To our customers,

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.

RENESAS

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

HEW

Effective Usage of HEW – Profile and Performance Analysis (PPAnalysis)

Introduction

This application note provides:

1. A general view on different usage of Performance Analysis, Profile (List Sheet, Tree Sheet) and Profile-Chart
2. A detail explanation on how to use Performance Analysis, Profile (List Sheet), Profile (Tree Sheet) and Profile-Chart
3. How it would help or benefit users when developing their codes

Two groups of window display will be introduce, namely Performance analysis and Profile. Performance analysis displays the number of execution cycle required for the specified functions. Profile (List Sheet, Tree Sheet) and Profile-Chart mainly display profile data that are relation of function calls.

Target Device

All

Contents

1. Overview

Performance Analysis window displays the number of execution cycles required and count for the specified functions.

Profile (List Sheet) window displays functions or global variables' address, size, number of times a function is called or a global variable is accessed and number of internal/external or I/O memory access.

Profile (Tree Sheet) window displays the relation of function calls in a tree structure that includes contents of address, size, stack size, number of function calls and profile data.

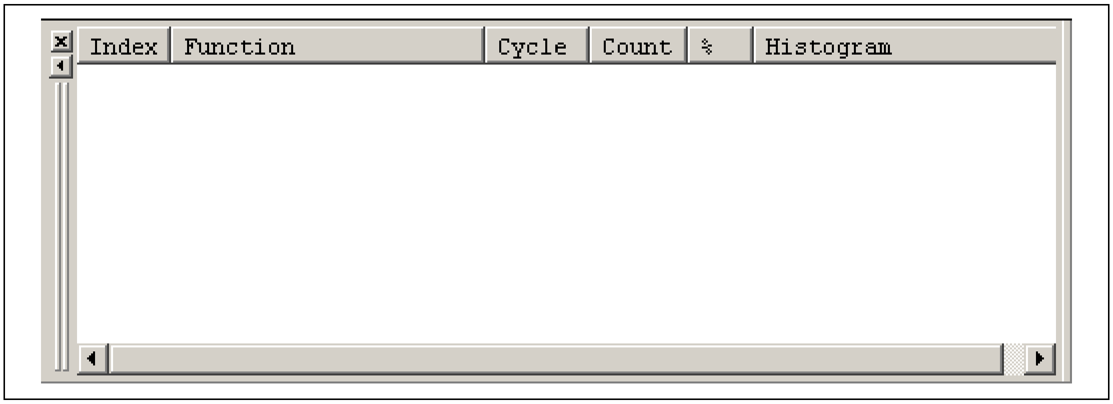
Profile-Chart window displays the relation of calls for a specific function.

These functions can be found in HEW2-Simulator and some series of Emulator.

2. How to utilize Performance Analysis function

Using a sample program given in section 6 Sample Code to demonstrate the advantage of utilizing Performance Analysis function when developing codes. Aim of this sample program is to show the difference in performance when sum of 2 to the power of 10 (2^0+2^1+2^2 …+2^{10}) being loop for 50 times are written in various ways.

With the above codes built in an application project in HEW2 and set the Debug Setting Target to Simulator mode. Select [View->Performance->Performance Analysis], Shift+Ctrl+P or icon to open to display shown below in figure 1.

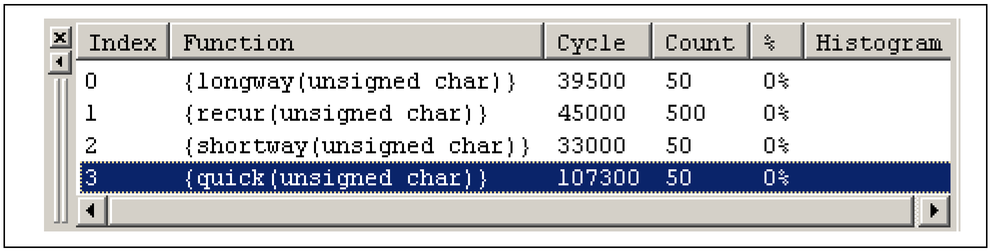


Index	Function	Cycle	Count	%	Histogram

Figure 1 Performance Analysis Window

2.1 Steps to follow for executing Performance Analysis function

1. Right-clicking within the view area, a popup menu contains options where user can add, edit, delete functions or reset counts/times.
2. Right-click again to activate "Enable Analysis"
3. Start running the program to end, then stop.
4. Performance Analysis display will be updated as shown below in figure 2 below.



