Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

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

April 1st, 2010 Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to 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.
- 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. 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; anticrime 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 majorityowned subsidiaries.
- (Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.



Peripheral Driver Generator V.1.04

Guide Book



Rev.1.00 2009.05

Notes regarding these materials

- This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
- Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
- 3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
- 4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, 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 products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
- Renesas has used reasonable care in compiling the information included in this document, but Renesas
 assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information
 included in this document.
- 6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
- 7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
- Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
 (1) artificial life support devices or systems

(2) surgical implantations

(3) healthcare intervention (e.g., excision, administration of medication, etc.)

(4) any other purposes that pose a direct threat to human life

Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.

- You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
 Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific
- 10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas 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 applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
- 11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
- 12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
- 13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.

Preface

This manual provides detailed examples of operating the Peripheral Driver Generator (hereafter referred to as PDG). For information on how to operate the PDG or the High-performance Embedded Workshop (hereafter referred to as HEW), refer to the user's manual of the PDG and HEW.

For inquiries about the contents of this document or product,

e-mail to your local distributor.

Renesas Tools Homepage

http://www.renesas.com/en/tools

Contents

1.	Over	view	1
2.	Using	the PDG	2
	2.1	Creating a Workspace with the HEW [HEW]	
	2.2	Creating a Project with the PDG [PDG]	
	2.3	Setting the Peripheral I/O Modules [PDG]	
	2.4	Registering the Output C Source files [PDG]	7
	2.5	Viewing the Created Functions [PDG]	
	2.6	Creating the Application [HEW]	10
	2.7	Compile/Link [HEW]	
	2.8	Execution [HEW]	
3.	Settir	ng a Build	18
	3.1	Specifying the Header File	
	3.2	Specifying Libraries	
	3.2.1	List of Libraries for Sample Projects	
	3.2.2	When Using HEW V.4.02 or Later	
	3.2.3	When Using Earlier Version than HEW V.4.02	
	3.3	Exculding Interrupt Vector Table	
	3.3.1	When Using H8/300H Tiny, H8S/Tiny, M16C/Tiny, and R8C/Tiny	
	3.3.2	When Using SH/Tiny	
4.	Exan	ple of Creating an Application	26
	4.1	Flow Chart of an Application to be created	
	4.2	Setting Peripherals with the PDG	
	4.2.1	Creating a Project	
	4.2.2	Setting Clocks	
	4.2.3	Setting Timer Mode	
	4.3	Creating a Program	
	4.3.1	Creating a Workspace	
	4.3.2	Creating a Program	
	4.3.3	Sample Programs	
	4.4	Build Work	
	4.4.1	Registering the Generated Filies	
	4.4.2	Setting Compile Options	
	4.4.3	Setting Link Options	
	4.4.4	Excluding Interrupt Vector Files	44
	4.4.5	Preventing Overlapping of the DTC Vector Table and Sections (for H8S/Tiny)	

1. Overview

This section gives an overview of developing applications with the PDG.

The PDG generates C source files including functions that reflect selected peripherals and settings.

Applications are developed by calling the functions generated by the PDG.

Basically, the development of an application is performed through the following steps:

(1) Setting peripheral I/O modules

Using the PDG, you will create a project, select a CPU group and peripherals, and generate necessary files.

(2) Creating an environment for build/debug

Using the HEW, you will create a workspace for developing the application. Selecting [Create a new project workspace] or other similar menu items allows you to create a workspace for developing the application.

The directory in which the PDG is installed contains sample workspaces separately for M16C/28, H8/3687, R8C/13, SH7125, and H8S/20103.

(3) Creating an application

You will call the functions generated by the PDG.

These functions must be called in the right places of the application.

For example, call the following function to initialize a timer mode with timer A0.

_CreateTimer_TA0_p1();

The header file generated simultaneously by the PDG must be included.

Name of the generated header file: <project name>.h

(4) Registering the source files generated by the PDG

Calling functions generated by the PDG is not the only step necessary for performing a build (or link) without errors.

It is required that the C source files containing the functions' bodies be registered in the workspace opened in the HEW.

(5) Build

You will register the options necessary for a build in the HEW build option. The options to be set are as follows:

- Path settings for include files with the -I compile option
- Specifying API libraries with the -L link option or other options.

These options are already set in the attached sample workspaces.

Other workspaces require these options.

These options are registered automatically when using certain version of HEW.

(6) **Debug and evaluation**

The development of the application is completed by debugging and evaluating the built application.

2. Using the PDG

This section shows a procedure for creating objects by using a PDG sample project and a HEW sample workspace. Note that the titles marked with **[PDG]** and **[HEW]** describe the operation of the PDG and HEW, respectively.

* Backup copies of the attached sample project file originals are stored in sample.bak under the PDG install directory (When recovering the files, copy the directories under sample.bak into the sample directory).

2.1 Creating a Workspace with the HEW [HEW]

In this section, you use an empty workspace for the HEW, which is included in the PDG package.

Select [Start] -> [Renesas] -> [Peripheral Driver Generator] -> [Sample Projects]. Then, open any one of the followings:

- Sample workspace for H8/3687
- Sample workspace for M16C/28
- Sample workspace for R8C/13
- Sample workspace for SH7125
- Sample workspace for H8S/20103

Here, the sample workspace for M16C/28 is used.



[1]: HEW sample workspace for M16C/28.

The HEW will be launched after [1] is selected. Proceed to the next step without closing the HEW.

Since this smaple workspace has been created with HEW V.4.00, a message dialog box asking whether to update or not will appear when the workspace is opened with a later version of the HEW. In this case, click [OK] to open it.

🏟 m16c28_sample - High-performance Embedded Workshop - [m16c28_sample.c]	×
🕗 Ble Edit Yew Brolect Bulid Debug. Setyp Iools Tegt Window Help 📃 🖻	×
□ 22 目前 32 No Ref 44 「No No N	
m16c28_sample	
B) ⊕ ⊕ m 16228, sample 1 /* C ⊆ Course He 3 /* FILT rink 500 Sample.c */ C ⊆ Course He 3 /* FILT rink 500 Sample.c */ C ⊆ Decode 3 /* PERFUTCH imain program file. */ D intro 2 6 /* CPU GBOUP 128 */ D intro 2 6 /* CPU GBOUP 128 */ D intro 2 6 /* This file is generated by Remease Project Generator (Ver.4.5). */ D intro 2 /* */ */ */ D intro 2 0 /* */ */ D intro 2 0 /* */ */ D intro 2 0 /* */ */ D intro 2 0 */ */ */ D intro 2 0 */ */ */ D intro 2 */ */ */ */ D intro 2 */ */ */ */ D intro 2 */ */ */ */ D intro 2	
22 23 23 24 24 24 25 25 25 25 25 25 25 25 25 25 25 25 25	-
	_
[#] 01 01 A1 A1 <u>81 8</u> 1 ⊘ 時 표 ?	
	< N
Neuld & Debug & Find in Files & Macro & Test & Version Control /	
Ready E 🖸 📴 🖽 Default 1 desktop Read-write 1/30 I INS CAP	

Note: H8/3687, M16C/28,R8C/13 sample workspaces are created with HEW V.4.00.03. SH7125 and H8S/20103 sample workspace are created with HEW V.4.04.01. These workspaces cannot be opened with earlier version of HEW. When opening with later version of HEW, the message box opens. Click [OK] to open the workspace.

