お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害などを生じさせないようお客様の責任において冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、詳細につきましては製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

注 1. 本資料において使用されている「当社」とは、ルネサスエレクトロニクス株式会社およびルネサスエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

お客様各位

資料中の「日立製作所」、「日立XX」等名称の株式会社ルネサス テクノロジへの変更について

2003年4月1日を以って三菱電機株式会社及び株式会社日立製作所のマイコン、ロジック、アナログ、ディスクリート半導体、及びDRAMを除くメモリ（フラッシュメモリ・SRAM等）を含む半導体事業は株式会社ルネサス テクノロジに承継されました。従いまして、本資料中には「日立製作所」、「株式会社日立製作所」、「日立半導体」、「日立XX」といった表記が残っておりますが、これらの表記は全て「株式会社ルネサス テクノロジ」に変更されておりますのでご理解の程お願い致します。尚、会社商標・ロゴ・コーポレートステートメント以外の内容については一切変更しておりませんので資料としての内容更新ではありません。

ルネサステクノロジ ホームページ（http://www.renesas.com）

2003年4月1日
株式会社ルネサス テクノロジ
カスタマサポート部

RENESAS

ご注意

安全設計に関するお願い

1. 弊社は品質、信頼性の向上に努めておりますが、半導体製品は故障が発生したり、誤動作する場合があります。弊社の半導体製品の故障又は誤動作によって結果として、人身事故、火災事故、社会的損害などを生じさせないような安全性を考慮した冗長設計、延焼対策設計、誤動作防止設計などの安全設計に十分ご留意ください。

本資料ご利用に際しての留意事項

1. 本資料は、お客様が用途に応じた適切なルネサス テクノロジ製品をご購入いただくための参考資料であり、本資料中に記載の技術情報についてルネサス テクノロジが所有する知的財産権その他の権利の実施、使用を許諾するものではありません。
2. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例の使用に起因する損害、第三者所有の権利に対する侵害に関し、ルネサス テクノロジは責任を負いません。
3. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他全ての情報は本資料発行時点のものであり、ルネサス テクノロジは、予告なしに、本資料に記載した製品または仕様を変更することがあります。ルネサス テクノロジ半導体製品のご購入に当たりましては、事前にルネサス テクノロジ、ルネサス販売または特約店へ最新の情報をご確認頂きますとともに、ルネサス テクノロジホームページ (http://www.renesas.com)などを通じて公開される情報に常にご注意ください。
4. 本資料に記載した情報は、正確を期すため、慎重に制作したものですが万一本資料の記述誤りに起因する損害がお客様に生じた場合には、ルネサス テクノロジはその責任を負いません。
5. 本資料に記載の製品データ、図、表に示す技術的な内容、プログラム及びアルゴリズムを流用する場合は、技術内容、プログラム、アルゴリズム単位で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断してください。ルネサス テクノロジは、適用可否に対する責任は負いません。
6. 本資料に記載された製品は、人命にかかわるような状況の下で使用される機器あるいはシステムに用いられることを目的として設計、製造されたものではありません。本資料に記載の製品を運輸、移動体用、医療用、航空宇宙用、原子力制御用、海底中継用機器あるいはシステムなど、特殊用途へのご利用をご検討の際には、ルネサス テクノロジ、ルネサス販売または特約店へご照会ください。
7. 本資料の転載、複製については、文書によるルネサス テクノロジの事前の承諾が必要です。
8. 本資料に関し詳細についてのお問い合わせ、その他お気付きの点がございましたらルネサス テクノロジ、ルネサス販売または特約店までご照会ください。

User's Manual

RENESAS

SuperH RISC engine
High-performance Embedded Workshop 2
チュートリアル

ルネサス エレクトロニクス
www.renesas.com

Rev.3.00　2002.12

ご注意

1. 本書に記載の製品及び技術のうち「外国為替及び外国貿易法」に基づき安全保障貿易管理関連貨物・技術に該当するものを輸出する場合，または国外に持ち出す場合は日本国政府の許可が必要です。
2. 本書に記載された情報の使用に際して，弊社もしくは第三者の特許権，著作権，商標権，その他の知的所有権等の権利に対する保証または実施権の許諾を行うものではありません。また本書に記載された情報を使用した事により第三者の知的所有権等の権利に関わる問題が生じた場合，弊社はその責を負いませんので予めご了承ください。
3. 製品及び製品仕様は予告無く変更する場合がありますので，最終的な設計，ご購入，ご使用に際しましては，事前に最新の製品規格または仕様書をお求めになりご確認ください。
4. 弊社は品質・信頼性の向上に努めておりますが，宇宙，航空，原子力，燃焼制御，運輸，交通，各種安全装置， ライフサポート関連の医療機器等のように，特別な品質・信頼性が要求され，その故障や誤動作が直接人命を脅かしたり，人体に危害を及ぼす恐れのある用途にご使用をお考えのお客様は，事前に弊社営業担当迄ご相談をお願い致します。
5. 設計に際しては，特に最大定格，動作電源電圧範囲，放熱特性，実装条件及びその他諸条件につきましては，弊社保証範囲内でご使用いただきますようお願い致します。
   保証値を越えてご使用された場合の故障及び事故につきましては，弊社はその責を負いません。
   また保証値内のご使用であっても半導体製品について通常予測される故障発生率，故障モードをご考慮の上，弊社製品の動作が原因でご使用機器が人身事故，火災事故，その他の拡大損害を生じないようにフェールセーフ等のシステム上の対策を講じて頂きますようお願い致します。
6. 本製品は耐放射線設計をしておりません。
7. 本書の一部または全部を弊社の文書による承認なしに転載または複製することを堅くお断り致します。
8. 本書をはじめ弊社半導体についてのお問い合わせ，ご相談は弊社営業担当迄お願い致します。

はじめに

本チュートリアルは、Hitachi Embedded Workshop 2（以下、HEWと略します）の簡単な使用方法を述べたものです。HEWの起動、プロジェクトの作成・編集方法から、アプリケーションの作成およびシミュレータデバッガを利用した一連のデバッグ作業について説明します。HEWの詳細機能をはじめ、C/C++コンパイラ、アセンブラ、最適化リンケージエディタ、およびシミュレータデバッガについては、下記のマニュアルを参照してください。

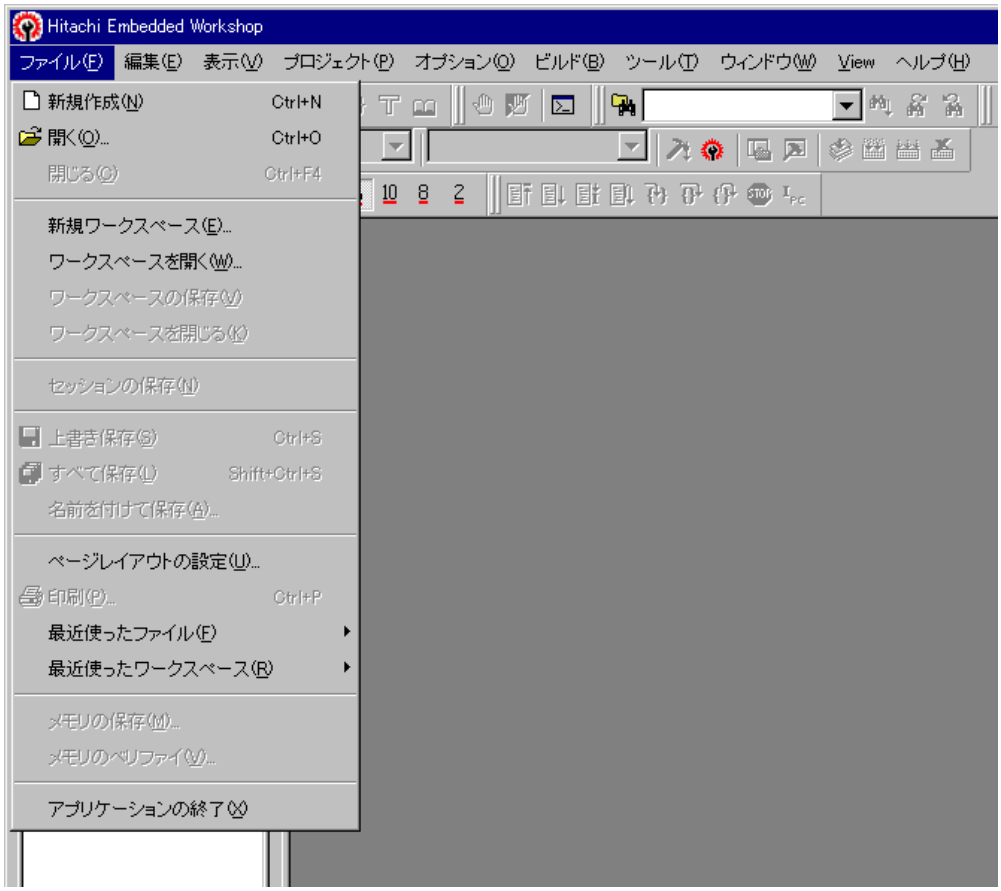
- SuperH™ RISC engine Hitachi Embedded Workshop 2 ユーザーズマニュアル
- SuperH™ RISC engine C/C++コンパイラ、アセンブラ、最適化リンケージエディタ ユーザーズマニュアル

SuperH™ RISC engineマイコンの詳細については、該当品種のプログラミングマニュアルおよびハードウェアマニュアルを参照してください。

なお、以降の説明はSuperH™ RISC engine C/C++コンパイラパッケージに含まれる全てのツールをインストールした状態で進めます。

また、[　　]内で -> が記述されている場合、->の左はメニューを示し、右はそのメニュー上の選択項目を示します。

例えば、図ＩのFileメニュー上の項目であるExitは、[File -> Exit]で表記します。



図Ｉ　HEWのメニュー

注意

本チュートリアルに記載しているHEWの画面は日本語版Windows®上で取得したものです。

目次

1　ＨＥＷの起動

HEWのインストーラは、インストール正常終了時、Windows®のスタートメニューのプログラムフォルダの下に”Hitachi Embedded Workshop 2”という名称のフォルダを作成し、そのフォルダ内にHEWの実行プログラムである”Hitachi Embedded Workshop 2”の他、各種サポート情報のショートカットを登録します。

なお、スタートメニューの表示内容は、ツールのインストール状況により異なる場合があります。

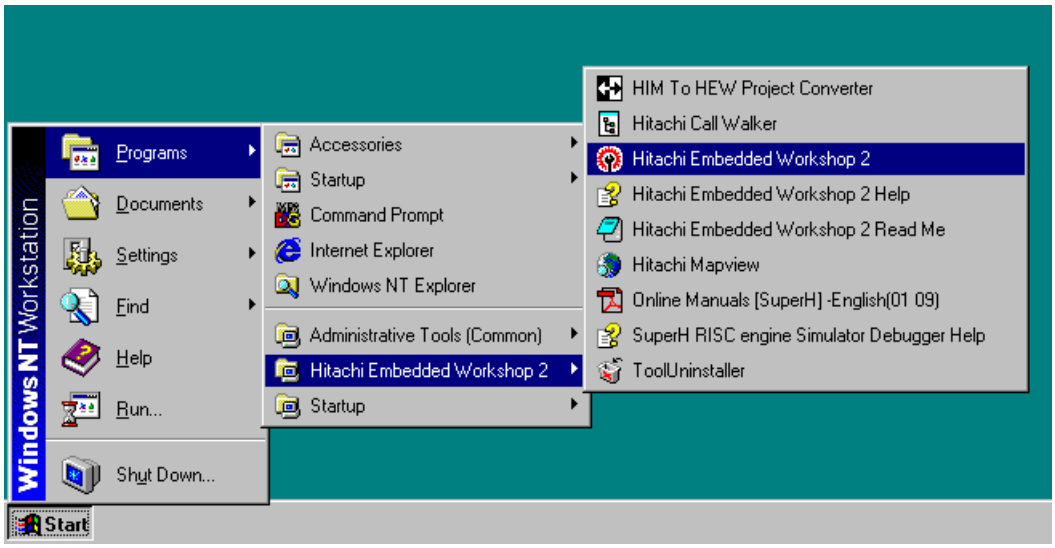


図 1.1 スタートメニューによるＨＥＷ起動

このスタートメニューで、”Hitachi Embedded Workshop 2”をクリックすると起動メッセージを表示し、引き続き[ようこそ!]ダイアログボックス（図 1.2）を表示します。なお、前の作業で開いていたプロジェクトを直接開きたい場合などは、[ツール -> オプション...]で作業環境の設定変更が可能です。詳しくは、「Hitachi Embedded Workshop 2 ユーザーズマニュアル」の「6章 環境のカスタマイズ」を参照してください。

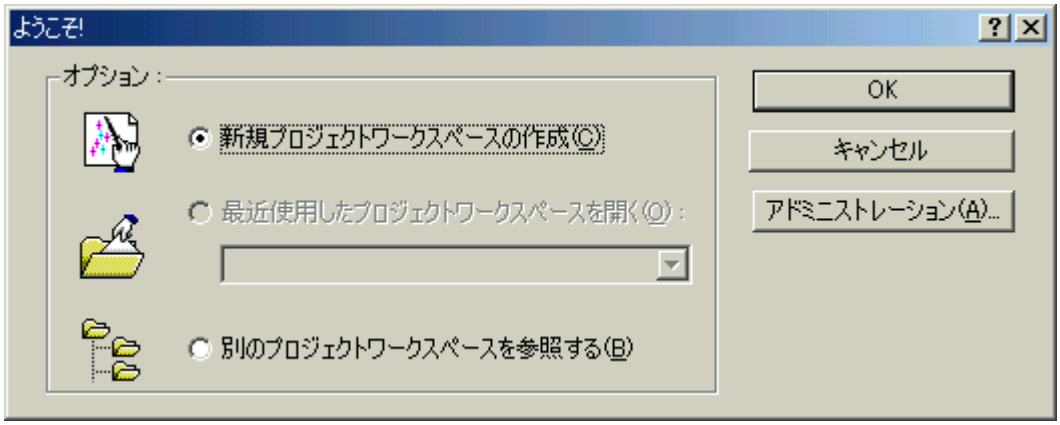


図 1.2 ようこそ!ダイアログボックス

HEWを初めて使用する場合や、新たにプロジェクトを作成して作業を開始する場合は、[新規プロジェクトワークスペースの作成]を選択して[OK]をクリックしてください。既に作成したプロジェクトで作業する場合は、[最近使用したプロジェクトワークスペースを開く]または[別のプロジェクトワークスペースを参照する]を選択して[OK]をクリックしてください。

ここでは、[新規プロジェクトワークスペースの作成]を選択して[OK]をクリックしてください。

2　プロジェクトの作成

2.1　新規ワークスペースを作成する

[ようこそ!]ダイアログボックスで[新規プロジェクトワークスペースの作成]を選択して[OK]をクリックすると、新しいワークスペースとプロジェクト作成用の[New Project Workspace]ダイアログボックス（図 2.1）を表示します。このダイアログボックスでワークスペース名（新規作成時はプロジェクト名もデフォルトで同名です）やCPUの種類、プロジェクトのタイプ（表 2.1）などを設定します。

例えば、[ワークスペース名]にワークスペース名として“ tutorial ”と入力すると、[プロジェクト名]も”tutorial”になり、[ディレクトリ]も“ c:\hew2\tutorial ”となります。プロジェクト名を変更する場合は、[プロジェクト名]に直接入力し、ワークスペースとして使用するディレクトリを変更する場合は、[参照...]をクリックしてディレクトリを選択するか、直接[ディレクトリ]に入力してください。

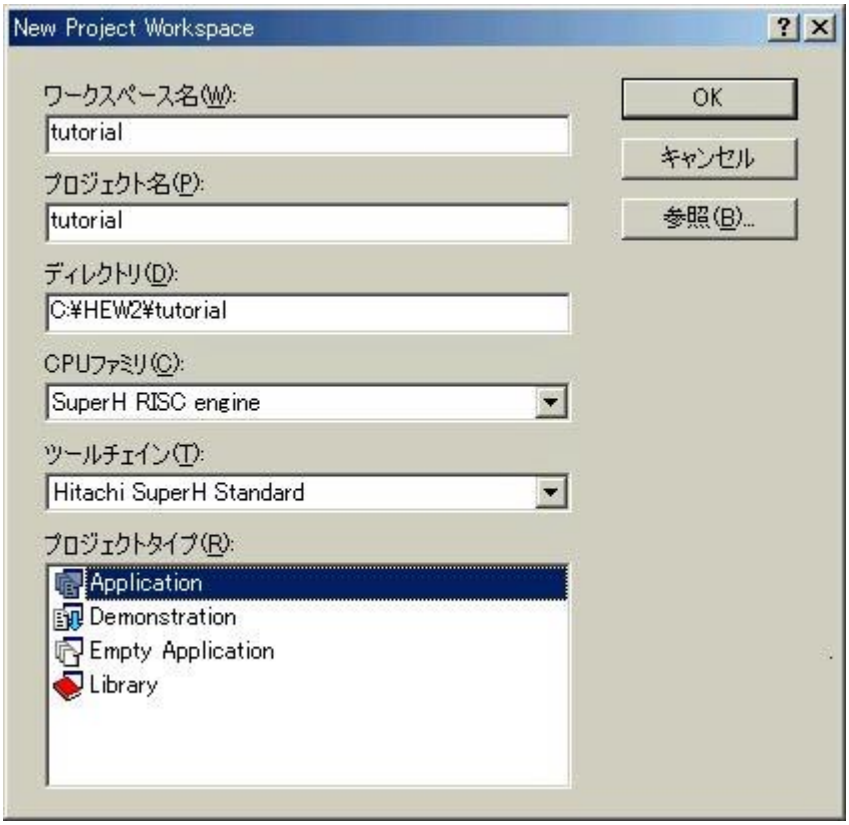


図 2.1 New Project Workspace**ダイアログボックス**

表 2.1 **プロジェクトタイプ**

プロジェクトタイプ	説明
Application	C/C++言語またはアセンブリ言語で書かれた初期ルーチンファイルを含む実行プログラムを作成するためのプロジェクト
Demonstration	C言語またはアセンブリ言語で書かれたデモンストレーション用プログラムを作成するためのプロジェクト
Empty Application	ツールチェイン環境の設定のみのプロジェクト（生成ファイルなし）
Library	ライブラリファイルを作成するためのプロジェクト（生成ファイルなし）

2.2　CPUの選択

“New Project Workspace”ダイアログボックスで[OK]をクリックすると、プロジェクトジェネレータを起動します。最初に使用するCPUを選択します。使用するCPUの種類（[CPU Type]）はCPUのシリーズ（[CPU Series]）ごとに分類しています。［CPU Series:］および［CPU Type:］の選択により、生成するファイルが異るので、開発するプログラムの対象となるCPUタイプを選択してください。選択したいCPUタイプがない場合は、ハードウェア仕様の近いCPUタイプまたは“ Other ”を選択してください。

- ［Next >］をクリックすると、次の画面を表示します。
- ［< Back］をクリックすると、この画面を表示する前の画面またはダイアログボックスを表示します。
- ［Finish］をクリックすると、”Summary”ダイアログボックスが開きます。
- ［Cancel］をクリックすると、”New Workspace”ダイアログボックスに戻ります。

［< Back］、［Next >］、［Finish］および［Cancel］の機能は、このウィザードダイアログボックスで共通の機能です。

ここでは、[CPU Series]で“ SH-1 ”を、[CPU Type]から“ SH7020 ”を選択（図 2.2）して、［Next >］をクリックしてください。