Index	Function	Cycle	Count	%	Histogram
0	{longway(unsigned char)}	39500	50	0%	
1	{recur(unsigned char)}	45000	500	0%	
2	{shortway(unsigned char)}	33000	50	0%	
3	{quick(unsigned char)}	107300	50	0%	

Figure 2 Performance Analysis of Displayed Functions

Above figure 2 displays performance of each specified functions by displaying the number of executing cycles required which can be obtained from the difference between the total number of execution when the target function is called and that when execution returns from the function.

2.2 Elaboration on Performance analysis window description

The following items displayed are:

1. Index: Index numbering
2. Function: User specified function name to be measured (or user specified start address of the function)
3. Cycle: Number of instruction execution cycles
4. Count: Number of calls for the function
5. %: Ratio of execution cycle count required for the function to the execution cycle count required for the whole program.
6. Histogram: Histogram display of the above ratio

By double-clicking on a function, displays a Performance Option dialog box where function can be modified.

2.3 Analysis on figure 2 results

From the above figure 2, four types of functions are written to obtain same results, but conclude that function “shortway”(using “for” loop “shift” method, refer to section 4 Sample Code for further information) is most efficient as it took the least execution time while function “quick” (using math.h library command) took the longest time. Function “recur” is using recursive method and “longway” is using a “for” loop “multiplication” method.

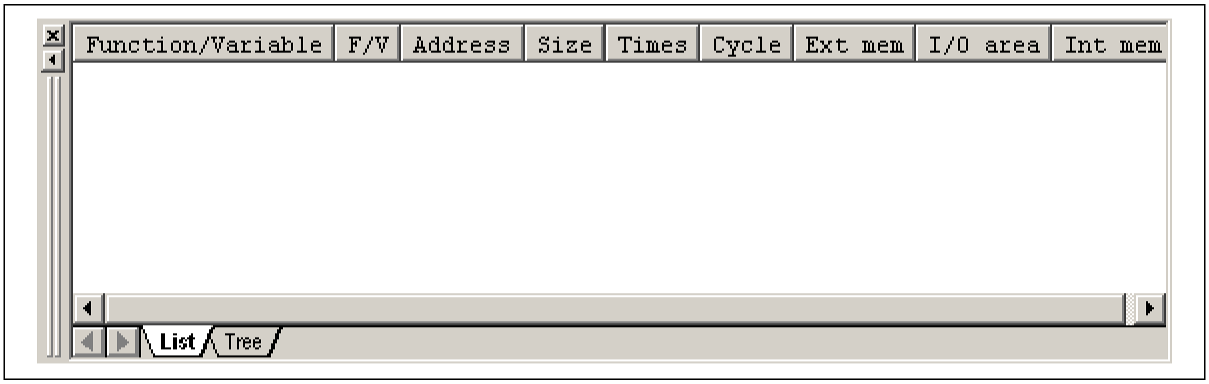
From these data, helps user in writing more efficient codes.

NOTE:

The codes are not optimized. The results are for illustration of the tool usage only.

3. How to Utilize Profile functions

Using the same project, select [View->Performance->Profile], Shift+Ctrl+F or icon to open the display as shown below in figure 3.