2.2 Creating a Project with the PDG [PDG]

In this section, you use an empty project for the PDG, which is included in the PDG package.

Select [Start] -> [Renesas] -> [Peripheral Driver Generator] -> [Sample Projects]. Then, open any one of the followings:

- Sample PDG project for H8/3687
- Sample PDG project for M16C/28
- Sample PDG project for R8C/13
- Sample PDG project for SH7125
- Sample PDG project for H8S/20103

Here, the sample PDG project for M16C/28 is used.



The PDG will be launched after [1] is selected.

Peripheral Driver Generator -						
File(E) Function(U) Display(V) To	100 · · · ·					_ 8 ×
🗅 🚅 🖶 🎬 🖹 🏛 🔯 ٩	? 🚺 🌢 🐉 🎦 🔤 🖪			8 8 8	1 8 8 8 8 8 8	
	CPU: M16C/28	Item		S	Setting value	•
CPU	Ca Setting	System clock sele	ction	N	fain clock	
		Main clock			Jsed	
) main clock oscillation circu	uit 2	20.000000	
option -						
CPU setting			ip oscillator frequency			
		Periodic value				
		Divide ratio of on-				
		On-chip oscillator	frequency			
		PLL clock				
		Input frequency to				
		Selection of PLL r	nultiplier			
		PLL frequency				
		Sub clock			Not used	
			sub clock oscillation circui	t		
		Sub clock dividing				
Serial		Sub clock frequer			20.00000	
Timer		System clock div	arl clock frequency	2	20.00000	
I/O		CPU clock divisio			Divided by 1	
Interrupt			n		nvided by i	▼
	Serial Timer I/O Interrupt A/	N				
Source file name			Generated function name	Functiona	l explanetion of functions	Related item
No source is generated yet.	. The generated source file informatio	on cannot be displayed.				
Set						
Gthers -						
1.18						
						>
Other Files						
Ready						NUM
keauy						

2.3 Setting the Peripheral I/O Modules [PDG]

The first thing to do here is to set CPU clocks.

Select **CPU setting** from the peripheral I/O window.

ᆇ Peripheral Driver Generator - [m16c28_sample.pdg]								
File(E) Function(U) Display(V) Tool(T) Window(W) Help(H)		_ @ ×						
21 July 1	i 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	5 5 5 5 5						
□ I I I I I I I I I I I I I I I I I I I	Item	Setting value						
CPU Setting	System clock selection	Main clock						
	Main clock	Used						
	Input frequency to main clock oscillation circuit	20.000000						
CPU setting	🛛 🗙 On-chip oscillater clock							
Cro setting	Selection of on-chip oscillator frequency							
	Periodic value							
	Divide ratio of on-chip oscillator							
	On-chip oscillator frequency							
	PLL clock							
	Input frequency to PLL circuit							
	Selection of PLL multiplier							
	PLL frequency Sub clock	Not used						
	Sub clock	Not used						
	Sub clock dividing ratio							
	Sub clock frequency							
Serial	CPU and peripherarl clock frequency	20.000000						
Timer	System clock divider selection	20.000000						
I/O	CPU clock division	Divided by 1						
Interrupt		• mass by t						
A/D Serial Timer I/O Interrupt A	D							
Source file name	Generated function name Functio	onal explanetion of functions Related item						
		ind explanedor of randoms Related term						
No source is generated yet. The generated source file informat	Sin cannot be aspiayed.							
<								
Other Files /								
Ready								

✓ Use as peripheral function clock source aput frequency to main clock circle: 20.000000 MH;	
On chip oscillator clock	2
Use as peripheral function clock source	
Frequency selection:	
Periodic value:	
Divider selection:	
On-chip oscillator frequency: 0.000000 MH;	2
PLL clock	CPU and peripherarl clock frequency:
Use as peripheral function clock source	20.000000 MHz
Input frequency to PLL circle: 10.000000 MH;	z System clock divider selection:
Selection of multiplication:	
PLL frequency: 0.000000 MH;	Z CPU Peripheral
Sub clock	Clock divider selection: Divided by 1
Use as peripheral function clock source	
Input frequency to sub clock circle: 0.032768 MH;	z Input Clock: 20.000000 MHz
sub clock divider:	D : I I I FO 000000
Input frequency to sub clock circle: 0.032768 MH; sub clock divider:	z Input Clock: 20.000000
sub clock: 0.000000 MH;	Peripheral: 50.000000 ns

In the [CPU clock setting] dialog box, perform settings according to the program to be created and click [Modify setting] to complete the settings.

The next thing to do is to set the peripherals.

According to the program to be created, first select a peripheral and then a function of the peripheral.

For example, in order to create an asynchronous serial comunication program, follow the steps bellow:

- [1] First, select the [Serial] tab in the peripheral setup window.
- [2] Then, select [Clock asynchronous SIO mode].



In the [Clock asynchronous SIO mode setting] dialog box, specify the items (bit number, parity bit, baud rate, etc,) and select a serial port (UART0, UART1, etc,).

Check [Generate batch source (M)] and click [Setting].

C source files reflecting these settings will be generated.

* C source files cannot be generated without specifying a serial port.

* When using SH/Tiny series and H8S/Tiny series, C source files cannot be generated by above procedure. To

generate C source files, it is necessary to setup the serial ports by the following operations.

Clock asynchronous SIO mode setting							
Serial port: UARTO	•	Bit rate register					
Clock selection:	Internal clock	Bit rate register settings: 129					
External clock frequency:	10.000000 MHz	Clock source selection: f1					
Stop bit:	One stop bit	Bit Rate: 9600 bps Set details					
Parity bit:	Parity disable						
Clock polarity selection:		Interrupt enable					
Bit number:	8 bit 💌	Transmit interruption level:					
LSB first, MSB first selection:	LSB first	Permit receive interruption					
Reverse data logic:		Receive interruption level:					
CTS/RTS function:	Do not use CTS/RTS function 🔻	Permit transmit-receive interruption					
		Transmit-Receive interruption level:					
Noise canceller:		Permit SI/0 interruption					
Transmit-Receive pins sele	eot: 🖳	SI/O interruption level:					
Clock pin select:		Interrupt function name:					
Break data:		SerialNotificationFunc					
Clock port output data:							
	🔽 Ger	herate batch source(M) Setting Cancel					

When using SH/Tiny series and H8S/Tiny series, it is necessary to select pin functions to setup the peripheral I/O modules from the I/O port setting dialog box.

For example, if you setup SCI/channel 0 using SH/Tiny, the corresponding pins need to be set to SCI function by the following operations.

[1] Select the [I/O] tab in the peripheral setup window.

[2] The pins corresponding to SCI/channel 0 are as follows.

RXD0 : PA0, PA10, or PE1 TXD0 : PA1, PA11, or PE2 SCK0: PA2, PA12, or PE3. When setupping the SCL SCK and R

When setupping the SCI, SCK and RXD, or SCK and TXD need to be selected. In the example below, PA0, PA1, and PA2 are selected as RXD0, TXD0 and SCK0.