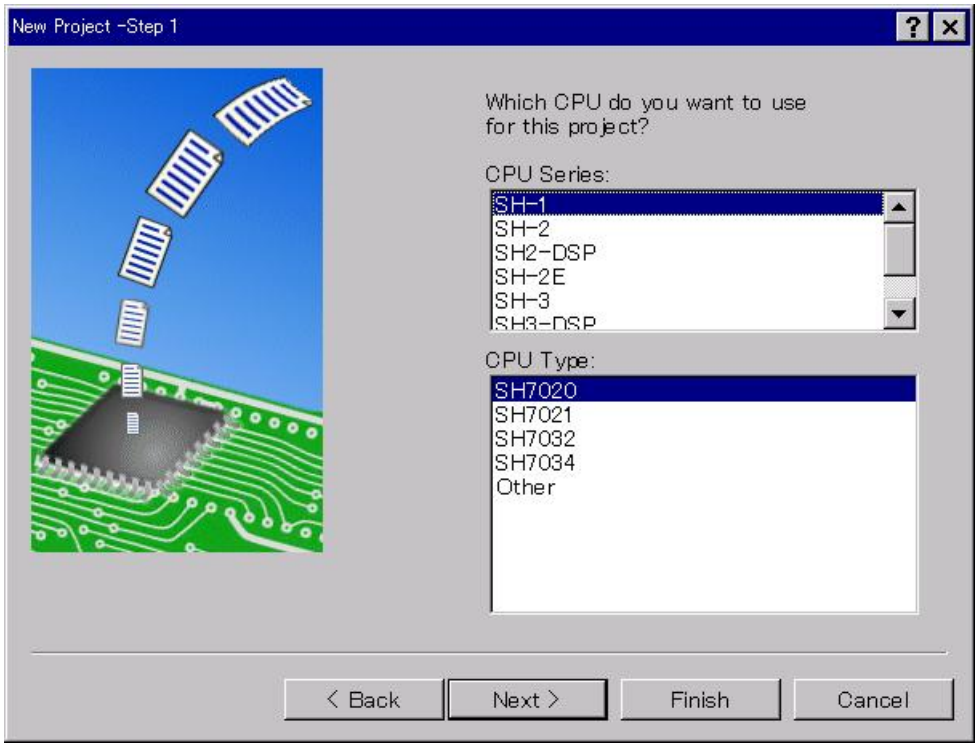


図 2.2 CPU**選択画面（**Step1**）**

2.3　オプション設定

Step1画面で［Next >］をクリックすると、図 2.3に示す画面を表示します。この画面で、全プロジェクトファイルで共通のオプションを設定します。オプション設定項目は、Step1画面で選択したCPUシリーズに合わせて設定変更ができるようになっています。また、プロジェクト作成後にオプションを変更する場合は、HEWの[オプション -> Hitachi SuperH RISC engine Standard Toolchain...]のCPUページで変更できます。

ここでは、設定を変更しないで［Next >］をクリックします。クリックすると、Step3の画面を表示します。

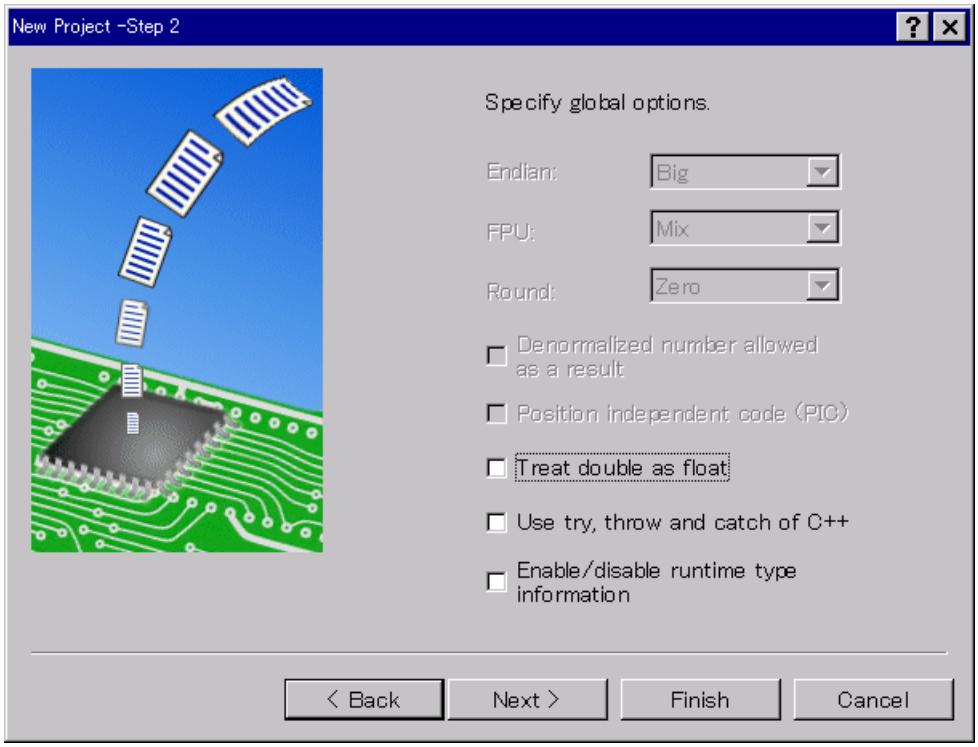


図 2.3 オプション設定画面（Step2）

2.4　生成ファイルの設定

Step2画面で［Next >］をクリックすると、図 2.4に示す画面を表示します。この画面で、生成するファイルを決定します。

［Use I/O Library］のチェックを付けると、標準入出力ライブラリを活用できます。

［Use Heap Memory］のチェックを付けると、ヒープ領域の管理用の関数sbrk()を活用できます。このとき、［Heap Size:］で、管理するヒープ領域のサイズを設定できます。

［Generate main() Function］では、main関数の雛型生成を選択できます。

［I/O Register Definition Files］のチェックを付けると、C言語で記述したI/Oレジスタの定義ファイルを生成します。このとき、［Generate Hardware Setup Function］でI/Oレジスタ初期設定プログラムの雛型生成を選択できます。

ここでは、［Use I/O Library］のチェックを追加して［Next >］をクリックします。

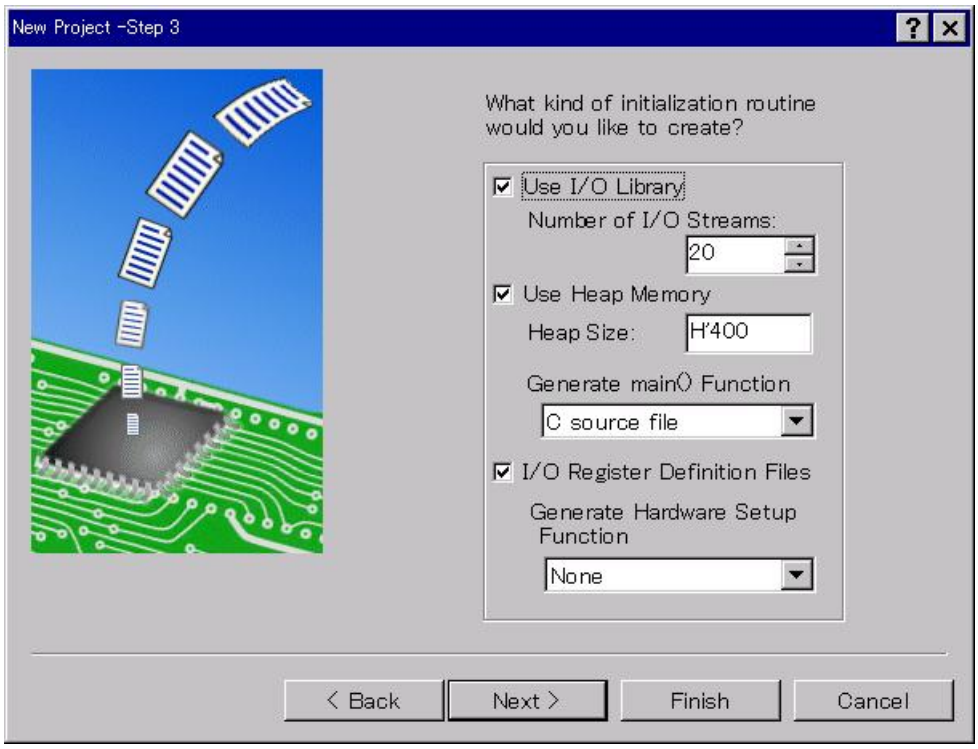


図 2.4 生成ファイル設定画面（Step3）

注意

既に作成したmain関数を組み込む場合は、［Generate main() Function］で"None"を選択し、プロジェクトを作成後に該当する（main関数を含んだ）ファイルをプロジェクトに追加してください。なお、組み込む関数名が異なる場合、resetprg.cの関数呼び出し部分を修正する必要があります。

また、プロジェクトジェネレータが生成するベクタテーブル定義およびI/Oレジスタ定義等のサンプルファイルの内容は、使用するCPUのハードウェアマニュアルで確認してください。

2.5　標準ライブラリの設定

Step3画面で［Next >］をクリックすると、図 2.5に示す画面を表示します。この画面で、C/C++コンパイラで使う標準ライブラリの構成を決定します。また、プロジェクト作成後に標準ライブラリの構成を変更する場合は、HEWの[オプション -> Hitachi SuperH RISC engine Standard Toolchain...]のStandard Libraryページで変更できます。

ここでは、設定を変更しないで［Next >］をクリックします。

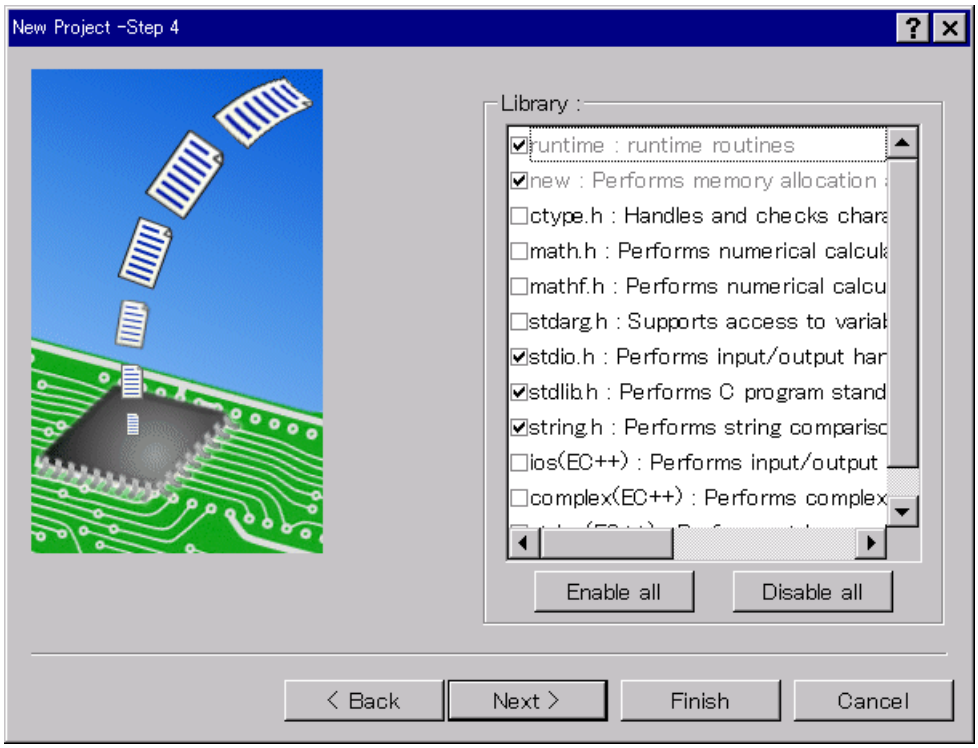


図 2.5 標準ライブラリ設定画面（Step4）

2.6　スタック領域の設定

Step4画面で［Next >］をクリックすると、図 2.6に示す画面を表示します。この画面で、スタック領域を設定します。スタック領域として設定する値の初期値は、Step1画面の［CPU Type:］により異なります。

また、プロジェクト作成後にスタックサイズを変更する場合は、HEWの[Project -> Edit Project Configuration]で変更できます。

この画面では、Step1画面の［CPU Type:］で“ SH7020 ”を選択したため、［Stack Pointer Address:］は“ H’7FFFFFF0 ”、［Stack Size:］は“ H’100 ”を設定しています。

ここでは、設定を変更しないで［Next >］をクリックします。

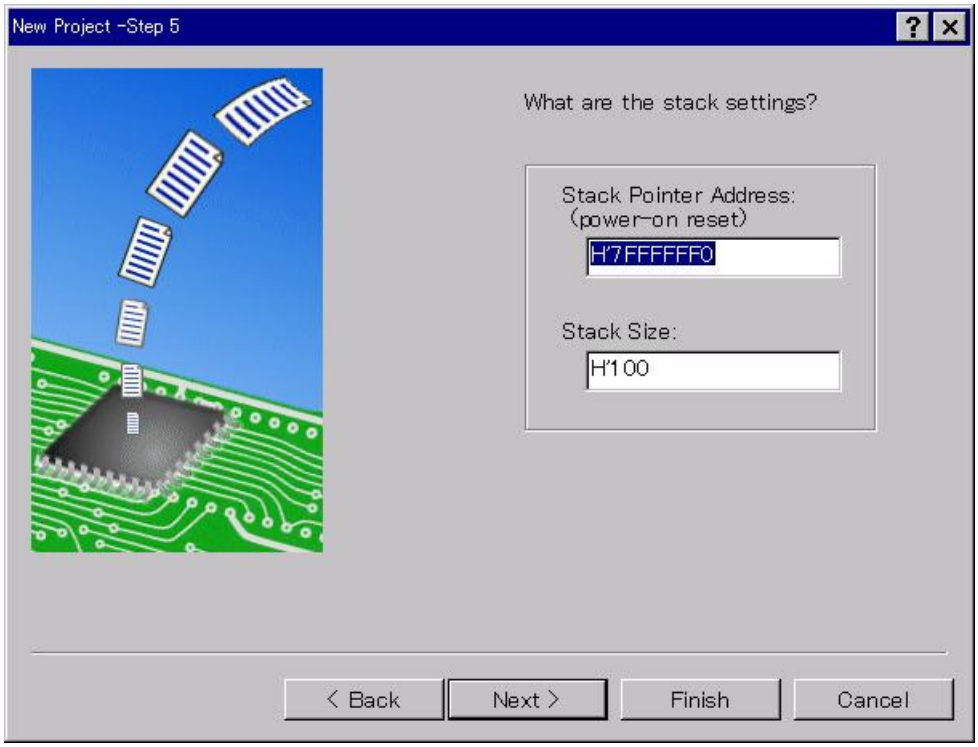


図 2.6 スタック領域設定画面（Step5）

注意

スタック領域は、HEWが生成する”stacksct.h”で定義しています。”stacksct.h”をエディタで編集した場合、HEWの[Project -> Edit Project Configuration]では変更できません。

2.7　ベクタの設定

Step5画面で［Next >］をクリックすると、図 2.7に示す画面を表示します。この画面で、ベクタを設定します。

［Vector Definition Files］のチェックを付けると、ベクタテーブル定義ファイルを生成します。

また、［Vector Handlers:］の［Handler］列はリセットベクタのハンドラプログラム名を、［Vector］列にはベクタの説明を表示しています。ハンドラプログラムを変更したい場合は、ハンドラプログラム名を選択してクリック後、入力してください。なお、ハンドラプログラムを変更すると、リセットプログラム（reserprg.c）は生成しません。

ここでは、設定を変更しないで［Next >］をクリックします。

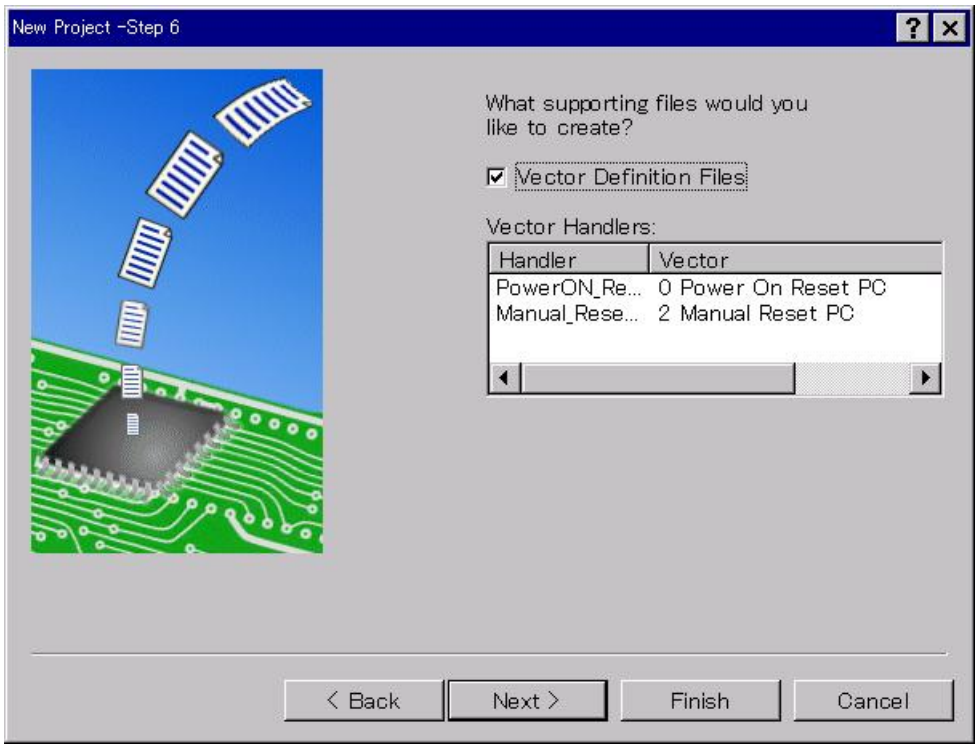


図 2.7 ベクタ設定画面（Step6）

2.8　デバッガターゲットの設定

Step6画面で［Next >］をクリックすると、図 2.8に示す画面を表示します。この画面で、デバッガターゲットを設定します。[Target:]から使用するデバッガターゲットを選択（チェック）してください。デバッガターゲットは、未選択でも複数選択してもかまいません。

ここでは、”SH-1 Simulator”を選択し、［Next >］をクリックします。

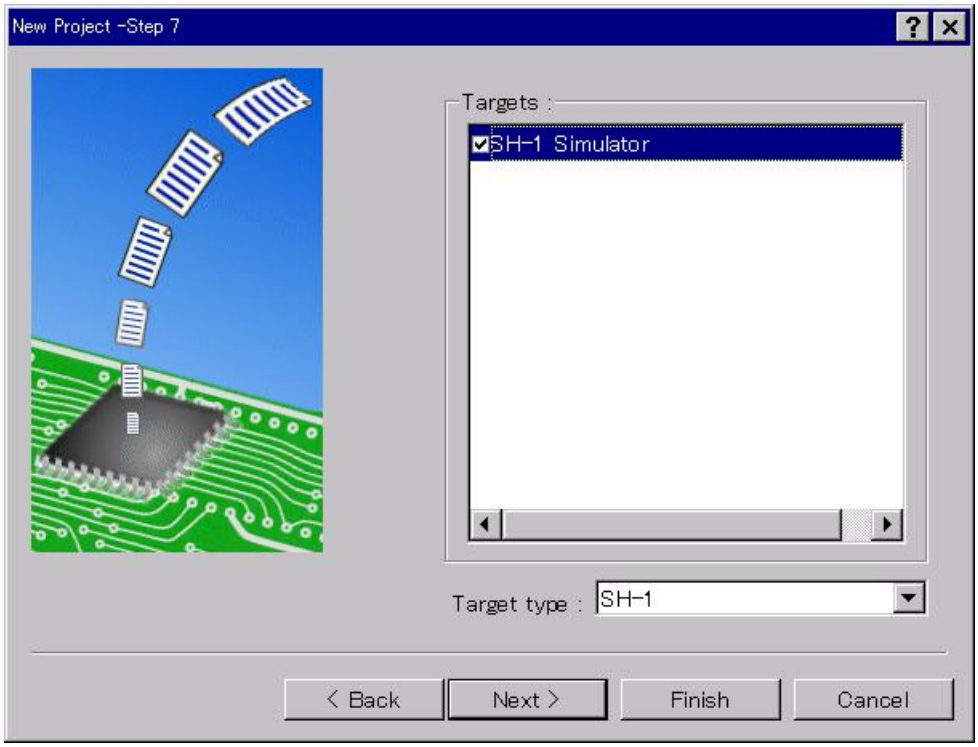


図 2.8 デバッガターゲット設定画面（Step7）

なお、[Target type:]を変更することにより他のデバッガターゲットを選択できるようになります。

2.9　デバッガオプションの設定

Step7画面で［Next >］をクリックすると、図 2.9に示す画面を表示します。この画面で、選択したデバッガターゲットのオプションを設定します。

HEWはデフォルトで、”Release”と”Debug”の二つのコンフィグレーションを作成しますが、デバッガターゲットを選択すると、選択したターゲット用のコンフィグレーションも作成します（ターゲット名を含んだ略称となります）。このコンフィグレーション名は、[Configuration name:]で変更できます。また、デバッガターゲットのオプションを、[Detail options:]で表示します。変更する場合は、[Item]を選択して[Modify]をクリックしてください。なお、変更できない項目の場合、[Item]を選択しても[Modify]はグレーのままです。