Function/Variable	F/V	Address	Size	Times	Cycle	Ext mem	I/O area	Int mem

Figure 3 Profile (List Sheet)

3.1 Elaboration on Profile (List and Tree Sheets) window description

Both figure 4 and 5 displayed the following profile data:

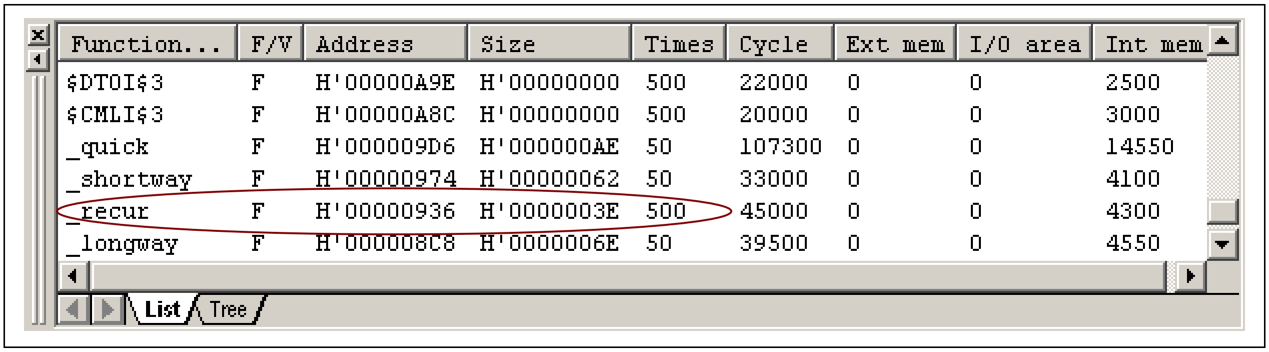
1.	Times:	Number of times a function is called
2.	Cycles:	Number of execution cycles
3.	Ext_mem:	Number of external memory access
4.	I/O area:	Number of input/output area excess
5.	Int_mem:	Number of internal memory access

Note: 1. The number of execution cycles and cache misses are calculated by subtracting the total execution cycles or cache misses at a specific function call instruction execution from the total execution cycles or cache misses at a return instruction execution of a specific function.

2. Displayed stack size does not represent the actual size. It can only be used as a reference value when function is called. It there is no stack information file (.sni) output from the optimizing linkage editor, the stack size will not be displayed. For more information on stack size or stack information file, refer to manual of the Optimizing Linkage Editor or application note on Stack Analysis Using Call Walker.

3.2 Steps to follow for executing Profile function

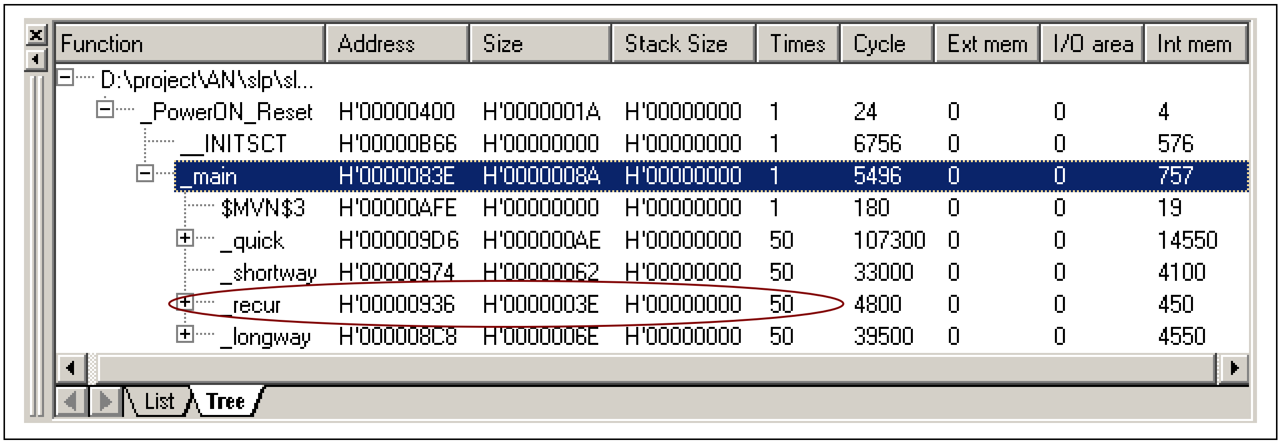
1. By right-clicking within the view area, a popup menu contains options, activate “Enable Profiler”.
2. Start running the program to end, then stop.
3. Profile displays will be updated as shown below. Figure 4 below displays Profile (List Sheet), double-click on the Function/Variable or Address column displays the source program corresponding to the address in the line.