Double-click PortPA(0_7) on the peripheral setup window to open the [I/O Port setting] dialog box.



[3] Check [PA(0_7)0], [PA(0_7)1], and [PA(0_7)2] in [Terminal used]. These checkboxes corresponds to PA0, PA1, and PA2. By checking the checkboxe, setting tab for each pin appears.

ort: A(0_7)							Show pir	n function list	
Terminal used	PA(0_7)0	PA(0_7)1	PA(0_7)2 PA	(0_7)3 PA(0_	.7)4 PA(0_7)	5 PA(0_7)6	PA(0_7)7	1	
PA(0_7)0 PA(0_7)4									
PA(0_7)1		Pin fu	Inction: SCI		•	>			
PA(0_7)2		۲	Used as an inp	ut port					
RA(0_7)3 PA(0_7)7		C	Used as an ou	tput port					→
Function	PA(0_7)0	PA(0_7)1	PA(0_7)2	PA(0_7)3	PA(0_7)4	PA(0_7)5	PA(0_7)6	PA(0_7)7	
Don't care the initial output:	Г	Г			Г	Г	Γ	Г	
Upgrade drive capacity:								Г	
	Г	Г	Г	Г	Г	Г		Г	
		Г	Г	Г	Г	Г	Г	Г	

- [4] Select [SCI] as pin function in each tab.
- [5] Click [Modify setting] to generate C source files.

2.4 Registering the Output C Source files [PDG]

It is required that the C source files generated in section 2.3, Setting the Peripheral I/O Modules, be registered in the HEW.

This step is necessary for compile and link processes.

In order to register the files, select [Register file(s) in HEW project] from the PDG tool menu.

Peripheral Driver Generator - [m16c28_sample.pdg]		
File(F) Function(U) Display(V) Tool(T) Window(W) Help(H)		_ 8 ×
🗅 😅 🖬 🞇 🛅 🔯 Setting(5)		*********
Option(O)	n	Setting value
CPU Register file(s) in HEW project(R)	Bit number	8 bit
Serial Display output function list(D)	Bit rate register settings	129
Place output function list in the latest stal	La (D) K Clock polarity selection	
	(P) Reverse data logic	
- STAC	CTS/RTS function	Do not use CTS/RTS functi
Clock synchronous SIO mode	LSB first, MSB first selection	LSB first
	Parity bit	Parity disable
	Stop bit	One stop bit
Gardy Contraction of the second se	Clock selection	Internal clock
Clock asynchronous SIO mode	Clock source selection	<u>f1</u>
	Interrupt function name	-
Timer		Transmit interrput inhibit
I/O	Transmit interrupt level	0
Interrupt	Receive interrupt	Receive interrupt inhibit
A/D CPU Serial Timer I/O Interrupt A/	D	
X Source file name Generated function	on name Eunctiona	l explanetion of functions
ି :\renesas\PDG_proj\m16c28_sample\Serial\ Boolean _ Opens		ialize) the appointed serial
		appointed serial I/F
		he appointed type of notif 🛛 🖉
		he appointed type of notif
	rialInterrupt_async_U0_p1(void) Set up se	naninterrupt 🗸 🗸 –
		>
Clock asynchronous SIO mode /		
Register generated source file in HEW project		
		, , , , , , , , , , , , , , , , , , , ,

Note:

- Once the source files generated by the PDG are registered in the HEW, you cannot cancel their registration via the PDG. Cancel the registration via the HEW, if necessay.
- Make sure that the CPU group selected for the HEW workspace is the same as for the PDG project before registering the source files.

2.5 Viewing the Created Functions [PDG]

The application is created by calling the functions created in section 2.3, Setting the Peripheral I/O Modules. Available functions are listed in the source display window.

🖀 Peripheral Driver Generato	r - [m16c28_sample.pdg]	
🔛 File(E) Function(U) Display(V)	Tool(<u>T)</u> Window(<u>W</u>) Help(<u>H</u>)	_ 8 ×
D 🛎 🖬 🔛 🐮 🖿 🕅	? * ? ? ! • • • • • • • • • • • • • • • • • •	*****
CPU Serial Clock synchronous SIO mode Clock asynchronous SIO mode Clock asynchronous SIO mode Timer I/O	Clock synchronous SIO mode Clock asynchronous SIO mode Setting1 Setting1 Clock polarity selection Clock selection	Setting value ▲ 8 bit 12 129 □ Do not use CTS/RTS functi LSB first Parity disable □ One stop bit □ Internal clock f1 □ Transmit interrput inhibit □ Receive interrupt inhibit ▼
A/D	CPU Serial Timer I/O Interrupt A/D	
Source file name c:\renesas\PDG_proj\m c:\renesas\PDG_proj\m c:\renesas\PDG_proj\m c:\renesas\PDG_proj\m c:\renesas\PDG_proj\m c:\renesas\PDG_proj\m c:\renesas\PDG_proj\m c:\renesas\PDG_proj\m c:\renesas\PDG_proj\m c:\renesas\PDG_proj\m c:\renesas\PDG_proj\m	16c28_sample\Ser BooleanOpenSerialDriver_async_U0_p1(void) Open(Initialized) 16c28_sample\Ser BooleanCloseSerialDriver_async_U0_p1(void) Close the app 16c28_sample\Ser BooleanConfigSerialDriverNotify_async_U0_p1(void) Register the a 16c28_sample\Ser BooleanConfigSerialDriverNotify_async_U0_p1(void) Register the a 16c28_sample\Ser BooleanSetSerialInterrupt_async_U0_p1(void) Change serial 16c28_sample\Ser BooleanSetSerialInterrupt_async_U0_p1(void) Set up serial in 16c28_sample\Ser BooleanSetSerialInterrupt_async_U0_p1(void) Set up serial in	nterrupt
Ready		

Peripheral Driver Generator

To list output functions in CSV file format, select [Display output function list] from the tool menu. The application associated with the *.csv extension will be launched.



2.6 Creating the Application [HEW]

The application is created by using the HEW.



For example, to send data to the UARTO serial port, call the following function:

__StartSerialSending_async_U0_p1()

if(__StartSerialSending_async_U0_p1(15,(unsigned int *)"hello PDG World") == 0)
printf("False\n"); //Failed to send

2.7 Compile/Link [HEW]

Perform a build after creating the application by clicking the build button on the HEW.



Note:

- (1) When a build is performed while the HEW is installed in other than the C drive, errors will occur during the link operation. In this case, specify the directory containing the API location (lib\M16C_Tiny, lib\H8_Tiny, lib\R8C_Tiny, lib\SH_Tiny, or lib\H8S_Tiny under the PDG installation directory) by using link options. For details on how to specify the API library, refer to section 3.2, Specifying Libraries.
- (2) When the M3T-NC30WA V.5.40 Release 00 compiler package for M16C series is used, errors may occur during updating dependency information. Use version 5.42 or another method of specifying libraries.

All the development work is completed.