ここでは、設定を変更しないで［Next >］をクリックします。

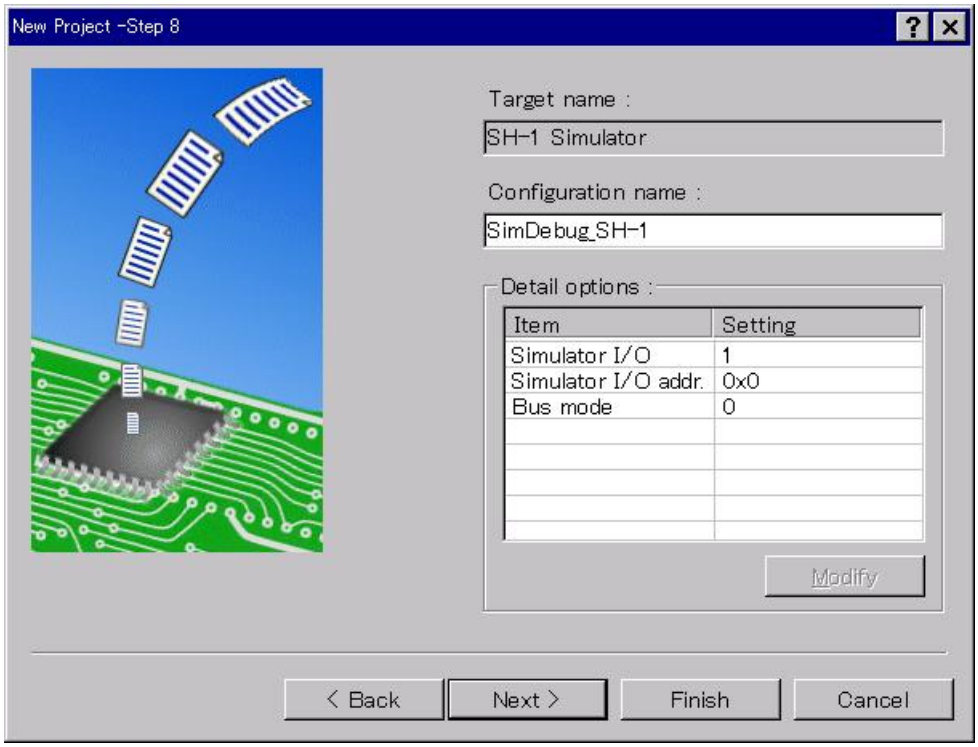


図 2.9 デバッガオプション設定画面（Step8）

2.10　生成ファイル名の変更

Step8画面で［Next >］をクリックすると、図 2.10に示す画面を表示します。この画面で、これまでの設定によりHEWが生成するファイルをリストに表示します。リストの［File Name］列はファイル名を、［Extension］列は拡張子を、［Description］列はファイルの説明を表示しています。ファイル名は、変更することができます。変更したい場合は、ファイル名を選択してクリック後、入力してください。

ここでは、設定を変更しないで［Finish］をクリックします。クリックすると、Summaryダイアログボックスを表示します。

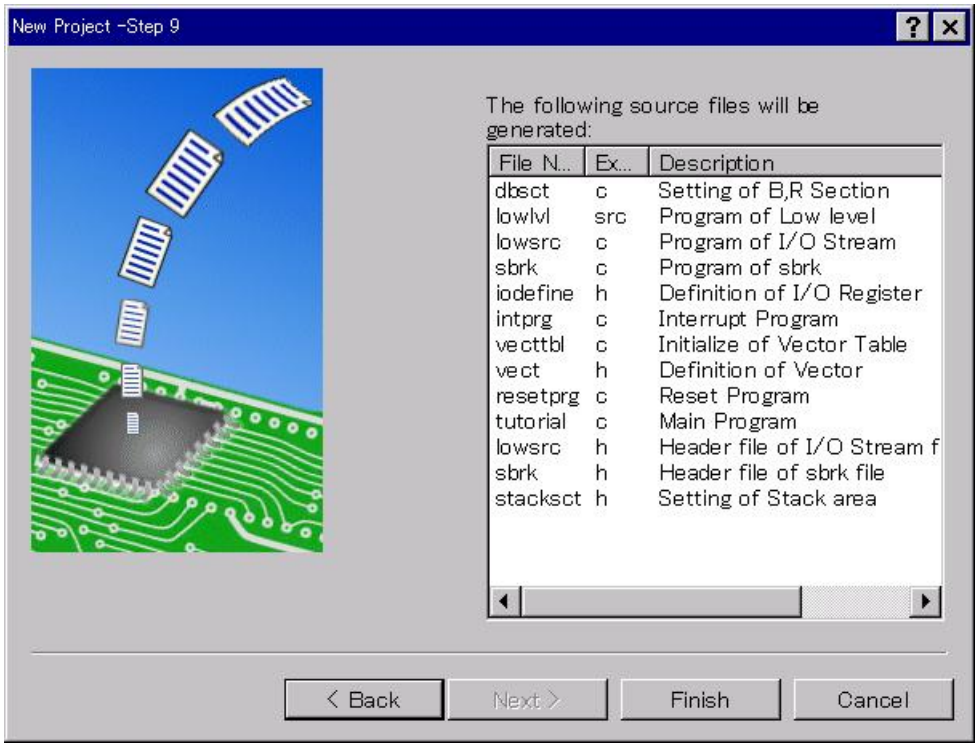


図 2.10 生成ファイル名変更画面（Step9）

注意

［Extension］が“ h ”または“ inc ”のファイルは、インクルードファイルです。これらのファイル名を変更した場合、インクルードしているファイルのinclude文も変更する必要があります。

2.11　設定確認（Summaryダイアログボックス）

プロジェクトジェネレータは、生成するプロジェクトに関する情報をSummaryダイアログボックス（図2.11）で表示しますので、確認後、[OK]をクリックしてください。

なお、[Generate Readme.txt as a summary file in the project directory]をチェックすると、Summaryダイアログボックスで表示したプロジェクトの情報を、”Readme.txt”という名称のテキストファイルでプロジェクトディレクトリに保存します。

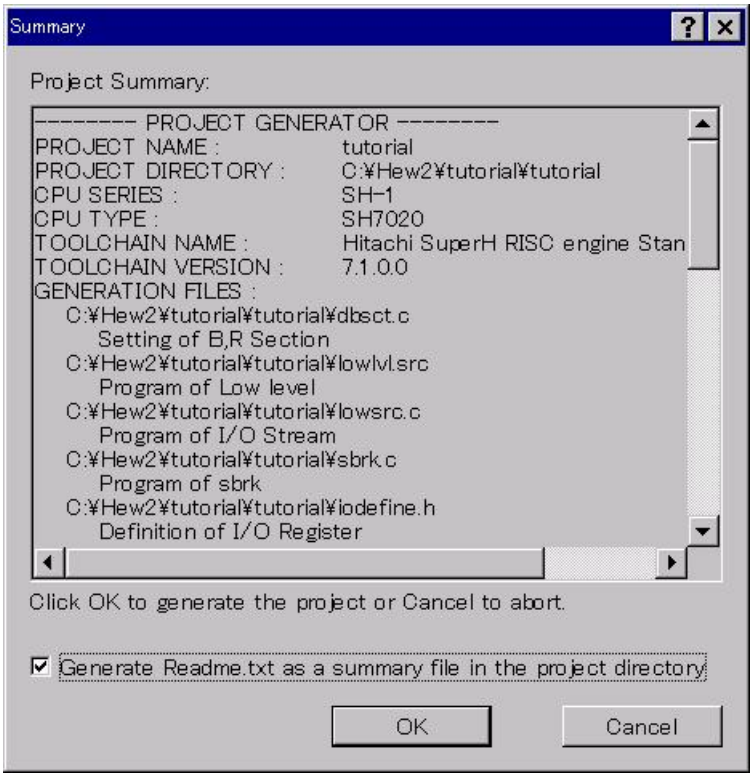


図 2.11 Summary**ダイアログボックス**

次に、HEWはプロジェクトジェネレータが生成したファイルを組み込んだプロジェクトを開きます。

HEWの初期ウィンドウ状態は、図 2.12に示すように、プロジェクトの内容を示すワークスペースウィンドウ（図中の左側）と、ビルドやファイル間の文字列検索などの結果を表示するアウトプットウィンドウ（図中の下側）と、テキストファイル編集用のエディタウィンドウ（図中の右側）に分割されています。HEWのウィンドウの状態を変更する場合や、エディタを含めた各種ウィンドウの機能については、「Hitachi Embedded Workshop 2 ユーザーズマニュアル」を参照してください。

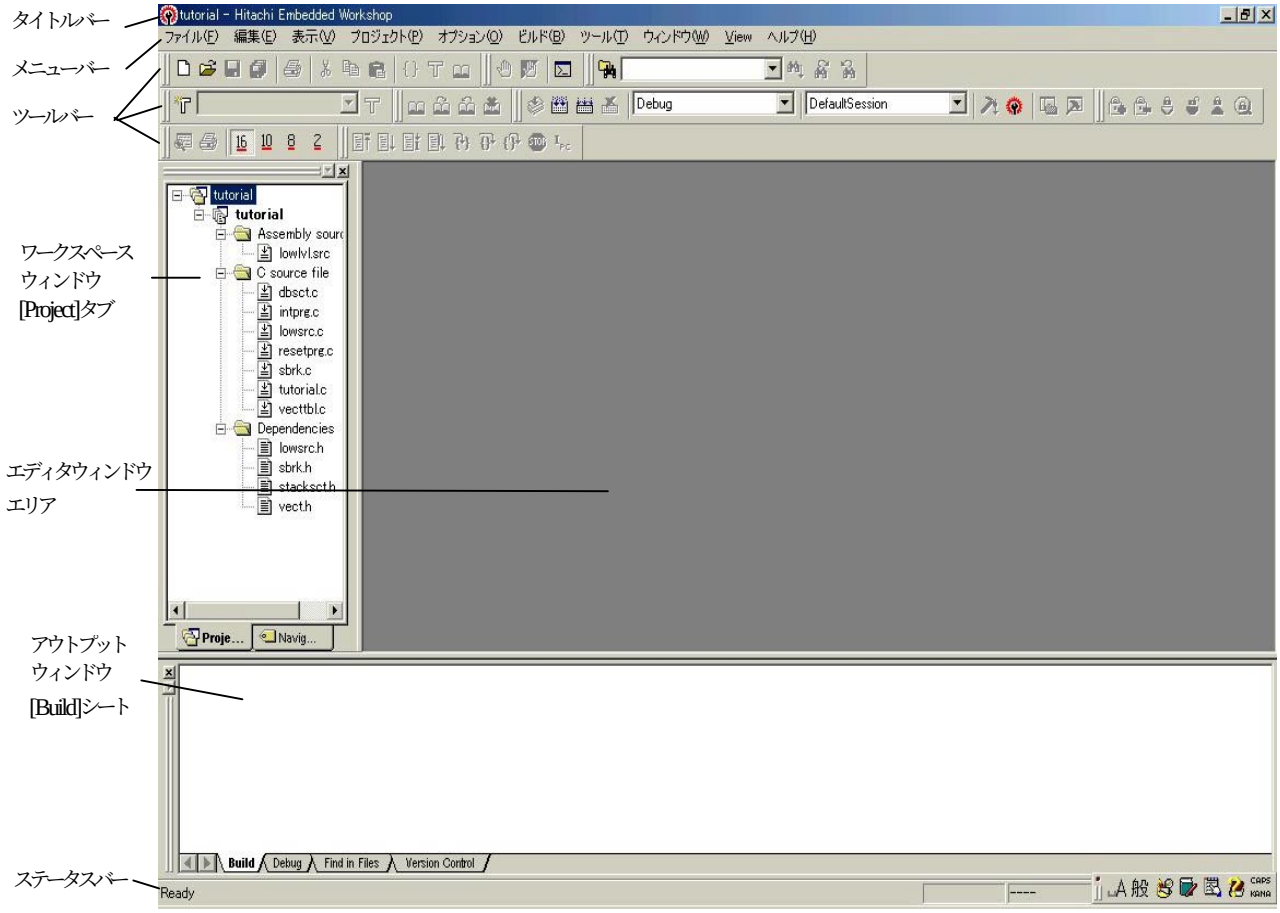


図 2.12 HEWのウィンドウ構成

また、プロジェクトジェネレータにより作成したプロジェクトでは、C/C++コンパイラ、アセンブラなどのビルドに使うツールの基本的なオプションは設定済みです。そのため、プロジェクト作成直後に[ビルド -> ビルド]によりビルドすることも可能です（図 2.13）。

なお、ビルドはツールバー上の

ボタンをクリックしても実行できます。

また、これらの基本的なオプションは、各ツールのデフォルト設定として定義していますので、以後プロジェクトにファイルを追加してもそのファイルに個別に設定するオプションを除いて、新たにオプションを設定する必要はありません。

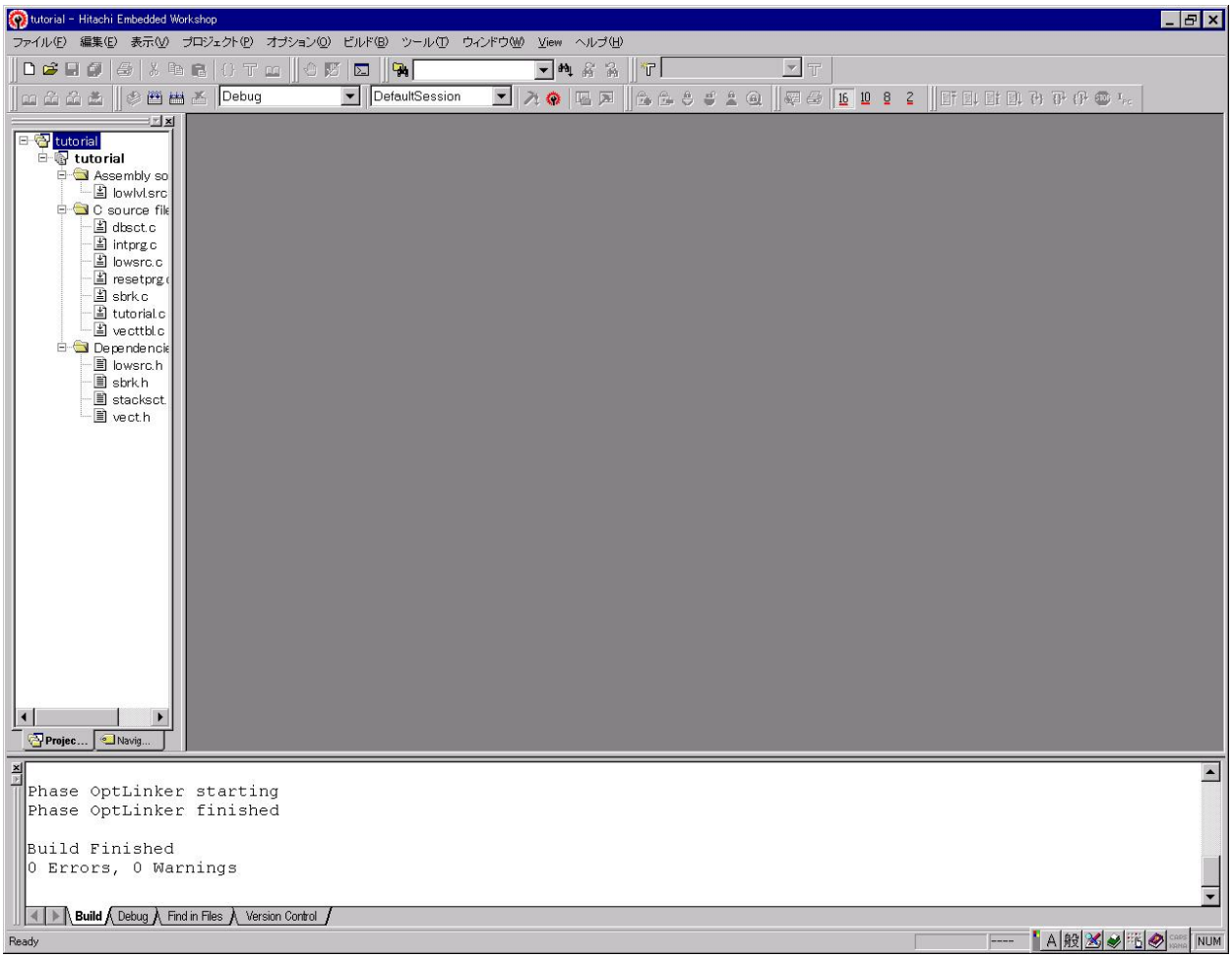


図 2.13 ビルド実行時のウィンドウ

3　プロジェクトの編集

プロジェクトジェネレータを使用してプロジェクトを作成すると、リセットルーチンやRAMの初期化ルーチンなどの基本的なプログラムを生成し、プロジェクトに組み込みます。ここでは、生成したファイルの修正や、新たなファイルのプロジェクトへの登録方法などについて説明します。

3.1　ソースファイルの編集と作成

プロジェクトに組み込まれたテキストファイルをワークスペースウィンドウの[Projects]シートで選択してダブルクリックすると、エディタウィンドウで編集することができます。ここでは、“ tutorial.c ”を開きます（図 3.1）。

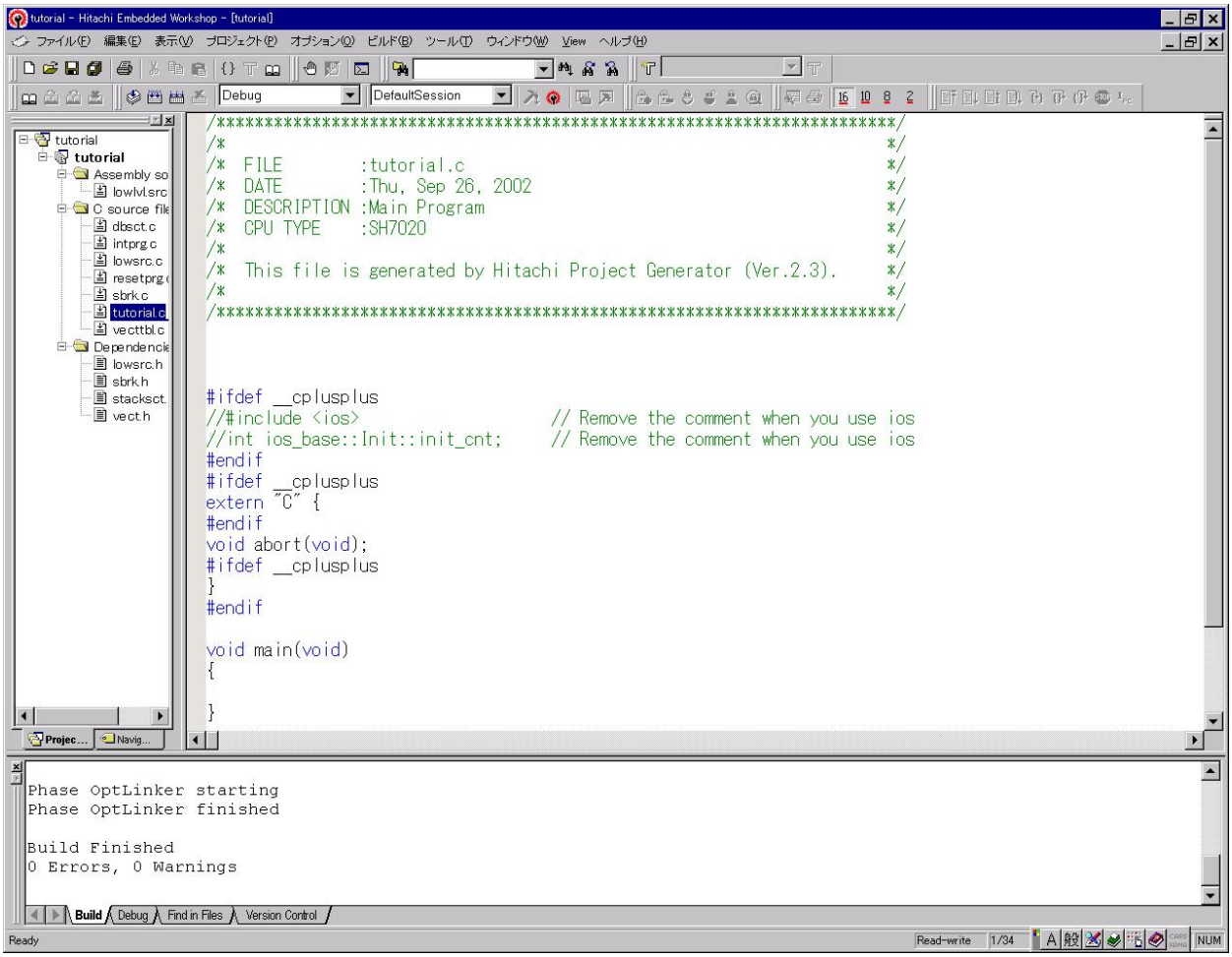


図 3.1 プロジェクトファイルのオープン

エディタウィンドウに開いたファイルは編集可能となり、編集するとウィンドウタイトルに表示するファイル名の右にアスタリスク（‘＊’）が付きます(図 3.2)。編集を終了し、ファイルを保存するとアスタリスクが消えます。

