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.

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.

SH/Tiny Series (SH7125) E10A-USB Emulator

Method for Using Sequential Breaks

Summary

The SH/Tiny Series (SH7125) E10A-USB Emulator permits you to execute a sequential break by setting sequential conditions for hardware breaks. This document describes how to set and execute a sequential break using the E10A-USB emulator.

Table of Contents

1. Specification

The E10A-USB emulator from Renesas permits you to set a sequential break condition, comprised of multiple event points set, which causes program execution to stop based on a combinatorial condition of those event points occurring in a specified sequence. Specifying the condition for event points in a sequence, it is possible to debug program failures or hardware failures encountered in a limited situation efficiently.

2. Facilities Used

In this document, the method on how to set and execute a sequential break is described using the sample program included in the CD-ROM supplied with the E10A-USB emulator or the downloaded package from the Renesas website.
The E10A-USB emulator software version used here is V.2.09 Release 00.

3. Preparing the Software

3.1 Getting Started

Install the software included in the CD-ROM that is supplied with the E10A-USB emulator. The sample program (tutorial workspace) used in this document will be expanded in your PC.
The software included in the CD-ROM that is supplied with the E10A-USB emulator may also be installed in a PC that has had the High-performance Embedded Workshop already installed. In this case, part of the dialogs displayed during the installation will be omitted.

3.2 Installing the Software Supplied with the E10A-USB Emulator

Execute HewInstMan.exe present in the CD-ROM supplied with the E10A-USB emulator.
For details on how to install, see the E10A-USB emulator setup guide published at the Renesas website. During installation work, follow the instructions displayed on the screen. The installation procedure is omitted here.

3.3 Installing Other Necessary Software

(1) If you've selected the auto update utility when installing software, it is possible to confirm via the Internet whether the latest version of each product is available.

(2) In this document, part of the sample program is modified, with which to verify the program behavior. For this reason, the C/C++ compiler package for the SuperH family is used. If you've already purchased a product-edition compiler package, install the product-edition compiler package.

(3) If you've not purchased a product-edition compiler package yet, a free evaluation-edition compiler package may be downloaded from the Renesas website. To find a free evaluation edition of the SuperH family C/C++ compiler package, choose Support → Download → Download Search from Renesas Top Page and then open category selection, in which select "Free Evaluation Edition" and search for the product you want. Links to the Renesas website are given in the last section of this document. Information about the limitations of and the method for installing the free evaluation edition may be obtained from the download page.

4. Preparing the Hardware

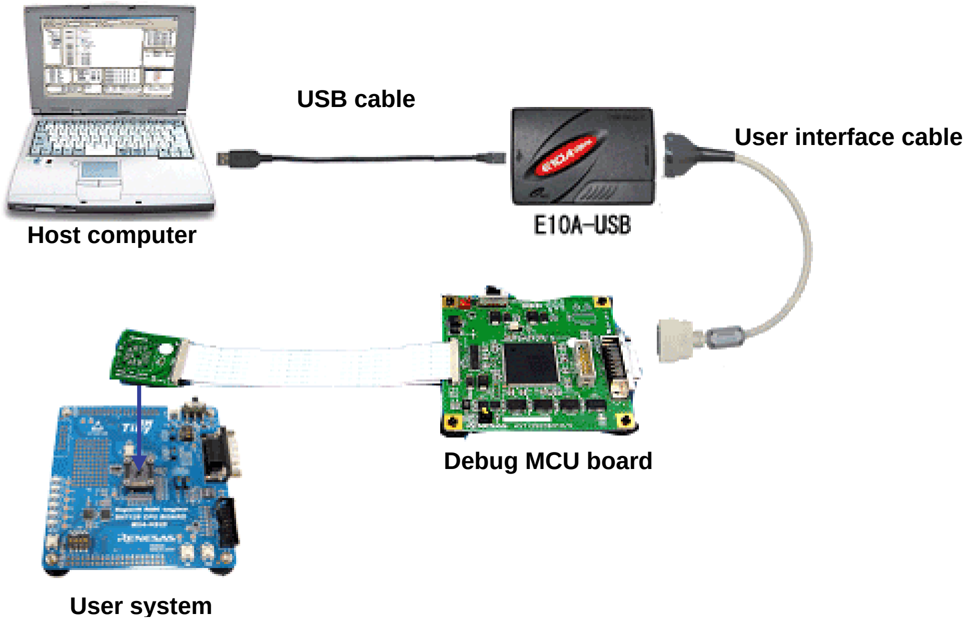
4.1 SH7125 Series Debug MCU Board (HS7125EDB01H Made by Renesas)

The SH7125 series debug MCU board (HS7125EDB01H) supports the SH/Tiny (SH7125)-series microcomputers from Renesas Technology.

The debug MCU board connects to the user system via the IC socket on the user system. Therefore, it is possible to debug in a manner close to finished product. Furthermore, when combined with the E10A-USB emulator, it is possible to debug in any desired place, whether a laboratory or field, using a PC (IBM PC or compatible) that incorporates USB1.1/2.0 (Full-Speed) as the host computer.

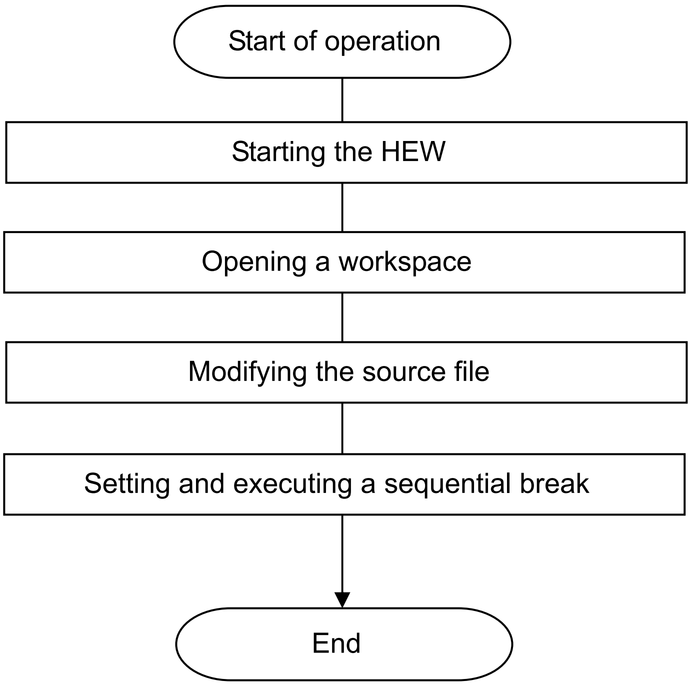
4.2 Checking the Connection Environment

Before using the debug MCU board, always be sure to check that the host computer, E10A-USB emulator, debug MCU board and the user system are connected with USB cable and user interface cable as shown below. If these pieces of equipment are improperly connected, correct the connections.



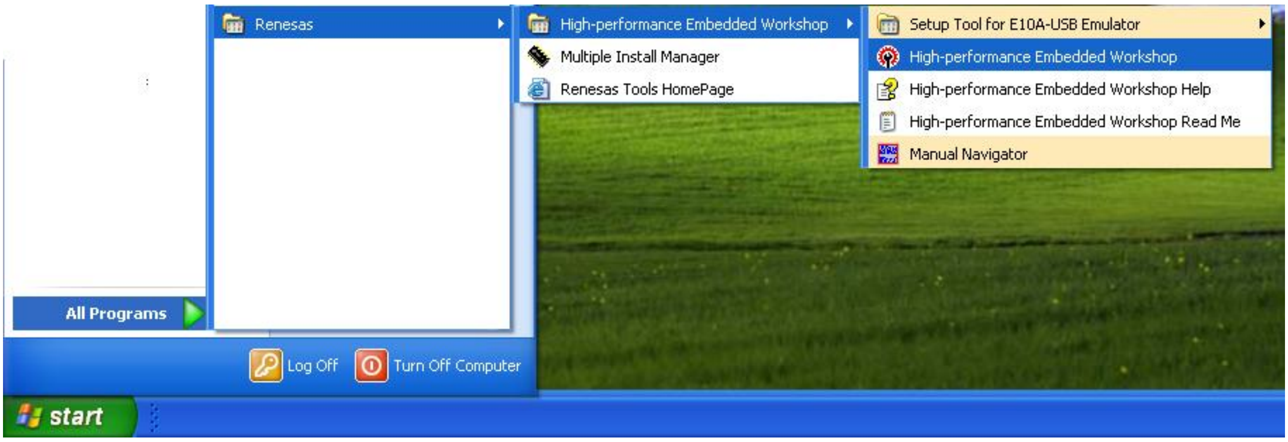
5. Operational Description

This section describes how to set and execute a sequential break after starting the High-performance Embedded Workshop (HEW). The procedure is shown below.


Start of operation
Starting the HEW
Opening a workspace
Modifying the source file
Setting and executing a sequential break
End

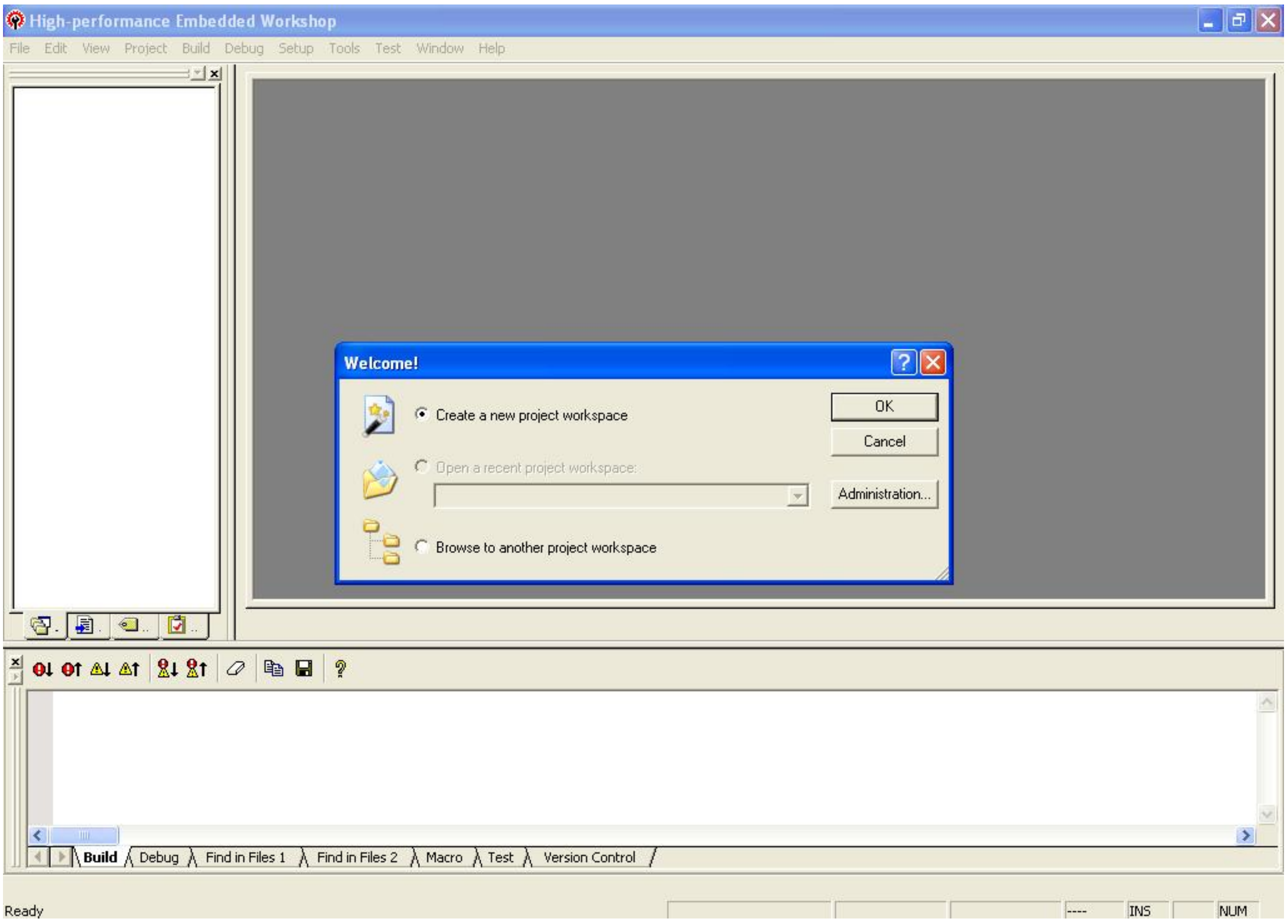
5.1 Starting the High-performance Embedded Workshop

To begin with, first connect the E10A-USB emulator that has the debug MCU board and user system connected to it and the host computer with USB cable, and check to see that everything is ready to debug. Next, start the High-performance Embedded Workshop.
From All Programs on the Start menu, choose Renesas → High-performance Embedded Workshop → High-performance Embedded Workshop, to start.

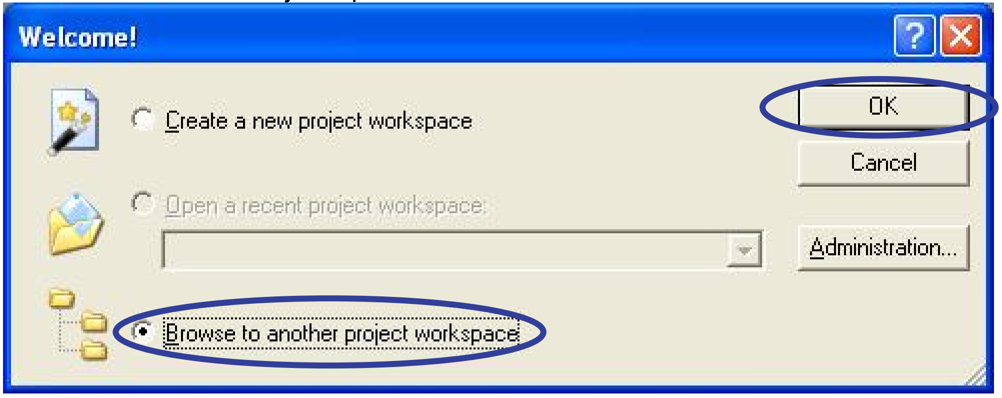

Renesas
High-performance Embedded Workshop
Multiple Install Manager
Renesas Tools HomePage
Setup Tool for E10A-USB Emulator
High-performance Embedded Workshop
High-performance Embedded Workshop Help
High-performance Embedded Workshop Read Me
Manual Navigator
All Programs
Log Off
Turn Off Computer
start

5.2 Opening a Workspace

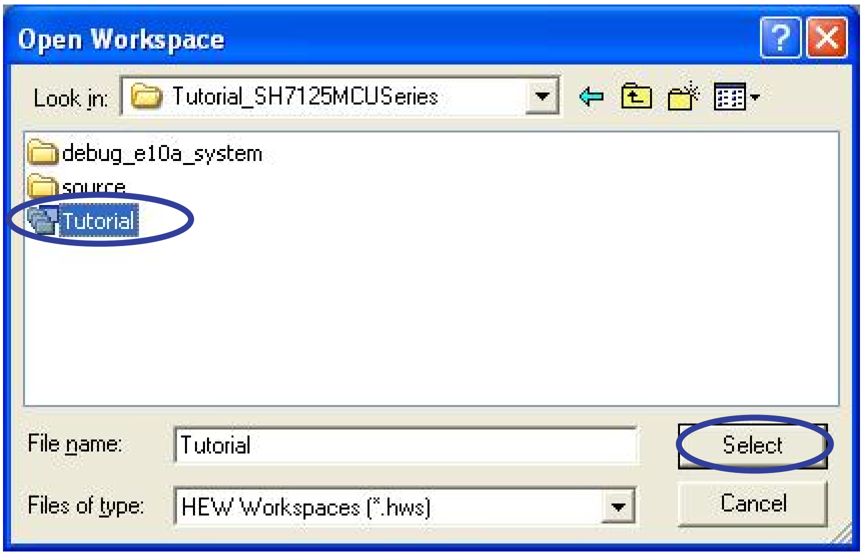
(1) The welcome dialog box will be displayed in the High-performance Embedded Workshop.