2.8 Execution [HEW]

This section explains how to use the simulator to evaluate the application after the development work. The following preparations for executing the application are required as explained so far:

(1) Uncomment the following lines written in the main function of the sample workspace to enable the program.

<pre>int BUFF[4] = L"PDG"; OpenSerialDriver_async_U0_p1();</pre>	//Initializes UART0
StartSerialSending_async_U0_p1(3, (unsigned int *)BUFF);	//Sends to UART0
PollingSerialSending_async_U0_p1();	//Polling processing
PollingSerialSending_async_U0_p1();	//Polling processing

- (2) Use the PDG to register the generated source files in the HEW.
- (3) Perform a build.
- Executing the simulator



- Select [SessionM16C_R8C_Simulator] for session.
 Select [Yes].

High-pe	erformance Embedded Workshop 🛛 🛛 🔀				
Session "DefaultSession" in project "m16c28_sample" has been modified Do you want to save it?					
	<u>Y</u> es <u>N</u> o Cancel				

[3] Select [OK].

Init (M16C R8C Simulator)	
MCU Debugging Information Script Trace 1/	O Script
MCU: M16c2x.mcu	lefer
Time Count Resource	
MCU Clock: 10 MHz / 1	
OK Cancel	Help
🔲 Do not show this dia	alog box again.

Peripheral Driver Generator

[4] Select [Download Modules] to download the following file.

 $C:\renesas\PDG\sample\mbox{m16c}_28\mbox{m16c}28\sample\debug\mbox{m16c}28\sample\x30$

(This file may automatically be downloaded after a build by following the steps [1] to [3].)

🏟 m16c28_sample - High-performance Emb	edded Workshop -	[resetprg.c]
A File Edit View Project Build Debug Setup	Tools Test Window	Help _ d
Debug Sessie	ons	💌 🐴 🙀 😵 🖽 📇 🖉 Debug 🔍 SessionM16C_R8C_Simule 🔍 🥀 🚳
🕀 👿 16 10 8 2 👼 💆 Debug Settin	gs	P 💷 🐛 🐛 💷 💭 🐺 🛤 🔛 🔛 🖉 🖗 🐼 🗔 🖳 🕦
Reset CPU		1.05 (25 (25 (25 (25 (25 (25 (25 (25 (25 (2
SetSe ▲≣↓ ⊆o	F5	
StartS I Reset Go	Shift+F5	// initialize heap
StartS StopS		
		p_IO// initialize standard I/O
	irsor	// Initialize Standard I/o
initset.c <u>R</u> un		T *)O; // initialize FB registe for debugger
intprg.c intprg.c imtprg.c imtfield intprg.c imtfield imtfield	Ctrl+Shift+Y	// call main routine
🔛 resetprg. 🛛 🔂 Step In	F11	// call exit
Download m 1 m16c28	F10	
Dependencie { Step Out	Shift+F11	
cregdef.ł Step		
heapdef. Step Mode	•	// infinite loop
initset.h	I	
initialize		
- 🗐 SerialFur 🔤 🦉 Connect		
sfr28 h		
Save Memor	/	
Verify Memo	y	
Download Me	dules 🕨	× 01 01 A1 A1 21 21 @ 🗈 🖬 ?
Port: UARTO Log Fi. Unload Mode		Connected
Number1		
2		
3		
4		
5		
		Userion Control / Build A Debug A Find in Files A Macro A Test A Version Control /
		III Default1 desktop Read-write 1/81 1 INS

[5] Click the reset button.

🏟 m16c28_sample - High-performance Embedded Workshop - [m16c28_sample.c]	X
Ele Edit View Broject Build Debug Setup Tools Test Window Help	F ×
🗅 😅 🖬 🕼 🕹 🎋 📴 😭 🙌 🙀 💽 🥠 🦓 💽 🦛 🚱 🏙 🎽 🆓 🕮 🛗 🖉 Debug 💽 SessionM16C_R8C_Simule 📝 🎘 🏟	
Close to the transformed processing of the transformed proces	
X Image: Connected Fort: UARTO Log File: Image: Connected 1 2 3 4 5 6 6 Image: Connected	

[6] Set a breakpoint at the place where exit(0) is called in resetprg.c.

fb = (_UINT *)0;	// initialize FB registe for debugger	
main();	// call main routine	
exit(0);	// call exit	

[7] Click the run button.

🖗 m16c28_sample - High-performance Embedded Workshop - [m16c28_sample.c]	×
🖉 Ele Edit View Project Build Debug Setup Iools Test Window Help – 🗗	×
🗋 🖸 🥔 🖉 🔏 🚱 🖻 🔒 🚱	
] ⊕ ⊠ [2 2 7 7 14] 2 (1 1 1 1 0 0 0 0 ∞ ∞ 1 1 1 1 0 0 0 0 0 ∞ ∞ 1 1 1 0 0 0 0	8
Consigned in the set of the	
X G G K K G X Port: UARTO Log File: Number+1+- 1 2 3 4 5 6 V Maro \ Test \ Version Control /	
Read-y III III III III Defaulti desktop Read-write 19/30 38 INS	

[8] Make sure that "PDG" is displayed.



The operation using the samples is completed.

3. Setting a Build

In order to perform a build/compile with the HEW, the following settings are required as well as the steps for registering the generated source files as explained in section 2.

- Specifying the directory of the reference header file (using -I option) *1
- Setting for linking API libraries (using –L option) *2
 - *1 When the High-performance Embedded Workshop V.4.05 or later is used, PDG specifies -I option automatically when PDG registers the source files. It is necessary to specify –I option manually when using High-performance Embedded Workshop V.4.04 or earlier version or adding source file to the project after source file registration from PDG.
 - *2 When the High-performance Embedded Workshop V.4.02 or earlier is used.

This section explains how to perform these settings.

Note that these are already set in the sample workspaces attached to the Peripheral Drive Generator package.

3.1 Specifying the Header File

To call the functions generated by the PDG, the header file containing the functions' prototype declarations must be included.

A header file name is <project name>.h.

When the directory created by the HEW is different from the project directory created by the PDG, the directory containing the header file, that is, the directory where the project is created, must be specified by using –I option.

```
<sample.c>

#include "m16c28_sample.h"

void main(void)

{

___OpenSerialDriver_sync_U0_p1();

___ConfigSerialDriverNotify_sync_U0_p1();

__SetSerialFormat_sync_U0_p1();

:

:

:

}
```

 From the build menu, select "Renesas M16C Standard Toolchain..." for M16C/Tiny and R8C/Tiny, select "H8S,H8/300 Standard Toolchain..." for H8/300H Tiny and H8S/Tiny, or select "SuperH RISC engine Standard Toolchain ..." for SH/Tiny.

H8S,H8/300 Standard Toolchai	n ? 🛛	
Configuration : Debug All Loaded Projects Building Constraints Constraints All Loaded Projects Constraints Constrai	C/C++ Assembly Link/Library Standard Library CPU Category: Source Show entries for: Include file directories Add Insert Remove Move gp Move gown	
	Options C/C++ : -cpu=300HN -object="\$(CONFIGDIR)\\$(FILELEAF).obj" -debug -nolist -chgincpath -nologo	

(2) Select the [C/C++] tab ([C] tab for M16C/Tiny and R8C/Tiny).