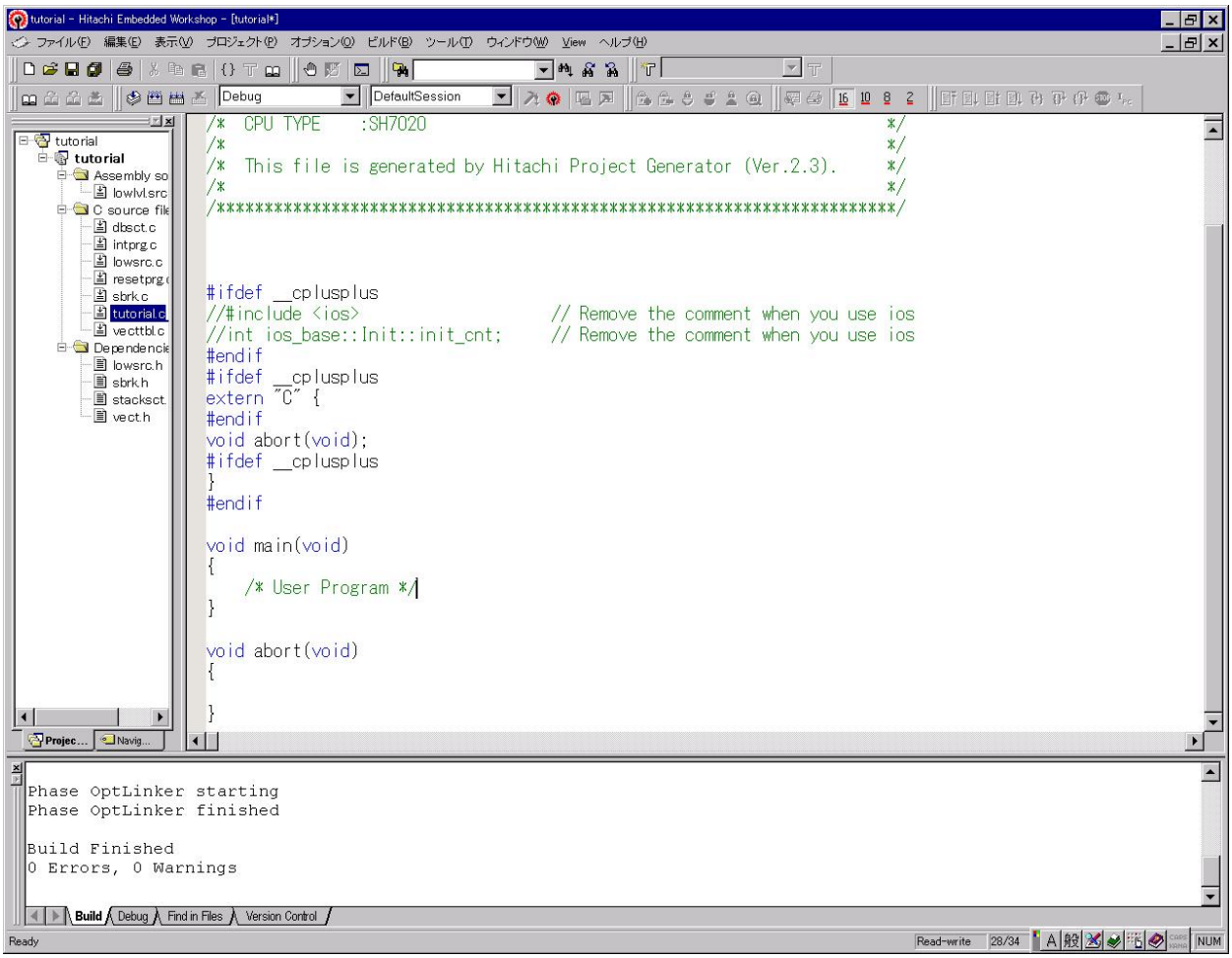


図 3.2 プロジェクトファイルの編集

また、既存のファイルを開く場合は、[ファイル -> 開く]で、新規にファイルを作成する場合は、[ファイル -> 新規作成]でエディタウィンドウにテキストファイルを開くことができます。

ここでは、[ファイル -> 新規作成]で新規ファイルを開き、次の３行を入力し、[ファイル -> 名前を付けて保存...]により“ newprog.c ”の名称でファイルを保存します。

```
void NewProgram(void)
{
}
```

3.2　プロジェクトへのファイル追加と削除

次に既存のファイルをプロジェクトに追加する方法について説明します。ここでは、例として前節で新しく作成した“ newprog.c ”をプロジェクトに追加します。

[プロジェクト -> ファイルの追加...]により、[ファイルの追加]ダイアログボックス（図 3.3）を表示します。追加するファイル（ここでは、“ newprog.c ”）を選択し、[Add]をクリックするとプロジェクトに“ newprog.c ”が追加されます。

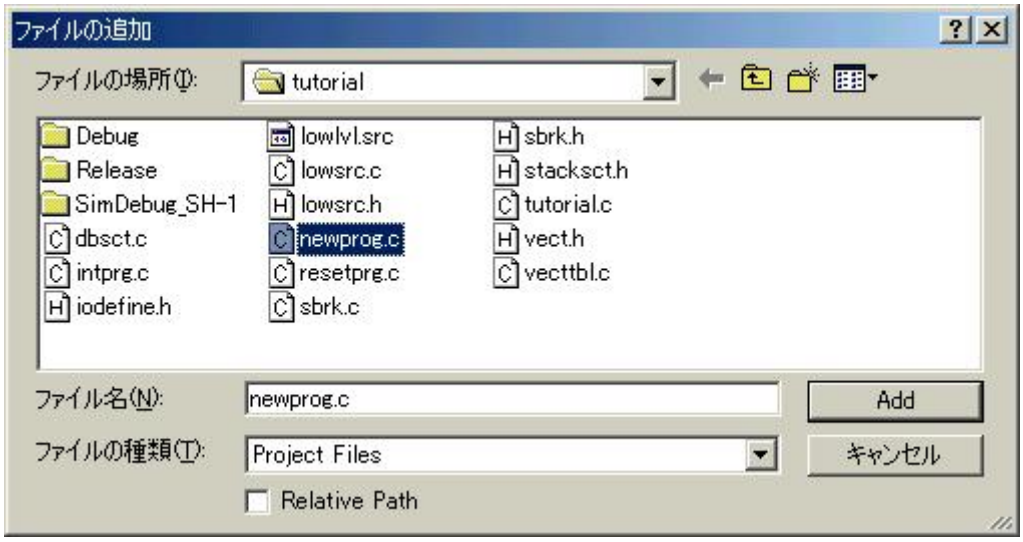


図 3.3 プロジェクトへのファイルの追加

プロジェクトに追加したファイルは、図 3.4に示すようにワークスペースウィンドウの[Projects]シートに表示されます。

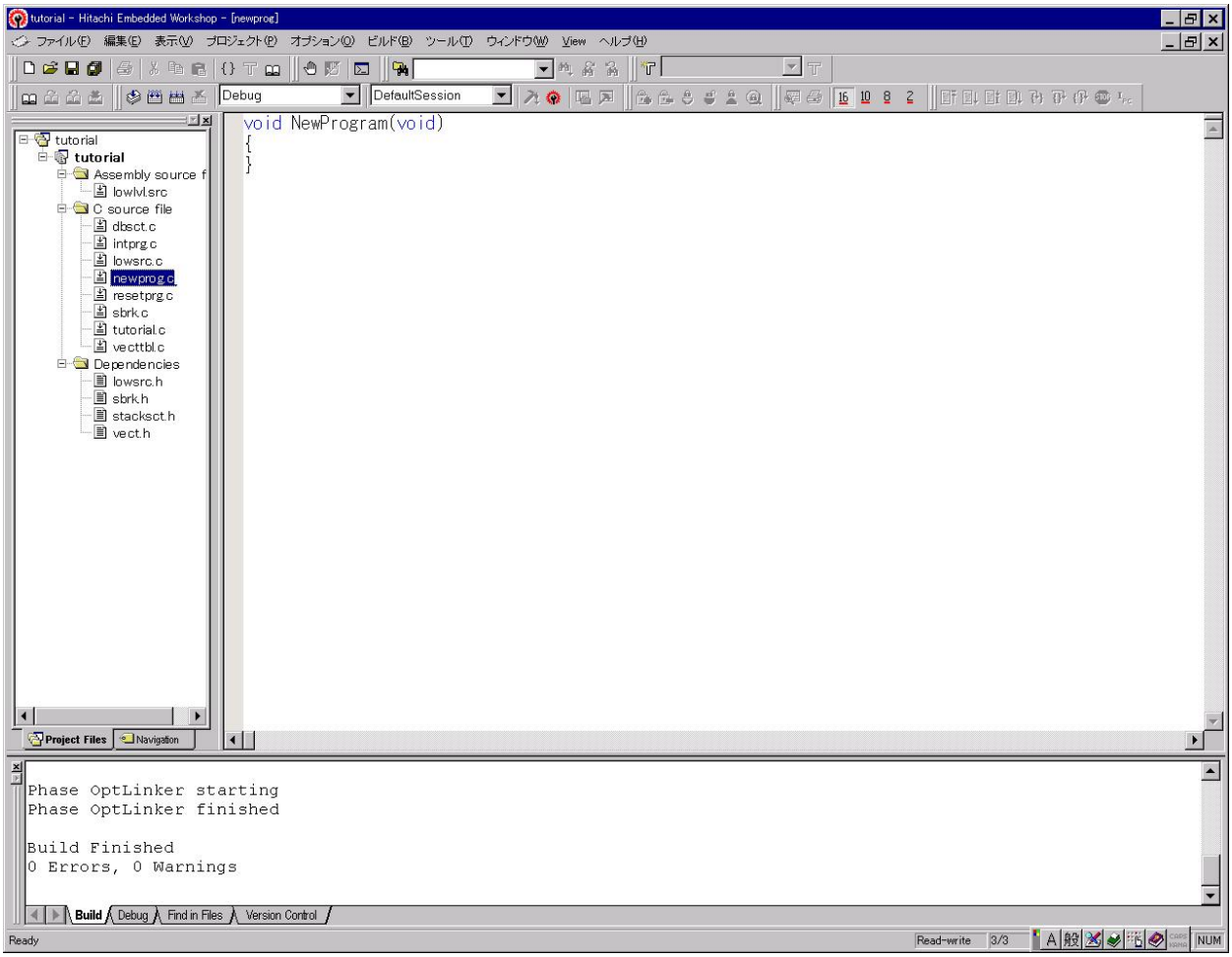


図 3.4 ファイルを追加したプロジェクト

次に、プロジェクトで不要となったファイルの削除方法を説明します。[プロジェクト -> ファイルの削除...]により、[プロジェクトの削除]ダイアログボックス（図 3.5）を表示します。ここで、削除したいファイルを選択（複数も可能）して[削除]をクリックすると[プロジェクトファイル]のリストからファイルを削除します。[OK]をクリックすると実際にプロジェクトからファイルを削除します。[キャンセル]をクリックすると削除は無効になります。

また、ワークスペースウィンドウの[Projects]シートでファイルを選択して[Delete]キーを押下してもプロジェクトからファイルを削除できます。

なお、ファイルをプロジェクトから削除するのではなく、一時的にビルドから外したい場合は、「4.4 プロジェクトファイルの除外」を参照してください。

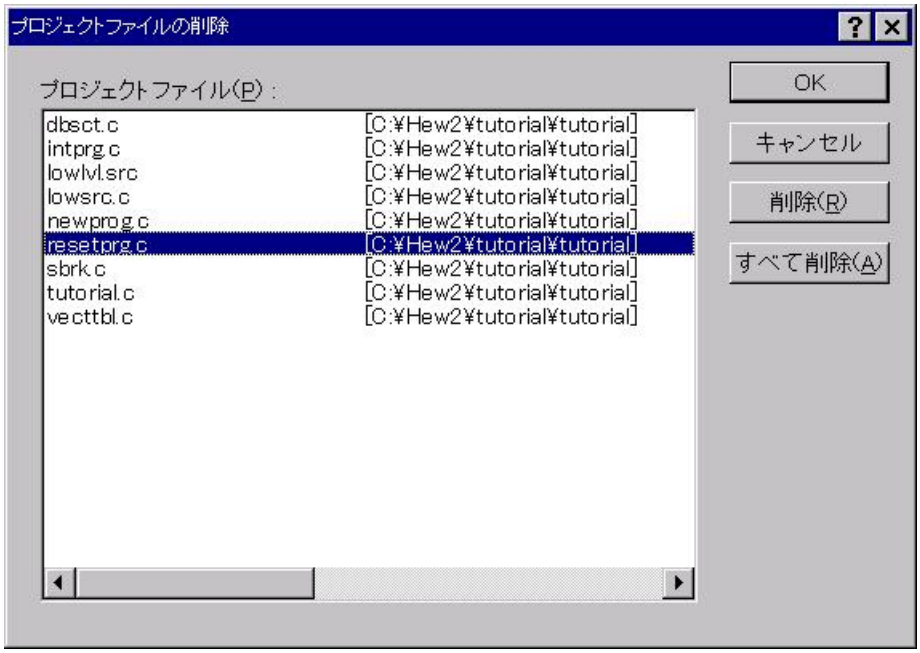


図 3.5 プロジェクトの削除ダイアログボックス

3.3　プロジェクトで使えるファイルの種類

プロジェクトジェネレータによりＣ言語とアセンブリ言語で記述したファイルをプロジェクトに登録しました。例えば、“ *.c ”はＣ言語ソースファイルに属し、“ *.asm ”はアセンブリソースファイルに属します。

次に、プロジェクトに追加できるファイルの種類について説明します。

HEWは、ファイルの種類をその内容ではなく拡張子で識別します。[プロジェクト -> ファイルの拡張子...]により、[ファイル拡張子]ダイアログボックス（図 3.6）を表示します。ここで、プロジェクトで使用できるファイルの拡張子を確認できます。すべての拡張子はファイルグループにより分類され、このファイルグループ単位でビルド時に使うツールを決めることができます。

ユーザが独自に使用している拡張子をHEWで使用したい場合は、[追加...]をクリックして新たな拡張子を定義する必要があります。

また、[Extension]で表示しているアイコンはファイルを開くためのツールを示しています。

ここで、アイコンが

で表示している拡張子のファイルはHEWのエディタで開けるテキストファイルを表わします。

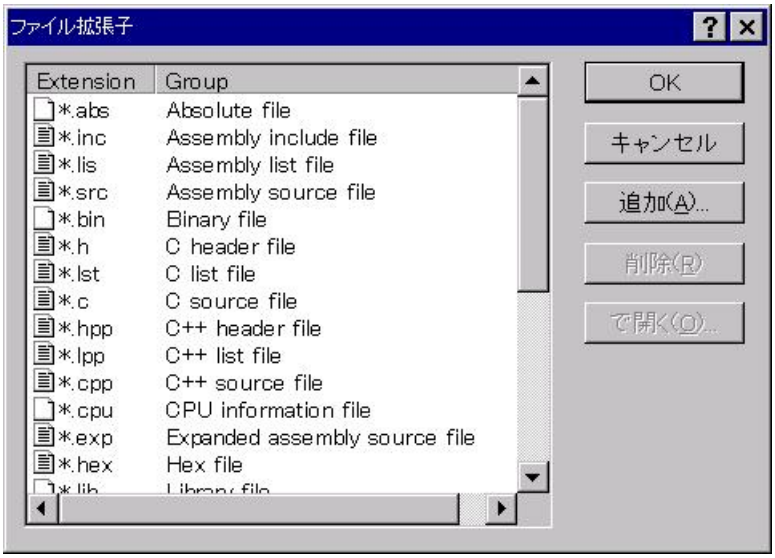


図 3.6 ファイル拡張子ダイアログボックス（初期画面）

[ファイル拡張子]ダイアログボックスで[追加...]をクリックすると、[ファイル拡張子の追加]ダイアログボックス（図 3.7）を表示します。このダイアログボックスの[ファイル拡張子]に新たに定義する拡張子名を入力し、[ファイルグループ]を設定します。既存のグループへの追加であれば[既存の拡張子グループ]を選択し、下のドロップダウンリストボックスから該当するグループを選択します。

また、新しいグループを作成する場合は、[新規拡張子グループ]を選択し、下の入力エリアにグループ名を入力します。

ここでは、拡張子“ asm ”を[ファイル拡張子]に指定し、ドロップダウンリストから“ Assembly source file ”を選択します。[OK]をクリックすると、“ *.asm ”ファイルも“ Assembly source file ”グループとしてプロジェクトに追加できるようになります（図 3.8）。

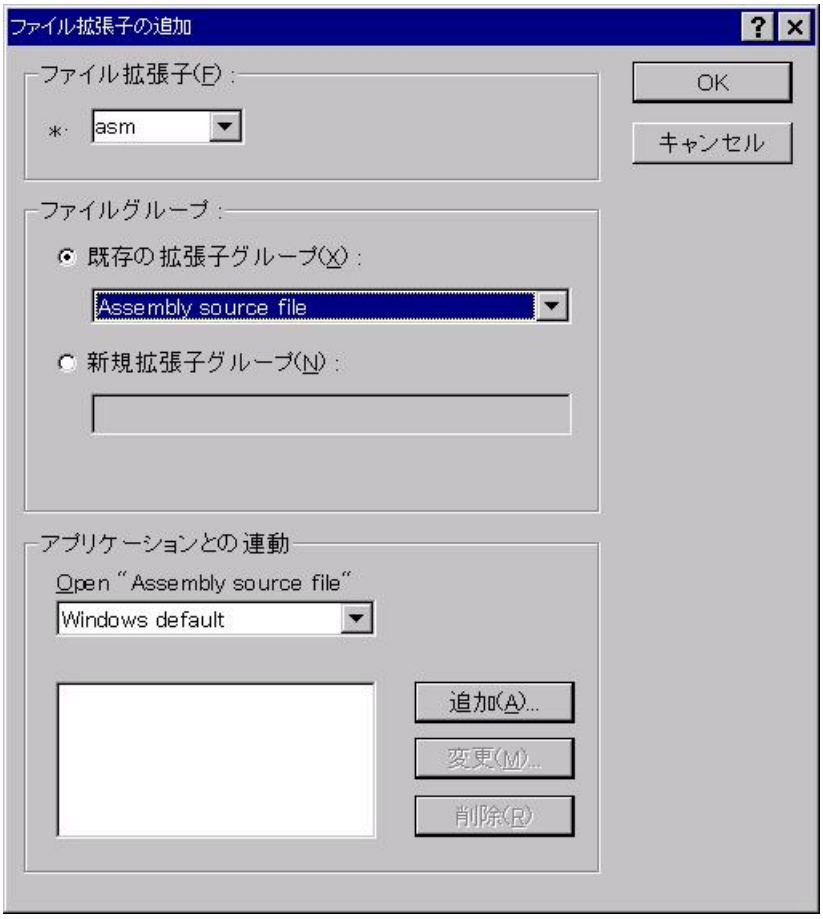


図 3.7 ファイル拡張子の追加ダイアログボックス

図 3.8 ファイル拡張子ダイアログボックス（追加後）

3.4　ワークスペースウィンドウのカスタマイズ

ワークスペースウィンドウの[Projects]シート上でマウスの右ボタンをクリックし、ポップアップメニューで[表示の構成...]を選択すると、[表示の構成]ダイアログボックス（図 3.9）を表示します。ここでの設定で、ファイルごとのインクルード関係やファイルのフルパスでの表示（図 3.10）などが可能になります。