High-performance Embedded Workshop
File Edit View Project Build Debug Setup Tools Test Window Help
Welcome!
● Create a new project workspace
○ Open a recent project workspace:
○ Browse to another project workspace
OK
Cancel
Administration...
Build Debug Find in Files 1 Find in Files 2 Macro Test Version Control
Ready
INS
NUM

Select the “Browse Another Project Space” radio button in this box and click the OK button.


Welcome!
○ Create a new project workspace
○ Open a recent project workspace:
● Browse to another project workspace
OK
Cancel
Administration...

(2) The Open a Workspace dialog box will be displayed.



Look in: Tutorial_SH7125MCUSeries

debug_e10a_system
source
Tutorial

File name: Tutorial

Files of type: HEW Workspaces (*.hws)

If the installation of the CD-ROM of this product is complete, the workspace “Tutorial.hws” is stored as standard in the folder position shown below. Check folder positions in order while you locate. When the workspace “Tutorial.hws” is found, specify it and click the Select button.

```
C:\WorkSpace\Tutorial\E10A-USB\SH-2\SH7080Series\Tutorial_SH7125MCUSeries\Tutorial.hws

C:\WorkSpace
    └Tutorial
        └E10A-USB
            └SH-2
                └SH7080Series
                    └Tutorial_SH7125MCUSeries
                        └Tutorial.hws
```

Note: Depending on the software version used, it will occur that the above directory cannot be specified. In such a case, select the directory given below.

<Directory in which the High-performance Embedded Workshop is installed>

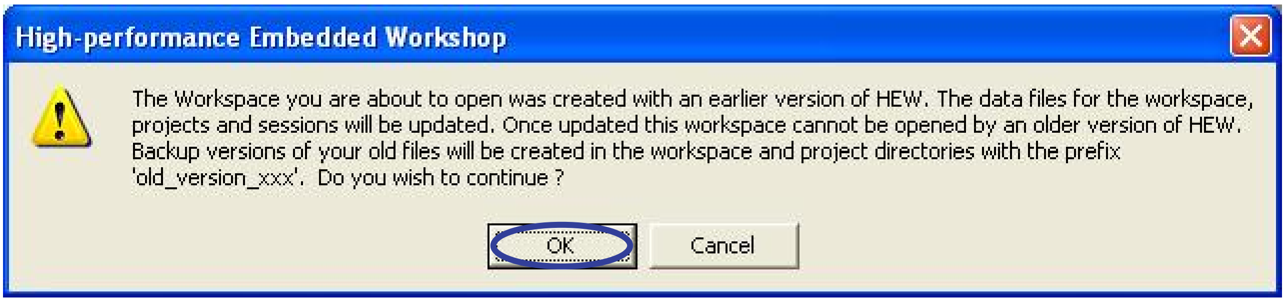
\Tools\Renesas\DebugComp\Platform\E10-USB\SH-2\SH7080Series\Tutorial_SH7125MCUSeries

Examples of directory:

C: \hew3\Tools\Renesas\DebugComp\Platform\E10-USB\SH-2\SH7080Series\Tutorial_SH7125MCUSeries

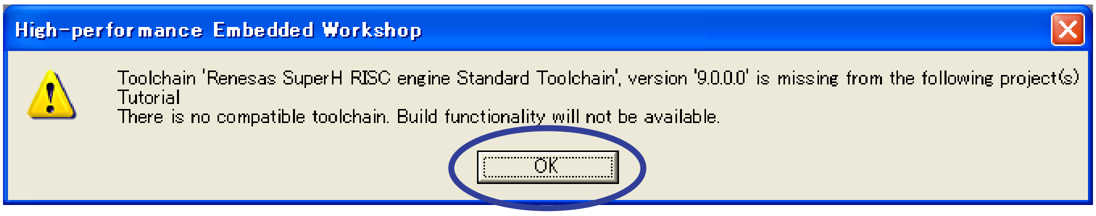
C: \hew2\Tools\Renesas\DebugComp\Platform\E10-USB\SH-2\SH7080Series\Tutorial_SH7125MCUSeries

(3) If the workspace version is old, the dialog box shown below is displayed. To update it to a new version, click the OK button.



The Workspace you are about to open was created with an earlier version of HEW. The data files for the workspace, projects and sessions will be updated. Once updated this workspace cannot be opened by an older version of HEW. Backup versions of your old files will be created in the workspace and project directories with the prefix 'old_version_xxx'. Do you wish to continue ?

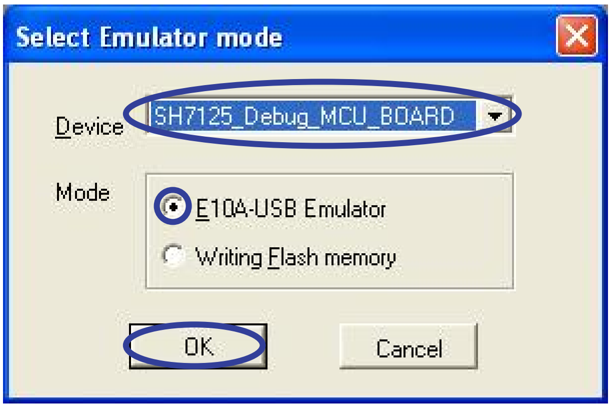
(4) If the dialog box shown below is displayed, click OK.



High-performance Embedded Workshop

Toolchain 'Renesas SuperH RISC engine Standard Toolchain', version '9.0.0.0' is missing from the following project(s)
Tutorial
There is no compatible toolchain. Build functionality will not be available.

(5) The Select Emulator Mode dialog box will be displayed. In this dialog box, select "SH7125_Debug_MCU_BOARD" in the Device column and then "E10A-USB Emulator" for Mode, and then click the OK button.

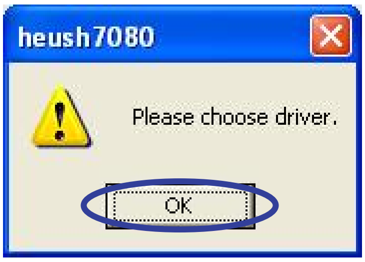


Select Emulator mode

Device: SH7125_Debug_MCU_BOARD

Mode:
● E10A-USB Emulator
○ Writing Flash memory

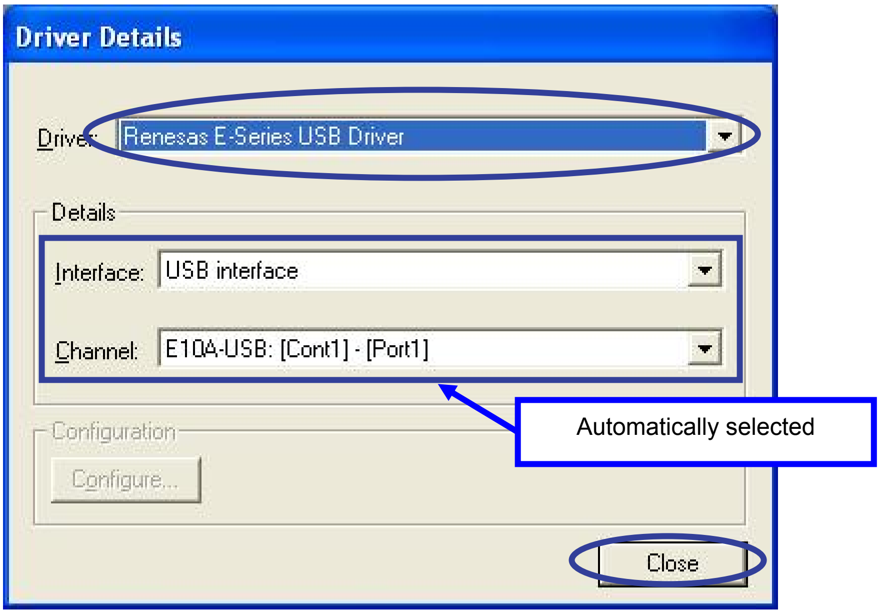
(6) The "heush7080" dialog box will be displayed only the first time you open a workspace. When this dialog box is displayed, click OK.



heush7080

Please choose driver.

(7) The Driver Details dialog box will be displayed only the first time you open a workspace. Select "Renesas E-Series USB Driver" for Driver and click the Close button.



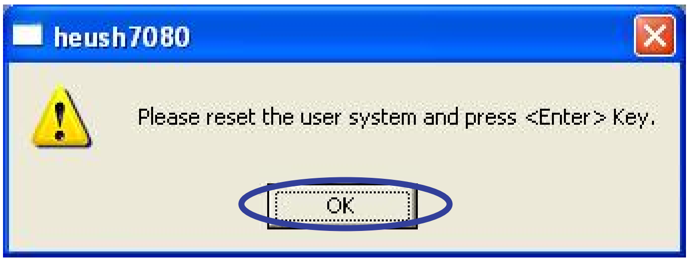
Driver Details

Driver: Renesas E-Series USB Driver

Details
Interface: USB interface
Channel: E10A-USB: [Cont1] - [Port1]

Configuration

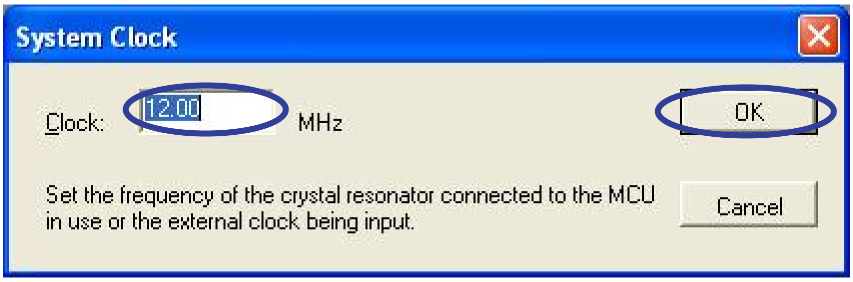
(8) The “heush7080” dialog box will be displayed. After turning RESET_SW on the debug MCU board on once, click the OK button in this dialog box.



Please reset the user system and press <Enter> Key.

For the debug MCU board used for explanation in this document, SW4 mounted on it corresponds to RESET_SW.

(9) The System Clock dialog box will be displayed. Enter the external clock frequency used and click OK.

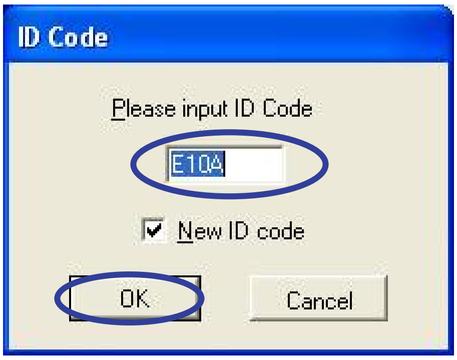


Clock: 12.00 MHz

Set the frequency of the crystal resonator connected to the MCU in use or the external clock being input.

In the explanation here, the external clock frequency used is 12.00 MHz.

(10)The ID Code dialog box will be displayed. Leave the default value “E10A” set for Input ID Code intact and simply click OK.

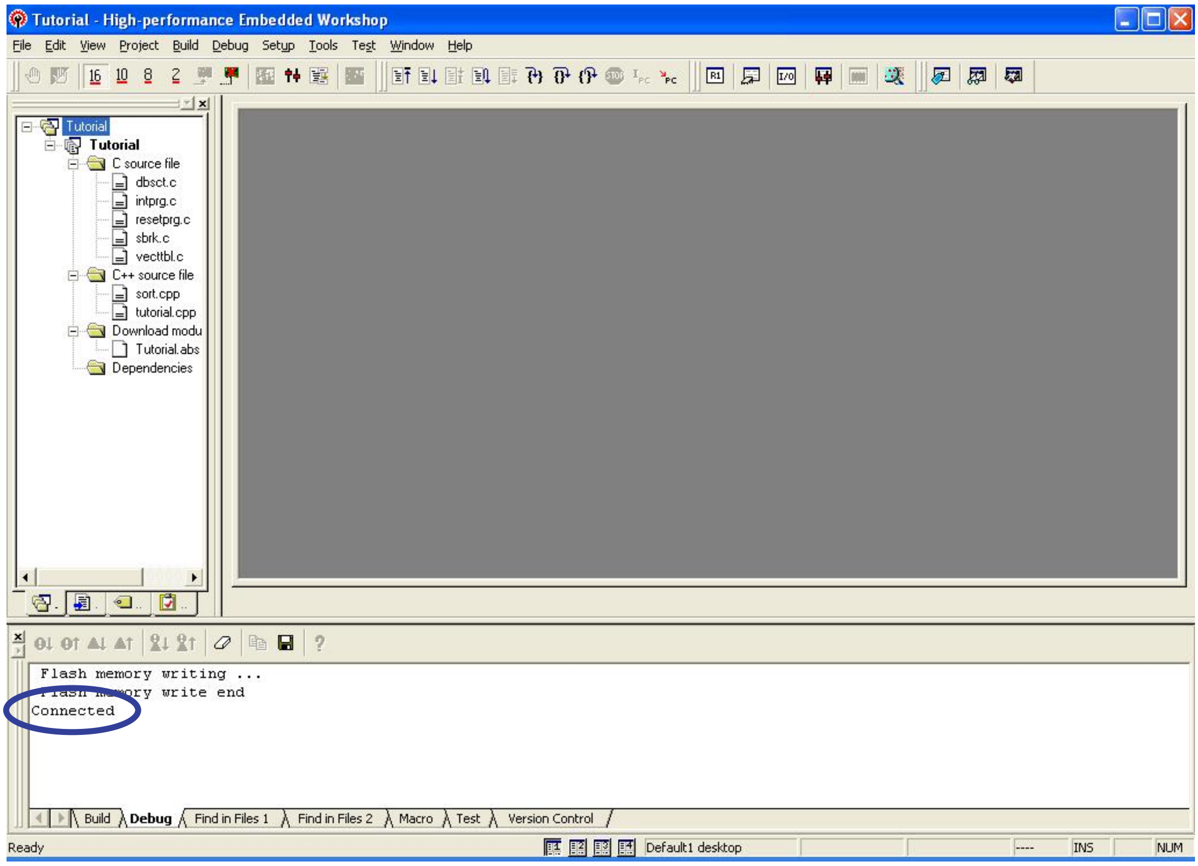


Please input ID Code

E10A

☑ New ID code

(11)A connection of the E10A-USB emulator is complete, so you're now ready to operate on the High-performance Embedded Workshop screen. When this connection is complete, a message "Connected" is displayed on the Debug tab pane of the output window.