When H8/300H Tiny, H8S/Tiny, or SH/Tiny is selected, select [Include file directories] for [Show entries for:]. When M16C/Tiny or R8C/Tiny is selected, select [Inlcude file directories] for [Show Entries For:].

[1] Click [Add...].

(3) Specify the directories containing include files.

Add include file directory		? 🗙
<u>R</u> elative to :		OK
Custom directory	-	Cancel
Directory :		
		<u>B</u> rowse

In the [Add include file directory] dialog box, select the relative path category and enter the directory name. For H8/300H Tiny, H8S/Tiny, or SH/Tiny, select and enter the followings:

- Relative to: HEW installation directory
- Sub-Directory: Directory name

For M16C/Tiny or R8C/Tiny, select and enter the followings:

- Relative to: Custom directory
- Directory: Directory name
- (4) Close all the dialog boxes to complete the settings.

3.2 Specifying Libraries

3.2.1 List of Libraries for Sample Projects

When a setting source of each peripheral I/O module is built, the libraries shown below must be linked. For the names of all library files of supported microcomputers, refer to the PDG's users manual.

CPU	Directory	Library File Name
H8/3687	lib\H8_Tiny	rapi_h8_3687.lib
R8C/13	lib\R8C_Tiny	rapi_r8c_13.lib
M16C/28	lib\M16C_Tiny	rapi_m16c_28.lib
SH7125	lib¥SH_Tiny	rapi_sh7125.lib
H8S/20103	lib¥H8S_Tiny	rapi_h8s_20103.lib

3.2.2 When Using HEW V.4.02 or Later

When the source is registered in the PDG while HEW V.4.02 or later is used, the following dialog box is displayed.



In this dialog box, determine the priorities of the libraries.

When the same symbol name exists in different libraries, the library that has the highest priority among them are selected.

When changing the priorities, select a library and click the buttons marked with [1]. A library with higher priority is listed higher.

3.2.3 When Using Earlier Version than HEW V.4.02

- When using an earlier HEW version than 4.02, follow the steps below to specify libraries.
 - From the build menu, select [Renesas M16C Standard Toolchain...] for M16C/Tiny and R8C/Tiny, select [H8S,H8/300 Standard Toolchain...] for H8/300H Tiny and H8S/Tiny, or select [SuperH RISC engine Standard Toolchain ...] for SH/Tiny.



(2) Select the [Link] tab in the [Renesas M16C Standard Toolchain] dialog box, or select the [Optimization linker] tab in the [H8S,H8/300 Standard Toolchain] dialog box or [SuperH RISC engine Standard Toolchain ...] dialog box.

Renesas M16C Standard Toolch	ain 🤗 🔀
Configuration :	C Assembly Link Ubrarian Lmc RTOS C
Debug 💌	Category : Input
	Show Entries Ear:
	[-L-LD] Specifies directory of library and library file to be referenced :
	Path File
	Lenore Lenore Lenore Lenore Lenore
	□ [-E] Specifies start <u>a</u> ddress of absolute module :
	Options Link :
×	L "r8clib" -G -MS -O "\$(CONFIGDIR)\\$(PROJECTNAME).x30" -ORDER data_SE=0400,bss_SE,data_S0,bss_S0,data_NE,bss_N
	OK Cancel

Select [Library files] ([Library files] for H8/300H Tiny and H8S/Tiny) for [Show Entries For] and click [Add].

(3) Specify the directory containing the API libraries.

🖗 r8c_13 - High-performance Embedded Workshop			🗖 🗗 🗖
File Edit View Project Build Debug Setup Tools Test Window Help			
	🏨 🥁 🏠 🛛 🗇 🎬 📇 👗 🗍 Debug	DefaultSession	- 2 🖗
Remesas M16C Standard Toold Create line: Create line: Create line: Create line: Create line: Create line: Denose inalDrive: asynt Create line: Denose inalDrive: asynt Disable line: Denose inalDrive: asynt Denose inalDrive: a	C Assembly Link Librarian Lmc RTO Category: Input Show Entries For : Library files {-1D) Specifies directory of [brary and [brary life to be referenced : Path File Inse	dd	[1
	Default1 desktop		INS
	Inter and the personal description	J	140

[1] Select [Library directory] (for R8C/Tiny, M16C/Tiny, and M16C/60).

Set library file directory	? 🔀
Relative to :	ОК
Directory :	Cancel
	<u>B</u> rowse

Relative to: Select [Custom directory].

Directory: Specify the directory containing the API libraries.

Example) C:\Renesas\PDG\lib\M16C_28

For H8/300H Tiny, H8S/Tiny, or SH/Tiny, specify the directory name and liblary file name.

(4) Specify the library name (For R8C/Tiny, M16C/Tiny, and M16C/60).



[2] Select [Library file].



- [3] Enter the name of the peripheral API library file to be used. Example) rapi_m16c28 (Do not add the .lib extension.)
- (5) Close all the dialog boxes to complete the settings.

3.3 Exculding Interrupt Vector Table

The PDG creates an interrupt vector simultaniously.

Since another interrupt vector is created when a workspace and a startup file are created at the same time by the HEW, the interrupt vector for the startup needs to be excluded from the compile target.

Note that when HEW V.4.02 or later is used, the interrupt vector is automatically excluded in the PDG. The method to use interrupt vectors is as follows.

3.3.1 When Using H8/300H Tiny, H8S/Tiny, M16C/Tiny, and R8C/Tiny

The startup program of HEW includes the source file intprg.c that declares interrupt vector functions. The source file created by PDG also includes interrupt vector functions. To avoid duplication of interrupt functions, intprg.c is excluded from the project of HEW when PDG registers the source files. (Please exclude it manually when using HEW V.4.02 or earlier version.) If you want to add interrupt operation, please add another file that includes interrupt function. When using M16C, if transmit interrupt of SCI/UART0 is enabled both intprg.c and m16c_28.c created by PDG has interrupt function of vector 17. Therefore intprg.c is excluded.

For example, if you want to add DMA0 interrupt operation, please add file of DMA0 interrupt function.



3.3.2 When Using SH/Tiny

The startup program of HEW includes the source file intprg.c that declares interrupt vector functions. The source files intprg_sh7125_pdg.c and intprg_sh7125.c created by PDG also includes interrupt vector functions. To avoid duplication of interrupt functions, intprg.c is excluded from the project of HEW when PDG registers the source files. (Please exclude it manually when using HEW V.4.02 or earlier version.)

In intprg_sh7125_pdg.c, the interrupt functions PDG can handle are collected. This file is created and edited only by PDG. PDG overwrite this file every code generation.

In intprg_sh7125.c, the interrupt functions PDG doesn't handle are collected. User can edit this file because PDG doesn't overwrite this file.



Note : When using SH7125 with HEW V.4.04 or earlier version, it is necessary to register the include file path of vect.h that is startup program of HEW manually for intprg_sh7125.c and intprg_sh7125_pdg.c. When using HEW V.4.05 or later, the include path is registered automatically.