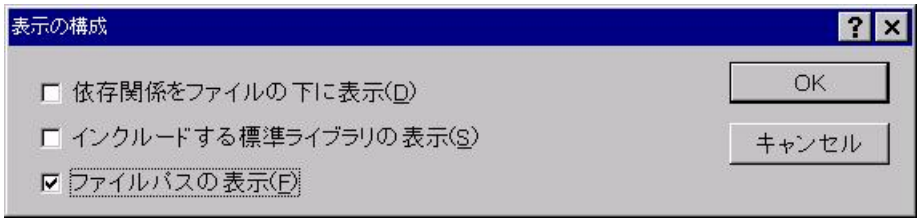


図 3.9 表示の構成ダイアログボックス

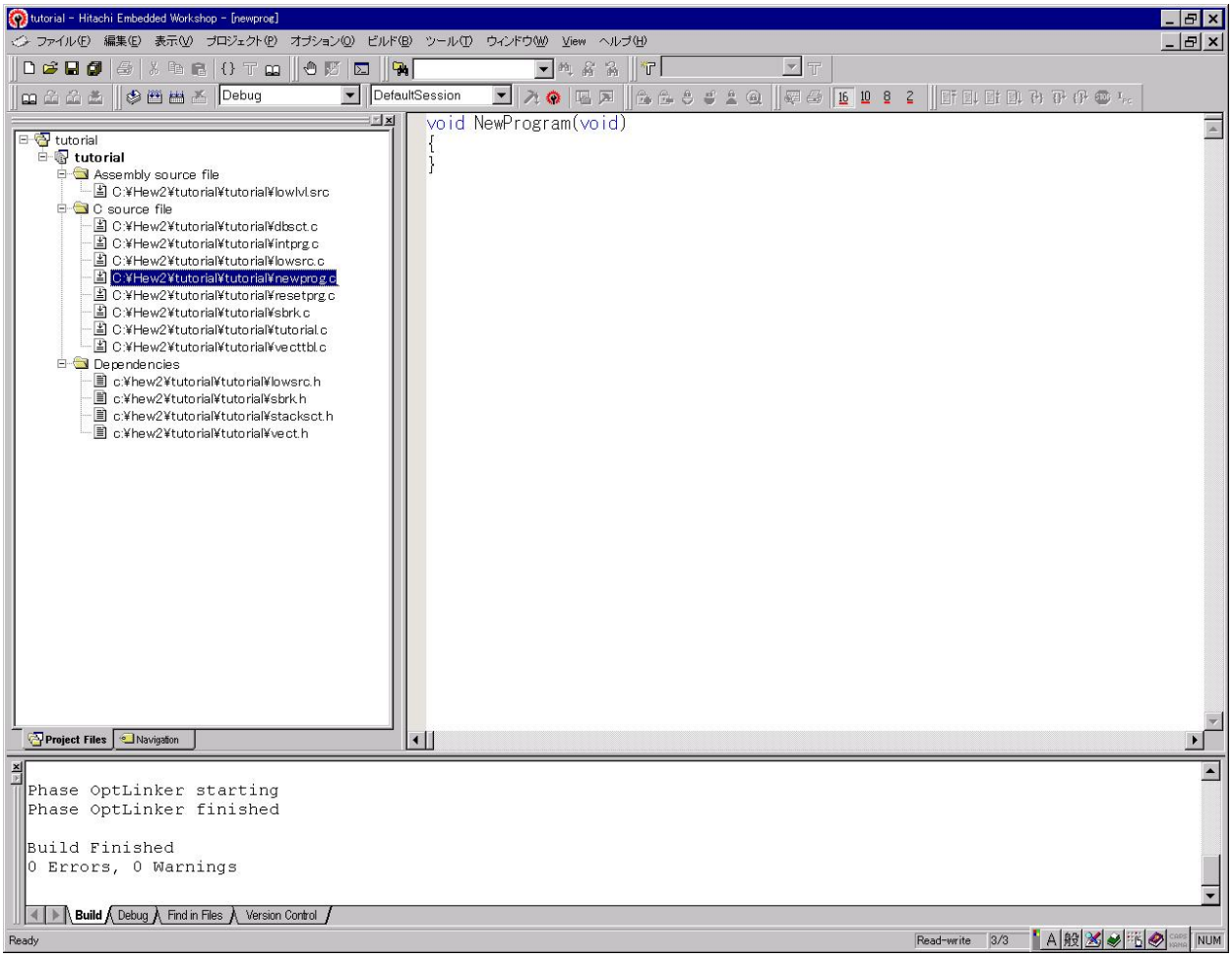


図 3.10 ワークスペースウィンドウの表示形式

4　オブジェクトの作成

4.1　ビルドの機能

オブジェクトの作成には、以下の２種類の方法があります。

１）全ビルド

コンパイル、アセンブル、リンクなどの一連のオブジェクト作成処理を、すべてのソースファイルに対して実施します。[ビルド -> すべてをビルド]で実行を開始します。

２）ビルド

前回のオブジェクト作成以降に修正したソースファイル（または、インクルードファイルを修正したソースファイル）だけに対して実施します。[ビルド -> ビルド]で実行を開始します。

ビルドは、ソースファイルなどの修正だけでなく、オプションの設定変更やプロジェクトファイルの構成を変更することにより影響を受けるファイルも対象になります。

なお、単一ファイルをコンパイル、または、アセンブルする場合は、ワークスペースウィンドウでファイルを選択し、[ビルド -> コンパイル]を選択します。

また、複数のプロジェクトやコンフィグレーションに対して、ビルドや全ビルド処理を実行する場合は、[ビルド -> 複数ビルド...]でビルド対象を指定することができます。

注意

- **ビルド実行開始時、エディタウィンドウの環境設定により、編集中のファイルをすべて保存することがあります。編集したファイルで保存しないものについては、あらかじめファイルを閉じておいてください。**
  エディタ機能およびエディタウィンドウのカスタマイズについては、「Hitachi Embedded Workshop 2ユーザーズマニュアル」を参照してください。
- **ビルド中にプロジェクトに関連するファイルはビルド処理と競合する可能性がありますので、保存しないでください。**

4.2　オプションの変更方法

プロジェクトジェネレータでプロジェクトを作成することにより、ビルドで必要となる基本的なオプションは設定済みです。新たにオプションを変更する場合は、[オプション -> Hitachi SuperH RISC engine Standard Toolchain]により表示するオプション設定ダイアログボックスで設定します。

図 4.1にオプション設定ダイアログボックスを示します。ダイアログボックスは、左側のプロジェクトファイルリストと右側のツールごとのオプション設定部分で構成しています。左側のファイルリストでファイルを選択すればそのファイル固有のオプションが設定でき、ファイルグループのフォルダを選択すればファイルグループに登録しているプロジェクトファイルで共通のオプションを設定できます。複数ファイルのオプションを同時に設定する場合は、[Ctrl]キーまたは[Shift]キーを押しながらマウスで該当するファイルを選択してください。

また、“ Default Options ”を選択すれば、ファイルグループのオプションの初期設定が変更できます。“ Default Options ”はファイルグループごとに設定できます。“ Default Options ”に設定したオプションは、そのファイルグループの拡張子を持つファイルをプロジェクトに追加した時に自動的に設定されます。

例えば、拡張子‘ C ’を持つファイル (“ *.C ”) をプロジェクトに追加すると、そのファイルには“ C source file ”の“ Default Options ”を設定します。

なお、各オプション内容の詳細については、「SuperH RISC engine C/C++コンパイラ、アセンブラ、最適化リンケージエディタ ユーザーズマニュアル」を参照してください。

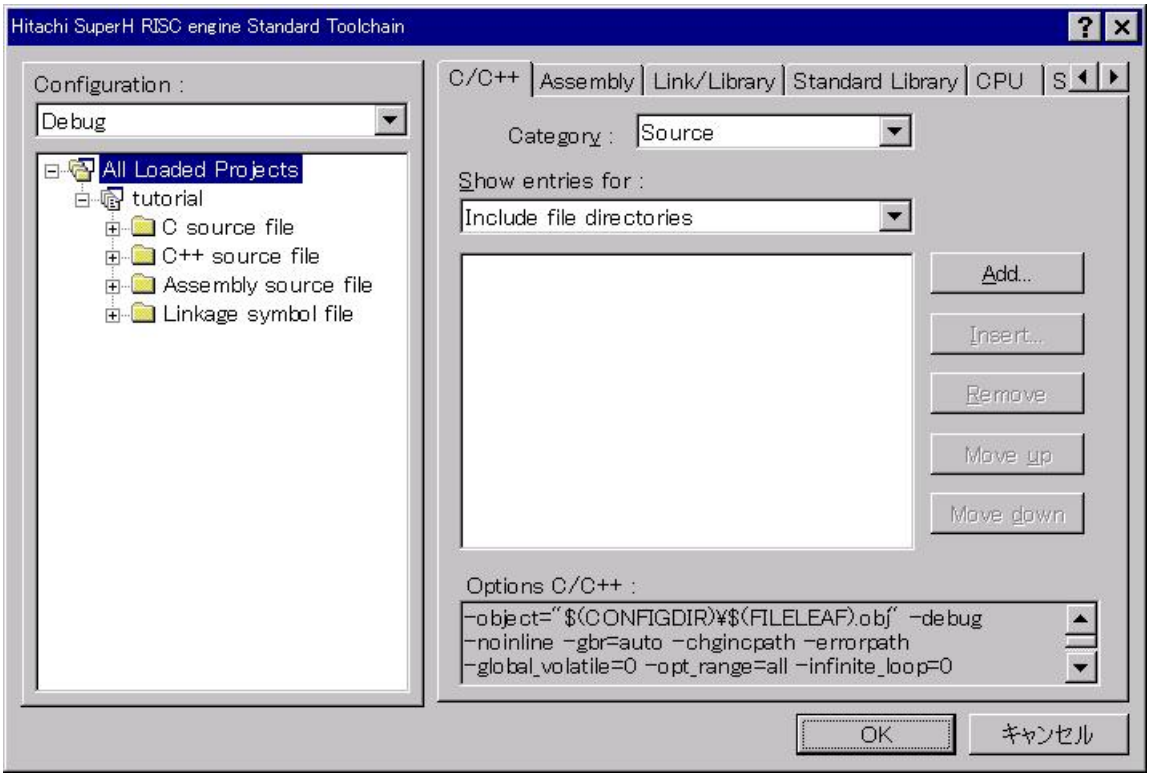


図 4.1 C/C++**コンパイラのオプション設定ダイアログボックス**

4.3　コンフィグレーションのカスタマイズ

前節で、プロジェクトファイルのオプション設定について説明しましたが、HEWでは、同じプロジェクトファイルに対して複数の組み合わせでオプションを設定することが可能です。

プロジェクトジェネレータは、[Debug]と[Release]の２種類のコンフィグレーション（デバッグオプションの有無が異なる）を作成します。また、デバッガターゲットを選択すると、選択したターゲットごとのコンフィグレーションも作成します。コンフィグレーションを変更する場合は、[オプション -> ビルドの構成...]により[ビルドコンフィギュレーション]ダイアログボックス（図 4.2）を表示し、[現在のコンフィギュレーション]ドロップダウンリストから使用するコンフィグレーションを選択し[OK]をクリックします。HEWのツールバーにあるドロップダウンリストからもコンフィグレーションの選択は可能です。

また、[ビルドコンフィギュレーション]ダイアログボックスでは、コンフィグレーションの新規作成や削除もできます。詳しくは、「Hitachi Embedded Workshop 2 HEW ユーザーズマニュアル」の「２章 ビルドの基本」を参照してください。

図 4.2 **ビルドコンフィギュレーションダイアログボックス**

4.4　プロジェクトファイルの除外

次にプロジェクトからファイルを除外する方法を説明します。これはプロジェクトからファイルを削除するのではなく、コンフィグレーション単位でファイルをビルドから外す場合に使います。

ワークスペースウィンドウ上で除外したいファイルを選択後、マウスの右ボタンをクリックし、ポップアップメニューから[Exclude Build <ファイル名>]を選択します。図 4.3に示すように、ファイルのアイコンに赤いバツ印を付け、ビルドから除外します。

なお、除外したファイルを再びビルドに入れる場合は、削除時と同様にポップアップメニューから[Include Build <ファイル名>]を選択します。

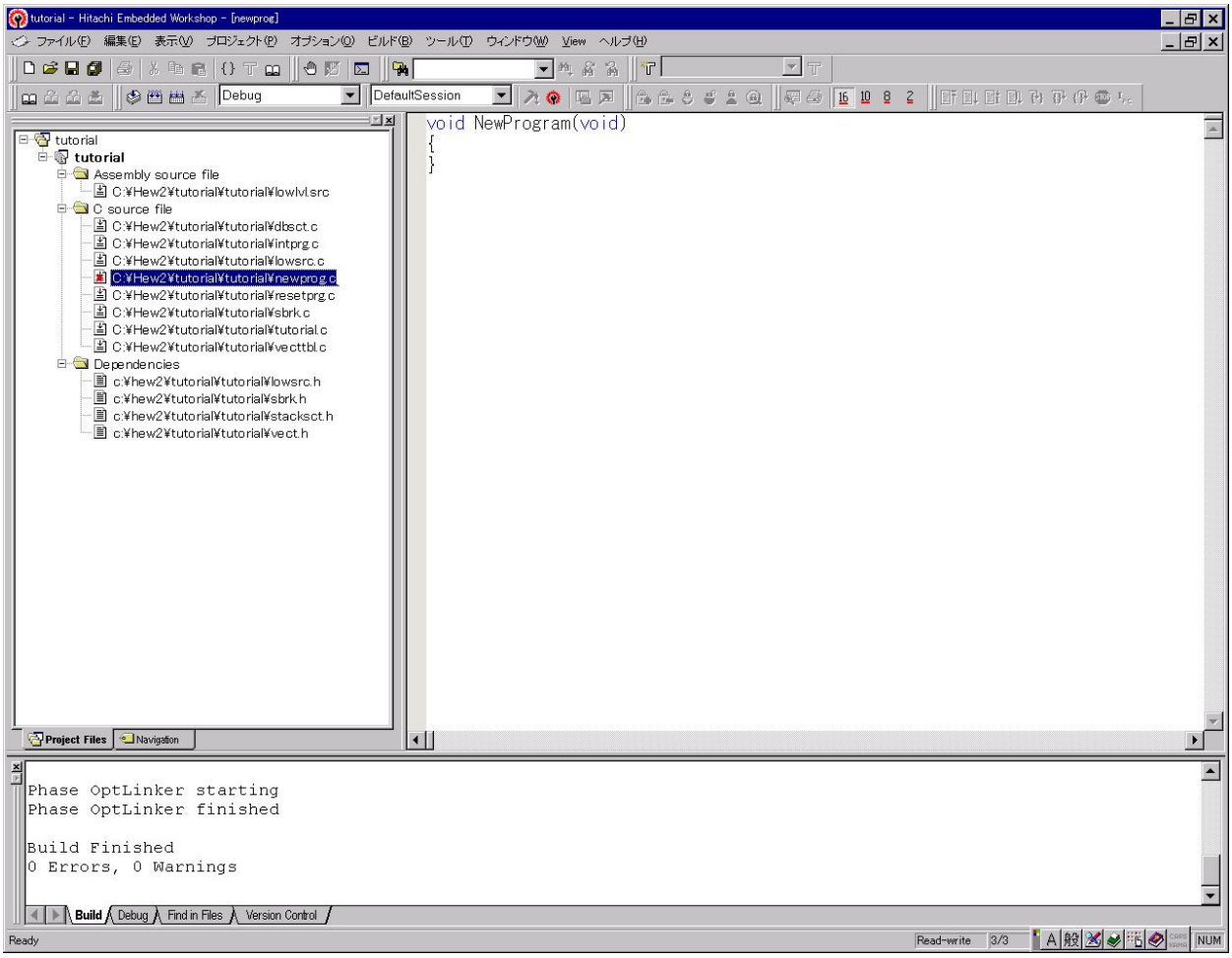


図 4.3 プロジェクトファイルの除外

4.5　ビルド時のエラー修正

次に、オブジェクトの作成処理中にコンパイラなどでエラーが発生した場合の対処について説明します。

図 4.4にエラー行を含むファイルをビルドした例を示します。ビルド実行により、アウトプットウィンドウにビルド結果（ここではコンパイル結果）が表示され、エラーが発生したことが分かります。ここで、アウトプットウィンドウの[Build]シート上のエラーメッセージ表示行をダブルクリックすると、エディタウィンドウ上にエラーが発生したファイルを開き、エラー行にカーソルが移動します（ただし、エラー原因の対策でファイルの行数が変わった場合、カーソルの移動先がずれることがあります）。

また、アウトプットウィンドウ上のエラー行にカーソルを置いた状態で、[F1]キーを押下すると、該当するエラーに関するヘルプを表示します。

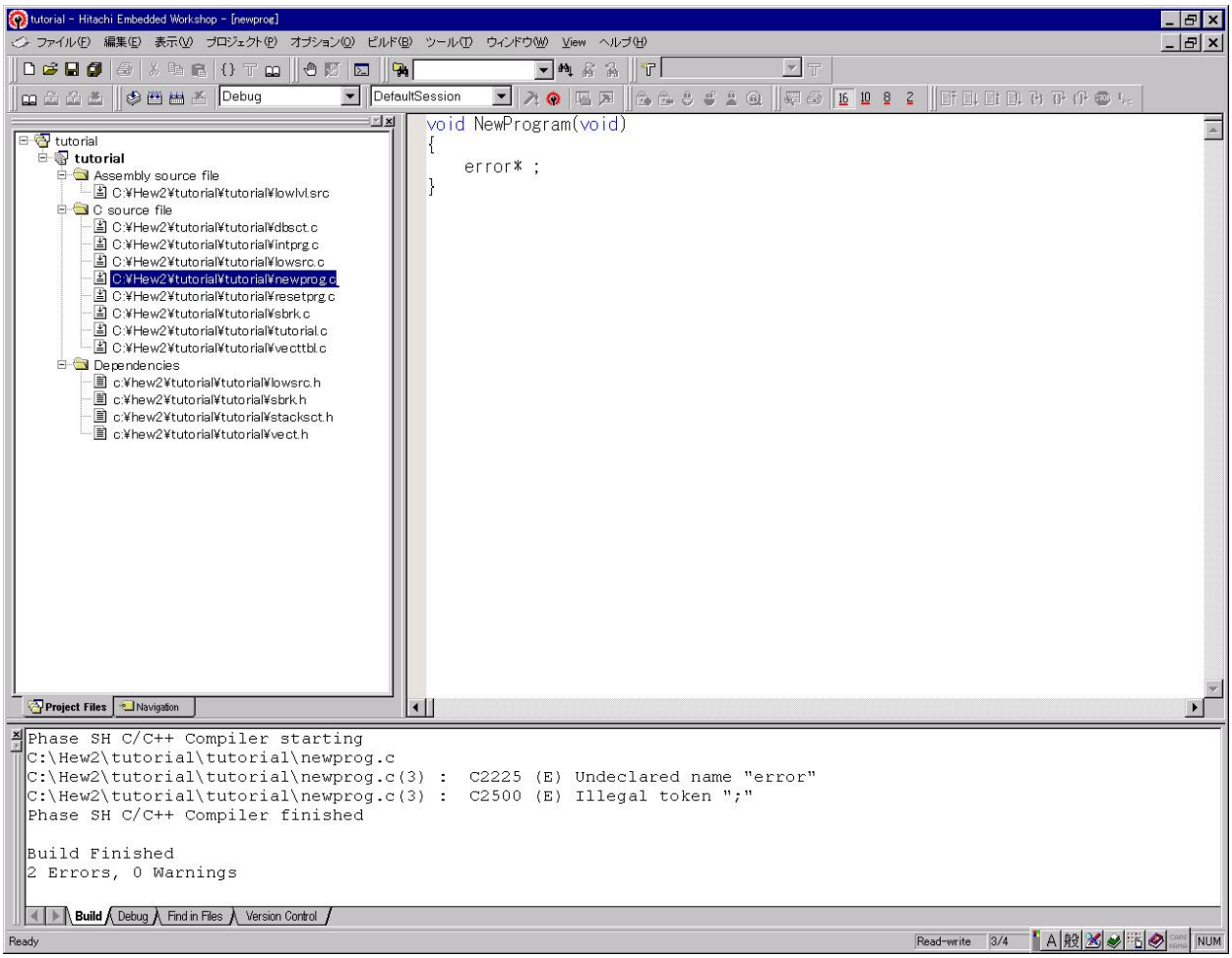


図 4.4 ビルド時のエラー修正

4.6 セッションのカスタマイズ

HEWでは、デバッガターゲットの設定をセッションに保存することが可能です。

プロジェクトジェネレータは、[DefaultSession]セッションを作成します。また、デバッガターゲットを選択すると、選択したターゲットごとのセッションも作成します。