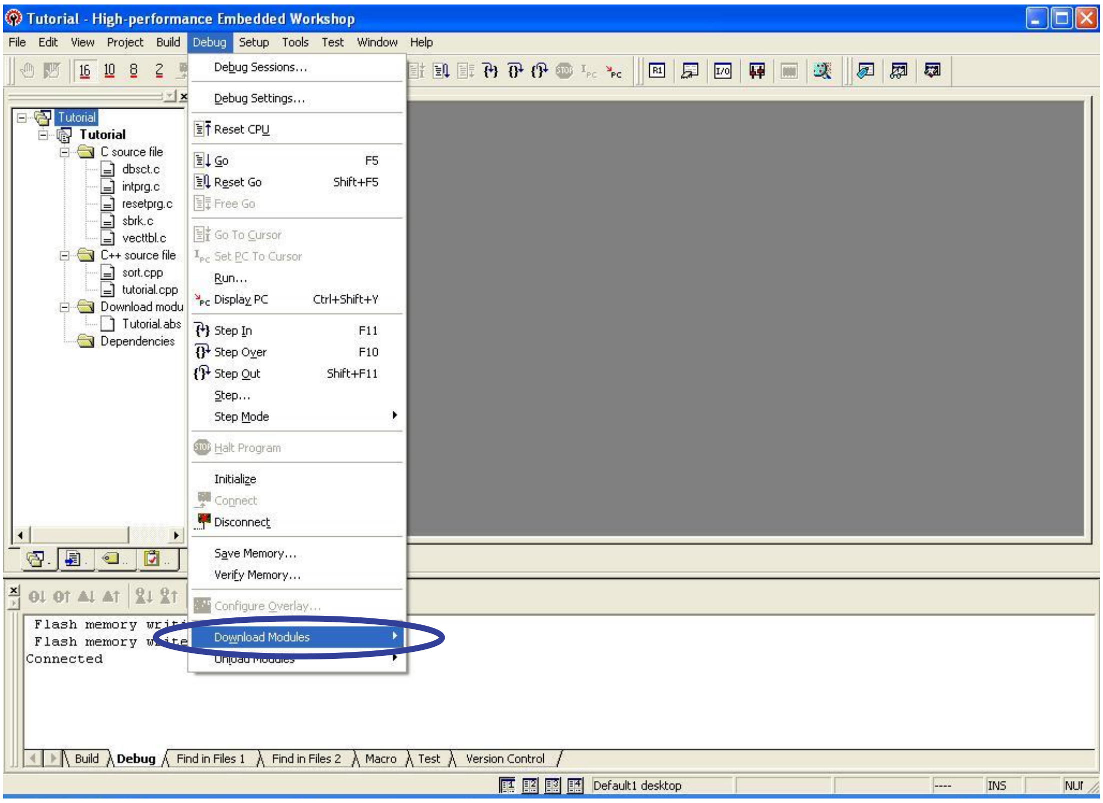


```
Flash memory writing ...
Flash memory write end
Connected
```

5.3 Modifying the Source File

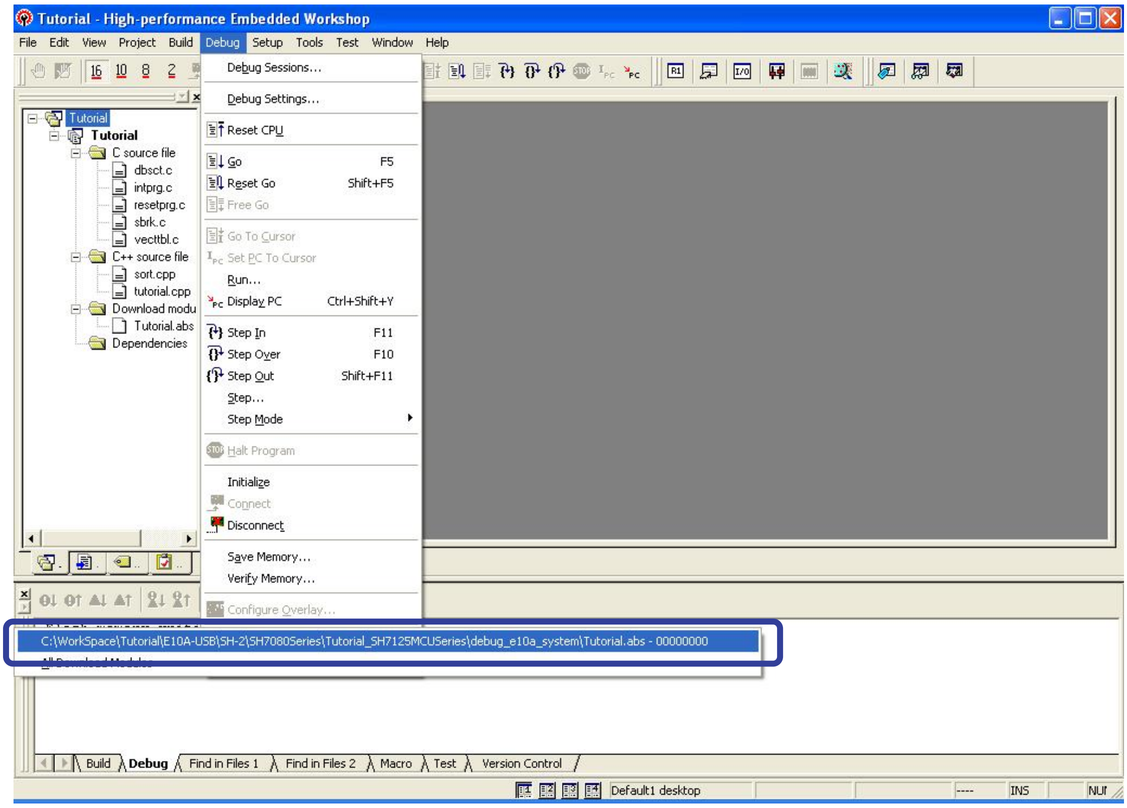
In this section we'll modify part of the source file to make the sequential break operation procedure easily understandable.

(1) To download the sample program, choose Download from the Debug menu.

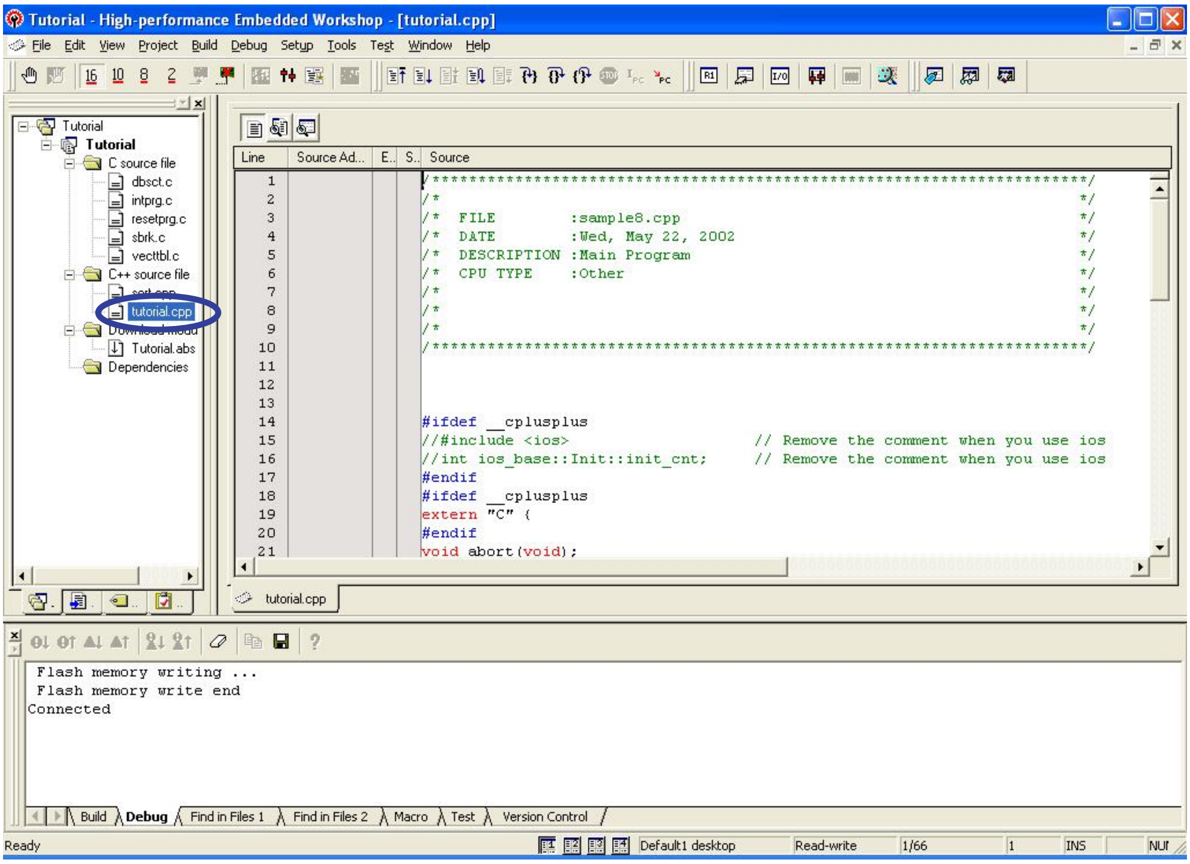


```
Flash memory writ
Flash memory write
Connected
```

(2) Select the file "Tutorial.abs" registered in the workspace.



(3) Double-click the workspace source file name “tutorial.cpp” to open the source file.



```
/*************************************************************************/
/*                                                                       */
/*  FILE        :sample8.cpp                                             */
/*  DATE        :Wed, May 22, 2002                                       */
/*  DESCRIPTION :Main Program                                            */
/*  CPU TYPE    :Other                                                   */
/*                                                                       */
/*                                                                       */
/*                                                                       */
/*************************************************************************/



#ifdef __cplusplus
//#include <ios>                        // Remove the comment when you use ios
//int ios_base::Init::init_cnt;         // Remove the comment when you use ios
#endif
#ifdef __cplusplus
extern "C" {
#endif
void abort(void);
```

```
 Flash memory writing ...
 Flash memory write end
Connected
```

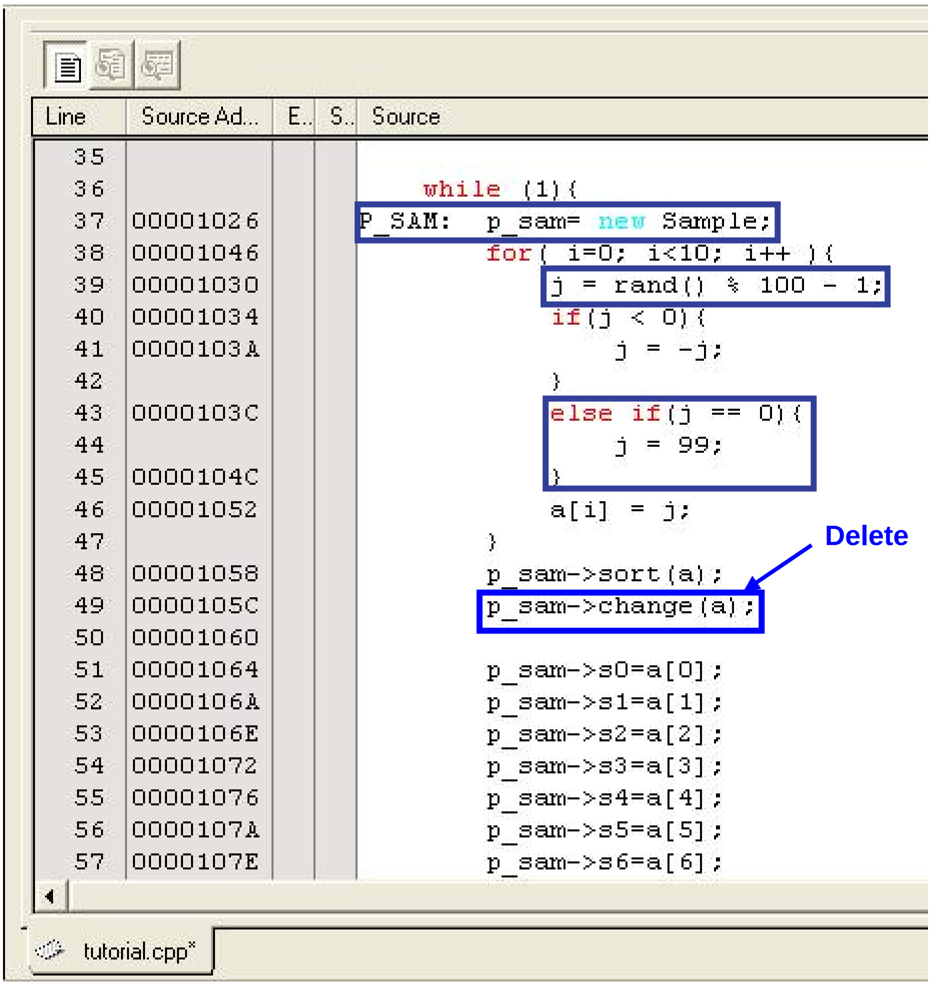
(4) Modify the source file.

(1) Add P_SAM: to the 37th line of the source file as shown below. Then add % 100 – 1 to the 39th line and

```
else if(j == 0){
        j = 99;
}
```

to the 43rd line and those that follow.

Delete the 49th line.