4. Example of Creating an Application

This section describes a procedure for creating an application with the PDG.

4.1 Flow Chart of an Application to be created

Create an application based on the flowchart below. Microcomputer used for the application is M16C/28.



4.2 Setting Peripherals with the PDG

4.2.1 Creating a Project

Select [File] -> [Create New Project] to open the new project window.



[1] Enter a project name.

Here, "M16C28" is chosen.

Note: The Peripheral Driver Generator creates a header file with "<project name>.h". If an existing file has the same name, enter another project name.

[2] Specify a directory to store the project file in.

By default, a directory with the same name as the project is created under c:\renesas\PDG_proj.

- [3] Select a CPU.
 Here, select the followings: Series: M16C/Tiny
 Group: M16C/28
 Type No: M30280F6
- [4] Click the [OK] button to complete creating a project.

4.2.2 Setting Clocks

The [CPU clock setting] window appears after the steps for creating a project in section 4.2.1 is completed.

CPU clock setting		
System clock selection: Pll clock		→ [1]
Main clock Use as peripheral function clock source Input frequency to main clock circle: 20.000000 MHz		
On chip oscillator clock		
Use as peripheral function clock source Frequency selection:		
Periodic value: 0		
On-chip oscillator frequency: 0.000000 MHz		
PLL clock Use as peripheral function clock source	CPU and peripherarl clock frequency:	
Input frequency to PLL circle: 10.000000 MHz	System clock divider selection:	▶ [2]
Selection of multiplication: Multiplied by 2 PLL frequency: 20.000000	CPU Peripheral	→ [3]
Sub clock	Clock divider selection: Divided by 1	• [5]
Input frequency to sub clock circle: 0.032768 MHz	Input Clock: 20.000000 MHz	→ [4]
sub clock divider: sub clock: 0.000000 MHz	Peripheral: 50.000000 ns	, L.J
	Modify setting Cancel	▶ [6]

- [1] Select [PLL clock] in the [System clock selection].
- [2] Specify the frequency as 10 MHz.
- [3] Select [Multiplied by 2] for [Selection of multiplication].
- [4] Do not use sub-clock as peripherarl clock.
- [5] Select [Devided by 1] for CPU clock divider.
- [6] Click [Modify setting] to complete the settings.

4.2.3 Setting Timer Mode



- [1] Select the [Timer] tab.
- [2] Select and click [Timer mode].

		Timer mode setting		×
[1] <	(Timer type: A0	Operation during initialization: Operation start	[4]
	Ì	Operation:	Counter clear function:	3
[2] <	(Count source: f8	Timer output: Pulse is not output] → [5]
	Ì	Frequency of count source: 2.500000 MHz	Clock output function:	1
		Clock edge:	Auto reload function:	1
[2]	ſ	Period: 1000.000000 µs	Control to write to timer:	
[3] <			Gate function: Do not use gate function	[6]
		Result of calculation	A/D Converter Start:	-
		The values in this frame are set.		
		Period: 1000.000000 µs	Enable Underflow interruption	
		(Error: 0.000000 %)	Interruption priority level:	
		Setting value:	Interruption function name: TimerIntFunc	
		Timer setting value 2499		
			Generate batch source(M)	1
			[7] [8]	

- [1] Select [A0] as the timer.
- [2] Select [f8] for the internal count source.
- [3] Specify the period as 1000 micro sec.
- [4] Select [Operation start] for the operation after initialzation.
- [5] Select [No pulse is output] for the timer output.
- [6] Select [Do not use gate function] for the gate function.
- [7] Check [Generate batch source] to generate sources reflecting these settings.
- [8] Click [Setting] to complete the settings.


After the settings of the timer mode are completed, the settings are displayed in [1], and the generated source files, generated function names, and functional explanations are listed in [2].

4.3 Creating a Program

A program is created with the HEW.

4.3.1 Creating a Workspace



[1] In the [Welcome!] dialog box, select [Create a new project workspace] and click [OK].

New Project Workspace	? 🗵	
Projects		
Project Types	Workspace Name: m16c_timer	• [2]
C source startup Application	Project Name: m16c_timer	
Library Debugger only - M16C Simulat	Directory: C:\WorkSpace\m16c_timer Browse	[3]
	CPU family: M16C	► [4]
	Tool chain: Renesas M16C Standard	▶ [5]
Properties		
	OK Cancel	

[2] Select [C source startup Application].

(When using an assembler startup, select [Application].)

[3] Enter a workspace name.

Here, [m16c_timer] is selected.

- [4] Select [M16C] for the CPU family.
- [5] Select [Renesas M16C Standard] for the tool chain.

Click the [OK] button.



[6] Specify the compiler version as 5.40.00 or later.

- [7] Select [M16C/Tiny] for [CPU Series:].
- [8] Select [28] for [CPU Group].

Click the [Next] button to proceed.

Operate the new project creation wizard to compelet creating the new workspace.

4.3.2 Creating a Program

Code a program on [1] of the HEW.

@m16c_timer - High-perform					
A File Edit View Project Build	<u>D</u> ebug Setyp <u>T</u> ools Te <u>s</u> t	Window Help		_ 🗗 🗙	
	8 😣 🙀	_ M & A S ∰ ∰ ∐ ▲ [Di	ebug 👤 DefaultSession	- 2 %	
The section of	6 /* CPU GR 7 /* 8 /* This f	:m16c_timer.c :Fri, Nov 16, 2007 PTION :main program file. OUP :28 ile is generated by Renesas Project			→ [1]
Opendencies	9 /* 10 /******* 11 12 void main(13 (14 (15) 16	void)	*/	2	
	✓ m16c_timer.c				
N 01 01 AL AT 121 21 0	🖻 🖬 ?				
Build (Debug) Find in	n Files À Macro À Test À Ver	sion Control /	dealthen Deard units 1/1/ 1	TAIC	

When coding a program, use the functions created with the PDG.



The generated functions can be viewed as Excel data as well as in the PDG's graphical user interface.

By clicking the button, Excel is launched and the functions are listed.

Note that since the extension of the output files is .csv, *.csv files must be associated with Excel. (Otherwise, the files are displayed as text files.)