デバッガターゲットを使用するためにセッションを変更します。 [オプション -> デバッグセッション...]により[デバッグセッション]ダイアログボックス（図 4.5）を表示し、[カレントセッション]ドロップダウンリストから［SimSessionSH-1］を選択し[OK]をクリックします。HEWのツールバーにあるドロップダウンリストからもセッションの選択は可能です。

また、[デバッグセッション]ダイアログボックスでは、セッションの新規作成や削除もできます。詳しくは、「Hitachi Embedded Workshop 2 HEW ユーザーズマニュアル」の「１．９ ビルドの基本」を参照してください。

図 4.5 デバッグセッションダイアログボックス

5　デバッグ

5.1　デバッグの準備

シミュレータ・デバッガの主な特徴をサンプルプログラムを用いて説明します。

注意

本章の使用例（図）の内容は、ご使用になるコンパイラのバージョンによって変わってくる場合がありますのでご了承ください。

5.1.1 サンプルプログラム

サンプルプログラムは、10個のランダムデータを昇順にソートした後、降順にソートするCプログラムに基づいています。

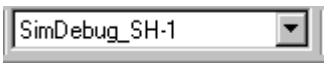
サンプルプログラムでは、以下の処理を行います。

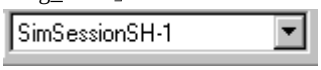
- main関数でソートするランダムデータを生成します。
- sort関数ではmain関数で生成したランダムデータを格納した配列を入力し、昇順にソートします。
- change関数では、sort関数で生成した配列を入力し、降順にソートします。
- printf関数を用いて、ランダムデータ、ソートしたデータを表示します。

サンプルプログラムは、HEWのデモンストレーションプログラムを用います。

5.1.2 サンプルプログラムの作成

１章～４章を参照し、下記に注意してHEWのデモンストレーションプログラムを作成してください。

- 「2.1　新規ワークスペースを作成する」では“Project Type”に Demonstration を指定してください。
- “CPU Series: ” は SH-1 を、”Target: “ は SH-1 Simulator を選択してください。
- コンフィギュレーションは、ツールバーより「SimDebug_SH-1」 

をビルド前に選択してください。
- セッションは、ツールバーより「SimSessionSH-1」 

を選択してください。

デバッグ機能の説明のため、Demonstrationは最適化なしの設定になっています。最適化の設定は変更しないでください。

5.2　デバッグのための設定

5.2.1 メモリリソースの確保

開発しているアプリケーションを動作させるためにメモリリソースの確保が必要です。デモンストレーションプロジェクトでは、自動的にメモリリソースを確保しますので、設定を確認してください。

- [オプション]メニューから [シミュレータ -> メモリリソース...]を選択し、現在のメモリリソースを表示してください。

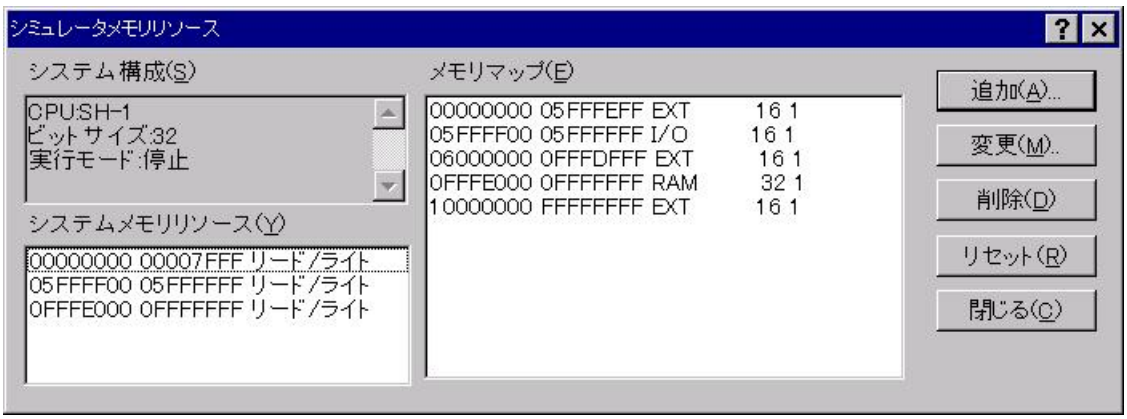


図 5.1　**シミュレータメモリリソースダイアログボックス**

プログラム領域としてH'00000000からH'00007FFFを、スタック領域としてH'0FFFE000からH'0FFFFFFFを読み出し/書き込み可能領域として確保しています。

- [閉じる]ボタンをクリックしてダイアログボックスを閉じてください。

メモリリソースは、Hitachi SuperH RISC engine Standard Toolchainダイアログボックスの[Simulator]ページでも参照／変更ができます。おたがいの変更は反映されます。

5.2.2 サンプルプログラムのダウンロード

デモンストレーションプロジェクトでは、ダウンロードするサンプルプログラムを自動的に設定しますので、設定を確認してください。

- [オプション]メニューから [デバッグの設定...]を選択して、Debug Settings ダイアログボックスを開いてください。

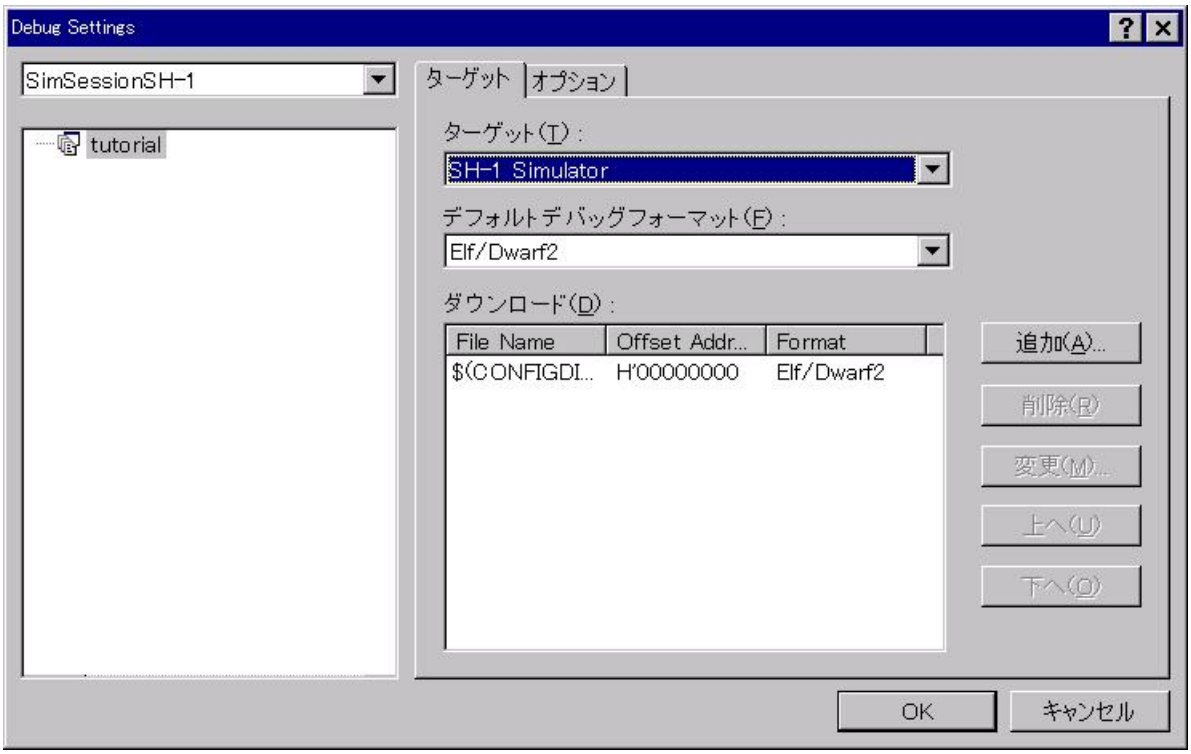


図 5.2　Debug Settings **ダイアログボックス**

- [ダウンロード]に設定しているファイルがダウンロードするファイルです。
- [OK]ボタンをクリックして Debug Settings ダイアログボックスを閉じてください。
- [デバッグ]メニューから [ダウンロード -> All Download Modules]を選択して、サンプルプログラムをダウンロードしてください。

5.2.3 ソースプログラムの表示

HEWでは、ソースレベルでプログラムをデバッグできます。「3.1 ソースファイルの編集と作成」を参照し、[Source]ウィンドウにソースプログラム（”tutorial.c”）を表示してください。

- [Workspace]ウィンドウの tutorial.c をダブルクリックして[Source]ウィンドウを開いてください。

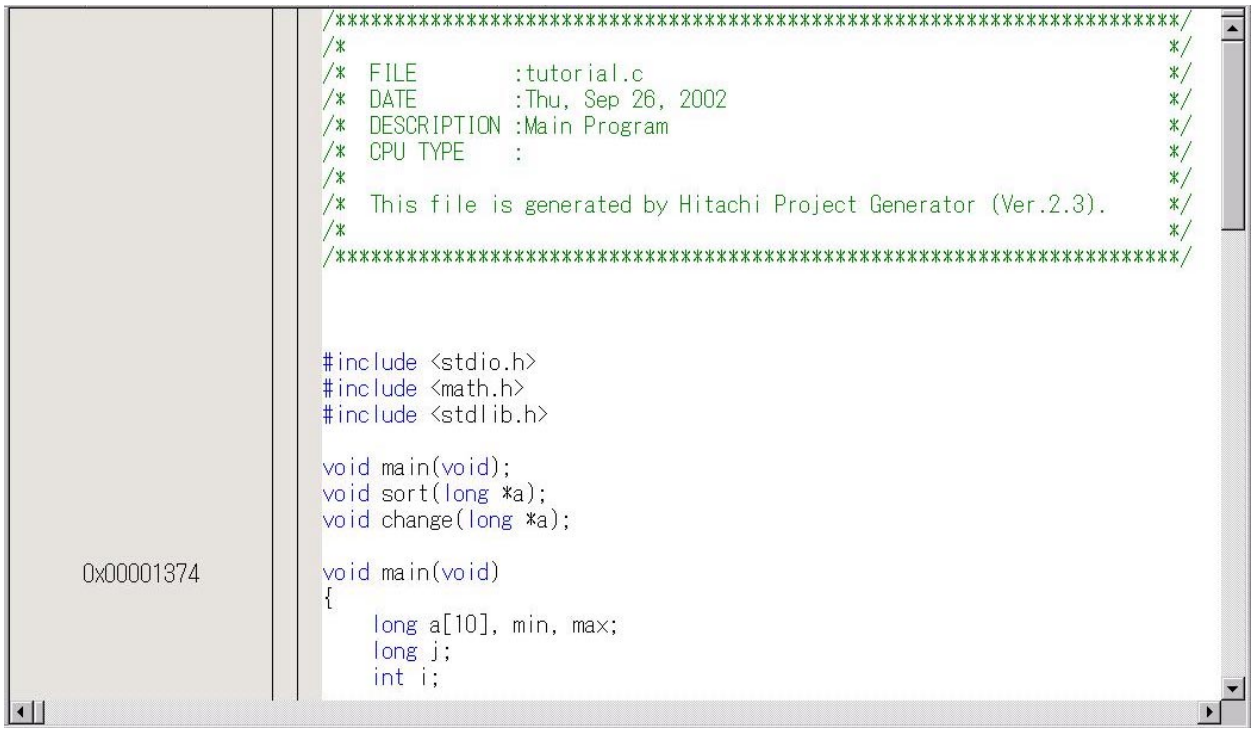


図 5.3　Source **ウィンドウ（ソースプログラムの表示）**

5.2.4 PCブレークポイントの設定

[Source]ウィンドウによって、ブレークポイントを簡単に設定できます。以下のようにしてsort関数のコール箇所にブレークポイントを設定します。

- sort 関数コールを含む行にカーソルを移動して右クリックし、ポップアップメニューの［ブレークポイントの挿入／削除］を選択してください。

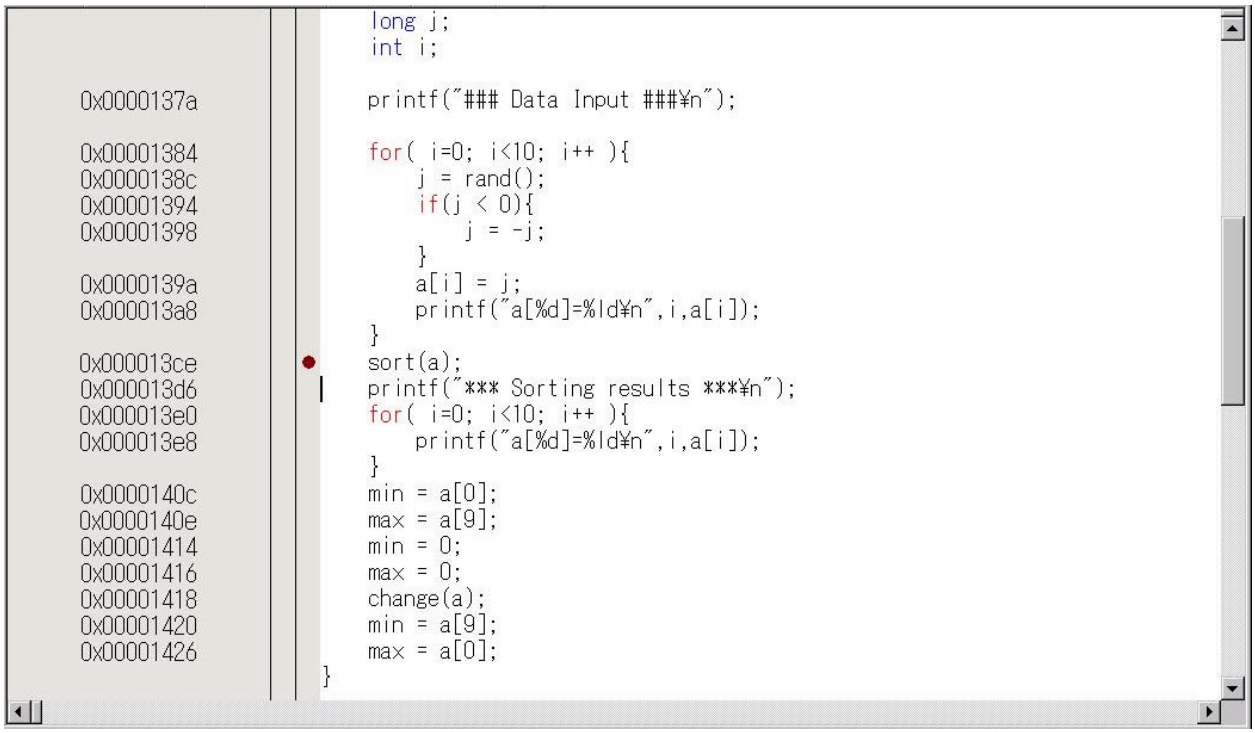


図 5.4　Source **ウィンドウ（ブレークポイントの設定）**

sort関数コールを含む行に●を表示し、そのアドレスにPCブレークポイントを設定したことを示します。

5.2.5 プロファイラの設定

- [表示]メニューから [パフォーマンス -> プロファイル]を選択し、Profile ウィンドウを開いてください。

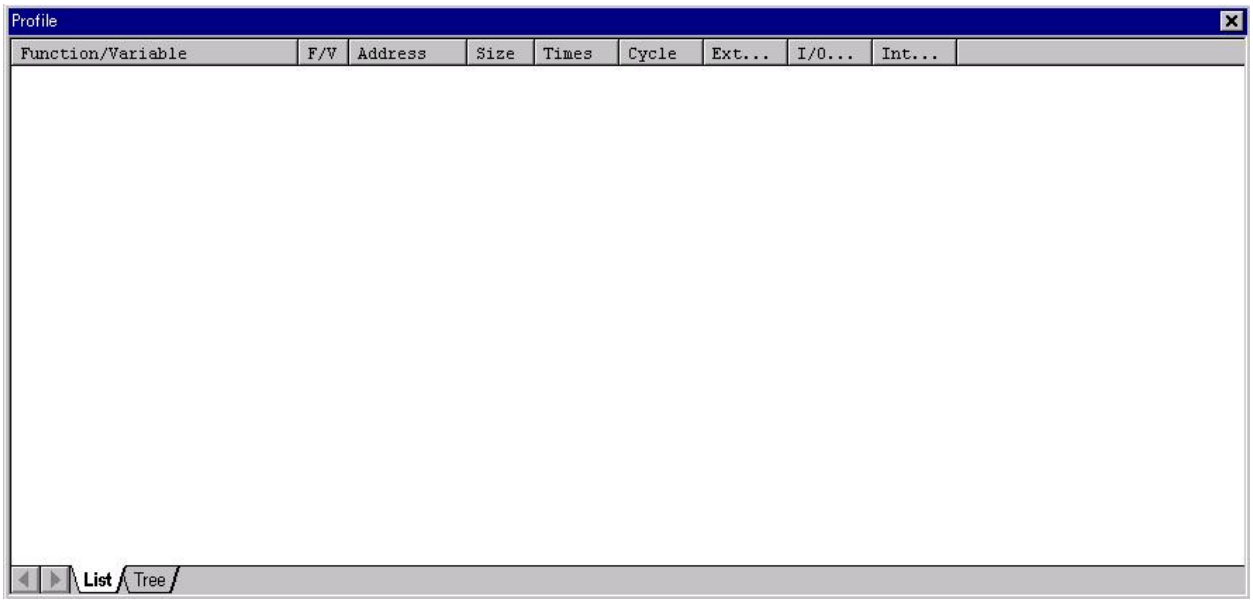


図 5.5 Profileウィンドウ

- [Profile]ウィンドウ上で右クリックしてポップアップメニューを表示し、[有効]を選択してプロファイラ情報取得を有効にしてください。

5.2.6 Simulated I/Oの設定

デモンストレーションプロジェクトでは、自動的にSimulated I/Oを設定しますので、設定を確認してください。

- [オプション]メニューから [シミュレータ -> システム]を選択して、[シミュレータシステム]ダイアログボックスを開いてください。

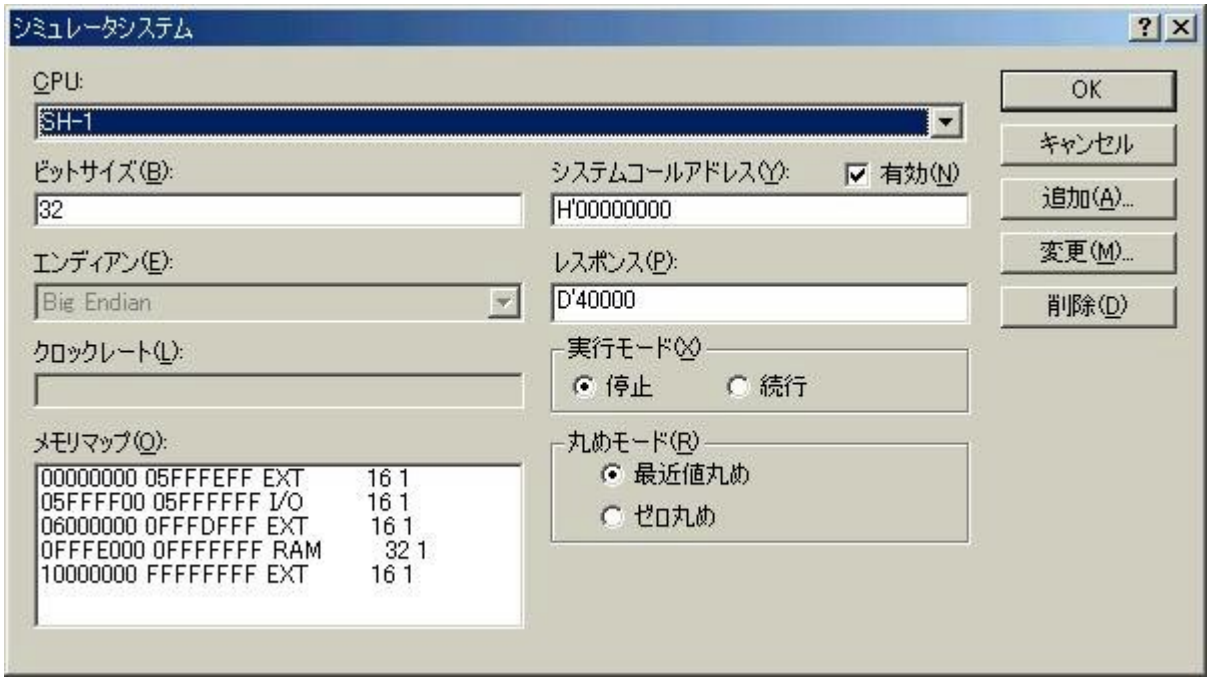


図 5.6　**シミュレータシステムダイアログボックス**

- [システムコールアドレス]の [有効]にチェックがあることを確認してください。
- [OK]ボタンをクリックして Simulated I//O を有効にしてください。
- [表示]メニューから [CPU -> I/O シミュレーション]を選択し、[Simulated I/O]ウィンドウを開いてください。[Simulated I/O]ウィンドウを開かなければ、Simulated I/O が有効になりません。