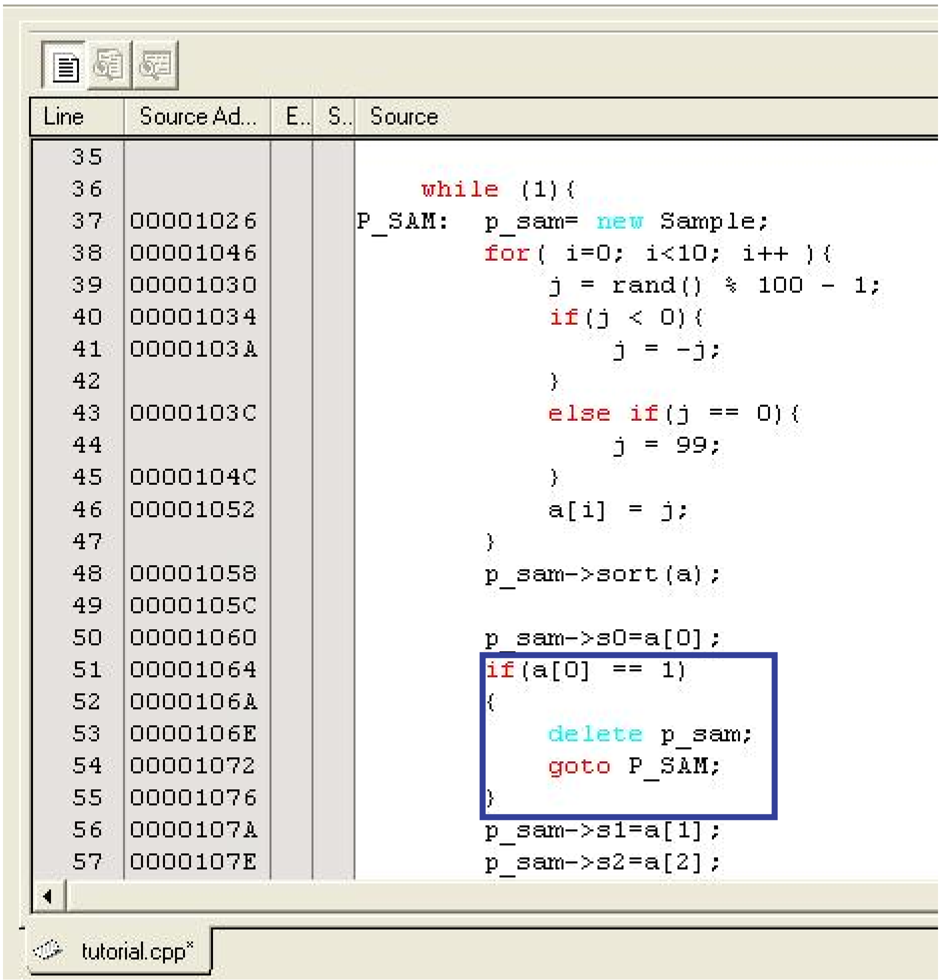
```
Line  Source Ad...  E..  S..  Source
35
36                                 while (1){
37    00001026                P_SAM:   p_sam= new Sample;
38    00001046                         for( i=0; i<10; i++ ){
39    00001030                             j = rand() % 100 - 1;
40    00001034                             if(j < 0){
41    0000103A                                 j = -j;
42                                         }
43    0000103C                             else if(j == 0){
44                                             j = 99;
45    0000104C                             }
46    00001052                             a[i] = j;
47                                     }
48    00001058                         p_sam->sort(a);
49    0000105C                         p_sam->change(a);
50    00001060
51    00001064                         p_sam->s0=a[0];
52    0000106A                         p_sam->s1=a[1];
53    0000106E                         p_sam->s2=a[2];
54    00001072                         p_sam->s3=a[3];
55    00001076                         p_sam->s4=a[4];
56    0000107A                         p_sam->s5=a[5];
57    0000107E                         p_sam->s6=a[6];
```

Delete

(2) In the 51st line of the source file and those that follow, insert

```
if(a[0] == 1)
{
   delete p_sam;
   goto P_SAM;
}
```

as shown below.



```
Line  Source Ad...  E..  S..  Source
 35
 36                                     while (1){
 37   00001026                 P_SAM:   p_sam= new Sample;
 38   00001046                          for( i=0; i<10; i++ ){
 39   00001030                                j = rand() % 100 - 1;
 40   00001034                                if(j < 0){
 41   0000103A                                      j = -j;
 42                                             }
 43   0000103C                                else if(j == 0){
 44                                                   j = 99;
 45   0000104C                                }
 46   00001052                                a[i] = j;
 47                                       }
 48   00001058                          p_sam->sort(a);
 49   0000105C
 50   00001060                          p_sam->s0=a[0];
 51   00001064                          if(a[0] == 1)
 52   0000106A                          {
 53   0000106E                                delete p_sam;
 54   00001072                                goto P_SAM;
 55   00001076                          }
 56   0000107A                          p_sam->s1=a[1];
 57   0000107E                          p_sam->s2=a[2];
```

(5) The above completes a modification of the source file.

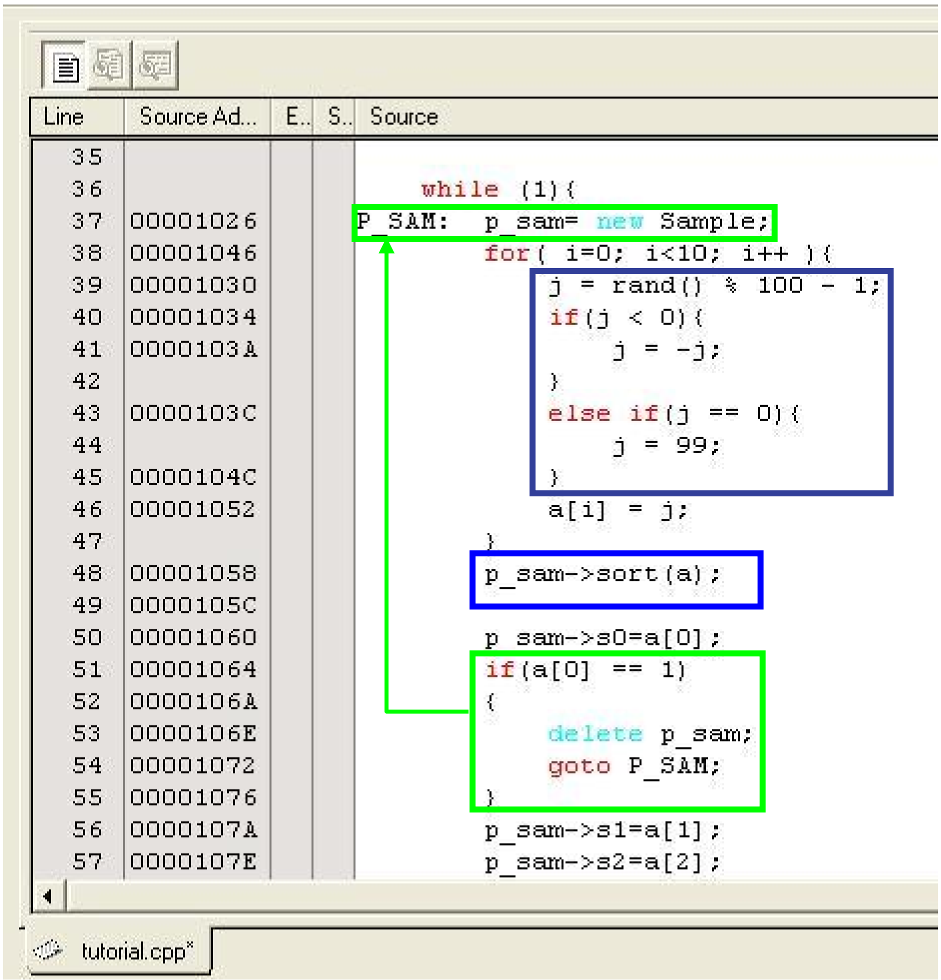
The addition of % 100 – 1 in the 39th line brings about such an effect that the value j takes on is 1 less than the remainder of the random number divided by 100 and therefore a random number from -1 to 98. However, since a negative value is converted to a positive value in the 40th thru 42nd lines, -1 becomes 1, so that if the value of j in the 43rd thru 45th lines is 0, j has 99 put in it and the value of j, therefore, is a random number from 1 to 98.

Since p_sam → change(a); in the 49th line has been deleted, values are sorted from a[0] to a[9] in ascending order.

Add P_SAM: to the 37th line and insert

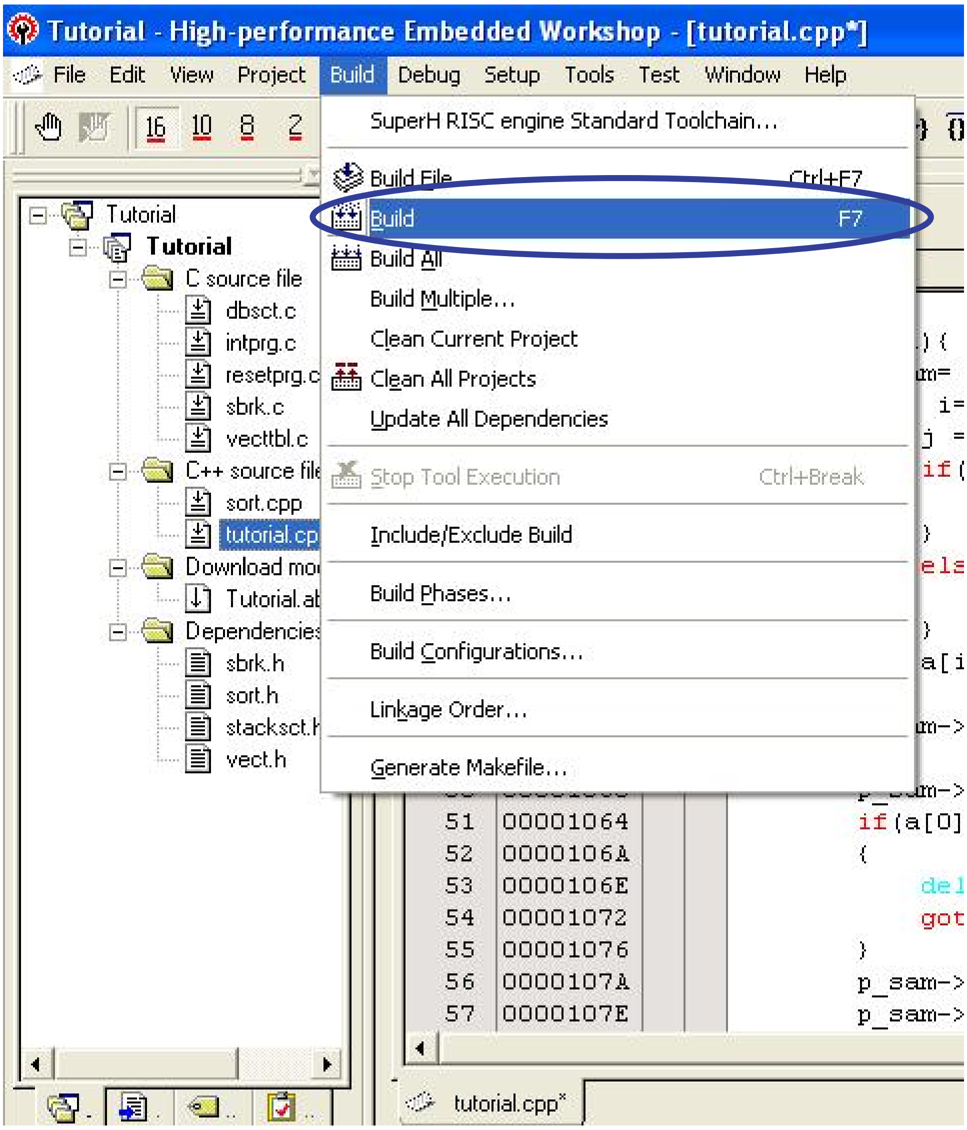
```
if(a[0] == 1)
  {
     delete p_sam;
     goto P_SAM;
  }
```

in the 51st line and those that follow. As a result, if the value of a[0] is 1, the execution statement that follows is not executed, and control returns to the 37th line, or the beginning of the loop.



```
Line  Source Ad...  E..  S..  Source
 35
 36                                   while  (1){
 37   00001026                  P_SAM:   p_sam= new Sample;
 38   00001046                           for( i=0; i<10; i++ ){
 39   00001030                                j = rand() % 100 - 1;
 40   00001034                                if(j < 0){
 41   0000103A                                     j = -j;
 42                                           }
 43   0000103C                                else if(j == 0){
 44                                                j = 99;
 45   0000104C                                }
 46   00001052                                a[i] = j;
 47                                      }
 48   00001058                           p_sam->sort(a);
 49   0000105C
 50   00001060                           p_sam->s0=a[0];
 51   00001064                           if(a[0] == 1)
 52   0000106A                           {
 53   0000106E                                delete p_sam;
 54   00001072                                goto P_SAM;
 55   00001076                           }
 56   0000107A                           p_sam->s1=a[1];
 57   0000107E                           p_sam->s2=a[2];
```

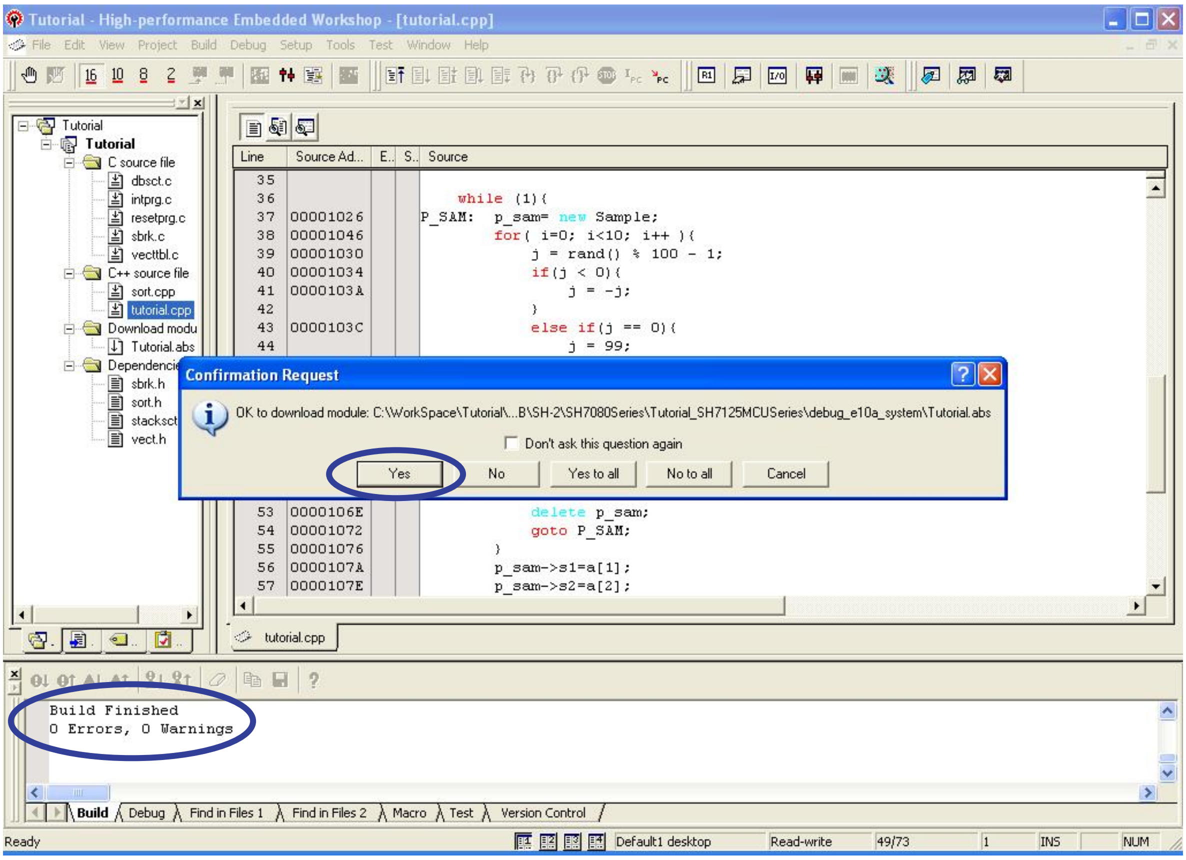
(6) Perform a build on the program comprised of the modified source file to make it downloadable.
Choose Build from the Build menu.



```
51 00001064    if(a[0]
52 0000106A    {
53 0000106E        del
54 00001072        got
55 00001076    }
56 0000107A    p_sam->
57 0000107E    p_sam->
```

See the Build tab pane of the output window to confirm the progress of build work.

