyzed source files are displayed in a tree structure in the Source files box. Click the Start button to analyze the specified file again.
- If there are any errors in the analysis results (tree structure in the Source files box), rectify the tree structure with the Add and Remove buttons.
- Click Next.
- 4. Follow the instructions according to the Wizard as it continues in the procedure.



# B.3. Usage Notices

#### B.3.1. TM-to-High-performance Embedded Workshop Portable and Non-Portable Information

When you port a project created using TM into High-performance Embedded Workshop, not all the components of the project can be ported.

Portable information is as follows:

- Paths of assembler source files
- Paths of C-language source files
- Assembling options
- C-compiling options
- Linking options (except linkage order)

Non-Portable Information:

- Linkage order
- Tool configurations, dependencies, and options other than Assembler, C Compiler, Linker

To transfer these items, edit the High-performance Embedded Workshop project as described in Section 3.4 and further after processing the Import Makefile.

# B.3.2. Cross Tools

Import Makefile cannot enable all cross tool versions for use in High-performance Embedded Workshop projects regardless of whether they are used with TM or not; only the following cross tools versions are valid for High-performance Embedded Workshop projects:

NC30WA:	V.5.20 Release1 or later
NC8C :	V.5.30 Release1 or later
NC308WA :	V.5.20 Release1 or later
CC32R :	V.4.20 Release1 or later

#### B.3.3. High-performance Embedded Workshop Versions

When TM projects are ported into High-performance Embedded Workshop, information portable to High-performance Embedded Workshop varies according to the High-performance Embedded Workshop version. The information that can be ported from each cross tool to various High-performance Embedded Workshop versions are shown in Table 2.

		High-performa nce Embedded Workshop				
		~V.3.01.02	V.3.01.04	V.3.01.05	V.3.01.06	V.4.00
NC30WA	V.5.20 Release1	В	В	В	В	А
	V.5.30 Release1	В	В	В	В	А
NC8C	V.5.30 Release1	В	В	В	В	А
NC308WA	V.5.20 Release1	В	В	В	В	А
CC32R	V.4.20 Release1	В	В	В	В	А
	V.4.20 Release1A	В	В	В	В	А
	V.4.30 Release 00(A)	В	В	В	В	А

#### Table 2 Portable Information and Corresponding High-performance Embedded Workshop Versions

A: All the items of information listed in Section 3.1 are portable.

B: Only the paths of assembler and C-language source files are portable.

#### B.3.4. Generated Project Workspace

Because the project workspace created for a TM project ported to the High-performance Embedded Workshop environment is simply the contents of the makefile itself, its configuration (object output directory) will be different than that of a newly generated project workspace in High-performance Embedded Workshop.

To validate the configuration, modify the output directory file names for the compiler, assembler and linker as follows:

Output Directory (compiler, assembler):	\$(CONFIGDIR)
Output Directory (linker):	\$(CONFIGDIR)¥\$(PROJECTNAME).x30

#### B.3.5. Load Module Converter

Import Makefile cannot port the information contained in any load module converter (for example, information on options, command executions, or dependencies) into the High-performance Embedded Workshop project. If using a load module converter to create projects in TM, change the settings of the load module converter as follows after completing the Makefile processing:

1. Open the Build menu and select the Build Phases command.

2. The Build Phases dialog box will open.

Build Phases	? 🛛
Build Order Build File Order File Mappings	
Build phase order:	
M16C C Compiler	Add
I M16C Assembler I M16C Linker	Modify
(1)  ✓ M16C Load Module Converter	Remove
M16C Librarian M16C Configurator M16C Lump Lable Composer	Move Up
	Move Down
	Import
(2) <u>ok</u>	Cancel

#### Figure 3 Build Phases Dialog Box

- Select the Mxxx Load Module Converter check box from the Order of Build Phases list.
- Click **OK**.
- 3. Open the Build menu and select Renesas Mxxx Standard Toolchain.



4. The Renesas Mxxx Standard Toolchain dialog box appears.



Figure 4 Renesas M16C Standard Toolchain Dialog Box

- Click the Lmc tab.
- Select the Category type from the Category drop-down list.
- Click **OK**.



#### B.3.6. Other Tools

Import Makefile cannot port any information (options, command executions, dependencies) contained in tools other than the assembler, C compiler, and linker. If any tools other than the assembler, C compiler, linker, and load module converter are used to create projects in TM, custom build phases must be created in High-performance Embedded Workshop. Custom build phases are specifically for operating other tools before, after, or during standard builds (in the assembler, C compiler, and linker).

For more details, see Section 3.2 "Creating Custom Build Phases" in the High-performance Embedded Workshop 4 User's Manual. The following is provided as an example of how to register the cross-reference generation tool xrf30 with High-performance Embedded Workshop.

1. Open the Build menu and select the Build Phases command.

2. The Build Phases dialog box appears; click Add.

Build Phases	? 🛛
Build Order Build File Order File Mappings	
Image of the fill         Image of the fill      <	Add Modify Remove Move Up Move Down
	Import
OK	Cancel

Figure 5 Build Phases Dialog Box



3. The New Build Phase- Step 1/4 wizard opens. Follow the instructions to register the tool as follows:



Figure 6 New Build Phase- Step 1/4 Wizard

• Click **Next** (the Create a New Custom Phase check box is selected by default); the New Build Phase-2/4 Step wizard opens.



Figure 7 New Build Phase- Step 2/4 Wizard

- In this wizard, select the Multiple Phase check box.
- Select Assembly Source file from the Select input file group.
- Click **Next**; the New Build Phase- Step 3/4 wizard opens.



New Build Phase - Step 3	3 of 4	? 🗙
A REAL OF	Phase name: xrf30 5) Command (excluding parameters): [C:\Renesas\NC30WA\V530R02\bin\xrf30.ex] Default options: \$(FULLFILE) Initial directory:	Browse
PAR IL	\$(CONFIGDIR)	Browse
Contraction of the second	(6) < Back Next >	Cancel

Figure 8 New Build Phase- Step 3/4 Wizard

- Type xrf30 and its fullpath name in the Phase Name and the Command text box.
- Click **Next**; the New Build Phase- Step 4/4 wizard opens.

New Build Phase - Step	4 of 4 ? 🔀
	Does the command require any environment variables to be set? If so, enter them into the list below:
P	(7) BIN30=C:\Renesas\NC30WA\V530R02\BIN INC30=C:\Renesas\NC30WA\V530R02\INC30 LIB30=C:\Renesas\NC30WA\V530R02\INC30 PATH=C:\Renesas\NC30WA\V530R02\IB30 PATH=C:\Renesas\NC30WA\V530R02\TMP Remove
The The	(8) ▼ Read Output On Fly
Courses a	< Back Finish Cancel

Figure 9 New Build Phase- Step 4/4 Wizard

- In this wizard, enter the necessary environment variables in the list.
- Select "Read Output On Fly" check box.
- Click **Finish**.



4. You return to the Build Phases dialog box at this point, where you can see that xrf30 has been registered as a build phase at the end of the Order of Build phase order.

Build Phases	? 🛛
Build Order   Build File Order   File Mappings   Build phase order:	
<ul> <li>✓ M16C C Compiler</li> <li>✓ M16C Assembler</li> <li>✓ xrf30</li> <li>✓ M16C Linker</li> <li>✓ M16C Load Module Converter</li> <li>M16C Librarian</li> <li>M16C Configurator</li> <li>M16C Jump table Composer</li> </ul>	Add Modify Remove (2) Move Up Move Down Import
	OK Cancel

Figure 10 Build Phases Dialog Box (Build Order Tab)

- Select xrf30 from the Order of Build phase order.
- Click **Move Up** to move xrf30 next to the assembler name (see Figure 10).
- Click the Build File Order tab.

Build Phases		? 🗙
Build Order Build File Order File Mappin	gs	
File group:	Phase order:	
Assembly source file	M16C Assembler	
(4)	✓ xrf3U	
		>
1	(5)	
		ncel

Figure 11 Build Phases Dialog Box (Build File Order Tab)

- Select the xrf30 check box in the Order of Phase order.
- Click **OK**.
- 5. Open the Options menu and select the xrf30 command.

6. The xrf30 Options dialog box appears; select options as necessary. This setting executes xrf30 for all assembler source files after assemble is completed at a build (before linking files).



# B.3.7. Linkage order

Import Makefile cannot port the linking order information to High-performance Embedded Workshop. High-performance Embedded Workshop arranges the linking order alphabetically. To change this order, go through the following steps:

- 1. Open the Build menu and select the Linkage Order command.
- 2. The Linkage Order dialog box opens.

Linkage Order	? 🛛
Use custom linkage order           Object order:           nert0.r30           src0001.r30           src0002.r30           src0003.r30           src0005.r30           src0005.r30           src0007.r30           src0005.r30           src0007.r30           src0008.r30           src0009.r30           test.r30	(3) OK Cancel (2) Move up Move down
Current configuration: Debug	Copy to

Figure 12 Linkage Order Dialog Box

- Select "Use custom linkage order" check box.
- Select a file from the Object order list, and click **Move up** or **Move down** to move the file. Repeat this step for all files that need to be rearranged.
- Click **OK**.

# B.3.8. Placing the Start Up program at the top of Linkage Order

As the Import Makefile cannot port linking order information to High-performance Embedded Workshop, and links are order alphabetically, the start up program may not be placed at the top of the linking order. To place it at the top, follow the steps described previously in Section C.3.7 "Linkage Order."