図 5.7　Simulated I/O **ウィンドウ**

5.2.7 トレース情報取得条件の設定

- [表示]メニューから [コード -> トレース]を選択し、[Trace]ウィンドウを開いてください。さらに、[Trace]ウィンドウ上で右クリックしてポップアップメニューを表示し、[設定...]を選択してください。

以下の[トレース取得]ダイアログボックスを表示します。

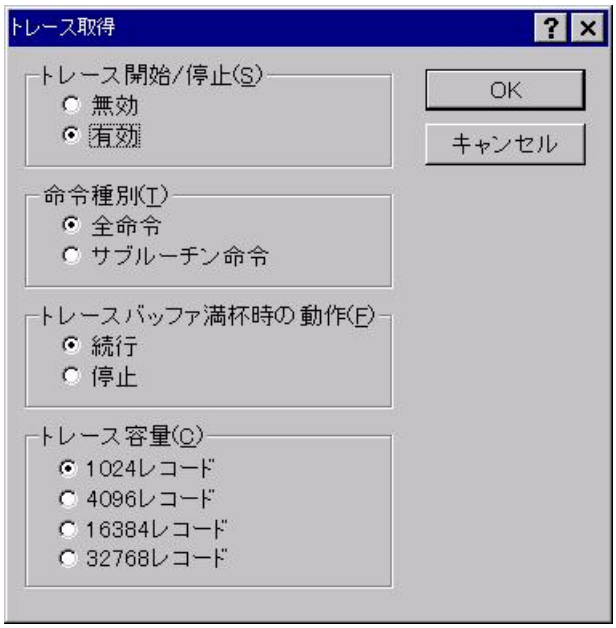


図 5.8 **トレース取得ダイアログボックス**

- [トレース取得]ダイアログボックスの [トレース開始／停止]を [有効]に設定し、[OK]ボタンをクリックしてトレース情報取得を有効にしてください。

5.2.8 スタックポインタ、プログラムカウンタの設定

プログラムを実行するために、プログラムカウンタおよびスタックポインタをリセットベクタから設定してください。サンプルプログラムのリセットベクタには、PC値としてH’800が、SP値としてH’0FFFFFF0が書いてあります。

- [デバッグ]メニューから [CPU のリセット]を選択するか、またはツールバー上の[Reset CPU]ボタンをクリックしてください。

リセットベクタからプログラムカウンタにH’800を、スタックポインタにH’0FFFFFF0を設定します。

図 5.9 Reset CPU **ボタン**

5.3　デバッグ開始

5.3.1 プログラムの実行

- プログラムを実行するには、[デバッグ]メニューから [実行]を選択するか、またはツールバー上の[Go]ボタンをクリックしてください。

図 5.10　Go **ボタン**

プログラムはブレークポイントを設定したところまで実行します。プログラムが停止した位置を示すために [Source]ウィンドウに⇒を表示します。また 停止要因として[Output]ウィンドウに”PC Breakpoint”を表示します。

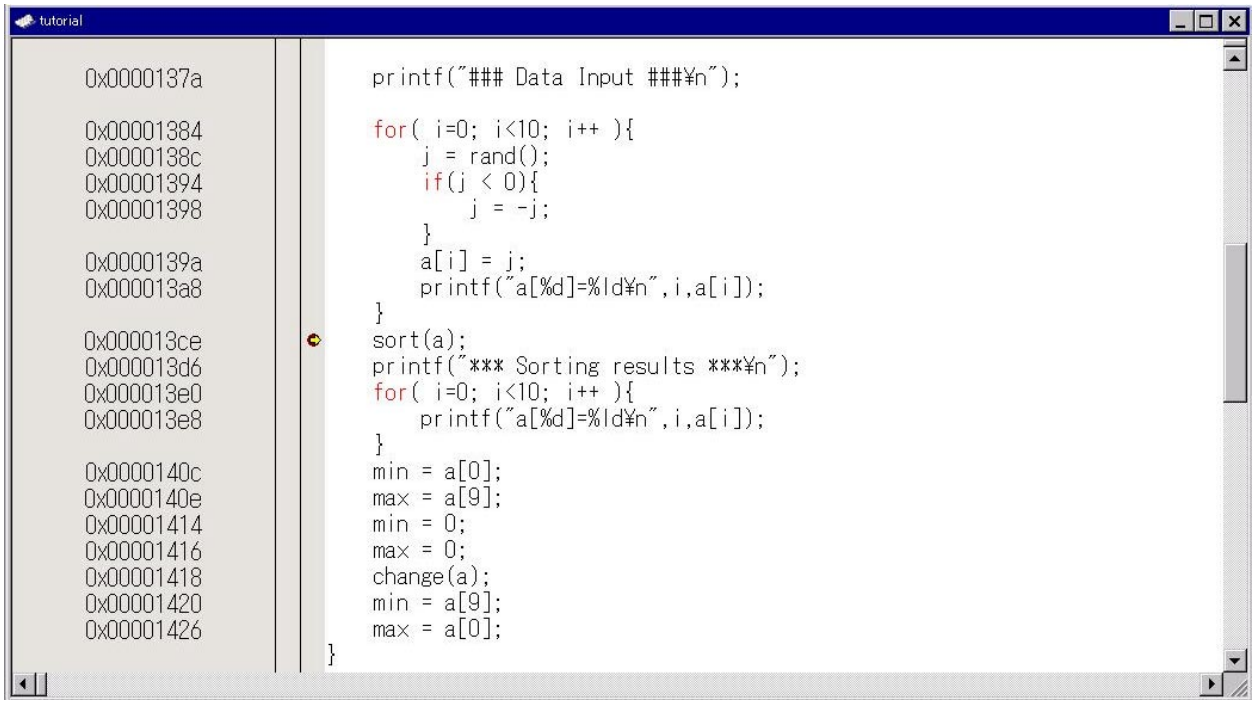


図 5.11　Source **ウィンドウ（ブレーク状態）**

[Status]ウィンドウで停止要因が確認できます。

- [表示]メニューから [CPU -> ステイタス]を選択し、[Status]ウィンドウを開いてください。さらに、[Status]ウィンドウのうち [Platform]シートを表示してください。

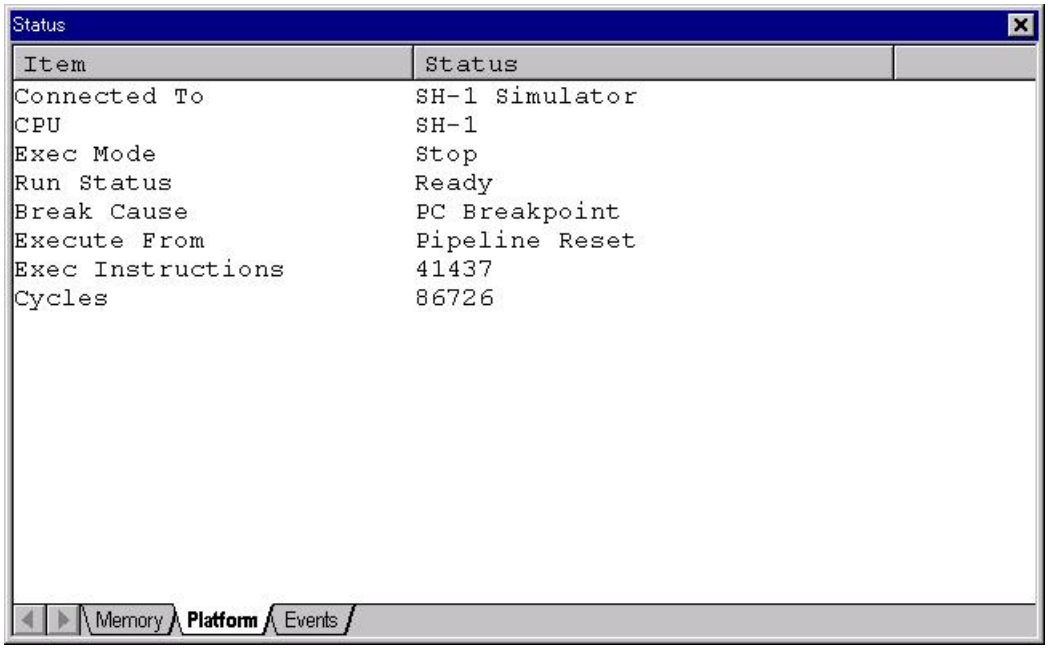


図 5.12　Status ウィンドウ

これは、以下の実行内容を表示します。

① ブレークの原因は PCブレーク

② パイプラインリセットからの実行

③ GOコマンドでの実行命令数は41,437命令

④ パイプラインリセットからの実行サイクル数は86,726サイクル

Registerウィンドウでレジスタの値が確認できます。

- [表示]メニューから [CPU -> レジスタ]を選択してください。

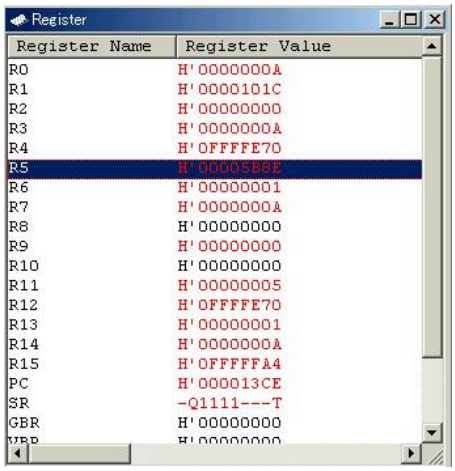


図 5.13　Register ウィンドウ

プログラム停止時の各レジスタの値を確認することができます。

5.3.2 トレースバッファの使い方

トレースバッファを使って、命令実行の履歴を知ることができます。

- [表示]メニューから [コード -> トレース]を選択し、[Trace]ウィンドウを開いてください。ウィンドウの最上部までスクロールアップしてください。

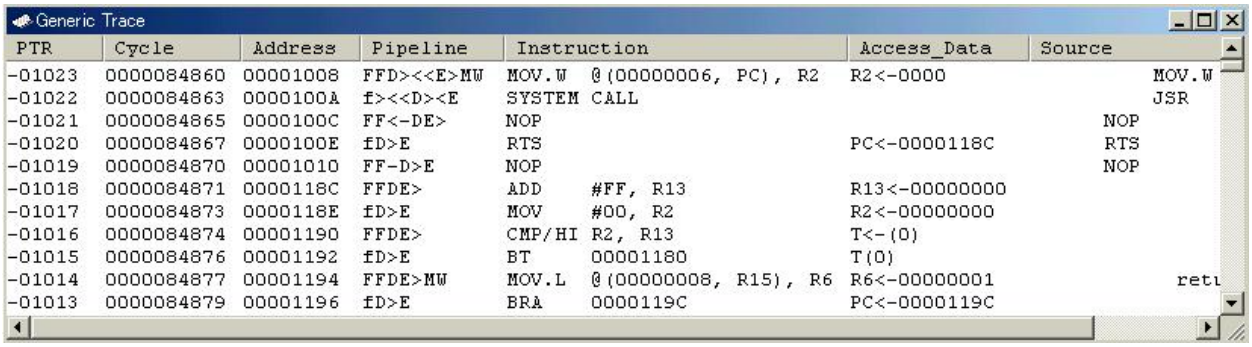


図 5.14　Trace ウィンドウ（トレース情報の表示）

5.3.3 トレース検索の実行

最初に、[Trace]ウィンドウ上で右クリックしてポップアップメニューを表示し、[トレース検索...]を選択して、[トレース検索]ダイアログボックスを開いてください。

図 5.15　トレース検索ダイアログボックス

サーチ項目 [項目]とサーチ内容 [値]を設定して [OK]ボタンをクリックすると、トレースサーチを実行します。該当するトレース情報があった場合、該当する最初の行を強調表示します。同じサーチ内容 [値]でトレースサーチを続ける場合は、[Trace]ウィンドウ上で右クリックしてポップアップメニューを表示し、[次を検索]を選択してください。次に該当する行を強調表示します。

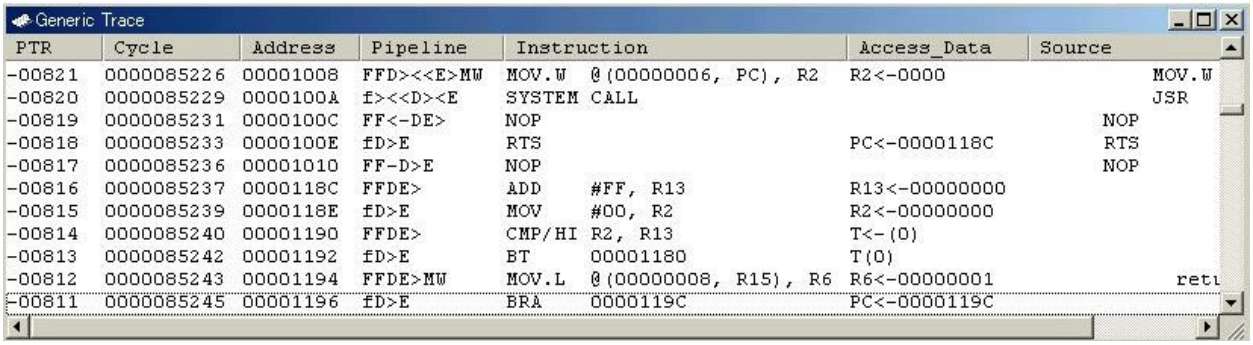


図 5.16　Trace ウィンドウ（検索結果）

5.3.4 Simulated I/Oの確認

printf関数で表示したランダムデータを[Simulated I/O]ウィンドウで確認することができます。

図 5.17　Simulated I/O **ウィンドウ**

- [Simulated I/O]ウィンドウは閉じないでください。

5.3.5 ブレークポイントの確認

プログラムに設定した全てのブレークポイントのリストを[Break]ウィンドウで確認することができます。

- [表示]メニューから [コード -> ブレークポイント]を選択してください。

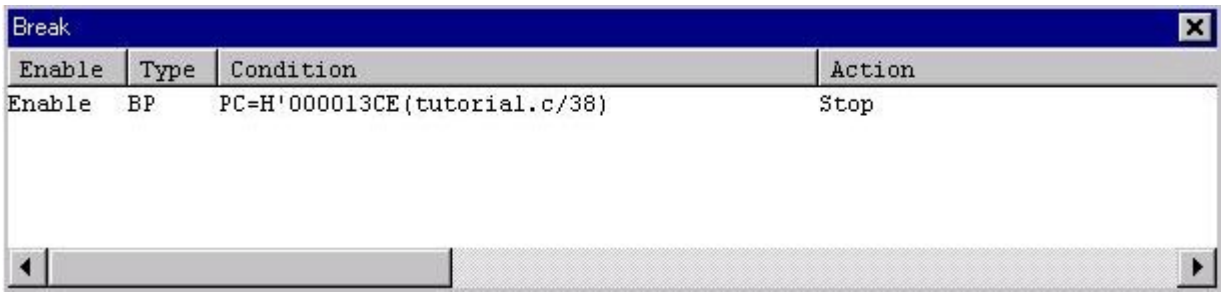


図 5.18　Break **ウィンドウ**

[Break]ウィンドウによって、ブレークポイントの設定、新しいブレークポイントの定義、およびブレークポイントの削除ができます。

- [Break]ウィンドウを閉じてください。

5.3.6 変数の参照

プログラムで使用する変数の値を[Watch]ウィンドウで確認することができます。たとえば、プログラムの始めに宣言したlong型の配列 a を見る場合：

- [表示]メニューから [シンボル -> ウォッチ]を選択し[Watch]ウィンドウを表示してください。さらに、[Watch]ウィンドウ上で右クリックしてポップアップメニューを表示し、[シンボル登録...]を選択してください。

以下の [シンボル登録]ダイアログボックスを表示します。

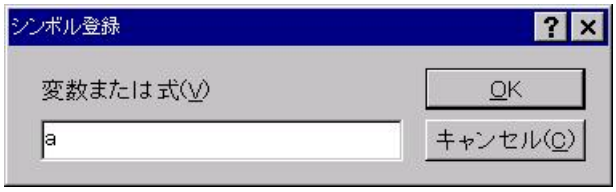


図 5.19 シンボル登録ダイアログボックス

- 配列 “a” をタイプし、[OK]ボタンをクリックします。

[Watch]ウィンドウに、long型の配列 a を表示します。
配列 a の前にあるシンボル＋をクリックすると、配列を拡張して表示します。

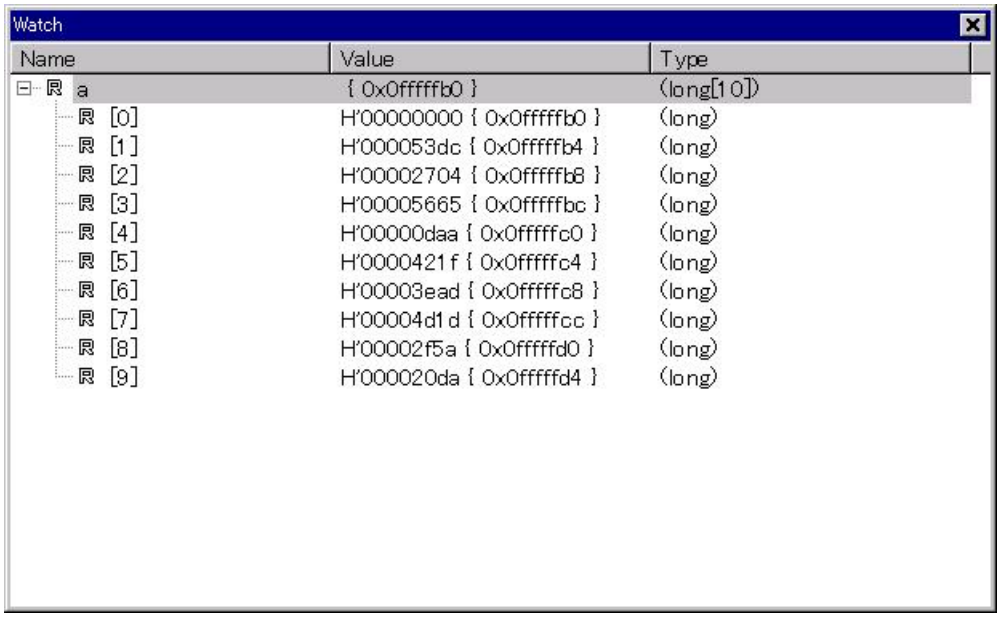


図 5.20 Watch ウィンドウ

- [Watch]ウィンドウを閉じてください。

5.3.7 プログラムのステップ実行

シミュレータ・デバッガは、プログラムのデバッグに有効な各種のステップメニューを備えています。

メニュー	説明
Step In	各ステートメントを実行します（関数内のステートメントを含む）。
Step Over	関数コールを１ステップとして、ステップ実行します。
Step Out	関数を抜け出し、関数を呼び出したプログラムにおける次のステートメントで停止します。
Step…	指定した速度で指定回数分ステップ実行します。

（１） **[Step In]**の実行

[Step In]はコール関数の中に入り、コール関数の先頭のステートメントで停止します。

- sort 関数の中に入るために､[デバッグ]メニューから [ステップイン]を選択するか､またはツールバーの [Step In]ボタンをクリックしてください。

図 5.21　Step In **ボタン**

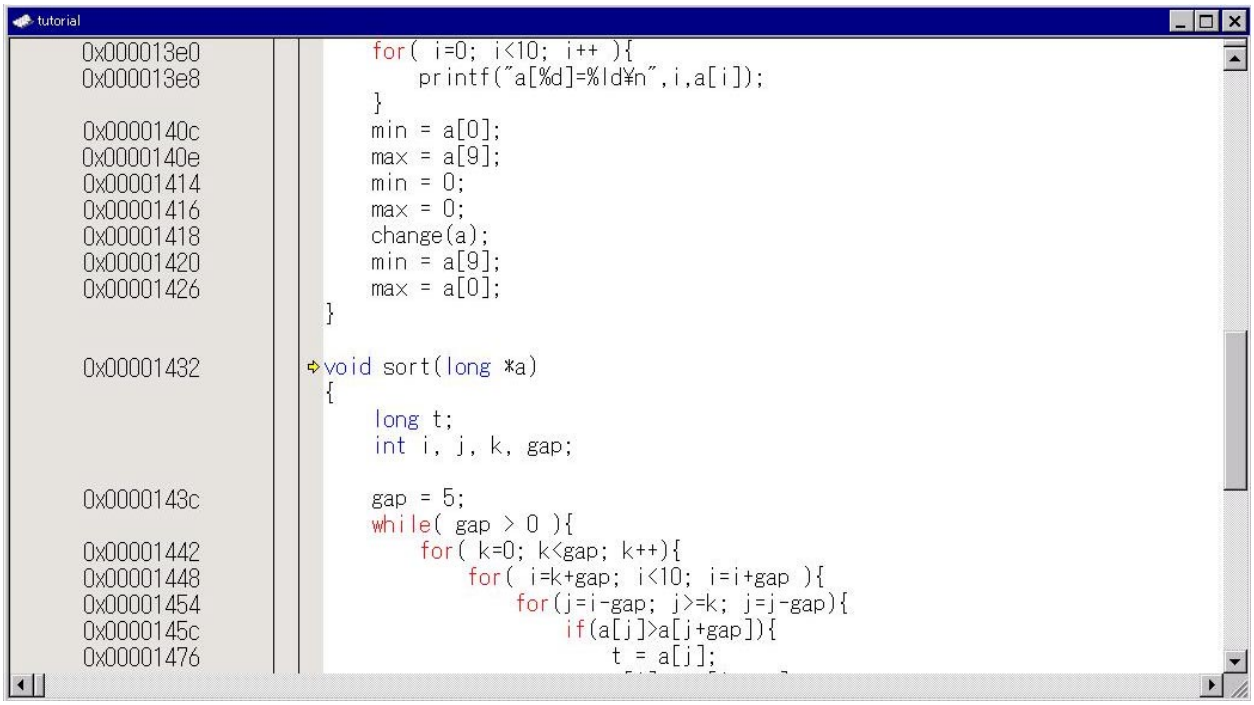


図 5.22　Source **ウィンドウ**（Step In）

- [Source]ウィンドウの PC 位置表示（⇒）が、sort 関数の先頭のステートメントに移動します。

（２） **[Step Out]**の実行

[Step Out]はコール関数の中から抜け出し、コール元プログラムの次ステートメントで停止します。

- sort 関数の中から抜け出すために、[デバッグ]メニューから [ステップアウト]を選択するか、またはツールバーの [Step Out]ボタンをクリックしてください。

図 5.23　Step Out **ボタン**

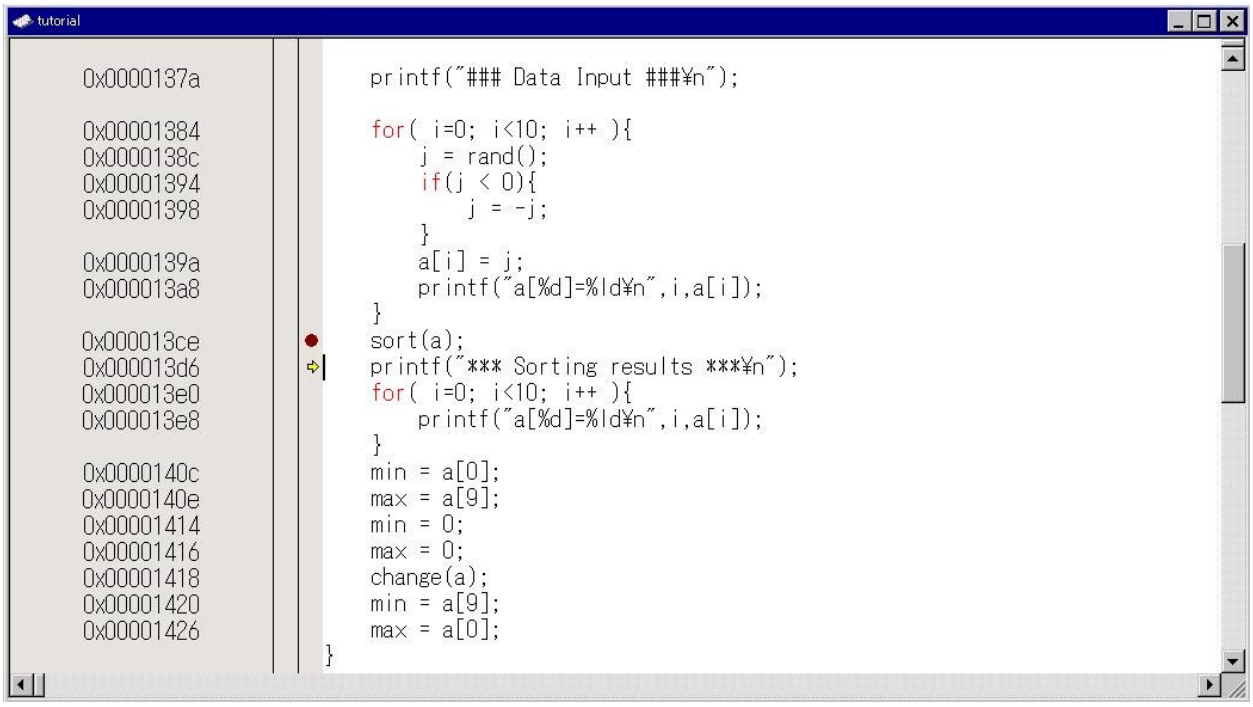


図 5.24　Source **ウィンドウ**（Step Out）

（３） **[Step Over]**の実行

[Step Over]は関数コールを１ステップとして実行し、メインプログラムの中の次のステートメントで停止します。

Printf関数中のステートメントを一度にステップ実行するために、[デバッグ]メニューから[ステップオーバ]を選択するか、またはツールバーの [Step Over]ボタンをクリックしてください。

図 5.25　Step Over **ボタン**

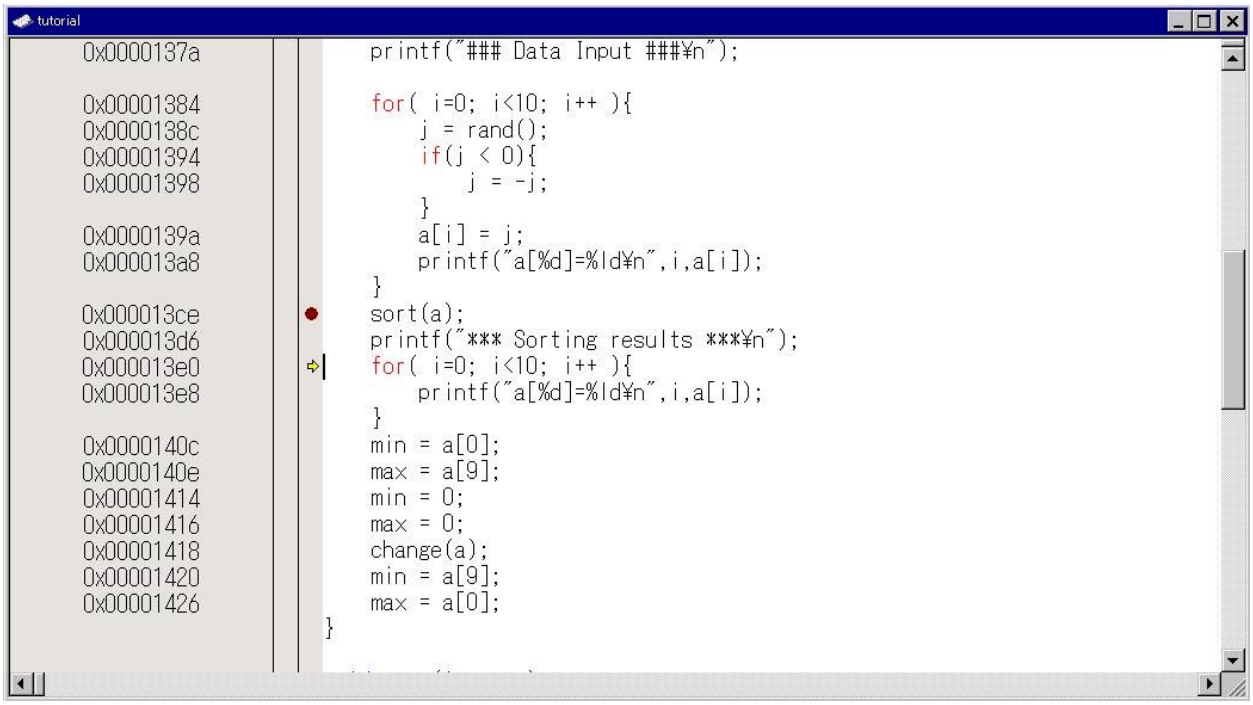


図 5.26　Source **ウィンドウ**（Step Over）

printf関数を実行すると、[Simulated I/O]ウィンドウに*** Sorting results ***を表示します。

5.3.8 プロファイラ情報の確認

プロファイラ情報を[Profile]ウィンドウで確認することができます。

- [Go]ボタンをクリックして現在の PC から継続実行すると、SLEEP 命令を実行して停止します。

（1） **[List]**シート

プロファイラ情報をリスト形式で表示します。

- [表示]メニューから [パフォーマンス -> プロファイル]を選択し、[Profile]ウィンドウを開いてください。[List]シートを表示します。

Profile

Function/Variable	F/V	Address	Size	Times	Cycle	Ext mem	I/O area	Int mem
_strlen	F	H'00003A64	H'00000012	40	1912	0	0	146
__flshbuf	F	H'000033D4	H'00000140	249	34114	250	0	5479
__modlu	F	H'000032DC	H'00000000	102	4978	0	0	138
__divlu	F	H'000030C0	H'00000000	102	4476	0	0	222
_fputc	F	H'00002E2C	H'000000DC	249	28137	249	0	3237
0x00002E12	F	H'00002E12	H'00000000	124	1962	0	0	496
0x00002E00	F	H'00002E00	H'00000000	22	308	0	0	22
0x00002DAA	F	H'00002DAA	H'00000000	102	10407	143	0	2162
0x000023AC	F	H'000023AC	H'00000000	40	15802	204	0	2256
__fmtout	F	H'000019E8	H'000009C4	22	32009	1099	0	3519
__flopen	F	H'00001884	H'00000164	3	782	54	0	50
__flclose	F	H'000017CC	H'000000B8	6	453	6	0	66
__muli	F	H'0000178C	H'00000000	10	510	0	0	60
_rand	F	H'00001754	H'00000038	10	430	40	0	40
_printf	F	H'00001714	H'00000040	22	1056	66	0	88
_freopen	F	H'000016A8	H'0000006C	3	228	6	0	45
_fclose	F	H'00001658	H'00000050	3	147	3	0	12
__quick_strcmp1	F	H'000015B4	H'00000000	18	645	42	0	42
__slow_strcmp1	F	H'000015AC	H'00000000	6	281	37	0	0
__INITSCT	F	H'00001544	H'00000000	1	3414	12	0	379

List Tree

図 5.27　List**シート（**Profile**ウィンドウ）**

__flclose関数を6回コールし、実行サイクルは453サイクル、外部メモリを6回アクセスし、内部メモリを66回アクセスしたことがわかります。

コール回数が多い関数や遅いメモリを多くアクセスする関数など、プログラム性能のクリティカルパスを探すことができます。

（2） **[Tree]**シート

プロファイラ情報をツリー形式で表示します。

- [Tree]シートを選択してください。関数名をダブルクリックすることによりツリー構造を拡張または縮小します。

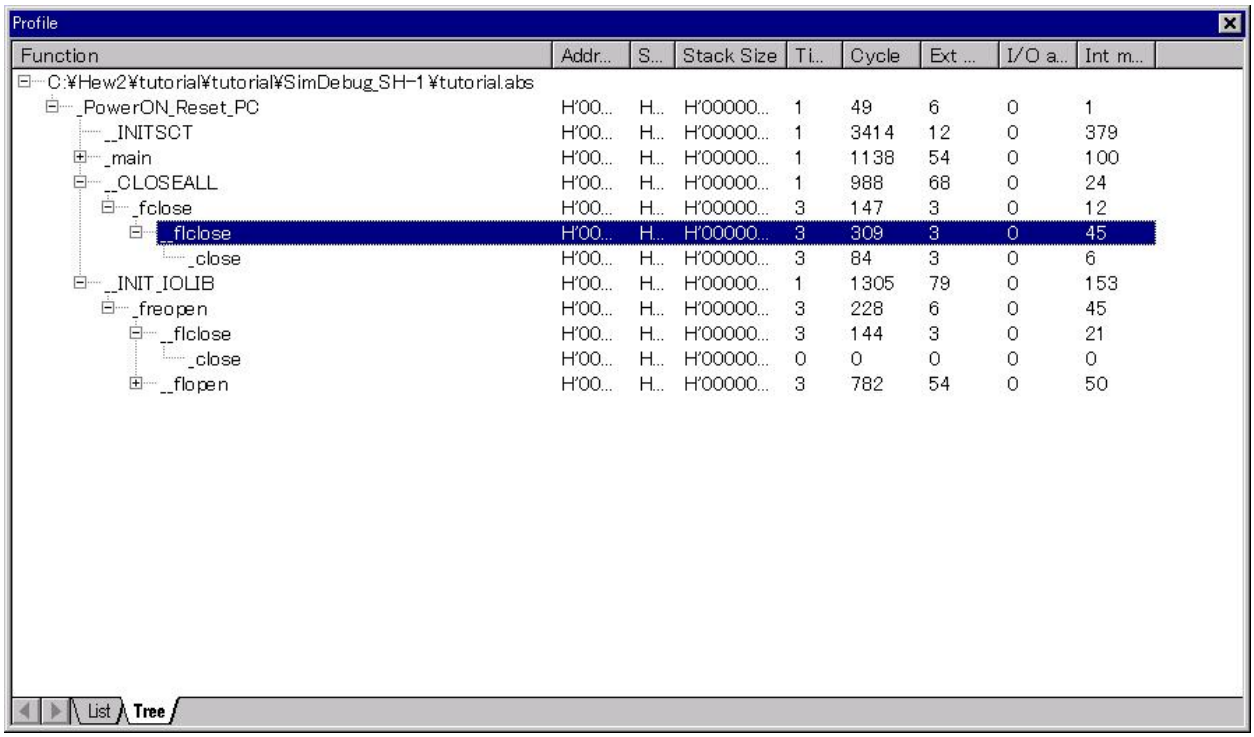


図 5.28　Tree**シート（**Profile**ウィンドウ）**

__flclose関数が_fclose関数から３回コールされ、その時の実行サイクルは309サイクル、外部メモリを3回アクセスし、内部メモリを45回アクセスしたことがわかります。

（3） プロファイルーチャートウィンドウ

[プロファイルーチャート]ウィンドウで関数の呼び出し関係を表示します。

- [Profile]ウィンドウ上で__flclose 関数を選択してから、右クリックしてポップアップメニューを表示し、[チャート表示]を選択して[プロファイルーチャート]ウィンドウを表示してください。

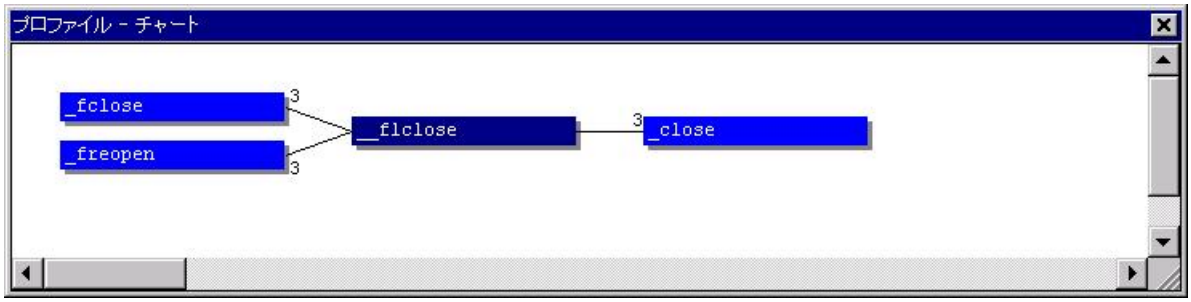


図 5.29 プロファイルーチャートウィンドウ

__flclose関数が_fclose関数から３回、_freopen関数から３回コールされたことがわかります。
また、_close関数を３回コールしたことがわかります。

以上で、シミュレータ・デバッガを使用したチュートリアルは終了です。

6　ＨＥＷの終了

通常、HEWは[ファイル -> アプリケーションの終了]で終了します。終了時、使用していたセッションの内容を保存するか確認するメッセージボックス（図 6.1）を表示します。この終了時の設定やHEW再起動時の設定は[ツール -> オプション...]で変更できます。詳しくは、「Hitachi Embedded Workshop 2 ユーザーズマニュアル」を参照してください。

なお、ユーザ登録ツールは、HEWが終了しても連動して終了しませんので個別に終了させてください。

図 6.1　HEW**終了時の確認**

SuperH RISC engine High-performance Embedded Workshop 2
チュートリアル

ルネサス エレクトロニクス株式会社
神奈川県川崎市中原区下沼部1753 〒211-8668

ADJ-702-354B