(7) When a build is complete, the number of errors and warnings that occurred during the process are displayed on the Build tab pane of the output window. If the program is downloadable, the dialog box shown below is displayed prompting for your confirmation of whether to download the build-complete program file automatically. Click the Yes button here to download the program into the target board.



```
                  while (1){
         P_SAM:   p_sam= new Sample;
00001026            for( i=0; i<10; i++ ){
00001046                j = rand() % 100 - 1;
00001030                if(j < 0){
00001034                    j = -j;
0000103A                }
                        else if(j == 0){
0000103C                    j = 99;
...
0000106E                delete p_sam;
00001072                goto P_SAM;
00001076            }
0000107A            p_sam->s1=a[1];
0000107E            p_sam->s2=a[2];
```

Confirmation Request

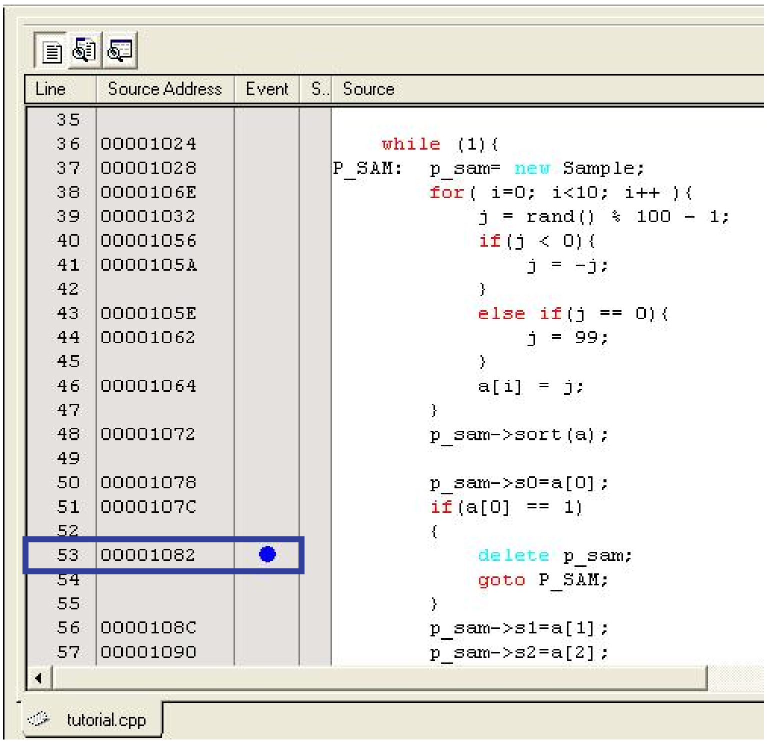
OK to download module: C:\WorkSpace\Tutorial\...B\SH-2\SH7080Series\Tutorial_SH7125MCUSeries\debug_e10a_system\Tutorial.abs

□ Don't ask this question again

```
Build Finished
0 Errors, 0 Warnings
```

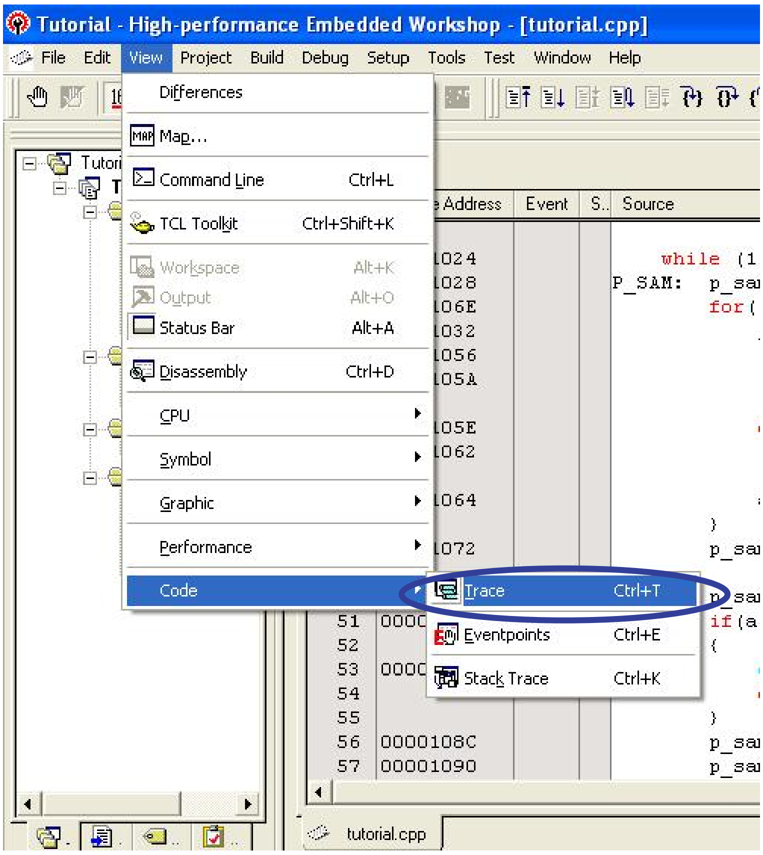
5.4 Setting and Executing Sequential Breaks

(1) Double-click in the Event column at the 53rd line position of the source file and set an event point there. Written here is an execution statement for the case where the value of a[0] is 1.

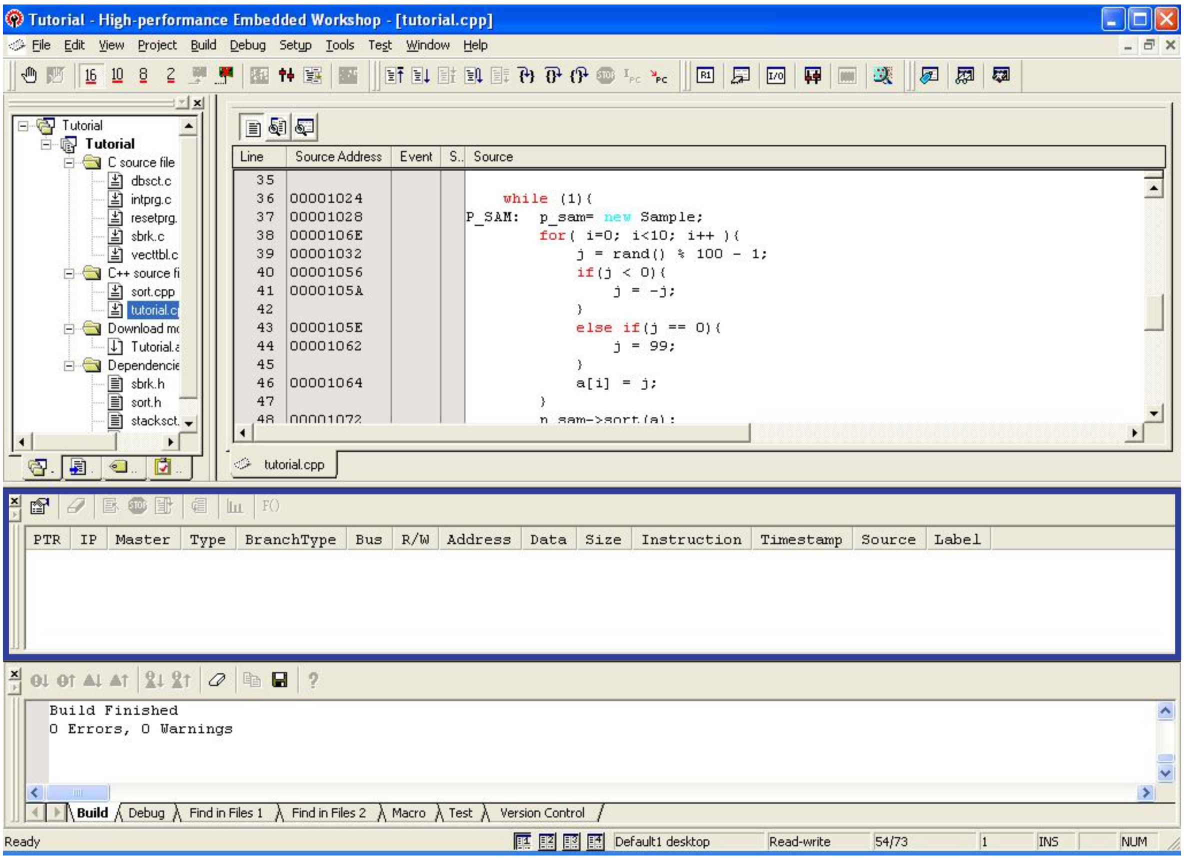


```
Line  Source Address  Event  S..  Source
 35
 36   00001024                        while (1){
 37   00001028                    P_SAM:  p_sam= new Sample;
 38   0000106E                            for( i=0; i<10; i++ ){
 39   00001032                                j = rand() % 100 - 1;
 40   00001056                                if(j < 0){
 41   0000105A                                    j = -j;
 42                                           }
 43   0000105E                                else if(j == 0){
 44   00001062                                    j = 99;
 45                                           }
 46   00001064                                a[i] = j;
 47                                       }
 48   00001072                            p_sam->sort(a);
 49
 50   00001078                            p_sam->s0=a[0];
 51   0000107C                            if(a[0] == 1)
 52                                       {
 53   00001082         ●                      delete p_sam;
 54                                           goto P_SAM;
 55                                       }
 56   0000108C                            p_sam->s1=a[1];
 57   00001090                            p_sam->s2=a[2];
```

(2) From the View menu, choose Code and then Trace.



(3) The trace window will be displayed.

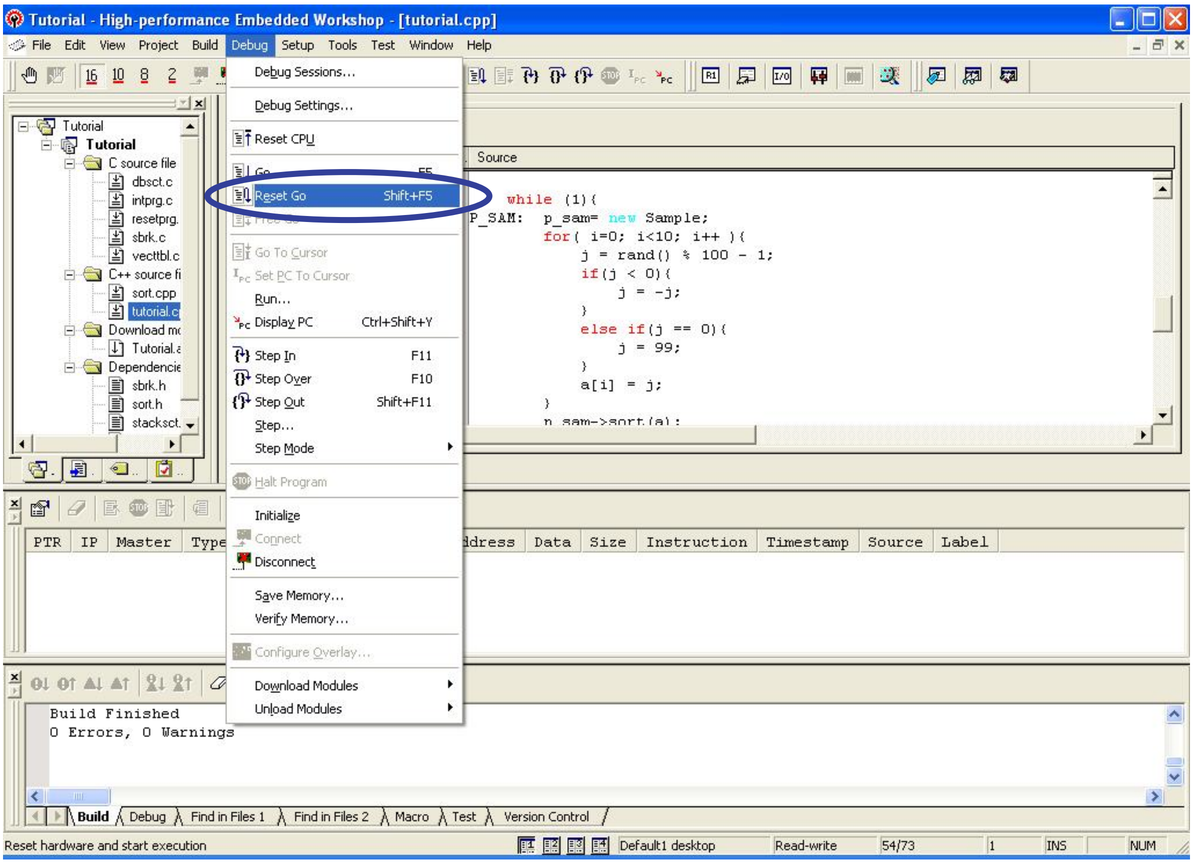


```
Line  Source Address  Event  S..  Source
 35
 36   00001024                        while (1){
 37   00001028                    P_SAM:  p_sam= new Sample;
 38   0000106E                            for( i=0; i<10; i++ ){
 39   00001032                                j = rand() % 100 - 1;
 40   00001056                                if(j < 0){
 41   0000105A                                    j = -j;
 42                                           }
 43   0000105E                                else if(j == 0){
 44   00001062                                    j = 99;
 45                                           }
 46   00001064                                a[i] = j;
 47                                       }
 48   00001072                            p_sam->sort(a);
```

PTR	IP	Master	Type	BranchType	Bus	R/W	Address	Data	Size	Instruction	Timestamp	Source	Label

```
Build Finished
0 Errors, 0 Warnings
```

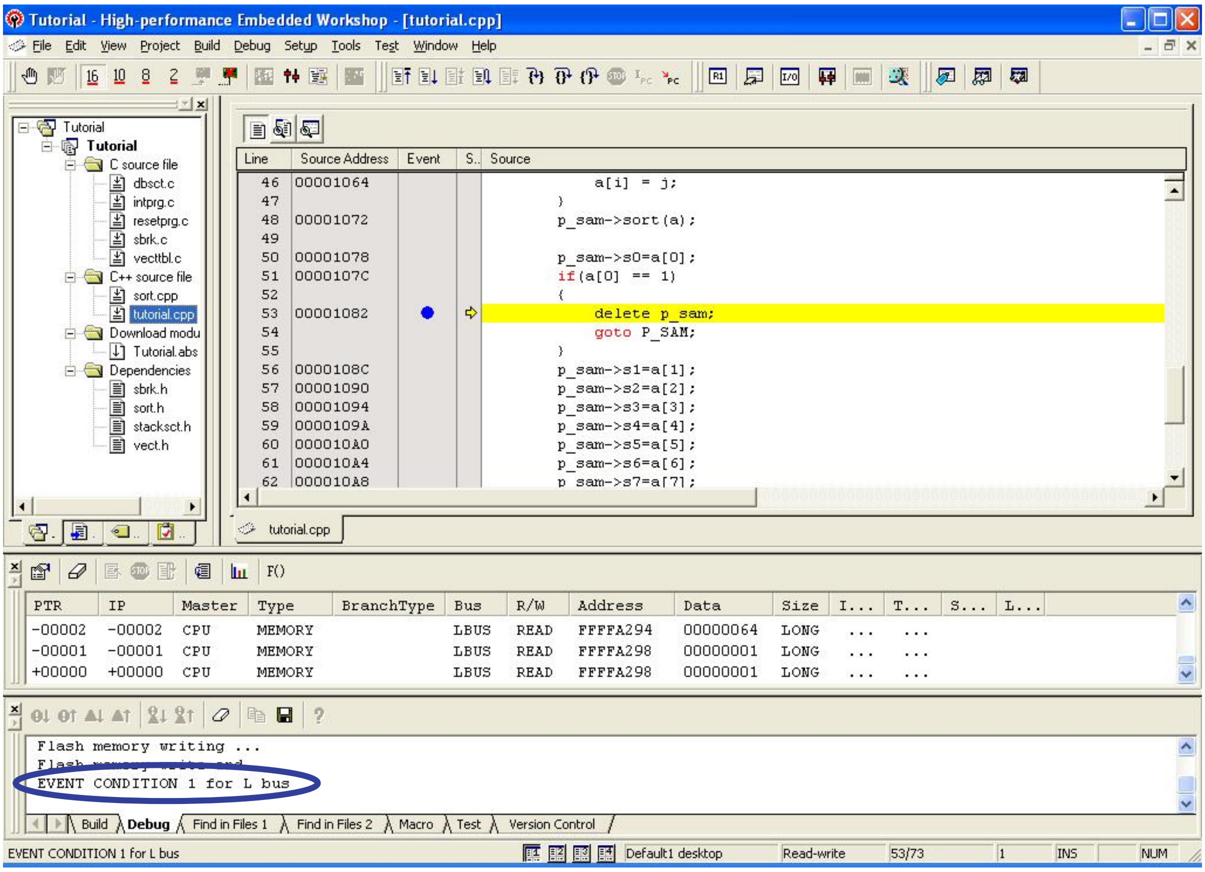
(4) From the Debug menu, choose Run After Reset to execute the program.



```
                        while (1){
                    P_SAM:  p_sam= new Sample;
                            for( i=0; i<10; i++ ){
                                j = rand() % 100 - 1;
                                if(j < 0){
                                    j = -j;
                                }
                                else if(j == 0){
                                    j = 99;
                                }
                                a[i] = j;
                            }
                            p_sam->sort(a);
```

```
Build Finished
0 Errors, 0 Warnings
```

(5) The program will be made to break (halt) by a hardware break, with a message “EVENT CONDITION 1 for L bus” displayed on the Debug tab pane of the output window.
The source window shows the place at which the program halted. The yellow arrow shows the position of the program counter, with the source line part highlighted in yellow color. At this time, a program execution history is displayed in the trace window.



Line	Source Address	Event	S..	Source
46	00001064			`a[i] = j;`
47				`}`
48	00001072			`p_sam->sort(a);`
49				
50	00001078			`p_sam->s0=a[0];`
51	0000107C			`if(a[0] == 1)`
52				`{`
53	00001082	●	⇨	`delete p_sam;`
54				`goto P_SAM;`
55				`}`
56	0000108C			`p_sam->s1=a[1];`
57	00001090			`p_sam->s2=a[2];`
58	00001094			`p_sam->s3=a[3];`
59	0000109A			`p_sam->s4=a[4];`
60	000010A0			`p_sam->s5=a[5];`
61	000010A4			`p_sam->s6=a[6];`
62	000010A8			`p_sam->s7=a[7];`

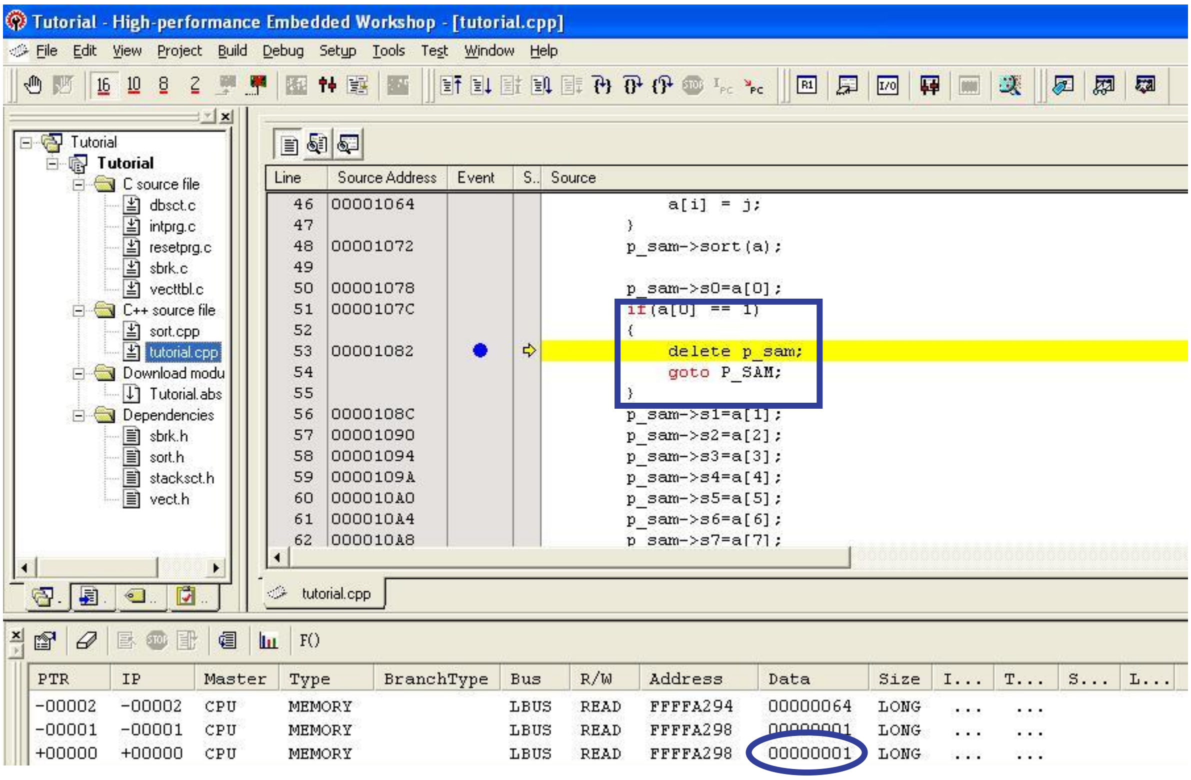
PTR	IP	Master	Type	BranchType	Bus	R/W	Address	Data	Size	I...	T...	S...	L...
-00002	-00002	CPU	MEMORY		LBUS	READ	FFFFA294	00000064	LONG	...	...		
-00001	-00001	CPU	MEMORY		LBUS	READ	FFFFA298	00000001	LONG	...	...		
+00000	+00000	CPU	MEMORY		LBUS	READ	FFFFA298	00000001	LONG	...	...		

```
Flash memory writing ...
Flash memory write end
EVENT CONDITION 1 for L bus
```

EVENT CONDITION 1 for L bus | Default1 desktop | Read-write | 53/73 | 1 | INS | NUM

(6) Since the event point is set in an execution statement for the a[0] value = 1 case, the program is made to break when the value of a[0] is 1.
However, this alone does not tell whether the value of a[0] is the one converted from -1 to 1 or its value was 1 from the beginning.

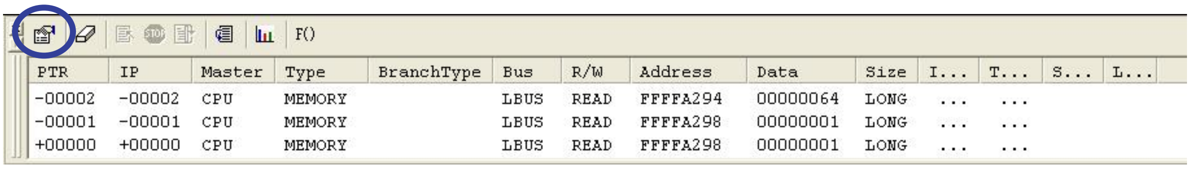
Therefore, we'll set a sequential break as shown below to cause the program to break when the value of a[0] is converted from -1 to 1.



Line	Source Address	Event	S..	Source
46	00001064			a[i] = j;
47				}
48	00001072			p_sam->sort(a);
49				
50	00001078			p_sam->s0=a[0];
51	0000107C			if(a[0] == 1)
52				{
53	00001082	●	⇨	delete p_sam;
54				goto P_SAM;
55				}
56	0000108C			p_sam->s1=a[1];
57	00001090			p_sam->s2=a[2];
58	00001094			p_sam->s3=a[3];
59	0000109A			p_sam->s4=a[4];
60	000010A0			p_sam->s5=a[5];
61	000010A4			p_sam->s6=a[6];
62	000010A8			p_sam->s7=a[7];

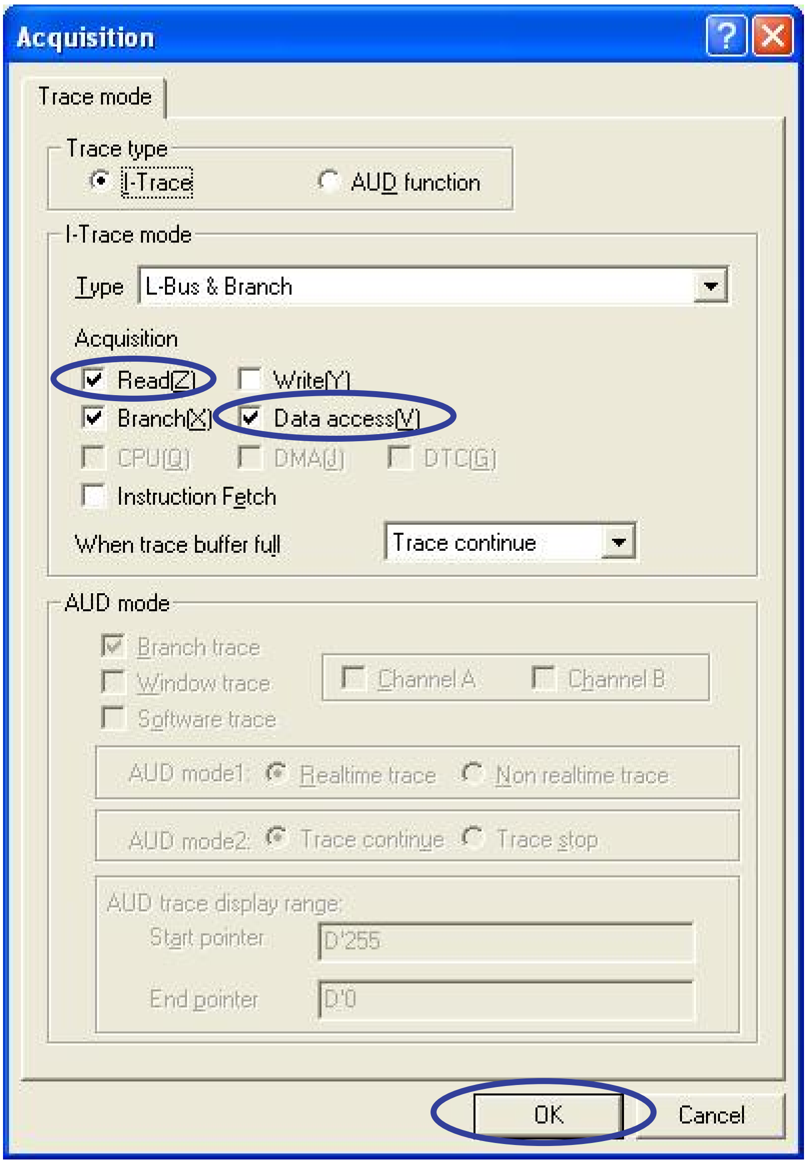
PTR	IP	Master	Type	BranchType	Bus	R/W	Address	Data	Size	I...	T...	S...	L...
-00002	-00002	CPU	MEMORY		LBUS	READ	FFFFA294	00000064	LONG	...	...		
-00001	-00001	CPU	MEMORY		LBUS	READ	FFFFA298	00000001	LONG	...	...		
+00000	+00000	CPU	MEMORY		LBUS	READ	FFFFA298	00000001	LONG	...	...		

Here, if the content of the trace window does not look like the one displayed in the preceding page, select the Setup toolbar button.



PTR	IP	Master	Type	BranchType	Bus	R/W	Address	Data	Size	I...	T...	S...	L...
-00002	-00002	CPU	MEMORY		LBUS	READ	FFFFA294	00000064	LONG	...	...		
-00001	-00001	CPU	MEMORY		LBUS	READ	FFFFA298	00000001	LONG	...	...		
+00000	+00000	CPU	MEMORY		LBUS	READ	FFFFA298	00000001	LONG	...	...		

The Acquisition dialog box will be displayed. Check that "Read" and "Data Access" are selected, and then click OK.
Then, choose Run After Reset from the Debug menu again to execute the program.



Acquisition

Trace mode

Trace type
● I-Trace ○ AUD function

I-Trace mode
Type: L-Bus & Branch

Acquisition
☑ Read(Z) □ Write(Y)
☑ Branch(X) ☑ Data access(V)
□ CPU(Q) □ DMA(J) □ DTC(G)
□ Instruction Fetch

When trace buffer full: Trace continue

AUD mode
☑ Branch trace
□ Window trace
□ Software trace
□ Channel A □ Channel B

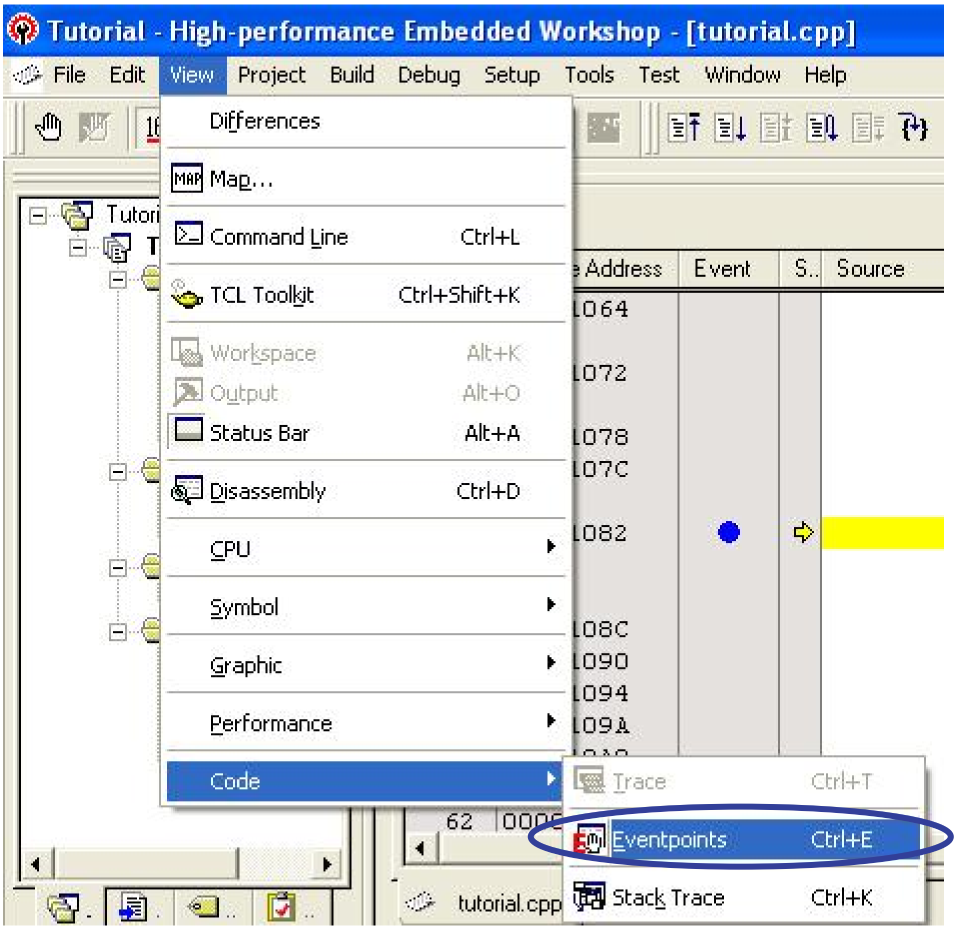
AUD mode1: ● Realtime trace ○ Non realtime trace

AUD mode2: ● Trace continue ○ Trace stop

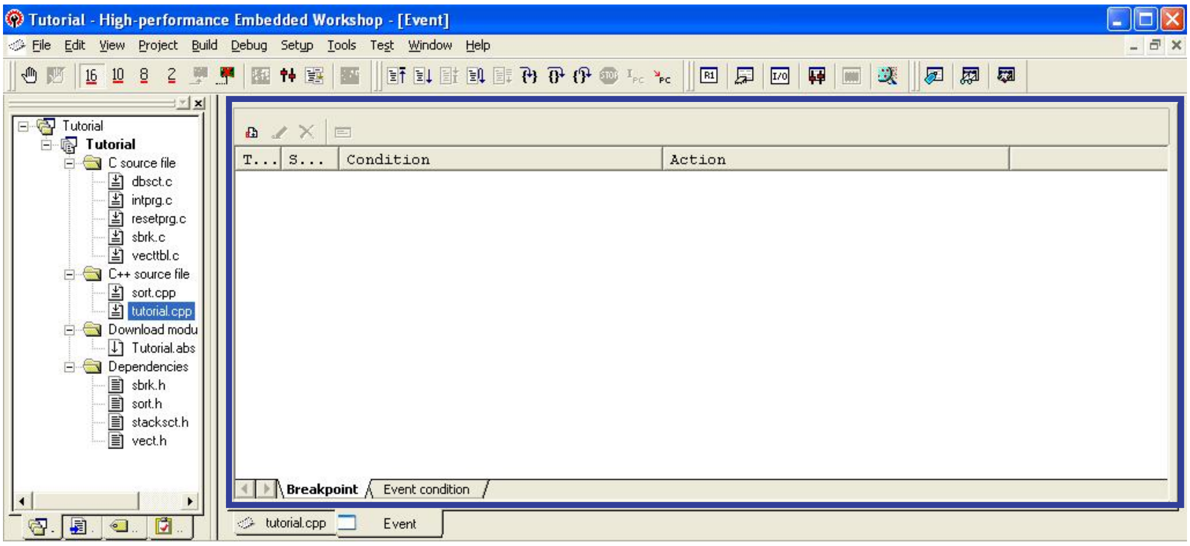
AUD trace display range:
Start pointer: D'255
End pointer: D'0

OK Cancel

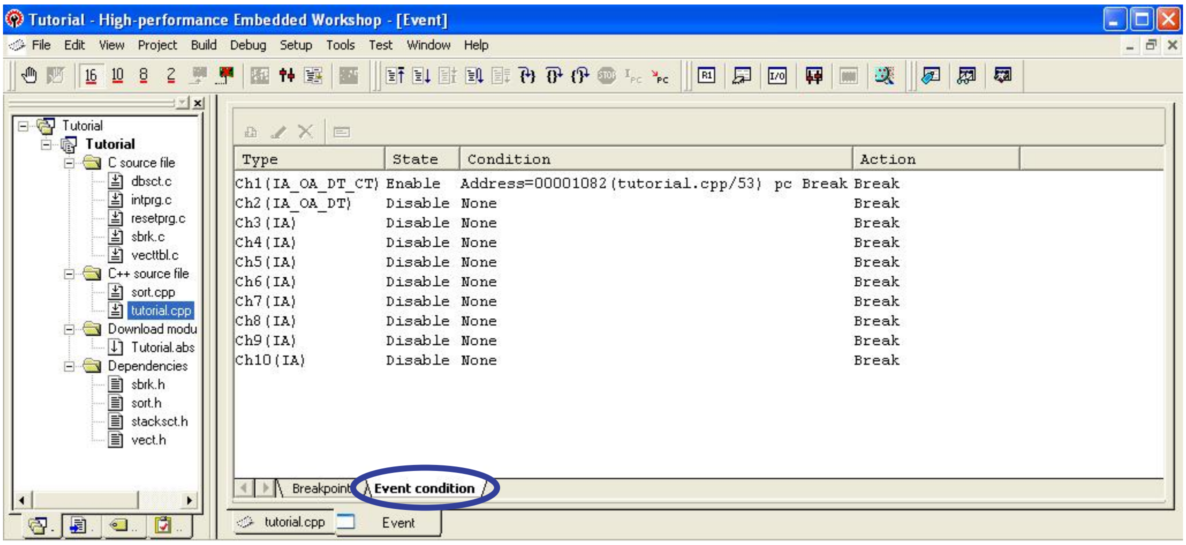
(7) We'll now set a sequential break.
From the View menu, choose Code and then Event Point.



(8) The event point window will be displayed. The diagram shown below depicts the state of this window with its docking view turned off by right-clicking in the window.

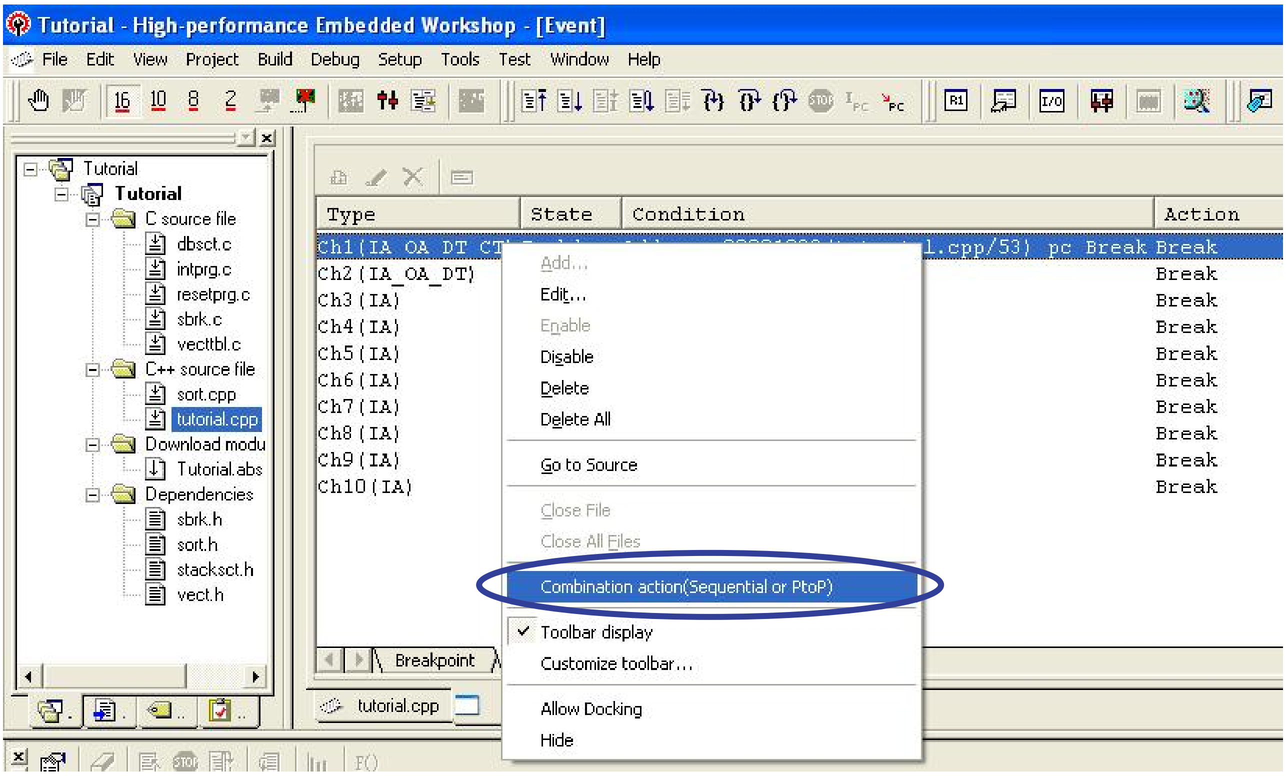


(9) Change the tabs of the event point window to the Event Condition tab to display event conditions.

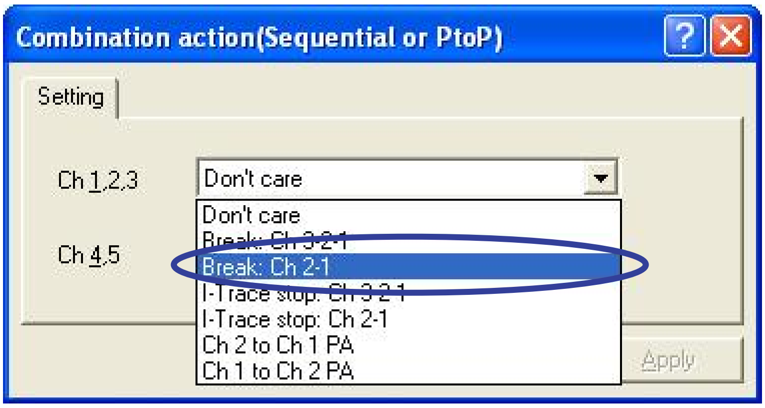


Type	State	Condition	Action
Ch1(IA_OA_DT_CT)	Enable	Address=00001082(tutorial.cpp/53) pc Break	Break
Ch2(IA_OA_DT)	Disable	None	Break
Ch3(IA)	Disable	None	Break
Ch4(IA)	Disable	None	Break
Ch5(IA)	Disable	None	Break
Ch6(IA)	Disable	None	Break
Ch7(IA)	Disable	None	Break
Ch8(IA)	Disable	None	Break
Ch9(IA)	Disable	None	Break
Ch10(IA)	Disable	None	Break

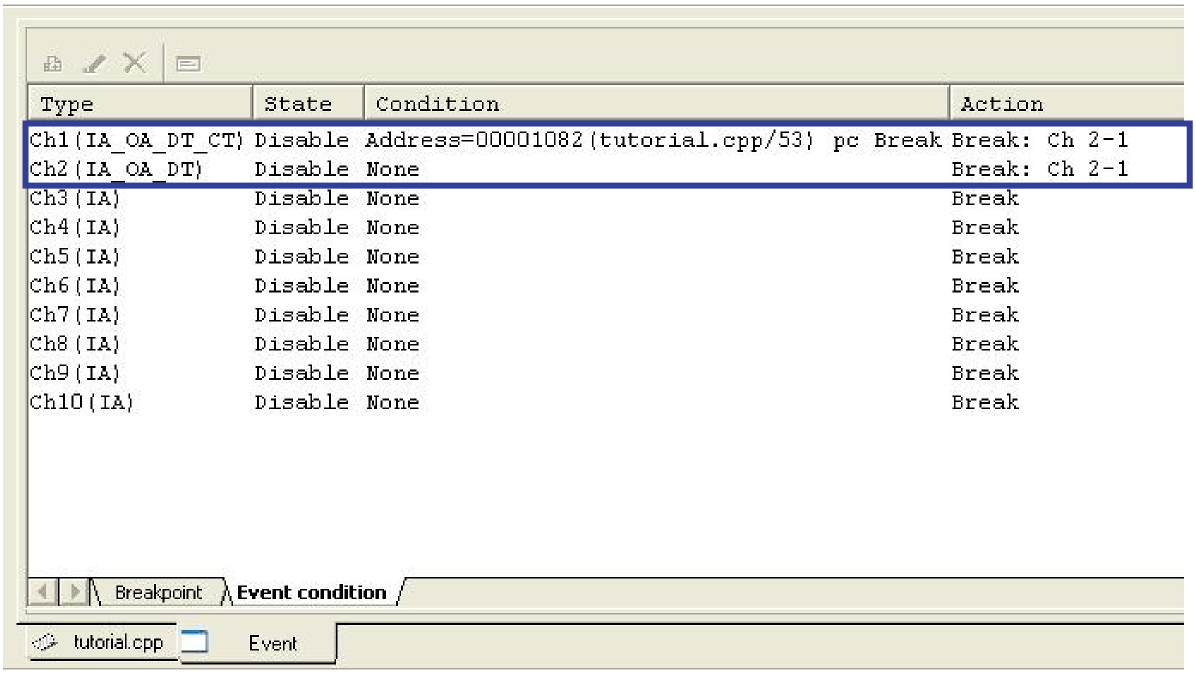
(10) Right-click in the event point window and choose “Combination Action (Sequential or PtoP)” from the ensuing menu.



(11) The Combination Action (Sequential or PtoP) dialog box will be displayed. In this dialog box, select "Break: Ch2–1)" from options in the Ch 1,2,3 list box and click OK.
This sets a sequential break that will cause the program to break when the condition of Ch1 is met after the condition of Ch2 is met.

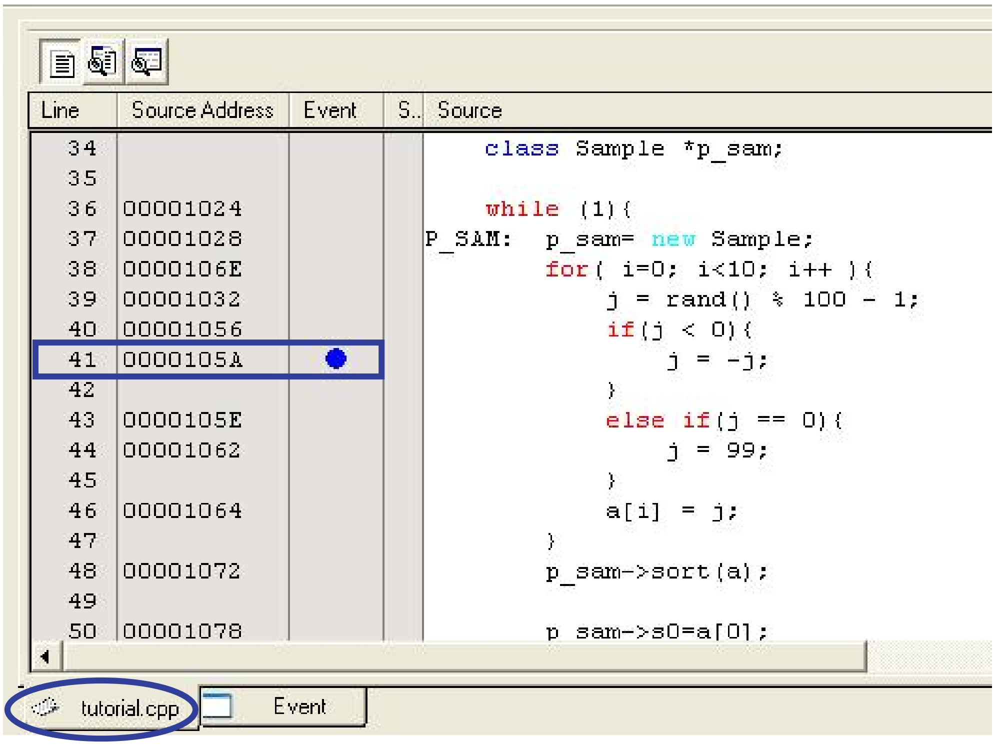


(12) The Event Condition list of the event point window will now look like the one shown below. So far only Ch1 has been set.



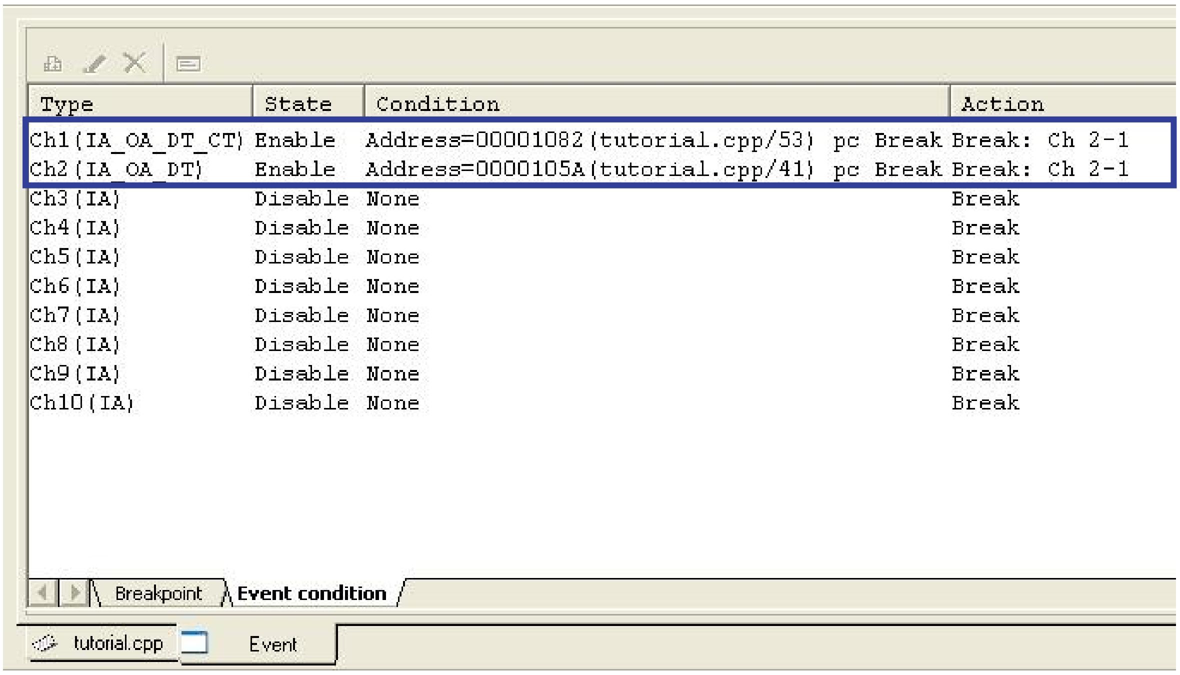
Type	State	Condition	Action
Ch1(IA_OA_DT_CT)	Disable	Address=00001082(tutorial.cpp/53) pc Break	Break: Ch 2-1
Ch2(IA_OA_DT)	Disable	None	Break: Ch 2-1
Ch3(IA)	Disable	None	Break
Ch4(IA)	Disable	None	Break
Ch5(IA)	Disable	None	Break
Ch6(IA)	Disable	None	Break
Ch7(IA)	Disable	None	Break
Ch8(IA)	Disable	None	Break
Ch9(IA)	Disable	None	Break
Ch10(IA)	Disable	None	Break

(13) Select “tutorial.ccp” to display the source file. Double-click in the Event column at the 41st line position of the source file and set a new event point. Written here is an execution statement that converts the value of j to a positive value (1) when its value is negative (-1).



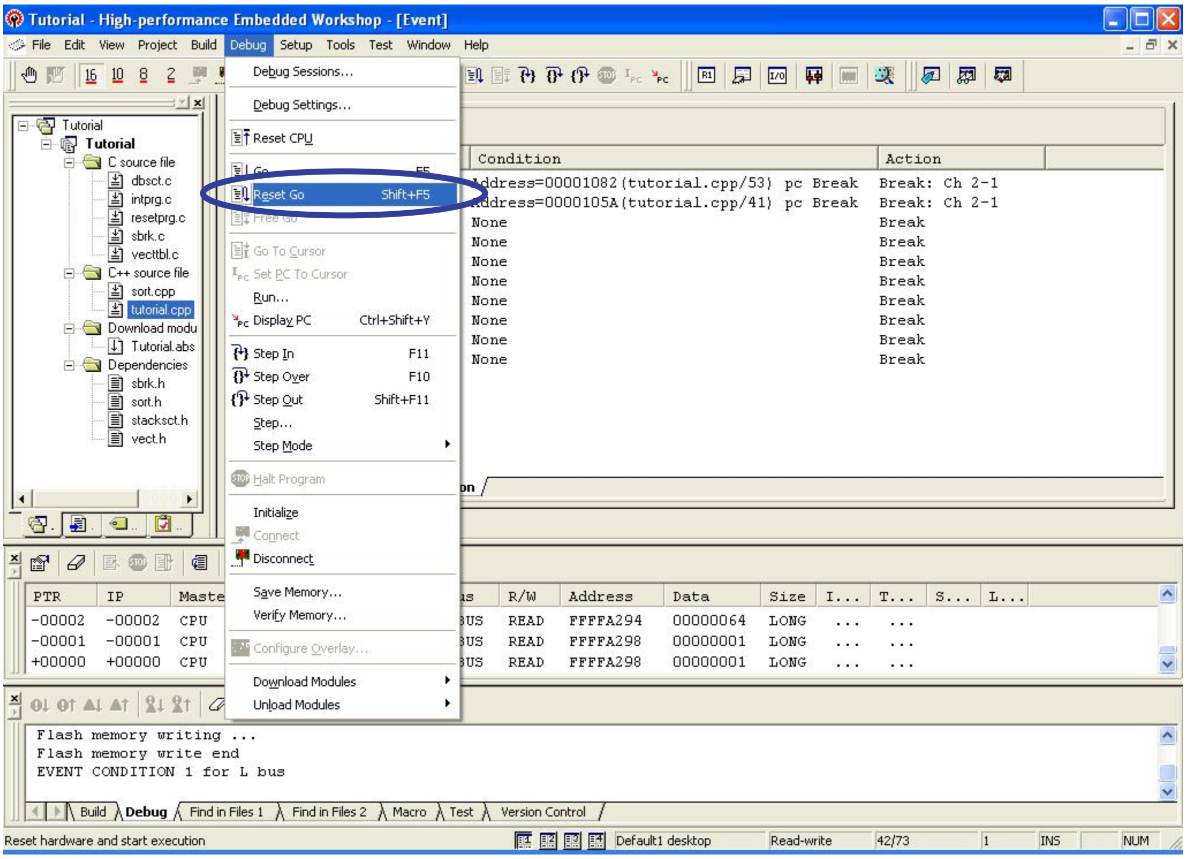
```
Line  Source Address  Event  S..  Source
 34                                  class Sample *p_sam;
 35
 36   00001024                       while (1){
 37   00001028                  P_SAM:  p_sam= new Sample;
 38   0000106E                          for( i=0; i<10; i++ ){
 39   00001032                              j = rand() % 100 - 1;
 40   00001056                              if(j < 0){
 41   0000105A        ●                         j = -j;
 42                                          }
 43   0000105E                              else if(j == 0){
 44   00001062                                  j = 99;
 45                                          }
 46   00001064                              a[i] = j;
 47                                      }
 48   00001072                          p_sam->sort(a);
 49
 50   00001078                          p_sam->s0=a[0];
```

(14) Select Event Point to display the event point window. The Event Condition list of the event point window will now look like the one shown below.
The event point you’ve set above is set for Ch2.
The program will break when the condition of Ch1 is met after the condition of Ch2 is met.
Be aware that if any event points other than a sequential break are set, there is a possibility that the intended sequential break will not work. Therefore, do not set any event points other than a sequential break.



```
Type              State    Condition                                   Action
Ch1(IA_OA_DT_CT)  Enable   Address=00001082(tutorial.cpp/53) pc Break  Break: Ch 2-1
Ch2(IA_OA_DT)     Enable   Address=0000105A(tutorial.cpp/41) pc Break  Break: Ch 2-1
Ch3(IA)           Disable  None                                        Break
Ch4(IA)           Disable  None                                        Break
Ch5(IA)           Disable  None                                        Break
Ch6(IA)           Disable  None                                        Break
Ch7(IA)           Disable  None                                        Break
Ch8(IA)           Disable  None                                        Break
Ch9(IA)           Disable  None                                        Break
Ch10(IA)          Disable  None                                        Break
```

(15) Choose Run After Reset from the Debug menu to execute the program.



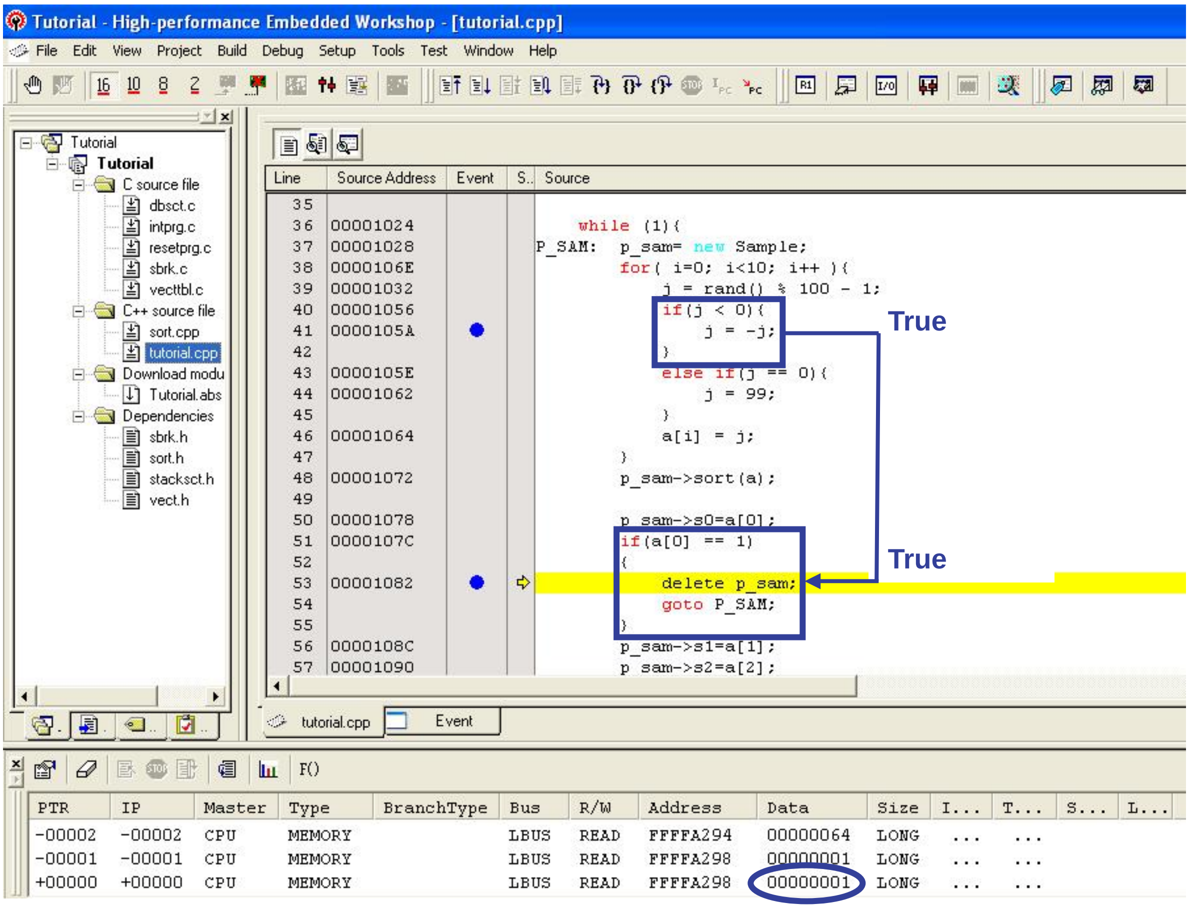
Condition	Action
ddress=00001082(tutorial.cpp/53) pc Break	Break: Ch 2-1
ddress=0000105A(tutorial.cpp/41) pc Break	Break: Ch 2-1
None	Break
None	Break
None	Break
None	Break
None	Break
None	Break
None	Break
None	Break

PTR	IP	Maste	s	R/W	Address	Data	Size	I...	T...	S...	L...
-00002	-00002	CPU	US	READ	FFFFA294	00000064	LONG	...	...		
-00001	-00001	CPU	US	READ	FFFFA298	00000001	LONG	...	...		
+00000	+00000	CPU	US	READ	FFFFA298	00000001	LONG	...	...		

```
Flash memory writing ...
Flash memory write end
EVENT CONDITION 1 for L bus
```

Reset hardware and start execution | Default1 desktop | Read-write | 42/73 | 1 | INS | NUM

(16) This time too, the value of a[0] is 1.
However, since we've made sequential settings above so as to break the program when the execution statement for the a[0] value on Ch1 = 1 case is true after the execution statement that converts the value of j on Ch2 to a positive value (1) when its value is negative (-1) is true, we know that the value of a[0] is the one converted from -1 to 1.



```
Source Address  Event  S..  Source

00001024                        while (1){
00001028                P_SAM:  p_sam= new Sample;
0000106E                        for( i=0; i<10; i++ ){
00001032                            j = rand() % 100 - 1;
00001056                            if(j < 0){
0000105A          ●                     j = -j;
                                    }
0000105E                            else if(j == 0){
00001062                                j = 99;
                                    }
00001064                            a[i] = j;
                                }
00001072                        p_sam->sort(a);

00001078                        p_sam->s0=a[0];
0000107C                        if(a[0] == 1)
                                {
00001082          ●      ⇨          delete p_sam;
                                    goto P_SAM;
                                }
0000108C                        p_sam->s1=a[1];
00001090                        p_sam->s2=a[2];
```

```
PTR     IP      Master  Type    BranchType  Bus   R/W   Address   Data      Size  I...  T...  S...  L...
-00002  -00002  CPU     MEMORY              LBUS  READ  FFFFA294  00000064  LONG  ...   ...
-00001  -00001  CPU     MEMORY              LBUS  READ  FFFFA298  00000001  LONG  ...   ...
+00000  +00000  CPU     MEMORY              LBUS  READ  FFFFA298  00000001  LONG  ...   ...
```

Since the program breaks when multiple conditions are met as described here, it is possible to debug program failures or hardware failures occurring in a limited situation efficiently.

6. Related Documents

The E10A-USB emulator and the HEW have numerous other convenient facilities and features not discussed in this document. Detailed specification, technical information, limitations and other useful information on each product are described in the related documents listed below. Please see these manuals along with this document.

[E10A-USB emulator related documents]

- SuperH Family E10A-USB Emulator User's Manual
- SuperH Family E10A-USB Emulator User's Manual, Separate Volume
  (Supplementary explanation to be referred to when using the SH7125 series debug MCU board)
- SuperH Family E10A-USB Emulator User's Manual, Separate Volume
  (Supplementary explanation to be referred to when using the SH7125 and SH7124)
- Limitations of the SuperH Family E10A-USB Emulator

[High-performance Embedded Workshop related documents]

- High-performance Embedded Workshop User's Manual
- High-performance Embedded Workshop Release Notes

[CPU related documents]

- SH7125 and SH7124 Group Hardware Manual
- SH-1/SH-2/SH-DSP Software Manual

[SuperH family C/C++ compiler related documents]

- SuperH C/C++ Compiler Package User's Manual

To see more information on the E10A-USB emulator, please visit the Renesas websites given below.

Japan site:	http://japan.renesas.com/e10a_usb
Global site:	http://www.renesas.com/e10a_usb

Home Page and Where to Contact for Support

Renesas Technology home page

http://www.renesas.com/

Where to contact

http://www.renesas.com/inquiry

Revision Record

Rev.	Issue date	Contents of revision	
		Page	Points
1.00	2009.02.13	—	First edition issued

Notes regarding these materials

1. 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.
2. 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)
5. 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.
8. 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.
9. 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.
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.