Function...	F/V	Address	Size	Times	Cycle	Ext mem	I/O area	Int mem
$DTOI$3	F	H'00000A9E	H'00000000	500	22000	0	0	2500
$CMLI$3	F	H'00000A8C	H'00000000	500	20000	0	0	3000
_quick	F	H'000009D6	H'000000AE	50	107300	0	0	14550
_shortway	F	H'00000974	H'00000062	50	33000	0	0	4100
_recur	F	H'00000936	H'0000003E	500	45000	0	0	4300
_longway	F	H'000008C8	H'0000006E	50	39500	0	0	4550

Figure 4 Profile (List Sheet) of Displayed Function

4. Click to the other sheet called Profile (Tree Sheet) that was display in figure 5.



Function	Address	Size	Stack Size	Times	Cycle	Ext mem	I/O area	Int mem
D:\project\AN\slp\sl...								
_PowerON_Reset	H'00000400	H'0000001A	H'00000000	1	24	0	0	4
__INITSCT	H'00000B66	H'00000000	H'00000000	1	6756	0	0	576
_main	H'0000083E	H'0000008A	H'00000000	1	5496	0	0	757
MVN3	H'00000AFE	H'00000000	H'00000000	1	180	0	0	19
_quick	H'000009D6	H'000000AE	H'00000000	50	107300	0	0	14550
_shortway	H'00000974	H'00000062	H'00000000	50	33000	0	0	4100
_recur	H'00000936	H'0000003E	H'00000000	50	4800	0	0	450
_longway	H'000008C8	H'0000006E	H'00000000	50	39500	0	0	4550

Figure 5 Profile (Tree Sheet) of Displayed Function

Above figure 4 and 5 display almost similar content except that figure 4 displays in groups of functions or global variables while figure 5 displays the relation of function calls in a tree structure with an additional stack size content.

Notice that there is a difference of ‘Times’ value in “recur” function from above figures 4 and 5 even though both are of the same Profile function. Figure 4 reflects that “recur” function is called 500 times because Profile (List Sheet) displays the number of times each function or variable is being referred. Figure 5 reflects “recur” function is called 50 times because Profile (Tree Sheet) displays the number of times the calling function (main) calls each function.

3.3 Description on right-click popup menu on both Profile (List and Tree Sheets)

Description of popup menu options when right-clicking mouse within display window are as shown below:

1. View source: Displays the source program corresponding to the address in the selected line
2. View Profile-Chart: Displays user specified function on a chart format.
3. Enable Profiler: Toggles acquisition profile data.
4. Not trace the function call: Stops tracing function calls while profile data is acquired. This menu is used when acquiring profile data of the program in which functions are called in a special way, example switching of task in OS.
5. Find…: Search for a character string in the function column.
6. Find Data…: Search for maximum or minimum profile data.
   (Profile (Tree Sheet) only)
7. Clear Data: Clears data in Profile (List and Tree Sheets)
8. Output Profile Information Files: Display profiling results are saved to a file extension (.pro) to be view in Call Walker function for stack analysis. For details of profile information, refer to Optimizing Linkage Editor manual or application note on Stack Analysis Using Call Walker.
9. Output Text File: Saves display contents in a text file.
10. Setting:
    A. Show Functions/Variables (Profile (List Sheet) only)
    B. Show Functions (Profile (List Sheet) only)
    C. Show Variables (Profile (List Sheet) only)
    D. Only Executed Functions
    E. Include Data of Child Functions

3.4 Advantages of Profile (List Sheet)

With Profile (List Sheet) showing all functions and variables profile data, examples of what users benefit:

1. Address: Able to assure that the functions and variables are save to the desired/correct memory location.
2. Size: Knowing the size of the functions or variables allows user to further optimize the codes if it is too large.
3. Times: Confirm the number of time functions/variables is being called.
4. Cycle: Knowledge of execution cycle, users can optimize the code speed if value is large.
5. Ext_mem: More accurate estimation of external memory allocation when creating a workspace.
6. I/O area: Confirm if the number of access I/O area is correct.
7. Int_mem: Double-checking of codes, if int_mem value is too large to find a more efficient way of coding.

3.5 Advantages of Profile (Tree Sheet)

If user is unfamiliar with the program or if the codes are too large, Profile (Tree Sheet) will be a summary that displays the relations of each function with profile data same as section 3.4, Profile (List Sheet) as explained above. In addition, Profile (Tree Sheet) has stack size information that allows users to know how much stack size is used to execute each symbol without tediously tracing/monitoring every symbol’s stack size manually (seeing the stack register changes in register R7 for H8 series and R15 for SH series).

3.6 Viewing of Profile-Chart

As describe in section 3.3 ii), Profile-Chart displays the relation of calls for a specific function displayed in the profile (List and Tree Sheets) window.

By placing the cursor on the specified function and right-click, the specified function is displayed in the middle, the calling function on the left side, and the called function on the right side. Values beside the calling functions and the called functions show the number of times the functions has been called.

Right-clicking within display window shows similar popup menu as describe in section 3.3. Below shows popup menu which are not present in section 3.3:

1. Multiple view: For viewing of multiple Profile-Chart displays.
2. Expands/Reduces size: Expands/reduces spaces between each function

Figure 6 below shows an example of Profile-Chart, selecting main as the specified function. Values beside the calling and the called functions show the number times the function has been called.

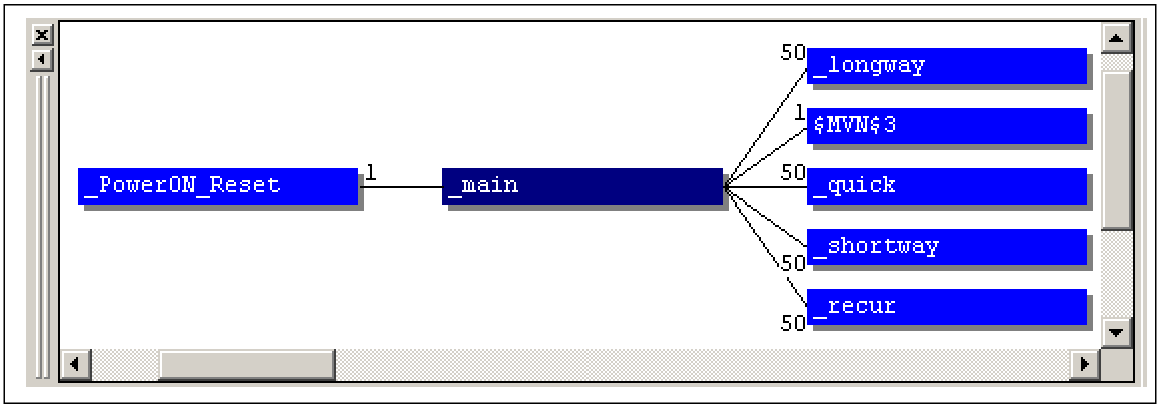
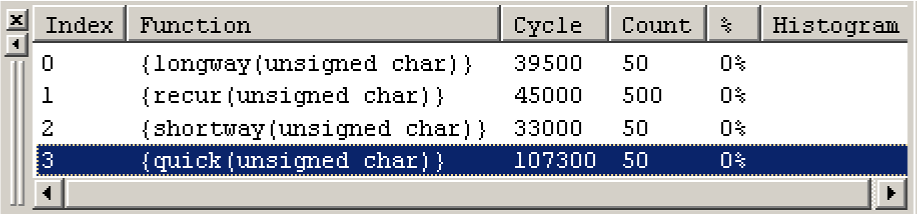


Figure 6 Profile-Chart Window of Displayed Functions

3.7 Advantages of Profile-Chart

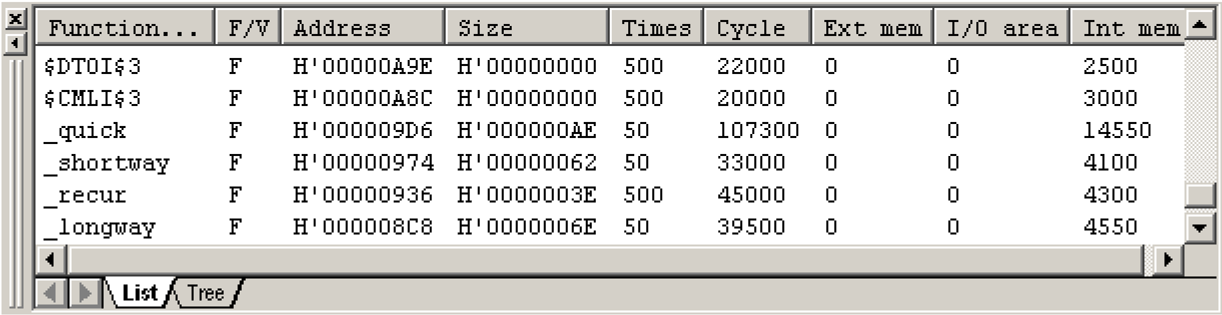
Gives the user a clearer picture of specific function structure, especially when the program is huge, multiple view displays save user trouble to scroll up and down repeatedly in order to view or compare differences between two functions at two different ends.

4. Comparison of Performance Analysis and Profile (List Sheet) Generated in HEW2-Simulator



Index	Function	Cycle	Count	%	Histogram
0	{longway(unsigned char)}	39500	50	0%	
1	{recur(unsigned char)}	45000	500	0%	
2	{shortway(unsigned char)}	33000	50	0%	
3	{quick(unsigned char)}	107300	50	0%	

Figure 7 Performance Analysis Results from HEW2-Simulator, same as in Figure 2



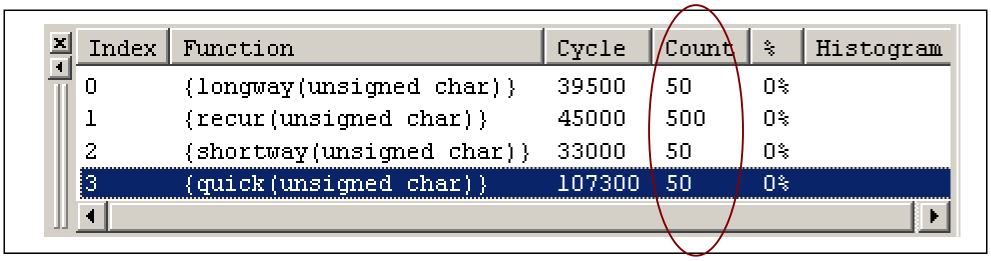
Function...	F/V	Address	Size	Times	Cycle	Ext mem	I/O area	Int mem
$DTOI$3	F	H'00000A9E	H'00000000	500	22000	0	0	2500
$CMLI$3	F	H'00000A8C	H'00000000	500	20000	0	0	3000
_quick	F	H'000009D6	H'000000AE	50	107300	0	0	14550
_shortway	F	H'00000974	H'00000062	50	33000	0	0	4100
_recur	F	H'00000936	H'0000003E	500	45000	0	0	4300
_longway	F	H'000008C8	H'0000006E	50	39500	0	0	4550

Figure 8 Profile (Tree Sheet) Results from HEW2-Simulator, same as in Figure 4

Figure 7 and figure 8 displayed ‘Count’ and ‘Times’ results are identical since both uses same method by counting the number of times each function is being referred to.

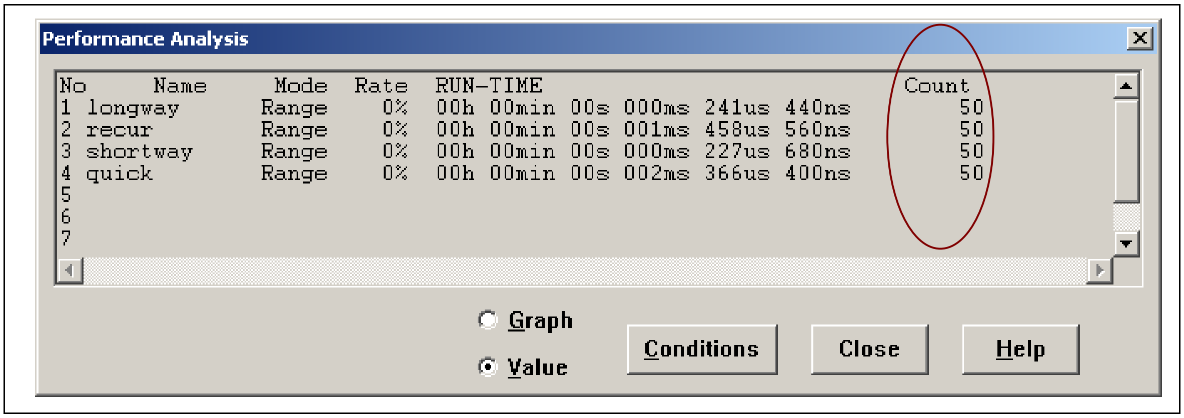
5. Comparison of Performance Analysis Generated in HEW2-Simulator and E6000 Series Emulator

Since Performance Analysis can be generated from both HEW2-Simulator and E6000 series emulator, figure 7 shows various function results from HEW2-Simulator and figure 8 from E6000 series emulator.



Index	Function	Cycle	Count	%	Histogram
0	{longway(unsigned char)}	39500	50	0%	
1	{recur(unsigned char)}	45000	500	0%	
2	{shortway(unsigned char)}	33000	50	0%	
3	{quick(unsigned char)}	107300	50	0%	

Figure 9 Performance Analysis Results from HEW2-Simulator, same as in Figure 2



Performance Analysis

```
No     Name      Mode   Rate  RUN-TIME                            Count
1 longway       Range    0%   00h 00min 00s 000ms 241us 440ns        50
2 recur         Range    0%   00h 00min 00s 001ms 458us 560ns        50
3 shortway      Range    0%   00h 00min 00s 000ms 227us 680ns        50
4 quick         Range    0%   00h 00min 00s 002ms 366us 400ns        50
5
6
7
```

○ Graph
● Value

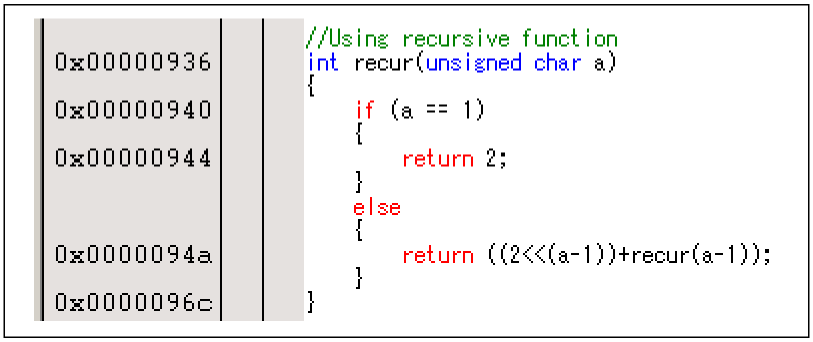
Figure 10 Performance Analysis Results from E6000 Series Emulator

Above figures 9 and 10, shows that each function is being called 50 times, except for "recur" function is different.

Since HEW2-Simulator is a software-controlled environment, the software is written in the way that 'Count' is determine by the number of times each function is being called. On the other hand, emulator is a 'real' environment that uses hardware tracing method to capture the start and end address of a function to obtain the 'Count' value.

Therefore, figure 9 shows that the simulated results of "recur" function are 500 times. Figure 10 shows the function is being called by the 'for loop' (same as other functions) 50 times and the other 10 times is the function is being referred or called to by itself at address 0x0000094a.

Figure 10 shows “recur” function is being called 50 times by E6000 series emulator. Figure 11 below explained that “recur” function in emulator counts from start address 0x00000936 to end address 0x0000096c, although there is a ‘self called’ at address 0x0000094c for ‘a=10’ times, but hardware trace treats it as 1 time.



```
                  //Using recursive function
0x00000936        int recur(unsigned char a)
                  {
0x00000940            if (a == 1)
                      {
0x00000944                return 2;
                      }
                      else
                      {
0x0000094a                return ((2<<(a-1))+recur(a-1));
                      }
0x0000096c        }
```

Figure 11 Recursive Function Address Display

In conclusion, emulator emulates the ‘real’ environment therefore ‘Run-Time’ results or execution time will be more accurate than HEW2-Simulator results. Due to H8 Series simplicity as compared to SH Series, HEW2-Simulator and emulator ‘cycle’ results are similar identical but due to complexity of SH pipelines, super scalar, cache access, emulator will be better choice for time measurement

In order to have a better estimation in stack size being used, HEW2-Simulator is better as it captures the exact number of times each function is being called.

6. Sample Code

```
/***********************************************************************/
/*                                                                     */
/*  FILE        :slp.c                                                 */
/*  DATE        :Tue, Jul 01, 2003                                     */
/*  DESCRIPTION :Main Program                                          */
/*  CPU TYPE    :H8/Other                                              */
/*                                                                     */
/*  This file is generated by Hitachi Project Generator (Ver.2.4).     */
/*                                                                     */
/***********************************************************************/


//Aim: To work out the difference between different ways of calculating
//  "Sum of 2^10"
//     eg. stack size used & performance analysis

#include <math.h>

void main(void)
{
   int count = 10;
   int databuf[4] = {0};
   int a;

   //4 different ways of obtaining same results and loop for 50 times.
   for(a=0;a<50;a++)
   {
      databuf[0] = longway(count);  //"for" loop
      databuf[1] = shortway(count); //shift
      databuf[2] = quick(count);    //math.h
      databuf[3] = recur(count);    //recursive function
   }
   while(1);
}

//Using "for" loop
int longway(unsigned char a)
{
   int d, power;
   int res1 = 0, two = 2;

   power = 1;
   for (d=0; d<a; d++)
   {
      power* = two;
      res1 += power; //1364d=554h
   }
   return res1;
}
```

```
//Using recursive function
int recur(unsigned char a)
{
   if (a == 1)
   {
      return 2;
   }
   else
   {
      return ((2<<(a-1))+recur(a-1));
   }
}

//Making use of "shift" command
int shortway(unsigned char x)
{
   int z, shif, res2;

   res2 = shif = 2;
   for (z=1; z<x; z++)
   {
      shif <<= 1;
      res2 += shif;
   }
    return res2;
}

//Making use of "math.h" library
int quick(unsigned char u)
{
   int p,q,r;
   int res3 = 0;

   for (p=1; p<=u; p++)
   {
      q = pow(2,p);
      res3 += q;
   }
   return res3;
}
```

REFERENCE

1. *H8S, H8/300 Series C/C++ Compiler Assembler Optimizing Linker Editor User's Manual, Revision 4.0, HEW On-line Manual, Renesas Technology Corp.*

2. *Application note on Stack Analysis of Using HEW Call Walker, Revision 1.0, 2003, Renesas Technology Corp.*

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	September.03	—	First edition issued

Keep safety first in your circuit designs!

1. Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
2. Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.