Microsoft Excel -		ols Data Window Help Acrobat			_ _ _ 8
			In		
D 🛩 🖬 🔒 (é	🗟 💐 🐰 🛛	🖻 💼 🝼 🖙 - 🖙 - 🍓 S 🍂 🛃 🚺 👫 100% 🗉 🕄 🗸	Arial • 10 • B I	🏿 📑 🗃 🗃 💲 % , 🐭 🗊 🖃 -	🔌 - 🗛 -
A25 🔽	=				
Α	В	C	D	E	F
1 Project name	M16C28				
2					
}					
Timer	0. W	0 0			D 1 1 1
Mode	Setting name	Source file name	Generated function name	Functional explanation of functions	Related it Gate func
i Timer mode	Setting1	c:\renesas\PDG_proj\M16C28\Timer\CreateTimer_TAD_p1.c	Boolean CreateTimer_TAD_p1(void)	Create timer mode setting	Gate func
3		c:\renesas\PDG_proj\M16C28\Timer_EnableTimer_TAD_p1.c	BooleanEnableTimer_TAD_p1(void)	Timer mode operation control (Operation start)	
		c:\renesas\PDG_proj\M16C28\Timer_DisableTimer_TAD_p1.c	BooleanDisableTimer_TAD_p1(void)	Timer mode operation control (Operation stop)	
		c:\renesas\PDG_proj\M16C28\Timer_DestroyTimer_TA0_p1.c	BooleanDestroyTimer_TA0_p1(void)	Destroy timer mode	
D		c:\renesas\PDG_proj\M16C28\Timer\TimerFuncs_Timer_TAD_p1.h		Prototype declaration file of generated function	
1					
3					
4					
5					
5					
7					
3					
9 Others					
J Mode	Setting name	Source file name	Generated function name	Functional explanetion of functions	Related it
1 -	Setting1	c:\renesas\PDG proj\M16C28\M16C28.h	Cenerated Idiction name	Header file for all output source files	rtelated it
2	Country	o. sonosso so_projasnoozoanoozo.n		riedeer me for an output source mes	
3					
4					
5	1				
6	•				
M16C28	/				•

The figure above is the displayed Excel data.

4.3.3 Sample Programs

The header file and the functions shown in red are generated by the PDG.

```
#include "sfr28.h"
#include "m16c28.h"
                                  // When the functions generated by the Peripheral Driver Generator are used,
                                  // this header file must be included.
#define _1SCNT (1000/10)
#define PLL_WAIT_1MS 10000 /* 1msec @10MHz */
                                  /* 20msec */
#define PLL_WAIT_CNT 20
void main(void)
{
       int
             counter = _1SCNT;
       int
             onoff = 1;
       unsigned int i,j;
       /* PLL clock setting */
       prcr = 0x01;
                                  /* protect register off */
       cm2 = 0x00;
                                  /* system register2 Initialize */
       cm07 = 0;
       cm1 \&= 0x3f;
       cm06 = 0;
                                  /* 2 multiplying */
       plc0 = 0x11;
                                  /* 2 wait */
       pm20 = 0;
                                  /* PLL operation */
       plc07 = 1;
       for (i = 0; i < PLL_WAIT_CNT; i++) {
                                                      /* about 20ms wait */
             for (j = 0; j < PLL_WAIT_1MS; j++) { /* Main clock 10MHz */
              }
       }
       cm11 = 1;
                                  /* protect register on */
       prcr = 0x00;
       prcr = 0x04;
                                  /* protect register off */
                                  /* 80pin type */
       pacr = 0x03;
       prcr = 0x00;
                                  /* protect register on */
\downarrow
[1]
```

```
[1]
\downarrow
      p0 = 0xff;
      p1 = 0xff;
      pd0 = 0xff;
      pd1 = 0xff;
      if( __CreateTimer_TA0_p1() == RAPI_TRUE ) /* timer setting */
      {
             if( __EnableTimer_TA0_p1( ) == RAPI_TRUE ) /* timer start */
             {
                   while(1)
                   {
                          while( (talic & 0x08) == 0); /* 10ms? */
                          ir_ta_1ic = 0;
                          counter--;
                          if( counter == 0 )
                                                         /* 1s? */
                          {
                                p1 = 0xfe;
                                if( onoff )
                                       p0 = 0xf9; /* LED1 on */
                                else
                                       p0 = 0xff;
                                                    /* LED1 off */
                                onoff ^= 1;
                                counter = _1SCNT; /* counter reset */
                          }
                   }
             }
      }
      return;
}
/* end */
```

4.4 Build Work

In order to perform a compile/link after completing coding a program, the following steps are required:

- Registering the files generated by the Peripheral Driver Generator in the High-performance Embedded Workshop.
- Setting the options necessary for a compile (specifying the header file include destination)
- Setting the options necessary for a link (specifying Renesas Embedded Library)

4.4.1 **Registering the Generated Filies**

You will register the files generated by the Peripheral Driver Generator in a workspace of the High-performance Embedded Workshop.



Select [Tool] -> [Register file(s) in HEW project].

PDG	\mathbf{X}
When the work space is being opened with HEW, the source file thus generat When no work space is opened with HEW, the generated source file is registe space was selected. When several work spaces are being opened, note that a file is registered in al Do you really want to start source file registration?	red in the active project after the work

The dialog box will appear asking whether to register the files or not. Select [Yes].

The files will be registered in the project currently opend by the HEW.

When registering them in another project, close the currently opend project.

When you attempt to register them while no workspaces are opend, the dialog box appears prompting you to select a workspace for the registration. Select one in the dialog box.

Open			? 🗙
Look in: 🔎	m16c_timer	- 🗧 🔁	r 📰 🕈
Commentation in the second sec	J		
File <u>n</u> ame:			<u>O</u> pen
Files of <u>type</u> :	HEW Workspaces(*.hws)	•	Cancel

When HEW V.4.02 or later is used, the [Library link priority setup] dialog box appears. Move the libraries up and down in the dialog box, according to their priorities. When [OK] is clicked, the files begin to be registered in the HEW project that you selected. *

* When several HEW workspaces are opened, files are registered in all active projects, as stated in the dialog box that asks whether to register the files. Close workspaces that you do not register the files in before performing registration.

Peripheral Driver Generator

Library li	ink priority setup	
Set the pri	ority in which order libraries are linked.	
Priority high	nc30lib.lib C:\renesas\PDG\lib\M16C_Tiny\rapi_m16c_28.lib	
		Up
		5
		F
Priority		Dowr
low		
	ОК	Cancel

The message dialog box appears telling you that the registration is completed.

PDG	
⚠	The source file has completely been registered.
	OK

4.4.2 Setting Compile Options

In order to call the functions generated by the PDG, their header file must be included.

#include "sfr28.h"	
#include "m16c28.h"	<pre>//When the functions generated by the Peripheral Driver Generator are used, //this header file must be included.</pre>
#define _1SCNT (1000/10)	

Compiling source files that includes this header file requires to specify its directory. The method to specify the include file directory is as follows.

When setting H8/300H Tiny, H8S/Tiny, or SH/Tiny with HEW V.4.05 or later, this operation is not necessary for the source file registered by PDG or the file already registered because this option is specified automatically with source registration.



Select [Renesas M16C Standard Toolchain...] from the HEW build menu.

Renesas M16C Standard Toolch	ain 🤶 🔀	
Configuration : Debug Configuration : Debug All Loaded Projects Configuration All Loaded Projects Configuration All Loaded Projects Assembly source file Assembly source file	C Assembly Link Librarian Lmc RTOS C Category: Source Show Entries For: Include file directories [4] Specifies the girectory containing the file(s) specified in tinclude. You can specify up to 8 directories: Add Insert Remove D Owr	→ [1]
	Options C : -c -finfo -dir ''\$(CONFIGDIR)''	

Select the [C] tab in the option dialog box and select the followings:

Category: Source

Show Entries For: Include file directories

Click the [Add] button marked with [1].

Set include file directory	? 🗙
<u>R</u> elative to :	ОК
Custom directory	Cancel
Directory :	
C:\renesas\PDG_proj\M16C28	Blowse

Specify the directory that contains the header file generated by the PDG. This is a project directory created by the PDG.

Click the [OK] button to complete the settings.

Note : When setting M16C/Tiny or R8C/Tiny with HEW V.4.05 or later, the message "Failed to set the include file option." appears depending on the toolchain version. If this message appears, specify the include file directory by the above procedure.

4.4.3 Setting Link Options

When HEW V.4.02 or later is used, it is not required to specify libraries.

(1) Select [Renesas M16C Standard Toolchain...] from the build menu.

Renesas M16C Standard Toolch	ain 🦳 🔀	
Configuration :	C Assembly Link Librarian Lmc RTOS C	
Debug	Category : Input Show Entries For : Library files [-L -LD] Specifies directory of library and library file to be referenced : Path File Nnc30lib Bernove Down	→ [1]
	[-E] Specifies start address of absolute module : Options Link : -L "nc30lib" -G -MS -0 "'\$(CONFIGDIR)\\$(PROJECTNAME).x30" -ORDER data_SE=0400,bss_SE,data_SO,bss_SO,data_NE,bss_N OK	

Select the [Link] tab in the option dialog box and select the followings: Category: Input Show Entries For: Library files

Click the [Add] button marked with [1].

(2) Specify the directory that contains the API library.

🖗 m16c_timer - High-performance Em	nbedded Workshop - [m16c_timer.c]	
🐲 File Edit View Project Build Debug S	Setup Tools Test Window Help	
🗅 😅 🖬 🕼 🔏 🙏 🖻 💼 😝	🙀 🔄 🖄 🖇 🕮 🖽 Ă Debug 🔽 DefaultSession 💌 🥕 📯	
X		
	S. Source /* /* /* FILE :m16c_timer.c */ Renesas M16C Standard Toolchain Configuration: C Assembly Link Librarian Lmc RTOS C * */ Configuration: C Assembly Link Librarian Lmc RTOS C * */ Category: hput * Show Entries For: Library file C Assembly source file R C sour	→ [1]
. .	L 'ma30B'' G MS.O. L'ma30B'' G MS.O. StormFollPargProJeCTNAMELx30".0PDER dela_SE=0400.bss_SE.dela_S0.bss_S0.dela_NE.bss,N	
× 01 01 AL AT 21 21 0 Pb E	0K Cancel	
	Morro à Tatt à Varrier Centrel /	
Build (Debug) Find in Files) 1	Macro λ Test λ Version Control /	
Ready	🔝 🔝 🛃 Default1 desktop Read-write 16/16 1 INS	

Select [Library directory] from [1].

Set library file directory	? 🔀
<u>R</u> elative to :	ОК
Custom directory	Cancel
Directory :	
C:\renesas\PDG\lib\M16C_Tiny	<u>B</u> rowse

Relative to: Select [Custom directory].

Directory: Specify the directory that contains the API library: C:\Renesas\PDG\lib\m16c_tiny

After entering the directory name, click [OK].

(3) Specify the API library name.

Click [Add] again and select [Library file] in [1].

Set library file	? 🛛
Library name :	ОК
rapi_m16c_28	Cancel

Enter the library name, "rapi_m16c_28." (Do not add the.lib extension.)

4.4.4 Excluding Interrupt Vector Files

Creating a workspace with the HEW registers intprg.c, which is one of several startup files. (When the H8 300H/Tiny is used, the workspace creation wizard asks whether to create intprg.c or not.)

Since the PDG creates interrupt vector functions, intprg.c is redundant for vectors. Therefore, intprg.c needs to be excluded from the build target.

Note that when HEW V.4.02 or later is used, the interrupt vector functions are automatically excluded from the PDG.

R m16c_timer - High-performance Embedded Wor	shop - [m16c_timer.c]	- 7 🗙
🌮 File Edit View Project Build Debug Setup Tools	iest Window Help	_ 8 ×
	▼ 帧 系 品 🛛 🕸 🖽 📇 Debug 💽 DefaultSession 💌	2: 🛛
Line S. Source		
- 🔂 m16c_timer 25 /*	PLL clock setting */	
	r = OxO1; /* protect register off */	
	: = 0x00; /* system register2 Initialize */ 17 = 0;	
	, = 0, . &= 0x3f;	
	16 = 0;	
— ≚ _DestroyTi 31 plo	0 = 0x11; /* 2 multiplying */	
	0 = 0; /* 2 wait */	
— ≝EnableTir 33 plo	:07 = 1; /* PLL operation */	
	: (i = 0; i < PLL WAIT CNT; i++) { /* about 20ms wait */	
⊥ initset c 36	for (j = 0; j < PLL WAIT 1MS; j++) { /* Main clock 10MHz */	
intprg.c 37		
- 실 finetoric 34 - 실 finetoric 35 for - 실 initistic 35 for - 실 initistic 36 - 실 initistic 37 - 실 mificio 77		
i resetp		
	r = 0x00; /* protect register on */	
Build Options	r = 0x04; /* protect register off */	
and the second second	r = 0x03; /* 80pin type */	
resetp	r = OxOO; /* protect register on */	
sfr28.t	= Oxff;	
Foreing F	= Oxff; = Oxff;	-1
		<u> </u>
Configure View		
Show Differences		
✓ Allow Docking		
Hide		
Properties		
Properties	1	
		4.0
		>
Build (Debug) Find in Files) Macro) Test	Version Control /	
Ready	[] [] [] Default1 desktop Read-write 74/74 1 INS	j [

Select intprg.c and right-click on it.

The pop-up menu opens. Select [Exclude Build intprg.c].

4.4.5 Preventing Overlapping of the DTC Vector Table and Sections (for H8S/Tiny)

When you use the source code generated through the PDG to operate the DTC, the area from 400 to 4DD (at maximum) will be used as the DTC vector table. If any section overlaps with the DTC vector table, the address of the section must be changed.

Select [Build -> H8S,H8/300 Standard Toolchain...] to open the build options dialog box. Then select [Section] for [Category] on the [Link/Library] page. A list of sections is displayed. Change the addresses of the sections as required.

H8S ,H8/300 Standard Toolchain				
Configuration : Debug All Loaded Projects All Loaded Projects C source file C source file C ++ source file Linkage symbol file	C/C++ Assembly Link/Library Standard Library CPU Category: Section Show entries for: Section Address Section Address Section Address Address Section Add Address Address Add Address			
	Options Link/Library : -noprelink -rom=D=R -nomessage -list="\$(CONFIGDIR)\\$(PROJECTNAME).map" -nooptimize OK Cancel			

All the settings are completed.

Perfom a build and execute the application by using the debugge

Peripheral Driver Generator V.1.04 Guide Book

Published by:	Sales Strategic Planning Div.			
Edited by:	Renesas Technology Corp. Microcomputer Tool Development Department Tool Business Division			
	Renesas Solutior	ns Corp.		

 $\ensuremath{\textcircled{O}}$ 2009. Renesas Technology Corp. and Renesas Solutions Corp., All rights reserved. Printed in Japan.

Peripheral Driver Generator V.1.04 Guide Book



Renesas Electronics Corporation 1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan