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。

User's Manual

RENESAS

H8S,H8/300シリーズ
High-performance Embedded Workshop 3

ユーザーズマニュアル

(Windows®98/Me、Windows NT®4.0、
Windows®2000およびWindows®XP用)

ルネサス エレクトロニクス
www.renesas.com

Rev.2.00　2004.06

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。

はじめに

High-performance Embedded Workshop （以下、HEW と略します）は、ルネサステクノロジマイクロコンピュータの組み込み用アプリケーションの開発を強力にサポートするツールです。おもな特徴をまとめると次のようになります。

- 使い勝手の良いインタフェースを活用したコンパイラ、アセンブラ、リンケージエディタなどのオプションが設定できるカスタマイズ可能なプロジェクトビルドシステム。
- プログラムを読みやすくするシンタックス色付け機能を持つ統合化テキストエディタ。
- ユーザ独自のツールを実行するための環境設定。
- 同一アプリケーション内でのビルドおよびデバッグを可能にする統合化デバッガ。
- バージョン管理サポート。

HEW は 2 つの目的で設計されています。一つはユーザに強力な開発ツールを提供すること、そしてもう一つは、それらのツール類を統合して使いやすくすることです。

このマニュアルについて

このマニュアルでは HEW システムについて述べています。本マニュアルは２つの部分から構成されています。HEW 編では HEW の基本的な使い方に関する情報、HEW 環境のカスタマイズ、および HEW のビルド機能について説明します。なお、HEW 編は SH シリーズの図を使用しています。シミュレータ・デバッガ編ではデバッグ機能について説明します。

このマニュアルでは C/C++言語、アセンブリ言語の書き方や、オペレーティングシステムの使い方、個々のデバイスに適したプログラムの書き方などについては説明していません。それらについては、各々のマニュアルを参照してください。

HEW は、インストール上、各種言語にカスタマイズされています。このマニュアルでは、HEW アプリケーションの日本語版について説明します。

このマニュアルの記号

このマニュアルで使われている記号の意味を説明します。

表 1: **記号一覧**

記号	意味
[Menu->Menu Option]	**太字と** '->' **はメニューオプションを示します** (**例** [File->Save As...])
FILENAME.C	**大文字の名前はファイル名を示します**
"**文字列の入力**"	**下線は入力する文字列を示します** ("" **を省く**)
Key + Key	**キー入力を示します。例えば、**CTRL+N **キーでは** CTRL **キーと** N **キーを同時に押します**
➲ (**「操作方法」マーク**)	**このマークが左端にあるとき、その右の文章は何かの操作方法を示します**

目次

HEW 編

1. 概要

この章では HEW の基本概念を説明します。Windows®アプリケーションに慣れていないユーザのために、次章以降で必要となる情報を提供します。

1.1 ワークスペース、プロジェクト、ファイル

ワードプロセッサでドキュメントを作成、修正できるのと同じように、HEW ではワークスペースを作成、修正できます。
ワークスペースはプロジェクトを入れる箱と考えることができます。同じように、プロジェクトはプロジェクトファイルを入れる箱と考えることができます。したがって各ワークスペースにはプロジェクトが一つ以上あり、各プロジェクトにはファイルが一つ以上あります。この構成を図 1.1 に示します。

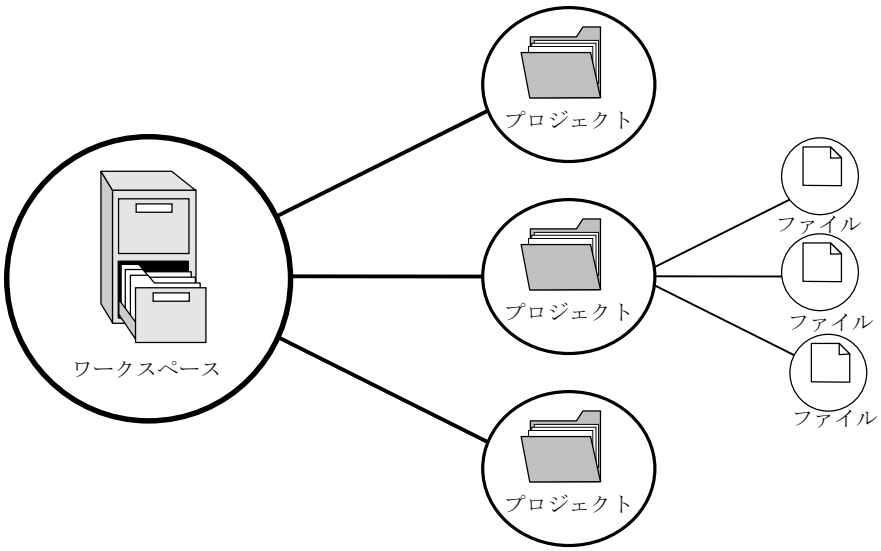


図 1.1: **ワークスペース、プロジェクト、ファイル**

ワークスペースでは関連したプロジェクトを一つにまとめることができます。例えば、異なるプロセッサに対して一つのアプリケーションを構築しなければならない場合、または、アプリケーションとライブラリを同時に開発している場合などに便利です。さらに、ワークスペース内でプロジェクトを階層的に関連づけることができます。つまり、一つのプロジェクトを構築すると、その子プロジェクトが最初に構築されます。

ワークスペースを活用するには、ユーザは、まずワークスペースにプロジェクトを追加して、そのプロジェクトにファイルを追加しなければなりません。

1.2 メインウィンドウ

HEW のメインウィンドウを図 1.2 に示します。

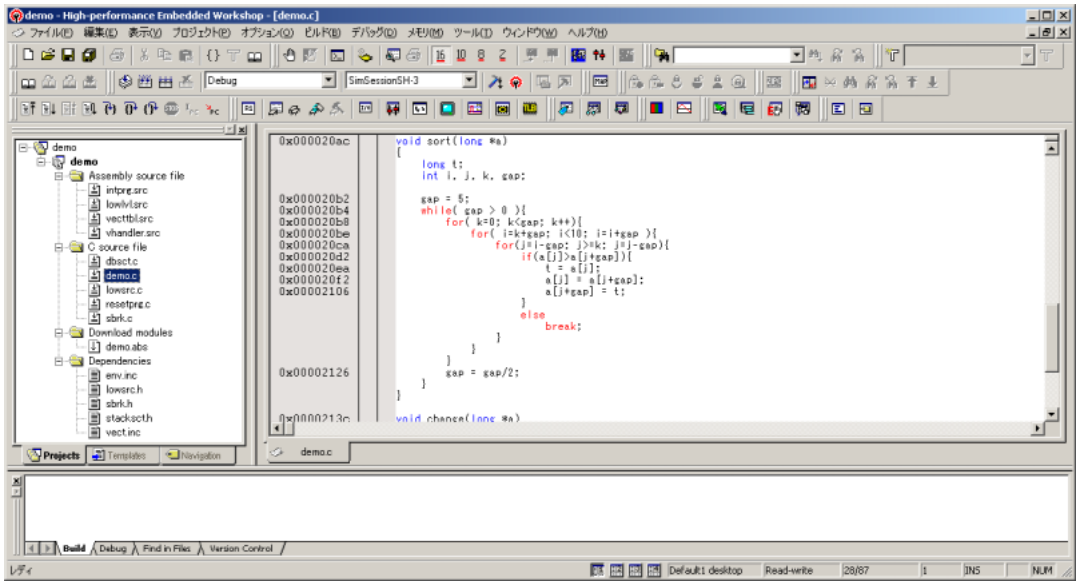


図 1.2: HEW **メインウィンドウ**

HEW にはメインウィンドウが 3 つあります。ワークスペースウィンドウ、エディタウィンドウ、アウトプットウィンドウです。ワークスペースウィンドウには現在そのワークスペースにあるプロジェクトやファイルを示します。エディタウィンドウではファイルを表示、編集できます。アウトプットウィンドウにはさまざまな処理結果（ビルド、バージョン管理コマンドなど）を表示します。

1.2.1 タイトルバー

タイトルバーには、現在開いている、プロジェクト名、ファイル名が表示されます。また、“最小化”ボタン、“最大化”ボタン、“閉じる”ボタンがあります。“最小化”ボタンをクリックすると Windows のタスクバー上に HEW が最小化されます。“最大化”ボタンをクリックすると HEW がフルスクリーンに表示されます。“閉じる”ボタンをクリックすると HEW を閉じることができます。（これは[ファイル->アプリケーションの終了] を選ぶか”ALT+F4”キーを押すのと同じです）。

1.2.2 メニューバー

メニューバーには次の 11 のメニューがあります。 [ファイル]、[編集]、[表示]、[プロジェクト]、[オプション]、[ビルド]、[デバッグ]、[メモリ]、[ツール]、[ウィンドウ]、[ヘルプ]です。メニューのオプションはすべてこれら 11 のメニューの下にグループ化されています。例えば、ファイルを開きたいときには[ファイル]メニューの下のオプションを選びます。ツールのセットアップをしたいときには、[ツール]メニューを選びます。メニューのオプションの機能については後の章で説明します。ここでは、各オプションの簡単な紹介をします。

1.2.3 ツールバー

ツールバーにより、使う頻度の高いオプションを簡単に利用できます。デフォルトでは[ブックマー

ク]、[デバッグ]、[デバッグラン]、[エディタ]、[検索]、[標準]、[テンプレート]、[バージョン管理]、[差分]、[Map]、[ツール]の 11 のツールバーがあります（図 1.3～図 1.13 参照）。ツールバーの作成や変更は [ツール->カスタマイズ...] メニューオプションで指定できます。 (詳細については 6 章「環境のカスタマイズ」を参照してください。)

図 1.3: **ブックマーク ツールバー**

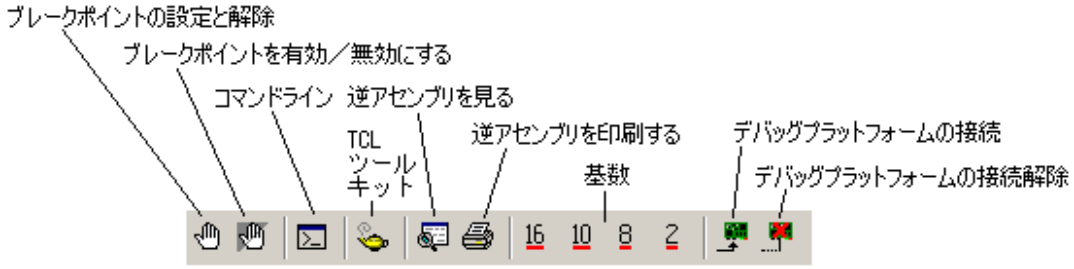


図 1.4: **デバッグ ツールバー**

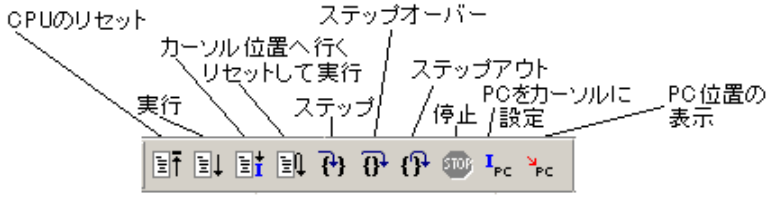


図 1.5: **デバッグランツールバー**

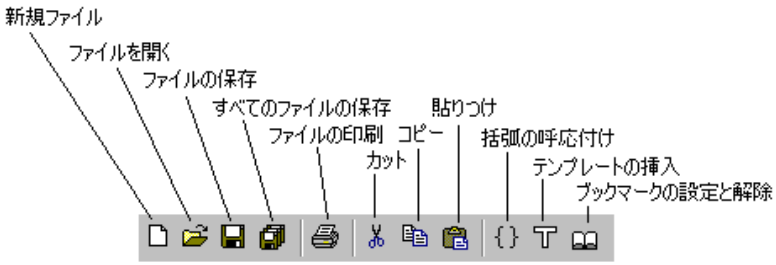


図 1.6: **エディタ ツールバー**

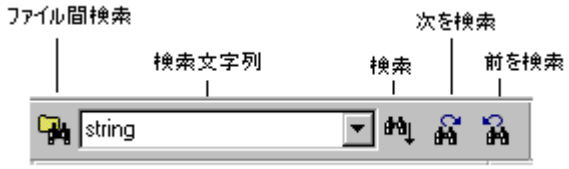


図 1.7: **検索ツールバー**

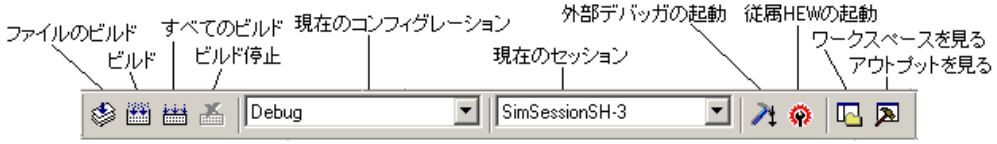


図 1.8: **標準ツールバー**

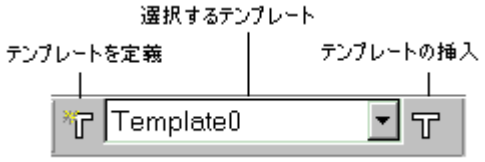


図 1.9: **テンプレートツールバー**

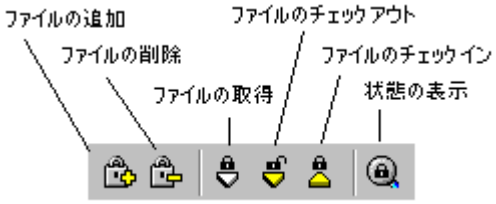


図 1.10: **バージョン管理ツールバー**

図 1.11: **差分ツールバー**

図 1.12: **マップツールバー**

図 1.13: **ツール ツールバー**

[標準]ツールバーがドッキング（連結）状態のとき、図 1.14.(i)に示すコントロールバーが表示されます。ドッキング状態の[標準]ツールバーの位置を移動したいときはコントロールバーを移動先までドラッグします。（ドラッグとは、マウスの左ボタンを押下したまま目的の場所まで移動してからボタンを離すことをいいます。）図 1.14.(i) がドッキング状態、1.14.(ii) がフローティング（浮遊）状態の[標準]ツールバーを示します。

(i)

(ii)

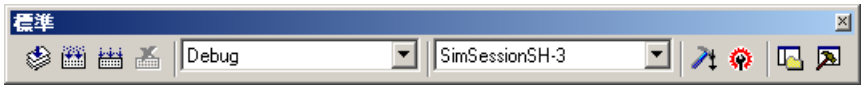


図 1.14: **標準ツールバーのドッキング／フローティング状態**

➲ ツールバーをドッキング状態にするには
　フローティング状態のツールバーのタイトルバーをダブルクリックしてください。
　または、フローティング状態のツールバーのタイトルバーをドッキング状態のウィンドウ、メニューバー、ツールバー、またはHEWメインウィンドウの端までドラッグします。バーの形が変わります。

➲ ツールバーをフローティング状態にするには
　ドッキング状態のツールバーのコントロールバーをダブルクリックしてください。
　または、ドッキング状態のツールバーのコントロールバーをHEWのメインウィンドウおよびその他のドッキング状態のウィンドウ、メニューバー、またはツールバーの端から外れるようにドラッグしてください。

1.2.4 ワークスペースウィンドウ

ワークスペースウィンドウにはタブが 3 つあります。 [Projects]タブには現在のワークスペース、プロジェクト、ファイルを示します（図 1.15）。アイコンをダブルクリックしてプロジェクトファイルや個々のファイルを開くことができます。

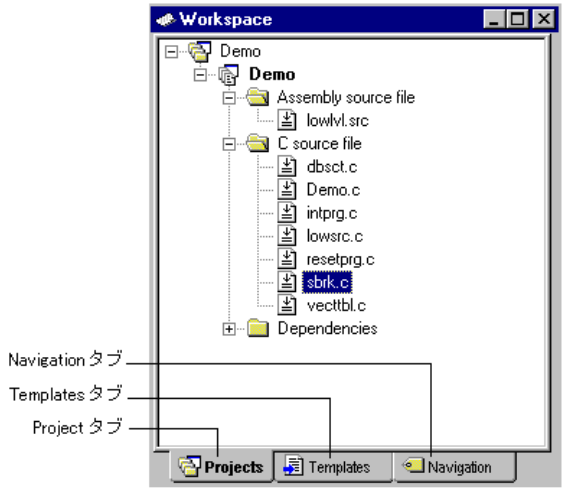


図 1.15: [**ワークスペース**] **ウィンドウ** [Projects] **タブ**

[Navigation]タブによりプロジェクトファイルの中のテキスト部へジャンプできます。[Navigation]タブに実際に表示される内容は、現在、何がインストールされているかによって異なります。図 1.16 には例えば ANSI 規格 の C 関数一覧を示します。ワークスペースウィンドウの詳細については 2 章「ビルドの基本」を参照してください。

[Templates]タブにはテンプレートの設定を表示します。テンプレートの詳細については 4.12 「テンプレート」を参照してください。

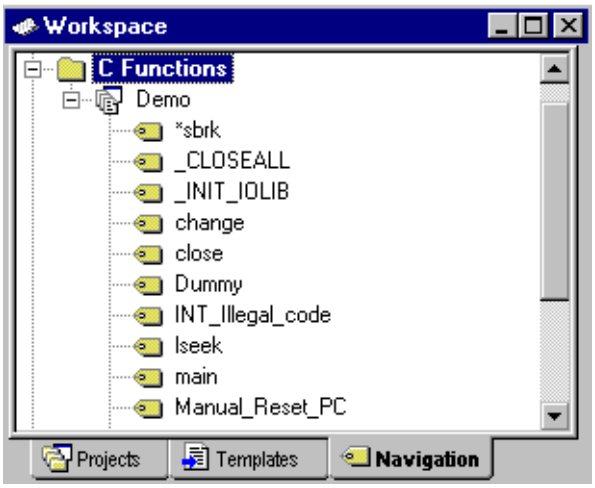


図 1.16: [**ワークスペース**] **ウィンドウ** [Navigation] **タブ**

➲ ワークスペースウィンドウやアウトプットウィンドウでドッキングを許すには

ウィンドウ上で右マウスボタンをクリックしてください。すると、ポップアップメニューが表示されます。ここで [ドッキングビュー] にチェック印が付いている場合、ドッキングが許されています。チェック印が外れている場合、ドッキングは許されていません。 [ドッキングビュー] を選択するとチェック印が付いたり外れたりします。

[ドッキングビュー] にチェック印が付いている場合、ウィンドウを HEW メインウィンドウや他のドッキング状態のウィンドウの端に連結できます。同じく[ドッキングビュー] にチェック印が付いている場合、ウィンドウを他の HEW のウィンドウ上や HEW メインウィンドウの外でフローティング状態にすることができます。図 1.17.(i) にはドッキング状態のワークスペースウィンドウ、図 1.17.(ii) にはフローティング状態のワークスペースウィンドウを示します。

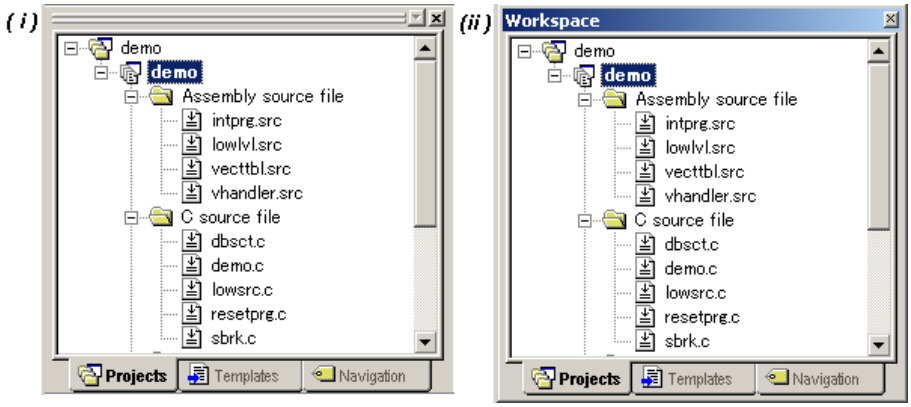


図 1.17: **ワークスペースウィンドウのドッキング／フローティング状態**

ワークスペースウィンドウやアウトプットウィンドウがドッキング状態のとき、図 1.18 に示すコントロールバーが表示されます。ドッキング状態のウィンドウを移動したいとき、コントロールバーを移動先までドラッグしてください。

図 1.18: **ドッキング状態ウィンドウのコントロールバー**

➲ ワークスペースウィンドウやアウトプットウィンドウをドッキング状態にするには

ワークスペースウィンドウやアウトプットウィンドウをドッキング状態にするにはポップアップメニューで[ドッキングビュー] にチェック印が付いている必要があります。（ポップアップメニューはウィンドウ上でマウスの右ボタンをクリックすると表示されます。）その上で、フローティング状態のウィンドウのコントロールバーをダブルクリックしてください。

または、フローティング状態のウィンドウのタイトルバーを、移動先のドッキング状態のウィンドウ、メニューバー、ツールバー、またはHEWのメインウィンドウの端までドラッグしてください。フローティング状態のウィンドウの形が変わります。

➲ ワークスペースウィンドウやアウトプットウィンドウをフローティング状態にするには

ワークスペースウィンドウやアウトプットウィンドウをフローティング状態にするにはポップアップメニューで[ドッキングビュー] にチェック印が付いている必要があります。（ポップアップメニューはウィンドウ上でマウスの右ボタンをクリックすると表示されます。）その上で、ドッキング状態のウィンドウのコントロールバーをダブルクリックしてください。

または、ドッキング状態のウィンドウのコントロールバーを、HEWのメインウィンドウや他のドッキング状態のウィンドウ、メニューバー、ツールバーの端から外れるようにドラッグしてください。

➲ ワークスペースウィンドウやアウトプットウィンドウを隠すには
ウィンドウの右上端にある“閉じる”ボタンをクリックしてください。または固定していないウィンドウの中で右マウスボタンをクリックし、ポップアップメニューから [非表示] を選んでください。

➲ ワークスペースウィンドウやアウトプットウィンドウを表示するには
ワークスペースウィンドウを表示するには[表示-> ワークスペース] を、 アウトプットウィンドウを表示するには[表示->アウトプット]を選んでください。

1.2.5 エディタウィンドウ

エディタウィンドウではプロジェクトのファイルを操作します。同時に複数のファイルを開いたり、任意の順序にファイルを切り替えたり、並べ替えたり、編集したりできます。デフォルトでは、エディタウィンドウはブック形式で表示されます。各テキストファイルにはタブがあります（図 1.19）。

エディタには、ウィンドウの左側に余白があります。これにより、ブックマークとブレークポイントの位置を速く簡単に設定できます。カラムの用途やその表示する情報について知りたい場合は、そのカラムにカーソルを置いてください。これを説明するツールチップが表示されます。

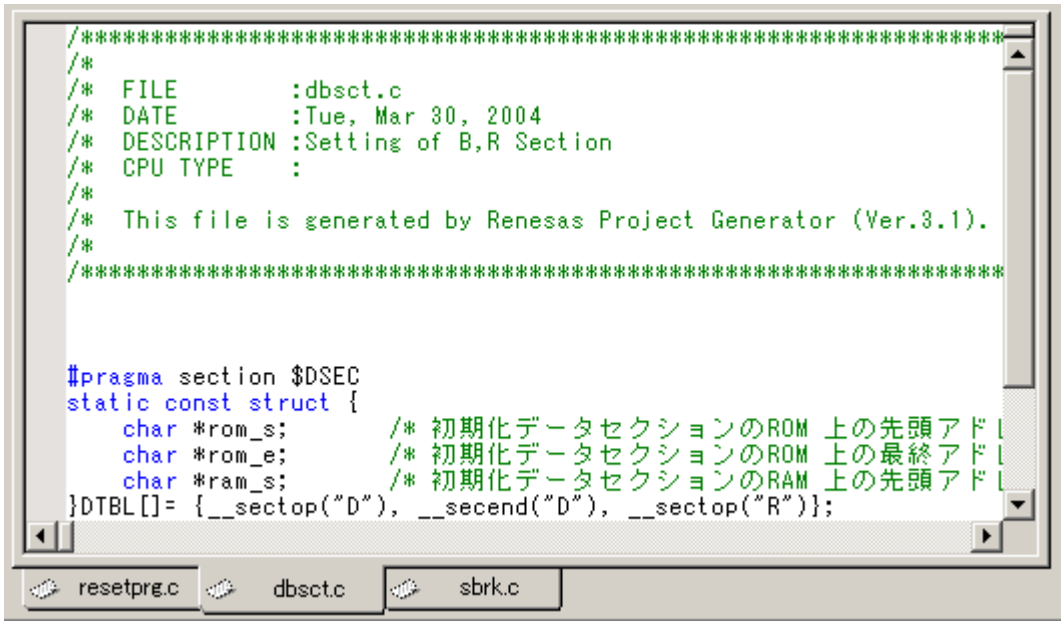


図 1.19: **エディタウィンドウ**

エディタウィンドウは[表示形式]ダイアログボックスでカスタマイズできます。[表示形式]ダイアログボックスは [ツール->表示形式...] メニューオプションから開くことができます。[表示形式]ダイアログボックスでは、エディタウィンドウのフォントやテキストの色、タブ文字の変更などができます。また、HEW でインストールした他のビューの外観も変更できます。HEW エディタ以外のエディタを使う場合は、使用するエディタを[オプション]ダイアログボックスで指定してください。[オプション]ダイアログボックスは、[ツール->オプション…] メニューオプションから開くことができます。エディタの使用方法や構築については、4 章「エディタの使用」を参照してください。

1.2.6 アウトプットウィンドウ

デフォルトではアウトプットウィンドウに 4 つのタブが表示されます。[Build]タブには任意のビルド実行（コンパイラ、アセンブラなど）の出力を示します。ソースファイルにエラーがある場合、[Build]タブにはエラーとソースファイル名と行番号が表示されます。エラーをダブルクリックすると、ソースファイルの行にジャンプするので、エラー箇所をすばやく発見できます。また、ダブルクリックにより、エラーまたは警告がステータスバーに表示されます。

注意　アウトプットウィンドウには、キーボードショートカット”SHIFT+ESC”があります。これを使用するとアウトプットウィンドウを閉じます。

図 1.20: **アウトプットウィンドウ**

[Debug]タブにはあらゆるデバッガ処理の出力を示します。情報を表示する必要のあるデバッグツールから、このウィンドウに出力が送られます。

[Find in Files]タブには最後の”Find in Files”操作の結果を示します。“Find in Files”を使用するには、[編集->ファイル内から検索…]メニューオプションを選ぶか、ツールバーの[ファイルの中から検索]ボタンをクリックしてください。“Find in Files”の使い方の詳細については、4 章「エディタの使用」を参照してください。

[Version Control]タブにはバージョン管理操作の結果を示します。このタブは、バージョン管理システムを使っているときだけ表示されます。バージョン管理の詳細については、7 章「バージョン管理」を参照してください。

1.2.7 ステータスバー

ステータスバーは 8 つの領域に分かれており、現在の HEW の状態を表示します。図 1.21 にステータスバーを示します。

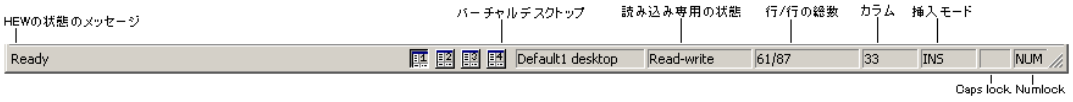


図 1.21:**ステータスバー**

1.3 ヘルプ機能

[ヘルプ]メニューは HEW メニューバーの右端にあります。[ヘルプ]メニューには[トピック]オプションがあります。[トピック]オプションを選ぶと、HEW ヘルプウィンドウのメイン画面が表示されます。

特定のダイアログボックスに関するヘルプを参照したいときは、各ダイアログボックスの右上端にあるコンテキスト依存ヘルプボタンをクリックしてください（図 1.22）。

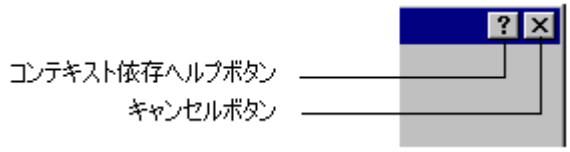


図 1.22: **ヘルプ ボタン**

コンテキスト依存ヘルプボタンをクリックすると、マウスポインタが？(クエスチョンマーク)付きのポインタに変わります。この状態で、ダイアログボックスの一部をクリックすると、その部分に関するヘルプを表示できます。

または、ある部分を選んで F1 キーを押下すると、その部分のヘルプを表示します。

1.4 HEW を起動する

HEW を起動するには Windows®の[スタート]メニューを開き、[プログラム]を選択し、 [Renesas High-performance Embedded Workshop]を選択し、HEW のショートカットを選びます。デフォルトで図 1.23 に示す[ようこそ!]ダイアログボックスが開きます。

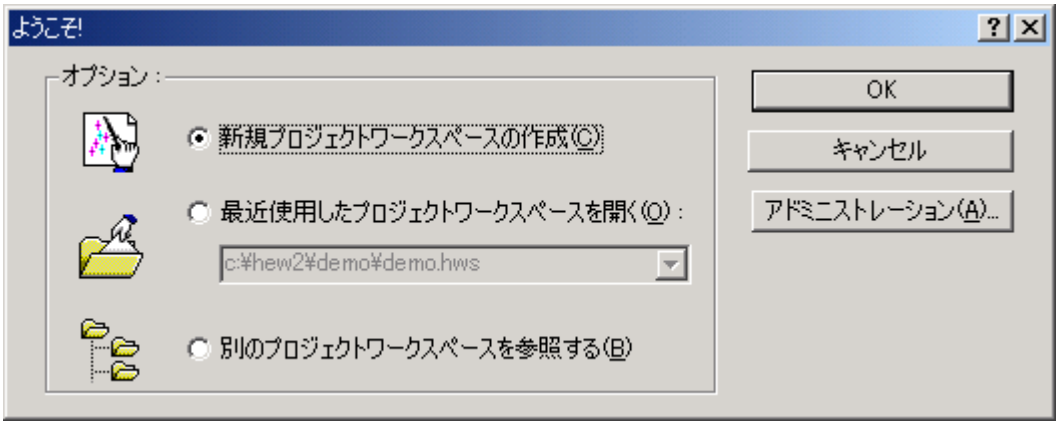


図 1.23: **ようこそ! ダイアログボックス**

新規ワークスペースを作成するには[新規プロジェクトワークスペースの作成]を選択し、[OK]ボタンをクリックしてください。最近開いたワークスペースを開くには[最近使用したプロジェクトワークスペースを開く]を選択し、ドロップダウンリストから開きたいワークスペースを選択し、[OK]ボタンをクリックしてください。最近開いたワークスペースのリストには、最近使ったワークスペースファイルリストで見るものと同じ情報が表示されます。このリストはファイルメニュー上に表示されます。ワークスペースファイルを指定してワークスペースを開くには[別のプロジェクトワークスペースを参照する]を選択し、[OK]ボタンをクリックしてください。HEW にツールを登録したり、HEW からツールの登録を外したりするには[アドミニストレーション]ボタンをクリックしてください。(詳細は、5 章、「ツール管理」を参照してください。) ワークスペースを開かないで HEW を使うには[キャンセル]をクリックしてください。

1.5 HEW を終了する

HEW を終了するには [ファイル->アプリケーションの終了]を選ぶか、“Alt+F4”キーを押下するか、システムメニューから[閉じる]オプションを選んでください。（システムメニューは HEW タイトルバーの最も左上側にあるアイコンをクリックすると開きます。）ワークスペースが開いているときは、前節で説明したワークスペースを閉じる操作を行います。

1.6 ツールシステム概要

ユーザは、更にツールを追加することによって、HEW の機能を拡張することができます。これを行うには、[ツールアドミニストレーション]ダイアログボックスでツールを登録しておくことが必要です。これらのツールを用いて、ウィンドウ、メニュー、およびツールバーを HEW システムに追加することができます。ツールの例としては、HEW のデバッガおよびビルダツールがあります。デバッガツールはデバッガに関連するすべてのメニューとツールバーを追加し、ビルダツールも同じようにビルド機能に関するすべてのメニューとツールバーを追加します。システムに登録したツールによって、HEW の使い方が変わります。そのため、このマニュアルに記載されたメニューのうち使用できないものもあります。例えば、デバッガツールがインストールされていない場合、HEW のメインウィンドウには[デバッグ]ツールがありません。

2. ビルドの基本

この章では HEW の一般的な機能を説明します。より高度な機能については 3 章 「ビルドの応用」を参照してください。

2.1 ビルド処理

ビルド処理の一般的な流れを図 2.1 に示します。HEW のインストール時に提供されるツールによってビルド処理は変わるので、図 2.1 の例とは少し異なるかもしれません（例えば、コンパイラが無いなど）。ビルドの各ステップまたはフェーズにおいて、1 セットのプロジェクトファイルについてビルド処理を行います。それが完了すると、次のステップまたはフェーズに移ります。

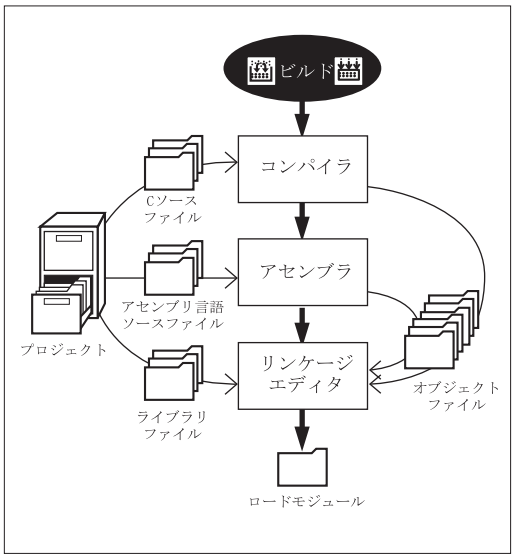


図 2.1: ビルド処理の一般的な流れ

図 2.1 に示す例では、第一のフェーズがコンパイラ、第二のフェーズがアセンブラ、そして最後のフェーズがリンケージエディタです。コンパイラのフェーズでは、プロジェクトの C/C++ソースファイルを順次コンパイルします。アセンブラのフェーズでは、アセンブリ言語のソースファイルを順次アセンブルします。リンケージエディタのフェーズでは、すべてのライブラリファイルと、コンパイラフェーズとアセンブラフェーズからの出力ファイルをリンクして、ロードモジュールを作成します。このモジュールは、HEW のデバッガ機能でダウンロードし、使用します。

ビルド処理をカスタマイズする方法はいくつかあります。例えば、独自のフェーズを追加したり、あるフェーズを無効にしたり、フェーズを削除できます。これらのビルド実行の応用については、3 章「ビルドの応用」を参照してください。

この章では、ビルドの一般的な原則や基本機能を説明します。

2.2 プロジェクトファイル

HEW を使ってアプリケーションをビルド処理するには、まず、どのファイルをプロジェクトに追加して、各ファイルをどのようにビルド処理すべきかを指定しなければなりません（図 2.2）。

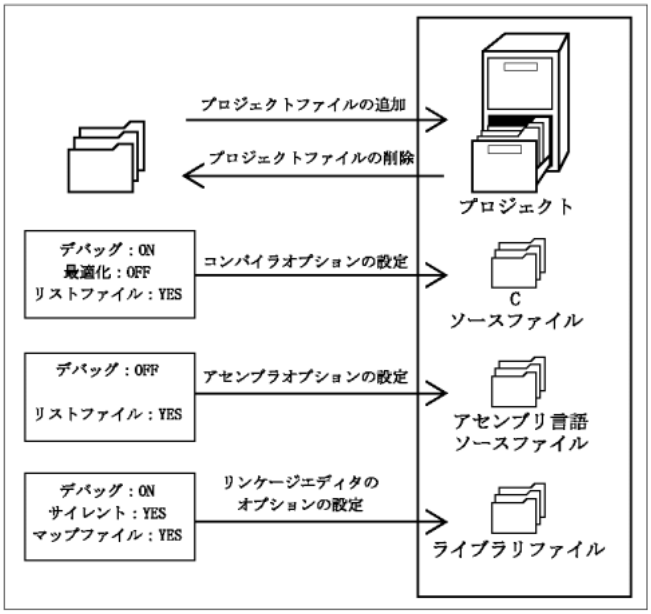


図 2.2: **プロジェクトの編集**

2.2.1 プロジェクトにファイルを追加する

アプリケーションをビルド実行する前に、まず、アプリケーションを構成するファイルを指定しなければなりません。

➲ プロジェクトにファイルを追加するには

1. **[プロジェクト->ファイルの追加...]**を選ぶか、ワークスペースウィンドウの[Projects]タブのプロジェクトアイコン上で右マウスボタンをクリックし、ポップアップメニューから**[ファイルの追加…]** を選ぶ（図2.3参照）か、ワークスペースウィンドウを選んで**INS**キーを押下してください。

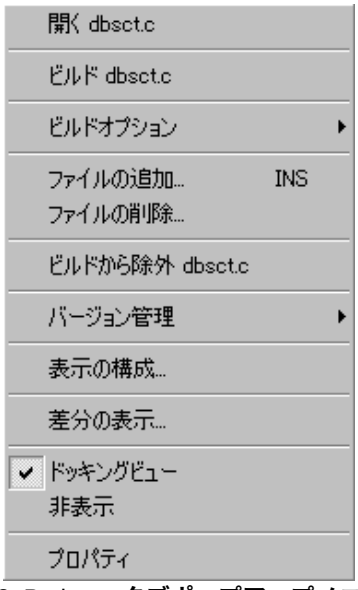


図 2.3: Projects **タブポップアップメニュー**

2. [ファイルの追加]ダイアログボックスが表示されます。
3. 追加するファイルを選び[追加]ボタンをクリックしてください。

プロジェクトに新しいファイルを追加するには、いくつか方法があります。これを以下で説明します。

エディタウィンドウ内で開いたファイルを右クリックすると、エディタウィンドウにポップアップメニューオプションが表示されます（図 2.4）。ファイルがすでにプロジェクト内にある場合、[プロジェクトにファイルの追加]メニューオプションは使用できません。[プロジェクトにファイルの追加]を選択すると、現在のプロジェクトにファイルを追加できます。

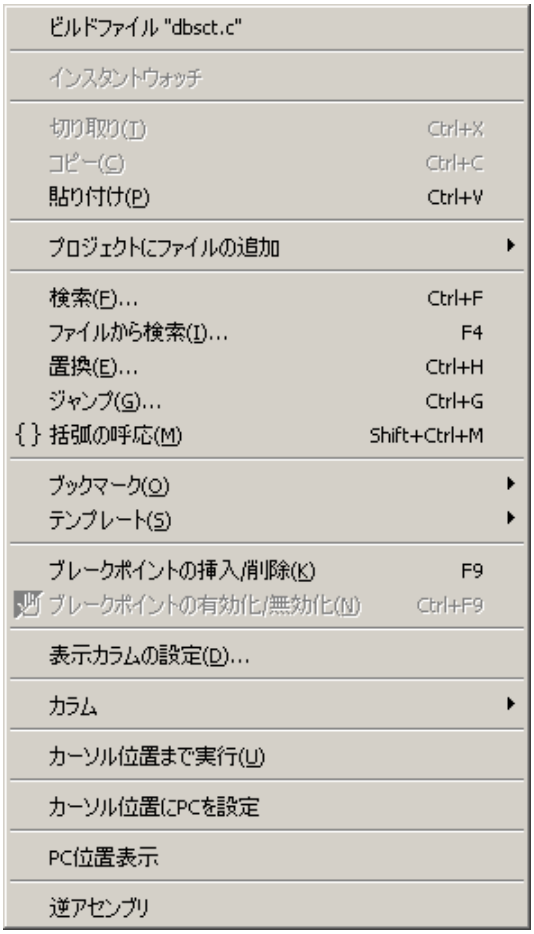


図 2.4: **エディタウィンドウポップアップメニュー**

HEW では、Windows® Explorer からワークスペースウィンドウにファイルを“ ドラッグしてドロップ ”することができます。こうしたファイルは自動的にプロジェクトに追加され、ドラッグされた先のフォルダに表示されます。

注意　プロジェクトに追加するファイルが、HEW の認識できない形式のファイルであっても、プロジェクトに追加されます。このファイルに関して、いくつかの機能が使用できなくなります。エディタ内でファイルを開く代わりにワークスペースウィンドウ内でこのファイルをダブルクリックすると、ファイルを開く動作が Windows オペレーティングシステムに受け渡されます。ファイルを Windows® Explorer 内で開いたかのように、ファイルを開くデフォルト動作が実行されます。現在、定義されている拡張子については、[ファイル拡張子]ダイアログボックスをご覧ください。（この章の後半にある「ファイルの拡張子とファイルグループ」を参照してください。）

2.2.2 プロジェクトからファイルを削除する

プロジェクトからファイルを削除できます。ファイルの削除は、一つでも、複数でも、すべてのファイルをまとめてでもできます。

➲ プロジェクトからファイルを削除するには

1. [プロジェクト->ファイルの削除...] を選択するか、ワークスペースウィンドウの[Projects]タブのプロジェクトアイコン上で右マウスボタンをクリックし、ポップアップメニューから [ファイルの削除…] を選択 (図2.5) してください。[プロジェクトファイルの削除]ダイアログボックスが表示されます（図2.6）。

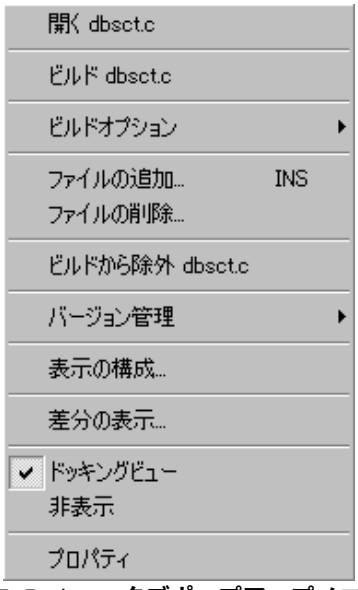


図 2.5: Projects **タブポップアップメニュー**

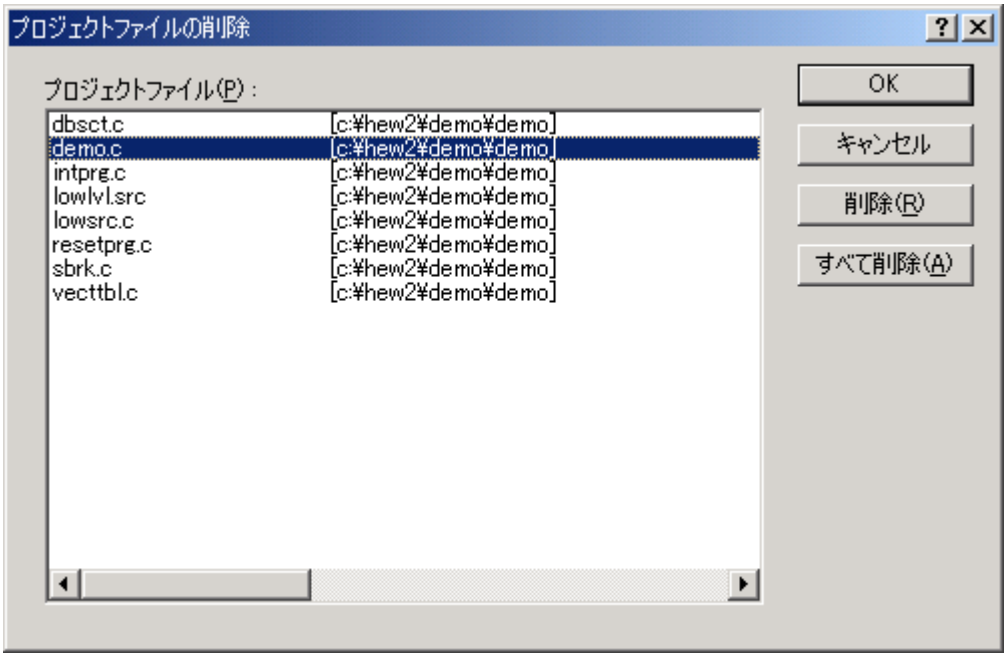


図 2.6: **プロジェクトファイルの削除 ダイアログボックス**

2. [プロジェクトファイル]リストから削除したいファイルを選んでください。複数のファイルを選ぶこともできます。
3. 選んだファイルを削除するには[削除]ボタンをクリックしてください。すべてのプロジェクトファイルを削除するには[すべて削除]ボタンをクリックしてください。
4. [OK]ボタンをクリックするとプロジェクトからファイルを削除します。

➲ 選んだファイルをプロジェクトから削除するには

1. ワークスペースウィンドウの[Projects]タブで削除したいファイルを選んでください。複数のファイルを選ぶときは”SHIFT”キーまたは”CTRL”キーを押下してください。
2. “DEL”キーを押してください。選んだファイルが削除されます。

2.2.3 ビルドからプロジェクトファイルを除外する

プロジェクトのファイルは、個々にビルドから除外することができます。

➲ ビルドからプロジェクトのファイルを除外するには

1. ワークスペースウィンドウの[Projects]タブで、ビルドから除外したいファイルを右マウスボタンでクリックしてください。
2. ポップアップメニュー（図2.5）から[ビルドから除外 ファイル名]を選んでください。すると、ファイルのアイコンに赤いバツ印がつけられ、ビルドから除外されます。

2.2.4 ビルドへプロジェクトファイルを入れる

除外したプロジェクトのファイルは再びビルドに入れることができます。

➲ 除外したファイルをビルドに再び入れるには

1. ワークスペースウィンドウの[Projects] タブでファイルを右マウスボタンで選んでください。
2. ポップアップメニューから[ビルドから除外の解除 ファイル名]を選んでください。 赤いバツ印が外され、ファイルがビルド可能になります。

2.3 ファイル拡張子とファイルグループ

HEW は拡張子でファイルを識別します。拡張子は使用するツールによって定義されます。例えば、コンパイラを使用すると拡張子.c が[C source file]グループに入り、コンパイラのフェーズに入力されます (図 2.1、ビルド処理の一般例)。さらに、独自の拡張子を定義することもできます。例えば、プロジェクトでアセンブリ言語ソースファイルを使っている場合、デフォルトの拡張子が.src だとします。.src の代わりに違う拡張子（例: .asm）を使うとき、新しい拡張子を定義してそれを.src ファイルと同様に扱うように指定できます。

ファイルの拡張子は、[ファイル拡張子]ダイアログボックスで表示、変更できます (図 2.7)。このダイアログボックスを表示するには **[プロジェクト->ファイルの拡張子...]**を選んでください。このダイアログボックスには現在のワークスペースで定義されたすべての拡張子とファイルグループを表示します。

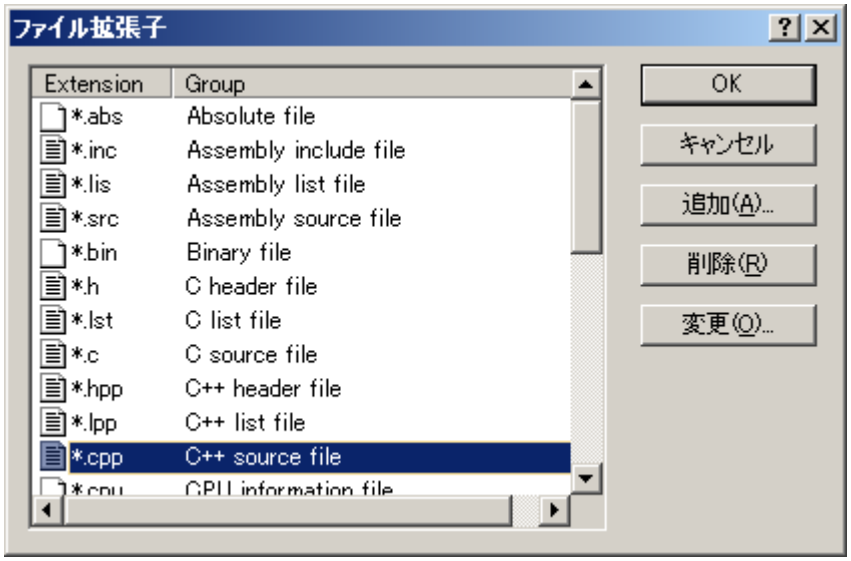


図 2.7: **ファイル拡張子 ダイアログボックス**

図 2.7 に示す[File Extensions]リストは 2 列に分かれています。左の列にはファイル拡張子、右の列にはファイルグループを表示します。図 2.8 に示すように、同じグループに多くのファイル拡張子が存在する場合があります。例えば、1 つのプロジェクト内でアセンブリ言語のソースファイルにいくつかの拡張子がある場合があります (例: .src, .asm, .mar など)。

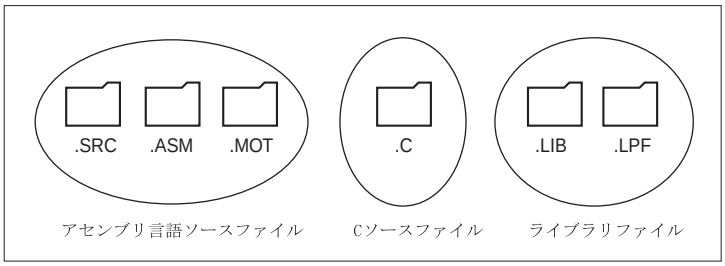


図 2.8:**ファイル拡張子とファイルグループ**

新しい拡張子を作成するときは、その拡張子がすでに定義されたファイルグループに属するのか、新しいファイルグループを作成する必要があるのか検討してください。新しい種類のファイルを追加するときには新しいファイルグループを作成してください。作成方法を以下に説明します。

➲ 新規ファイルグループに新規ファイル拡張子を作成するには

1. メニューバーから[プロジェクト->ファイルの拡張子...] を選んでください。[ファイル拡張子]ダイアログボックスが表示されます（図2.7）。
2. [追加…]ボタンをクリックしてください。[ファイル拡張子の追加]ダイアログボックスが表示されます（図2.9）。
3. [ファイル拡張子]フィールドに定義する拡張子を入力してください。ピリオド (.) の入力は不要です。ドロップリストには、現在のプロジェクトで未定義のすべての拡張子が含まれます。この拡張子のうち１つを選ぶと、ファイル拡張子フィールドにテキストが自動的に追加されます。
4. [新規拡張子グループ]オプションを選んで新しいファイルグループを定義する名前を入力してください。
5. この段階では、関連するアプリケーションの変更が可能です。“<拡張子グループ>を開くアプリケーション”のドロップダウンリストでは次の４つから選択が可能です。

   [Editor]
   [None]
   [Other]
   [Windows default]

   [Editor]を選択すると、ワークスペースウィンドウ内のファイルを開く機能によって、ファイルを HEW エディタ内で開くことができます。[None]を選択した場合、ファイルを開く機能を使用してもファイルを開くことはできません。[Other]を選択すると、ファイルを開くための他のツールを構築できます。詳細は、「アプリケーションとファイルグループを関連付けるには」を参照してください。[Windows default]を選択すると、ワークスペースウィンドウ内のファイルを開く機能によって、開いたファイルが Windows オペレーティングシステムに受け渡されます。また、Windows® Explorer で定義されたとして、このファイル拡張子のデフォルト動作を選択します。
6. [OK]ボタンをクリックすると[File Extension]リストに拡張子が追加されます。

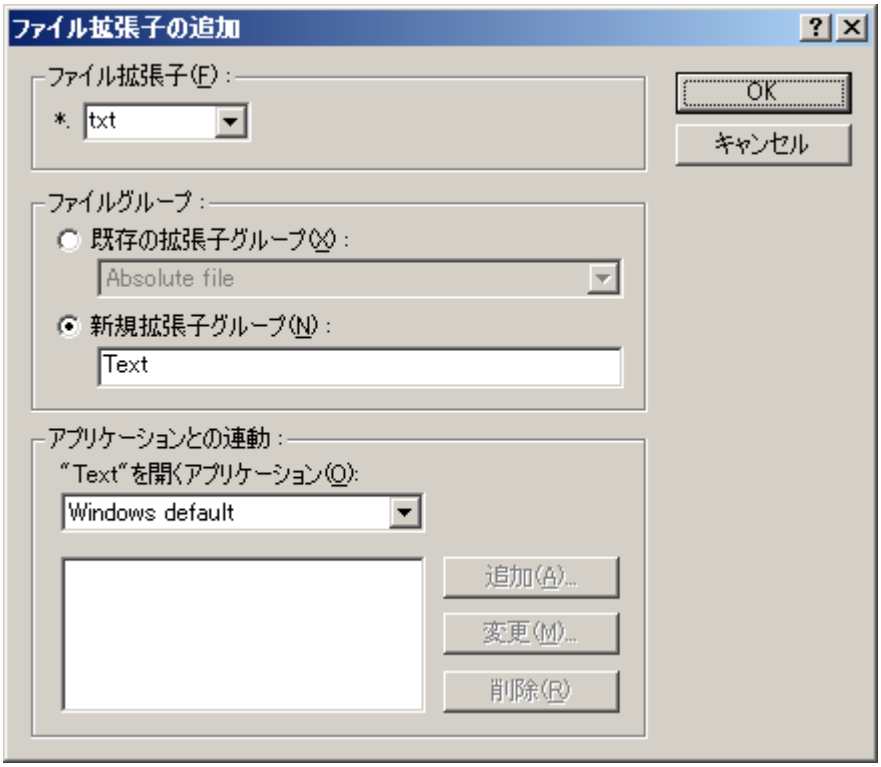


図 2.9: **ファイル拡張子の追加ダイアログボックス (新規グループ)**

HEW に登録されていない拡張子をもつプロジェクトを使っているときは、新しい拡張子を作成し、（例えば、フェーズ内のデフォルトの拡張子が.asm でビルダの認識する拡張子が.src のとき）新しい拡張子を既存のファイルグループに追加することが必要です。追加方法を下記に説明します。

➲ 新規ファイル拡張子を既存のファイルグループに追加するには

1. メニューバーから [プロジェクト->ファイルの拡張子...] を選んでください。[ファイル拡張子]ダイアログボックスが表示されます（図2.7）。
2. [追加…]ボタンをクリックすると [ファイル拡張子の追加]ダイアログボックスが表示されます(図2.10)。
3. [ファイル拡張子]フィールドに定義する拡張子を入力してください。 ピリオド (.) の入力は不要です。ドロップリストには、現在のプロジェクトで未定義のすべての拡張子が含まれます。この拡張子のうち１つを選ぶと、ファイル拡張子フィールドにテキストが自動的に追加されます。
4. [既存の拡張子グループ]オプションを選んでこの新しい拡張子をどのファイルグループに追加するか指定してください。
5. [OK]ボタンをクリックすると[File Extensions]リストに拡張子が追加されます。

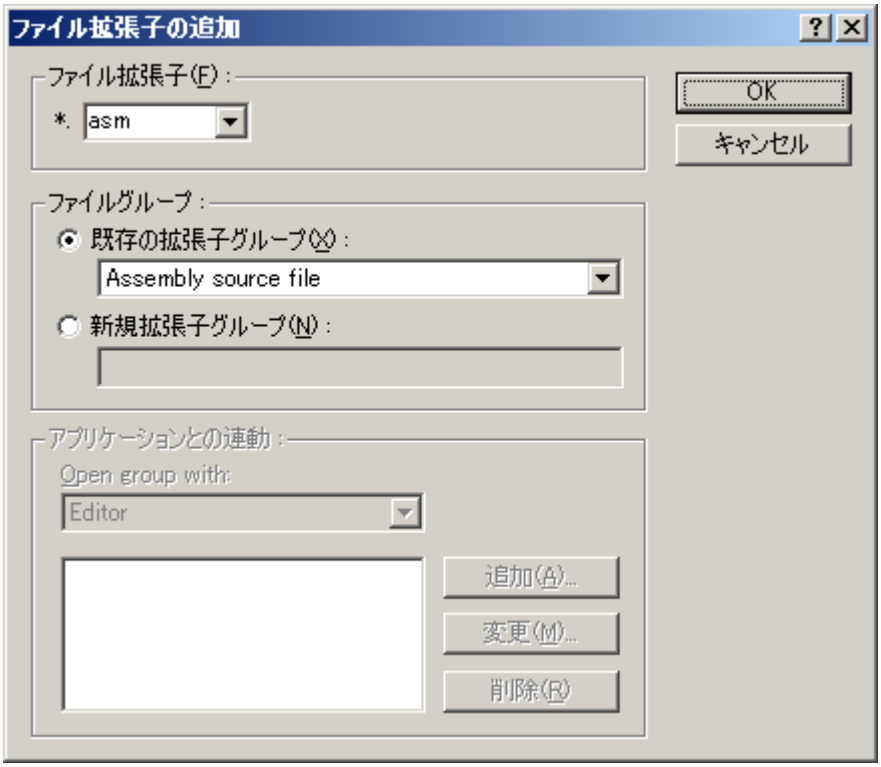


図 2.10: **ファイル拡張子の追加ダイアログボックス (既存グループ)**

– [ファイル拡張子]ダイアログボックスでは、エディタでファイルを開く指定だけでなく、ファイルグループとアプリケーションとの関連付けができます。これを行うと、ワークスペースウィンドウの[Projects]タブでファイルをダブルクリックすると、適切なアプリケー

ションでファイルを開きます。図 2.11 にワードプロセッサと拡張子.DOC の関連付けを示します。

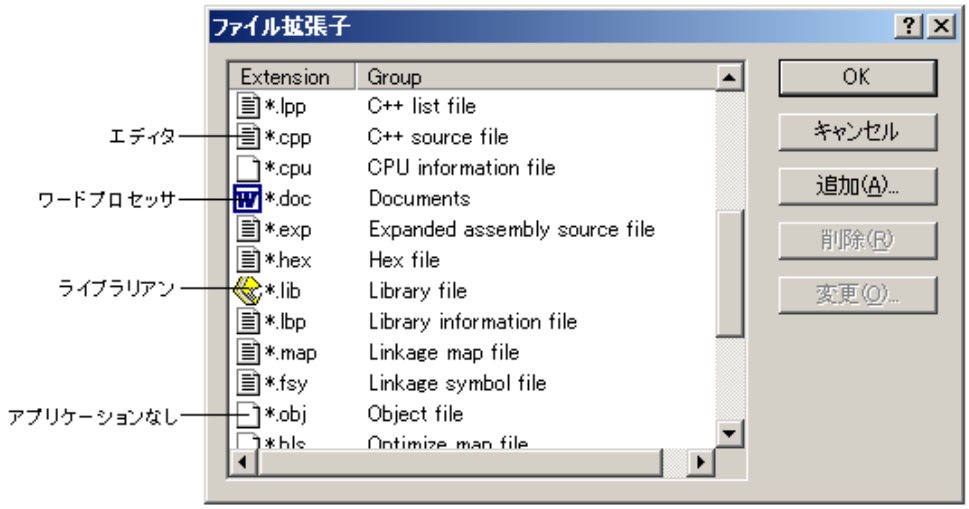


図 2.11:**ファイルグループとアプリケーション**

➲ アプリケーションとファイルグループを関連付けるには

1. [ファイル拡張子]ダイアログボックスで関連付けるファイルグループを選んでください（図2.11）。
2. [変更…]ボタンをクリックしてください。 [ファイル拡張子の変更]ダイアログボックスが表示されます（図2.12）。

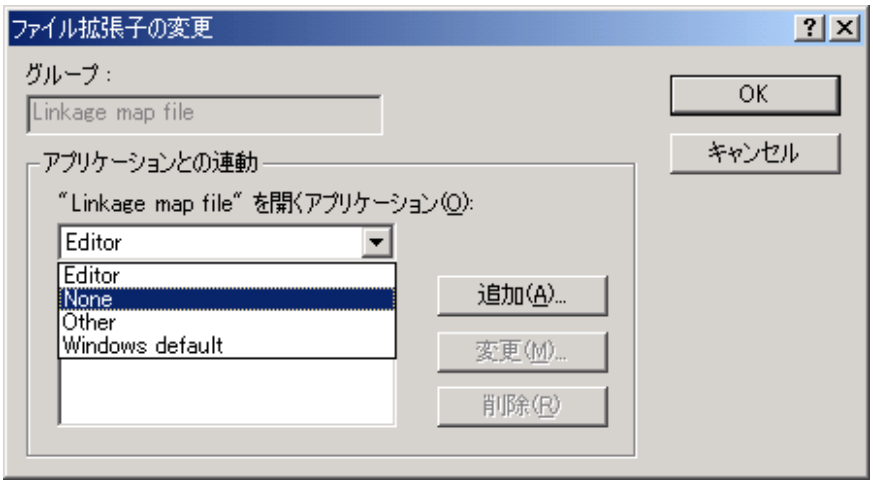


図 2.12: **ファイル拡張子の変更 ダイアログボックス**

3. 関連付けをしない場合は[None]を選びます。HEWのエディタまたは他のエディタでこの種類のファイルを開くには[Editor]を選びます。特定のアプリケーションでこの種類のファイルを開くには[Other]を選びます。

[Other]を選ぶと、すでに定義されたアプリケーションをドロップダウンリストから選ぶか、または、新しいアプリケーションを定義することができます。
新しいアプリケーションを定義する場合は[追加...]をクリックしてください。[アプリケーションの追加]ダイアログボックスが表示されます （図2.13）。
[名前]フィールドにアプリケーション名を入力します。[コマンド]フィールドにアプリケーションのフルパスを入力します(パラメータは含めません)。[パラメタ]フィールドにファイルを開くのに必要なパラメータを入力します。必ず$(FULLFILE) プレースホルダを使って入力ファイルを指定してください。 (プレースホルダの詳細と使用方法については、付録 C ，「プレースホルダ」を参照してください。) [初期ディレクトリ]フィールドにアプリケーションを実行させる初期ディレクトリを入力します。[OK]ボタンをクリックするとアプリケーションが定義されます。

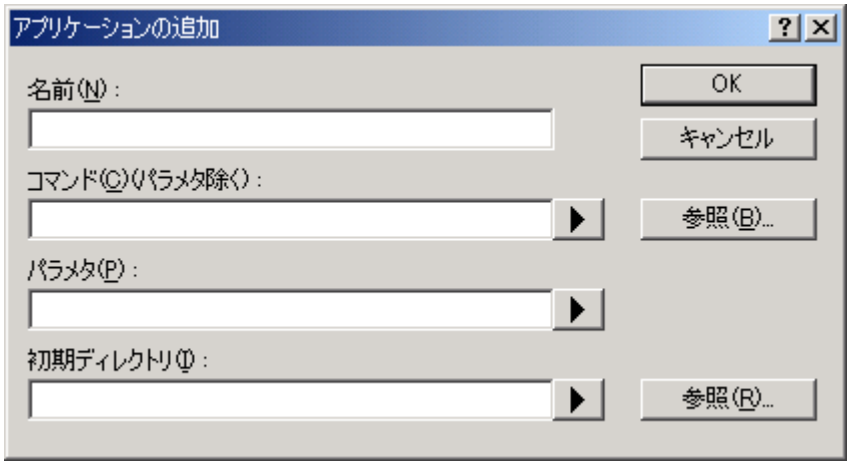


図 2.13: **アプリケーションの追加 ダイアログボックス**

アプリケーションを変更するには[変更...]ボタンを押してください。[アプリケーションの変更]ダイアログボックスが表示されます。このダイアログボックスは、[名前]フィールドが変更できないことを除いて[アプリケーションの追加]ダイアログボックスと同じです。設定を変更してから[OK]ボタンをクリックします。

4. [OK]ボタンをクリックすると選んだファイルグループに対するアプリケーションが関連付けられます。

2.4 ファイルのビルド方法を設定する

プロジェクトに必要なファイルを追加したら、次のステップは HEW に各ファイルのビルドを指示することです。このためには、[オプション]メニューからメニューオプションを選ばなければなりません。このメニューの内容は使用するツールによって異なります。

➲ ビルドフェーズにオプションを設定するには

1. オプションを変更するフェーズを[オプション]メニューから選んでください。
2. オプションを指定するダイアログボックスが表示されます。
3. オプションを設定して、[OK]ボタンをクリックしてください。

詳細は、コンテキスト依存のヘルプボタンをクリックするか、ヘルプを参照したい場所を選び"**F1**"キーを押下してください。

2.5 ビルドのコンフィグレーション

HEW では、ビルドのコンフィグレーションの中に、すべてのビルドのオプションを格納できます（図 2.14）。つまり、すべてのオプションを格納してそれらに名前を付けることができます。後にそのコンフィグレーションを選ぶと、各ビルドフェーズの各オプションを復帰させることができます。また、こうしたビルドのコンフィグレーションにより、ユーザはビルドのコンフィグレーション用にデバッガの設定を行うことができます。つまり、コンフィグレーションごとに異なるエンドプラットフォームをターゲットとすることができます。（詳細は、本マニュアルの「シミュレータ・デバッガ編」を参照してください。）

図 2.14 に"Default"、"MyDebug"、"MyOptimized"の 3 つのビルドコンフィグレーションを示します。"Default"ビルド構成では、各フェーズ（コンパイルとアセンブル）が標準設定されています。"MyDebug"ビルド構成では、各ファイルがデバッグ情報付きでビルドされています。"MyOptimized"ビルド構成では、各ファイルが最大限に最適化されデバッグ情報はありません。このプロジェクトの開発者は、オプションを設定するダイアログボックスに戻ってこれらを設定することなく、これらのビルドコンフィグレーションのうちどれでも選ぶことができます。

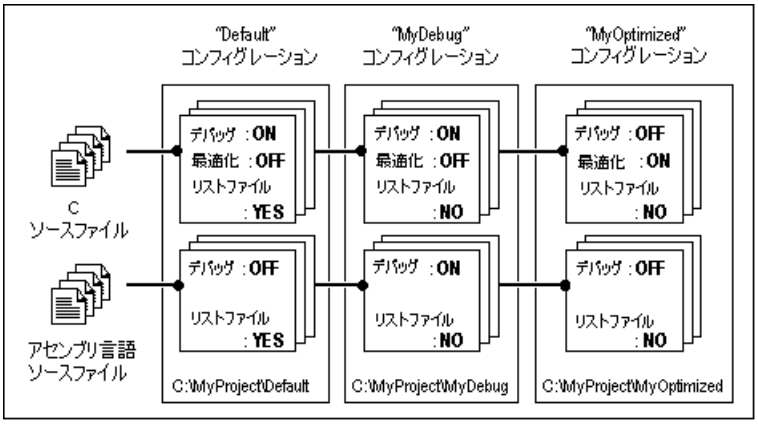


図 2.14: **ビルドコンフィグレーションとファイルオプション**

2.5.1 ビルドコンフィグレーションを選択する

使用するビルドコンフィグレーションを選択する方法は二つあります。

➲ ビルドコンフィグレーションを選択するには

1. ツールバーのドロップダウンリストボックス（図2.15）から選んでください。
2 ビルドコンフィグレーションが選ばれます。

図 2.15: **ツールバーの選択**

または

1. [オプション->ビルドの構成...]を選ぶと、[ビルドコンフィグレーション]ダイアログボックスが表示されます（図2.16）。

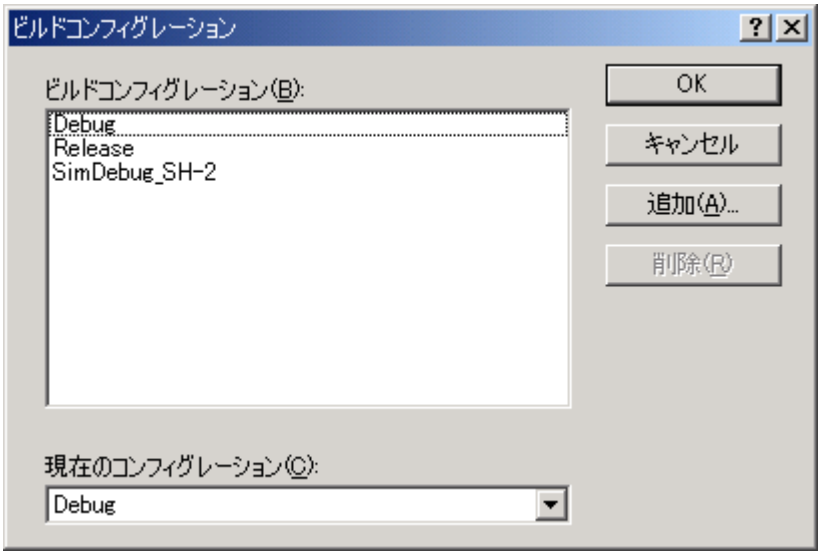


図 2.16: **ビルドコンフィグレーション ダイアログボックス**

2. [現在のコンフィグレーション]から使用するビルド構成を選んでください。
3. [OK]ボタンをクリックするとビルドコンフィグレーションが選ばれます。

2.5.2 ビルドコンフィグレーションを追加、削除する

ビルドコンフィグレーションの設定を他のビルドコンフィグレーションからコピーして追加したり、ビルドコンフィグレーションを削除したりできます。これらの操作を以下に説明します。

➲ ビルドコンフィグレーションを追加するには

1. [オプション->ビルドの構成...] を選ぶと[ビルドコンフィグレーション]ダイアログボックスが表示されます（図2.16）。
2. [追加...]ボタンをクリックすると[コンフィグレーションの追加]ダイアログボックスが表示されます （図2.17）。

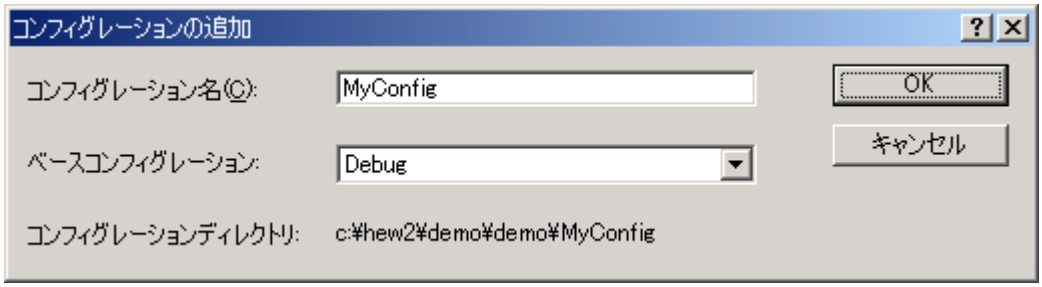


図 2.17: **コンフィグレーションの追加 ダイアログボックス**

3. [コンフィグレーション名]フィールドに新しいビルドコンフィグレーション名を入力してください。 入力すると、下に表示されるディレクトリがビルドコンフィグレーションに使われるディレクトリに変わります。[ベースコンフィグレーション]フィールドのドロップダウンリストにある既存コンフィグレーションの中から、コンフィグレーションの設定をコピーする元となるコンフィグレーションを選びます。両方のダイアログボックスの[OK]ボタンをクリックすると新しいビルドコンフィグレーションが作成されます。

➲ ビルドコンフィグレーションを削除するには

1. [オプション->ビルドの構成...] を選ぶと[ビルドコンフィグレーション]ダイアログボックスが表示されます（図2.16）。
2. 削除するビルドコンフィグレーションを選び[削除]ボタンをクリックしてください。
3. [OK]ボタンをクリックすると[ビルドコンフィグレーション]ダイアログボックスを閉じます。

2.6 プロジェクトをビルド実行する

ビルド実行の概要は図 2.1 を参照してください。

2.6.1 プロジェクトをビルド実行する

[ビルド]オプションでは前回のビルド実行後に変更のあったファイルだけをコンパイルまたはアセンブルします。さらに、前回のビルド実行以後に変更のあったファイルに依存するソースファイルを再ビルド実行します。例えば、 “test.c”にファイル “header.h”が含まれており“header.h”が前回のビルド実行以後に変更された場合、ファイル“test.c”が再コンパイルされます。

➲ ビルド実行するには

1. [ビルド->ビルド] を選ぶか、[ビルド]ツールバーボタン( ) をクリックするか、F7キーを押下してください。または、ワークスペースウィンドウの[Projects]タブのプロジェクトアイコン上で右マウスボタンをクリックし、ポップアップメニューから[ビルド]を選んでください。

[すべてをビルド]オプションでは変更の有無に関わらず、すべてのソースファイルをコンパイルまたはアセンブルして、新しく作成されたオブジェクトファイルをすべてリンクします。

➲ “すべてをビルド”処理を実行するには

1. [ビルド->すべてをビルド] を選ぶか、[すべてをビルド]ツールバーボタン ( )をクリックしてください。または、ワークスペースウィンドウの[Projects]タブのプロジェクトアイコン上で右マウスボタンをクリックし、ポップアップメニューの[ビルド]メニューからサブメニュー[すべてをビルド]を選んでください。

2.6.2 1 つのファイルをビルド実行する

プロジェクトにある 1 つのファイルをビルド実行できます。

➲ 1 つのファイルをビルド実行するには

1. プロジェクトウィンドウからビルド実行するファイルを選んでください。
2. **[ビルド->コンパイル]**を選ぶか、[ファイルのビルド]ツールバーボタン ( ) をクリックするか、**”CTRL+F7”**キーを押してください。または、ワークスペースウィンドウの[Projects]タブのファイルアイコン上で右マウスボタンをクリックして、ポップアップメニューから **[ビルド <ファイル名>]**を選んでください。

2.6.3 ビルド実行を中止する

ビルド実行を途中で中止できます。

➲ ビルド実行を中止するには

1. **[ビルド->ビルドの中止]** を選ぶか、“ビルド中止”ツールバーボタン ( )をクリックしてください。その時点のファイルのビルド実行を完了後、ビルド実行は中止されます。
2. アウトプットウィンドウに“Compiler Finished”というメッセージが表示されるのを確認してから操作を続けてください。

➲ ビルド実行中のツールを強制的に中止するには

1. [ビルド->ツールの終了]を選んでください。ビルダはツールの実行をすぐに中止します。

注意　中止したツールによって出力されたファイルは有効ではない場合があります。作成した出力ファイルをすべて削除して、そのフェーズを再実行してください。

2.6.4 複数のプロジェクトをビルド実行する

複数のプロジェクトやコンフィグレーションのビルド処理が行えます。

➲ 複数のプロジェクトをビルド実行するには

1. [ビルド->複数ビルド]を選択してください。図2.18に示します。
2. 複数のビルド実行では、どのプロジェクトまたはコンフィグレーションを処理するかを選択できます。どのプロジェクトまたはコンフィグレーションを実行するか選択するには、実行したいプロジェクトとコンフィグレーションの組み合わせの横にあるチェックボックスを選んでください。例えば、プロジェクト“hewtest2”全体にビルド実行したい場合、“hewtest2-Debug”、“hewtest2-Release”を選んでチェックし、その他のボックスのチェックを外してください。
3. 項目を選んだら、[ビルド]ボタンをクリックしてください。選んだプロジェクトやコンフィグレーションのビルド実行をHEWが行います。
4. また、選んだ項目に対して、全ビルド実行したい場合、[すべてをビルド]ボタンをクリックしてください。
5. 通常のビルド実行と同じ方法で、ビルドの結果がビルドウィンドウに表示されます。

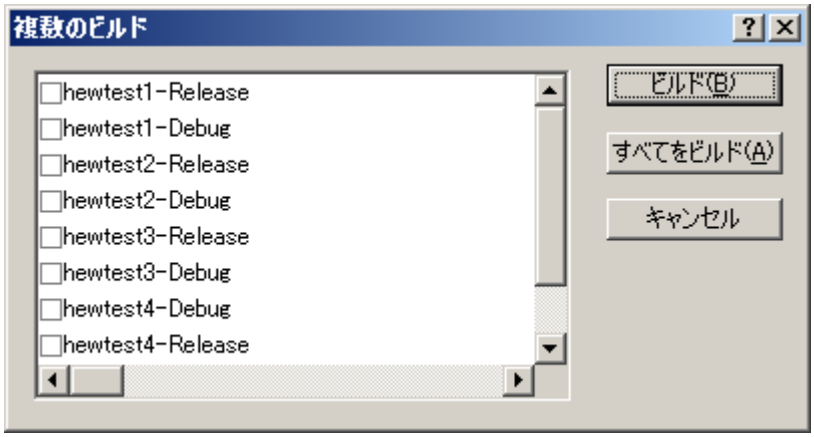


図 2.18　複数のビルドダイアログボックス

2.6.5 アウトプットウィンドウ

ツール（コンパイラ、アセンブラ、リンケージエディタなど）が実行されると、その出力がアウトプットウィンドウに表示されます。エラーまたはウォーニングが起きると、エラーメッセージまたはウォーニングメッセージと、ソースファイル名と行番号が表示されます。すぐにエラーまたはウォーニングが発生したところをエディタで見るには、表示されたエラーメッセージまたはウォーニングメッセージをダブルクリックします。

2.6.6 アウトプットウィンドウの内容の制御

ビルド実行の途中にビルド実行情報（ファイルに適用したコマンドラインオプションなど）を表示することができます。HEW では、“ビルド”、 “すべてをビルド”、 “ファイルのビルド”処理中、アウトプットウィンドウにそのオプションを表示するかどうかを[オプション]ダイアログボックスで指定できます。

➲ ビルド実行中にビルド実行情報の表示の有無を指定するには

1. **[ツール->オプション...]**を選ぶと[オプション]ダイアログボックスが表示されます。
2. [ビルド]タブ（図2.19）を選んでください。
3. [表示]グループのチェックボックスを以下のように設定します。 [コマンドライン]にはツール実行時のコマンドライン表示の有無を指定します。[環境]にはツール実行時の環境変数の表示の有無を指定します。[初期ディレクトリ]にはツールが起動されるディレクトリパスの表示の有無を指定します。

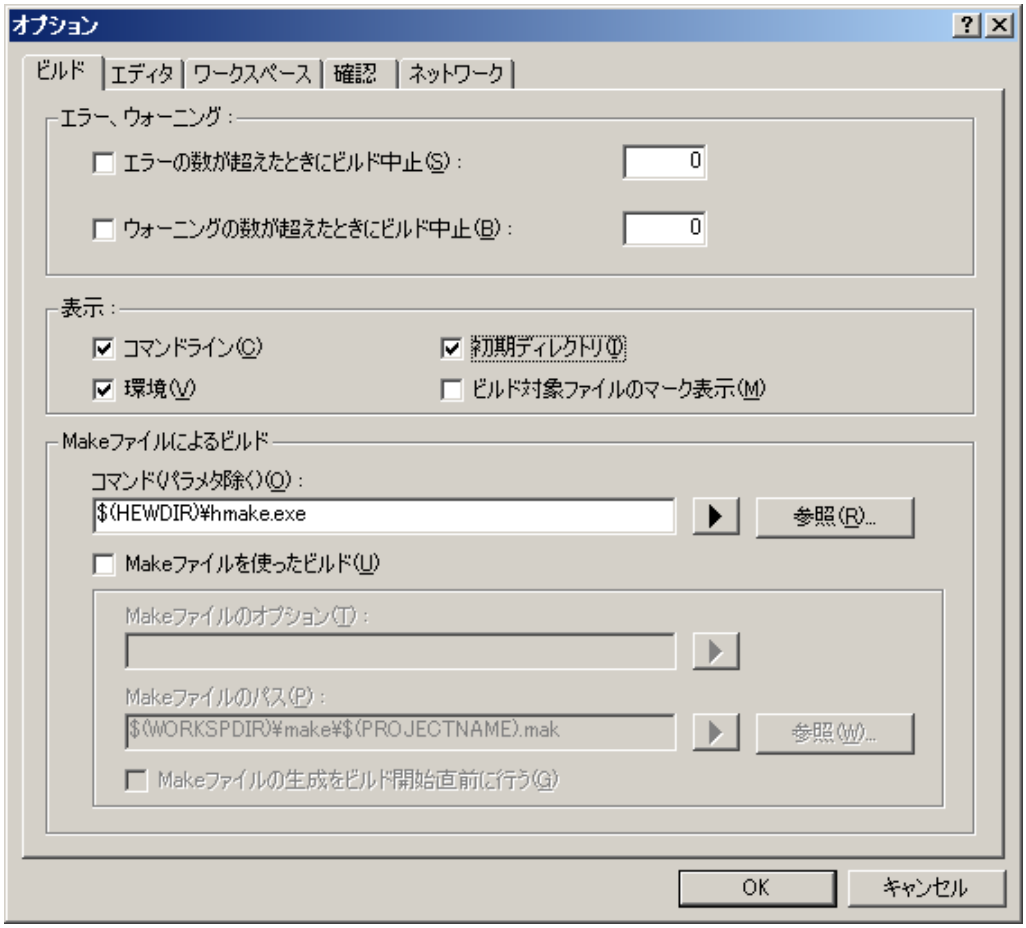


図 2.19: **オプションダイアログボックスビルドタブ**

2.6.7 ビルド対象ファイルをワークスペースウィンドウにマーク表示する

前回のビルドで生成されたファイルより後に更新されたファイル（ビルド対象ファイル）をワークスペースウィンドウ上にマーク表示します。図 2.20 では、ビルド対象ファイルは"demo.c"です。

次に[ビルド]をクリックしたとき、このファイルがビルドされます。アクティブプロジェクトの依存プロジェクトのファイルにもマーク表示します。

これらファイルのマーク表示は、ビルドに影響することが起こるたびに更新します。例えば、オプションの変更、ファイルの追加、依存関係の変更、ファイルの修正などです。

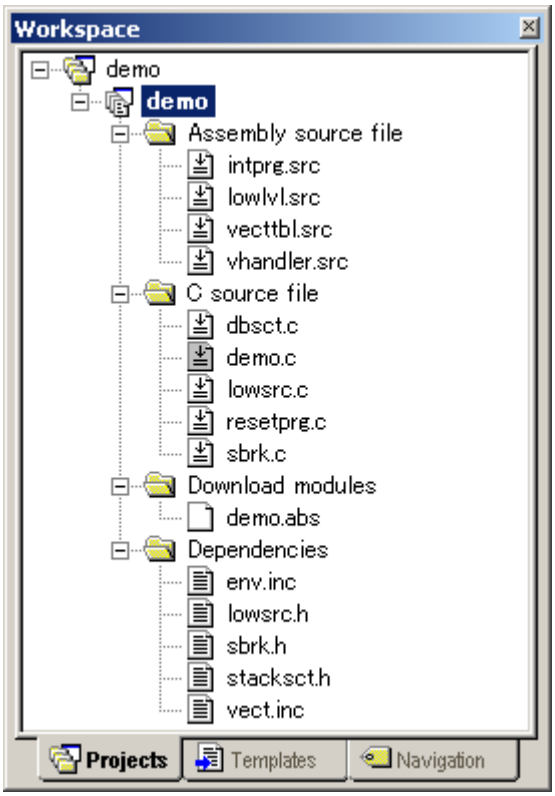


図 2.20: **ビルド対象ファイルのマーク表示**

➲ ビルド対象ファイルをワークスペースウィンドウにマーク表示するには

1. **[ツール->オプション...]**を選ぶと[オプション]ダイアログボックスが表示されます。
2. [ビルド]タブ（図2.19）を選んでください。
3. [ビルド対象ファイルのマーク表示]チェックボックスをチェックしてください。
4. [OK]ボタンをクリックしてください。

2.7 ファイル依存関係

多くの場合プロジェクトにはファイル間の依存関係があります。例えば、1 つの C ファイルはいくつかのヘッダファイルをインクルードします。複雑なプロジェクトでは、ソースファイルが他のインクルードファイルに依存するため、管理が複雑になります。しかし、HEW にはファイル依存関係をスキャンする機能があり、そのプロジェクトにあるすべてのファイルの依存関係をチェックできます。

スキャンが完了すると、プロジェクトのファイル依存関係を示す最新のリストをプロジェクトウィンドウに表示します。

➲ プロジェクトのファイル依存関係を更新するには

1. **[ビルド->すべての依存関係を更新]**を選んでください。または、ワークスペースウィンドウの[Projects]タブのプロジェクトアイコンを右マウスボタンでクリックし、ポップアップメニューの**[ビルド]**から**[すべての依存関係を更新]**サブメニューを選んでください。

最初に、すべての依存ファイルは[Dependencies]フォルダに表示されています（図 2.21.(i)）。

2.8 ワークスペースウィンドウの構成

ワークスペースウィンドウの[Projects]タブの中を右マウスでクリックすると、ポップアップメニューが表示されます。その中から、[表示の構成...]メニューオプションを選び、以下の情報の表示方法を設定してください。 以下に、[表示の構成]ダイアログボックスの各オプションについて説明します。

2.8.1 各ファイルの下に依存を表示する

[依存関係をファイルの下に表示]を選ぶと、依存ファイルがそれをインクルードするソースファイルの下に平坦な構造で表示されます（ファイル自体がフォルダになります）。これを図 2.21.(ii)に示します。このオプションを選ばないと、別のフォルダにすべての依存ファイルを示します（図 2.21.(i)）。

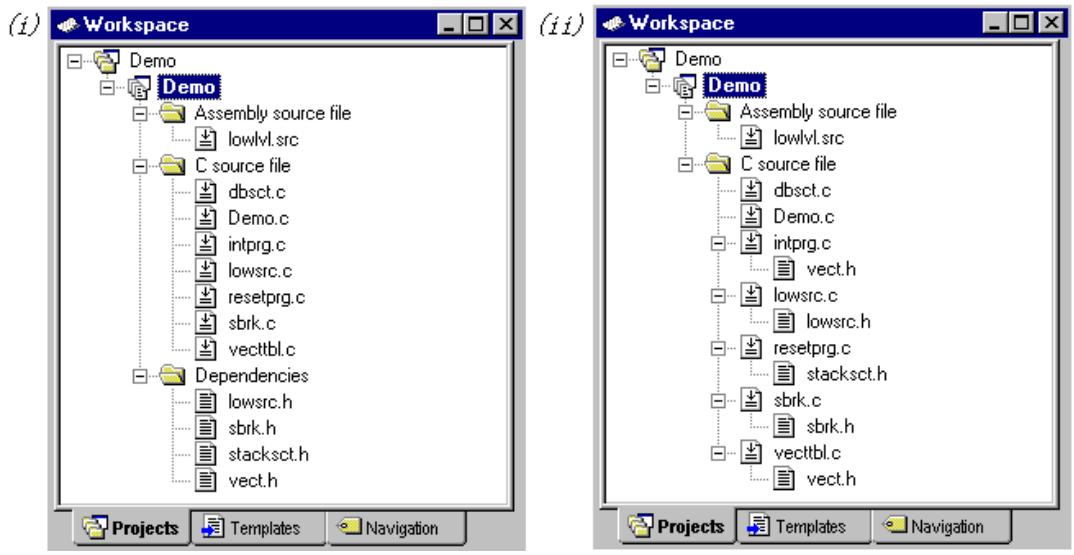


図 2.21: **各ファイルの依存**

2.8.2 標準ライブラリファイルのインクルードを表示する

デフォルトでは、標準インクルードパスの依存ファイルは表示されません（図 2.22.(i)）。例えば、C コードで#include <stdio.h>などのインクルード文を書くと、“stdio.h”は依存ファイルとして表示されません。そのようなシステムインクルードファイルを表示するには、[インクルードする標準ライブラリの表示]オプションを選んでください（図 2.22.(ii)）。

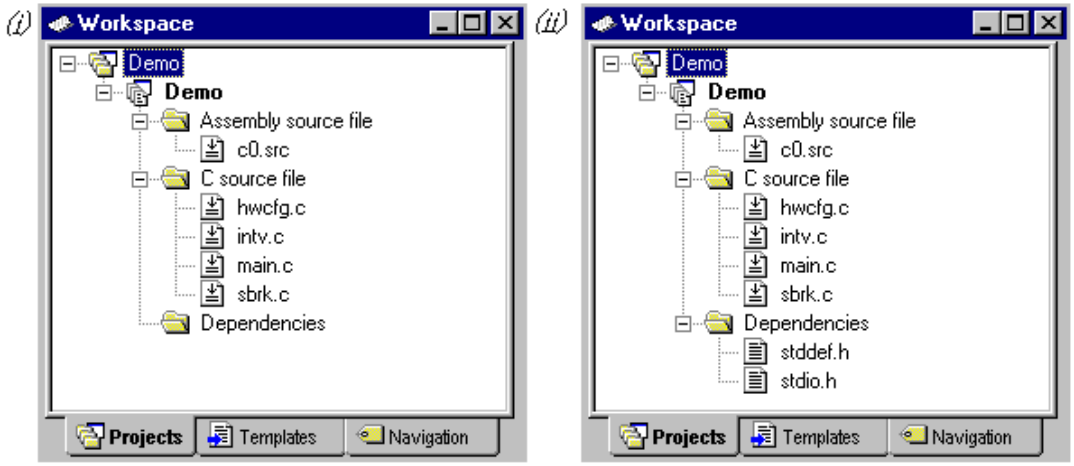


図 2.22: **標準ライブラリファイルのインクルード**

2.8.3 ファイルのパスを表示する

[ファイルパスの表示]を選ぶと、ワークスペースウィンドウのすべてのファイルがフルパス（ドライブ名からのパス）で表示されます （図 2.23）。

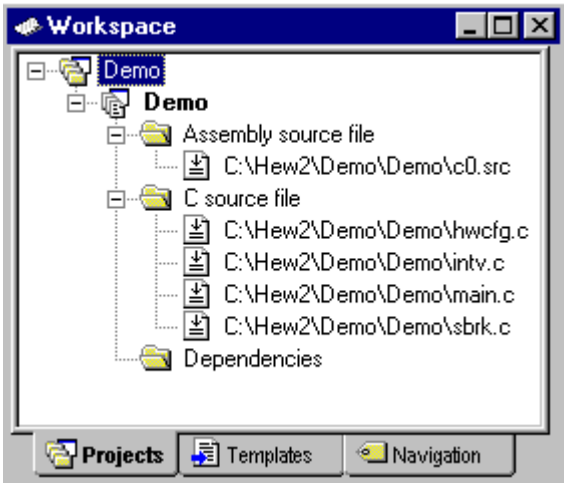


図 2.23: **ファイルのパスの表示**

2.8.4 ファイルをタイムスタンプの順序でソートする

[ファイルのタイムスタンプによるソート]を選ぶと、ワークスペースウィンドウのファイルがタイムスタンプの順序にソートして表示されます。

このオプションを設定した後でファイルを更新した場合は、手動で表示順を更新しなければいけません。

➲手動で表示順を更新するには

1. ワークスペースウィンドウの[Projects]タブのポップアップメニューから**[リフレッシュ]**を選んでください。

2.9 アクティブプロジェクトを設定する

ワークスペースには複数のプロジェクトを含めることができますが、ひとつだけがアクティブです。このアクティブプロジェクトでビルド動作とデバッグ動作が実行されます。そして、そのプロジェクト用のビルダやデバッガのオプションを変更できます。また、アクティブプロジェクトは太字で表示されます。

➲ プロジェクトをアクティブにするには

1. ワークスペースウィンドウの [Projects]タブからアクティブでないプロジェクトを選んでください。
2. マウスの右ボタンをクリックしてポップアップメニューを表示させ、**[アクティブプロジェクトに設定]** オプションを選んでください。

または

1. **[プロジェクト->アクティブプロジェクトに設定]** サブメニューからプロジェクトを選んでください。

2.10 ワークスペースにプロジェクトを追加する

ワークスペースを作成したとき、最初はプロジェクトが一つしかありません。しかし、後で、新しいプロジェクトや既存のプロジェクトを追加することができます。

➲ ワークスペースに新しいプロジェクトを追加するには

1. **[プロジェクト->プロジェクトの挿入...]**を選んでください。 [プロジェクトの挿入] ダイアログボックスが表示されます（図2.24）。
2. [新規プロジェクト] オプションを設定してください。
3. [OK]ボタンをクリックしてください。 [新規プロジェクトの挿入] ダイアログボックスが表示されます。
4. [名前] フィールドにプロジェクト名を入力してください。32文字以内で、半角英数字、半角下線が入力できます。プロジェクト名を入力すると、HEWは自動的にサブディレクトリを追加します。これは不要であれば削除できます。
5. [参照...]ボタンをクリックしてプロジェクトを作成するディレクトリを選んでください。または、 [ディレクトリ] フィールドにディレクトリを入力できます。
6. [プロジェクトタイプ] リストには使用可能なプロジェクトの種類を示します（アプリケーション、ライブラリなど）。このリストから作成するプロジェクトの種類を選んでください。
7. [OK]ボタンをクリックすると、プロジェクトが作成されワークスペースに追加されます。

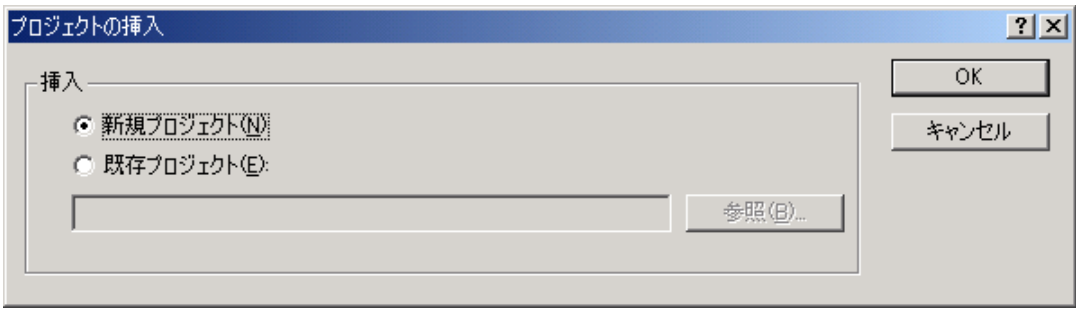


図 2.24: **プロジェクトの挿入ダイアログボックス**

➲ 既存のプロジェクトをワークスペースに追加するには

1. **[プロジェクト->プロジェクトの挿入...]**を選んでください。[プロジェクトの挿入] ダイアログボックスが表示されます。
2. [既存プロジェクト] オプションを設定してください。
3. プロジェクトデータベースファイル（.HWPファイル）へのフルパスを入力するか、 [参照...]ボタンをクリックしてプロジェクトデータベースファイルを指定してください。
4. [OK]ボタンをクリックするとそのプロジェクトがワークスペースに追加されます。

2.11 プロジェクト間の依存関係を指定する

ワークスペースのプロジェクトは、他のプロジェクトに依存することができます。ビルド処理をすると、依存プロジェクトが最初にビルドされます。これは、ワークスペースのプロジェクトを他のプロジェクトが使用しているときなどに使用します。例えば、ワークスペースに 2 つのプロジェクトがあるとします。1 つはアプリケーションプロジェクトに含まれたライブラリだとします。この場合 2 番目のアプリケーションのビルド前にライブラリは正確にビルドされ、また最新でなくてはなりません。そのため、ライブラリをアプリケーションプロジェクトの依存プロジェクトに指定します。こうすると、最新でないライブラリが先にビルドされます。

依存プロジェクトをビルドするとき、HEW は依存プロジェクトがアクティブプロジェクトのビルドコンフィグレーションになるようにします。上記の例では、アクティブプロジェクトのビルドコンフィグレーションが “Debug” であるとき、HEW は、依存プロジェクトで“Debug” ビルドコンフィグレーションが選択されるようにします。このような一致したコンフィグレーションが存在しない場合、HEW は依存プロジェクトで最近使われたコンフィグレーションを使用します。

➲ 依存プロジェクトを指定するには

1. **[プロジェクト->依存プロジェクト]**を選んでください。 [依存プロジェクト] ダイアログボックスが表示されます（図2.25）。
2. 依存させたいプロジェクトを選んでください。 [依存プロジェクト] リストに（選択したプロジェクト以外の）ワークスペース内のすべてのプロジェクトが表示されます。
3. [依存プロジェクト]リストには各プロジェクトにチェックボックスがあります。選んでプロジェクトが依存するプロジェクトのチェックボックスをチェックしてください。
4. [OK]ボタンをクリックしてください。

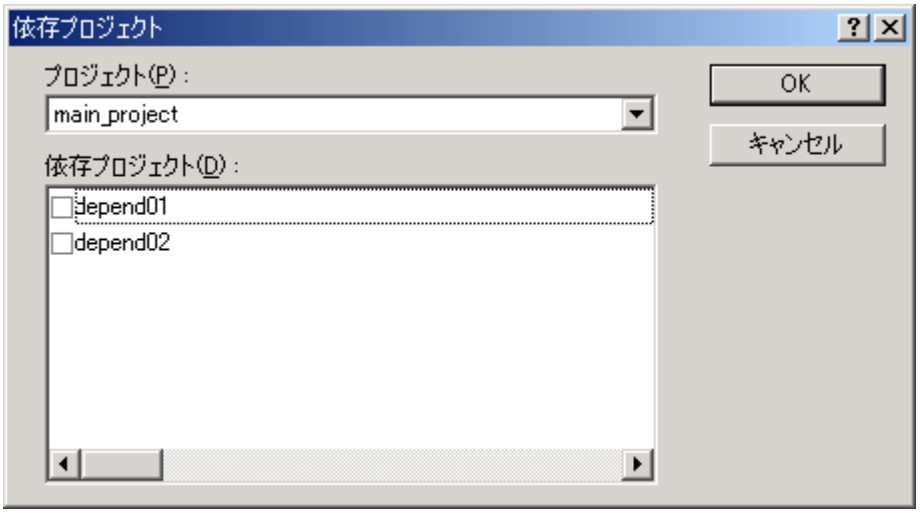


図 2.25: **依存プロジェクトダイアログボックス**

2.12 ワークスペースからプロジェクトを削除する

➲ ワークスペースからプロジェクトを削除するには

1. ワークスペースウィンドウの [Projects] タブを選び、右マウスボタンでポップアップメニューを表示してください。
2. **[プロジェクトの削除]** オプションを指定してください。
3. ダイアログボックスが表示され、この操作を誤って行っていないことを確認します。[ツール]メニューの[オプション]ダイアログボックスでは、この確認をオフにすることができます。

注意　ワークスペースからアクティブプロジェクトを削除することはできません。

2.13 ワークスペースのプロジェクトをロード、アンロードする

➲ ワークスペースのプロジェクトをロードするには

1. ワークスペースウィンドウの [Projects] タブから、アンロードしたプロジェクトを選んでください。
2. マウスの右ボタンをクリックしてポップアップメニューを表示させ、[プロジェクトのロード]オプションを選んでください。

➲ ワークスペースのプロジェクトをアンロードするには

1. ワークスペースウィンドウの [Projects] タブから、アクティブでないプロジェクトを選んでください。
2. マウスの右ボタンをクリックしてポップアップメニューを表示させ、[プロジェクトのアンロード]オプションを選んでください。

注意　一度に複数のプロジェクトを選び、それらをすべてロード、またはアンロードすることができます。

2.14 ワークスペースの相対プロジェクトパス

HEW では、プロジェクトをワークスペースに追加する場合、相対パスを使用してワークスペースに追加することが選択できます。これにより、プロジェクトをワークスペースディレクトリの上に置くことができ、HEW ワークスペースの再配置も正確にできます。プロジェクトは常にワークスペースと相対的であるため、プロジェクトがワークスペースの上のディレクトリにある場合、再配置のあと、HEW は、同じ相対場所でプロジェクトを見つけようとします。このことは、複数のワークスペース間で共有したプロジェクトを使用するとき、特に便利です。

HEW の古いバージョンでは、このプロジェクトは再配置されておらず、オリジナルのプロジェクトパスをアクセスしようとしました。また、ワークスペースディレクトリのサブディレクトリにあったプロジェクトを再配置することのみ可能でした。これは現在のバージョンでも、HEW の標準的な動作です。

➲ プロジェクト相対パスフラグを変更するには

1. ワークスペースウィンドウでプロジェクトを選択してください。
2. マウスの右ボタンをクリックしてプロパティを選択してください。
3. [プロジェクト相対パス]チェックボックスをクリックし、相対ファイルパスの特徴を切り替えてください（図2.26）。
4. [OK]ボタンをクリックしてください。

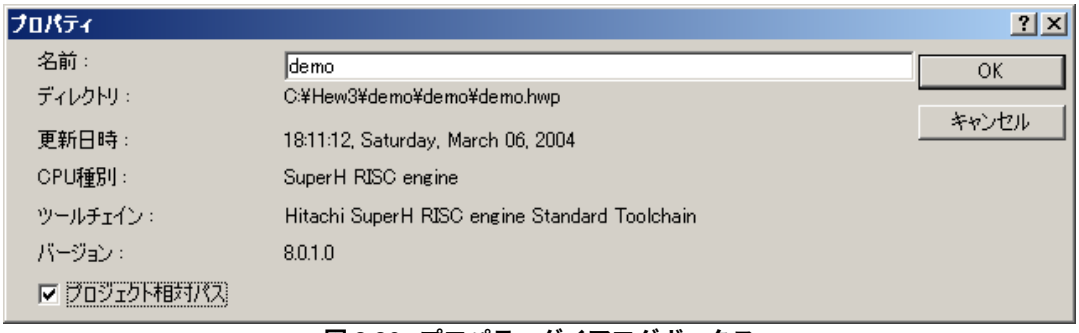


図 2.26: **プロパティダイアログボックス**

2.15 ワークスペース内のユーザフォルダ

HEW では、ユーザのワークスペースウィンドウにフォルダを追加することができます。これにより、プロジェクト内の特定の領域にファイルを論理的に分類することが可能です。フォルダ名として任意の名前をダイアログボックスで入力します。

➲ ユーザフォルダを追加するには

1. ワークスペースウィンドウの[Projects]タブでプロジェクトを選択してください。
2. マウスの右ボタンをクリックし、[フォルダの追加...]を選択してください。
3. 名前を入力して[OK]ボタンをクリックしてください。
4. ファイルをドラッグ＆ドロップして、論理的に分類することができます。

➲ ユーザフォルダを削除するには

1. ワークスペースウィンドウの[Projects]タブでフォルダを選択してください。
2. マウスの右ボタンをクリックし、[フォルダの削除]を選択してください。このときフォルダは空である必要があります。

➲ ユーザフォルダ名を変更するには

1. ワークスペースウィンドウの[Projects]タブで名前を変更したいフォルダを選択してください。
2. マウスの右ボタンをクリックし、[フォルダ名の変更]を選択してください。
3. ダイアログで新しい名前を入力してください。
4. [OK]ボタンをクリックしてください。

3. ビルドの応用

この章ではより高度なビルドの概念を説明します。

3.1 ビルド実行の復習

2 章「ビルドの基本」では、ビルド実行をコンパイラ、アセンブラ、リンケージエディタを用いて説明しました（図 2.1）。HEW ではこれが一般的な実行環境です。しかし、ビルド実行を変更（フェーズの追加や削除など）するためには、ビルドの機能についてさらに知ることが必要です。

3.1.1 ビルドとは?

プロジェクトのビルド実行とは、複数の特定の入力ファイルに複数のツールを適用して期待する出力を得ることです。つまり、オブジェクトファイルを得るために、C/C++ソースファイルにコンパイラを適用したりアセンブリ言語のソースファイルにアセンブラを適用したりします。ビルドの各ステップ、または、各「フェーズ」において、様々な入力ファイルの集まりに各種ツールを適用します。図 3.1 にビルド処理を示します。

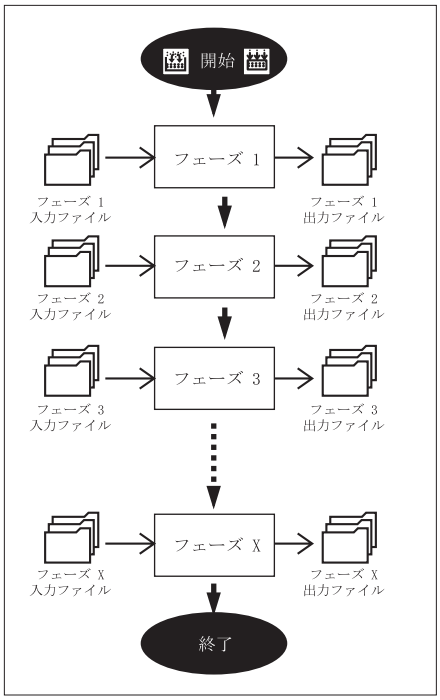


図 3.1: ビルド実行

HEW ではビルド処理を変更できます。 **[オプション->ビルドフェーズ...]**を選んで[ビルドフェーズ]ダイアログボックス （図 3.2） を表示します。左には現在のプロジェクトで定義されたフェーズを示します（図 3.2 では標準のビルドフェーズを示します）。この章では[ビルドフェーズ] ダイアログボックスが提供する様々な機能について説明します。

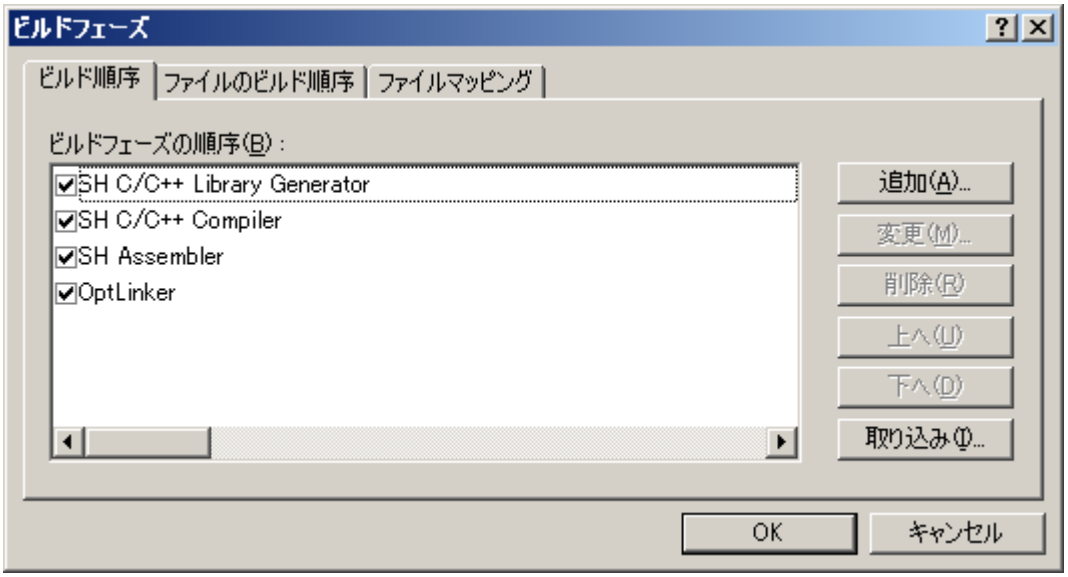


図 3.2: **ビルドフェーズ ダイアログボックス**

3.2 カスタムビルドフェーズを作成する

標準のビルド実行の前後または途中に他のツールを実行する場合、独自のビルドフェーズ（カスタムビルドフェーズ）を作成します。カスタムビルドフェーズを略してカスタムフェーズということがあります。

[オプション->ビルドフェーズ...] を選ぶと [ビルドフェーズ] ダイアログボックス （図 3.2） が表示されます。 [追加...] ボタンをクリックしてください。新しいビルドフェーズを作成するための[新規ビルドフェーズ]ダイアログボックス（図 3.3a）が表示されます。

ステップ 1（図 3.3a）では、カスタムビルドフェーズを新規に作成するか、システムビルドフェーズを追加するかを選択します。システムビルドフェーズは、使用しているツールチェイン（コンパイラ、アセンブラ、リンケージエディタ、ライブラリアンなど）内で定義済みのすぐに使用できるフェーズ、または、ユーティリティフェーズ（例えば、ファイルコピー、ソースコード複雑度解析ツールなど）です。システムフェーズがこれ以上ない場合、[既存のシステムフェーズの追加]ボタンが非アクティブになります。（システムビルドフェーズを略してシステムフェーズといいます。）

[新規カスタムフェーズの作成]を選んでカスタムビルドフェーズを作成してください。

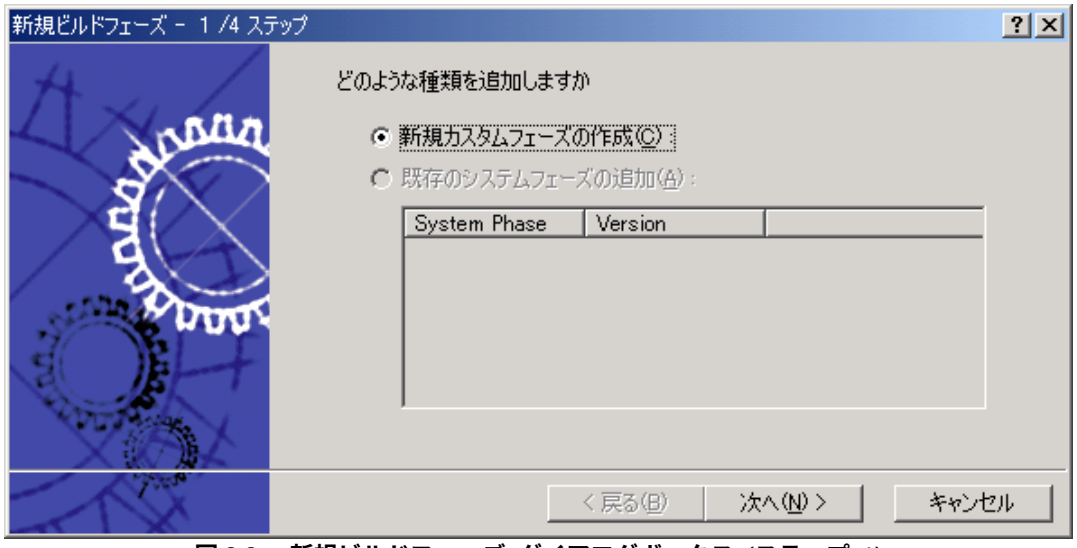


図 3.3a: **新規ビルドフェーズ ダイアログボックス (ステップ** 1)

ステップ 2（図 3.3b）では作成するフェーズの種類を選びます。2 つの選択肢（ [複数フェーズ]または [単一フェーズ]）があります。複数フェーズを実行すると特定のファイルグループに属するプロジェクト内の各ファイルにコマンドが適用されます。例えば、[入力ファイルの選択]フィールドに[C source file]を選ぶと、プロジェクト内の各ファイルに 1 回ずつコマンドが実行されます。単一フェーズを選ぶとビルド実行中に一度だけ実行されます。

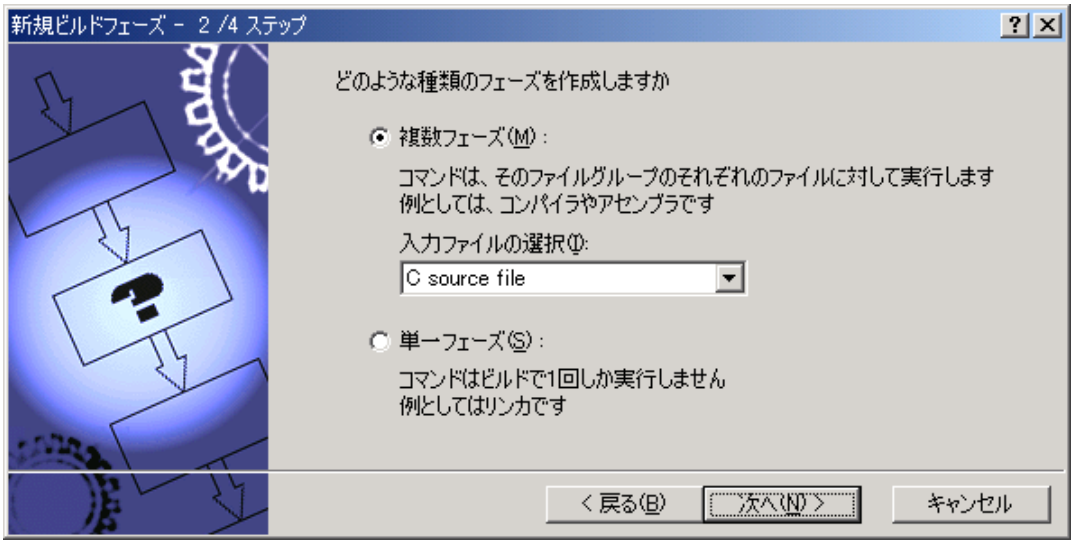


図 3.3b: **新規ビルドフェーズ ダイアログボックス（ステップ** 2)

入力ファイルグループリストは、そのプロジェクト用に定義された現在のファイルグループを含みます。入力ファイルグループリストの中の[Multiple Groups…]エントリを選択すると、複数の入力ファイルグループを定義することができます。このリストのエントリは、図 3.3c に示すダイアログボックスを表示します。

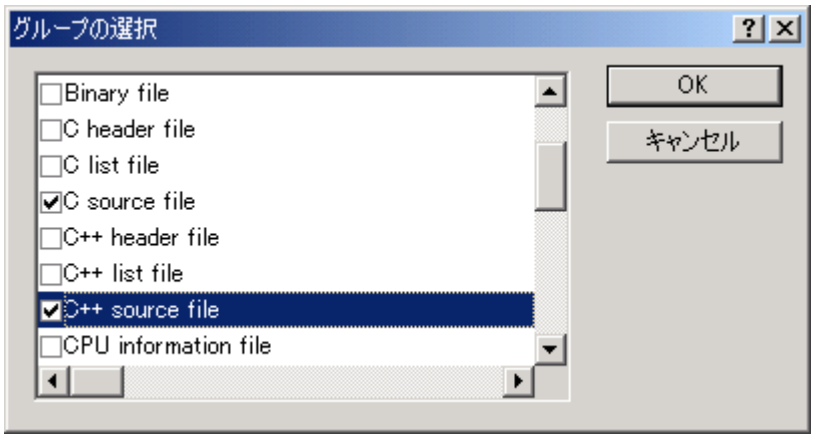


図 3.3c: **複数の入力ファイルグループの変更**

複数選択すると、入力ファイルグループは[Multiple Groups…]と表示されます。プロジェクトに追

加されているカスタムフェーズ用に、複数の入力ファイルグループをこのダイアログボックスで選ぶことができます。ファイルグループを選択するには、ファイルグループ名の隣にあるボックスをチェックしてください。このダイアログボックスでは、１つ以上のファイルグループが選択できます。

ステップ 3（図 3.3d）では、新しいビルドフェーズについての基礎的な情報を入力します。 [フェーズ名] フィールドにフェーズ名を入力します。 [コマンド] フィールドにプログラムファイルのパスを入力します （コマンドラインオプションは含めません。オプションは HEW のメニューバーの[オプション]メニューで指定します）。[デフォルトオプション]フィールドにフェーズのデフォルトのオプションを指定します。デフォルトオプションはプロジェクトに新しいファイルを追加するときに付加されるオプションです。[初期ディレクトリ] フィールドにはそのプログラムをどのディレクトリから実行するか（つまり、ツール実行前にどこにカレントディレクトリを設定するか）を入力します。

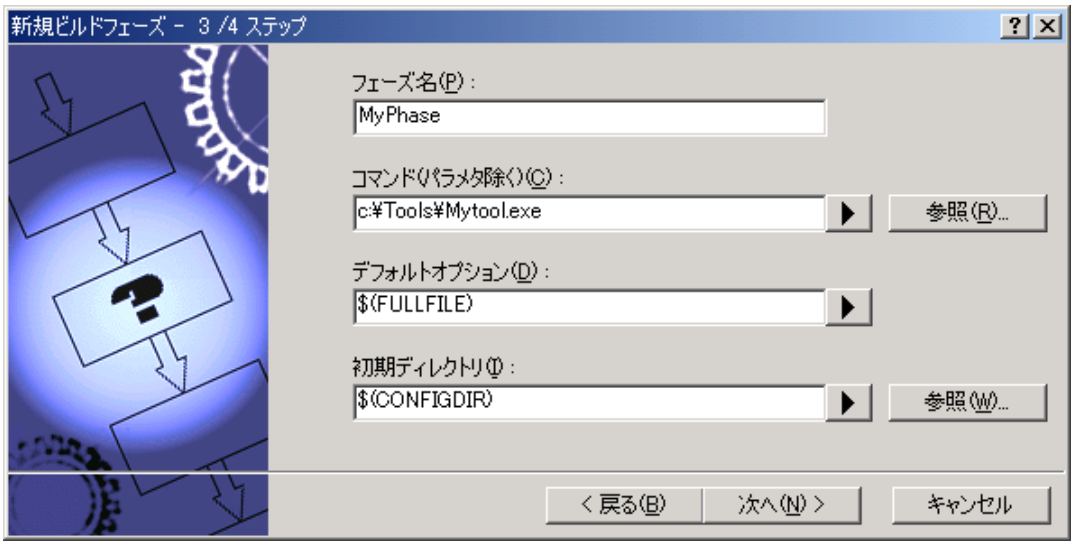


図 3.3d: **新規ビルドフェーズ ダイアログボックス（ステップ** 3)

最後のステップ 4（図 3.3e）では、そのフェーズに必要な環境変数を指定します。

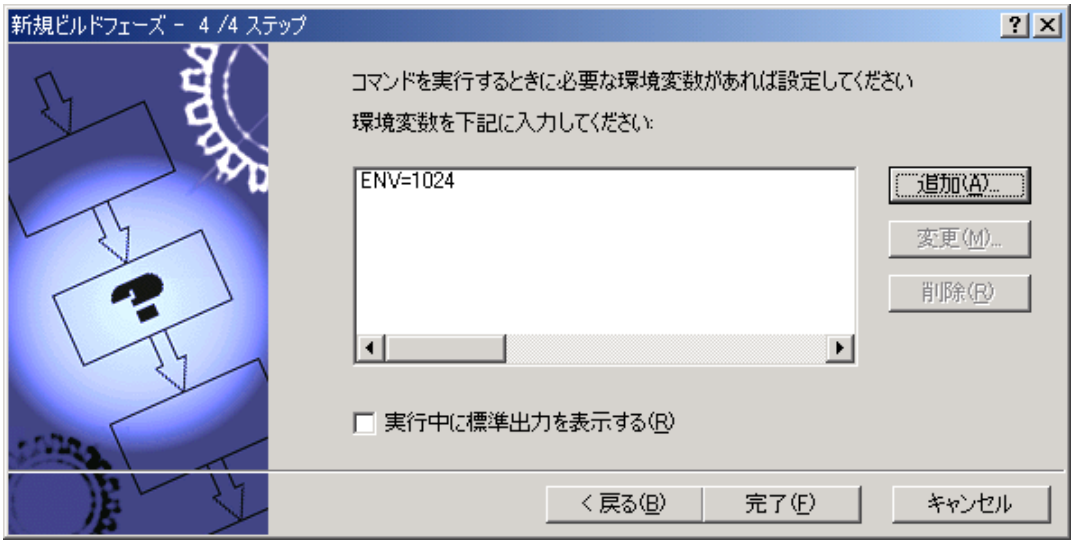


図 3.3e: **新規ビルドフェーズ ダイアログボックス (ステップ** 4)

新しい環境変数を追加するには、[追加...] ボタンをクリックしてください。図 3.4 のダイアログボックスが表示されます。新しい環境変数を追加するには[変数] フィールドに環境変数名を入力して [値] フィールドに環境変数の値を入力して [OK]ボタンをクリックします。

環境変数を変更するには、ステップ 4 のダイアログボックスのリストから環境変数を選んで、[変更...] ボタンをクリックします。 [変数] フィールドと [値] フィールドを変更して [OK]ボタンをクリックすると、リストに変更した変数が追加されます。

環境変数を削除するには、ステップ 4 のダイアログボックスのリストから削除する環境変数を選んで、[削除] ボタンをクリックします。

図 3.4: **環境変数 ダイアログボックス**

追加するツールが動作中に出力を表示できる場合、ステップ 4 のダイアログボックスの[実行中に標準出力を表示する]オプションを使用してください。出力が発生するごとにツールの出力を表示します。このオプションが off に設定されると、HEW はツールに表示されている全出力を保存し、ツールが動作を終了したときアウトプットウィンドウに表示します。ただ、ツールが長時間かかる作業を実行中である場合、実行の進行状況を見るのが難しいため、このオプションは問題となることがあります。

注意　[実行中に標準出力を表示する]を用いると、特定のオペレーティングシステムで特定のツールを使用するとき問題を引き起こすことがあります。もしツールが HEW の中でロックアップ、またはフリーズするといった問題がありましたら、[実行中に標準出力を表示する]オプションのチェックを外してください。

指定した内容で新しいフェーズを作るには、 [完了] ボタンをクリックしてください。デフォルトでは[ビルドフェーズ] ダイアログボックス（図 3.2）の[ビルド順序]タブの[ビルドフェーズの順序]リストの最後に新しいフェーズが追加されます。

3.3 ビルドのフェーズ順序

図 3.5 の標準的ビルド処理では、①コンパイラの前、②アセンブラの前、③リンケージエディタの前、④リンケージエディタの後、の四ヶ所にフェーズを追加できます。ビルドの順序の中で好きな場所にカスタムフェーズを追加したりシステムフェーズを移動したりできます。ビルド処理を正しく実行させるためには、カスタムフェーズの出力が他のフェーズに入力される場合、フェーズの順序を正しく設定することが必要です。

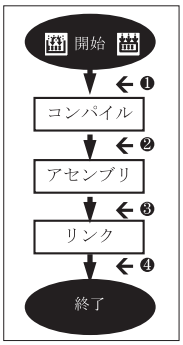


図 3.5: 標準的ビルド処理

[ビルドフェーズ] ダイアログボックスでビルドフェーズの順序を変更できます。このダイアログボックスにはフェーズの順序に関するタブが二つあります。 [ビルド順序] タブと[ファイルのビルド順序]タブです。変更後 [OK]ボタンをクリックしてください。

3.3.1 ビルドのフェーズ順序

[ビルド順序] タブ(図 3.6)は、[ビルド] ()または[すべてをビルド] () 操作で実行されるフェーズの現在の順序を示します。各フェーズの左にあるチェックボックスのチェックの有無はそのフェーズの有効/無効を示します。このチェックボックスをチェックするとそのフェーズが実行されます。

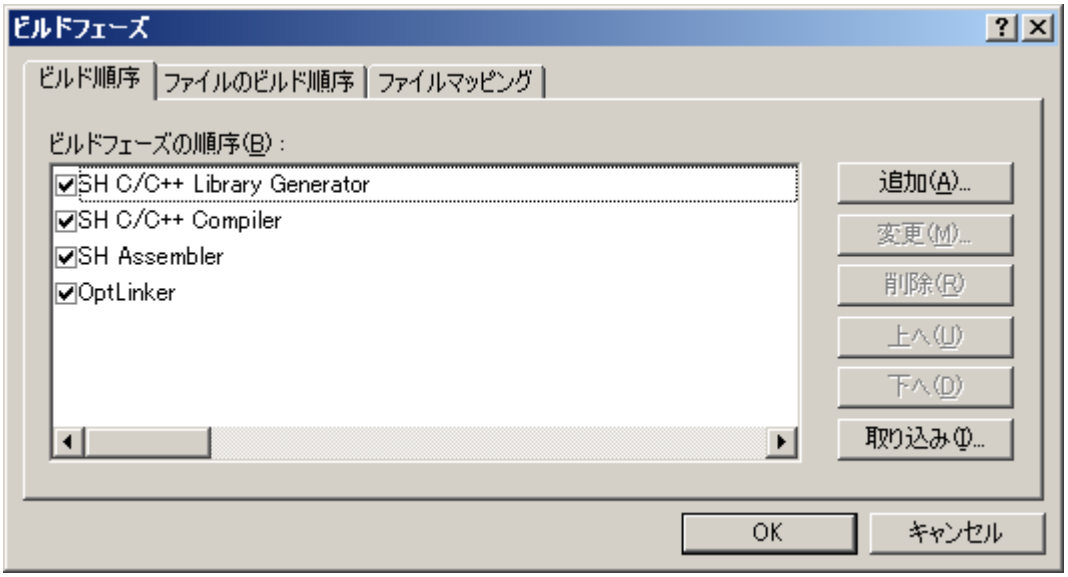


図 3.6: **ビルドフェーズ ダイアログボックス ビルド順序タブ**

また、以下の操作ができます。

➲ フェーズを削除するには

1. 削除するフェーズを選んでください。
2. [削除] ボタンをクリックしてください。
3. 確認ダイアログボックスが表示された場合は、 [はい] ボタンをクリックしてください。

➲ システムフェーズのプロパティを表示するには

1. プロパティを表示するシステムフェーズを選んでください。
2. [変更...]ボタンをクリックしてください。
3. ダイアログボックスで、システムフェーズのプロパティが表示されます。[OK]ボタンをクリックしてください。

➲ フェーズを移動するには

1. 移動するフェーズを選んでください。
2. [上へ] または [下へ] ボタンをクリックすると上下に移動します。

➲ フェーズを取り込むには

1. [取り込み] ボタンをクリックしてください。ダイアログボックスが表示され、カスタムフェーズを取り込むための既存のプロジェクトを見ることができます。
2. カスタムフェーズを取り込みたいプロジェクトの位置を選んでください。選択すると、取り

込み可能なプロジェクトのカスタムフェーズを並べたダイアログボックスが表示されます。

3 どのフェーズを取り込むか決定したら、そのフェーズをリスト上でハイライト表示し、[OK] ボタンをクリックしてください。ビルド順序で一番下にあるビルドフェーズダイアログボックスに、そのフェーズが追加されます。

➲ カスタムフェーズを変更するには

1. 変更するカスタムフェーズを選んでください。
2. [変更…] ボタンをクリックしてください。[MyPhaseの変更] ダイアログボックスの[コマンド] タブが表示されます（図3.7）。
3. 必要に応じてフィールドの内容を変更してください。
4. 入力ファイルがなくてもフェーズの実行を中断したくない場合、 [実行前に入力ファイルが存在するか否かのチェックを行わない] チェックボックスをチェックしてください。
   標準出力への出力をコマンド実行中に表示したい場合は、[実行中に標準出力を表示する] チェックボックスをチェックしてください。

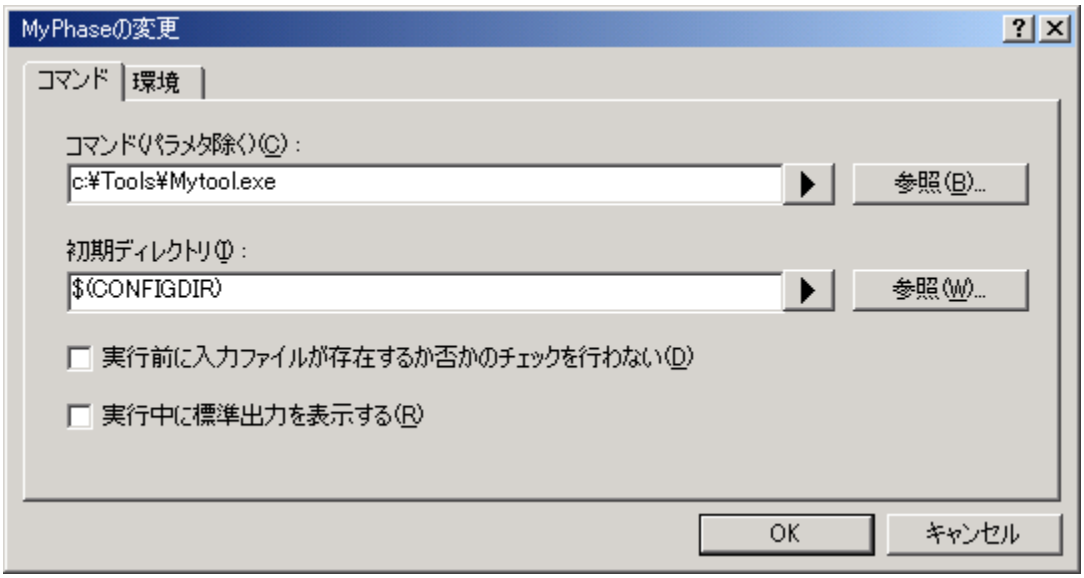


図 3.7: MyPhase **の変更 ダイアログボックスコマンド タブ**

5. [環境] タブ（図3.8）を選んでフェーズの環境設定を行ってください。
6. 環境変数の追加は [追加…] ボタン、変更は [変更…] ボタン、削除は [削除] ボタンを使用してください。操作は前節と同じです。
7. 変更後 [OK] ボタンをクリックしてください。

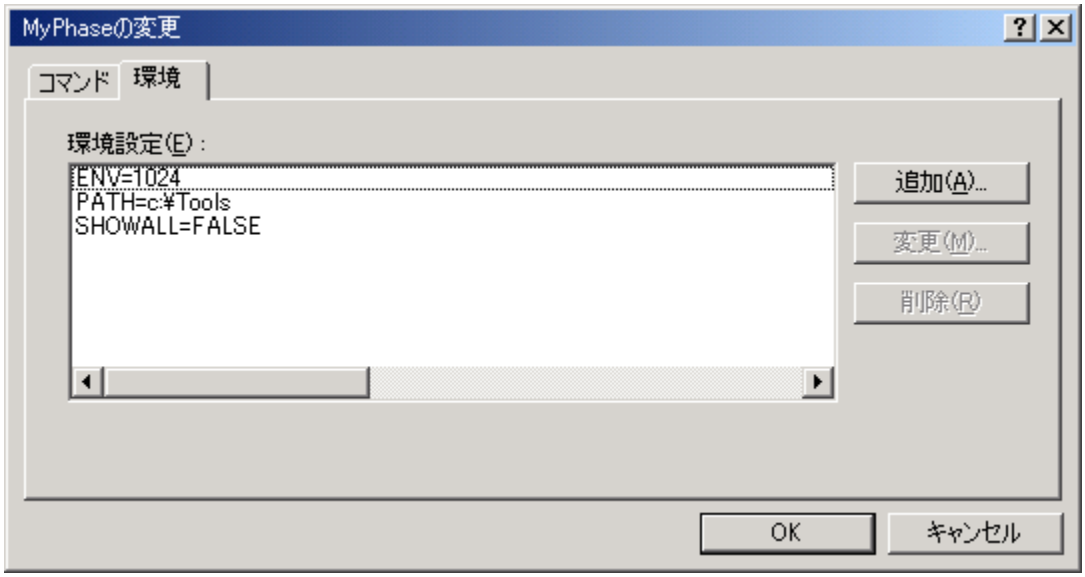


図 3.8: MyPhase **の変更 ダイアログボックス環境 タブ**

3.3.2 ビルドファイルのフェーズ順序

ワークスペースウィンドウから C ソースファイルを選んで**[ビルド->コンパイル]**を選ぶか、を押すと、ファイルがコンパイルされます。同じように、ワークスペースウィンドウからアセンブリ言語ソースファイルを選んで**[ビルド->コンパイル]**を実行すると、ファイルがアセンブルされます。ファイルグループと実行するフェーズの関係は[ビルドフェーズ]ダイアログボックスの [ファイルのビルド順序] タブ（図 3.9）で管理されています。

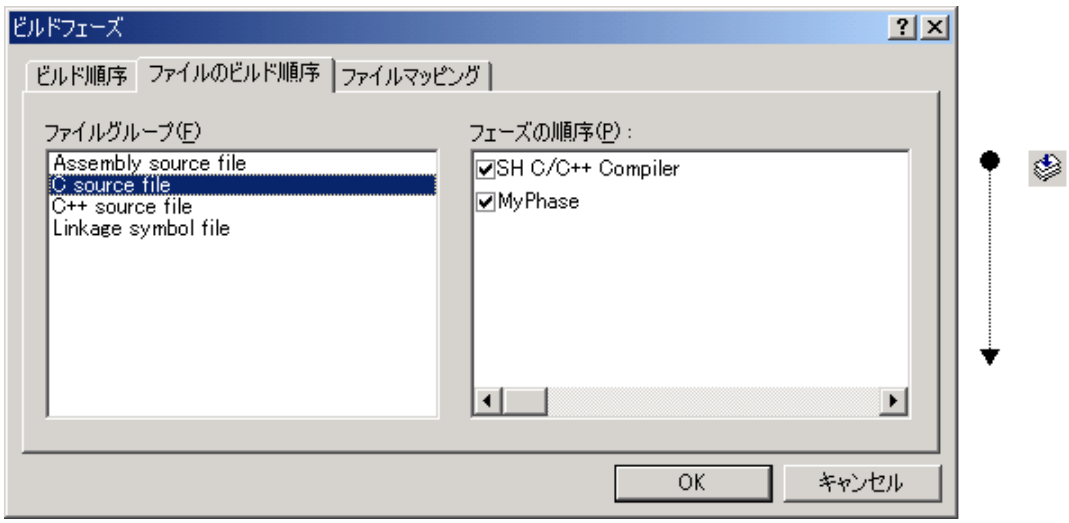


図 3.9: **ビルドフェーズ ダイアログボックス ファイルのビルド順序 タブ**

リストには、[ファイルグループ] リストボックスのファイルグループに対してビルドファイル操

作を選択したとき実行されるすべての現在のフェーズを表示します。図 3.9 では“C source file” ファイルグループが選ばれており、 “Compiler” フェーズと “MyPhase” フェーズが関連付けされています。

[ビルド順序]タブのリストに新しいエントリを追加すると、自動的に[ファイルのビルド順序]タブの[フェーズの順序]リストに新しいエントリが追加されます。例えば、“C source file”を入力とするフェーズを追加します。このフェーズは“ファイルのビルド”操作を“C source file”に適用する時に実行されるフェーズのリストに自動的に追加されます。**[ビルド->コンパイル...]**を選んだときに実行したくないフェーズがある場合、[フェーズの順序]リストのフェーズ名の左にあるチェックボックスのチェックを外してください。

3.4 カスタムビルドフェーズのオプション設定

カスタムフェーズを定義後、フェーズ実行時に使用するコマンドラインオプションを指定します。定義されたフェーズにはそれぞれ **[オプション]**メニューにオプション設定用のメニューがあります。指定するフェーズのオプションを選んでください。起動するダイアログボックスは、選んだカスタムフェーズが単一フェーズか複数フェーズかによって異なります（図 3.3b [単一フェーズ]/[複数フェーズ]ラジオボタン指定）。

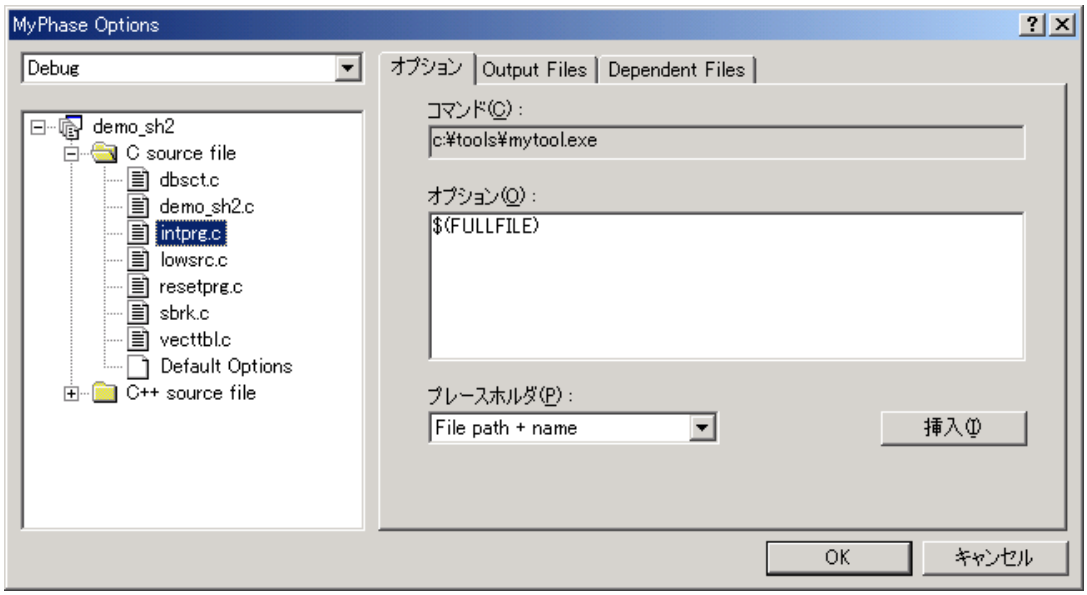


図 3.10: **カスタムフェーズのオプションダイアログボックス（複数フェーズ）**

図 3.10 にカスタムフェーズオプションダイアログボックス（複数フェーズ）を示します。左側にはプロジェクトとファイルのリストがあります。Windows® Explorer と同様の方法で、オプションを変更するプロジェクトとファイルを複数、一回以上選ぶことができます。右側には 3 つのオプションタブがあります。選んだファイルに適用するオプションをここで設定してください。

また、ダイアログボックス左上のコンフィグレーションリストで、どのコンフィグレーションの情報を表示するか選択できます。各コンフィグレーションは[Multiple configurations...]という名のエントリと一緒に並べられています。[Multiple configurations...]を選択すると、ダイアログボックスが表示され、複数のコンフィグレーションを選択できます。

3.4.1 オプションタブ

[Options] タブ（図 3.11）ではフェーズに渡されるコマンドラインオプションを定義できます。 [コマンド] フィールドではフェーズを定義したときに入力したコマンドを表示します （図 3.3d）。 [オプション] フィールドにはコマンドに渡すコマンドライン引数を入力してください。プレースホルダを挿入する場合は、対応するプレースホルダを [プレースホルダ] ドロップダウンリストボックスから選び、 [挿入] ボタンをクリックしてください。プレースホルダの詳細については、付録 C, 「プレースホルダ」を参照してください。

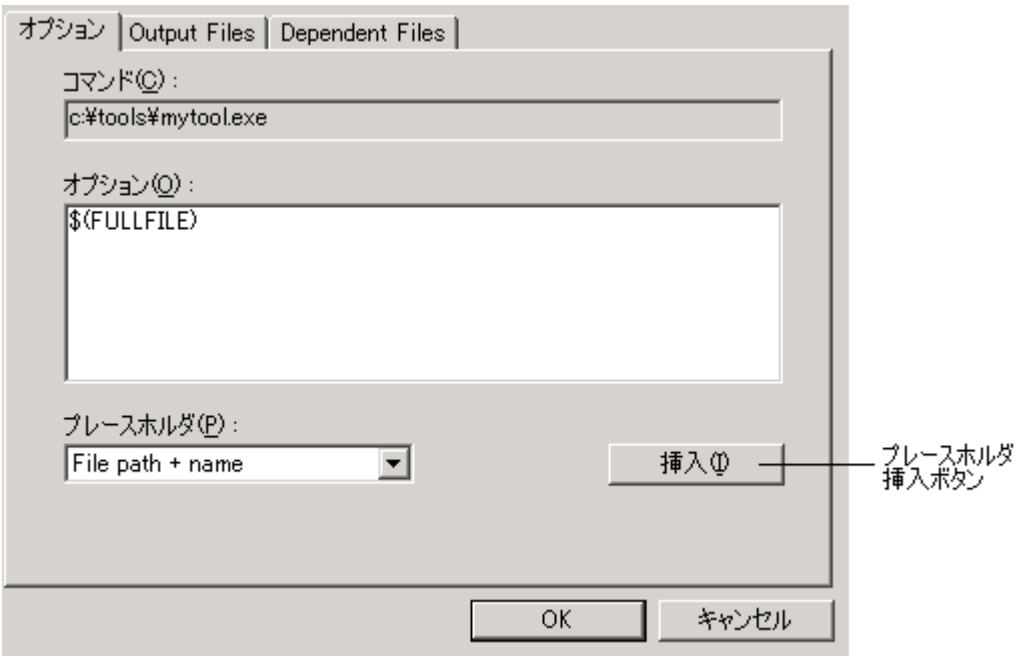


図 3.11: **カスタムフェーズのオプション オプション タブ**

3.4.2 Output Files タブ

“Output Files” タブ（図 3.12）ではフェーズで作成される出力ファイルを指定します。HEW では、ファイルがこのフェーズを通過する前に、出力ファイルの日付が入力ファイルの日付より古いことをチェックしています。出力ファイル作成後入力ファイルが変更された場合、入力ファイルに対してこのフェーズが実行されます。出力ファイルが最新の場合、入力ファイルに対してこのフェーズは実行されません。

注意　出力ファイルを指定しない場合フェーズは常に実行されます。

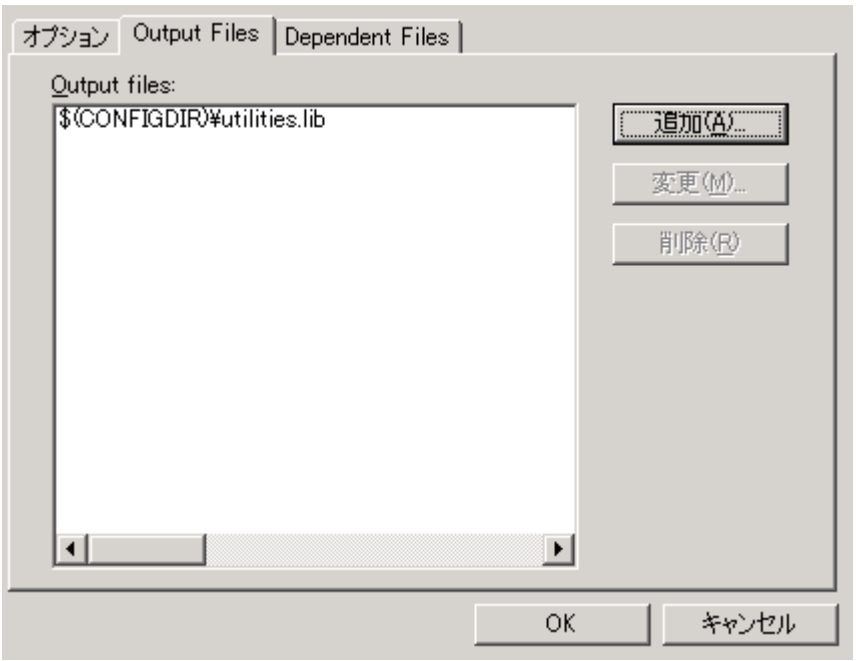


図 3.12: **カスタムフェーズのオプション** Output Files **タブ**

➲ 出力ファイルを追加するには

1. [追加…] ボタンをクリックしてください。[Add Output File] ダイアログボックスが表示されます（図3.13）。
2. ファイル名を入力するか、 [参照…] ボタンで選んでください。
3. [OK] ボタンをクリックすると、リストに出力ファイルを追加します。

図 3.13: Add Output File **ダイアログボックス**

➲ 出力ファイルを変更するには

1. 変更する出力ファイルを選んでください。
2. [変更…] ボタンをクリックすると [Modify Output File] ダイアログボックスが表示されます（タイトル以外は図3.13と同様）。
3. フィールドを変更して [OK] ボタンをクリックしてください。変更した項目がリストに追加されます。

➲ 出力ファイルを削除するには

1. 削除する出力ファイルを選んでください。
2. [削除] ボタンをクリックしてください。

3.4.3 Dependent Files タブ

“Dependent Files” タブ（図 3.14）ではフェーズに必要な依存ファイルを指定します。HEW では、各ファイルがこのフェーズを通過する前に、依存ファイルの日付が入力ファイルの日付より新しいか否かをチェックしています。チェック後、依存ファイルの日付が新しい場合（つまり、入力ファイル作成後に依存ファイルが変更された場合）このフェーズでファイルが実行されます。依存ファイルの日付が入力ファイルの日付より古い場合、このフェーズは実行されません。

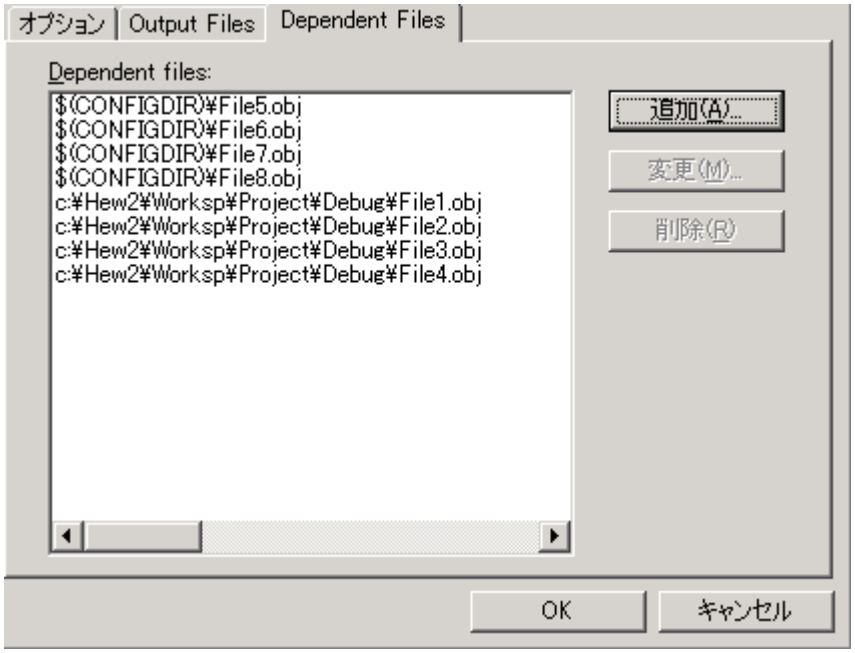


図 3.14: **カスタムフェーズのオプション** Dependent Files **タブ**

➲ 依存ファイルを追加するには

1. [追加…] ボタンをクリックしてください。[Add Dependent File]ダイアログボックスが表示されます（図3.15）。
2. ファイル名を入力するか、 [参照…] ボタンで選んでください。
3. [OK] ボタンをクリックすると、リストに出力ファイルを追加します。

図 3.15: Add Dependent File **ダイアログボックス**

➲ 依存ファイルを変更するには

1. 変更する依存ファイルを選んでください。
2. [変更…] ボタンをクリックすると [Modify Dependent File] ダイアログボックスが表示されます（タイトル以外は図3.15と同じ）。
3. フィールドを変更して [OK] ボタンをクリックすると変更した項目がリストに追加されます。

➲ 依存ファイルを削除するには

1. 削除するファイルを選んでください。
2. [削除] ボタンをクリックしてください。

3.5 ファイルのマッピング

デフォルトでは、フェーズに入力されるファイルはプロジェクトから取得したのものだけです。つまり、[新規ビルドフェーズ]ダイアログボックス（図 3.3b）の[入力ファイルの選択] ドロップダウンリストに指定した種類のプロジェクトファイルだけです。もし前のフェーズから出力されたファイル（中間ファイル）をフェーズで使いたい場合は、[ビルドフェーズ] ダイアログボックス （図 3.16）の[ファイルマッピング] タブで定義してください。

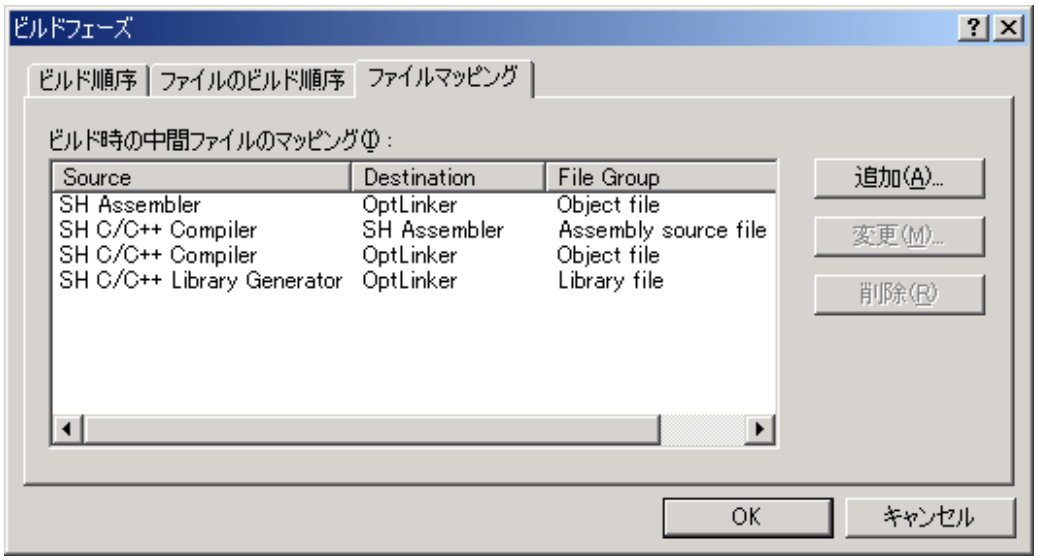


図 3.16: **ビルドフェーズ ダイアログボックス ファイルマッピング タブ**

ファイルマッピングでは、あるフェーズ（出力元フェーズという）で作成したある種類の出力ファイル（中間ファイル）を、他のフェーズ（出力先フェーズという）に渡すように指定します。プロジェクトファイルに加えて中間ファイルも渡されます。

➲ ファイルのマッピングを追加するには

1. [追加...] ボタンをクリックしてください。 [ファイルマッピングの設定] ダイアログボックスが表示されます（図3.17）。
2. [ファイルグループ] ドロップダウンリストボックスから、フェーズ間で渡したいファイルグループを選んでください。
3. [フェーズ元] ドロップダウンリストボックスから出力元フェーズ（ファイルを作成するフェーズ）を選んでください。
4. [フェーズ先] ドロップダウンリストボックスから出力先フェーズ（ファイルを渡す先のフェーズ）を選んでください。
5. [OK] ボタンをクリックすると新しいマッピングが追加されます。

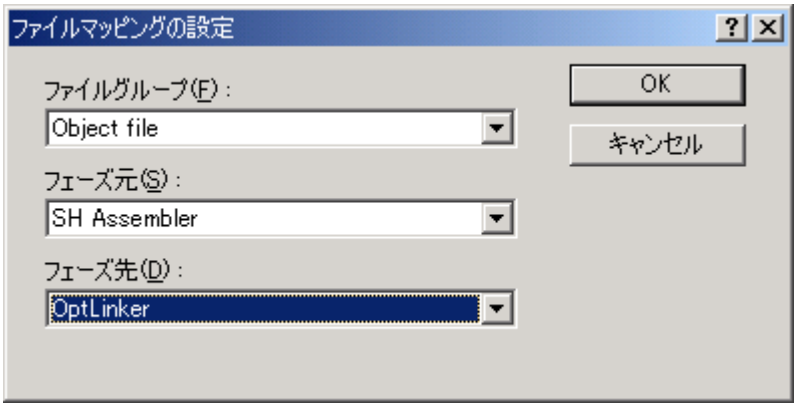


図 3.17: **ファイルマッピングの設定 ダイアログボックス**

➲ ファイルマッピングを変更するには

1. 変更するマッピングを選んでください。
2. [変更...] ボタンをクリックしてください。 [ファイルマッピングの設定] ダイアログボックスが表示されます（図3.17）。
3. 必要に応じてオプションを変更してください。
4. [OK] ボタンをクリックすると変更が有効になります。

3.6　ビルドを管理する

デフォルトでは、HEW はビルドのすべてのフェーズを実行し、途中で致命的なエラーが起こったときだけ中止します。これは [オプション] ダイアログボックスの [ビルド]タブ（図 3.18）で変更することができます。

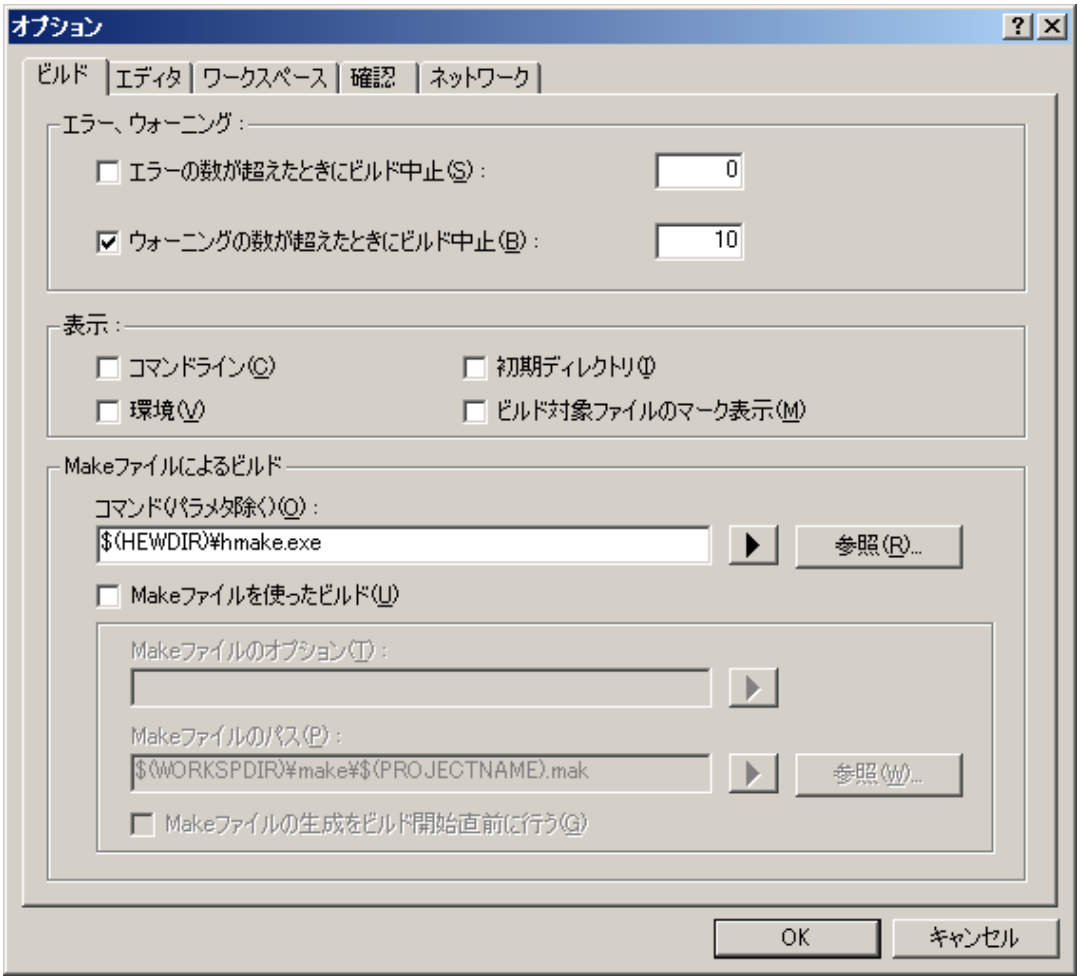


図 3.18: オプションダイアログボックス ビルド タブ

[ツール->オプション...] を選んでダイアログボックスを表示してください。ある一定の回数を超えるエラーが起きた後にビルドを中止したいとき、[エラーの数が超えたときにビルド中止]チェックボックスをチェックして右のフィールドにエラー数を入力してください。指定した数を超えるウォーニングが表示されたときにビルドを中止したいとき、[ウォーニングの数が超えたときにビルド中止]チェックボックスをチェックして右のフィールドにウォーニング数を入力してください。

注意　　上記設定にかかわらず、致命的エラーが発生した場合、ビルドは停止します。

[ビルド]タブでは、エラー数やウォーニング数の制限のほかに、コマンドライン、環境、初期ディ

レクトリの表示の有無を設定することができます。表示するには、それぞれのチェックボックスをチェックしてください。

3.7 ビルドの出力のログを取る

ファイルに各ビルドの結果を保存したいときには、**[ツール-> カスタマイズ...]**を選んで[カスタマイズ]ダイアログボックスを表示して [ログ] タブを選んでください （図 3.19）。 [ログファイル生成] チェックボックスをチェックして、[ディレクトリ] フィールドにログファイルのフルパスを入力するか、[参照...]ボタンをクリックしてパスを選択してください。

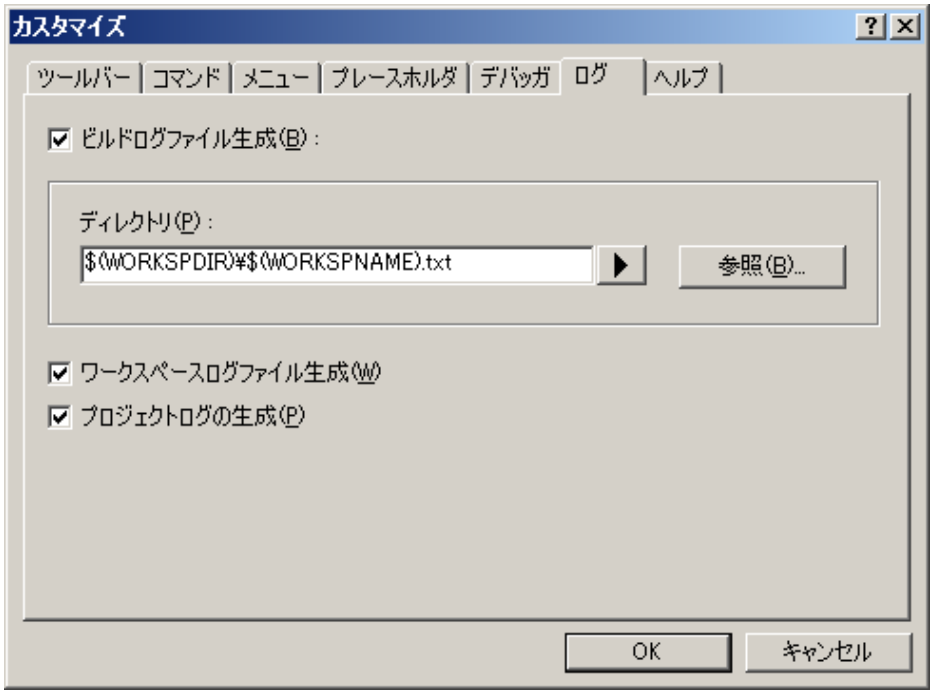


図 3.19: **カスタマイズダイアログボックス ログ タブ**

3.8 ツールチェインのバージョンを変更する

同じツールチェインの２つ以上のバージョンが HEW に登録されているとき図 3.20 に示す[ツールチェインのバージョンの変更]ダイアログボックスでバージョンを選択できます。このダイアログボックスを表示されるには**[ツール->ツールチェインバージョンを変更...]**を選択してください。このダイアログボックスの[ツールチェインバージョン]ドロップダウンリストからバージョンを選択し、[OK]ボタンをクリックしてください。

図 3.20: **ツールチェインのバージョンの変更 ダイアログボックス**

ツールチェインを構成するツールの情報を表示するには[ツールチェインのバージョンの変更]ダイアログボックスの[ツールチェインビルドフェーズ]リストからツールを一つ選択し、[情報...]ボタンをクリックしてください。ツール情報ダイアログボックス（図 3.21）にそのツールの情報が表示されます。このダイアログボックスを閉じるには[閉じる]ボタンをクリックしてください。

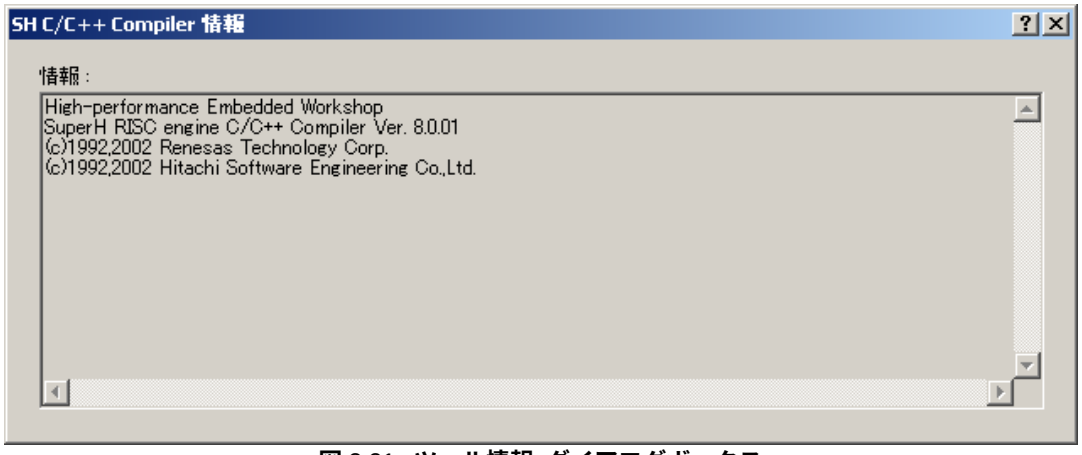


図 3.21: **ツール情報 ダイアログボックス**

3.9 外部デバッガを使う

HEW は外部デバッガツールを起動することができます。他のデバッガを使用する場合は、[ツール]メニューに追加する必要があります。

[カスタマイズ] ダイアログボックスの [デバッガ] タブに外部デバッガに関連する情報を設定します（図 3.22a、図 3.22b、図 3.22c）。一部のターゲットが新しい環境でサポートされていない場合は、古いバージョンのデバッガを使用することができます。**[ツール-> カスタマイズ...]**を選んでダイアログボックスを表示させ、[デバッガ]タブを選んでください。
[外部デバッガの選択]ドロップダウンリストで使用する外部デバッガを選択します。

[Hitachi Debugger Interface (version 4.x or greater)] ：HDIを使用する
[Mitsubishi PD debugger] ：PDデバッガを使用する
[Other external debugger] ：HDIまたはPDデバッガ以外の外部デバッガを使用する
[Non selected] ： 外部デバッガを使用しない

外部デバッガを使用する場合、そのデバッガ固有の情報を構築する必要があります。

➲ HDI を使用するには
1. HDIの場所を指定します。HDI 4.0以降のバージョンでない場合この動作は保証されません。
2. セッションファイルの場所を指定します。起動時にどのセッションをロードするかHDIに伝えます。
3. ダウンロードモジュールの場所を指定します。

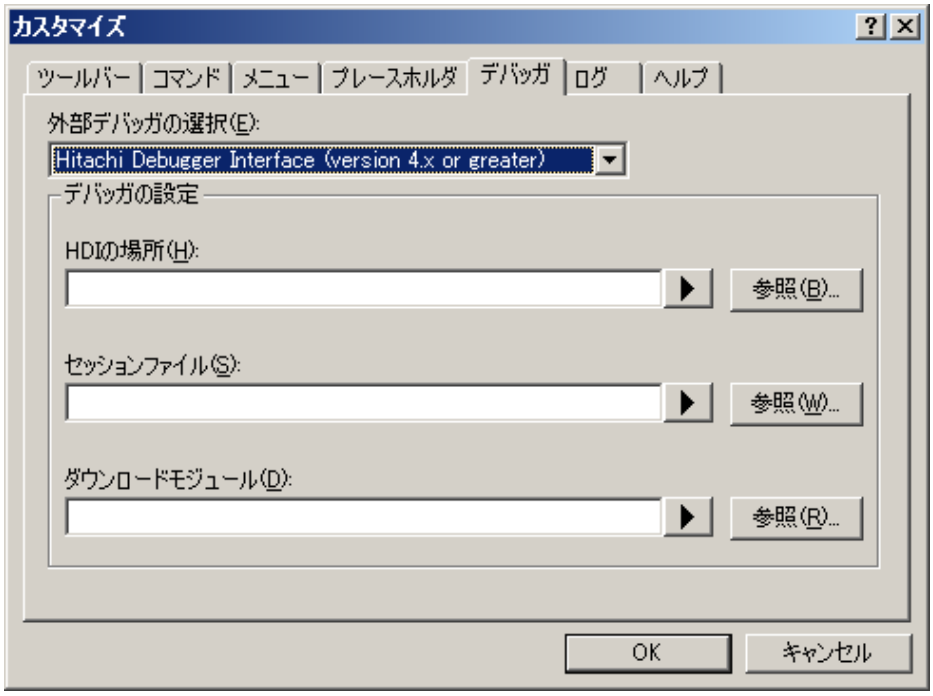


図 3.22a: **外部デバッガの選択** - HDI

➲ PD デバッガを使用するには

1. PDデバッガの場所を指定します。
2. PD Profileファイルの場所を指定します。起動時にどのセッションをロードするかPDデバッガに伝えます。
3. コマンドラインオプションです。PDデバッガの動作を変更するオプションを指定できます。
4. ダウンロードモジュールの場所を指定します。

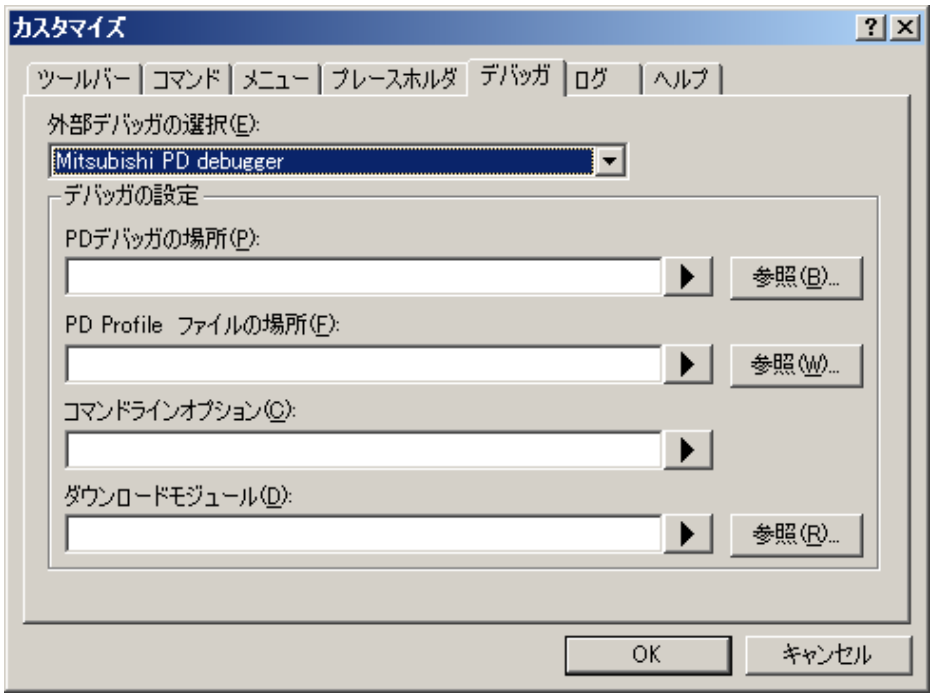


図 3.22b: **外部デバッガの選択** - PD **デバッガ**

➲ HDI または PD デバッガ以外の外部デバッガを使用するには

1. 外部デバッガの場所を指定します。
2. コマンドラインオプションです。外部デバッガの動作を変更するオプションを指定できます。
3. ダウンロードモジュールの場所を指定します。

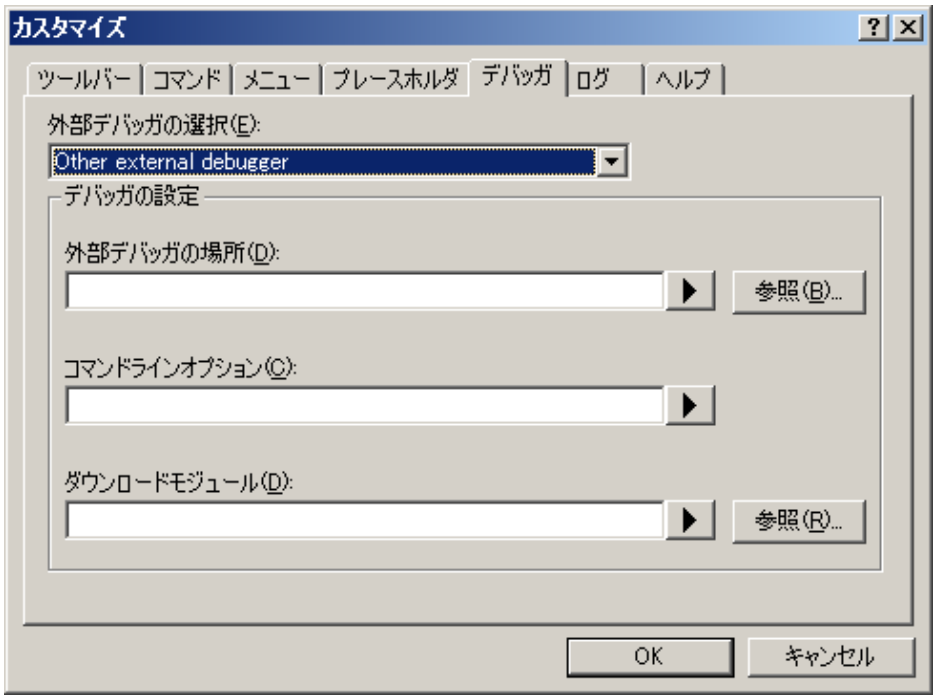


図 3.22c: **外部デバッガの選択** - HDI **または** PD **デバッガ以外の外部デバッガ**

外部デバッガを起動するには、[外部デバッガの実行]ツールバーボタンをクリックしてください。

ビルド後にダウンロードモジュールが変更された場合 HEW から外部デバッガに切り替わり、すぐデバッグできるようになります。

外部デバッガを使用しているときにソースウィンドウのどれかをダブルクリックすると、HEW に切り替わりダブルクリックした行のソースファイルを開いた状態になります。[Other external debugger]を選んだ場合は、この操作を行うことはできません。

3.10 メイクファイルの生成

HEW では、メイクファイルを生成することができます。メイクファイルを使用すると、完全に HEW をインストールしていなくても、現在のプロジェクトをビルドすることができます。HEW をインストールしていない相手にプロジェクトを送ったり、メイクファイルを含むビルド全体をバージョン管理したりする場合に便利です。

本バージョンの HEW では、生成したメイクファイルと Make ツールを使用することもできます。[ビルド]メニューまたはツールバーボタンをクリックし、HEW の標準のビルド処理ではなく、Make ツールを起動することができます。詳細は、「3.11 HEW システム内部のメイクファイルを使用する」を参照してください。

➲ メイクファイルを生成するには

1. メイクファイルを生成するプロジェクトが現在のプロジェクトであることを確認してください。
2. プロジェクトをビルドするビルドコンフィグレーションが現在のコンフィグレーションであることを確認してください。
3. [ビルド->Makeファイルの生成] を選んでください。
4. [Makeファイルの生成]ダイアログボックスが表示されます。（図3.23）
5. [Makeファイルの生成タイプ]で生成したいメイクファイルをラジオボタンで選んでください。
6. 選択した[Makeファイルのフォーマット] が適切であるか確認してください。HEWでは、HMakeおよびNMake、GNUMakeに対応したファイルを生成できます。
7. [サブコマンドファイルの生成]チェックボックスをチェックすると、メイクファイルが出力されたディレクトリ内にコマンドファイルを生成します。GNUMakeファイル形式の場合、このチェックボックスを有効にする必要があります。
8. [Makeファイルの生成時に依存関係をスキャン]チェックボックスをチェックすると、最新のメイクファイルが生成されるよう、依存性のスキャンを行います。
9. [OK]ボタンをクリックしてください。

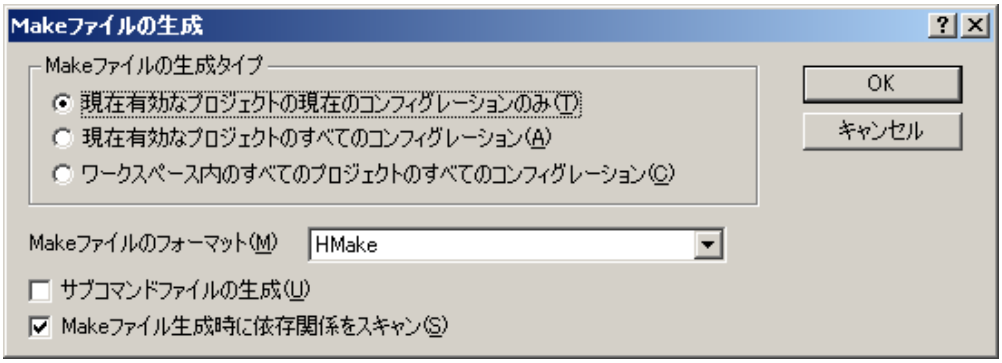


図 3.23: Make **ファイルの生成 ダイアログボックス**

HEW は現在のワークスペースディレクトリ内に“make”サブディレクトリを作り、その中にメイクファイルを生成します。メイクファイルの名前は、現在のプロジェクトやコンフィグレーションに拡張子.mak を付けたものです(例：project　debug.mak)。HEW により生成されたメイクファイルは、HEW をインストールしたディレクトリにある実行ファイル HMAKE.EXE で実行できます。ただし、ユーザが変更したメイクファイルは実行できません。HMAKE の詳細については、付録 F,「HMAKE ユーザガイド」を参照してください。

➲ メイクファイルを実行するには

1. コマンドウィンドウを開き、メイクファイルが生成された“make”ディレクトリに移動してく

ください。

2. HMAKEを実行してください。コマンドラインは”HMAKE.EXE△<メイクファイル名>”です。

注意　生成したメイクファイルが移動可能か否かは、プロジェクト自体が移動可能か否かに依存します。たとえば、出力ディレクトリやインクルードファイルディレクトリへのフルパスを含むコンパイラオプションがあると、異なるインストール環境下の別のユーザがビルドした場合、失敗する可能性があります。一般的に、できるだけプレースホルダを使用して、フルパス、または特定のパスの使用はなるべく避けてください。

3.11 HEW システム内部のメイクファイルを使用する

HEW では、[ビルド]をクリックすると内部ビルドを構築します。本機能は、標準の HEW のビルド処理ではなく外部のメイクファイルツールを使用することができます。

ユーザの必要とするビルドには標準の HEW のビルドで十分ですが、時には標準の HEW のビルド処理に依存せず、メイクファイルを使用したい場合もあります。この方法を以下に説明します。

➲ メイクファイルの実行を設定するには

1. メイクファイルでビルドするためのHEWのワークスペースを必要に応じて作成してください。そのときは使用するツールチェインをメイクファイルと同じにする必要があります。
2. **[ツール-> オプション]** をクリックしてください。
3. [ビルド] タブをクリックしてください。（図3.24）
4. [コマンド（パラメタ除く）] にメイクファイルツールを設定してください。デフォルトでは、HEWと一緒に出荷されるルネサス製Makeツールに設定されています。名称はHMAKE.exeで、格納先はHEWインストールディレクトリ“$(HEWDIR)”です。
5. [Makeファイルを使ったビルド] チェックボックスをチェックしてください。[ビルド]ボタンをクリックしたときに、標準のビルド処理ではなくメイクファイルを実行することをHEWに伝えます。
6. [Makeファイルのパス] を設定してください。HEWがメイクファイルを生成するディレクトリがデフォルト値として設定されています。本項目はコンフィグレーションがベースであるため、各コンフィグレーションに異なるメイクファイルを設定することもできます。

注意　必ずしも、HEW で生成したメイクファイルを使用する必要はありません。ご使用の Make ツールでサポートしていれば、どの形式でも使用することができます。

7. [Makeファイルのオプション] を設定してください。HMAKEを使用してプロジェクトまたはコンフィグレーションの選択オプションを指定することで、同一ファイルから異なるビルドを起動することができます。ここでも本項目はコンフィグレーションがベースであるため、各コンフィグレーションに異なるメイクファイルオプションを設定することができます。HMAKEのオプションについては、付録 F, 「HMAKEユーザガイド」の「F.1.3 パラメータ」を参照してください。
8. 最後のオプションは [Makeファイルの生成をビルド開始直前に行う] です。メイクファイルを実行する前に、デフォルトのメイクファイル生成機能を起動します。つまり、メイクファイルは常に最新の状態のHEWプロジェクトに対応しています。
9. [OK] ボタンをクリックしてください。

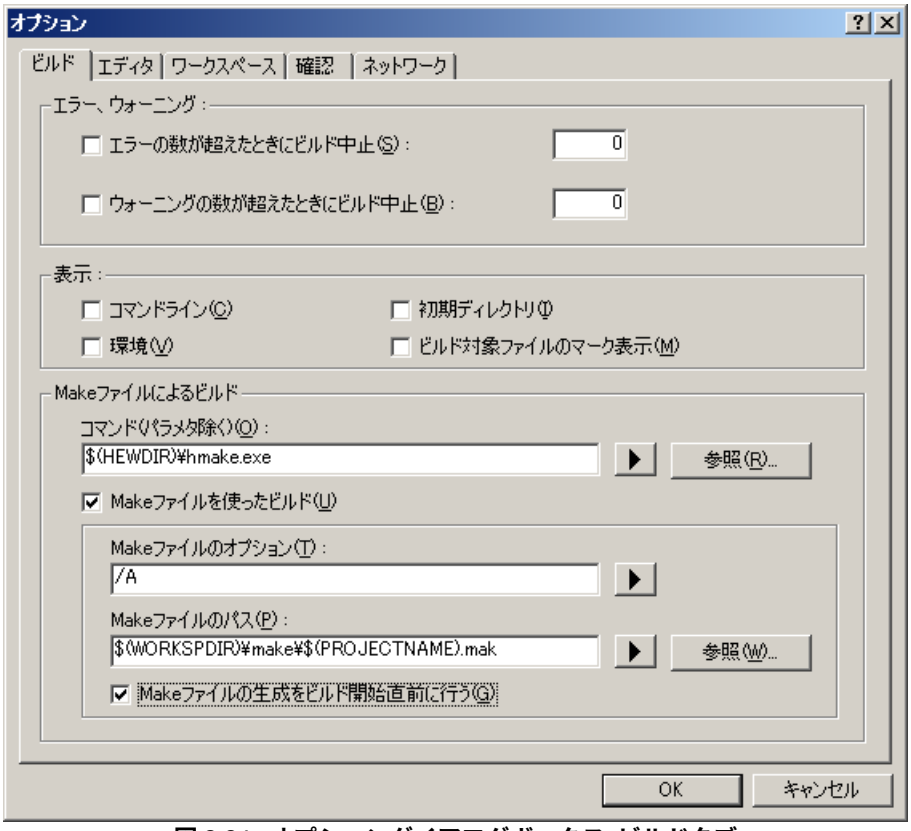


図 3.24: **オプションダイアログボックス ビルドタブ**

本システムを設定すると、[ビルド]ボタン、メニュー、およびキーボードのショートカットは Make ツールの実行とリンクします。標準のビルド処理同様、出力先はすべてアウトプットウィンドウです。HEW がサポートしているツールチェインを使用する場合、エラーおよび警告をダブルクリックするとソースファイルにジャンプします。エラー等の行を選択して F1 キーを押下すると、詳細を表示する機能をサポートしています。HMAKE.exe システムをご使用になる場合、[すべてをビルド]ボタンをクリックすると、HMAKE にコマンドを送り強制的に再ビルドすることにご注意ください。GNUMake ツールをご使用の場合には、本機能をサポートしていません。

3.12 リンク順序をカスタマイズする

HEW では、通常オブジェクトファイルは、ファイル名のアルファベット順にリンクしますが、リンクする順番をカスタマイズすることができます。

➲ リンク順序をカスタマイズするには

1. **[ビルド->リンク順の指定...]** を選んでください。
2. 表示されるダイアログボックスで [リンク順序のカスタマイズの使用] チェックボックスをチェックしてください。（図3.25）
3. オブジェクトの順序を並びかえることができます。モジュールを選択し、[上へ]ボタンまたは[下へ]ボタンをクリックして任意の位置に移動してください。
4. 各モジュールには、そのモジュールの属性によって異なるアイコンを表示しています。
5. [OK] ボタンをクリックしてください。

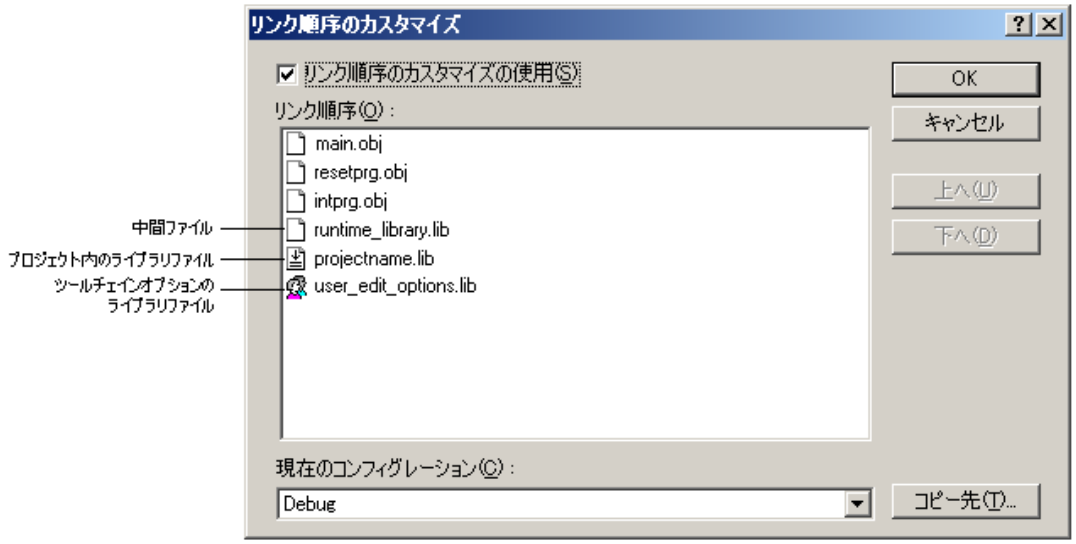


図 3.25: **リンク順序のカスタマイズ ダイアログボックス**

複数のコンフィグレーションを使用する場合、リンク順序を同じにする場合があります。現在のリンク順序の設定を他のコンフィグレーションにコピーする方法を以下に説明します。

➲ 他のコンフィグレーションにリンク順序をコピーするには

1. **[ビルド->リンク順の指定...]** を選んでください。
2. [現在のコンフィグレーション] リストで、コピー元のコンフィグレーションを選択してください。現在のコンフィグレーションがデフォルトで表示されます。
3. [コピー先...] ボタンをクリックしてください。ダイアログボックスが表示され、現在のリンク順序を現在のプロジェクトのどのコンフィグレーションにコピーするかを指定します。コンフィグレーションを選択し、[OK]ボタンをクリックしてください。
4. [OK] ボタンをクリックしてください。

4. エディタの使用

この章では HEW が提供するエディタの使い方を説明します。

4.1 エディタウィンドウ

エディタウィンドウには、表示中や編集中のファイルウィンドウが含まれます（図 4.1）。

常に 1 つのウィンドウだけがアクティブです。それをアクティブウィンドウ（または現在のウィンドウ）と呼びます。アクティブウィンドウのタイトルバーは他のウィンドウとは色が異なります。（図 4.1 では“dbsct.c”がアクティブウィンドウです。）文字入力やテキスト貼りつけなどのテキスト操作はアクティブウィンドウでのみ行うことができます。

アクティブウィンドウから他のソースファイルウィンドウへ切り替えるには（他のウィンドウをアクティブウィンドウにする場合）、以下の方法があります。

(a) そのウィンドウが表示されていればそれをクリックする。
(b) CTRL+TAB キーを押下して順次ウィンドウをアクティブにする。
(c) [ウィンドウ]メニューからウィンドウ名を選ぶ。
(d) エディタウィンドウの下の該当するタブを選ぶ。

ファイルが変更されている場合、ファイル名のタイトルバーにアスタリスク（＊）が追加されます。アスタリスクはファイルを保存するまで表示されます。現在のウィンドウですべての変更が戻された(Undo)場合には、アスタリスクは削除されます。

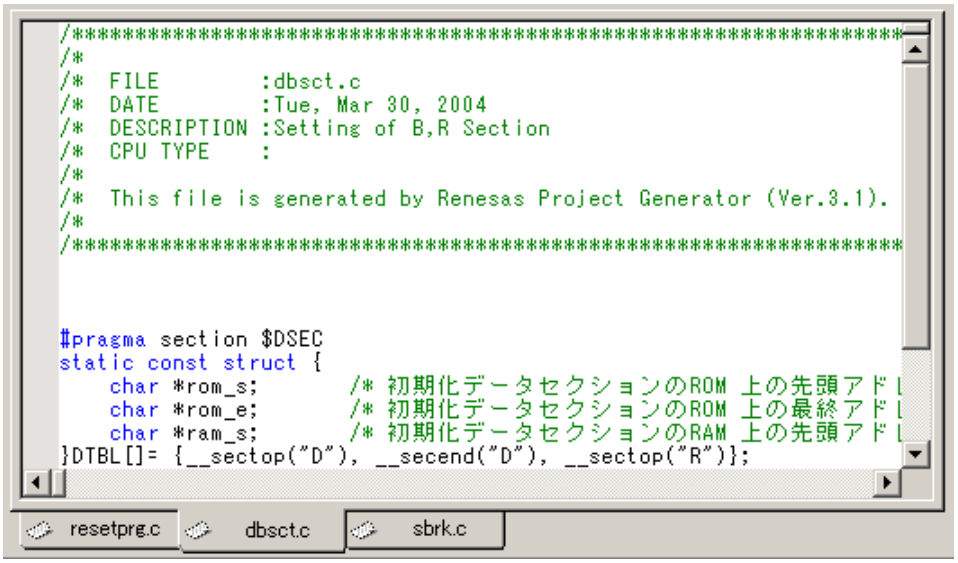


図 4.1: **エディタウィンドウ**

4.2 複数のファイルを使う

プロジェクトのファイルを操作するときにはエディタウィンドウを使います。エディタによって、一度に複数のファイルを開いたり、ファイル間の切り替えをしたり、異なる構成に並べ替えたり、任意の順序に編集できます。ウィンドウ上での操作はWindows®のアプリケーションの一般的な機能であり、[ウィンドウ]メニューから選ぶことができます。

[ウィンドウ->重ねて表示]
開いているすべてのウィンドウの左上が見えるように重ねて並べます。

[ウィンドウ->上下に並べて表示]
開いているすべてのウィンドウが互いに重ならずにエディタウィンドウの全体を占めるように並べます（水平方向）。

[ウィンドウ->左右に並べて表示]
開いているすべてのウィンドウが互いに重ならずにエディタウィンドウの全体を占めるように並べます（垂直方向）。

[ウィンドウ->アイコンの整列]
すべての最小化したウィンドウをエディタウィンドウの下に並べます。

[ウィンドウ->すべて閉じる]
開いているエディタウィンドウをすべて閉じます。

また、エディタのファイルはノートブック形式で表示することができます。つまり、各ファイルにタブが付いていてファイル間の行き来を容易にできます。

➲ ファイルをノートブック形式で表示するには
1.[ツール->オプション…] を選んでください。 [オプション] ダイアログボックスが表示されるので [エディタ] タブを選んでください。
2.[ブック形式でファイルを表示] チェックボックスをチェックしてください。
3.[OK] ボタンをクリックしてください。

4.2.1 エディタツールバー

エディタツールバーには、エディタ、検索、ブックマーク、テンプレートの４つの関連するツールバーがあります。これにより、頻繁に使うエディタの機能を簡単に選ぶことができます。以下に各ボタンの機能を説明します。

4.2.2 エディタツールバーボタン

新規ファイル
新規のソースファイルウィンドウをデフォルト名で作成します。ファイルを保存するときにファイルの名前を変えることができます。

ファイルを開く

ファイルを開きます。標準のファイル選択用ダイアログボックスが表示されます。開きたいファイルを選び“開く”ボタンをクリックします。

ファイルの保存

アクティブウィンドウのソースファイルを保存します。

すべてのファイルの保存

開いているすべてのファイルを保存します。

ファイルの印刷

クリックすると、現在のアクティブウィンドウの内容を印刷します。

カット

選択されたテキストを削除してWindows® のクリップボードにコピーします。（貼りつけ操作をすると、再びファイルに貼りつけることができます。）

コピー

選択されたテキストを Windows® のクリップボードにコピーします。

貼りつけ

アクティブウィンドウのカーソル位置にクリップボードの内容を貼りつけます。

かっこの組み合わせ

かっこ{ }, [ ] , ( )内のテキストをハイライト表示します。これは、開きかっこ（{）で始まり閉じかっこ（}）で終わるC/C++言語のコードの構造を知りたいときに便利です。始まりのかっこを選択するか、始まりのかっこの前にカーソルを置いて、このボタンをクリックしてください。かっこの組み合わせの詳細については、この章の後半の「かっこの組み合わせ」を参照してください。

テンプレートの挿入

あらかじめ決められたテンプレートを現在のカーソル位置に挿入します。[テンプレートの挿入]ダイアログボックスが表示されます。テンプレート名を選び、[OK]ボタンをクリックしてください。テンプレートの挿入の詳細については、この章の後半の「テンプレート」を参照してください。

ブックマークの設定と解除

HEWはブックマーク機能を提供します。ブックマークを設定するには、行を選び、このボタンをクリックしてください（エディタウィンドウの左余白に緑色のマークが付きます）。ブックマークを解除するには、行を選び、このボタンをクリックしてください（エディタウィンドウの左余白のマークが消えます）。ブックマークの詳細については、この章の後半の「ブックマーク」を参照してください。

4.2.3 検索ツールバーボタン

ファイル間検索

複数のファイルを指定された文字列で検索します。検索結果はすべてアウトプットウィンドウの“Find in Files”タブに表示します。詳細は、この章の後半の「複数のファイル間でのテキスト検索」を参照してください。

検索

現在のファイルを指定された文字列で検索します。[検索]ダイアログボックスが表示されるので、検索パラメータを指定してください。

次を検索

現在の検索文字列の次の出現を検索します。

前を検索

現在の検索文字列の直前の出現を検索します。

4.2.4 ブックマークツールバーボタン

ブックマークの設定と解除

現在の行にブックマークを設定します。または、現在の行のブックマークを解除します。

次のブックマーク

現在の行から現在のファイルの次のブックマークに飛びます。

前のブックマーク

現在の行から現在のファイルの前のブックマークに飛びます。

全ブックマーク解除

現在のファイルのすべてのブックマークを解除します。

4.2.5 テンプレートツールバーボタン

テンプレートの定義
挿入するテンプレートテキストを定義します。

テンプレートの挿入
ドロップダウンリストで選んだテンプレートを、現在のカーソル位置に挿入します。

4.3 標準のファイル操作

4.3.1 新規ファイルの作成

➲ 新しいエディタウィンドウを作成するには
[ファイル->新規作成] を選択するか、“新規ファイル”ツールバーボタン () をクリックするか、”CTRL+N”キーを押下してください。

新しいウィンドウにはデフォルトで名前がつきます。ファイルを保存するときに名前を変更できます。

4.3.2 ファイルの保存

➲ エディタウィンドウの内容を保存するには
1. 内容を保存するウィンドウがアクティブであることを確認してください。
2. [ファイル->上書き保存] を選択するか、 [ファイルの保存] ツールバーボタン () をクリックするか、”CTRL+S”キーを押下してください。
3. ファイルを新規に保存する場合、名前を付けて保存するダイアログボックスが表示されます。ファイル名とディレクトリを指定して、[保存]をクリックしてください。
4. すでに保存したことのあるファイルの場合は、ダイアログボックスを表示せずにファイルが更新されます。

➲ エディタウィンドウの内容を新しい名前で保存するには
1. 内容を保存するウィンドウがアクティブであることを確認してください。
2. [ファイル->名前を付けて保存...] を選んでください。
3. 名前を付けて保存するダイアログボックスが表示されます。ファイル名とディレクトリを指定して[保存]をクリックしてください。

4.3.3 全ファイルの保存

➲ すべての開いているエディタウィンドウの内容を保存するには
1. [ファイル->すべて保存] を選ぶか、 [すべて保存] ツールバーボタン () をクリックしてください。
2. 新規にファイルを保存する場合、名前を付けて保存するためのダイアログボックスが表示されます。ファイル名とディレクトリを指定してください。

3. すでに保存したことのあるファイルの場合は、ダイアログボックスを表示せずにファイルが更新されます。

4.3.4 ファイルを開く

➲ ファイルを開くには

1. [ファイル->開く...] を選ぶか、 [ファイルを開く] ツールバーボタン () をクリックするか、 “CTRL+O”キーを押下してください。
2. ファイルを開くダイアログボックスが表示されます。右のディレクトリブラウザを使って、開きたいファイルのあるディレクトリに移動してください。[ファイルの種類]コンボボックスでファイルの種類を選んでください。 (“All Files” (*.*) を指定すると、そのディレクトリのすべてのファイルが表示されます。)
3. ファイルを選んで [開く] ボタンをクリックしてください。

開いた最新のファイル 4 つは[ファイル]メニューの[最近使ったファイル]サブメニューに追加されます。この機能は最近開いたファイルを再び開きたいときに便利です。

➲ 最近利用したファイルを開くには
[ファイル->最近使ったファイル] メニューオプションを選択してこのサブメニューから開きたいファイルを選択してください。または、ワークスペースウィンドウの[Projects]タブで開きたいファイルをダブルクリックするか、ファイルを選び、右マウスボタンをクリックしてポップアップメニューから[開く <ファイル名>]オプションを選んでください。

4.3.5 ファイルを閉じる

個別にファイルを閉じるには、以下のいずれかの方法で閉じます。

- エディタウィンドウのシステムメニュー（[最大化]表示されていないときの各ウィンドウの左上）をダブルクリックする。
- エディタウィンドウのシステムメニュー（[最大化]表示されていないときの各ウィンドウの左上）をクリックして[閉じる]メニューオプションを選ぶ。
- アクティブウィンドウであることを確認後”CTRL+F4”キーを押下する。
- アクティブウィンドウであることを確認後[ファイル->閉じる]を選ぶ。
- “閉じる”ボタン（[最大化]表示されていないときの各ウィンドウの右上）をクリックする。

➲ すべてのウィンドウを閉じるには
[ウィンドウ->すべて閉じる]を選んでください。

4.4 ファイルを編集する

HEW エディタは標準的な編集機能をサポートします。通常の方法（メニュー、ツールバー、キーボードのショートカット）で編集できるほか、各エディタウィンドウにあるポップアップメニュー（またはローカルメニューという）でも編集することができます。ポップアップメニューを表示するには、開いたウィンドウにポインタを置き、マウスの右ボタンをクリックしてください。表 4.1 に編集の基本操作を示します。

表 4.1: 編集の基本操作

名称	機能	操作
切り取り	ハイライト表示されたテキストを削除し Windows®クリップボードに貼りつける	“カット”ツールバーボタンをクリック [編集->切り取り]を選択 [切り取り] -ローカルメニューを選択 CTRL+X キーを押下
コピー	ハイライト表示されたテキストをコピーし Windows®クリップボードに貼りつける	“コピー ”ツールバーボタンをクリック [編集->コピー] を選択 [コピー] -ローカルメニューを選択 CTRL+C キーを押下
貼り付け	Windows®クリップボードの内容をコピーしてアクティブウィンドウのカーソル位置に貼りつける	“貼り付け”ツールバーボタンをクリック [編集->貼り付け]を選択 [貼り付け] - ローカルメニューを選択 CTRL+V キーを押下
削除	ハイライト表示されたテキストを削除する（Windows®クリップボードに貼りつけない）	[編集->削除] を選択 Delete キーを押下
すべて選択	アクティブウィンドウの内容すべてを選択（ハイライト表示）する	[編集->すべて選択] を選択 CTRL+A キーを押下
元に戻す	最新の編集操作を取り消す	[編集->元に戻す] を選択 CTRL+Z キーを押下
やり直し	最新の取り消した編集操作をやり直す	[編集->やり直し] を選択 CTRL+Y キーを押下

4.5 検索とファイル内の移動

HEW には、検索、置換、ファイル間の操作の機能があります。次の 3 節にそれらの機能の使い方を示します。

4.5.1 テキストの検索

➲ 現在のファイルのテキストを検索するには

1. 内容を検索するウィンドウがアクティブウィンドウであることを確認してください。
2. 検索開始位置にカーソルを置いてください。
3. [編集->検索...] を選ぶか、”CTRL+F”キーを押下するか、エディタウィンドウのローカルメニューから [検索...] を選ぶか、 [検索] ツールバーボタン () をクリックしてください。[検索]ダイアログボックスが表示されます （図4.2）。

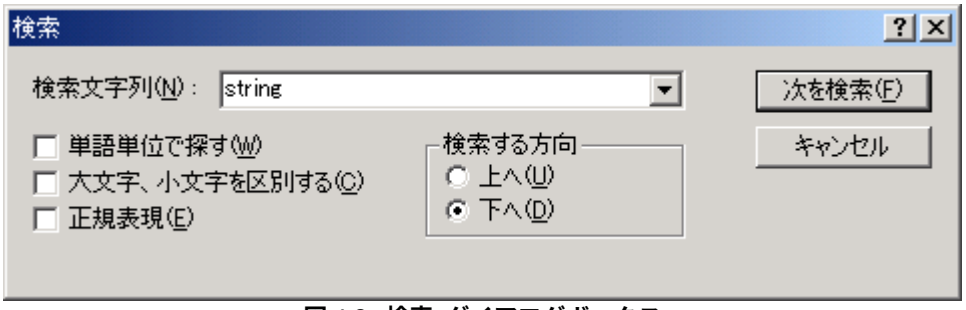


図 4.2: **検索 ダイアログボックス**

4. [検索文字列] フィールドに検索するテキストを入力するか、ドロップダウンリストボックスから以前検索した文字列を選んでください。ファイルウィンドウでテキストを選んでから[検索]ダイアログボックスを開くと、選んだテキストが自動的に[検索文字列]フィールドに入ります。
5. 単語単位で文字列を探す場合は、 [単語単位で探す] チェックボックスをチェックしてください。このオプションを選択しない場合は、一部でも一致する文字列が検索されます。
6. 大文字と小文字を識別したいときは [大文字、小文字を区別する] チェックボックスをチェックしてください。
7. 正規表現を使ってテキストを検索する場合 [正規表現] チェックボックスをチェックしてください。詳細は付録B, 「正規表現」を参照してください。
8. [検索する方向] ラジオボタンにより、検索する方向を指定できます。[下へ]を選ぶと、カーソル位置からファイルの下の方向に検索します。[上へ]を選ぶと、カーソル位置から上の方向に検索します。
9. [次を検索] ボタンをクリックすると検索を始めます。検索を終了するには[キャンセル] ボタンをクリックしてください。

また、複数のファイル間での検索もできます。

4.5.2 複数のファイル間でのテキスト検索

➲ 複数のファイルでテキストを検索するには

1. [編集->ファイル内から検索...] を選ぶか、エディタウィンドウのローカルメニューから [ファイル内から検索...] を選ぶか、 [ファイルの中から検索] ツールバーボタン () をクリックしてください。[ファイルから検索]ダイアログボックスが表示されます（図4.3）。

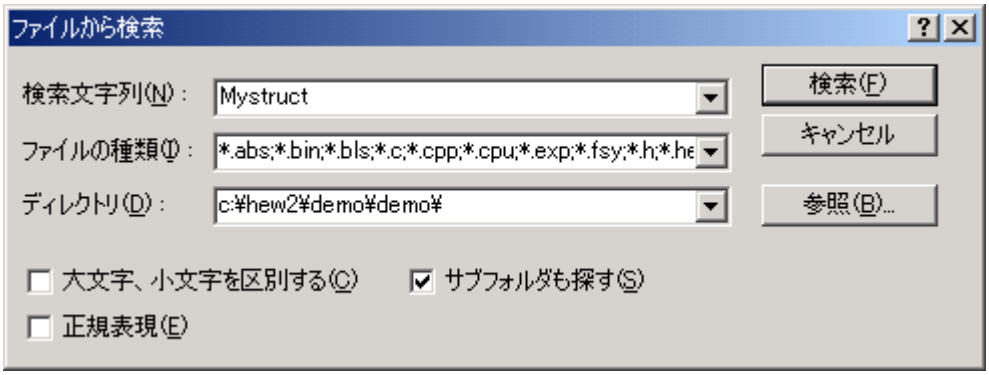


図 4.3: **ファイルから検索 ダイアログボックス**

2. [検索文字列] フィールドに検索したいテキストを入力するか、ドロップダウンリストボックスから以前検索した文字列を選んでください。ファイルウィンドウでテキストを選んでから [ファイルから検索]ダイアログボックスを開くと、そのテキストが自動的に[検索文字列] フィールドに入ります。
3. [ファイルの種類] フィールドに、検索するファイルの拡張子を指定してください。複数の拡張子を指定するときは、セミコロンで区切ってください（例：*.c;*.h）。
4. [ディレクトリ] フィールドに検索するファイルのディレクトリを指定してください。または、[参照...] ボタンをクリックして画面上で検索するファイルを指定してください。
5. 指定したディレクトリとその下のディレクトリすべてを検索したいときは、 [サブフォルダも探す] チェックボックスをチェックしてください。[ディレクトリ]フィールドで指定したディレクトリだけを検索したいときは、[サブフォルダも探す]チェックボックスがチェックされていないことを確認してください。
6. 大文字と小文字を識別したいときは [大文字、小文字を区別する] チェックボックスをチェックしてください。
7. 正規表現を使ってテキストを検索したいときは、 [正規表現] チェックボックスをチェックしてください。 詳細は付録 B,「正規表現」を参照してください。
8. [検索] ボタンをクリックすると検索を始めます。 検索結果はアウトプットウィンドウの [Find in Files]タブに表示されます。アウトプットウィンドウの文字列をダブルクリックすると、ファイルの当該文字列にジャンプします。

4.5.3 テキストを置換する

テキストの置換は前節で説明したテキストの検索に似ています。異なる点は、テキスト検索後、他のテキストに置き換えるところです。

➲ ファイルのテキストを置換するには

1. 内容を置換するウィンドウがアクティブであることを確認してください。
2. 検索を始める位置にカーソルを置いてください。
3. [編集->置換...] を選ぶか、”CTRL+H”キーを押下するか、エディタウィンドウのローカルメニューから [置換...] を選んでください。[置換]ダイアログボックスが表示されます（図4.4）。
4. [置換文字列] フィールドに置換前のテキストを入力してください。または、ドロップダウンリストボックスから以前に検索した文字列を選んでください。ファイルウィンドウでテキストを選んでから[置換]ダイアログボックスを開くと、選んだテキストが自動的に[置換文字列]フィールドに入ります。
5. [置換後の文字列] フィールドに置換後のテキストを入力してください。または、ドロップダウンリストボックスから以前に検索した文字列を選んでください。

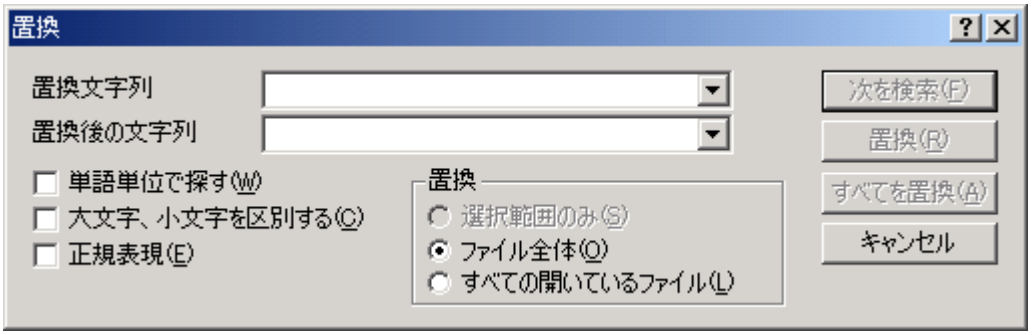


図 4.4: **置換 ダイアログボックス**

6. 単語単位で文字列を探す場合は、 [単語単位で探す] チェックボックスをチェックしてください。このオプションを選択しない場合は、一部でも一致する文字列が検索されます。
7. 大文字と小文字を識別したいときは [大文字、小文字を区別する] チェックボックスをチェックしてください。
8. 正規表現を使ってテキストを検索したいときは、 [正規表現] チェックボックスをチェックしてください。 詳細は付録 B, 「正規表現」を参照してください。
9. [次を検索] をクリックすると、検索文字列、条件に合致したものの次を検索します。置換したい場合は[置換]をクリックしてください。[すべてを置換]をクリックすると、該当するすべての文字列を置換します。[キャンセル]をクリックすると、置換動作を停止します。
   なお、[置換]で[選択範囲のみ]を選択している場合はテキストの選択範囲が置換対象となり、[ファイル全体]を選択している場合はファイル全体が置換対象となります。[すべての開いているファイル]を選択している場合は、エディタで現在開いているファイルがすべて置換対象となります。

4.5.4 指定した行にジャンプする

➲ ファイルの指定した行にジャンプするには

1. ファイルがアクティブであることを確認してください。
2. [編集->ジャンプ...] を選ぶか、”CTRL+G”キーを押下するか、エディタウィンドウのローカルメニューから [ジャンプ...] を選んでください。[ジャンプ]ダイアログボックスが表示されます（図4.5）。
3. ダイアログボックスに指定する行番号を入力して、 [OK] ボタンをクリックしてください。
4. カーソルが指定した行の先頭に移ります。

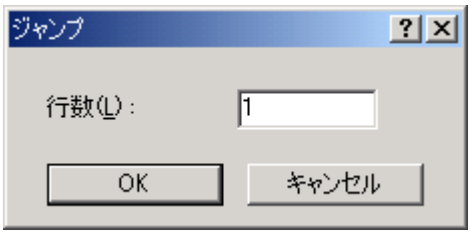


図 4.5: **ジャンプダイアログボックス**

4.6 ブックマーク

一度に多くの大容量のファイルを扱うとき、必要な行や領域を見つけるのが難しくなります。ブックマークをあらかじめ特定の行に設定しておくと、後でその行にジャンプできます。例えば、行数の多い C 言語ファイルの各関数の定義位置にブックマークを設定すると便利です。ブックマークは、解除するまでまたは、設定したファイルを閉じるまで有効です。

➲ ブックマークを設定するには

1. カーソルを設定する行に置いてください。
2. [編集->ブックマーク->ブックマークの挿入/削除] を選ぶか、”CTRL+F2”キーを押下するか、ローカルメニューから [ブックマーク->ブックマークの挿入/削除] を選ぶか、 [ブックマークの挿入/削除] ツールバーボタン ( ) をクリックしてください。
3. ブックマークが設定されると、その行の左余白に緑のマークが表示されます。

➲ ブックマークを解除するには

1. カーソルを解除する行に置いてください。
2. [編集->ブックマーク->ブックマークの挿入/削除] を選ぶか、”CTRL+F2”キーを押下するか、ローカルメニューから [ブックマーク->ブックマークの挿入/削除] を選ぶか、 [ブックマークの挿入/削除] ツールバーボタン ( ) をクリックしてください。
3. 解除されると左余白部分は通常のテキスト表示に戻ります。

➲ 次のブックマークにジャンプするには

1. カーソルが検索するファイルの中にあることを確認してください。
2. [編集->ブックマーク->次のブックマーク] を選ぶか、”F2”キーを押下するか、ローカルメニューから [ブックマーク->次のブックマーク] を選ぶか、または [次のブックマークへ] ツールバーボタン ( ) をクリックしてください。

➲ 同じファイルの一つ前のブックマークに戻るには

1. カーソルが検索するファイルの中にあることを確認してください。
2. [編集->ブックマーク->前のブックマーク] を選ぶか、”SHIFT+F2” キーを押下するか、ローカルメニューから [ブックマーク->前のブックマーク] を選ぶか、 [前のブックマークへ] ツールバーボタン () をクリックしてください。

➲ ファイル内のすべてのブックマークを解除するには

1. すべてのブックマークを解除するファイルのウィンドウをアクティブにしてください。
2. [編集->ブックマーク->すべてのブックマークの削除] を選ぶか、ローカルメニューから [ブックマーク->すべてのブックマークの削除] を選ぶか、 [すべてのブックマークの削除] ツールバーボタン () をクリックしてください。

4.7 ファイルを印刷する

➲ ファイルを印刷するには

1. 印刷するファイルのウィンドウをアクティブにしてください。
2. [ファイル->印刷...] を選ぶか、[印刷] ツールバーボタン () をクリックするか、”CTRL+P” キーを押下してください。

4.8 テキストのレイアウト

この章では、エディタウィンドウの中でテキストのレイアウトを設定する方法を説明します。

4.8.1 ページ設定

HEW エディタからファイルを印刷するとき、印刷ダイアログボックスの設定により、印刷方法が変わります（例えば、片面印刷、両面印刷など）。また、[ページ設定]ダイアログボックスでは、印刷するテキストの余白（上下左右）を指定できます。プリンタによっては、A4 サイズの端まで印刷できない場合があるので、この指定が必要です。また、用紙の左端にとじしろを残したい場合などにも便利です。

➲ ページの余白を設定するには

1. [ファイル->ページレイアウトの設定...] を選んでください。[ページ設定]ダイアログボックスが表示されます（図4.6）。
2. [余白] フィールドに必要な余白を入力してください（ [インチ] または [ミリメートル] ラジオボタンを選んでください）。
3. [OK] ボタンをクリックすると余白が設定されます。

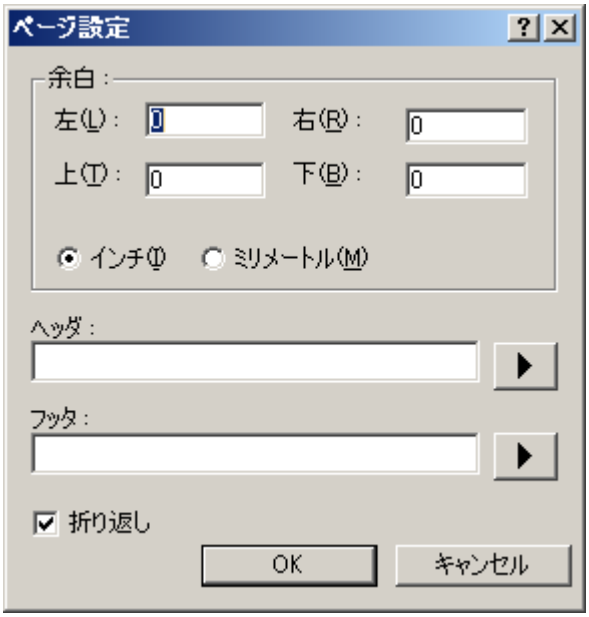


図 4.6: **ページ設定 ダイアログボックス**

➲ ページのヘッダとフッタを設定するには

1. [ファイル->ページレイアウトの設定...] を選んでください。[ページ設定]ダイアログボックスが表示されます（図4.6）。
2. ヘッダおよびフッタの編集フィールドに表示するテキストを入力してください。ページ番号、テキスト配置、日付の各フィールドとともに通常のプレースホルダが利用できます。これらは、ページが印刷される前にすべて拡張されます。
3. [OK] ボタンをクリックすると、新しい設定が有効になります。

➲ 印刷の折り返しを設定するには

1. [ファイル->ページレイアウトの設定...] を選んでください。[ページ設定]ダイアログボックスが表示されます（図4.6）。
2. [折り返し] チェックボックスをクリックしてください。折り返し機能が有効になり、印刷時にテキストが省略されずにすべて表示されます。
3. [OK] ボタンをクリックすると、新しい設定が有効になります。

4.8.2 タブを変更する

➲ タブの文字数を変更するには

1. [ツール->オプション...] を選んでください。[オプション]ダイアログボックスが表示されます。[エディタ]タブを選びます（図4.7）。
2. [タブ幅] フィールドに必要なタブの文字数を指定してください。
3. [OK] ボタンをクリックするとタブの文字数が設定されます。

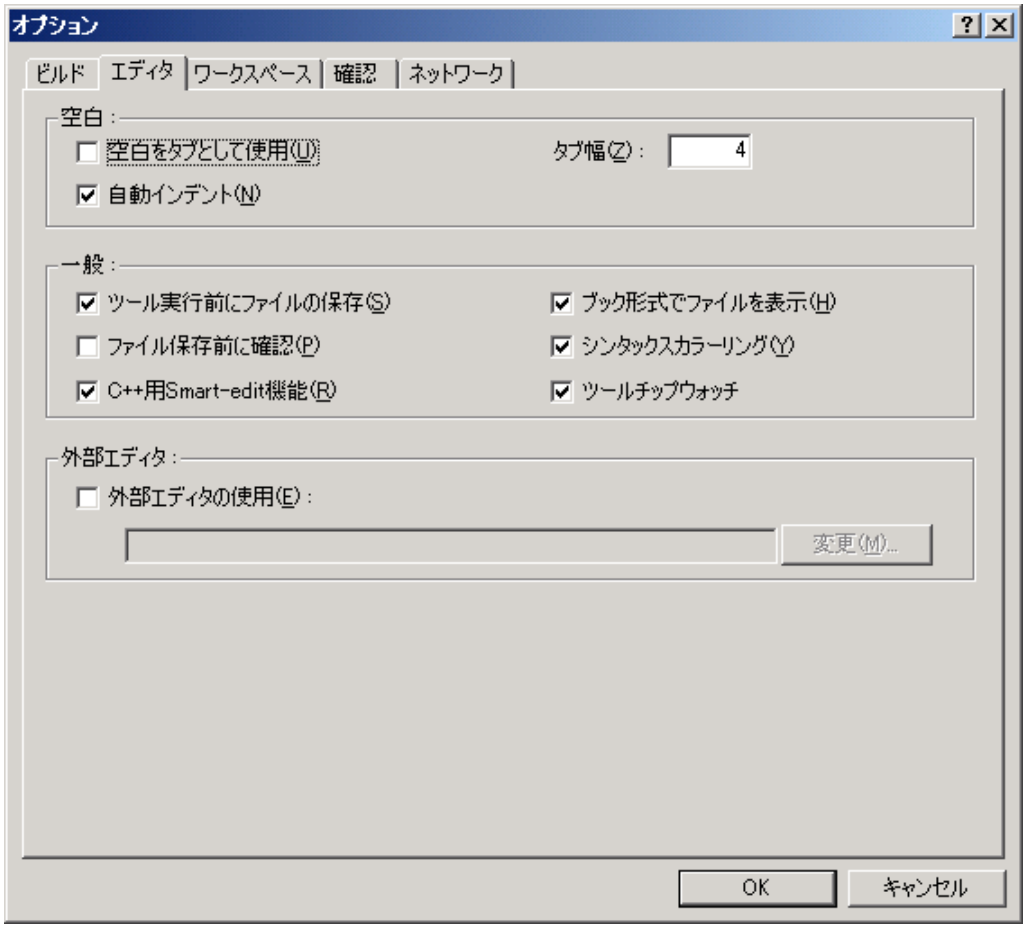


図 4.7: **オプション ダイアログボックス エディタ タブ**

“TAB”キーを押下すると、通常、タブ文字がファイルに入力されます。しかし、タブ文字のかわりに空白文字列を入力したほうがよい場合があります。[オプション]ダイアログボックスの[エディタ]タブで、タブ文字を空白文字列として指定できます。

➲ タブ文字の代わりに空白文字列を使用するには

1. [ツール->オプション...] を選んでください。[オプション]ダイアログボックスが表示されます。[エディタ] タブを選んでください（図4.7）。

2. タブ文字の代わりに空白文字列を使用する場合は [空白をタブとして使用] チェックボックスをチェックしてください。
3. [OK] ボタンをクリックすると設定内容が有効になります。

4.8.3 自動インデント

標準のエディタでリターンキーを押下すると、カーソルは次の行の左端に移ります。自動インデントを設定すると、カーソルは、改行時、前行の最初の文字の下に移ります。したがって、自分でタブや空白を入力することなく、C/C++言語またはアセンブリ言語のコードをより速く見やすく入力できます。

図 4.8 に 2 つの例を示します。 (i) は自動インデントの設定をしなかったときにリターンキーを押した場合の例を示します。カーソルは次の行の左端に移ります。int z=20 の行は、上の 2 行とそろっていません。(ii)は自動インデントを設定してリターンキーを押下した場合の例を示します。カーソルは前行の i の下に移ります。したがって、int z=20 行を入力すると、テキストは自動的に前の行と整列（つまり、自動インデント）します。

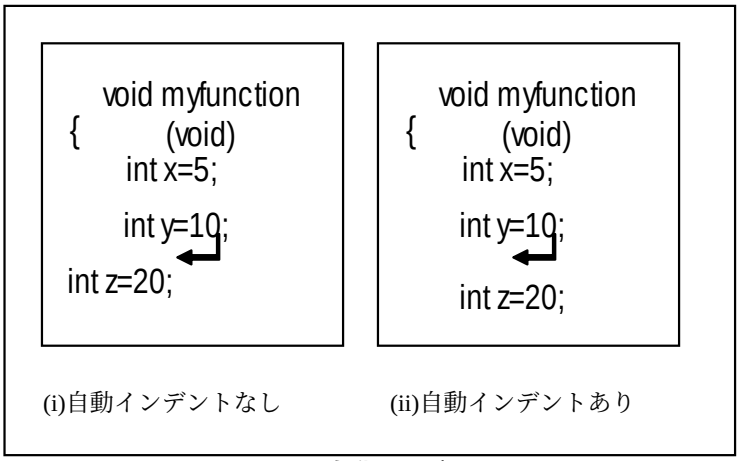


図 4.8: **自動インデント**

➲ 自動インデントを設定するには

1. [ツール->オプション...] を選んでください。[オプション]ダイアログボックスが表示されます。[エディタ]タブを選んでください（図4.7）。
2. [自動インデント] チェックボックスをチェックしてください。
3. [OK] ボタンをクリックすると、自動インデントが設定されます。

4.9 ウィンドウを分割する

HEW では、テキストウィンドウを 2 つに分割できます。図 4.9 に、テキストウィンドウの右上にある“閉じる”ボタンのすぐ下にある スプリットバーボタンを示します。

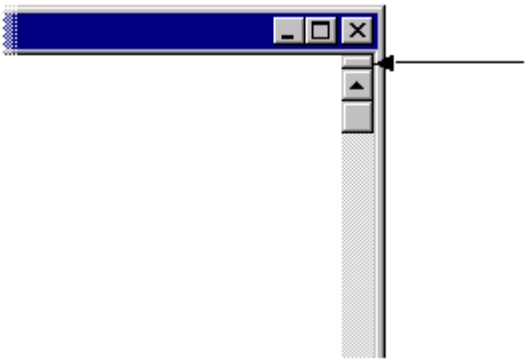


図 4.9: **スプリットバー ボタン**

➲ ウィンドウを分割するには
スプリットバーボタンをダブルクリックしてウィンドウを二つに分割するか、スプリットバーボタンを押したままマウスを下に移動して分割したい位置でマウスボタンを離してください。

➲ スプリットバーの位置を調節するには
スプリットバーボタンを押したままマウスを下に移動して、分割したい位置でマウスボタンを離してください。

➲ ウィンドウの分割を解除するには
スプリットバーボタンをダブルクリックするか、スプリットバーボタンをウィンドウの一番上か一番下に移動してください。

4.10　テキストの表示の変更方法

この節では、エディタウィンドウのテキスト表示の変更方法を説明します。

4.10.1　エディタのフォントを変更する

HEW ではエディタのフォントを指定できます。ファイルの種類にかかわらず、エディタウィンドウはすべて同じフォントを使用します。

➲ エディタのフォントを変えるには

1. [ツール->表示形式...] を選んでください。[表示形式]ダイアログボックスが表示されます。ツリーの中の[Source]アイコンを選んでください（図4.10）。
2. [フォント] リストからフォントの種類を選んでください。
3. [サイズ] リストからフォントの大きさを選んでください。
4. [OK] ボタンをクリックするとフォントの種類と大きさが設定されます。

図 4.10: **表示形式 ダイアログボックス フォント タブ**

4.11 シンタックスを色づけする

コードをより読みやすくするため、指定した文字列（キーワードなど）を異なる色で表示できます。例えば、C ソースコードのコメントを緑色で、C 言語の型（int など）を青色で表示できます。

色づけの方法は、ファイルグループ単位で指定できます。例えば、C ソースファイル、テキストファイル、マップファイル、自分のファイルでそれぞれ異なった色づけ方法を定義できます。

➲ 既存の色を変えるには

1. [ツール->表示形式...] を選んでください。[表示形式]ダイアログボックスが表示されます。
2. 色の変更をしたいアイテムを、ツリーの中のアイコンの下から選んでください。このアイテムはファイルタイプ（例：C言語のソースファイル）および適切なキーワードグループ（例：識別子またはプリプロセッサ）となります。
3. [カラー] タブを選んでください。
4. [前面] リストと [背景] リストの色を変更してください。[SYSTEM]を選択するとコントロールパネルで現在設定してある文字色と背景色になります。
5. [OK] ボタンをクリックすると新しい設定になります。

➲ 新規のキーワードグループを作るには

1. [ツール->表示形式...] を選んでください。[表示形式]ダイアログボックスが表示されます。
2. キーワードグループを追加したいファイルタイプをツリーから選んでください。
3. ツリー下の [追加…] をクリックすると [カテゴリ追加] ダイアログボックスが表示されます（図4.11）。[カテゴリ名]フィールドにキーワードグループを入力し、[OK]ボタンを押下すると新しいキーワードグループが追加されます。

図 4.11: **カテゴリ追加ダイアログボックス**

➲ 新規のキーワードを作るには

1. [ツール->表示形式...] を選んでください。[表示形式]ダイアログボックスが表示されます。
2. シンタックスのハイライト表示を変更したいアイテムを、ツリーの中のアイコンの下から選んでください。このアイテムはファイルタイプ（例：C言語のソースファイル）および適切なキーワードグループ（例：識別子またはプリプロセッサ）となります。
3. [キーワード] タブを選んでください（図4.12）。

図 4.12: **表示形式ダイアログボックス　キーワードタブ**

4. キーワードを追加するには [追加...] ボタンをクリックしてください。すると[キーワード追加]ダイアログボックスが表示されます（図4.13）。[キーワード]フィールドにキーワードを指定し[OK]ボタンを押してダイアログを閉じてください。キーワードを削除するにはキーワードを選択して[削除]ボタンをクリックしてください。

図 4.13: **キーワード追加ダイアログボックス**

新規ファイルを作成すると、デフォルトではファイルの拡張子がないためシンタックスの色付けは行いません。（エディタが自動的に新規ファイルに付ける名前には拡張子がありません）。新規ファイルにシンタックスの色付けをするには、上記の拡張子をもつ名前でファイルを保存してください。

➲ シンタックスの色付けを有効／無効にするには

1. [ツール->オプション...] を選んでください。[オプション]ダイアログボックスが表示されます。[エディタ]タブを選んでください。（図4.7）。
2. [シンタックスカラーリング] チェックボックスをチェックすると有効になり、チェックしないと無効になります。設定後、[OK]ボタンをクリックしてください。

4.12 テンプレート

ソフトウェア開発時、同じテキストを繰り返し入力する場合がよくあります。例えば、関数定義、ループ、関数の機能のコメント欄などです。HEW では、現在アクティブなエディタウィンドウに、定型テキスト（テンプレート）を挿入できます。テンプレート設定後は定型テキストを、手作業で入力するかわりに、簡単に挿入できるようになります。

4.12.1 テンプレートを設定する

➲ テンプレートを設定するには

1. [編集->テンプレート->テンプレートの定義...] を選ぶか、ローカルメニューから [テンプレート->テンプレートの定義...] を選ぶか、 [テンプレートの定義] ツールバーボタン ( ) を選んでください。図4.14 に示すダイアログボックスが表示されます。

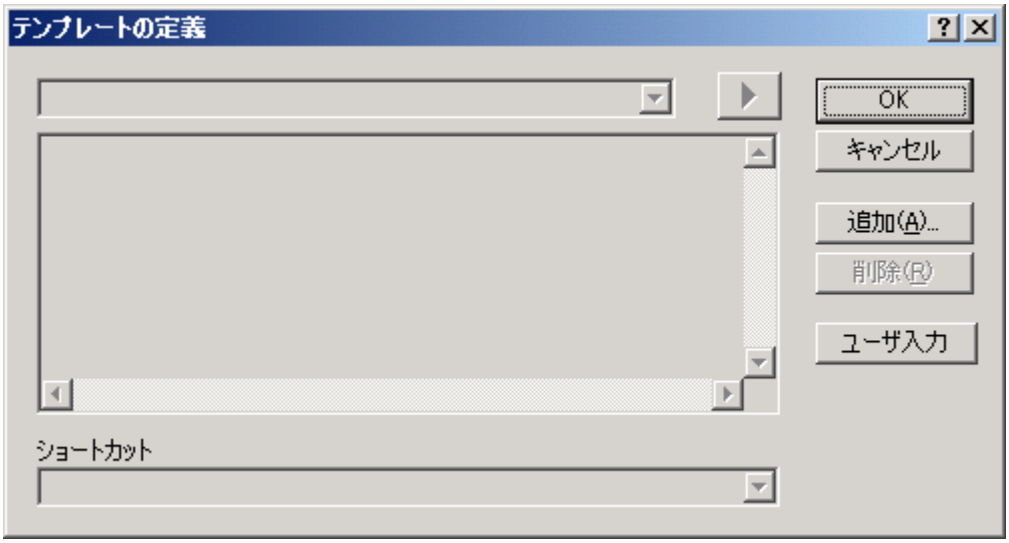


図 4.14: **テンプレートの定義 ダイアログボックス**

2. [追加…] ボタンをクリックしてください。ダイアログボックスが表示され、選択したテンプレート名を尋ねます。この場合、独自のテンプレート名を選択してください。そうしないと、複製したテンプレート名のメッセージが表示され、テンプレートは追加されません。
3. 既存のテンプレートを変更したいときは、 [テンプレート名] ドロップダウンメニューから変更するテンプレートを選んでください。
4. テンプレートテキストエリアにテキストを入力してください。他のエディタウィンドウからテキストをコピーして”CTRL+V”キーを押下してこのダイアログボックスに貼りつけることができます。
5. テンプレート用に確保されているショートカットキーは10個あります。このうち1つを使いたい場合、[テンプレートの編集]ダイアログ下部にあるドロップリストでキーを選択してください。CTRL+0 ～CTRL+9の中から選択できます。
6. テンプレートを設定するとき、次のプレースホルダを使うことができます。

表 4.2: テンプレートで使用するプレースホルダ

メニューエントリ	プレースホルダテキスト	説明
ファイルパス+ファイル名	$(FULLFILE)	ファイル名（フルパスを含む）
ファイル名	$(FILENAME)	ファイル名（パスを除き拡張子を含む）
ファイルリーフ	$(FILELEAF)	ファイル名（パスと拡張子を除く）
ワークスペース名	$(WORKSPNAME)	ワークスペース名
プロジェクト名	$(PROJECTNAME)	プロジェクト名
行番号	$(LINE)	テンプレートを挿入する最初の行番号
時間	$(TIME)	テンプレートが挿入される時間
日付，テキスト	$(DATE_TEXT)	現在の年月日をテキスト表示
日付，日/月/年	$(DATE_DMY)	現在の日/月/年
日付，月/日/年	$(DATE_MDY)	現在の月/日/年
日付，年/月/日	$(DATE_YMD)	現在の年/月/日
ユーザ名	$(USER)	現在のユーザ
カーソル位置	$(CURSOR)	挿入カーソル：テンプレートを挿入した後カーソルをこの位置に設定

7. ‘$(CURSOR)’ を入力すると、テンプレートが挿入された後のカーソルはこの位置になります。‘$(CURSOR)’ を設定しないと、テンプレートが挿入された後のカーソルはテンプレート最後の文字の後ろになります（通常の貼りつけ操作と同じ）。

ユーザ入力

テンプレートを定義する際、ユーザ入力フィールドも定義することができます。以下のプレースホルダを使って指定します。

$(USERINPUT<n:1-10>|”<*some text*>”)

n はユーザ入力識別子を表す数字です。これらのプレースホルダは手動で追加することも可能ですが、[テンプレートの定義]ダイアログの[ユーザ入力]ボタンを使うとプレースホルダを自動的に追加できます。テンプレートをファイルに挿入するとダイアログが表示され、ユーザは各フィールドにカスタムのテキストを入力することができます。その後、このテキストはプレースホルダの替わりに挿入されます。ユーザはこのようなユーザ入力フィールドを 10 個定義できます。

図 4.15 にテンプレート一覧を示します。この一覧は[Workspace]ウィンドウの[Templates]タブにあります。

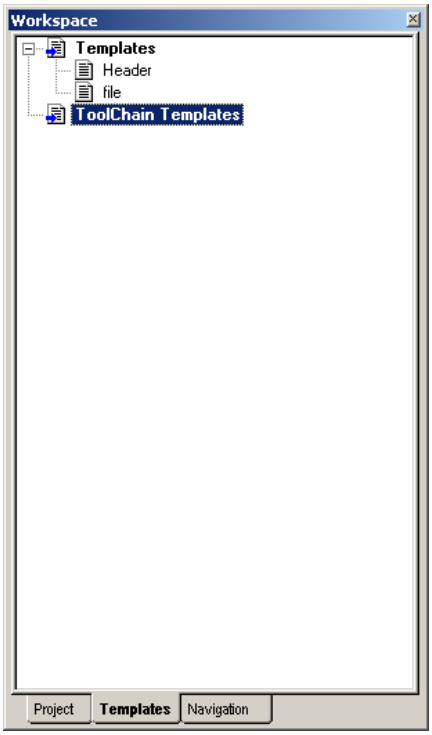


図 4.15: テンプレート一覧

HEW に追加された新しいテンプレートは、[Templates]フォルダの下に表示されます。[Toolchain Templates]フォルダには、HEW システムで使用するため現在のツールチェインによって提供される読み取り専用のテンプレートが表示されます。この一覧にあるテンプレートはドラッグしてエディタファイルに挿入できます。さらに、エディタからテキスト領域を[Templates]フォルダにドラッグして、簡単にテンプレートを作成することも可能です。この一覧でマウスの右ボタンをクリックするとポップアップメニューが表示され、ユーザは簡単に新しいテンプレートを追加したり、現在選択しているテンプレートを削除または編集したりすることができます。

4.12.2 テンプレートを削除する

➲ テンプレートを削除するには

1. [編集->テンプレート->テンプレートの定義...] を選ぶか、ローカルメニューから [テンプレート->テンプレートの定義...] を選ぶか、[テンプレートの定義] ツールバーボタン（  ）をクリックしてください。図4.14 に示すダイアログボックスが表示されます。
2. [テンプレート名] ドロップダウンリストから削除したいテンプレート名を選び、 [削除] ボタンをクリックしてください。
3. 確認ダイアログボックスが表示された場合は、[はい]ボタンをクリックしてください。
4. [OK]ボタンをクリックすると、ダイアログボックスは終了されます。

4.12.3 テンプレートを挿入する

➲ テンプレートを挿入するには
ツールバーでテンプレートを選択し“テンプレートの挿入”ツールバーボタン()をクリックするか、 [編集->テンプレート->テンプレートの挿入...]を選ぶか、ローカルメニューから [テンプレート->テンプレートの挿入...]を選んでください。選んだテンプレートが現在のエディタウィンドウに追加されます。

4.12.4 かっこの組み合わせ

複雑なソースコードは扱いにくいことがあります。特に、C 言語のブロックが互いに深いネスト構造になっている場合や、if 文で複雑な論理文が表現されている場合などです。HEW では、そのような場合のために、かっこの種類{ }, ( ) , [ ]ごとにかっこの間のテキストをハイライト表示できます。

➲ かっこの組み合わせを見つけるには

1. かっこの始めをハイライト表示するか、カーソルをかっこの前に置いてください。
2. [括弧の呼応] ツールバーボタン（  ） をクリックするか、 “Shift+CTRL+M”キーを押下するか、 [編集->括弧の呼応] を選ぶか、 ローカルメニューから [括弧の呼応] を選んでください。

ファイル全体の構造をチェックするために、カーソルをファイルの始めに置いて、かっこの組み合わせの操作を繰り返し行ってください。組み合わせがなくなるまで、かっこの組み合わせごとに次々とハイライト表示されます。

4.13 Editor カラムの管理

Editor のカラムは、デバッガの機能に応じて追加します。また、ユーザがカラムを表示、非表示を選択することが可能です。

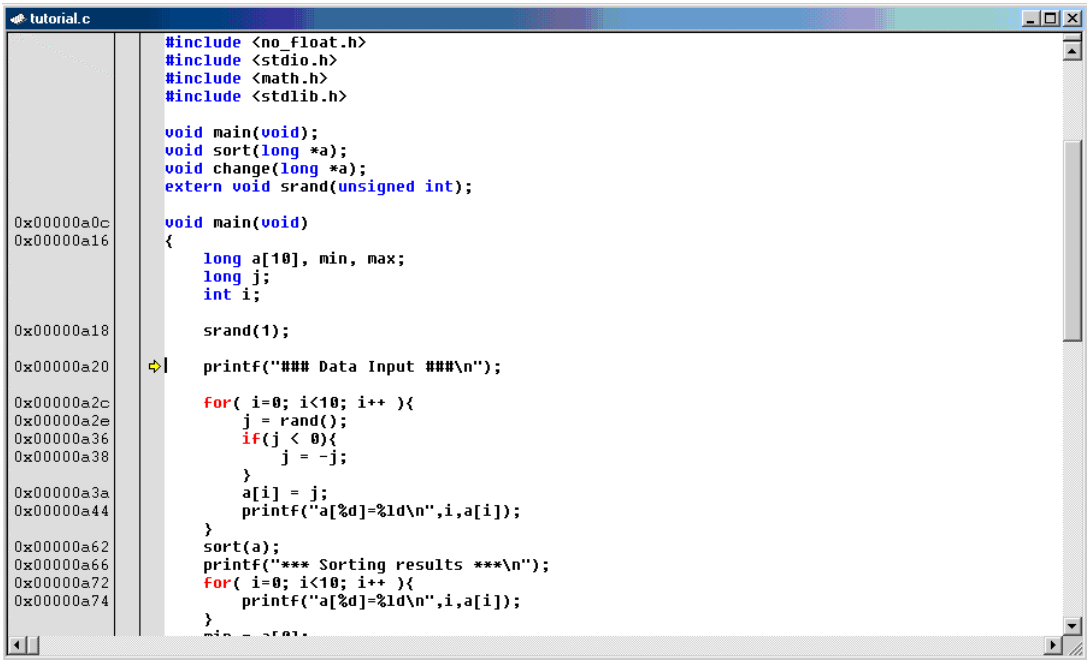


図 4.16: Editor **カラム**

➲すべてのソースファイルでカラムをオフにするには

1. [エディタ] ウィンドウを右クリックしてください。
2. [表示カラムの設定...] メニュー項目をクリックしてください。
3. [エディタ全体のカラム状態] ダイアログボックスが表示されます（図4.17）。
4. チェックボックスは、そのカラムが有効か無効かを示します。チェックされている場合は有効です。チェックボックスがグレー表示の場合、一部のファイルではカラムが有効で、別のファイルでは無効であることを意味します。
5. [OK] ボタンをクリックして、新しいカラム設定を有効にしてください。

➲1つのソースファイルでカラムをオフにするには

1. 削除したいカラムのある [エディタ] ウィンドウを右クリックしてください。
2. [カラム] メニュー項目をクリックしてください。カスケードされたメニュー項目が現れます。各カラムは、このポップアップメニューに表示されます。カラムが有効である場合、名前の横にチェックマークがあります。エントリをクリックすると、カラムの表示、非表示を切り替えます。

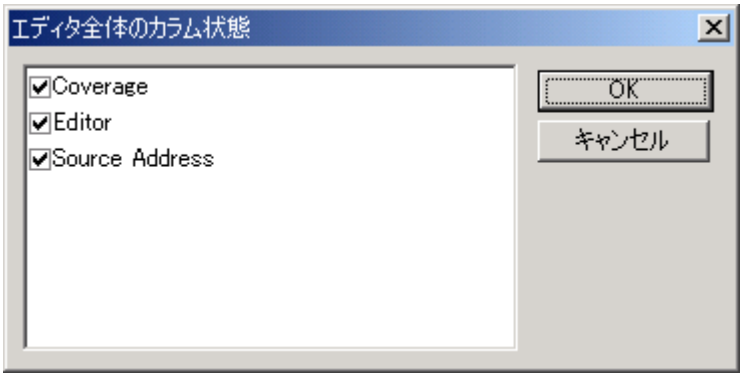


図 4.17: **エディタ全体のカラム状態ダイアログボックス**

4.14　ツールチップウォッチ

[ツール->オプション...]の[エディタ]タブを選んでください(図 4.7)。[ツールチップウォッチ]チェックボックスをチェックすると、ソースプログラム上の変数の値を早く確認するツールチップウォッチ機能を使用することができます。

➲ [エディタ]ウィンドウにツールチップウォッチを表示するには

1. 確認したい変数を含むソースファイルを[エディタ]ウィンドウに表示します（図4.18）。
2. 確認したい変数名の上にマウスのカーソルを静止させます。変数の近くに変数の値を表示したツールチップを表示します。

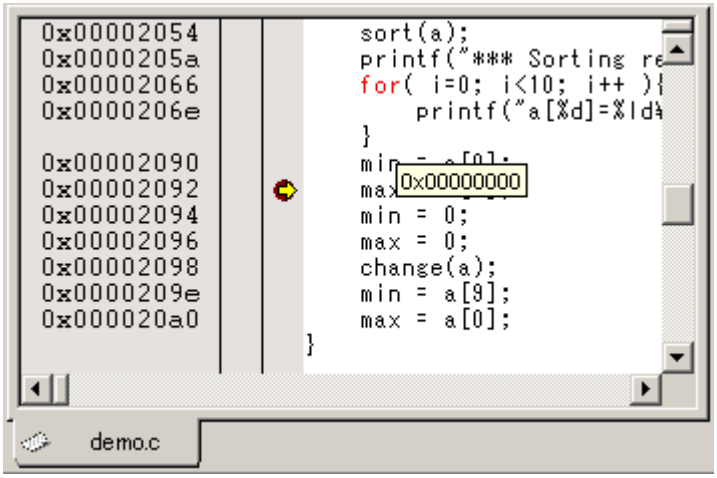


図 4.18: **ツールチップウォッチの表示**

5. ツール管理

[ツールアドミニストレーション]ダイアログボックスで、HEW で使うツールを管理します (図 5.1)。このダイアログボックスは[ツール-> アドミニストレーション…]で開きます。すべてのワークスペースが閉じているときは変更可能で、ワークスペースが開いているときは参照のみ可能です。

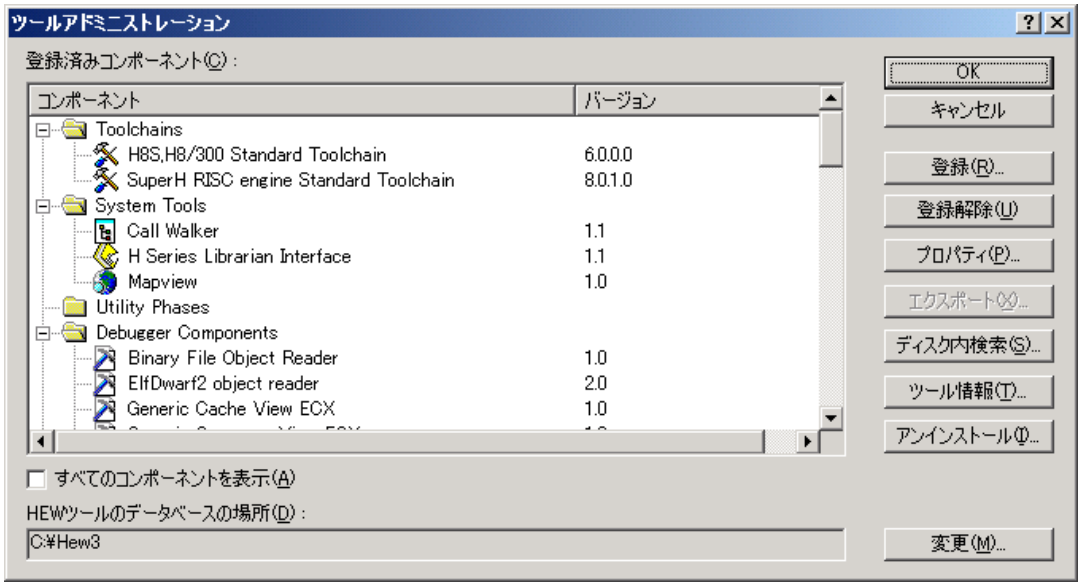


図 5.1: ツールアドミニストレーション ダイアログボックス

5 種類の標準ツールがあります。

Toolchains： 一連のビルドフェーズ （例:コンパイラ、アセンブラ、リンケージエディタ）。 ビルド機能を実現。

System Tools： [ツール]メニューから選ぶことのできるアプリケーション （.EXE）。ツールチェインをサポートする追加のアプリケーション（例: HDI などの外部デバッガまたは対話式グラフィカルライブラリアン）。

Utility Phases： 特定のビルド機能をサポートする、あらかじめ用意されたビルドフェーズ （例: ソースコードの複雑度解析、ソースコードの行カウントなど）。特定のツールチェインに依存しない追加のビルド機能。

Debugger Components： 特定のデバッガ機能をサポートするツール（例: ターゲットプラットフォーム、オブジェクトリーダなど）。

Extension Components： HEW システムのある領域における特定のキー機能をサポートするツールです。これらのツールはインストールすると、必ず登録されます（例: HEW ビルダ、デバッガ、フラッシュサポート）。

5.1 ツールの位置

HEW では、新しいツールがインストールされるたびに HEW との連動に必要なツールの位置を自動的に保持します。インストール後、HEW はそのツールに関する情報（位置を含む）を保持します。これを登録と呼びます。初期登録は自動で行いますが、開発の途中で、プロジェクトのツールをより効率良く利用するためにユーザ自身でツールを登録することが必要になることがあります。この章では登録について説明します。

5.2 HEW 登録ファイル (*.HRF)

HEW と互換性のあるツール（ツールチェイン、システムツール、またはユーティリティフェーズ）をインストールすると、拡張子**.HRF**（図 5.2 i）のファイルもインストールされます。この拡張子は“**H**EW **R**egistration **F**ile”の略であり、.HRF ファイルには HEW への登録に必要な情報が記述されています。登録するには、そのツールの.HRF ファイルを[ツールアドミニストレーション]ダイアログボックスにロードします（図 5.2 ii）。

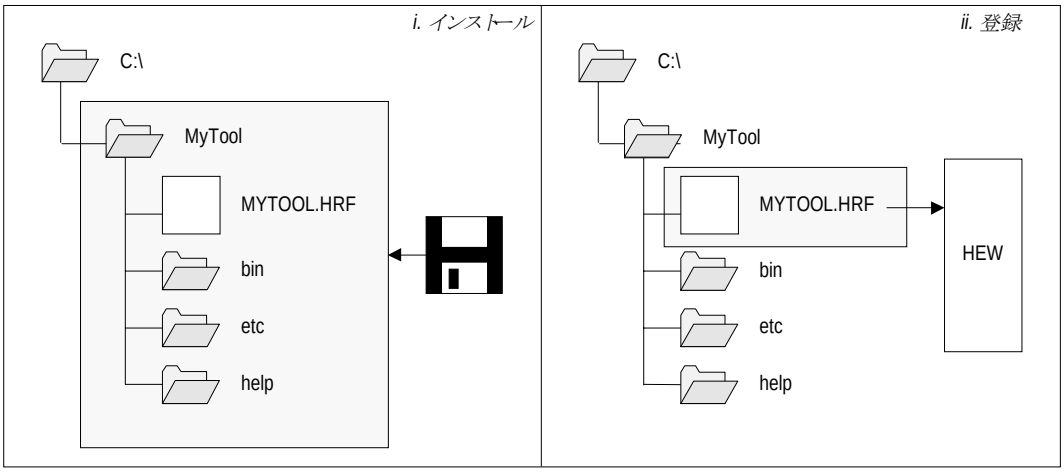


図 5.2: HRF **ファイルの位置と登録**

HEW でツールを使うには、まず登録が必要です。[ツールアドミニストレーション]ダイアログボックス（図 5.1）は現在登録されているツールを表示します。このダイアログボックスを開くには、ワークスペースがすべて閉じていることを確認して[ツール-> アドミニストレーション…]を選んでください。ワークスペースを開いた状態で[ツールアドミニストレーション]ダイアログボックスをアクセスすると、ダイアログが開きますが変更はできません。HEW がデフォルトでインストールされると、新しいツールは自動的に登録されます。

HEW はツールデータベースファイルに、ツール情報を格納します。デフォルトでは、このファイルは HEW アプリケーションディレクトリに作成されます。しかし、ネットワーク環境で作業を行っている場合は、このディレクトリは他の場所に設定されることがあります。ツールディレクトリの場所は変更が可能です。

➲ ツールの場所を変更するには

1. [ツール-> アドミニストレーション…] を選んでください。
2. [HEWツールのデータベースの場所] の [変更…] ボタンをクリックしてください。
3. 新しいツールがあるディレクトリのルートディレクトリを選択し、 [OK] ボタンをクリックしてください。
4. ディレクトリが切り替わり、ツールの場所が新しいディレクトリに移ります。この場所にある新しいツールはスキャンする必要があります。スキャンには、スキャンディスクまたはレジスタツール機能を使用します。

5.3 ツールを登録する

HEW は起動後にインストールしたすべてのツールを自動的に登録します。しかし、ときには、ユーザがツールを登録する必要があります。

5.3.1 ドライブのツール検索

ドライブを検索して HEW に互換性のあるツールを見つけることは、HEW のツールインストール情報が削除されたり破壊されたりしたときなどに有益です。なぜなら、ツール情報を再びすぐに作成することができるからです。

➲ ツールを検索するには

1. [ツールアドミニストレーション] ダイアログボックス（図5.1）の [ディスク内検索…] ボタンをクリックすると [コンポーネントのディスク内検索] ダイアログボックスが表示されます（図5.3）。

図 5.3: **コンポーネントのディスク内検索 ダイアログボックス**

2. [どのディレクトリを検索するか選択してください] フィールドに、検索するディレクトリを入力してください。または、 [参照…] ボタンをクリックしてディレクトリを選んでください。
3. [サブフォルダを含む] チェックボックスをチェックすると、指定したディレクトリとその下のディレクトリをすべて検索します。
4. [開始] ボタンをクリックすると検索を始めます。検索中、 [開始] ボタンは [中断] ボタンに変わります。検索を途中で止めるときには [中断] ボタンをクリックしてください。
5. [コンポーネント] リストに検索結果を表示します。個別にツールを登録するにはそのツールを選んで [登録] ボタンをクリックしてください。すべてのツールを登録するには [すべてを登録] ボタンをクリックしてください。
6. [閉じる] ボタンをクリックするとダイアログボックスを終了します。

5.3.2 ツールを一つ登録する

HEW では、ツールを検索しなくても、ツールを一つずつ登録できます。HEW 登録ファイル (*.HRF) はツールがインストールされたルートディレクトリにあります。

➲ ツールを登録するには

1. [登録…] ボタンをクリックすると標準のファイルを開くダイアログボックスが開きます。フィルタが [HEW Registration Files (*.hrf)] に設定されています。
2. 登録するファイルの.HRFファイルをアクセスして選び、 [選択] ボタンをクリックしてください。
3. 選んだツールに関する情報を示すダイアログボックスが表示されます。ツールを登録するには [登録] ボタンをクリックしてください。登録しない場合は [閉じる] ボタンをクリックしてください。

5.4 ツールの登録を取り消す

登録したツールによって、HEW は影響を受けます。例えば、新しいプロジェクト作成時に、登録された互換性のあるすべてのシステムツールが、ツールメニューに追加されます。ときにはユーザにとっては、これによって効率が下がり、使いにくいかもしれません。そのようなときは、登録を取り消すことができます。[ツールアドミニストレーション]ダイアログボックスでツールを選び [登録解除] ボタンをクリックしてください。確認のダイアログボックスが表示されます。 登録を取り消す場合は[はい] をクリックしてください。

注意 ツールの登録を取り消しても、ハードディスクからツールがなくなることはありません。単に、HEW に格納されているそのツールに関する情報を削除するだけです（HEW から切断されます）。この動作はいつでも元に戻すことができ、ツールを再登録できます（前節「ツールを登録する」参照）。ハードディスクからツールを削除（アンインストール）したいときはこの章後半の「ツールのアンインストール」を参照してください。

5.5 ツールのプロパティの参照と編集

ツールに関する情報を参照するには、ツールを [登録済みコンポーネント] リストから選んで [プロパティ…] ボタンをクリックしてください。[一般] タブを選んだ状態で （図 5.4）プロパティのダイアログボックスが表示されます。このタブでは、名前、バージョン、位置を示します。このタブは編集できません。

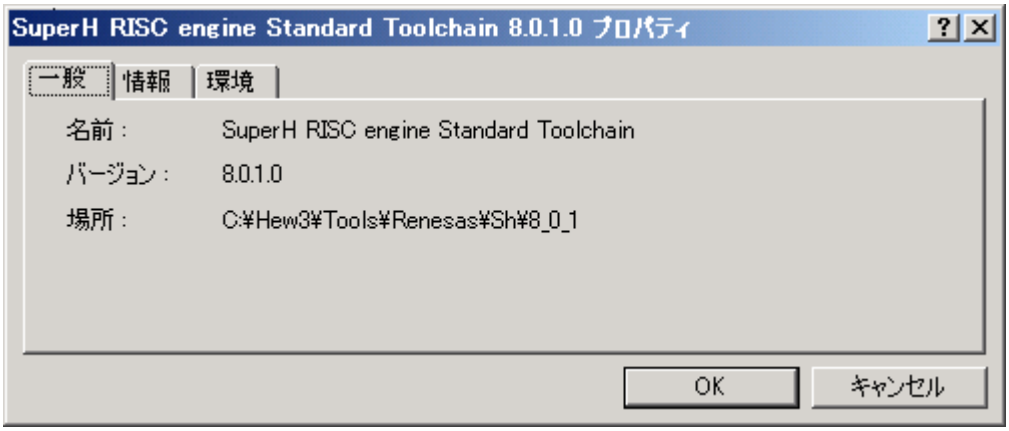


図 5.4: プロパティのダイアログボックス 一般 タブ

ツールについての情報を参照するには [情報] タブをクリックしてください（図 5.5）。例えば、著作権、履歴、バグの修正、ユーザへのお知らせなどが表示されます。

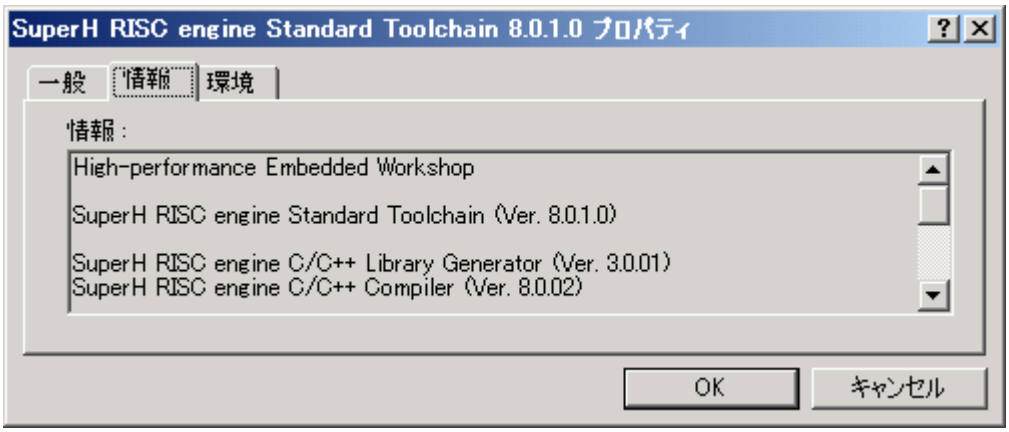


図 5.5: プロパティのダイアログボックス 情報 タブ

ツールの互換性に問題がある場合、この[情報]エディットフィールドで問題を報告することがあります。

ツールの環境設定を参照、編集するには [環境] タブを選んでください（図 5.6）。このダイアログボックスは、ツールチェインの環境を変更するとき、もっとも一般的に使われています。

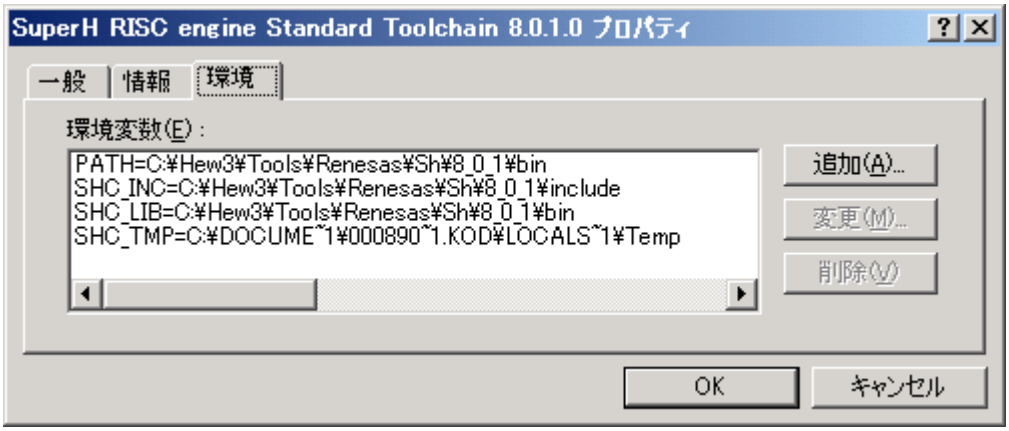


図 5.6: **プロパティのダイアログボックス 環境 タブ**

新しい環境変数を追加するには、[追加...]ボタンをクリックしてください。図 5.7 に示すダイアログボックスが表示されます。[変数]フィールドに変数名を入力して[値]フィールドに変数の値を入力して [OK]ボタンをクリックしてください。環境変数の値を柔軟に指定できるようにするため、プレースホルダポップアップメニューがあります。プレースホルダの詳細は、付録 C 「プレースホルダ」を参照してください。

環境変数を変更するには、[環境]タブで変更する環境変数を選び[変更...]ボタンをクリックしてください。[変数] フィールドと [値]フィールドを必要に応じて変更して [OK]ボタン をクリックすると、変更した環境変数が[環境]タブに加わります。環境変数を削除するには、その環境変数を選び [削除] ボタンをクリックしてください。

図 5.7: **環境変数 ダイアログボックス**

5.6 ツールのアンインストール

HEW には登録されていないツールをハードディスクから削除するための、アンインストール方法があります。

➲ ツールをアンインストールするには

1. [ツール->アドミニストレーション…] を選んでください。
2. [アンインストール] ボタンをクリックしてください。[HEWツールのアンインストール]ダイアログボックスが表示されます(図5.8)。

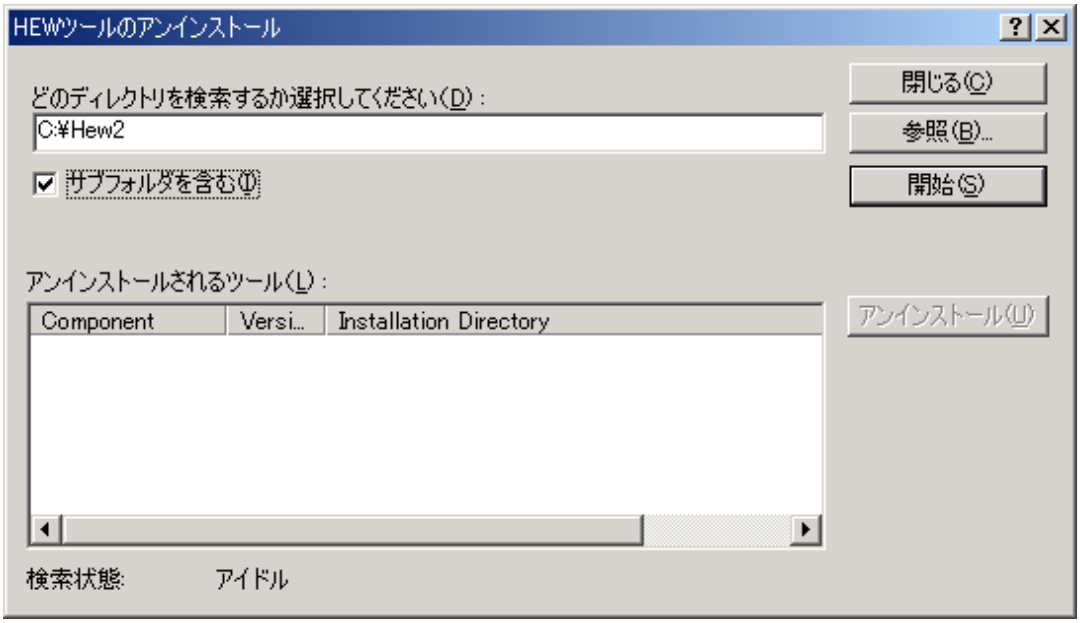


図 5.8: HEW **ツールのアンインストールダイアログボックス**

3. 一番上のフィールドに検索するディレクトリを入力するか、 [参照…] ボタンをクリックしてブラウズしてください。
4. [サブフォルダを含む] チェックボックスをチェックすると、指定したディレクトリの下のディレクトリをすべて検索します。
5. [開始] ボタンをクリックすると検索を始めます。検索中、 [開始] ボタンは [中断] ボタンに変わります。 [中断] ボタンをクリックすると、検索の途中でも検索を中止します。
6. 検索結果は [アンインストールされるツール] リストに表示されます。ツールを選んで [アンインストール] ボタンをクリックするとツールをアンインストールします。
7. [閉じる] ボタンをクリックしてダイアログボックスを終了してください。

現在 HEW で登録されていないツールのみアンインストールできます。登録されているツールをアンインストールしようとすると、図 5.9 に示すダイアログボックスが表示されます。このようなときは、[ツール->アドミニストレーション…]で HEW の[ツールアドミニストレーション]ダイアログボックスに戻り、ツールの登録を取り消してから、アンインストールを再実行してください。

図 5.9:**アンインストール不可のダイアログボックス**

HEW から登録をはずしたツールを選んで[登録解除] ボタンをクリックすると、図 5.10 に示す確認のダイアログボックスが表示されます。このダイアログボックスには削除されるすべてのファイルやフォルダが表示されます。これらのファイルやフォルダを削除してよいことを確認して [はい] ボタンをクリックしてください。アンインストールを中止するときは、 [いいえ] または [キャンセル] ボタンをクリックしてください。

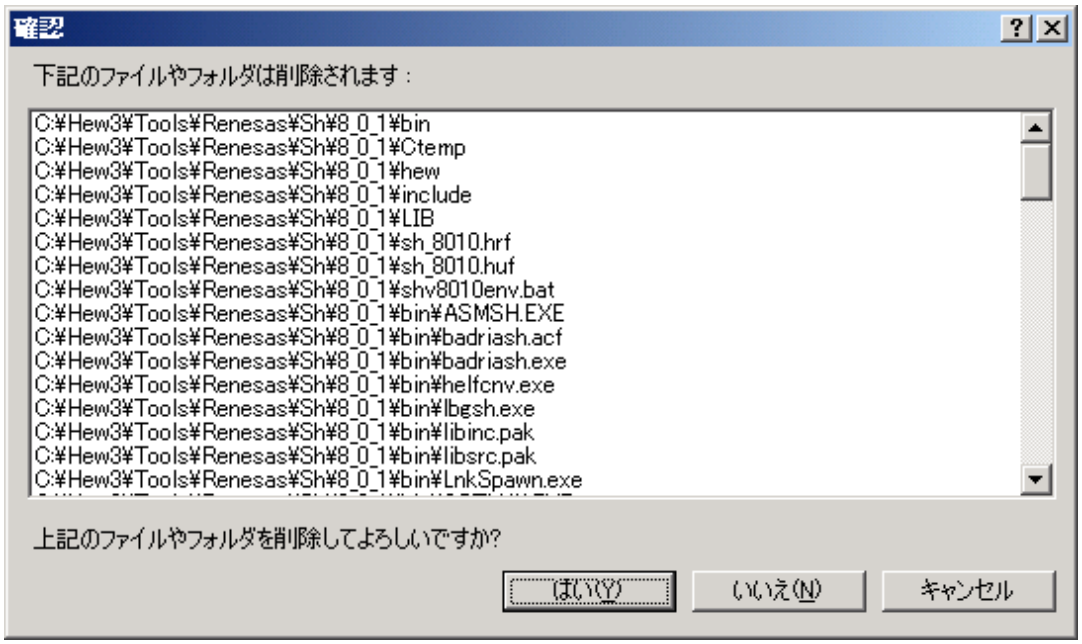


図 5.10: **確認ダイアログボックス**

5.7 テクニカルサポートについて

[ツールアドミニストレーション] ダイアログボックスでは、“隠れている”システムツールに関する情報を表示できます。これらのツールは HEW の一部であり、手動で登録または登録取り消しすることができません。[ツールアドミニストレーション]ダイアログボックスの [すべてのコンポーネントを表示]チェックボックスをチェックすると、隠れていたツールフォルダを表示します （図 5.11）。

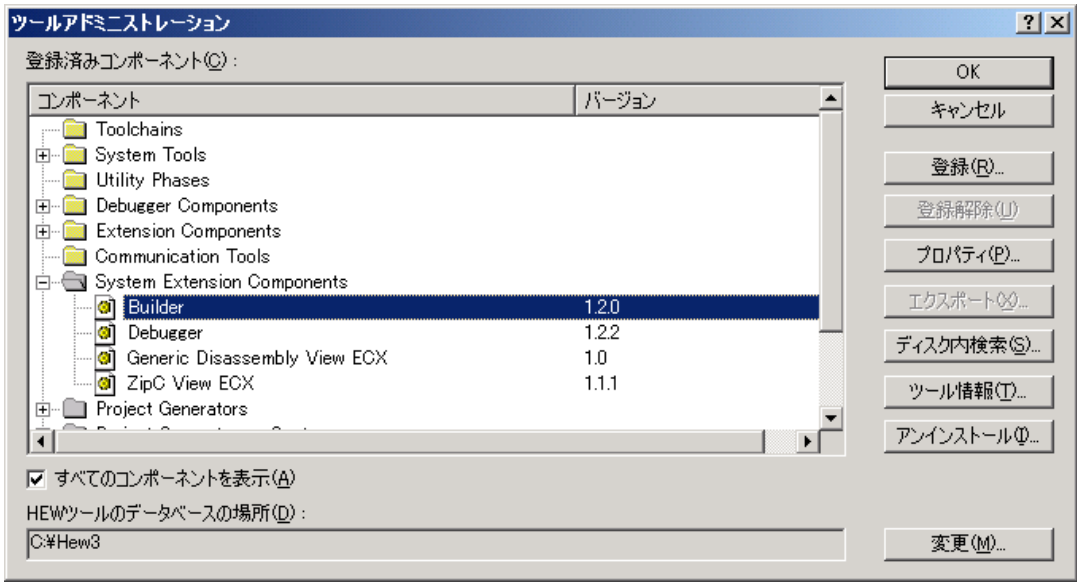


図 5.11: すべてのツールの表示

テクニカルサポートを受ける際、ツールに関する詳細をお尋ねすることがあります。そのときには、そのツールのフォルダを開き、ツールを選び、 [プロパティ] ボタンをクリックしてください。ここで表示されるダイアログボックスはこの章の前半で説明したものと同じように動作します。ただし、[環境]タブはありません。

HEW にはまた、登録されたツールに関する情報をファイルに出力する機能があるので、HEW システム全体の情報を得ることができます。もし HEW で問題がありましたら、テクニカルサポート担当宛にこの情報を送付してください。

➲ ツール情報を出力するには

1. [ツール->アドミニストレーション] を選んでください。
2. [ツール情報] ボタンをクリックすると、 [ツール情報ファイルの保存] ダイアログボックスが表示されます。
3. ファイルの場所を選び、 [保存] ボタンをクリックしてください。
4. HEWに現在登録されているツールの情報を選んだ場所にファイル出力します。

また、問題のあるコンポーネントは、[ツールアドミニストレーション]ダイアログボックスで確認できます。一つは登録済みコンポーネントが見つからない場合で、図 5.12 のアイコンを表示します。もう一つは、コンポーネントは見つかったが、古いバージョンであったり他の依存するコンポーネントが使用不可であったりして使用できない場合で、図 5.13 のアイコンを表示します。

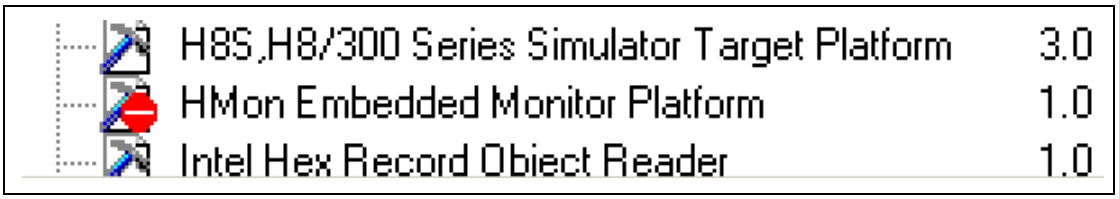


図 5.12: [Component not found] **アイコン**

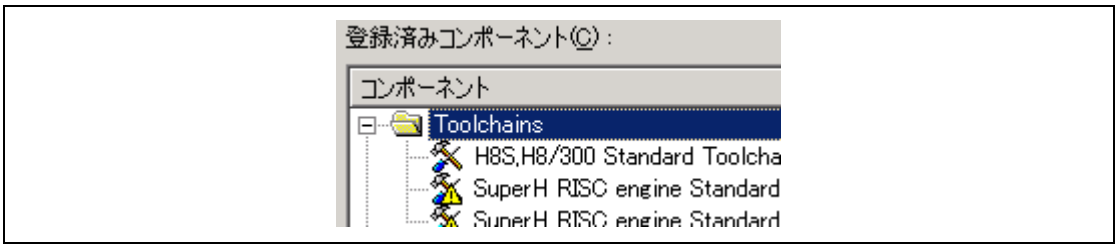


図 5.13: [Incompatible component found] **アイコン**

注意　ツールにいずれかのエラーがある場合、以下の方法でさらに情報を得ることができます。

➲ ツールエラーのフィードバックを取得するには

1. [ツール->アドミニストレーション] を選んでください。
2. 問題のあるツールをリストから選択してください。
3. [プロパティ] ボタンをクリックしてください。
4. [情報]タブを選択し、エディットフィールドを下までスクロールしてください。
5. この領域に問題の原因が表示されます。

5.8 オンデマンドのコンポーネント

HEW version 3.0 以降にはオンデマンドのコンポーネントという概念があります。これらのコンポーネントはアプリケーションやデバッガコンポーネントが自動的にロードするわけではありません。プロジェクト生成過程の一部として、ユーザがロードするものです。

➲ オンデマンドのコンポーネントを手動でロードまたはアンロードするには

1. [プロジェクト->コンポーネント...] を選んでください。
2. [コンポーネントギャラリー] ダイアログボックスが表示されます。これを図5.14に示します。
3. ロードしたいコンポーネントを選択して [ロード] ボタンをクリックしてください。コンポーネントのイメージはロードされた状態に変わります。
4. コンポーネントをアンロードしたい場合は、コンポーネントを選択して [アンロード] ボタンをクリックしてください。コンポーネントのイメージはアンロードの状態に変わります。
5. [OK] ボタンをクリックし、変更を確認してください。

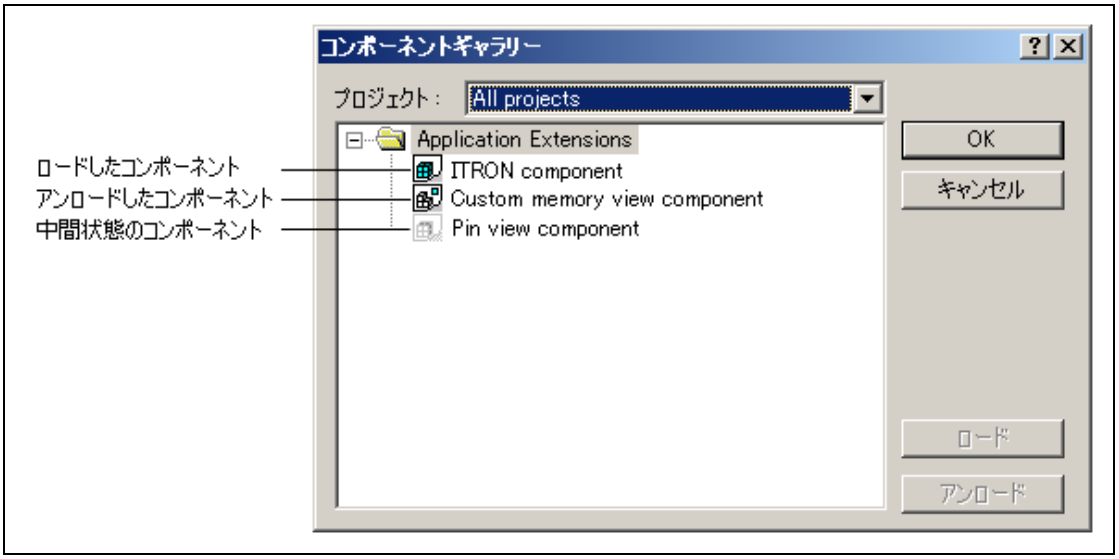


図 5.14: **[コンポーネントギャラリー] ダイアログボックス**

注意 **ユーザのワークスペースにある各プロジェクトには、ロード／アンロードされた異なるコンポーネントを格納できます。複数のプロジェクトがある場合は、[複数のプロジェクト]および[すべてのプロジェクト]を使うことで、複数のプロジェクトにわたってコンポーネントのロード状態を変更することができます。コンポーネントがあるプロジェクトでロードされ、他のプロジェクトではロードされない状態をユーザが選択した場合は、[中間状態]アイコンが表示されます。**

5.9 カスタムプロジェクトタイプ

HEW の [プロジェクト->プロジェクトタイプの作成...] メニュー機能で、現在のプロジェクトの設定を利用してプロジェクトのテンプレートを作成することができます。このテンプレートをカスタムプロジェクトジェネレータと呼びます。新しいプロジェクトタイプの名称とプロジェクト生成ウィザードのスタイルを指定できます。作成したカスタムプロジェクトジェネレータは、[ツールアドミニストレーション] ダイアログボックスで表示できます。カスタムプロジェクトジェネレータを他のユーザが使用するには、[ツールアドミニストレーション] ダイアログボックスで該当するカスタムプロジェクトジェネレータを選択して、[エクスポート...] ボタンをクリックしてください。カスタムプロジェクトジェネレータの実行環境がインストール可能な実行ファイルにまとまります。このファイルを目的とする他のマシンで実行すれば、カスタムプロジェクトジェネレータがインストールされます。

一度カスタムプロジェクトタイプを作成すると、その後[新規プロジェクトワークスペース]ダイアログボックスに表示されます。これを使用してプロジェクトの複製を作成することができます。

➲ カスタムプロジェクトジェネレータを作成するには

1. カスタムプロジェクトジェネレータに変換したいプロジェクトを含むワークスペースを開いてください。
2. 現在のプロジェクトとして変換したいプロジェクトを設定してください。
3. [プロジェクト->プロジェクトタイプの作成...] を選んでください。
4. 以下のダイアログボックスが表示されます。

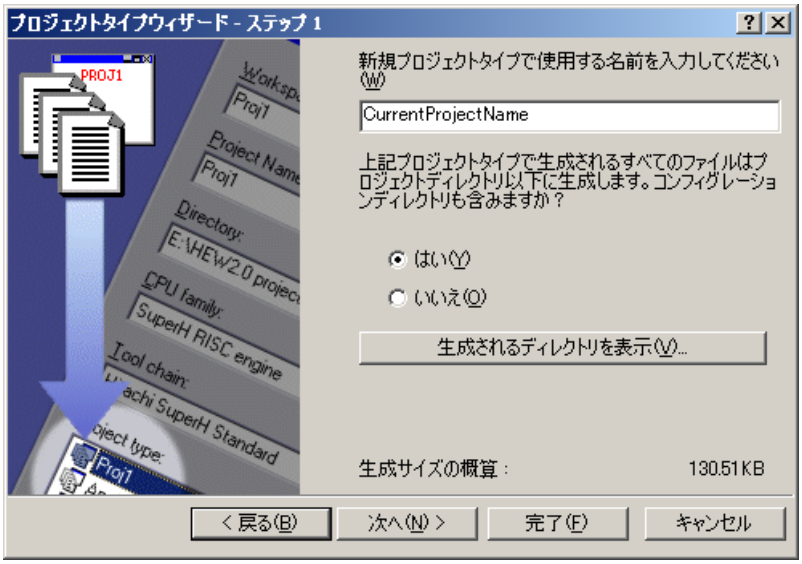


図 5.15: [プロジェクトタイプウィザード - ステップ 1] ダイアログボックス

5. 編集フィールドに現在のプロジェクト名が設定されます。これが、[新規プロジェクトワークスペース]ダイアログボックスに表示される、新しいプロジェクトジェネレータのプロジェクトタイプ名になります。必要に応じて変更してください。
6. コンフィグレーションディレクトリを含むかどうか選択することができます。コンフィグレーションディレクトリ内のファイルは単なる中間ファイルであるため、ここでは含めない

のが一般的です。

7. [次へ] ボタンをクリックしてください。以下のダイアログボックスが表示されます。

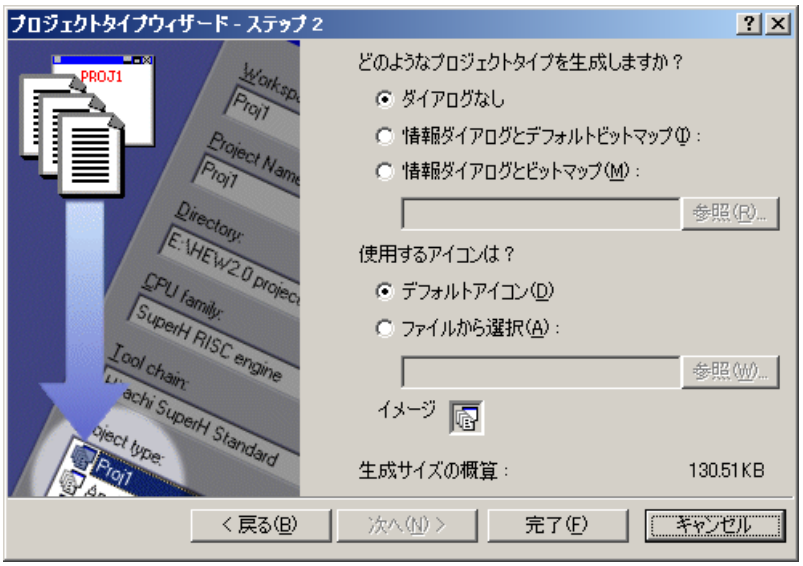


図 5.16: [**プロジェクトタイプウィザード - ステップ** 2] **ダイアログボックス**

8. ウィザードのグラフィックとプロジェクトタイプのアイコンを変更することができます。[ダイアログなし]を選ぶと、 [Project information] ダイアログボックス（図5.19参照）は表示しません。 [Project information] ダイアログボックスを表示する場合は、 [情報ダイアログとデフォルトビットマップ]と[情報ダイアログとビットマップ]のいずれかを選んでください。[Project information] ダイアログボックス内のグラフィックにデフォルトのグラフィックを使用するか、カスタムのグラフィックを使用するかを選択できます。
9. プロジェクトタイプのアイコンを変更することもできます。デフォルトのアイコンが表示されていますが、カスタムのアイコンを使用することができます。これは [新規プロジェクトワークスペース] ダイアログボックスに表示されます。
10. ここで [Project information] ダイアログボックスを表示しない場合は、 [完了] ボタンをクリックしてください。表示する場合は、 [次へ] ボタンをクリックしてください。次のダイアログボックスが表示されます（図5.17）。

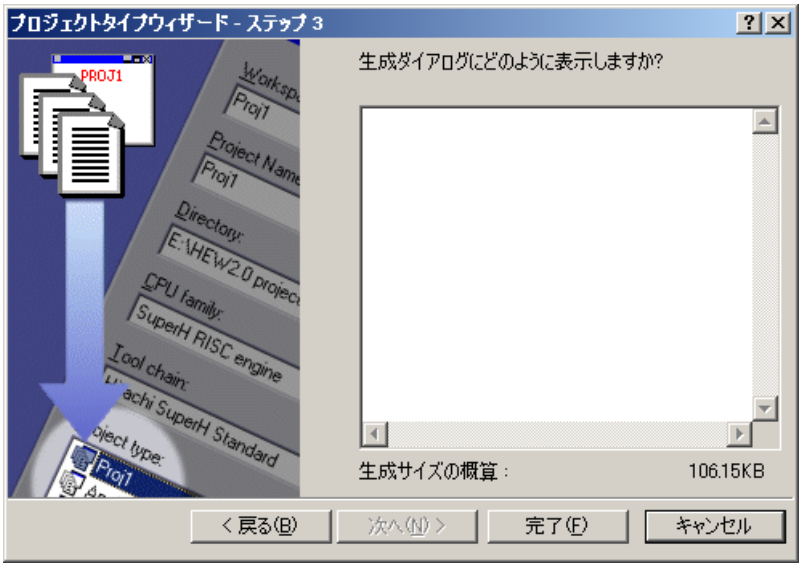


図 5.17: [**プロジェクトタイプウィザード - ステップ** 3] **ダイアログボックス**

11. プロジェクト情報を入力してください。 [Project information] ダイアログボックスの [プロジェクト情報] に表示されます。
12. [完了] ボタンをクリックしてください。

HEW インストールディレクトリ下の¥system¥pgc ディレクトリにカスタムプロジェクトジェネレータが作成されます。

カスタムプロジェクトジェネレータを作成すると、新しいプロジェクトタイプ（ここでは“ CurrentProjectName ”）が [新規プロジェクトワークスペース] ダイアログボックスに表示されます。

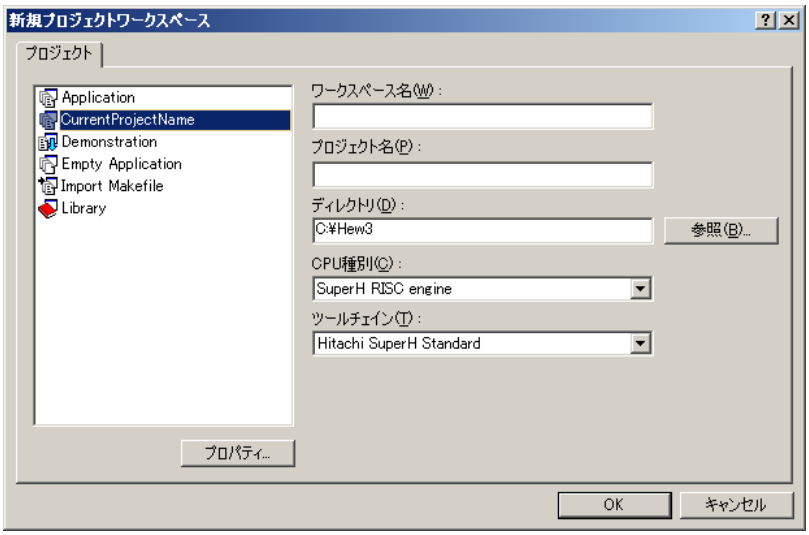


図 5.18: **[新規プロジェクトワークスペース] ダイアログボックス**

プロジェクトタイプ、ワークスペース名、プロジェクト名を設定し、 [OK] ボタンをクリックしてください。

[Project information] ダイアログボックスは、[プロジェクトタイプウィザード - ステップ 2] ダイアログボックスと[プロジェクトタイプウィザード - ステップ 3] ダイアログボックスで設定した内容を表示します。

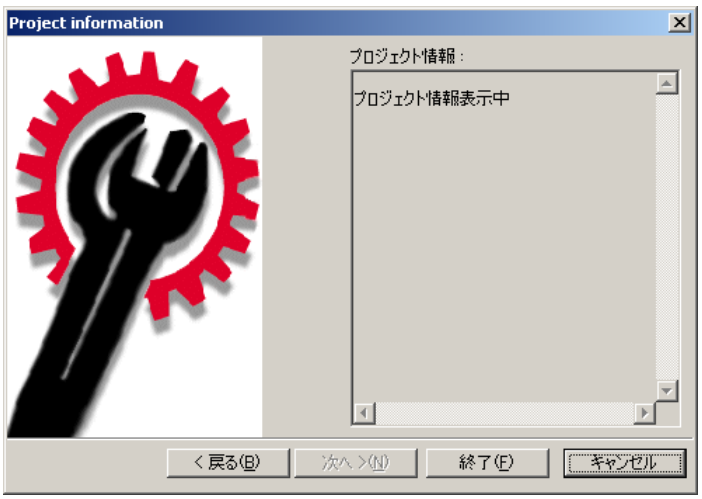


図 5.19: [Project information] **ダイアログボックス**

複製したプロジェクトを HEW システムからエクスポートし、そのプロジェクトを使用する他のユーザに渡すことができます。

➲ カスタムプロジェクトジェネレータを他のユーザにエクスポートするには

1. [ツール->アドミニストレーション] を選んでください。
   [すべてのコンポーネントを表示] チェックボックスをチェックしてください。
2. [Project Generators – Custom] フォルダを拡張してください。
3. エクスポートするカスタムプロジェクトジェネレータを選択してください。
4. [エクスポート…] ボタンをクリックしてください。

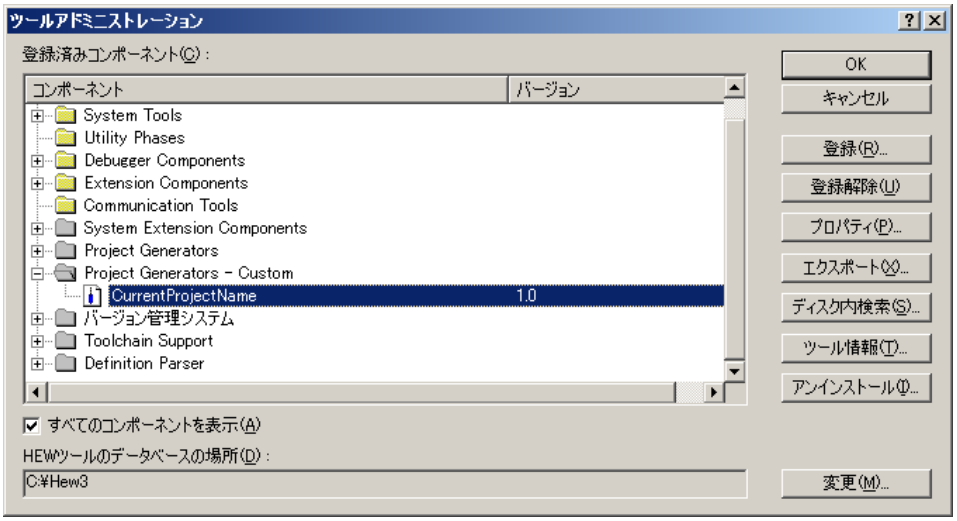


図 5.20: [ツールアドミニストレーション] ダイアログボックス

5. 表示されるダイアログボックスで、新しいカスタムプロジェクトジェネレータのエクスポート先のディレクトリを指定し、 [エクスポート] ボタンをクリックしてください。他の HEW 環境でカスタムプロジェクトジェネレータを使用するために必要なファイルセットを作成します。他のマシン環境へエクスポートする場合は、このダイアログボックスで指定したディレクトリ下のすべてのファイルが必要になります。

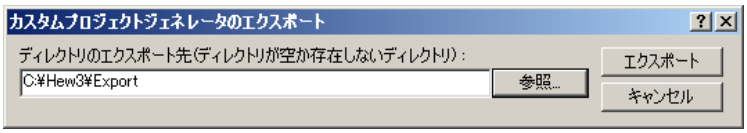


図 5.21: [カスタムプロジェクトジェネレータのエクスポート] ダイアログボックス

他の HEW 環境へカスタムプロジェクトジェネレータをエクスポートするには、エクスポートしたディレクトリ下にある“Setup.exe”を実行し、新しいユーザの HEW の場所を指定します。 [Install] ボタンをクリックすると、HEW インストールディレクトリ下の¥system¥pgc ディレクトリにカスタムプロジェクトジェネレータがインストールされます。

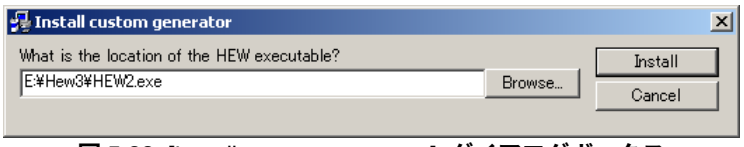


図 5.22: [Install custom generator] **ダイアログボックス**

エクスポートしたプロジェクトタイプが [新規プロジェクトワークスペース] ダイアログボックスに表示されます。

注意 **カスタムプロジェクトジェネレータは、同等もしくはそれ以上のバージョンの** HEW **でしか使用できません。古い** HEW **でこのプロジェクトジェネレータを使用すると、エラーメッセージが表示されます。**

使用しているプロジェクトジェネレータがエクスポート先の HEW **への異なるツールチェインを含んでいる場合、プロジェクトを最初に作成したときにアップグレードされます。**

6. 環境のカスタマイズ

6.1 ツールバーのカスタマイズ

HEW では 2 つのツールバーを標準で提供します。また、[カスタマイズ] ダイアログボックス（図 6.1）を使用して、新しいツールバーを作成することができます。

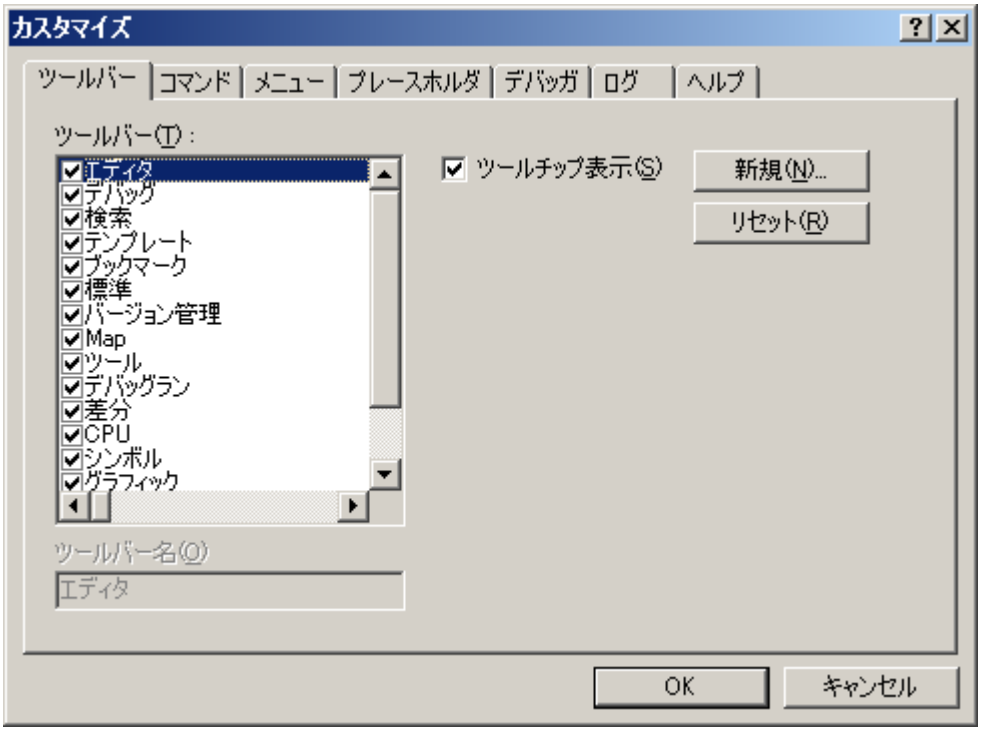


図 6.1: **カスタマイズ ダイアログボックス ツールバー タブ**

➲ 新しいツールバーを作るには

1. [ツール->カスタマイズ…] を選んでください。図6.1に示すダイアログボックスが表示されます。
2. [新規...] ボタンをクリックしてください。図6.2に示すダイアログボックスが表示されます。
3. [ツールバー名] フィールドに新しいツールバー名を入力してください。
4. [OK] ボタンをクリックすると新しいツールバーが作成されます。

図 6.2: **新規ツールバー ダイアログボックス**

作成した新しいツールバーにはボタンがなく、位置を固定せずに表示されます。

➲ ツールバーにボタンを追加するには

1. [ツール->カスタマイズ…] を選んでください。図6.1に示すダイアログボックスが表示されます。 [コマンド] タブを選んでください（図6.3参照）。
2. [カテゴリ] リストからボタンのカテゴリーを選び、使用できるボタンを参照してください。[ボタン]エリアからボタンを選ぶとそのボタンの機能が表示されます。
3. ボタンをクリックしてダイアログボックスからツールバーにドラッグしてください。

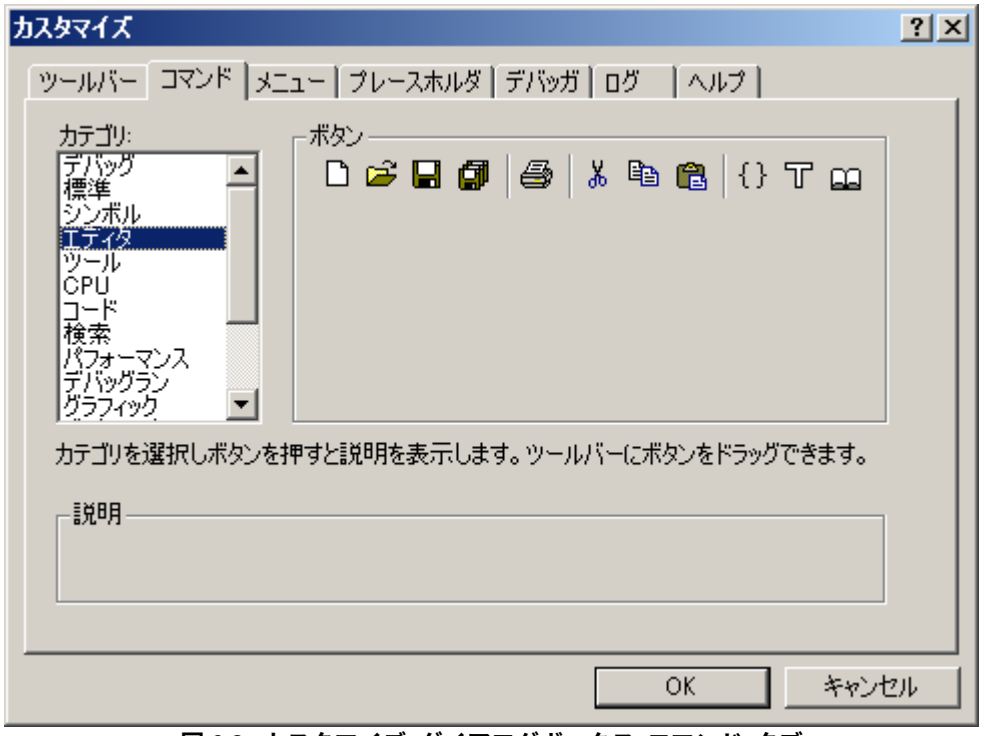


図 6.3: **カスタマイズ ダイアログボックス コマンド タブ**

➲ ツールバーからボタンを削除するには

1. [ツール->カスタマイズ…] を選んでください。図6.1に示すダイアログボックスが表示されます。 [コマンド] タブ （図6.3参照）を選んでください。
2. ボタンをツールバーから [ボタン] エリアへドラッグしてください。

➲ ユーザ定義のツールバーを削除するには

1. [ツール->カスタマイズ…] を選んでください。図6.1に示すダイアログボックスが表示されます。
2. [ツールバー] リストからユーザ定義のツールバーを選ぶと図6.1の [リセット] ボタンが [削除] ボタンに変化します。[削除] ボタンをクリックしてください。

➲ 標準のツールバーを初期状態に戻すには

1. [ツール->カスタマイズ…] を選んでください。図6.1に示すダイアログボックスが表示されます。
2. [ツールバー] リストから標準のツールバーを選んで [リセット] ボタンをクリックしてください。

➲ ツールバーのツールチップを表示/非表示するには

1. [ツール->カスタマイズ…] を選んでください。図6.1に示すダイアログボックスが表示されます。
2. [ツールチップ表示] チェックボックスをチェックすると表示、チェックしないと非表示になります。

➲ ユーザが作成したツールバーのツールバー名を変えるには

1. [ツール->カスタマイズ…] を選んでください。図6.1に示すダイアログボックスが表示されます。
2. [ツールバー] リストでユーザが作成したツールバーで名前を変更したいものを選んでください。
3. [ツールバー名] フィールドでツールバー名を変更してください。

6.2 ツールメニューのカスタマイズ

ツールメニューをカスタマイズして新しいメニューオプションを含めることができます。

➲ 新しいメニューオプションを追加するには

1. [ツール->カスタマイズ…] を選んでください。図6.1に示すダイアログボックスが表示されます。 [メニュー] タブを選んでください（図6.4）。最初に、全ワークスペースに適用できるグローバルアプリケーションワイドツールを追加する（[アプリケーション内有効:] に追加）か、または現在のワークスペースのみに適用できるワークスペースワイドツールを追加する（[ワークスペース内有効:] に追加）かどうかを決めてください。一度決定したら、ダイアログボックスの該当部分を選ぶようにしてください。

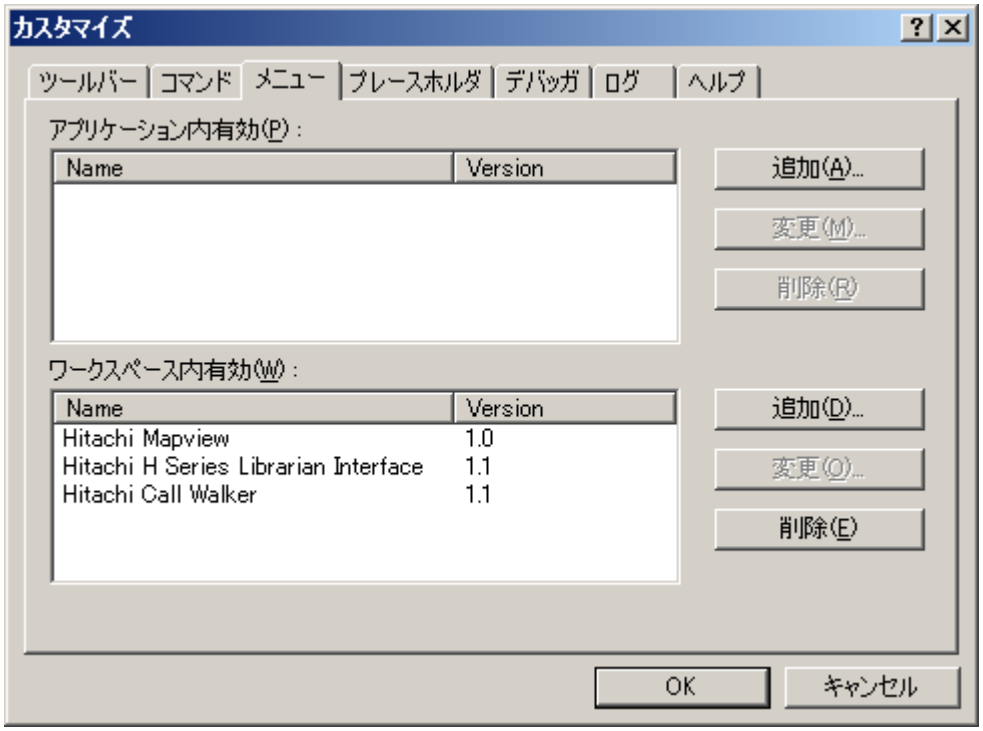


図 6.4: **カスタマイズ ダイアログボックス メニュー タブ**

2. [追加…] ボタンをクリックしてください。図6.5に示すダイアログボックスが表示されます。既存のシステムツールをメニューに追加するには、 [既存ツールからの選択] ラジオボタンを選び、ドロップダウンリストからシステムツールを選び、 [OK]ボタンをクリックしてください。
3. [名前] フィールドにツール名を入力してください。
4. [コマンド] フィールドにコマンドを入力してください。ただし、コマンドに渡す引数は入力しないでください。
5. [引数] フィールドにコマンドに渡す引数を入力してください。
6. [初期ディレクトリ] フィールドにツールを実行する初期ディレクトリを入力してください。
7. [OK] ボタンをクリックするとメニューオプションが [ツール] メニューに追加されます。

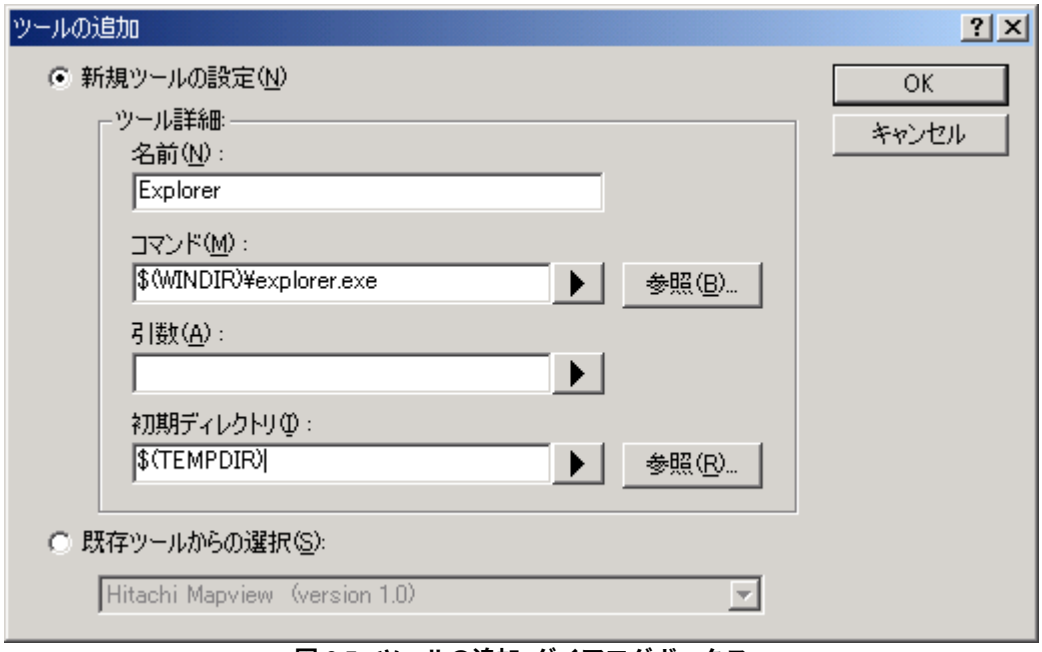


図 6.5: **ツールの追加 ダイアログボックス**

新しいメニューオプションはリストの最後に追加されます（ツールメニューの一番下）。

➲ メニューオプションを変更するには

1. [ツール->カスタマイズ...] を選んでください。図6.1に示すダイアログボックスが表示されます。[メニュー]タブ（図6.4参照）を選んでください。
2. 変更するメニューオプションを選んで [変更...] ボタンをクリックしてください。
3. [ツールの変更] ダイアログボックス（図6.6）を変更後、 [OK] ボタンをクリックしてください。

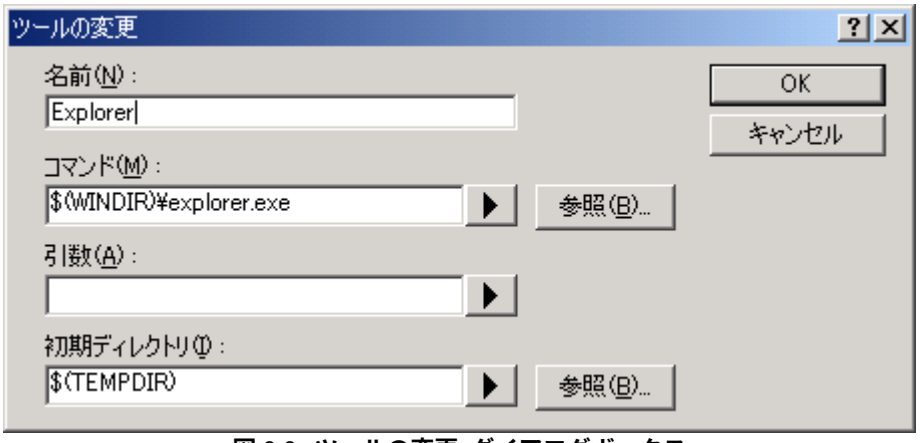


図 6.6: **ツールの変更 ダイアログボックス**

➲ メニューオプションを削除するには

1. [ツール->カスタマイズ…] を選んでください。図6.1に示すダイアログボックスが表示されます。[メニュー]タブ（図6.4参照）を選んでください。
2. 削除するメニューオプションを選んで [削除] ボタンをクリックしてください。

6.3 ヘルプシステムを構築する

HEW ではエディタウィンドウでコンテキスト依存ヘルプを提供します。エディタウィンドウでテキストを選び F1 キーを押下すると、選んだテキストに関するヘルプを検索します。検索するヘルプファイルは[カスタマイズ]ダイアログボックスの[ヘルプ]タブに表示されます。

➲ 新しいヘルプファイルを追加するには

1. [ツール->カスタマイズ…] を選んでください。図6.1に示すダイアログボックスが表示されます。 [ヘルプ] タブ （図6.7参照）を選んでください。

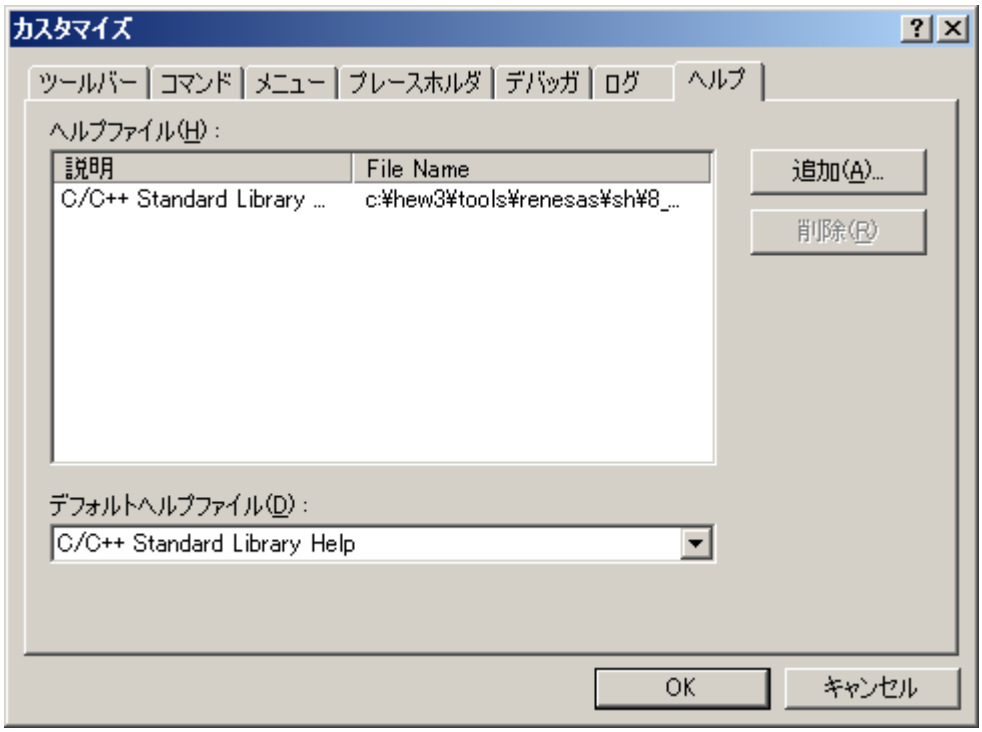


図 6.7: **カスタマイズ ダイアログボックス ヘルプ タブ**

2. [追加…] ボタンをクリックしてください。 [ヘルプファイルの追加]ダイアログボックス（図6.8）が表示されます。
3. [説明] フィールドにヘルプファイルの説明を入力してください。
4. [ファイル名]フィールドにヘルプファイルへのフルパスを入力してください（または[参照…]ボタンをクリックしてファイルを選んでください）。
5. [OK]ボタン をクリックすると新しいヘルプファイルが定義されます。

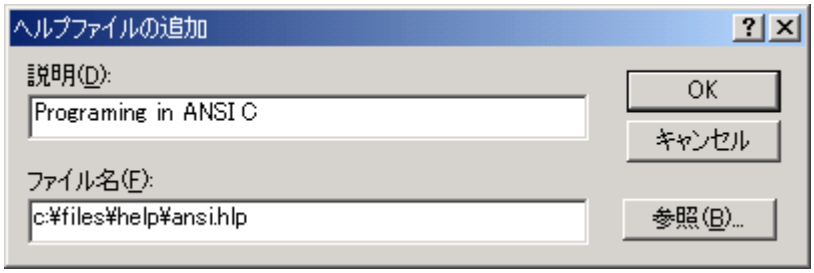


図 6.8: **ヘルプファイルの追加 ダイアログボックス**

デフォルトのヘルプファイルを設定する場合は[デフォルトヘルプファイル] ドロップダウンリストからヘルプファイルを選んでください。F1 キーを押下したときに任意のヘルプファイルを参照するには [(None)] を選んでください。

6.4　ワークスペースオプションを指定する

HEW では[オプション] ダイアログボックス（図 6.9）でワークスペースの様々なオプションを設定することができます。 [ツール->オプション...]の[ワークスペース]タブを選んでください。

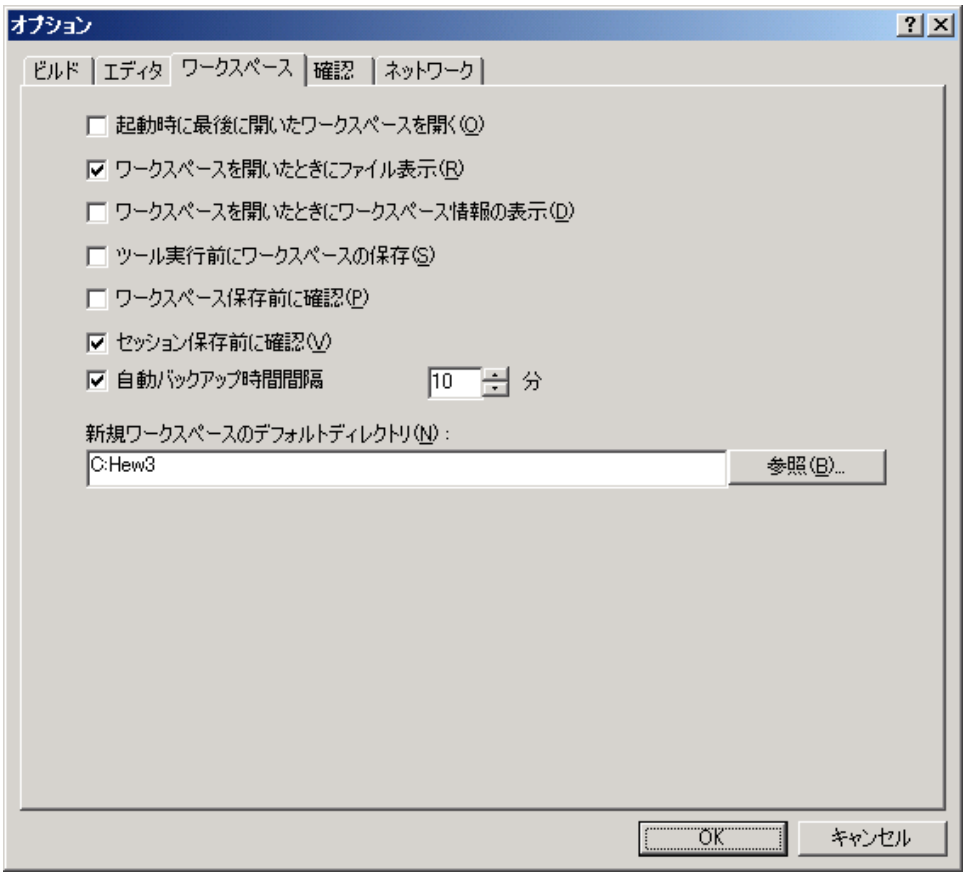


図 6.9: **オプション ダイアログボックス ワークスペース タブ**

次にこのタブで制御できるオプションを説明します。

6.4.1 起動時に最後に開いたワークスペースを開くチェックボックス

このチェックボックスをチェックすると、HEW 起動時に、最後に開いたワークスペースを自動的に開きます。

6.4.2 ワークスペースを開いたときにファイル表示チェックボックス

HEW は、ワークスペースを閉じるとき、開いていたファイルを記憶します。そして、再びワークスペースを開くとき、HEW は同じファイルを復帰させる（つまり開く）ことができます。これにより、中断したセッションを引き続き行うことができます。このチェックボックスをチェックすると、ワークスペースを開いたときに前回開いていたファイルを開きます。

6.4.3 ワークスペースを開いたときにワークスペース情報の表示チェックボックス

多くのワークスペースを使うと、各ワークスペースの内容を正確に覚えておくのは難しくなります。この問題を解決するために、HEW では、各ワークスペースの説明を入力しておくことができ、このチェックボックスをチェックすると、ワークスペースを開くときに表示することができます。

➲ ワークスペースの説明を入力するには

1. [ワークスペース] ウィンドウの [Projects] タブからワークスペースのアイコンを選んでください。
2. マウスの右ボタンをクリックしてポップアップメニューを表示させ、 [プロパティ] オプションを選んでください。図6.10に示すダイアログボックスが表示されます。
3. [情報] フィールドに説明を入力してください。
4. ワークスペースを開いたときにワークスペースプロパティダイアログボックスを開かせたいときは [ワークスペースを開いたときにワークスペース情報の表示] チェックボックスをチェックしてください。このチェックボックスは[オプション]ダイアログボックスの[ワークスペース] タブの[ワークスペースを開いたときにワークスペース情報の表示]チェックボックスと同じ役割を持っています。

図 6.10: ワークスペースプロパティダイアログボックス

HEW では、ワークスペースを開くときにこの説明を表示することができます。したがって、そのワークスペースが目的のワークスペースかどうかを判断することができます。この説明を表示するには、[ワークスペースを開いたときにワークスペース情報の表示]チェックボックスをチェックしてください。

6.4.4　ツール実行前にワークスペースの保存チェックボックス

このチェックボックスをチェックすると、ビルドフェーズを実行（[ビルド]、[すべてビルド]、[コンパイル]操作）する前や、バージョン管理コマンドを実行する前に、現在のワークスペースを保存します。

6.4.5　ワークスペース保存前に確認チェックボックス

上記の[ツール実行前にワークスペースの保存]チェックボックスに加えてこのチェックボックスをチェックすると、保存する前に確認の画面が表示されます。

6.4.6　セッション保存前に確認チェックボックス

このオプションをチェックすると、HEW はセッションをディスクに保存する前にプロンプトを表示します。

6.4.7　自動バックアップ時間間隔チェックボックス

このオプションをチェックすると、指定した時間間隔（デフォルト 10 分）で、ワークスペース、プロジェクト、およびセッションファイルをバックアップします。加えた変更はすべて、テンポラリファイルに保存されます。

ワークスペースを開いて以下のダイアログボックスが表示される場合、前回そのワークスペースを使用したときに問題が発生したことを意味します。ファイルを回復するには、回復するファイルの名前の横にあるチェックボックスをチェックし、[OK]ボタンをクリックしてください。[キャンセル]ボタンをクリックすると、自動バックアップファイルを削除し、元のファイルからロードを行います。

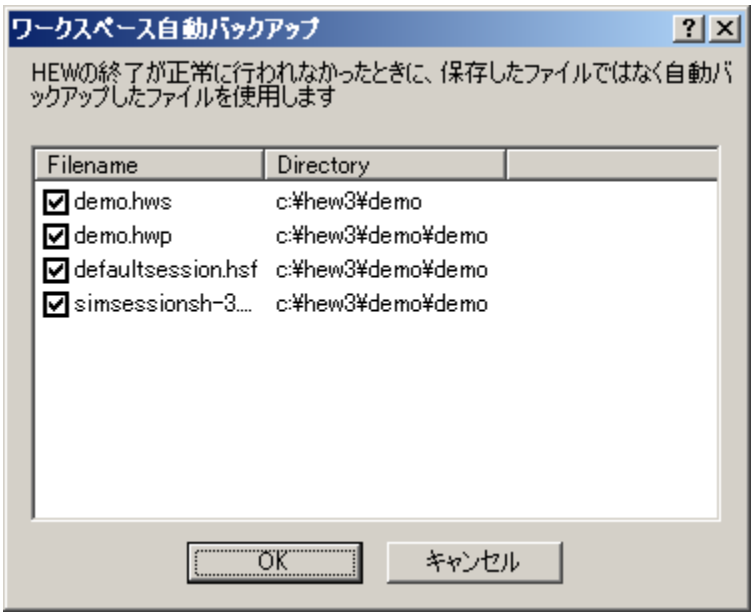


図 6.11: **ワークスペース自動バックアップダイアログボックス**

6.4.8 新規ワークスペースのデフォルトディレクトリエディットボックス

新しいワークスペースを作成すると [新規プロジェクトワークスペース] ダイアログボックスが起動します。このダイアログボックスにはその新しいワークスペースが作成されるディレクトリを入力するフィールドがあります。デフォルトでは、HEW インストールディレクトリが入力してあります。しかし、他のディレクトリ（例：“C:¥Workspaces”）にデフォルトを変更したい場合、このフィールドにそのディレクトリを入力するか、 [参照…] ボタンで位置を指定してください。

6.5 HEW エディタ以外のエディタを使う

HEW エディタ以外のエディタも使うことができます。外部のエディタを指定してあると、以下の操作をしたときにそれが起動します。

- ワークスペースウィンドウの[Projects]タブのファイルをダブルクリックしたとき
- ワークスペースウィンドウの[Navigation]タブのエントリをダブルクリックしたとき
- アウトプットウィンドウの[Build] タブのエラーやウォーニングをダブルクリックしたとき
- アウトプットウィンドウの[Find in Files] タブのエントリをダブルクリックしたとき
- ワークスペースウィンドウのポップアップメニューから[開く <ファイル名>] を選んだとき

➲ HEW エディタ以外のエディタを指定するには

1. [ツール->オプション…] を選んでください。 [オプション] ダイアログボックスが表示されるので [エディタ] タブ（図6.12）を選んでください。

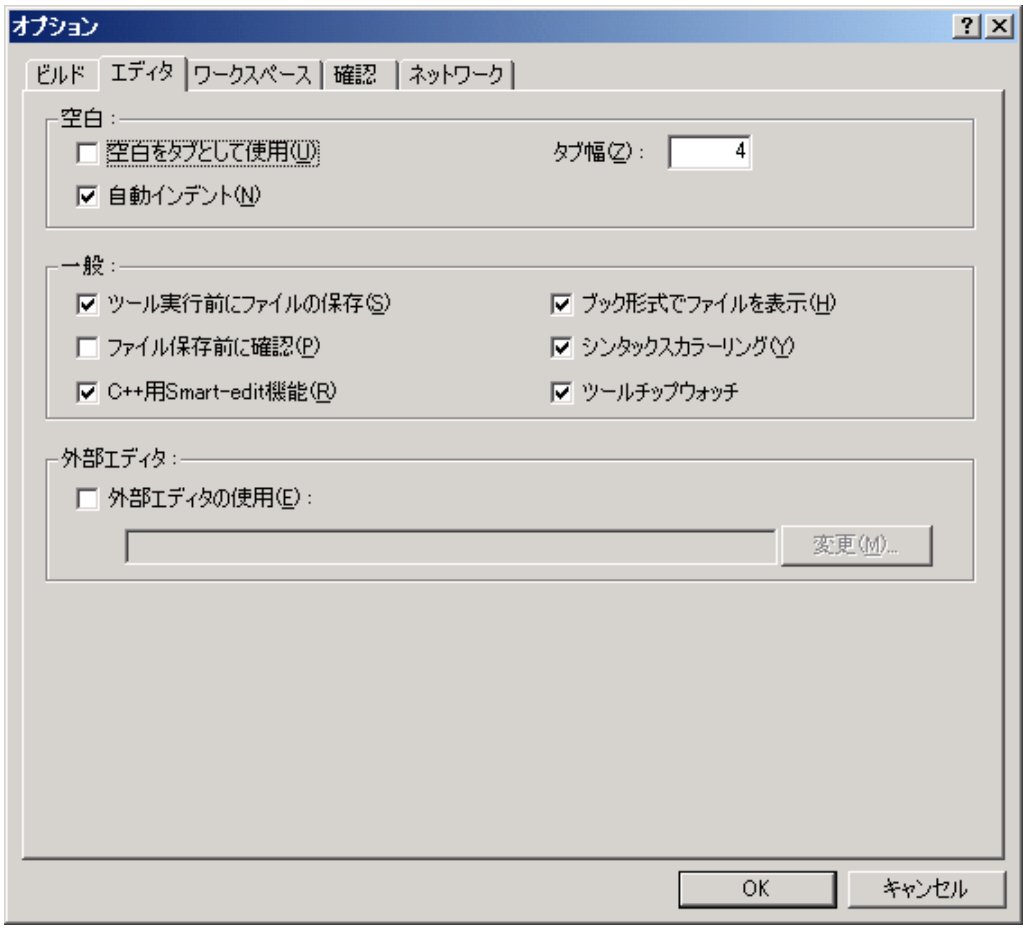


図 6.12: **オプション ダイアログボックス エディタ タブ**

2. [外部エディタの使用] チェックボックスをチェックしてください。 [外部エディタ] ダイアログボックスが表示されます （図6.13）。

図 6.13: **外部エディタ ダイアログボックス**

3. [コマンド] フィールドに実行可能ファイルのパス（引数を除く）を入力してください。
4. [ファイルを開く時の引数] フィールドにファイルを開くのに必要な引数を入力してください。開くファイルのパスには $(FULLFILE) プレースホルダを使ってください。
5. [行番号を指定してファイルを開く時の引数] フィールドにファイルの特定の行を開くのに必要な引数を入力してください。開くファイルのパスには $(FULLFILE) プレースホルダを使ってください。また、カーソルを最初に置く行の番号には $(LINE) プレースホルダを使ってください。
6. [OK] ボタンをクリックするとエディタが指定されます。

注意 HEW **エディタ以外のエディタを使う場合、以下のことに注意してください。**

- どのように起動しても、外部エディタを起動するたびに、エディタは新規に起動します。
- ビルド操作（[ビルド]、[すべてビルド]、[コンパイル]）を行う前にファイルを保存してください。
- デバッグ時には、外部エディタは使用できません。

6.6 ファイルの保存をカスタマイズする

[オプション] ダイアログボックス （図 6.12）の[エディタ] タブで、ファイルの保存方法をカスタマイズすることができます。[ツール->オプション…] で[エディタ] タブを選んでください。

ファイル保存に関するチェックボックスを以下に説明します。

6.6.1 ツール実行前にワークスペースの保存チェックボックス

このチェックボックスをチェックすると、ビルドフェーズ（[ビルド]、[すべてビルド]、[コンパイル]操作）またはバージョン管理コマンドを実行する前に編集したファイルを保存します。

6.6.2 ワークスペース保存前に確認チェックボックス

上記の[ツール実行前にワークスペースの保存]チェックボックスと、このチェックボックスをチェックすると、保存する前に確認メッセージを表示します。

6.7 カスタムプレースホルダを使う

HEW では、ディレクトリを定義するときに、以前に定義したプレースホルダを指定することができます。これによって、プロジェクトを再配置することができます。

また、HEW では、カスタムプレースホルダを定義することもできます。このことは、ユーザがカスタムプレースホルダを定義し、そのディレクトリの値を決められることを意味します。一度定義されると、このプレースホルダは HEW の別の場所で有効になります。

[アプリケーション内有効カスタムプレースホルダ]に定義したプレースホルダは、HEW で使用するすべてのワークスペースやプロジェクトに対して有効です。それに対して、[ワークスペース内有効カスタムプレースホルダ]に定義したプレースホルダは、現在のワークスペースにおいてのみ有効です。

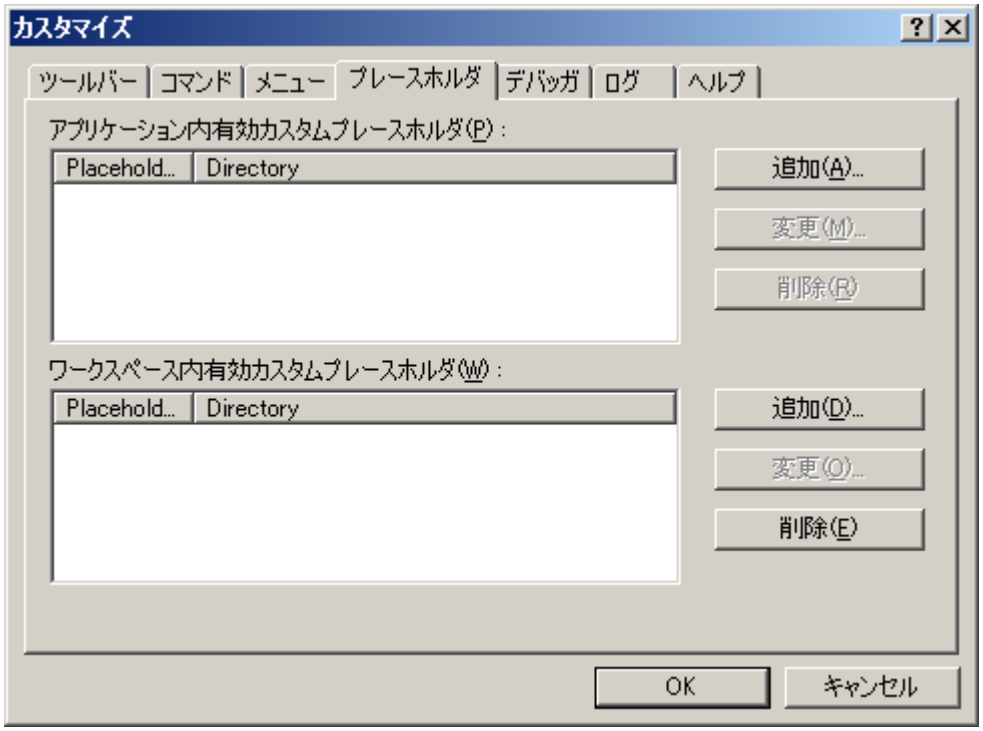


図 6.14: **カスタマイズ ダイアログボックス プレースホルダ タブ**

➲ カスタムプレースホルダを追加するには

1. [ツール->カスタマイズ...] を選んでください。図6.1に示すダイアログボックスが表示されます。
   [プレースホルダ]タブを選んでください（図6.14）。
2. [アプリケーション内有効カスタムプレースホルダ] 、あるいは [ワークスペース内有効カスタムプレースホルダ] のどちらを使いたいか、選んでください。リスト横にある[追加…]ボタンをクリックしてください。
3. [新規カスタムプレースホルダ] ダイアログボックスが表示されます（図6.15）。
4. このフィールドでは、プレースホルダの適切な名前、およびプレースホルダが意味する内容

の説明を選んでください。

5. 次に、このプレースホルダと関連するディレクトリを選んでください。$(PROJDIR)のように、オプション設定等で定義済みのプレースホルダとして使用することができます。

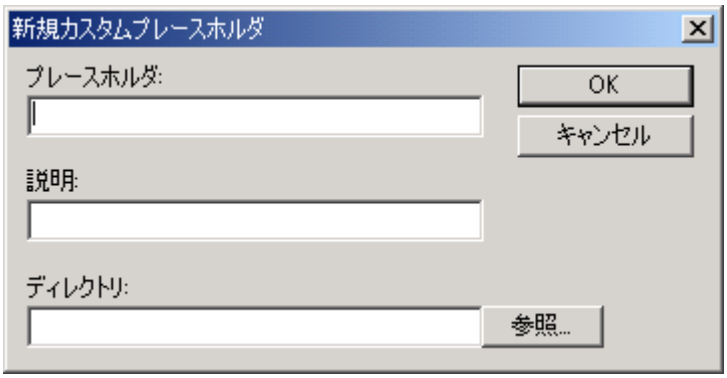


図 6.15: 新規カスタムプレースホルダダイアログボックス

6.8 ワークスペースやプロジェクトのログ機能を使う

HEW3.0 では、ワークスペースとプロジェクトのログ機能がアプリケーションの中に統合されています。この機能は、[カスタマイズ]ダイアログボックスの[ログ]タブで設定することができます。この設定により、ユーザ名や変更内容についてログファイルを作成できるため、ネットワークデータベースが動作しているときなどに特に便利です。このダイアログボックスを図 6.16 に示します。

[ワークスペースログファイル生成]をクリックすると、ワークスペースのすべての変更箇所がワークスペースと同じ名前で拡張子が.log のファイルにログされます。このファイルの位置は、ワークスペースファイルと同じディレクトリです。

[プロジェクトログの生成]をクリックすると、現在のワークスペースでのすべてのプロジェクトの変更箇所がプロジェクトと同じ名前で拡張子が.log のファイルにログされます。このファイルの位置は、ワークスペースファイルと同じディレクトリです。

ワークスペースを保存するとログファイルは更新されます。

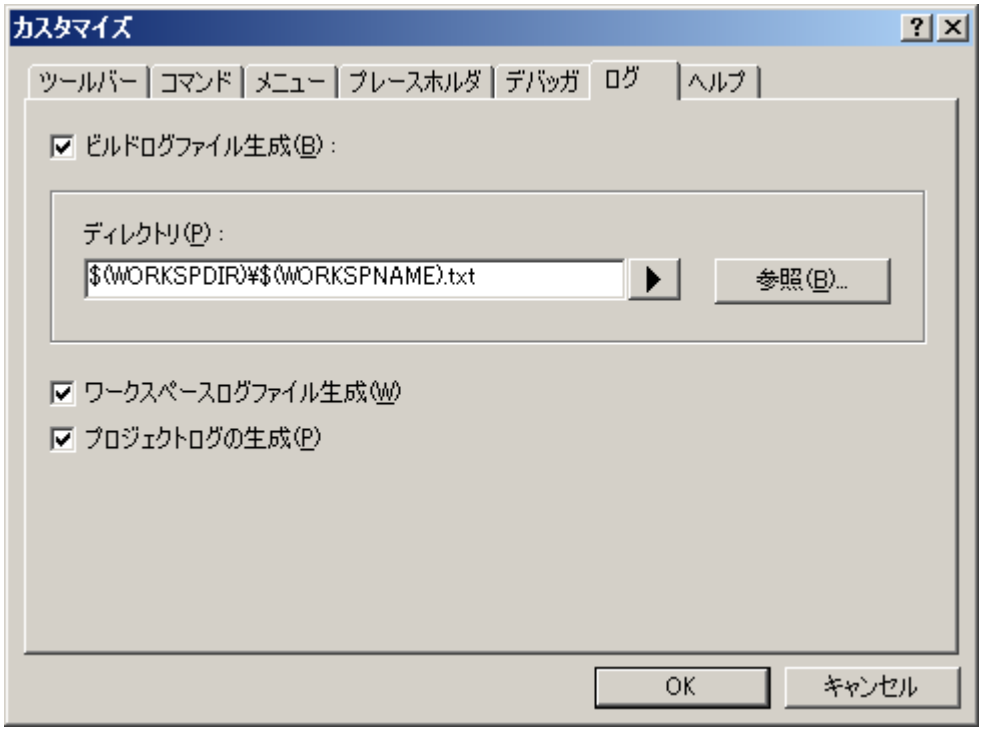


図 6.16: **[カスタマイズ]ダイアログボックスの[ログ]タブ**

6.9 バーチャルデスクトップを使用する

HEW ではバーチャルデスクトップの概念を導入しています。これにより、ボタンをクリックすることでウィンドウのコンフィグレーションの切り替えができるようになっています。ウィンドウのコンフィグレーションによって、ツールバーとウィンドウを表示または非表示にすることができます。

最大 4 つまでのウィンドウのコンフィグレーションを使用することが可能です。セッションを保存すると、各コンフィグレーションのウィンドウ位置がセッションファイルに保存されます。ウィンドウのコンフィグレーションを切り替えるだけで、他のウィンドウへアクセスすることができます。ツールバーとウィンドウは、コンフィグレーションに依存します。ソースファイルはバーチャルデスクトップシステムから独立しており、引き続き表示されます。

➲ コンフィグレーションに新しい名前を付けるには

1. [ウィンドウ->バーチャルデスクトップ] サブメニューから [デスクトップマネージャ] を選んでください。
2. [デスクトップマネージャ] ダイアログボックスで、名前を変更する [ウィンドウコンフィグレーション] を選んでください。
3. [名前の変更…] ボタンをクリックしてください。新しい名前を入力し、[OK]ボタンをクリックしてください。
4. [OK] ボタンをクリックしてください。

➲ コンフィグレーションを切り替えるには

ウィンドウコンフィグレーションの切り替えには、いくつかの方法があります。

もっとも簡単な方法は以下の通りです。

1. ステータスバー上のバーチャルデスクトップボタンをクリックします。

図 6.17 の例では、クリックしたバーチャルデスクトップボタンの番号は"1"で、コンフィグレーションに"Build"という名前が付けられています。ボタンの右側に " Build desktop " と表示しています。

別のバーチャルデスクトップボタンをクリックするとそのボタンが選択状態になり、ボタンの右側の表示も切り替わります。一度クリックすると、HEW は新しいコンフィグレーション形式でウィンドウをロードします。

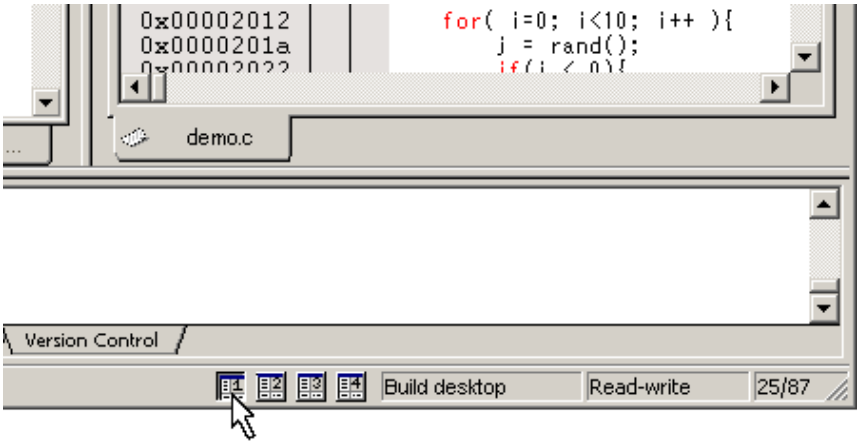


図 6.17: バーチャルデスクトップボタン

コンフィグレーションを切り替えるもう一つの方法は以下の通りです。

1. [ウィンドウ->バーチャルデスクトップ] サブメニューから表示したいコンフィグレーションを選んでください。

注意 コンフィグレーションが一度も使用されていない場合、HEW は現在のウィンドウコンフィグレーションを新しいコンフィグレーションにコピーします。ユーザはこの設定を変更することができます。セッションを保存すると、各ウィンドウコンフィグレーションでのウィンドウの位置を保存します。

7. バージョン管理

HEW はバージョン管理システムと接続することができます。プロジェクトでバージョン管理システムを使用する理由を以下に示します。

- プロジェクト開発環境の統合性を維持するため
- プロジェクトのバージョンを記録･保存するため
- ソースファイルに対するバージョン管理を行い、複数のユーザが一つのプロジェクトを共同開発できるようにするため

図 7.1 にバージョン管理システムを使用するプロジェクトの一般例を示します。ここでは 3 人のユーザがソースコードを相互参照するために同じ共有ネットワークドライブを使用しています。バージョン管理システムはソースファイルの参照や更新を管理するために使用します。

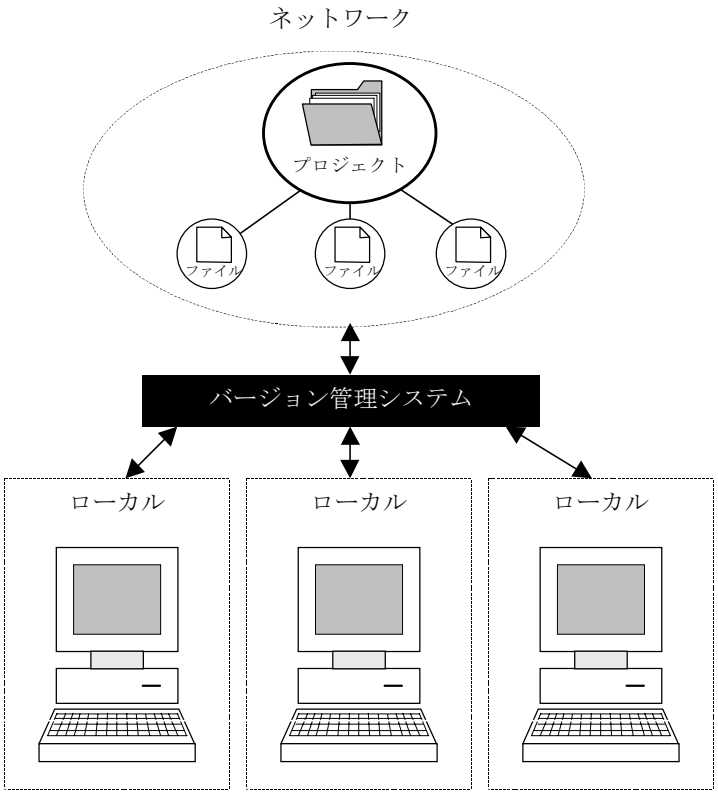


図 7.1: **バージョン管理**

7.1 バージョン管理システムを選択する

初期設定では、バージョン管理サブメニューが表示されます（図 7.2）。このとき、まだバージョン管理システムが現在のワークスペースで有効でないため、 [ツール->バージョン管理->選択...] オプションだけが利用できます。

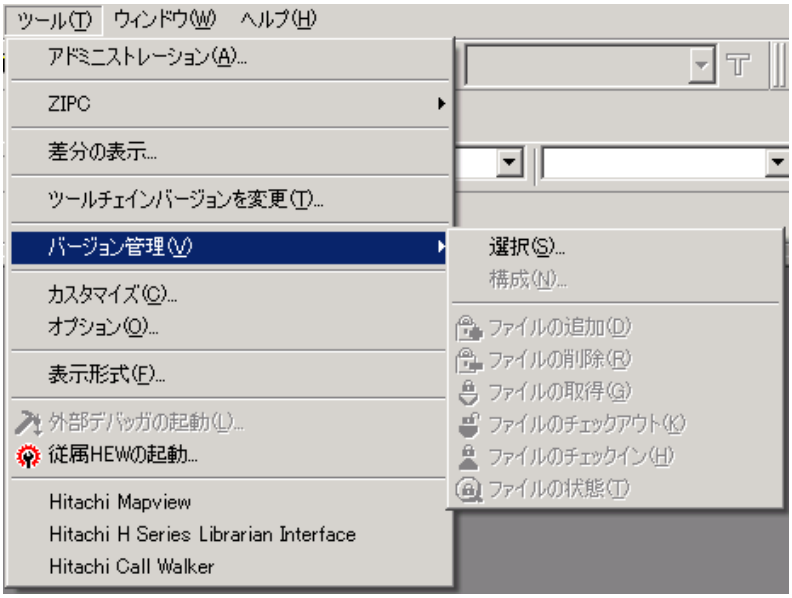


図 7.2: バージョン管理サブメニュー

➲ バージョン管理システムを選ぶには

1. [ツール->バージョン管理->選択...] を選んでください。図7.3に示すダイアログボックスが表示されます。このダイアログボックスにはサポートするバージョン管理システムがすべて表示されます。
2. [バージョン管理ツール] リストからバージョン管理システムを選んで [選択] ボタンをクリックしてください。 [現在のバージョン管理ツール] には新しい設定が表示されます。
3. [OK] ボタンをクリックしてください。

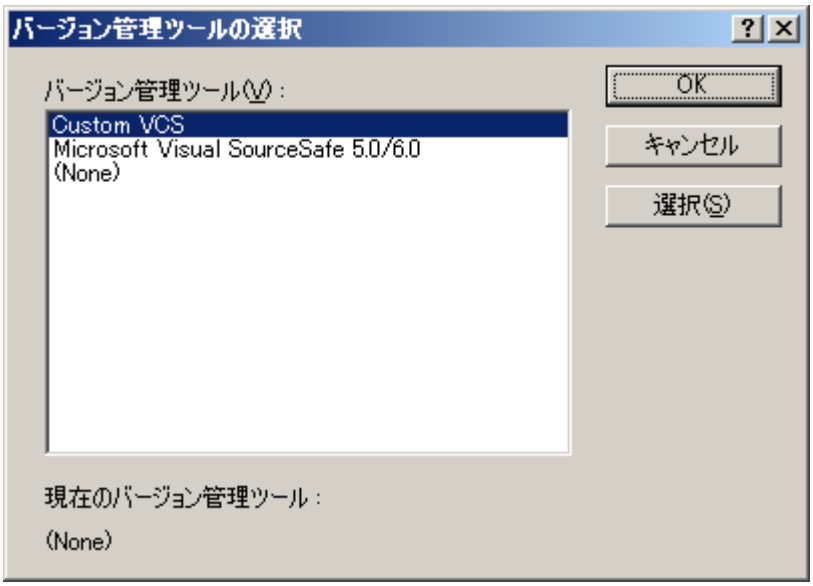


図 7.3: **バージョン管理ツールの選択 ダイアログボックス**

注意 HEW **と共にインストールされたバージョン管理システムだけが[バージョン管理ツールの選択] ダイアログボックスに表示されます（図** 7.3）。

バージョン管理システムを選択すると、[ツール->バージョン管理->構成...] オプションが使用できるようになります。

次の章ではバージョン管理システムの使用方法について説明します。

8. カスタムバージョン管理システム

HEW にカスタムのバージョン管理システムを接続できます。HEW は、すでにマシンにインストールされたバージョン管理システムと接続できます。つまり、HEW がバージョン管理ツール自体を提供するわけではなく、ワークスペースやプロジェクトで使用するバージョン管理システムを統合する手段を提供するだけです。

8.1 バージョン管理メニューオプションを定義する

カスタムのバージョン管理システムでは、[ツール->バージョン管理]サブメニューからオプションを選ぶか、バージョン管理ツールバーボタンでバージョン管理コマンドを起動することができます。そうすると、関連するコマンドを実行し、その出力が[アウトプット]ウィンドウの[Version Control] タブに表示されます。

➲ バージョン管理メニューオプションまたはツールバーボタンを実行するには

1. [ワークスペース] ウィンドウからバージョン管理コマンドを適用する項目（ワークスペース、プロジェクト、フォルダ、ファイルなど）を選んでください。コマンドが選択されると、すべてのファイルが選んだ項目から抽出され、バージョン管理コマンドに渡します。例えば、ワークスペースアイコンを選ぶと、そのプロジェクトのすべてのファイルがバージョン管理コマンドに渡します。これには、システムファイルも含まれます。
2. [ツール->バージョン管理] サブメニューからメニューオプションを選ぶか、バージョン管理ツールバーボタンを選んでください。

カスタムバージョン管理のサポートにより、バージョン管理システムの指定が任意に構築できます。[ツール->バージョン管理->構成…] を選んで [Version Control Setup] ダイアログボックスを表示してください （図 8.1）。

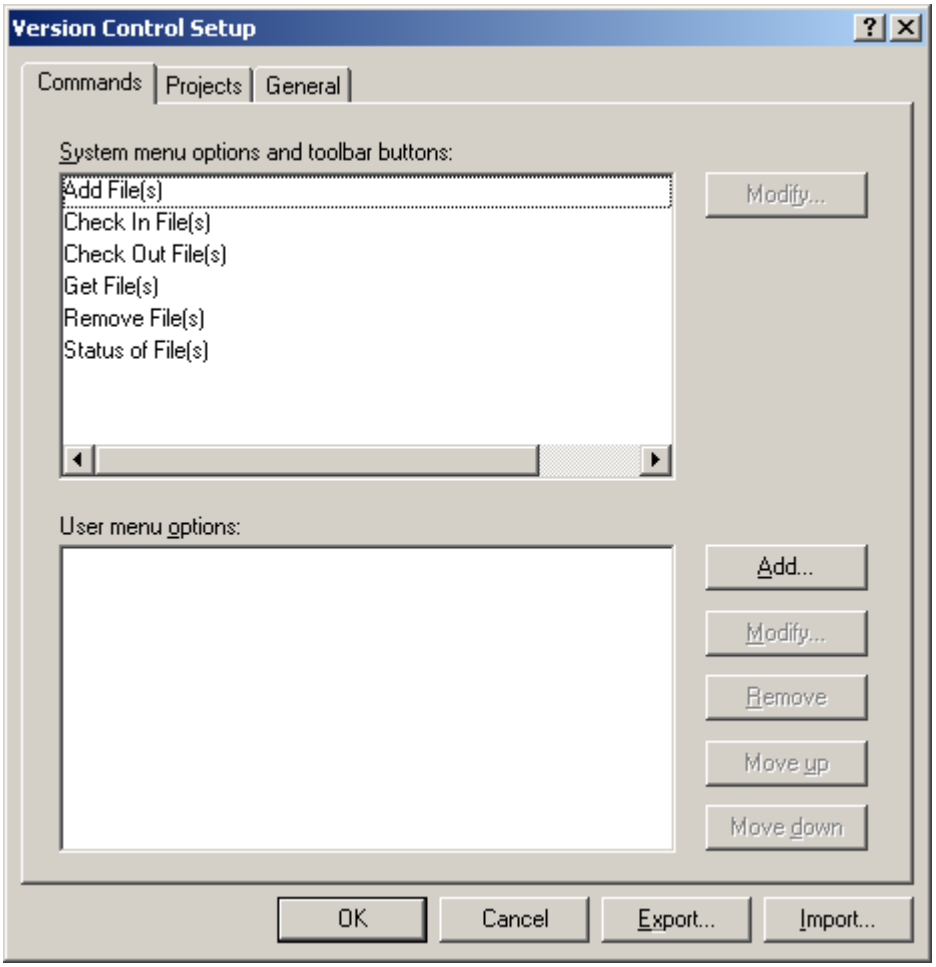


図 8.1: Version Control Setup **ダイアログボックス** Commands **タブ**

[Commands]タブには 2 つのメニューオプションのリストがあります。1 つめのリスト[System menu options and toolbar buttons]は常にバージョン管理サブメニューに表示されるメニューオプションです。これに対応するバージョン管理ツールバーがあります。2 つめのリスト [User menu options]はユーザ定義の追加メニューオプションで、バージョン管理サブメニューの最後に追加されます。図 8.2 にバージョン管理サブメニューの構成を示します。

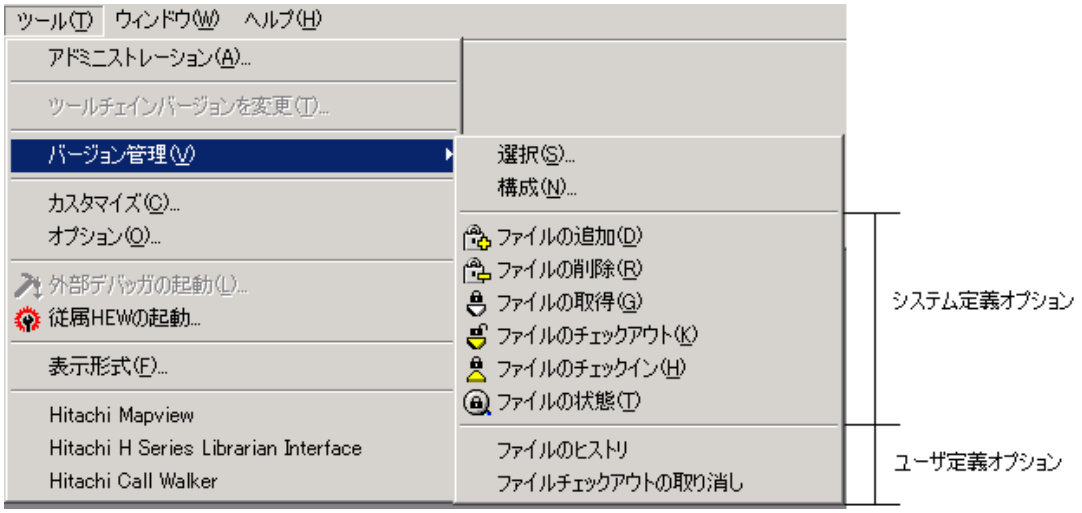


図 8.2: **バージョン管理 サブメニュー**

8.1.1 システムメニューオプションとツールバーボタン

ツールバーまたは[ツール->バージョン管理]サブメニューのシステム定義オプションからコマンドを起動するには、まず、起動したときに実行する関連したコマンドを定義しなければなりません。オプション名とその説明を表 8.1 に示します。

表 8.1: システムメニューオプションの説明

オプション	説明
Add File(s)	バージョン管理に選択したファイルを追加する
Remove File(s)	バージョン管理から選択したファイルを削除する
Get File(s)	バージョン管理から選択したファイルの読み取り専用ローカルファイルを取得する
Check In File(s)	バージョン管理に選択したファイルのローカルコピーを戻し更新する
Check Out File(s)	バージョン管理から選択したファイルの書き込み可能なローカルファイルを取得する
Status of File(s)	選択したファイルの状態を表示する

➲ システムメニューやツールバーボタンを変更するには

1. [バージョン管理->構成…] を選ぶと図8.1 に示すダイアログボックスが表示されます。
2. [System menu options and toolbar buttons] リストから変更するオプションを選んで [Modify…] ボタンをクリックすると 図8.3 に示すダイアログボックスが表示されます。図8.3は[Add File(s)]を選択した場合の例です。
3. [Add…] ボタンでコマンドを追加してください。詳細はこの章の後半の「バージョン管理コマンドを定義する」を参照してください。
4. [OK] ボタンをクリックして [Define Command for “<コマンド名>”] ダイアログボックスを閉じてください。
5. [OK] ボタンをクリックして [Version Control Setup] ダイアログボックスを閉じてください。

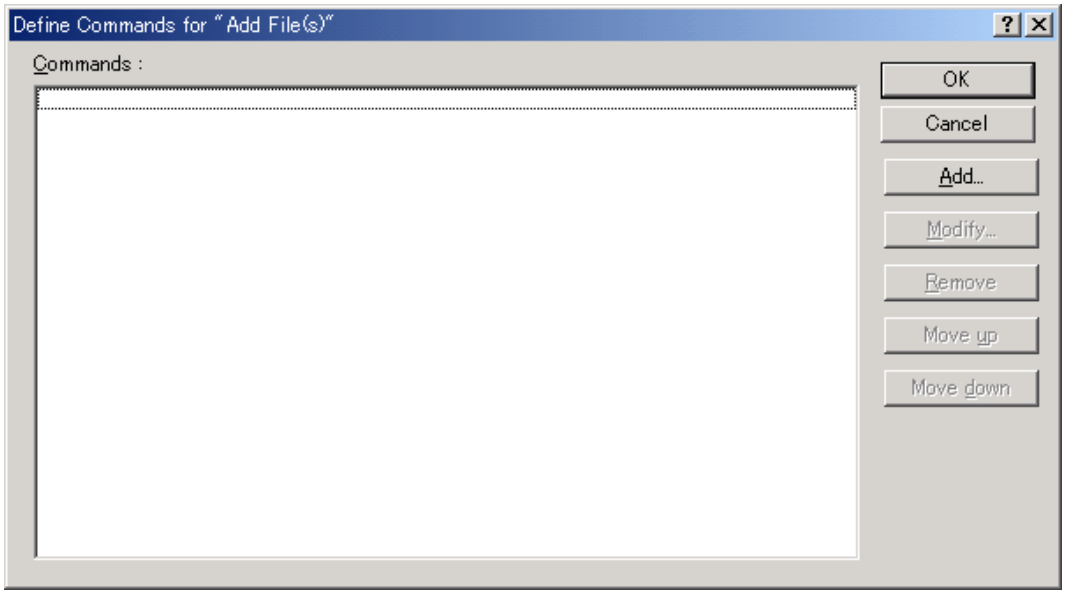


図 8.3: システムメニューオプションの変更（例）

8.1.2 ユーザ定義メニューオプション

ユーザ定義のメニューオプションはいくつでも作成できます。名前も自由につけられます。また、メニューに表示する順序も指定できます。しかし、ユーザ定義のメニューオプションはバージョン管理ツールバーには表示されません。

➲ 新しいバージョン管理メニューオプションを作成するには

1. [バージョン管理->構成…] を選んでください。図8.1に示すダイアログボックスが表示されます。
2. [Add...] ボタンをクリックしてください。図8.4に示すダイアログボックスが表示されます。
3. [Option] フィールドにメニューオプション名を入力してください。
4. [Add…] ボタンでメニューオプションにコマンドを追加してください。詳細は、この章の後半の 「バージョン管理コマンドを定義する」を参照してください。
5. [OK] ボタンをクリックして [Add Menu Option] ダイアログボックスを閉じてください。
6. [OK] ボタンをクリックして [Version Control Setup] ダイアログボックスを閉じてください。

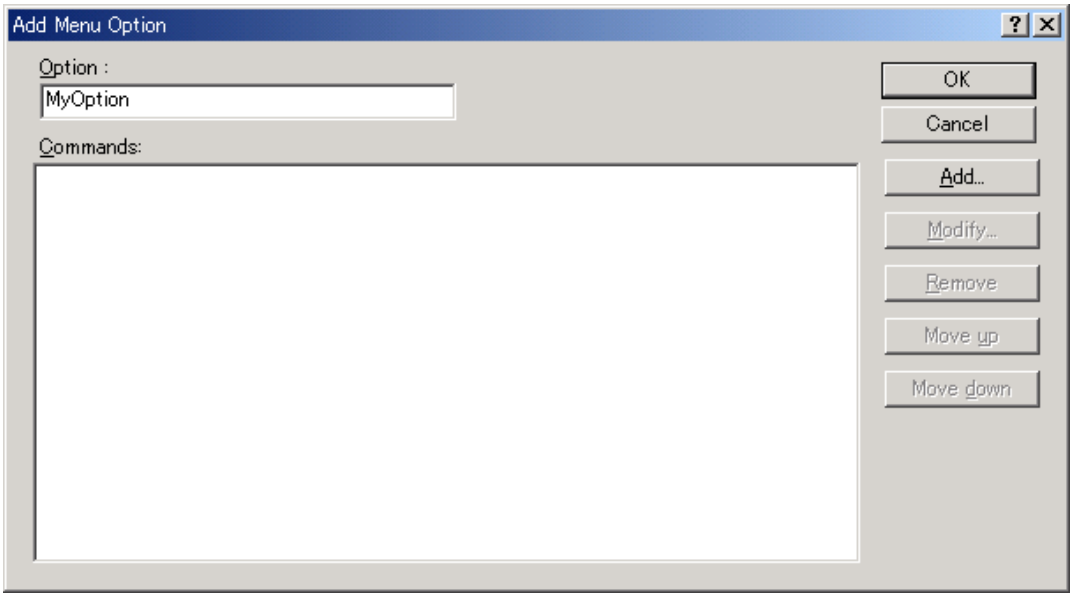


図 8.4: Add Menu Option **ダイアログボックス**

➲ 既存のバージョン管理メニューオプションを削除するには

1. [バージョン管理->構成…] を選んでください。図8.1に示すダイアログボックスが表示されます。
2. [User menu options] リストから削除するメニューオプションを選んで [Remove] ボタンをクリックしてください。
3. [OK] ボタンをクリックして [Version Control Setup] ダイアログボックスを閉じてください。

➲ 既存のバージョン管理メニューオプションを変更するには

1. [バージョン管理->構成…] を選んでください。図8.1に示すダイアログボックスが表示されます。
2. [User menu options] リストから変更するメニューオプションを選び、リストの横の

[Modify…] ボタンをクリックしてください。図8.4に示すダイアログボックスが表示されます（ただしタイトルは[Modify Menu Option]です）。

3. メニューオプション名を変更して [OK] ボタンをクリックしてください。
4. [OK] ボタンをクリックして [Version Control Setup] ダイアログボックスを閉じてください。

➲ バージョン管理メニューオプションの順序を変更するには

1. [バージョン管理->構成…] を選んでください。図8.1に示すダイアログボックスが表示されます。
2. 移動するメニューオプションを選んで、上に移動するには [Move up] ボタンを、下に移動するには [Move down] ボタンをクリックしてください。
3. [OK] ボタンをクリックして [Version Control Setup] ダイアログボックスを閉じてください。

8.2 バージョン管理コマンドを定義する

図 8.3 や図 8.4 で [Add...] や [Modify…] ボタンをクリックすると、コマンドを定義できます。どちらの場合も、図 8.5 に示すダイアログボックスが表示されます。

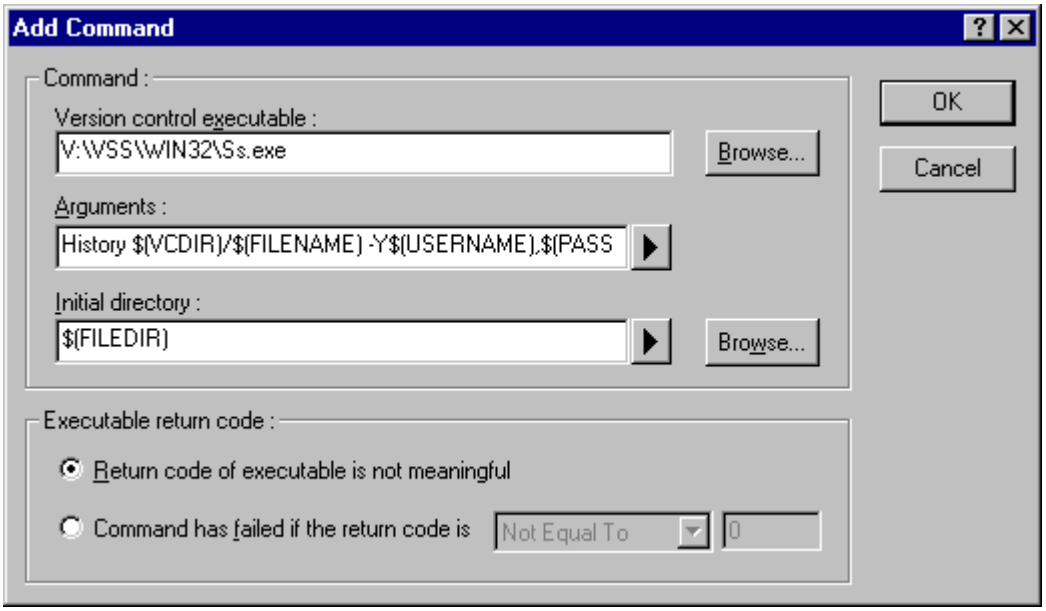


図 8.5: Add/Modify Command **ダイアログボックス**

➲ コマンドを定義するには

1. [Version control executable] フィールドにコマンド実行ファイルのフルパスを入力するか、[Browse…] ボタンをクリックしてコマンド実行ファイルの位置を指定してください。
2. [Arguments] フィールドにコマンドラインパラメータを入力してください。
3. [Initial directory] フィールドに実行ファイルを起動するディレクトリを入力するか、[Browse…] ボタンをクリックして起動ディレクトリを指定してください。ただし、特に起動ディレクトリを指定する必要がない場合は、 “$(FILEDIR)” プレースホルダを指定してファイルのディレクトリと同じディレクトリでコマンドを実行してください。
4. [Executable return code] オプションを設定してください。オプションの設定は次節を参照してください。

5. [OK] ボタンをクリックすると新しいコマンドが定義されます。

8.2.1 Executable return code オプション

コマンドのリターンコードでエラーを示す場合には、[Command has failed if the return code is] オプションを選択して右の 2 つのフィールドを設定してください。

[Command has failed if the return code is] オプションが選択されている場合、HEW は各コマンドのリターンコードをチェックしてエラーが起こったかどうか判定します。エラーが起こった場合、コマンド実行は停止してそれに続くコマンド実行やコマンドの後に続く処理（例：ビルド）は実行されません。

[Return code of tool is not meaningful] オプションが選択されている場合、HEW は各コマンドのリターンコードをチェックしません。したがって、すべてのコマンドが実行されます。

8.3 変数を指定する

変数は正しく指定しなければなりません。もし間違っていると、バージョン管理システムが正しく動作しません。また、1 つのバージョン管理コマンドを複数のファイルに適用する場合があるため、変数をフレキシブルな方法で指定することが重要です。これを行うために、[Arguments] フィールドにはプレースホルダボタンがあります（プレースホルダの詳細は、付録 C, 「プレースホルダ」を参照してください）。プレースホルダボタンをクリックすると、使用できるプレースホルダがポップアップメニューで表示されます（図 8.6）。表 8.2 に各プレースホルダの説明と実際の値を示します。

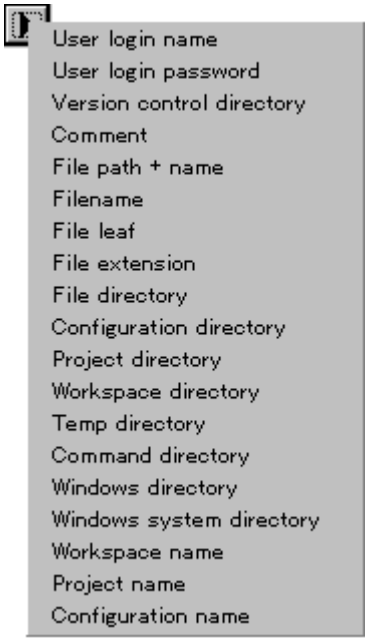


図 8.6: 変数フィールドプレースホルダのポップアップメニュー

表 8.2: 変数フィールドのプレースホルダ

プレースホルダ	実際の値
User login name	現在のユーザログイン (“General”タブ)
User login password	現在のユーザパスワード (“General”タブ)
Version control directory	「仮想的」バージョン管理マップ (“Projects”タブ)
Comment	コマンド実行前に指定したコメント
File path + name	操作するファイル名とそのフルパス
Filename	操作するファイル名（拡張子を含む）
File leaf	操作するファイル名（拡張子を含まない）
File extension	操作するファイルの拡張子
File directory	操作するファイルのディレクトリ
Configuration directory	現在のコンフィグレーションディレクトリ
Project directory	現在のプロジェクトのディレクトリ
Workspace directory	現在のワークスペースディレクトリ
Temp directory	テンポラリディレクトリ
Command directory	バージョン管理実行ディレクトリ
Windows directory	Windows® がインストールされているディレクトリ
Windows system directory	Windows®のシステムファイルがあるディレクトリ
Workspace name	現在のワークスペース名
Project name	現在のプロジェクト名
Configuration name	現在のコンフィグレーション名

8.3.1 ファイルの位置を指定する

ファイルの位置を指定するときには、プレースホルダを使用してください。そうでないと、そのコマンドは指定したファイルにしか適用できません。例えば、バージョン管理のアプリケーションに–GET コマンドが使われていて、ファイルの読み出し専用コピーを作成するとします。このとき、[Arguments] フィールドは以下のように指定できます。

```
-GET “c:¥vc¥files¥project¥main.c”
```

しかし、このコマンドを実行しても、ファイル MAIN.C しかコピーできません。この問題を解決するために、HEW にはプレースホルダとディレクトリのマッピングというシステムがあります。マッピングにより、どの「作業中の」ディレクトリ（作業中のソースファイルがあるディレクトリ）がどの「管理」ディレクトリ（バージョン管理システムに保存されているソースファイルのディレクトリ）に対応するかを指定します。これら 2 つのディレクトリ間のマッピングは、[Version Control Setup] ダイアログボックスの“Projects”タブ（図 8.7）で指定できます。

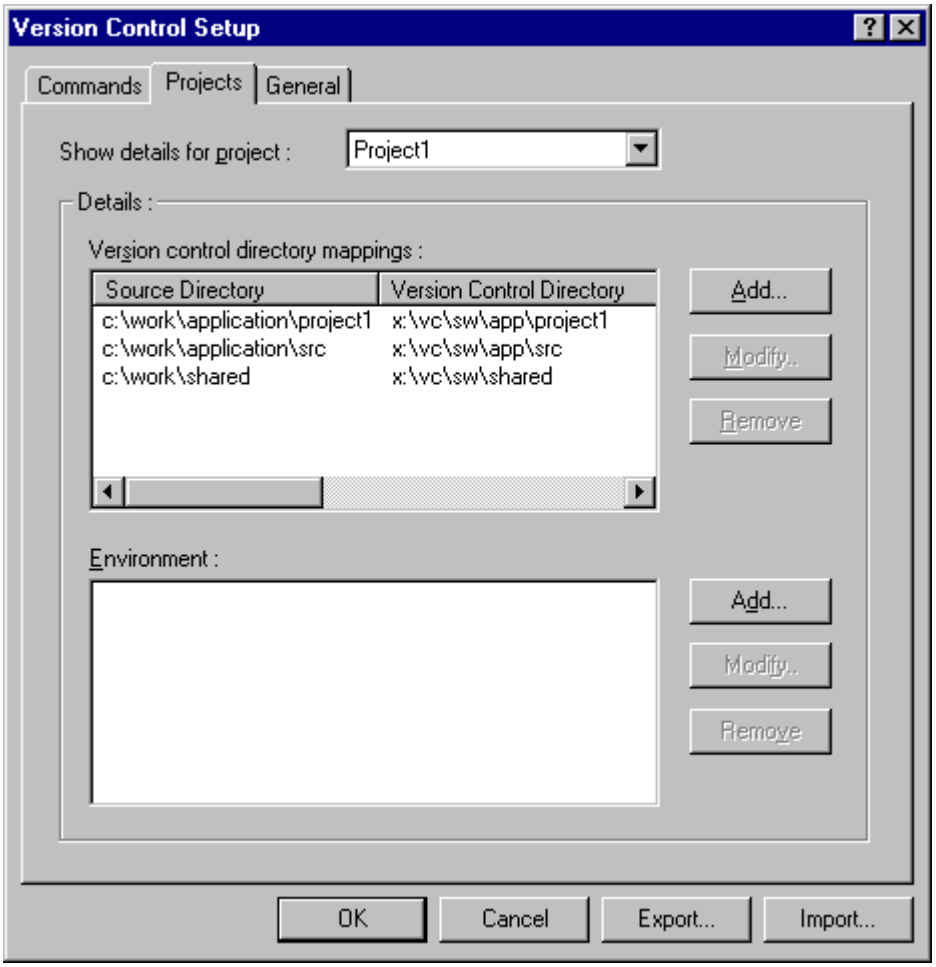


図 8.7: Version Control Setup **ダイアログボックス** Projects **タブ**

➲ 新しいマッピングを定義するには

1. [バージョン管理->構成…] を選んでください。図8.1に示すダイアログボックスが表示されます。 [Projects]タブを選んでください。図8.7に示すダイアログボックスが表示されます。
2. [Version control directory mappings] リストの横にある [Add…] ボタンをクリックしてください。図8.8に示すダイアログボックスが表示されます。
3. [Source directory] フィールドにもとの作業中のディレクトリを入力するか、 [Browse…] ボタンでディレクトリを選んでください。
4. [Version control directory] フィールドにバージョン管理ディレクトリを入力してください。

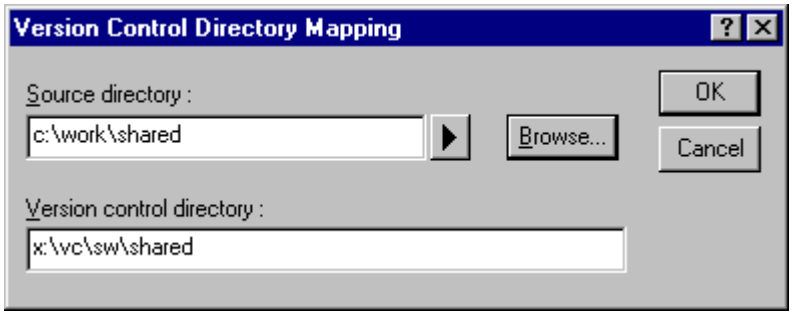


図 8.8: Version Control Directory Mapping **ダイアログボックス**

➲ 既存のマッピングを変更するには

1. [バージョン管理->構成…] を選んでください。図8.1に示すダイアログボックスが表示されます。 [Projects] タブを選んでください。図8.7に示すダイアログボックスが表示されます。
2. [Version control directory mappings] リストから変更するマッピングを選び、 [Modify…] ボタンをクリックしてください。 図8.8に示すダイアログボックスが表示されます。
3. 二つのディレクトリを変更して [OK] ボタンをクリックしてください。

➲ 既存のマッピングを削除するには

1. [バージョン管理->構成…] を選んでください。図8.1に示すダイアログボックスが表示されます。 [Projects] タブを選んでください。図8.7に示すダイアログボックスが表示されます。
2. [Version control directory mappings] リストから削除するマッピングを選んで [Remove] ボタンをクリックしてください。

マッピングを定義すると、[Version control directory] プレースホルダ $(VCDIR)を使用してプロジェクトファイルを保存するディレクトリを示すことができます。図8.9に例を示します。ここではネットワークを共有したバージョン管理ドライブ(X:¥) と開発が行われているローカルドライブ(C:¥)からマッピングされた3つのディレクトリがあります。

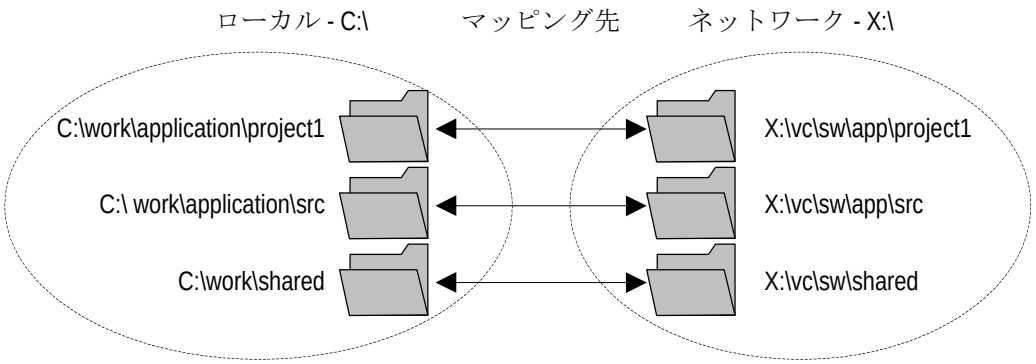


図 8.9:**マッピングの例**

例えば、–GET コマンドが使われていてファイルの読み出し専用コピーを作成するバージョンコントロールのアプリケーションが選ばれているとします。プロジェクトのすべてのファイルを取得するには、次のコマンドを使用します。

```
-GET “$(VCDIR)¥$(FILENAME)”
```

そのプロジェクトファイルのコマンドを実行すると、$(VCDIR) をファイルマッピングの中の対応するバージョン管理ディレクトリに置き換えます。

例えば、 FILE1.C が C:¥work¥application¥project1¥file1.c にあるとします。FILE1.C に-get コマンドが適用されると、次のようになります。

1. $(VCDIR)がx:¥vc¥sw¥app¥project1に置き換えられます。これはC:¥work¥application¥project1のバージョン管理ディレクトリマッピングに対応しているからです。
2. $(FILENAME) がFILE1.Cに置き換えられます。

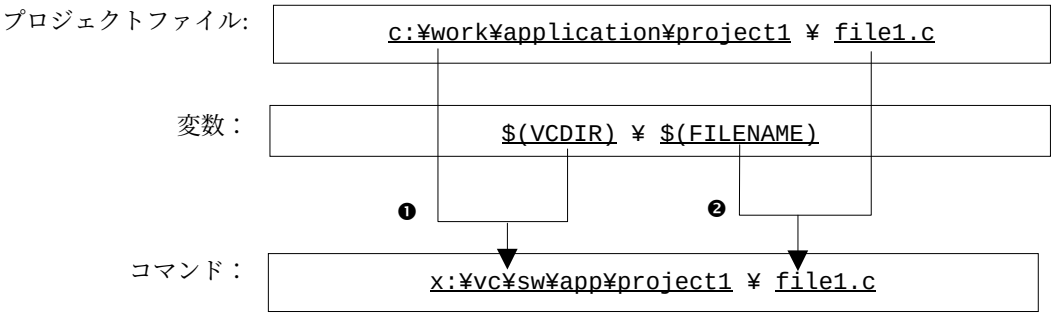


図 8.10: **置き換えの例**

8.3.2 環境変数の設定

現在の設定を確認するには[Version Control Setup]ダイアログボックスの[Projects] タブを表示してください（図 8.7）。

➲ 新しい環境変数を追加するには

1. [Environment] リストの横にある [Add…] ボタンをクリックしてください。図8.11に示すダイアログボックスが表示されます。 [Variable] フィールドに環境変数名を、[Value]フィールドに環境変数の値を入力して [OK]ボタン をクリックしてください。[Environment] リストに新しい環境変数が追加されます。

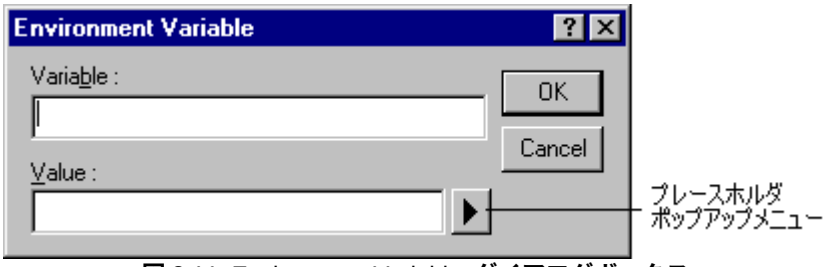


図 8.11: Environment Variable **ダイアログボックス**

➲ 環境変数を変更するには

1. 変更する環境変数を [Environment] リストから選んで、 [Modify…] ボタンをクリックしてください。[Variable] フィールドと [Value] フィールドを必要に応じて変更して[OK]ボタンをクリックすると、変更した環境変数がリストに追加されます。

➲ 環境変数を削除するには

1. 削除する環境変数を [Environment] リストから選んで、 [Remove] ボタンをクリックしてください。

8.3.3 コメントを指定する

コマンドに “$(COMMENT)” プレースホルダを含む場合、コマンド実行時に図 8.12 に示すダイアログボックスにコメントを入力してください。

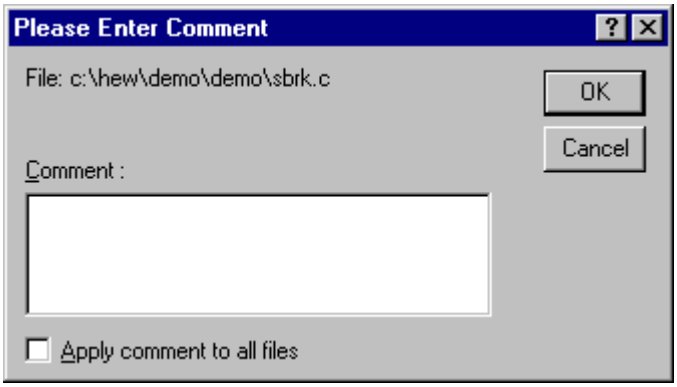


図 8.12: Please Enter Comment **ダイアログボックス**

コメントは各コマンドごとに指定できます。また、[Apply comment to all files] チェックボックスをチェックして [OK]ボタンをクリックすると、すべてのファイルに同じコメントが指定できます。

8.3.4 ユーザ名とパスワードを指定する

バージョン管理ツールでは一般的にユーザ名とパスワードの入力をコマンドラインで行う必要があります。これは、ファイルを保護し、どのファイルがどのユーザによって変更されたか記録するためです。バージョン管理システムでは 2 つのプレースホルダ “User login name”, $(USERNAME), と “User login password”, $(PASSWORD)をサポートします。コマンドを実行すると、これらのプレースホルダは[Version Control Setup]ダイアログボックスの[General] タブ（図 8.13）の現在の設定に置き換えます。

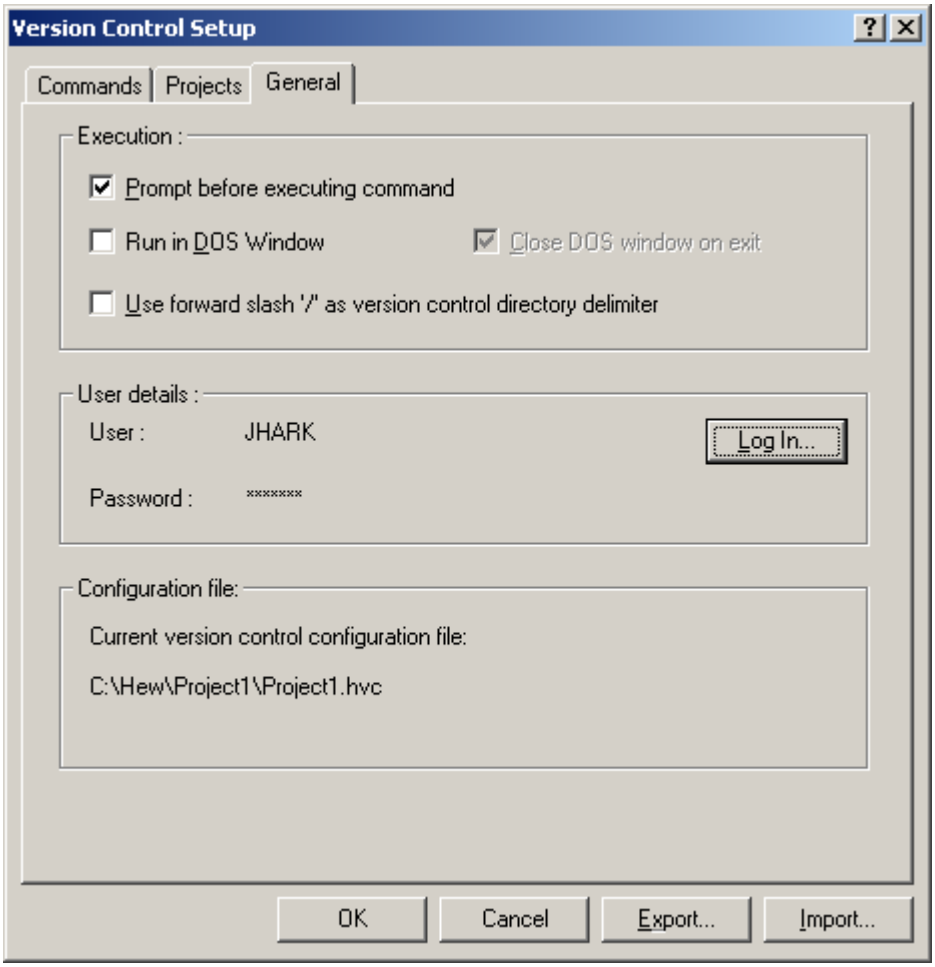


図 8.13: Version Control Setup **ダイアログボックス** General **タブ**

プレースホルダ$(USERNAME) と $(PASSWORD)に値を設定するには、最初にログインの指定をしておく必要があります。もしこれらのプレースホルダのうちのどちらかのプレースホルダを使うコマンドを実行する前にログイン指定をしていなかった場合、コマンド実行前にログインするよう要求されます。

➲ ユーザ名とパスワードを設定するには（ログインの指定）

1. [Log in…] ボタンをクリックしてください。図8.14に示すダイアログボックスが表示されます。
2. [User name] フィールドにユーザ名を入力してください。
3. [Password] フィールドにパスワードを入力してください。
4. [Confirm password by retyping it below] フィールドにパスワードを再入力してください。.
5. [OK] ボタンをクリックすると新しいユーザ名とパスワードが設定されます。[Password] フィールドと[Confirm password by retyping it below] フィールドで異なるパスワードが入力された場合はもう一度パスワードを入力するよう要求されます。

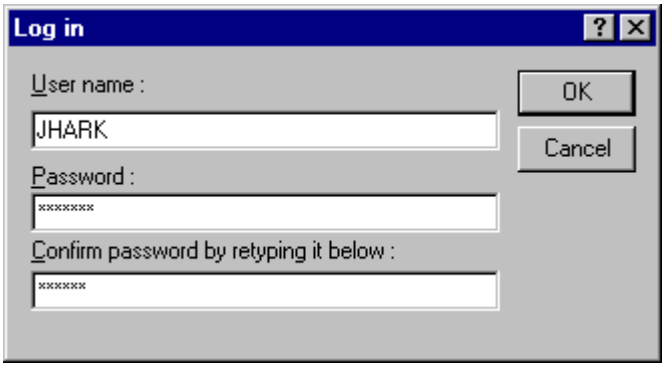


図 8.14: Log in **ダイアログボックス**

8.4 実行を制御する

[Version Control Setup]ダイアログボックスの [General] タブ（図 8.13）では、使用するバージョン管理ツールと HEW を制御することができます。また、現在のバージョン管理コンフィグレーションファイルへのフルパスを表示します。

8.4.1 Prompt before executing command チェックボックス

バージョン管理コマンドを実行する前にこのチェックボックスをチェックすると、図 8.15 に示すダイアログボックスが表示されます。このダイアログボックスではコマンドの実行の有無を確認することができます。チェックボックスのチェックを外すとそのコマンドは実行しません。[OK]ボタン をクリックすると、選んだコマンドを実行します。[Cancel]ボタン をクリックするとコマンドは実行しません。

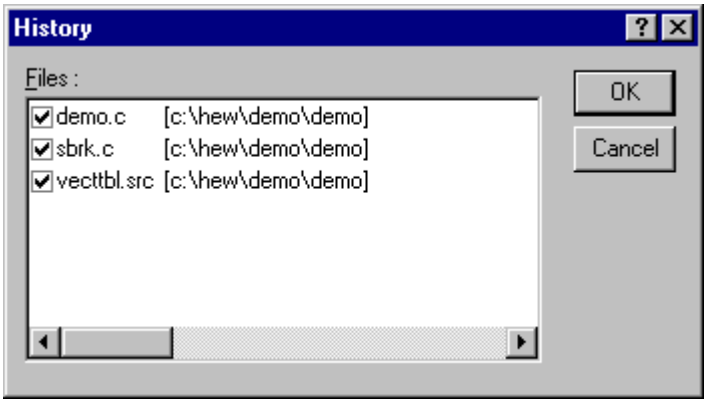


図 8.15: **コマンドプロンプト ダイアログボックス**

8.4.2 Run in DOS Window チェックボックス

デフォルトでは、バージョン管理コマンドの出力は[アウトプット]ウィンドウの[Version Control]タブに結果が表示されます。このチェックボックスをチェックすると、各コマンドを別の DOS ウィンドウで実行します。

8.4.3 Use forward slash ‘/’ as version control directory delimiter チェックボックス

デフォルトでは、HEW がプレースホルダ$(VCDIR)を置き換えるときバックスラッシュ文字 ‘¥’（日本語 Windows®では円記号）を使ってディレクトリを区切ります。しかし、お使いのバージョン管理システム(例 Visual SourceSafe)で、ディレクトリを区切るときにスラッシュ文字‘/’を使う場合は 、[Use forward slash ‘/’ as version control directory delimiter]チェックボックスをチェックしてスラッシュ文字‘/’でディレクトリを区切ってください。

8.5 設定内容の保存と適用

ワークスペースごとに異なるバージョン管理設定を行うことができます。HEW ではそれぞれのバージョン管理設定を保存して他のワークスペースに適用することができます。これにより、複数のワークスペースで何度も同じバージョン管理設定を行う必要がなく、時間を節約することができます。

➲ バージョン管理設定を保存するには

1. [バージョン管理->構成…] を選んでください。図8.1に示すダイアログボックスが表示されます。
2. [Export…] ボタンをクリックしてください。標準のファイル保存ダイアログボックスが表示されます。設定内容を保存するディレクトリを選んでください。
3. ファイル名を入力して [OK] ボタンをクリックしてください。

➲ バージョン管理設定を適用するには

1. [バージョン管理->構成…] を選んでください。図8.1に示すダイアログボックスが表示されます。
2. [Import…] ボタンをクリックしてください。標準のファイルを開くダイアログボックスが表示されます。適用するファイル（*.HVC） を選んでください。
3. [OK] ボタンをクリックしてください。

9. Visual SourceSafe を使用する

HEW は Visual SourceSafe バージョン管理システムをサポートしています。現在、HEW は Visual SourceSafe のバージョン 5.0 と 6.0 をサポートしています。

Visual SourceSafe バージョン管理システムでは、Visual SourceSafe データベースのなかのプロジェクトとワークスペースのなかのプロジェクトを関連付けます。[ツール->バージョン管理] サブメニューからオプションを選ぶか、バージョン管理ツールバーボタンをクリックすることにより、標準コマンドをすばやく起動することができます。

9.1 ワークスペースに Visual SourceSafe を関連づける

以下の節では、Visual SourceSafe と現在のワークスペースとを関連付ける方法を説明します。

9.1.1 Visual SourceSafe を選ぶ

まず、Visual SourceSafe をバージョン管理システムとして選びます。

➲ Visual SourceSafe 5.0 または 6.0 を使うには

1. [ツール->バージョン管理->選択…] を選んでください。 [バージョン管理ツールの選択] ダイアログボックス（図7.3）に、サポートするバージョン管理システムを表示します。
2. バージョン管理システムのリストから [Visual SourceSafe 5.0/6.0] の項目を選び、 [選択] ボタンをクリックしてください。
3. [OK] ボタンをクリックしてください。[SourceSafe Login] ダイアログボックス（図9.1）が表示されます。
4. [Username] にVisual SourceSafeのログインを、 [Password] にパスワードを入力してください。
5. プロジェクトを追加するVisual SourceSafeデータベース（SRCSAFE.INI）へのフルパスをDatabase pathフィールドに入力してください。
6. [OK] ボタンをクリックしてください。 [Create SourceSafe Project] ダイアログボックス（図9.2）が表示されます。
7. [Project name] フィールドに、データベースに作成するプロジェクト名（フォルダ名）が表示されます。プロジェクト名は変更できます。
8. [Project name] フィールド下のツリーにはステップ6で指定したデータベースの構造を示します。[Project name]フィールドに指定したフォルダをどのフォルダ内に作成するかを選んでください。
9. [OK] ボタンをクリックしてください。
10. 現在のワークスペースにあるプロジェクトの数だけ、ステップ７～９を繰り返してください。

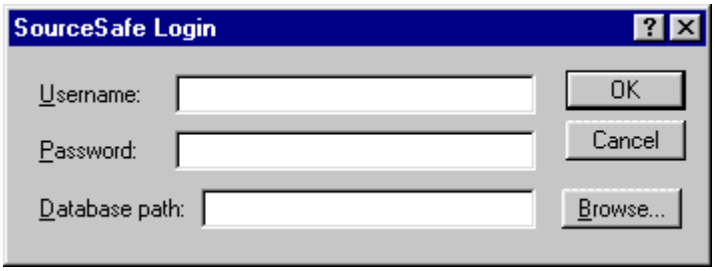


図 9.1: SourceSafe Login **ダイアログボックス**

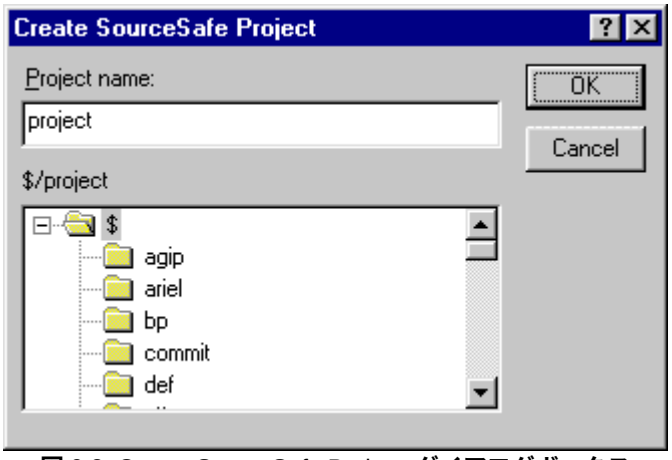


図 9.2: Create SourceSafe Project **ダイアログボックス**

HEW は、Visual SourceSafe 内に必要なプロジェクトを作成して、迅速にアクセスできるように、バージョン管理ツールバーやメニューを設定します。Visual SourceSafe プロジェクト自体は作成しましたが、まだファイルは追加していません。

9.1.2 Visual SourceSafe にファイルを追加する

前節では、ハードディスク上のプロジェクトディレクトリ（作業中のディレクトリ）と Visual SourceSafe のプロジェクトディレクトリ（Visual SourceSafe によって管理されたディレクトリ）との間でマッピングを確立しただけです。ハードディスクのプロジェクトディレクトリ（サブディレクトリを含む）に複数のソースファイルがあってもマッピング先の Visual SourceSafe のディレクトリにはファイルがありません。

まず、バージョン管理システムとして Visual SourceSafe を選びます。

➲ Visual SourceSafe にファイルを追加するには

1. Visual SourceSafeに追加するファイル（複数選択可）を選んでください。ファイルフォルダ、プロジェクトフォルダ、ワークスペースフォルダ、またはそれらを混在して選ぶこともできます。プロジェクトフォルダ、またはワークスペースフォルダを選ぶと、システムファイルがファイルリストに追加されます。例えば、プロジェクトフォルダを選ぶと、プロジェクトファイルをファイルリストへ追加します。プロジェクトファイルがチェックアウトされ、

バージョンが最後にロードしたときより新しい場合、そのプロジェクトをリロードしたいかどうかをユーザに問い合わせます。

2. ファイル追加ツールボタン（  ）をクリックするか、 [ツール->バージョン管理->ファイルの追加] メニューオプションを選んでください。

Visual SourceSafe にファイルを追加すると、作業中のディレクトリのローカルバージョンは読み取り専用になります。ファイルが追加されたことの確認や、プロジェクト内のすべてのファイルの状態を表示するには、

1. チェックしたいファイルのプロジェクトフォルダを選んでください。
2. ファイルの状態ツールボタン（  ）をクリックするか、 [ツール->バージョン管理->ファイルの状態] メニューオプションを選んでください。
3. [アウトプット] ウィンドウの [Version Control] タブに各ファイルの状態が表示されます。表示された情報により、ファイルがプロジェクトに追加されたかどうか、ファイルがチェックアウトされたかどうか、また、誰によってチェックアウトされたかがわかります。

9.2 Visual SourceSafe コマンド

以下の 8 つの操作ができます。

- バージョン管理にファイルを追加する
- バージョン管理からファイルを削除する
- 読み取り専用ファイルを取得する
- 読み取り/書き込みファイルをチェックアウトする(編集するため)
- チェックアウトしたファイルをチェックインする（編集後 Visual SourceSafe を更新する）
- ファイルのチェックアウト操作をキャンセルする（編集結果をキャンセルする）*
- ファイルの状態を表示する
- ファイルの履歴を表示する*

*他のコマンドは、ツールバーまたはメニューからアクセスできますが、これらのコマンドは [ツール->バージョン管理] サブメニューからしかアクセスできません。

9.2.1 バージョン管理からファイルを削除する

HEW プロジェクトにファイルが表示（[ワークスペース]ウィンドウの[Projects]タブ）されても、それらのファイルが Visual SourceSafe によって管理されているとは限りません。

➲ Visual SourceSafe からファイルを削除するには

1. Visual SourceSafeから削除するファイルを選んでください。ファイルフォルダ、プロジェクトフォルダ、ワークスペースフォルダ、またはこれらを混在させて選ぶこともできます。
2. ファイル削除ツールバーボタン（  ）をクリックするか、 [ツール->バージョン管理->ファイルの削除] メニューオプションを選んでください。

9.2.2 バージョン管理からファイルの読み取り専用コピーを取得する

Visual SourceSafe はソースファイルを保護して、管理しているファイルの書き込み可能なコピーを一度に一人のユーザだけが取得できるようにします。しかし、どのユーザもすべてのファイルの読み取り専用コピーを取得できます。

➲ Visual SourceSafe から読み取り専用コピーを取得するには

1. Visual SourceSafeから取得するファイルを選んでください。ファイルフォルダ、プロジェクトフォルダ、ワークスペースフォルダ、またはこれらを混在して選ぶこともできます。
2. ファイル取得ツールバーボタン（）をクリックするか [ツール->バージョン管理->ファイルの取得] メニューオプションを選んでください。

9.2.3 バージョン管理からファイルの書き込み可能コピーをチェックアウトする

Visual SourceSafe はソースファイルを保護して、管理しているファイルの書き込み可能コピーを一度に一人のユーザだけが取得できるようにします。チェックアウト操作をすると、Visual SourceSafe からファイルの書き込み可能コピーをローカルドライブに取得します。これは、チェックアウトしようとするファイルがすでに他のユーザによりチェックアウトされていない場合のみ可能です。

➲ Visual SourceSafe からファイルの書き込み可能コピーをチェックアウトするには

1. Visual SourceSafeからチェックアウトしたいファイルを選んでください。ファイルフォルダ、プロジェクトフォルダ、ワークスペースフォルダ、またはこれらを混在して選ぶこともできます。
2. ファイルチェックアウトツールバーボタン（）をクリックするか [ツール->バージョン管理->ファイルのチェックアウト] メニューオプションを選んでください。
3. 動作が完了すると、ファイルの名前の横に印が付き、チェックアウトされたことを示します。

9.2.4 バージョン管理にファイルの書き込み可能コピーをチェックインする

Visual SourceSafe はソースファイルを保護して、管理しているファイルの書き込み可能コピーを一度に一人のユーザだけが取得できるようにします。チェックアウト操作をすると、Visual SourceSafe からファイルの書き込み可能コピーをローカルドライブに取得します。チェックアウトしたファイルを編集してチェックインすると、編集結果を他のユーザが見られるようになります。

➲ 編集した Visual SourceSafe のファイルをチェックインするには

1. Visual SourceSafeに再びチェックインするファイルを選んでください。複数のファイルを選ぶことができます。ファイルフォルダ、プロジェクトフォルダ、ワークスペースフォルダ、またはこれらを混在して選ぶこともできます。
2. ファイルチェックインツールバーボタン（）をクリックするか [ツール->バージョン管理->ファイルのチェックイン] メニューオプションを選んでください。

9.2.5 チェックアウト操作を取り消す

Visual SourceSafe はソースファイルを保護して、管理しているファイルの書き込み可能コピーを一度に一人のユーザだけが取得できるようにします。チェックアウト操作をすると、Visual SourceSafe からファイルの書き込み可能コピーをローカルドライブに取得します。チェックアウトしたファイルを編集してチェックインすると、編集結果を他のユーザが見られるようになります。しかし、もしチェックアウト操作を誤って行った場合、または、必要なくなった場合、チェックアウト操作を取り消すことができます。

➲ Visual SourceSafe からのファイルのチェックアウト操作を取り消すには

1. 以前のチェックアウト操作を取り消したいファイルを選んでください。ファイルフォルダ、プロジェクトフォルダ、ワークスペースフォルダ、またはこれらを混在して選ぶこともできます。
2. [ツール->バージョン管理->チェックアウトの取り消し] メニューオプションを選んでください。

9.2.6 ファイルの状態を表示する

HEW プロジェクトにファイルが表示されても（[ワークスペース] ウィンドウの[Projects] タブ）、ファイルが Visual SourceSafe に管理されているとは限りません。Visual SourceSafe に管理されているファイルのうち、チェックインされたり、チェックアウト（ユーザが編集するため）されたりするものがあります。状態コマンドは、現在のファイルの状態を表示します。

➲ Visual SourceSafe のファイルの状態を表示するには

1. 状態を表示するファイルを選んでください。複数のファイルを選ぶこともできます。ファイルフォルダ、プロジェクトフォルダ、ワークスペースフォルダ、またはこれらを混在して選ぶことができます。
2. ファイルの状態ツールバーボタン（  ）をクリックするか [ツール->バージョン管理->ファイルの状態] メニューオプションを選んでください。
3. ファイル名の横に青い印があるとき、そのファイルはチェックアウトされていることを示します。

9.2.7 ファイル履歴を表示する

Visual SourceSafe はプロジェクトのファイルへの編集を管理します。ファイルが最初にプロジェクトに追加されたときからの編集内容の完全なファイル履歴を表示できます。

➲ Visual SourceSafe のファイル履歴を表示するには

1. ファイル履歴を表示するファイルを選んでください。複数のファイルを選ぶことができます。ファイルフォルダ、プロジェクトフォルダ、ワークスペースフォルダ、またはこれらを混在して選ぶことができます。
2. [ツール->バージョン管理->Show History] メニューオプションを選んでください。

9.3 Visual SourceSafe 統合化オプション

[ツール->バージョン管理->構成…]を選ぶことにより、履歴コマンドや状態コマンドの表示方法を変更できます。

履歴コマンドの結果をダイアログボックスで表示するには、[Display dialog box for history] チェックボックスをチェックしてください。履歴コマンドの結果を[アウトプット] ウィンドウの[Version Control]タブで表示するには、[Display dialog box for history] チェックボックスのチェックをはずしてください。

状態コマンドの結果をダイアログボックスで表示するには、[Display dialog box for file status]チェックボックスをチェックしてください。状態コマンドの結果を[アウトプット] ウィンドウの[Version Control]タブで表示するには、[Display dialog box for file status]チェックボックスのチェックをはずしてください。

10. ネットワーク機能

10.1 概要

High-performance Embedded Workshop は、ネットワークを介してワークスペースやプロジェクトを共有できます。したがって、ユーザは共有したプロジェクトを同時に操作してお互いの変更を見ることができます。このシステムは、バージョン制御と連動して使用できます。このシステムの主な特長は、ほかのすべてのユーザがプロジェクトをリロードして変更を無効にすることなく、各ユーザがワークスペースやプロジェクトを変更したり更新したりできることです。

このシステムは、ネットワーク上のひとつのコンピュータをサーバマシンとして使うことで実現します。ほかのすべてのクライアントマシンはサーバマシンが提供するサービスを利用できます。例えば、クライアントマシンのひとつが新規ファイルを追加すると、サーバマシンに通知されます。すると、サーバマシンはほかのすべてのクライアントマシンに追加を通知します。この構成を図 10.1 に示します。

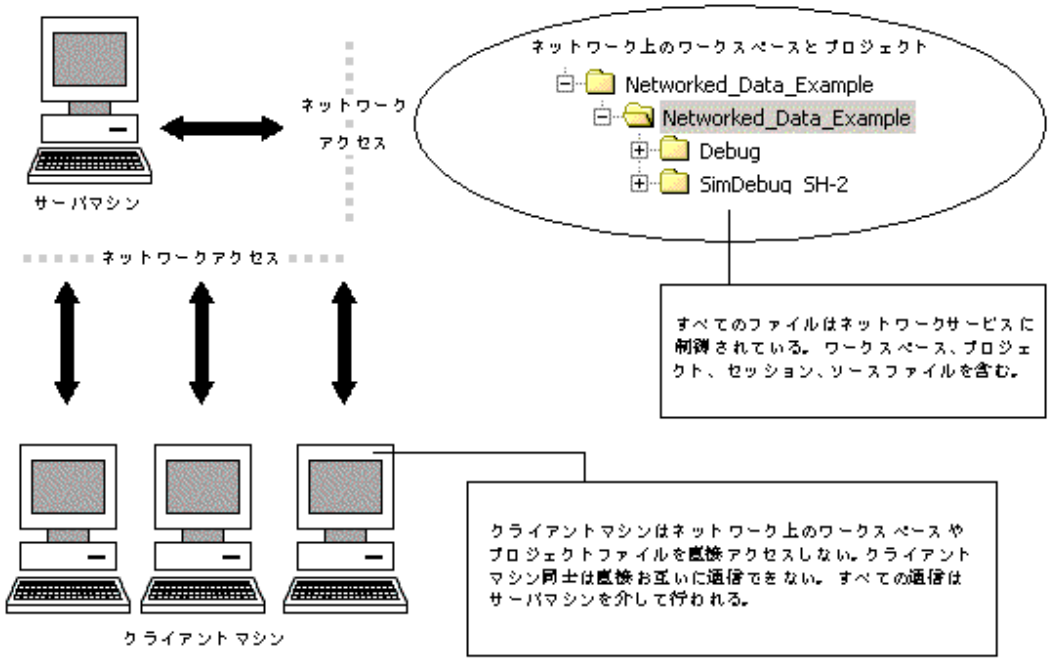


図 10.1: ネットワークデータベースの構成

ネットワーク機能によってユーザはファイルへのアクセス権を与えられます。これによって、プロジェクトアドミニストレータから書き込みを許可されたユーザのみが、プロジェクトやソースファイ

ルを変更できるようになります。アドミニストレータは、各ユーザの書き込み権を、プロジェクトの各自のエリアに制限して、ほかのエリアを読み出し専用にすることができます。こうして、各ユーザによるプロジェクトのほかの部分への競合や破壊を未然に防きます。これらの制限は異なるレベルに設定できます。詳細は、この章で後述します。

【注】 動作によっては、あるクライアントマシンが実行している間は、他のクライアントマシンでロックされていることがあります。例えば、ひとつのマシンがツールチェインオプションを変更している間、ほかのすべてのマシンではこのデータが読み出し専用になります。

【注】 ネットワーク機能を使うと、HEW の性能は低下します。小規模なチームで作業する場合は、単一ユーザモードとバージョン制御を使うことを推奨します。

10.1.1 ネットワークアクセスを使う

➲ ネットワークアクセスを使うには

1. [ツール->オプション] を選んでください。HEWの[オプション]ダイアログボックスが表示されます。
2. [ネットワーク] タブを選んでください（図10.2）。
3. [ネットワークデータアクセス有効化] チェックボックスをチェックしてください。パスワードなしでアドミニストレータがシステムに追加されます。
   アドミニストレータのみが、ユーザを追加したり、ユーザアクセス権を変更したりできます。アドミニストレータは最高のレベルのアクセス権を持ちます。
4. [ネットワーク] ダイアログボックスで、アドミニストレータはパスワードを設定しなければなりません。以下に説明します。

10.1.2 アドミニストレータユーザのパスワードを設定する

➲ パスワードを設定するには

1. 前述の手順に従って操作してください。
2. [パスワード] ボタンをクリックしてください。ネットワークデータアクセスを有効にしたとき、このボタンも有効になります。
3. [パスワード変更] ダイアログボックスが表示されます（図10.3）。
4. 上のフィールドでは、ユーザ名は読み出し専用です。この場合、Adminになります。
5. 両方のフィールドに新規パスワードを入力して、 [OK] ボタンをクリックしてください。
6. [ツール -> オプション] ダイアログボックスの [ネットワーク] タブにユーザとパスワードが設定されます。
7. これで、 [ツール -> オプション] ダイアログボックスを閉じることができます。
8. ダイアログボックスを閉じると、ワークスペースを保存して再び開くかどうか質問されます。これは、共有アクセスモードでワークスペースを再び開く必要があるためです。変更を保存しないと、変更は無効になります。
9. ワークスペースを再び開くと、システムに再びログインするかどうかを決めるダイアログボックスが開きます。ログインすると、現在のあなたのアクセス権を示すダイアログボックスが表示されます。例えば、アドミニストレータユーザの場合、レベルはアドミニストレータになります。このダイアログボックスを閉じると、HEWサーバウィンドウが開き、ネットワーク機能が使えるようになります。

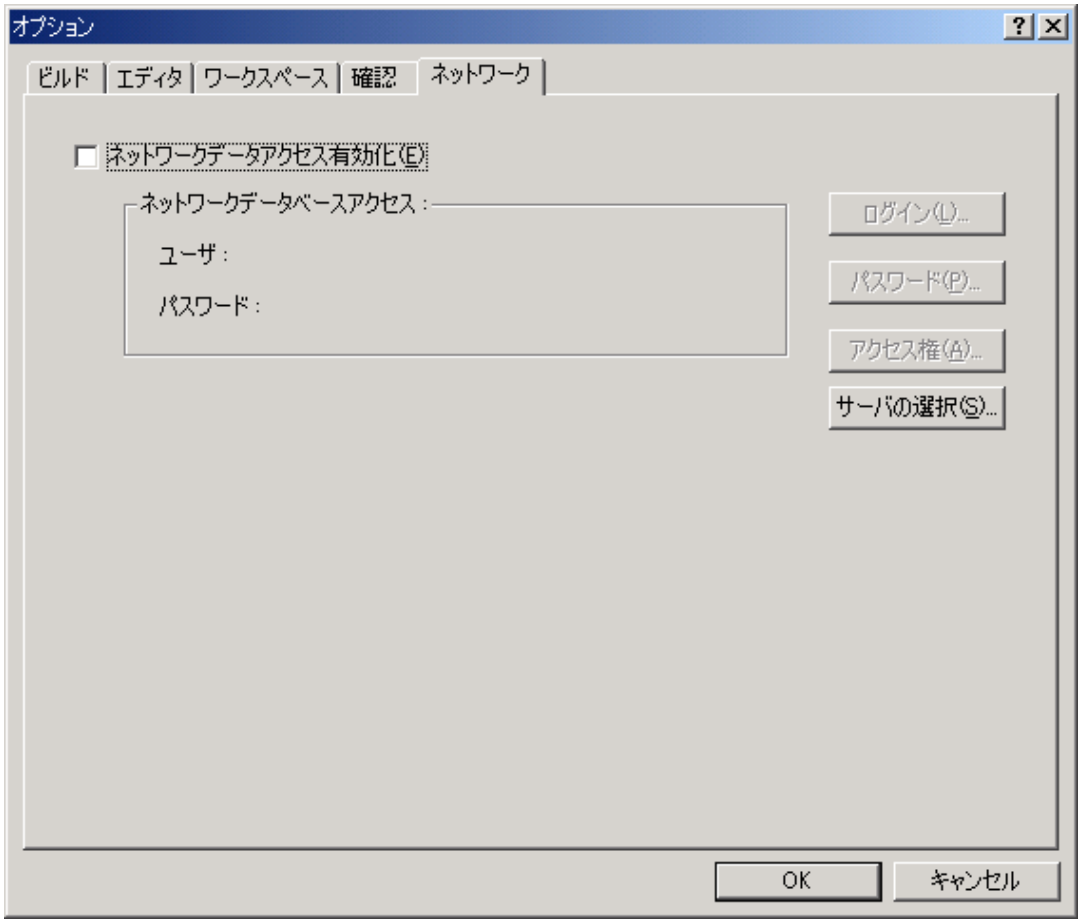


図 10.2: [**ネットワーク**]**タブの初期設定**

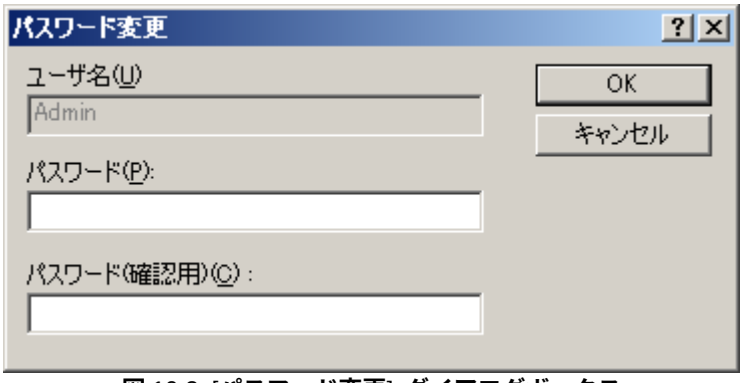


図 10.3: [**パスワード変更**] **ダイアログボックス**

10.1.3 新規ユーザを追加する

ネットワークデータベースの初期設定では、アドミニストレータユーザとゲストユーザをシステムに追加します。HEW システムでは、以下のアクセス権のレベルを設定できます。

- アドミニストレータ
  HEW のすべての機能にフルアクセスできます。プロジェクトからユーザを削除したり、アクセス権レベルを変更したりできます。ワークスペース、プロジェクトファイル、ソースファイルを変更できます。

- 読み出し/書き込みフルアクセス
  ワークスペース、プロジェクトファイル、ソースファイルを変更できます。ユーザのアクセス権レベルは変更できません。

- 読み出し/書き込みファイルアクセス
  ソースファイルのみ変更できます。すべてのプロジェクト設定を表示できますが、変更できません。

- 読み出し専用
  すべてのソースファイルとプロジェクトファイルが表示できます。変更はできません。

どのユーザも、ネットワーク上で使用できるプロジェクトを開くときにはユーザ名とパスワードを入力しなければなりません。入力しないと、アクセス権は与えられません。入力すると、ユーザには上記のアクセス権レベルのうちのいずれかのレベルを与えられます。

➲ 新規ユーザを追加するには

1. アドミニストレータのアクセス権をもつユーザがログインします。この詳細は、上記を参照してください。
2. [ツール->オプション] を選んでください。HEWの[オプション]ダイアログボックスが表示されます。
3. [ネットワーク] タブを選んでください（図10.2）。
4. [アクセス権...] ボタンをクリックしてください。図10.4に示すダイアログボックスが表示されます。
5. [追加…] ボタンをクリックしてください。[新規ユーザ追加]ダイアログボックスが表示されます。あなたをアドミニストレータユーザとして新規のログイン名とパスワードを追加できます。通常、パスワードはデフォルトのテキストか空白のままで設定します。次に、[OK] ボタンをクリックしてください。
6. [OK] ボタンをクリックすると、ユーザは読み出し専用のアクセス権レベルで追加されます。アクセス権レベルを変更するには、ユーザ名を選んで必要なラジオボタンをクリックしてください。[OK] ボタンをクリックすると、アクセス権レベルの変更が保存されます。

➲ 既存ユーザを削除するには

1. アドミニストレータのアクセス権をもつユーザがログインします。この詳細は、上記を参照してください。
2. [ツール->オプション] を選んでください。HEWの[オプション]ダイアログボックスが表示されます。
3. [ネットワーク] タブを選んでください（図10.2）。
4. [アクセス権…] ボタンをクリックしてください。図10.4に示すダイアログボックスが表示されます。

5. 削除するユーザ名を選んでください。
6. [削除] ボタンをクリックしてください。
7. [OK] ボタンをクリックするとアクセス権の変更が保存されます。

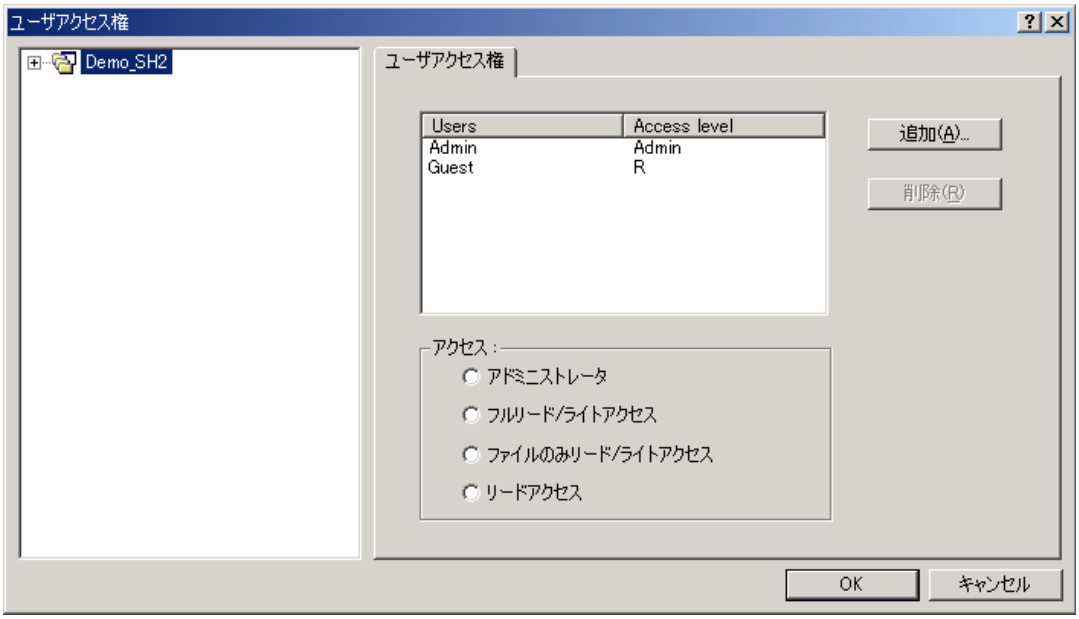


図 10.4: **[ユーザアクセス権] ダイアログボックス**

10.1.4 パスワードを変更する

➲ パスワードを変更するには

1. パスワードを変更するHEWネットワークデータベースにログインして、 [ツール -> オプション] を選んでください。[オプション]ダイアログボックスが表示されます。
2. [ネットワーク] タブを選んでください（図10.2）。
3. [パスワード] ボタンをクリックしてください。
4. 新しいパスワードを入力して 2 番目のエディットボックスで確認してください。
5. [OK] ボタンをクリックしてください。
6. 次に、 [OK] ボタンをクリックするとパスワードの変更が保存されます。

10.1.5 ネットワーク HEW サービスを利用する

最初にネットワーク上のプロジェクトに接続するとき、HEW は自動的に正しいネットワーク HEW サービスを探して接続します。これはマシン名で定義されます。マシン名でワークスペースのサービスが見つからないとき、図 10.5 に示すダイアログボックスが表示されます。サービスのある場所を入力するか参照して、[OK] ボタンをクリックしてください。設定するマシンをサーバマシンとして使用する場合は、ラジオボタンをデフォルトの設定(ローカル)のままにしてください。

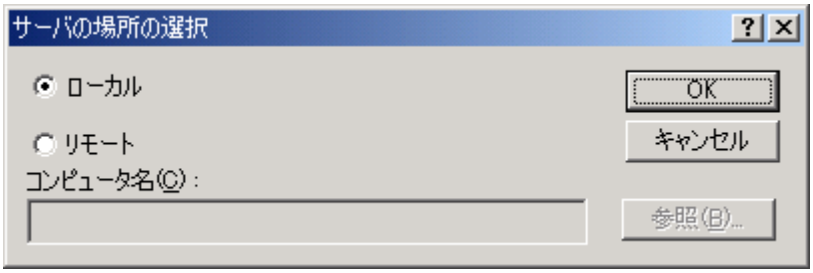


図 10.5: **マシン名の選択**

ワークスペースのサーバマシンとして使用していたマシンを他のマシンに接続しようとすると、図 10.6 のメッセージが表示されます。[OK] ボタンをクリックすると新しいマシンに接続します。

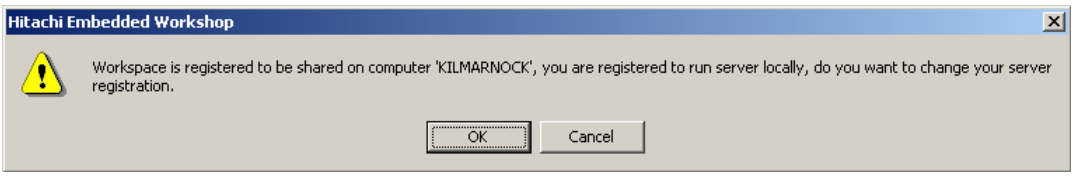


図 10.6: **マシン名の選択**

【注】 ネットワークサービスを有効にして複数の HEW **ワークスペースを実行している場合、ユーザは一度にひとつのワークスペースにしかアクセスできません。例外は、同じマシンがすべてのネットワークワークスペースを提供している場合です。**

11. 相違点の表示

High-performance Embedded Workshop には、相違点の表示があります。ドライブのローカルファイルやバージョン制御システムのファイルを比較して相違点を詳細に表示します。HEW のバージョン 3.0 以降の Visual SourceSafe にこの機能があります。

[差分]ウィンドウを起動するには、２通りの方法があります。１つめは、[ツール]メニューの[差分の表示...]メニュー項目を選びます。２つめは、[ワークスペース]ウィンドウのポップアップメニューの[差分の表示]メニュー項目です。ポップアップメニューを使うと、現在のファイルの選択が自動的にエディットボックスに追加されます。[差分の表示]メニュー項目をクリックすると、図 11.1 に示すダイアログボックスが表示されます。

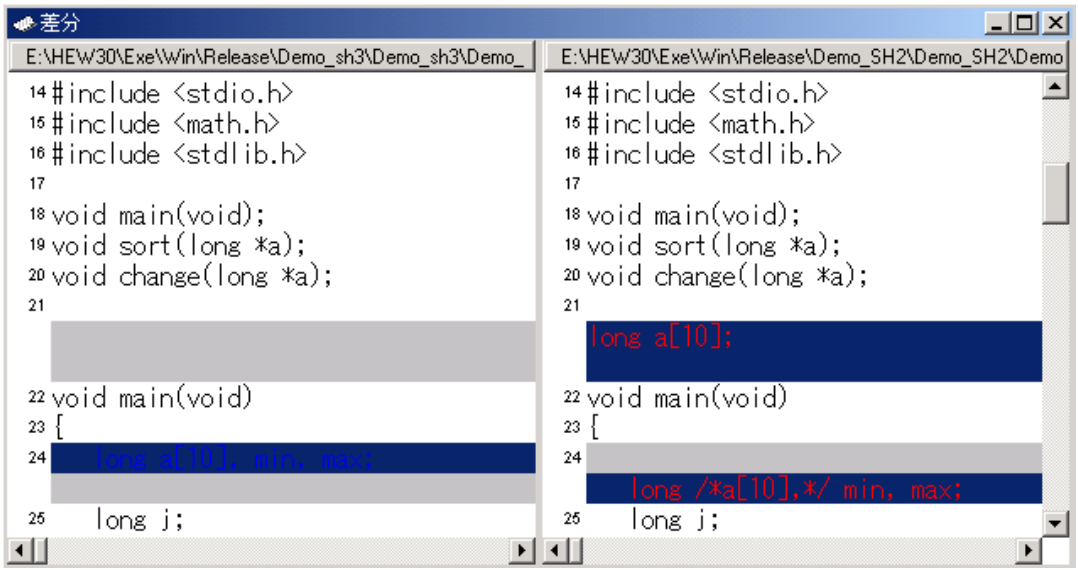


図 11.1: 相違点の表示

➲ ローカルドライブの２つのファイルの相違点を比較するには

1. [ツール->差分の表示...] を選んでください。図11.2の[ファイルの比較]ダイアログボックスが表示されます。
2. [ドライブ上のファイルと比較] ラジオボタンが有効であることを確認してください。
3. 比較する２つのファイルを入力してください。前回の相違点の比較を選ぶか新しいファイルをブラウズできます。
4. [詳細…] ボタンをクリックすると、図11.3のダイアログボックスが表示されます。空白を考慮せずに相違点の比較ができます。この[詳細オプション]ダイアログボックスを終了するには[OK]ボタンをクリックしてください。
5. [比較] ボタンをクリックしてください。
6. 相違点が表示されます。比較する２つのファイルは左右に分割された表示にロードされま

す。ファイル名は各ウィンドウの上に表示されます。

➲ ローカルファイルと SourceSafe のファイルの相違点を比較するには

1. SourceSafeが有効であることを確認してください。また、ファイルはバージョン制御システムに追加してください。
2. [ツール -> 差分の表示...] を選んでください。図11.2の[ファイルの比較]ダイアログボックスが表示されます。
3. [バージョン管理されているファイルとの比較] ラジオボタンが有効になっていることを確認してください。
4. 比較する１つめのファイルを入力してください。前回の相違点の比較を選ぶか新しいファイルをブラウズできます。
5. [詳細…] ボタンをクリックすると図11.3のダイアログボックスが表示されます。空白を考慮せずに相違点の比較ができます。この[詳細オプション]ダイアログボックスを終了するには[OK]ボタンをクリックしてください。
6. [比較] ボタンをクリックしてください。
7. 図11.1のように、相違点が表示されます。

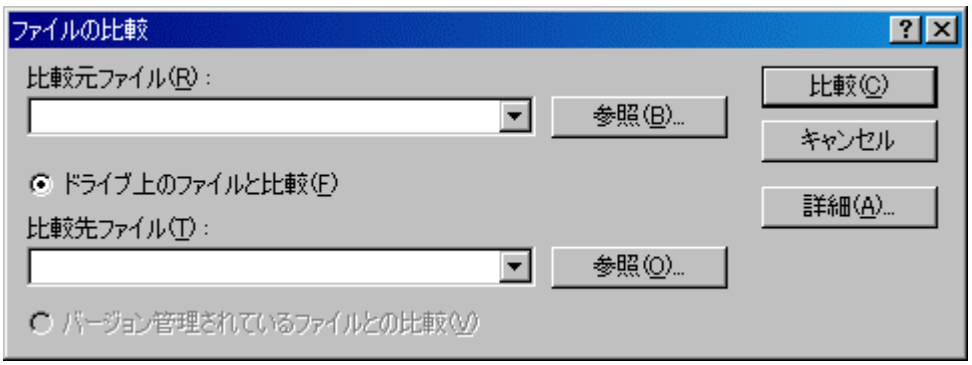


図 11.2: [**ファイルの比較**]**ダイアログボックス**

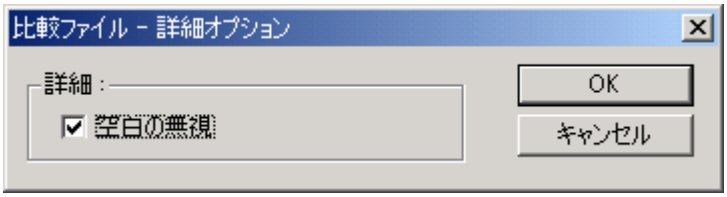


図 11.3: [**詳細オプション**] **ダイアログボックス**

相違点の表示のアクセスには２通りの方法があります。図 11.4 に示す相違点の表示用のツールバーを使用するか、または、[差分]ウィンドウ上でマウスを右クリックすると、図 11.5 のポップアップメニューが表示されます。

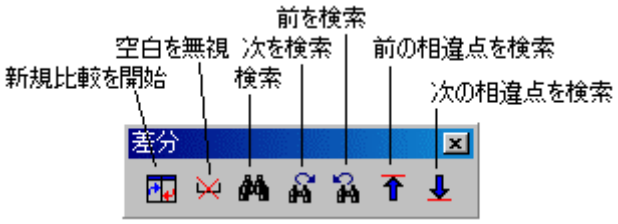


図 11.4: **[差分] ツールバー**

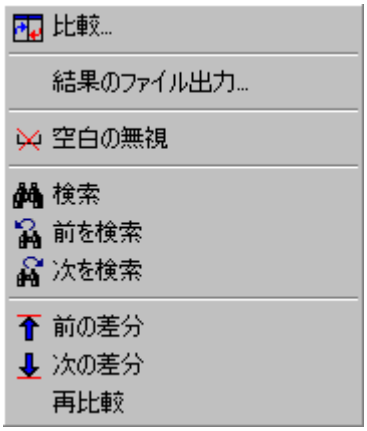


図 11.5: **[差分] ポップアップメニュー**

各メニュー項目の機能を以下に説明します。

- 比較
  新しい[ファイルの比較]ウィンドウを開きます。新しいファイルを比較して相違点を表示できます。
- 結果のファイル出力
  ダイアログボックスが開き、現在の相違点の結果をテキストフォーマットで保存することができます。
- 空白の無視
  空白文字を無視します。このメニュー項目は[詳細オプション]ダイアログボックスの[空白の無視]オプションから選択することもできます。
- 検索
  標準の[検索]ダイアログボックスを表示します。HEW エディタと同じダイアログボックスです。
- 次を検索
  検索項目に当てはまる次の前方の文字列を検索します。
- 前を検索
  検索項目に当てはまる次の後方の文字列を検索します。
- 前の差分
  次の後方の相違点に自動的にジャンプします。
- 次の差分
  次の前方の相違点に自動的にジャンプします。

- 再比較
  表示をリフレッシュして再び手動で相違点の比較を実行します。この機能は、どちらかのファイルを比較後、変更したときに便利です。

[差分]ウィンドウでは、表示の色を変えることができます。この手順を以下に示します。

➲ [差分]ウィンドウで表示する色を変更するには

1. [ツール -> 表示形式...] を選んでください。図11.6 に示す[表示形式]ダイアログボックスが表示されます。
2. [差分] を拡張表示してください。ダイアログボックスは、図11.6のようになります。
3. 変更するカテゴリを選んでください。[カラー]タブで変更できます。選択できるカテゴリは、左側で異なる行、左側で移動した行、右側で異なる行、右側で移動した行です。
4. [OK] ボタンを押すと変更が有効になります。

図 11.6: **[表示形式] ダイアログボックス**

12. テクニカルサポート

High-performance Embedded Workshop には、いくつかの統合されたテクニカルサポート機能があります。

➲ HEW 製品のバージョンアップやサービスパックを確認するには

1. [ヘルプ -> テクニカルサポート -> アップデートのWeb確認] を選んでください。
2. デフォルトのウェブブラウザが起動して、あなたの地域のHEWダウンロードページが表示されます。
3. HEWのバグの修正や新規機能の追加の更新情報を確認してください。

ときには、HEW アプリケーションで予期せぬ問題が起きるかもしれません。問題が起きてアプリケーションが破壊されると、自動的に HEW バグトラッキングプログラムが起動します。これにより、バグの情報を集約でき、様々な方法でテクニカルサポートに送ることができます。このバグトラッキングプログラムは手動で起動することもできます。この方法を以下に説明します。

➲ HEW バグ情報を作成するには

1. [ヘルプ -> テクニカルサポート -> 不具合レポートの作成] を選んでください。
2. HEWシステムから詳細情報が生成されます。これには多少，時間がかかるかもしれません。その後、図12.1のような、[Submit a Bug Report]ダイアログボックスが表示されます。
3. 大きなエディットボックスに、起きた問題に関する情報を追加することができます。
4. 報告を作成したら、[How would you like to submit the report?] ドロップリストで報告を提出する方法を選ぶことができます。印刷、電子メール、ディスクに保存の3種類があります。
5. [Submit] ボタンをクリックすると選んだ方法で報告を提出することができます。

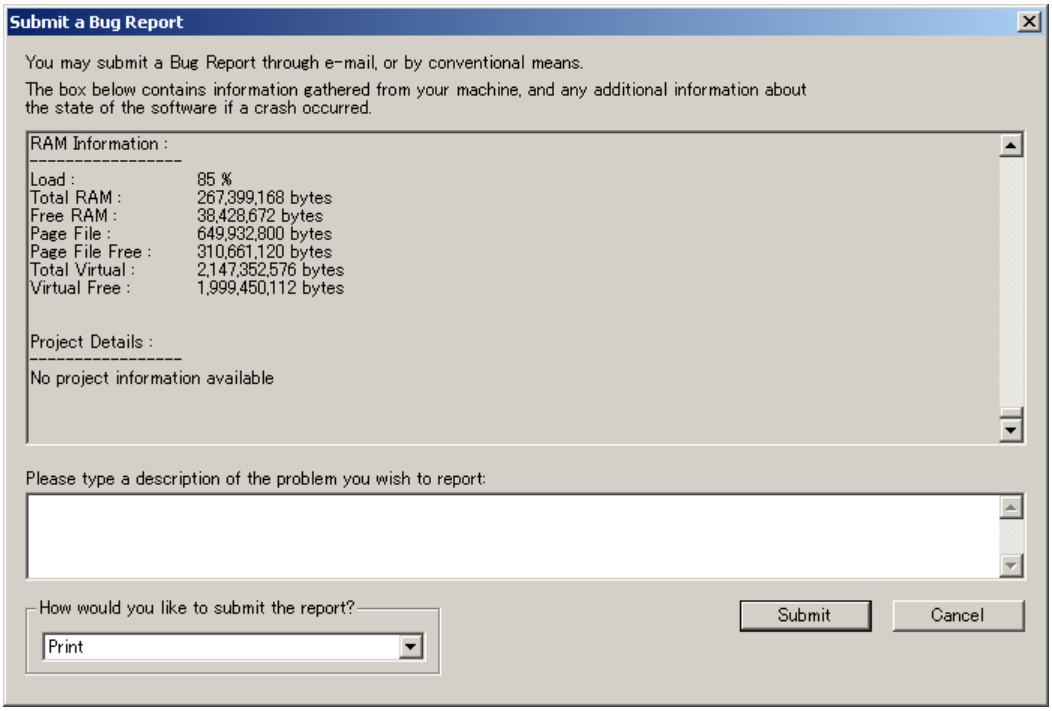


図 12.1: [Submit a Bug Report]**ダイアログボックス**

13. ナビゲーション機能

High-performance Embedded Workshop には、いくつかの統合されたナビゲーション機能があります。[Navigation]ウィンドウは[Projects]ウィンドウ、および[Templates]ウィンドウの横にあります（図 13.1）。

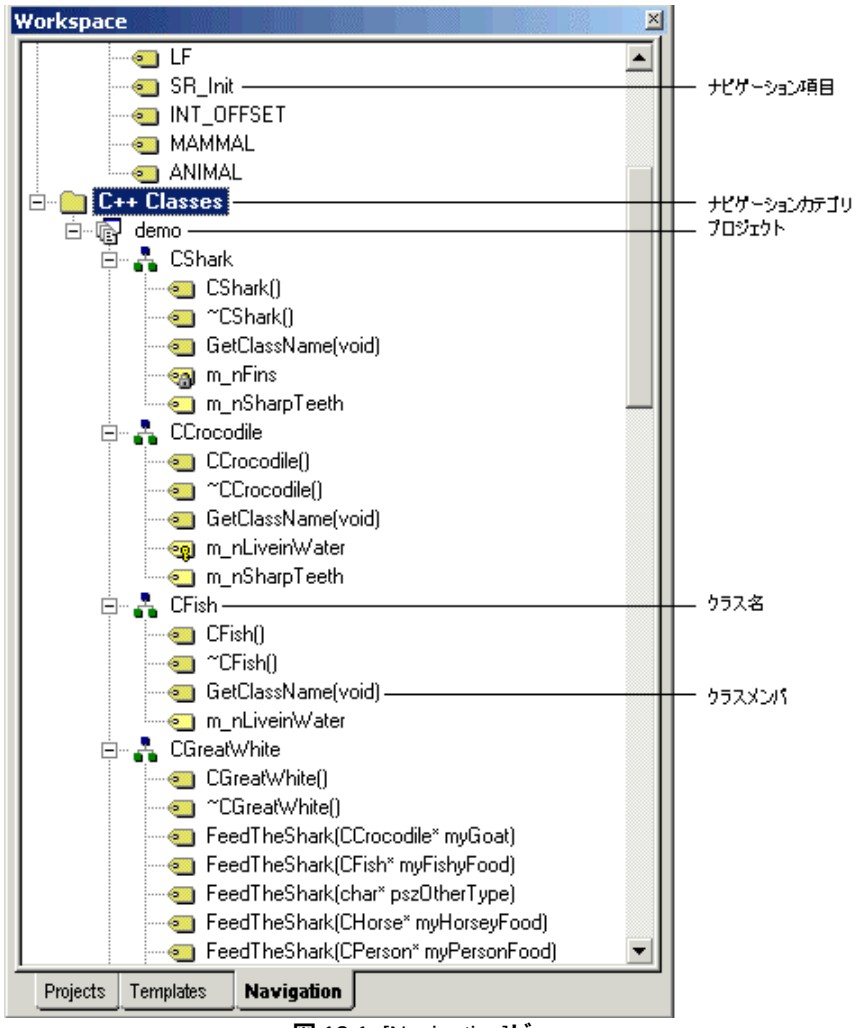


図 13.1: [Navigation]**ビュー**

[Navigation]ビューではサポートされているすべてのナビゲーションの種類を表示します。HEW3.0 以降では、以下のナビゲーション機能を標準でサポートしています。

- C Defines：C, C ソースファイルのすべての#define の表示

- C Functions：C ソースファイルのすべての ANSI C 標準関数の表示
- C++ Classes：C++ソースファイルのすべてのクラス、関数、メンバの表示

各カテゴリは、ビューの上位レベルに表示されます。各カテゴリの下にワークスペースの各プロジェクトが表示されます。特定のプロジェクトに属する項目はプロジェクトのアイコンの下に表示されます。

特定のナビゲーションカテゴリの情報を必要としない場合はスキャンを無効にすることができます。

➲ ナビゲーションカテゴリを無効にするには

1. [Navigation] ウィンドウ上でマウスの右ボタンをクリックしてください。図13.2のポップアップメニューが表示されます。
2. [カテゴリの選択…] メニュー項目を選んでください。
3. 図13.3に示すダイアログボックスが表示されます。
4. 定義を参照する必要のないカテゴリのチェックをはずしてください。
5. [OK] ボタンをクリックしてください。

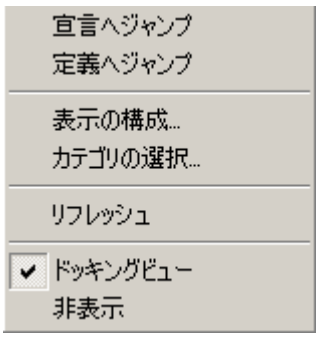


図 13.2: [C++ **ナビゲーション]ポップアップメニュー**

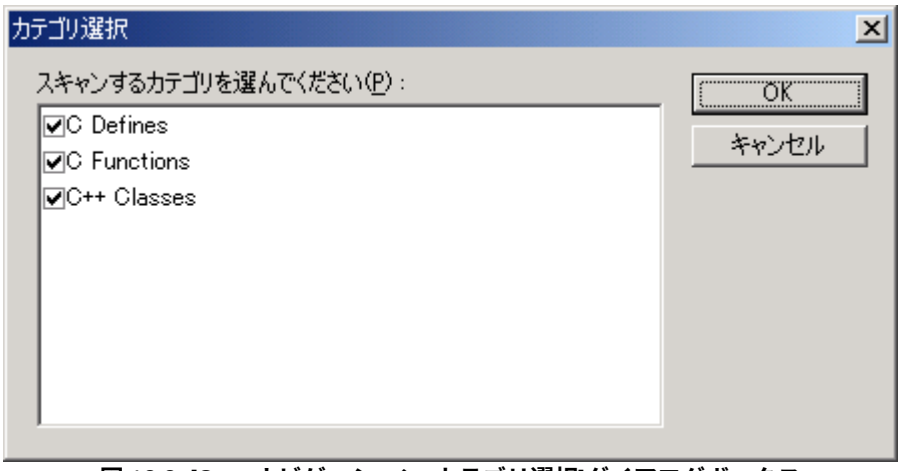


図 13.3: [C++ ナビゲーション カテゴリ選択]ダイアログボックス

【注】 ファイルがスキャンされるにしたがい、ナビゲーション項目が表示されます。つまり、ファイルが多数あると、ナビゲーションビューの更新を終了するのに時間がかかります。
ファイルは、保存するときに再びスキャンされます。つまり、ファイルを保存するまで、新しいクラスや関数のナビゲーション情報は表示されません。

13.1 C++ナビゲーション機能

C++ナビゲーション機能は、サポートする３つのナビゲーション機能の中で最も複雑です。C++ソースファイルの表示で以下の構成をサポートします。情報の基本構成を図 13.4 に示します。

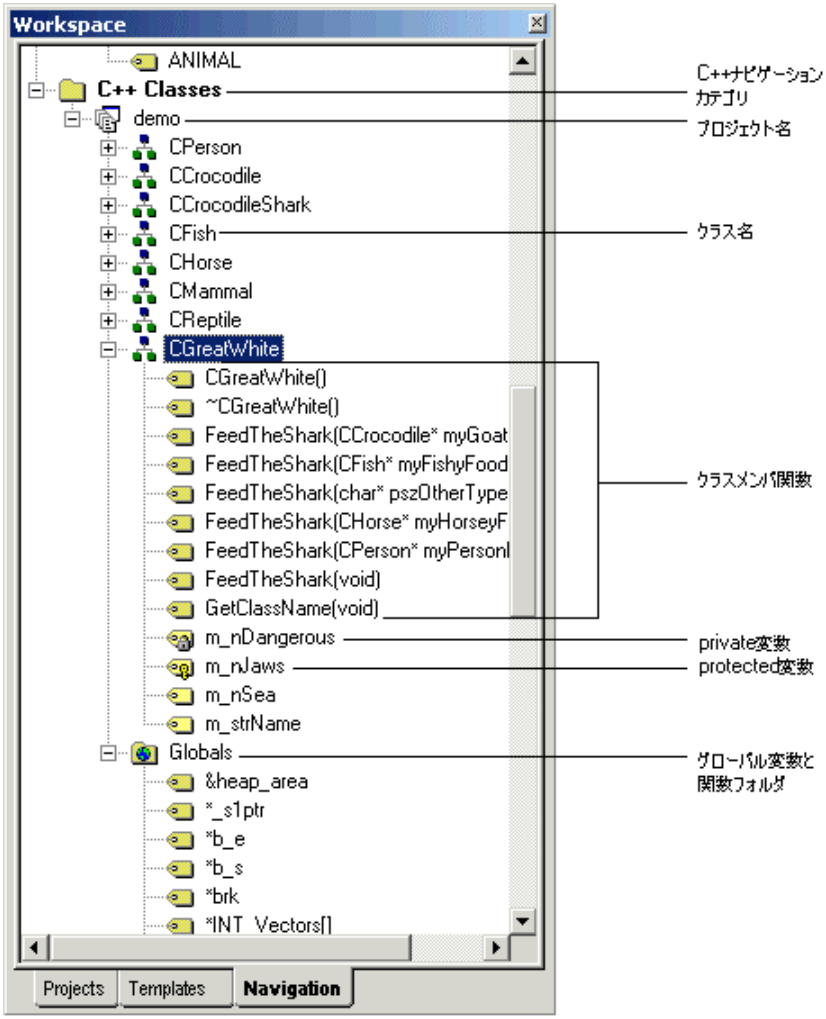


図 13.4: C++**ナビゲーション情報**

C++ナビゲーション表示では、アイコンの示す関数や変数の種類を説明するために多くのアイコンを使います。これらを図 13.5 に示します。

アイコン	説明
	public関数
	protected関数
	private関数
	public変数
	protected変数
	private変数

図 13.5: C++**ナビゲーションアイコン**

ナビゲーションビューでは、ソースコードを効率良く見て回ることができます。デフォルトで、ナビゲーション項目をダブルクリックすると、関連するナビゲーション項目の定義にジャンプします。通常、定義はソースファイルにあり、宣言はヘッダファイルにあります。

このデフォルト動作は、[表示の構成]ダイアログボックスで変更できます。[表示の構成]ダイアログボックスではナビゲーションビューでのデータの表示方法を変えることができます。

➲ C++ナビゲーション表示データを構成するには

1. [Navigation] ウィンドウのC++ナビゲーション項目上でマウスを右クリックしてください。図13.2に示すポップアップメニューが表示されます。
2. [表示の構成…] メニュー項目を選んでください。
3. 図13.6に示すダイアログボックスが表示されます。
4. 変更する項目を決めて [OK] ボタンをクリックしてください。

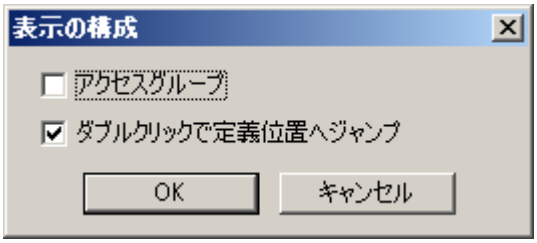


図 13.6: [**表示の構成**]**ダイアログボックス**

[表示の構成]ダイアログボックスで使用できる項目を以下に示します。

- アクセスグループ
  ナビゲーション表示で、public、private、protected メンバ変数と関数をグループ化します。
- ダブルクリックで定義位置にジャンプ
  デフォルトでこのオプションはチェックされています。このオプションをオフにすると、ダイアログボックスが表示され、どこにジャンプするかを質問します。HEW ではなくユーザが位置を決めます。このダイアログボックスを図 13.7 に示します。

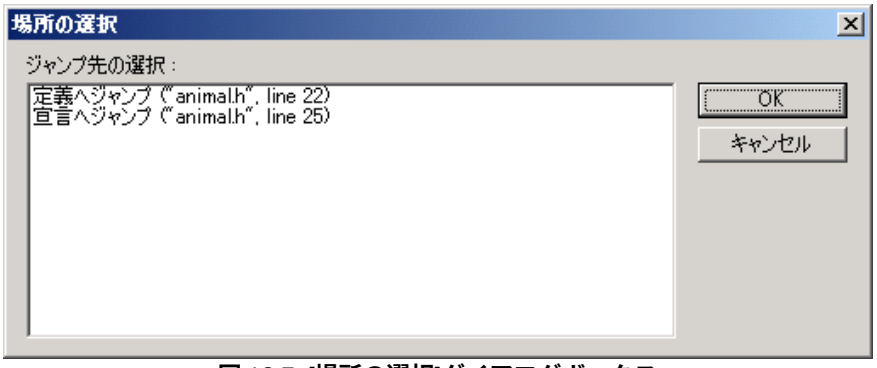


図 13.7: [場所の選択]ダイアログボックス

他の便利な機能としては、特定の範囲の基底クラス、または抽出クラスを表示します。

➲ 基底クラス、抽出クラスを表示するには

1. ナビゲーションビューでクラスを選んでください。
2. マウスを右クリックして、ポップアップメニューを表示してください。
3. [派生クラスの表示]メニュー項目をクリックすると、選択した範囲の導出クラスが表示されます。[基底クラス表示]メニュー項目をクリックすると、選択した範囲の基底クラスが表示されます。
4. 選択により、拡張ツリー形式で選択されたクラス構造がダイアログボックスに表示されます。
5. 必要な情報を取得後、[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。

13.2 C 関数と#define ナビゲーション機能

この機能では、ナビゲーション表示に関数と#define 定義を追加します。ナビゲーション表示に追加すると、ラベルをダブルクリックしてこれらの定義にジャンプできるようになります。

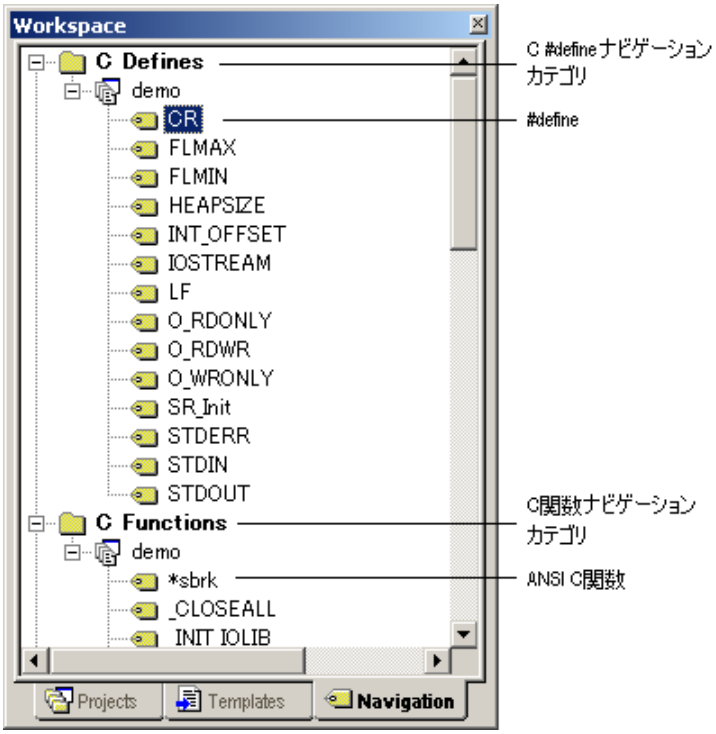


図 13.8: C 関数と#define ナビゲーション情報

14. スマートエディタ

High-performance Embedded Workshop のもうひとつの機能にスマートエディタがあります。これはデフォルトではすべてのC++ソースファイルで有効です。これにより、C++クラスやメンバ関数を使っているとき、HEW エディタは C++ナビゲーション情報にアクセスして自動補完ヘルプを提供します。

➲ スマートエディタの状態を表示するには

1. [ツール->オプション] を選んでください。
2. [オプション] ダイアログボックスの [エディタ] タブを選んでください。
3. 図14.1に示すダイアログボックスが表示されます。[C++用Smart-edit機能]チェックボックスをチェックしてください。
4. [OK] ボタンをクリックしてください。

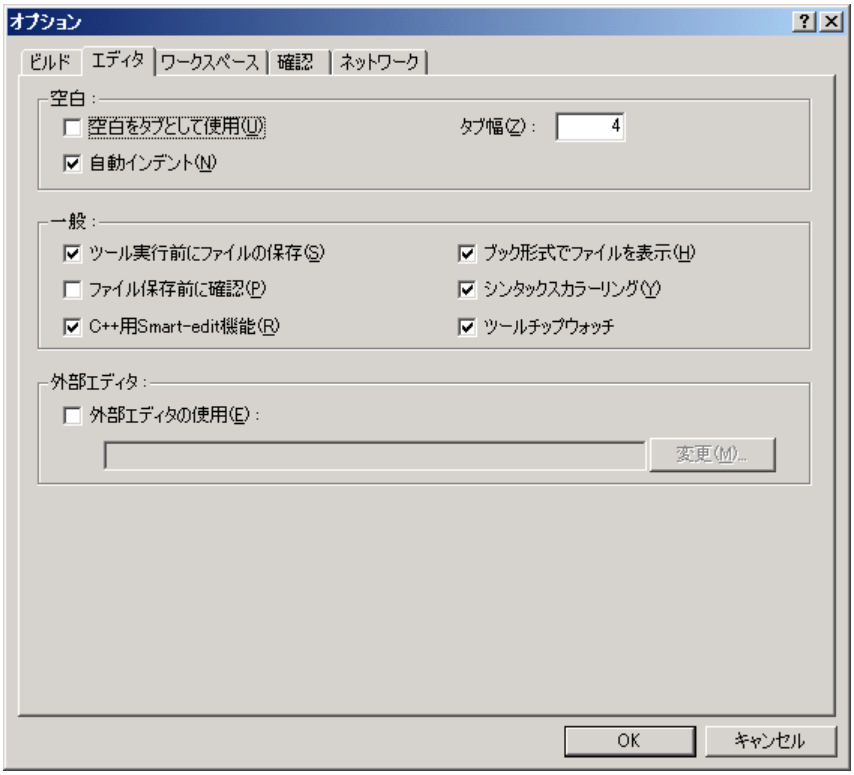


図 14.1: **オプションダイアログボックス　エディタ　タブ**

このオプションをオンにすると、操作している C++ファイルでスマートエディタ機能が有効になります。

【注】 **C++クラスナビゲーションカテゴリがオフのとき、HEW のスマートエディタ機能は無効です。C++クラスナビゲーションの詳細は「13. ナビゲーション機能」を参照してください。**

通常の使い方では、以下のエディタ動作によりスマートエディタ機能が表示されます。

- オブジェクトを使っていて「.」や「->」でメンバをアクセスしているときは、ポップアップメニューが表示され、入力するより効率良くメンバを選ぶことができます。入力中、ポップアップメニューはユーザが押下したキーを覚えているため、選びやすくなります。リターンキーを押すと、選択したメンバが追加されます。「::」を使っているときもこのポップアップメニューが表示されます（図 14.2 参照）。

- C++関数を使っているときは、最初に開きのかっこを入力すると、図 14.3 のダイアログボックスが表示されます。このダイアログボックスにより、現在のオブジェクトでどの関数が使用できるかわかります。関数を選ぶと自動的に残りのパラメータが入力されます。

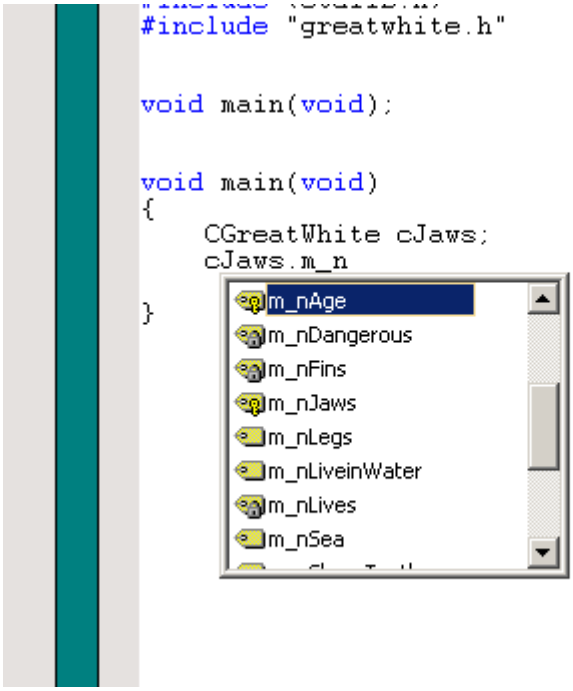


図 14.2: **スマートエディタのメンバの選択**

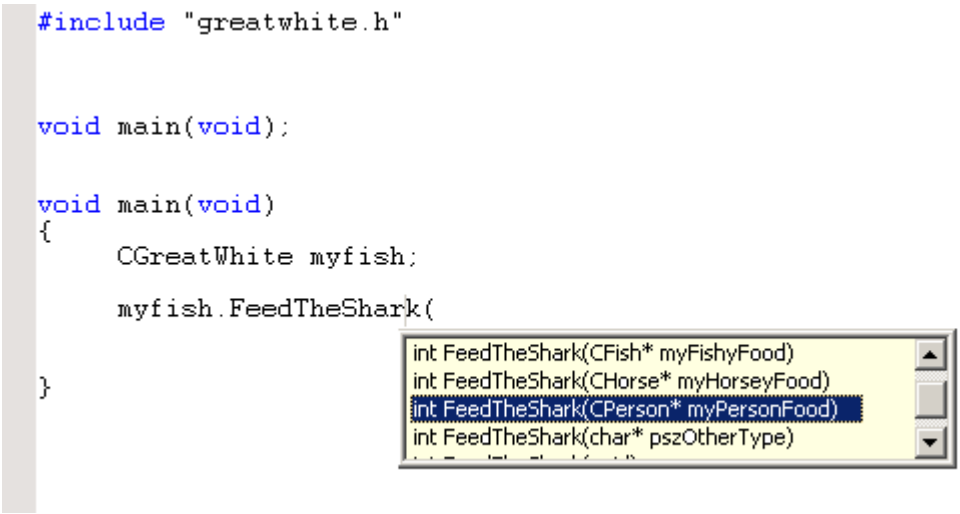


図 14.3:**スマートエディタの関数の選択**

【注】 **スマートエディタの自動補完は様々な理由で動作しないことがあります。例えば、ファイルがナビゲーション機能によりスキャンされていなければなりません。スキャンするには、ファイルが保存されている必要があります。また、**#define **が考慮されていないため、コードの冗長な部分が表示される可能性があります。さらに、コードの構文が不正確な場合、スマートエディタはコードを正しく解析できず、スマートエディタが機能しないことがあります。このとき、ポップアップメニューは表示されません。スマートエディタ機能はマクロの使用をサポートしないことに注意してください。**

シミュレータ・デバッガ編

1. はじめに

この章ではシミュレータ・デバッガの利用に必要なHEWの基本概念を説明します。Windows®操作に慣れていないユーザのために、次章以降で必要となる情報を提供します。

1.1 ワークスペース、プロジェクト、ファイル

ワードプロセッサでドキュメントを作成、修正できるのと同じように、HEWではワークスペースを作成、修正できます。

ワークスペースはプロジェクトを入れる箱と考えることができます。同じように、プロジェクトはプロジェクトファイルを入れる箱と考えることができます。したがって各ワークスペースにはプロジェクトが一つ以上あり、各プロジェクトにはファイルが一つ以上あります。この構成を図 1-1に示します。

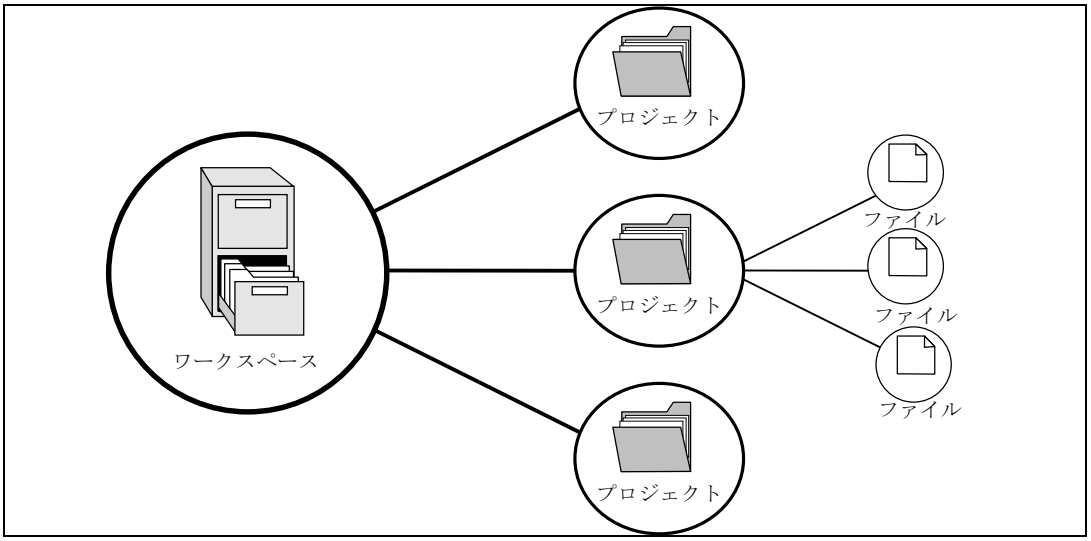


図 1-1 ワークスペース、プロジェクト、ファイル

ワークスペースでは関連したプロジェクトを一つにまとめることができます。例えば、異なるプロセッサに対して1つのアプリケーションを構築しなければならない場合、または、アプリケーションとライブラリを同時に開発している場合などに便利です。さらに、ワークスペース内でプロジェクトを階層的に関連づけることができます。つまり、1つのプロジェクトを構築すると、その子プロジェクトを最初に構築します。

ワークスペースを活用するには、ユーザは、まずワークスペースにプロジェクトを追加して、そのプロジェクトにファイルを追加しなければなりません。

1.2 HEW を起動する

HEWを起動するにはWindows®の[スタート]メニューを開き、[プログラム]を選択し、 [Renesas High-performance Embedded Workshop]を選択し、HEWのショートカットを選びます。デフォルトで図1-2に示す[ようこそ!]ダイアログボックスが開きます。

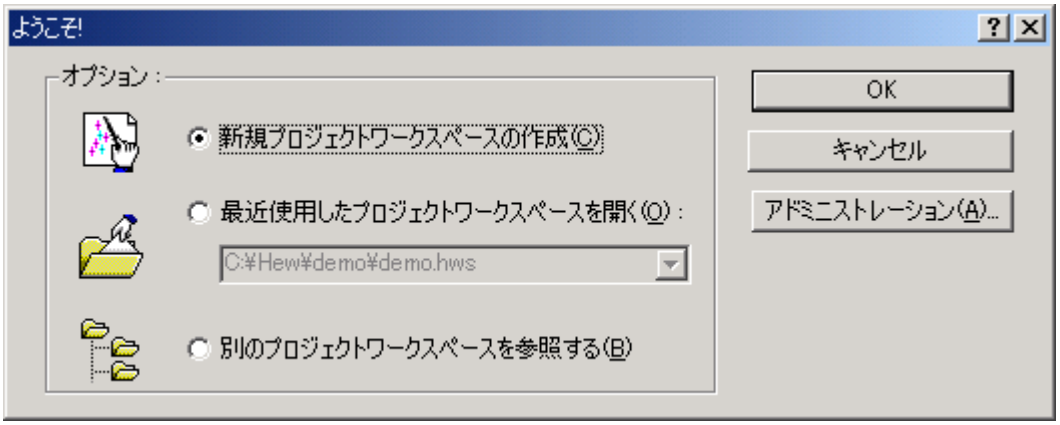


図 1-2 ようこそ!ダイアログボックス

新規ワークスペースを作成するには[新規プロジェクトワークスペースの作成]ボタンを選択し、[OK]ボタンをクリックしてください。最近開いたワークスペースを開くには[最近使用したプロジェクトワークスペースを開く]ボタンを選択し、ドロップダウンリストから開きたいワークスペースを選択し、[OK]ボタンをクリックしてください。[最近使用したプロジェクトワークスペース]のリストには、最近使ったワークスペースファイルリストで見るものと同じ内容を表示します。このリストはファイルメニュー上にも表示します。ワークスペースファイル（”.HWS”ファイル）を指定してワークスペースを開くには[別のプロジェクトワークスペースを参照する]ボタンを選択し、[OK]ボタンをクリックしてください。HEWにツールを登録したり、HEWからツールの登録を外したりするには[アドミニストレーション]ボタンをクリックしてください。ワークスペースを開かないでHEWを使うには[キャンセル]ボタンをクリックしてください。

1.3　新規ワークスペースを作成する

➲新規にワークスペースを作成するには

1. [ようこそ!]ダイアログボックス（図 1-2）から[新規プロジェクトワークスペースの作成]オプションを選び、[OK]ボタンをクリックするか、[ファイル->新規ワークスペース...]を選んでください。[新規プロジェクトワークスペース] ダイアログボックスを表示します（図 1-3）

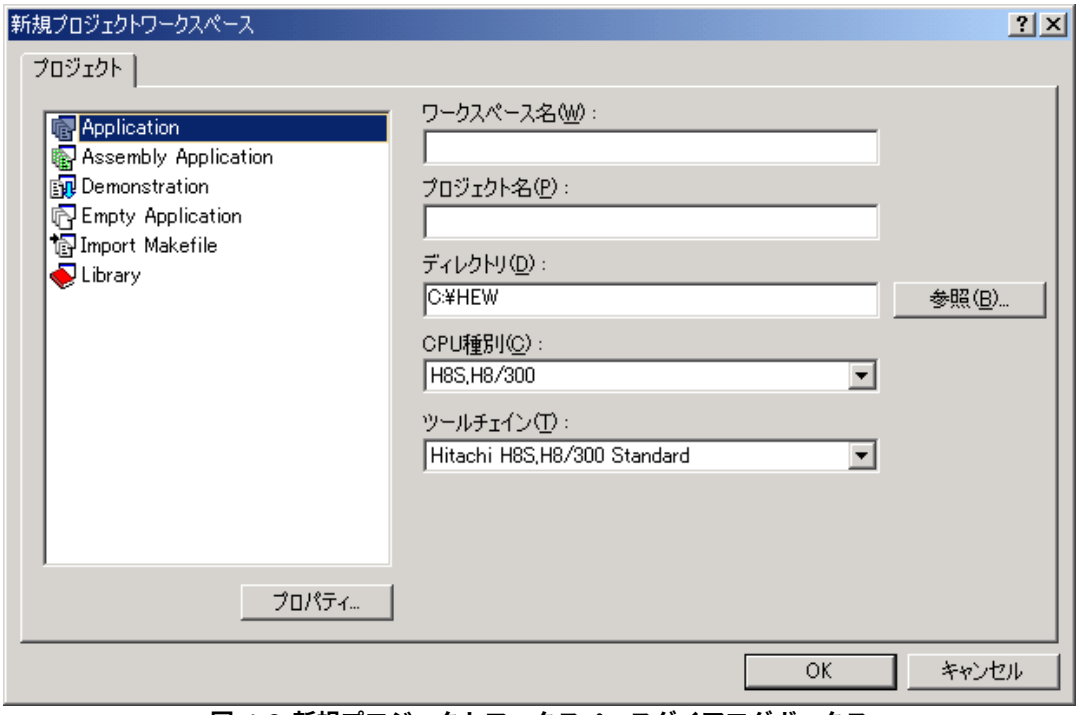


図 1-3 新規プロジェクトワークスペースダイアログボックス

2. [ワークスペース名]フィールドに新規ワークスペース名を入力してください。新規ワークスペース名は32文字以内で、半角英数字、半角下線を使用できます。ワークスペース名を入力すると、自動的にワークスペースのサブディレクトリおよびプロジェクト名をHEWに追加します。[参照...]ボタンをクリックしてワークスペースを作成するディレクトリを選んだり、[ディレクトリ]フィールドに、ワークスペースを作成するディレクトリを手入力することができます。この場合ワークスペース名とプロジェクト名が異なります。
3. ワークスペースの基盤となるCPUファミリおよびツールチェインを選んでください。
4. 新規ワークスペースを作成するとき、HEWは自動的にプロジェクト名フィールドで指定したプロジェクトを作成して、新規ワークスペースに追加します。[プロジェクトタイプ]リストには、使用可能なプロジェクトの種類（アプリケーション、ライブラリなど）を表示します。作成するプロジェクトの種類をリストから選んでください。表示するプロジェクトの種類は、現在のCPUファミリおよびツールチェインの組み合わせに有効な全種類となります。ツールチェイン専用、デバッガ専用、またはHEWのデバッガおよびツールチェインの両方を構築するフルプロジェクトジェネレータがあります。
5. [OK]ボタンをクリックすると、新規のワークスペースとプロジェクトを作成します。

【注】 同一ディレクトリにすでにワークスペースが存在する場合は、ワークスペースを作成できません。

1.4 ワークスペースを開く

➲ワークスペースを開くには

1. [ようこそ!]ダイアログボックス（図 1-2）から[別のプロジェクトワークスペースを参照する]オプションを選んで[OK]ボタンをクリックするか、 [ファイル->ワークスペースを開く…]を選んでください。[ワークスペースを開く]ダイアログボックスを表示します。
2. 開きたいワークスペースファイルを選びます（”.HWS” ファイルのみ）。
3. [Open]ボタンをクリックしてワークスペースを開いてください。ワークスペースを開くときに情報を表示するように設定している場合、[ワークスペースプロパティ]ダイアログボックスを表示します（図 1-4）。設定していない場合、ワークスペースを開きます。
   [ワークスペースプロパティ]ダイアログボックスを表示するかどうかは[ワークスペースプロパティ]ダイアログボックスの[ワークスペースを開いたときにワークスペース情報の表示]チェックボックス、または、[ツール->オプション]ダイアログボックス[ワークスペース]タブの [ワークスペースを開いたときにワークスペース情報の表示] チェックボックスのチェックの有無によります。[ワークスペースプロパティ]ダイアログボックスで[OK]ボタンをクリックするとワークスペースを開きます。[キャンセル]ボタンをクリックするとワークスペースを開きません。

図 1-4 ワークスペースプロパティダイアログボックス

HEWでは、最近開いたファイル5つまでを[ファイル]メニューの[最近使ったワークスペース]サブメニューに追加します。最近利用したファイルを再び開きたいときに便利です。

➲最近使ったワークスペースを開くには

1. [ようこそ!]ダイアログボックスから[最近使用したプロジェクトワークスペースを開く]を選び、ドロップダウンリストからワークスペース名を選び、[OK]ボタンをクリックします。
   または、
2. [ファイル->最近使ったワークスペース]を選び、そのサブメニューからワークスペース名を選びます。

【注】 HEW では、一度に 1 つのワークスペースしか開けません。したがって、すでに開いているワークスペースがあるときに別のワークスペースを開こうとすると、すでに開いているワークスペースを閉じてから新しいワークスペースを開く必要があります。

1.5 ワークスペースを保存する

[ファイル->ワークスペースの保存]を選ぶと、HEWのワークスペースが保存できます。

1.6 ワークスペースを閉じる

HEWのワークスペースを閉じるには、[ファイル->ワークスペースを閉じる]を選んでください。ワークスペースまたはそのプロジェクトに変更があった場合は、保存するかどうかを選んでください。

[ファイル->ワークスペースの保存]を選ぶと、HEWのワークスペースが保存できます。

1.7 古いワークスペースの使用

HEWでは、以前のバージョンのHEWで作成したワークスペースも開くことができます。これによって問題が発生することはなく、ワークスペースファイルの細部における違いもワークスペースを開いたときにアップグレードします。アップグレードしたファイルの現在のディレクトリに、初期のワークスペースまたはプロジェクトファイルのバックアップを保存しておく必要があります。HDIで使用したセッションファイルをアップグレードすることはありません。

1.8 HEW を終了する

HEWを終了するには [ファイル->アプリケーションの終了]を選ぶか、 “Alt+F4”アクセラレータを使用するか、システムメニューから[閉じる]オプションを選んでください。(システムメニューはHEWタイトルバーの最も左上側にあるアイコンをクリックすると開きます。) ワークスペースが開いているときは、前節で説明したワークスペースを閉じる操作を行います。

1.9 デバッガセッション

HEWは、ビルダオプションをコンフィグレーションへ保存することができます。同様に、HEWは、デバッガオプションをセッションに保存することもできます。セッションには、デバッギングプラットフォーム、ダウンロードするプログラム、各デバッギングプラットフォームのオプションを保存することができます。

セッションは、コンフィグレーションとは直接関連がありません。これは、複数のセッションが同

じダウンロードモジュールを共有し、プログラムの不要なリビルドを避けられることを意味します。
　各セッションのデータは、別々のファイルで HEW プロジェクトに保存します。詳細については、「3.4　デバッガセッション」で説明します。

2. シミュレータ・デバッガの機能

本章では、H8S,H8/300シリーズ シミュレータ・デバッガの機能について説明します。

2.1 特長

本シミュレータ・デバッガには、次のような特長があります。

（1） ホスト計算機上で動作するので、実機がなくてもプログラムのデバッグを開始することができ、システム全体の開発期間を短縮できます。

（2） シミュレーション時にプログラムの命令実行サイクル数を計算します。これにより実機がなくても性能評価が行えます。

（3） 擬似割込み機能、およびI/Oシミュレーション機能を持ち、簡易的なシステムレベルシミュレーションを行えます。

（4） 下記のような機能を持ち、プログラムのテスト、およびデバッグを効率よく進めることができます。

- H8S、H8/300 シリーズの各 CPU に対応
- ユーザプログラムの実行中に異常が発生した場合、異常を無視して続行するか、または停止するかを制御する機能
- プロファイルデータ取得、および関数単位のパフォーマンス測定
- 豊富なブレーク機能
- メモリマップの設定・編集
- 関数呼び出し履歴の表示
- C/C++およびアセンブラソースレベルのカバレジ表示
- イメージ表示、波形表示による視覚的デバッグ機能

（5） Windows®上で動作し、ブレークポイント・メモリマップ・パフォーマンス・トレースをダイアログボックス上で設定することができます。H8S、H8/300マイコンの各々のメモリマップに対応した環境設定もダイアログボックス上で行うことができます。
また、下記のような特徴を持ちます。

- 直観的なユーザインタフェース
- オンラインヘルプ
- 共通した表示と操作性

2.2 デバッグ対象プログラム

シミュレータ・デバッガでは、ELF/DWARF2 フォーマットのロードモジュールがシンボリックデバッグ可能です。その他のフォーマットのロードモジュールについては、ダウンロードと命令実行はできますが、シンボリックデバッグはできません。詳しくは、「4.11　Elf/Dwarf2 のサポート」を参照してください。

2.3 シミュレーション範囲

（1） H8/300、H8/300L、H8/300H、H8S/2600、H8S/2000およびH8SXシリーズのシミュレーションをサポートします。

（2） シミュレータ・デバッガは、H8S、H8/300シリーズマイコンの下記機能をサポートしています。

- 全実行命令
- 例外処理
- レジスタ
- 全アドレス空間
- 表 2-1に示す各 CPU モード

表2-1 プラットフォームと CPU モードの対応

デバッグ プラットフォーム名	対応 CPU
H8/300 Simulator	H8/300
H8/300L Simulator	H8/300L
H8/300HA Simulator	H8/300H **アドバンストモード**
H8/300HN Simulator	H8/300H **ノーマルモード**
H8S/2600A Simulator	H8S/2600 **アドバンストモード**
H8S/2600N Simulator	H8S/2600 **ノーマルモード**
H8S/2000A Simulator	H8S/2000 **アドバンストモード**
H8S/2000N Simulator	H8S/2000 **ノーマルモード**
H8SX Normal Simulator	H8SX **ノーマルモード**
H8SX Advanced Simulator	H8SX **アドバンストモード**
H8SX Maximum Simulator	H8SX **マキシマムモード**

（3） シミュレータ・デバッガは、H8S、H8/300シリーズマイコンの下記機能をサポートしていません。下記機能を使用したプログラムは、H8S、H8/300シリーズ用エミュレータを使用してデバッグしてください。

- デュアルポート RAM
- タイマ
- パルス幅変換器（PWM）
- シリアルコミニュケーションインタフェース（SCI）
- A/D 変換器
- I/O ポート
- 割込みコントローラ

2.4　メモリ管理

(1)　メモリマップの設定

メモリマップは、シミュレーション時のメモリアクセスサイクル数の計算に使用します。設定できる項目は次の通りです。

- メモリ種別
- メモリ領域の先頭位置、終了位置
- メモリアクセスのサイクル数
- メモリのデータバス幅

詳細は、**「**4.22.2　**メモリマップおよびメモリリソースの設定を変更する」**を参照してください。なお、ユーザプログラムは、内蔵 I/O 空間を除くすべてのメモリで実行可能です。

(2)　メモリリソースの設定

ユーザプログラムをロードして実行させるためにメモリリソースを設定する必要があります。設定できる項目は以下の通りです。

- 開始アドレス
- 終了アドレス
- アクセス種別

アクセス種別は、読み書き可能、読み出しのみ可能、書き込みのみ可能があります。

ユーザプログラムで、読み出しのみ可能メモリへ書き込みを行う等の不正なアクセスを行ったときはエラーとなるので、誤ったメモリアクセスを検出することができます。

EEPROM については、他のメモリと異なりアクセス種別が読み出しのみ可能な場合でも、EEPMOV 命令により書き込みできます。反対に書き込み可能でも、EEPMOV 命令以外では書き込みできません。

メモリリソース設定の詳細は、「3.3.2　メモリリソース」を参照してください。

2.5　命令実行リセット処理

シミュレータ・デバッガでは、以下の場合に命令実行数および命令実行サイクル数をリセットします。

① 命令シミュレーション停止後再実行までにPCを変更した
② 実行開始アドレスを指定したRunコマンドを実行した
③ イニシャライズまたはプログラムをロードした

2.6 例外処理

シミュレータ・デバッガでは、TRAPA 命令（H8/300H、H8S シリーズのみ）、トレース例外（H8S シリーズのみ）の発生を検出し、例外処理をシミュレーションします。これにより、例外発生時のシミュレーションも行うことができます。

例外処理のシミュレーションは、次の手順で行います。

(a) 命令の実行中に例外の発生を検出します。
(b) スタック領域に PCと CCRを退避します。EXRの有効ビットが ONのときには EXRも退避します。退避処理でエラーが発生した場合は、例外処理を中止し、例外処理エラーが発生したことを表示後、シミュレータ･デバッガのコマンド待ちに戻ります。
(c) CCRの Iビットを１にセットします。
(d) ベクタ番号に対応するベクタアドレスから、スタートアドレスを読み出します。読み出し処理でエラーが発生した場合は、例外処理を中止し、例外処理エラーが発生したことを表示後、シミュレータ･デバッガのコマンド待ちに戻ります。
(e) スタートアドレスから命令実行を行います。スタートアドレスが０の場合は、例外処理を中止し、例外処理エラーが発生したことを表示後、シミュレータ・デバッガのコマンド待ち状態に戻ります。

2.7 H8S/2600CPU 特殊機能

(1) MAC 命令

H8S/2600CPU では、積和演算（MAC 命令）が行えます。この命令では飽和演算、非飽和演算が選択できます。本シミュレータ･デバッガでは内蔵 I/O の SYSCR レジスタのビット７（以下 MACS ビットとする）の値により判定します。

MACS ビット　０：　非飽和演算
　　１：　飽和演算

(2) EXR レジスタ

H8S/2600CPU では EXR レジスタを使用できます。また、このレジスタの有効/無効を設定することもできます。本シミュレータ･デバッガでは内蔵 I/O の SYSCR レジスタのビット５（以下 EXR ビットとする）の値により判定します。

EXR ビット　０：EXR 無効
　　１：EXR 有効

SYSCR アドレスは、[シミュレータシステム]ダイアログボックスの [SYSCR Address]で設定します。

【注】SYSCR アドレスは内蔵 I/O に設定してください。内蔵 I/O 以外に SYSCR アドレスを設定している場合は MACS ビットを０（非飽和）、EXR ビットを０（EXR 無効）と判断しますのでご注意ください。

詳しくは、**「4.22.1 シミュレータシステムの設定を変更する」**を参照してください。

2.8　H8SX シリーズ CPU 特殊機能

(1)　乗算器、除算器の有効／無効切替

乗算器および除算器の有効／無効を切り替えできます。これにより、乗算器、除算器有無によるプログラムパフォーマンスの違いを確認できます。乗算器および除算器の有効／無効は、[シミュレータシステム]ダイアログボックスの [乗算器]および[除算器]で設定します。

(2)　フェッチモード

16ビットフェッチおよび32ビットフェッチを指定できます。これにより、フェッチサイズによるプログラムパフォーマンスの違いを確認できます。指定は、ワークスペース作成時に行ないます。「３．２ デバッギングプラットフォームを選択する」のステップ８を参照してください。

(3)　エンディアン

MCUモードがアドバンストまたはマキシマムモードでアドレス空間サイズが16Mまたは256Mの場合のみENDIANCRレジスタによりエンディアンを設定できます。

メモリ上のデータ格納形式としてビッグエンディアンの他にリトルエンディアンのバイト順をサポートしています。これにより、リトルエンディアンで作成したユーザプログラムのシミュレーション、デバッグが可能になります。

(4)　割り込み制御選択モード

INTCR(割り込みコントロールレジスタ)の INTM1 ビットのみサポートします。本ビットにより割り込み制御モードを選択できます。

2.9　制御レジスタ

H8S/2600 シリーズおよび H8SX シリーズでは、メモリにマッピングした制御レジスタとして、SYSCR（システムコントロールレジスタ）をサポートしています。これにより、積和演算、EXR アクセスを行っているユーザプログラムのシミュレーション、デバッグを行うことができます。

SYSCR アドレスは、[シミュレータシステム]ダイアログボックスの [SYSCR アドレス]で設定します。制御レジスタの変更や表示は [IO]ウィンドウをご利用ください。

詳しくは、**「**4.22.1　**シミュレータシステムの設定を変更する」**、**「**4.5　I/O**メモリを見る」**を参照してください。

2.10　H8SX のタイマ

2.10.1　サポート範囲

H8SX シリーズシミュレータ・デバッガでは 16 ビットタイマパルスユニット 0 のチャネル 0 をサポートしています。

また、オーバーフロー／コンペアマッチによる割り込みだけをサポートしており、インプットキャプチャ等の端子入出力をともなう機能はサポートしていません。

2.10.2 制御レジスタ

シミュレータ・デバッガではタイマおよび割込みの優先順位設定をサポートしています。シミュレータ・デバッガでサポートしている制御レジスタを表 2-2、表 2-3に示します。

表中サポート状況の○はサポート、△は「2.10.1 サポート範囲」で説明した機能に関するビットのみサポートしています。

表2-2 シミュレータ・デバッガでサポートするタイマ制御レジスタ

タイマ名	サポートする制御レジスタ	サポート状況
TPU0	TSTR	△
	TCR	△
	TIER	○
	TSR	○
	TCNT	○
	TGRA	○
	TGRB	○
	TGRC	○
	TGRD	○

表2-3 シミュレータ・デバッガでサポートするタイマ制御レジスタ

サポートする制御レジスタ	サポート状況
IPRF	△

2.10.3 クロック

シミュレータ・デバッガでは、メモリアクセスにかかわる外部クロック、周辺モジュール用クロック、タイマを動かすクロックをサポートします。

メモリマップで指定するサイクル数が外部クロックになります。外部クロックと周辺モジュール用クロックの比は、[シミュレータシステム]ダイアログまたは P_CLOCK_RATE コマンドで設定してください。

周辺モジュールからタイマを動かすクロックを作るときの分周率は、タイマ制御レジスタで指定してください。

2.10.4 タイマを使用する

初期設定では、タイマは未使用に設定されています。[シミュレータシステム]ダイアログボックスでタイマを使用する設定にしてください。

タイマを未使用に設定している間は、タイマのカウント、割り込みの検出をしないので、高速にシミュレーションを実行できます。

なお、タイマを未使用に設定しても、その時の制御レジスタの状態を保持します。

2.10.5 タイマ使用時の注意

(1) 0 クリアだけ可能なビットも、１をライト可能となっています。

(2)タイマ割込み発生時にブレークするか、しないかを選択できます。

[シミュレータシステム]ダイアログまたは EXEC_STOP_SET コマンドで設定してください。

2.11　トレース

シミュレータ・デバッガは、実行結果をトレースバッファに書き込みます。トレース情報の取得条件は、[トレース取得]ダイアログボックスで指定します。[トレース取得]ダイアログボックスは、[トレース]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...]を選択することによって表示できます。取得したトレース情報は、[トレース]ウィンドウに表示します。

トレース情報はサーチすることができます。サーチ条件は、[トレース検索]ダイアログボックスで設定します。[トレース検索]ダイアログボックスは、[トレース]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、[検索...]を選択することによって表示できます。

詳しくは、**「4.15　トレース情報を見る」**を参照してください。

2.12　標準入出力およびファイル入出力処理

シミュレータ・デバッガでは、ユーザプログラムから標準入出力およびファイル入出力を行うことができます。入出力機能を利用する場合は、必ず [I/O シミュレーション]ウィンドウをオープンしておいてください。

サポートしている入出力処理は以下の通りです。機能コードには、16 ビットアドレス版、24 ビットアドレス版、32 ビットアドレス版があります。使用する CPU に合わせて選択してください。

表2-4　入出力機能一覧

番号	機能コード	機能名	内　容
1	H'01 (16 **ビットアドレス**) H'11 (24 **ビットアドレス**) H'21 (32 **ビットアドレス**)	GETC	**標準入力からの１バイト入力**
2	H'02 (16 **ビットアドレス**) H'12 (24 **ビットアドレス**) H'22 (32 **ビットアドレス**)	PUTC	**標準出力への１バイト出力**
3	H'03 (16 **ビットアドレス**) H'13 (24 **ビットアドレス**) H'23 (32 **ビットアドレス**)	GETS	**標準入力からの１行入力**
4	H'04 (16 **ビットアドレス**) H'14 (24 **ビットアドレス**) H'24 (32 **ビットアドレス**)	PUTS	**標準出力への１行出力**
5	H'05 (16 **ビットアドレス**) H'15 (24 **ビットアドレス**) H'25 (32 **ビットアドレス**)	FOPEN	**ファイルのオープン**
6	H'06	FCLOSE	**ファイルのクローズ**
7	H'07 (16 **ビットアドレス**) H'17 (24 **ビットアドレス**) H'27 (32 **ビットアドレス**)	FGETC	**ファイルからの１バイト入力**
8	H'08 (16 **ビットアドレス**) H'18 (24 **ビットアドレス**) H'28 (32 **ビットアドレス**)	FPUTC	**ファイルへの１バイト出力**

番号	機能コード	機能名	内　容
9	H'09 (16 ビットアドレス) H'19 (24 ビットアドレス) H'29 (32 ビットアドレス)	FGETS	ファイルからの１行入力
10	H'0A (16 ビットアドレス) H'1A (24 ビットアドレス) H'2A (32 ビットアドレス)	FPUTS	ファイルへの１行出力
11	H'0B	FEOF	エンドオブファイルのチェック
12	H'0C	FSEEK	ファイルポインタの移動
13	H'0D	FTELL	ファイルポインタの現在位置を得る

入出力機能の詳細は、「4.23　標準入出力およびファイル入出力を行う」を参照してください。

2.13　命令実行サイクル数の計算

シミュレータ･デバッガでの命令実行サイクル数計算は、H8S、H8/300 シリーズ プログラミングマニュアルに記載している計算式とメモリマップのデータバス幅およびアクセスサイクル数を使用して行います。ただし、以下の理由によりシミュレータ･デバッガで計算した命令実行サイクル数と、実機でプログラムを実行した場合の命令実行サイクル数が異なることがあります。

(1)　MOVFPE、MOVTPE 命令

E クロック同期命令のデータ転送サイクル数は､9～16 と幅を持った値となっています｡シミュレータ･デバッガでは「11+オペランドアクセスサイクル数」で計算します。オペランドアクセスサイクル数は、メモリのデータバス幅およびアクセスサイクル数より求めます。

(2)　EEPMOV 命令

EEPROM 書き込み専用命令のサイクル数は、命令読み出しに要するサイクル数と、データを転送するのに要するサイクル数の合計となります。

(3)　SLEEP 命令

シミュレータ･デバッガでは、SLEEP 命令がプログラム停止用命令として使用している場合を考慮して、サイクル数を加算しません。

(4)　標準入出力およびファイル入出力処理

標準入出力およびファイル入出力処理は、シミュレータ･デバッガ固有の機能であるため、サイクル数に加算しません。なお、標準入出力およびファイル入出力処理とは、BSR、JSR 命令でシステムコールアドレスで指定した位置への分岐が完了してから入出力処理を行いコール元に戻るまでです。

2.14 ブレーク条件

ユーザプログラムのシミュレーションを中断する条件として以下のものがあります。

- ブレーク系コマンドの条件成立によるブレーク
- ユーザプログラムの実行時エラー検出によるブレーク
- トレースバッファ満杯によるブレーク
- SLEEP 命令実行によるブレーク
- [停止]ボタンによるブレーク

(1) ブレーク系コマンドの条件成立によるブレーク

ブレーク条件を設定するコマンドには次の 5 種類があります。

- BREAKPOINT ：命令実行位置によるブレーク
- BREAK_ACCESS ：メモリ範囲のアクセスによるブレーク
- BREAK_CYCLE ：実行サイクル数によるブレーク
- BREAK_DATA ：メモリ書き込みデータ値によるブレーク
- BREAK_REGISTER ：レジスタ書き込みデータ値によるブレーク
- BREAK_SEQUENCE ：実行順序を指定したブレーク

ブレーク条件成立時の動作を[Stop]と指定した場合、そのブレーク条件が成立するとプログラムを中断します。詳しくは、**「4.16 シミュレータ・デバッガのブレークポイントを使用する」**を参照してください。

ユーザプログラム実行中にブレーク条件が成立しプログラムが中断した場合、ブレークポイントの命令を実行しないで停止するか、実行してから停止するかを**表** 2-5に示します。

表2-5 **ブレーク条件成立時の処理**

コマンド名	ブレーク条件成立命令	
	実行する	実行しない
BREAKPOINT		○
BREAK_ACCESS	○	
BREAK_CYCLE	○	
BREAK_DATA	○	
BREAK_REGISTER	○	
BREAK_SEQUENCE		○

BREAKPOINT、BREAK_SEQUENCE の場合、実行命令の先頭位置以外にブレークポイントを設定するとブレークを検出できません。

ユーザプログラム実行中にブレーク条件が成立すると、ブレーク条件成立のメッセージをステータスバーに表示して、命令実行を中断します。

(2) ユーザプログラムの実行時エラー検出によるブレーク

シミュレータ・デバッガでは、CPU の例外発生機能では検出できないプログラムの誤りを検出するためにシミュレーションエラーを設けています。これらのエラーが発生した場合に、シミュレーションを停止するか、続行するかを [シミュレータシステム]ダイアログボックスにより選択できます。エラーの種類、エラーメッセージ、エラー発生要因、および続行時のシミュレータ・デバッガの動作を**表** 2-6に示します。

表2-6 シミュレーションエラー一覧

エラーの種類/メッセージ	エラー発生要因	続行モード時処理
アドレスエラー/Address Error	• PC 値が奇数 • 内蔵 I/O 空間からの命令フェッチ • 奇数アドレスからのワードアクセス • 奇数アドレスからのロングワードアクセス	デバイスと同一動作をする
メモリアクセスエラー/ Memory Access Error	• 確保していないメモリ領域をアクセスしようとした • 書き込み不可属性を持つメモリへ書き込みを行おうとした • 読み出し不可属性を持つメモリから読み出しを行おうとした • メモリが存在しない領域をアクセスしようとした • EEPMOV 命令以外の命令での EEPROM 書き込み	メモリへの書き込み時、何も書き込まない メモリ読み出し時、全ビット"1"を読み出す
不当命令/ Illegal Instruction	• 命令ではないコードの実行	常に停止する
	• MOV.B Rn,@-SP または MOV.B @SP+,Rn の実行	そのまま実行するが結果は保証しない
命令実行不正/ Illegal Operation	• DAA 命令、DAS 命令で CCR の C フラグ、H フラグと補正前の値の関係不正 • DIVXU 命令、DIVXS 命令のゼロ除算またはオーバーフロー	そのまま実行するが結果は保証しない

停止モードの場合、シミュレーションエラーが発生するとシミュレータ・デバッガは、命令実行を中止してエラーメッセージを表示後、コマンド待ち状態に戻ります。シミュレーションエラー停止後の PC の状態を**表** 2-7に示します。なお、シミュレーションエラー停止後 CCR の内容は変化しません。

表2-7 シミュレーションエラー停止時のレジスタ

エラーの種類	PC の内容
アドレスエラー、 メモリアクセスエラー	命令読込時： エラーが発生した命令の先頭アドレス 命令実行時： エラーが発生した命令の次命令のアドレス
不当命令	エラーが発生した命令の先頭アドレス
命令実行不正	エラーが発生した命令の次命令のアドレス

シミュレーションエラーが発生する命令を組み込んだプログラムのデバッグは、次の手順で行ってください。

(a) 最初は停止モードで実行させて、意図している箇所以外にエラーがないかどうかを確認してください。

(b) 確認が完了したら、続行モードで実行してください。

【注】 停止モードでエラーが発生して停止した状態から、モードを続行モードに変更してシミュ

レーションを再開すると、正しくシミュレーションできない場合があります。シミュレーションを再開する場合は、レジスタ内容、メモリの内容をエラー発生前の状態に戻してから再実行するようにしてください。

(3) トレースバッファ満杯によるブレーク

[トレース取得]ダイアログボックスの [トレースバッファ満杯時の動作]で [停止]モードを指定し、命令実行中にトレースバッファが満杯になると、シミュレータ・デバッガは、実行を中断します。中断時には以下のメッセージを[アウトプット]ウィンドウに表示します。

Trace Buffer Full

(4) SLEEP 命令実行によるブレーク

命令実行時に、SLEEP 命令を実行すると、シミュレータ・デバッガは実行を中断します。中断時には、以下のメッセージを[アウトプット]ウィンドウに表示します。

Sleep

【注】実行を再開する場合は、PC の値を再開位置の命令アドレスに変更してください。

(5) [停止]ボタンによるブレーク

命令実行中にユーザにより強制的に実行を中断することができます。中断時には以下のメッセージをステータスバーに表示します。

Stop

Go、Step コマンドにより実行を再開できます。

2.15 浮動小数点データ

実数データとして浮動小数点数を指定することができます。これにより、データ値等で浮動小数点を扱う場合の操作が容易になります。浮動小数点を指定できる項目は次の通りです。

・[ブレーク設定]ダイアログボックスにおいて、ブレーク種別を [ブレークデータ] や [ブレークレジスタ]と指定したときのデータ
・[メモリ]ウィンドウにおけるデータ
・[メモリフィル]ダイアログボックスにおけるデータ
・[メモリ検索]ダイアログボックスにおけるデータ
・レジスタ値編集ダイアログボックスでの入力値

浮動小数点データフォーマットは、ANSI C の浮動小数点フォーマットに準拠しています。

シミュレータ・デバッガでは、[シミュレータシステム]ダイアログボックスにより、浮動小数点数の 10 進->2 進変換で発生する丸めのモードを選択することができます。次の 2 通りより選択します。

- RN（最近値丸め）
- RZ（ゼロ丸め）

なお、10 進->2 進変換および 2 進->10 進変換で非正規化数を指定した場合、RZ モードでは 0 に変換し、RN モードでは非正規化数のまま処理します。また、10 進->2 進変換時にオーバーフローが発生した場合、RZ モードでは浮動小数点数の最大値を、RN モードでは無限大を設定します。

2.16 関数呼び出し履歴の表示

シミュレーションの中断時に、関数の呼び出し履歴を [スタックトレース]ウィンドウに表示します。これにより、プログラムの動作の流れを確認することができます。また、[スタックトレース]ウィンドウ上で関数名を選択することにより、該当するソースプログラムを [エディタ]ウィンドウ上に表示します。これにより、中断している関数の他に、その関数を呼び出した元の関数をチェックすることができます。

関数呼び出し履歴を更新するのは、以下のような場合です。

- **「**2.12 **ブレーク条件」**に示す条件によりシミュレーションが中断した時
- 上記中断した状態で、レジスタの値を変更した時
- シミュレーションをステップ実行している時

詳しくは、**「**4.14 **関数呼出し履歴を見る」**を参照してください。

2.17 パフォーマンス測定

シミュレータ・デバッガはユーザプログラムのパフォーマンスを測定するためにプロファイラ機能およびパフォーマンス解析機能を提供します。

2.17.1 プロファイラ

プロファイラは、ユーザプログラムの全体について関数とグローバル変数のアドレス、サイズ、関数の呼び出し回数、およびプロファイルデータを表示します。プロファイルデータは CPU により異なります。

プロファイル情報はリスト形式、ツリー形式、チャート形式で表示します。

プロファイル情報を用いることにより、サイズが小さく、呼び出し回数が多い関数をインライン関数にするなどの最適化を検討することが出来ます。

また、ファイルに出力したプロファイル情報を用いて、最適化リンケージエディタで動的情報に基づいた最適化を実行することができます。

詳しくは、**「**4.13 **プロファイル情報を見る」**および「最適化リンケージエディタマニュアル　4.2.3 Optimize オプション　PROfile」を参照してください。

2.17.2 パフォーマンス解析

パフォーマンス解析はユーザプログラム内の指定関数について実行サイクル数、呼び出し回数を表示します。指定関数のみについてパフォーマンスデータを取得するため、プロファイラよりも高速なシミュレーションが可能です。詳しくは、**「**4.17 **パフォーマンスを解析する」**を参照してください。

2.18 擬似割込み

シミュレータ・デバッガでは、シミュレーション中に擬似割込みを発生させることができます。擬似割込みを発生させるには、以下の 2 通りの方法があります。

(1) ブレーク条件成立時の動作による擬似割込み発生

ブレーク系コマンドで、ブレーク条件成立時の動作に[Interrupt]を指定することにより、擬似割込

みを発生させることができます。
　詳しくは、**「**4.16.2(2)　**ブレーク条件成立時の動作を設定する」**を参照してください。

(2)　　トリガウィンドウによる擬似割込み発生

[トリガ]ウィンドウのボタンをクリックすることにより、擬似割込みを発生させることができます。
　詳しくは、**「**4.21　**手動で擬似割込みを発生させる」**を参照してください。

　なお、擬似割込みが発生してからその割込みを受け付けるまでの間に、次の擬似割込みが発生した場合は、優先順位の高い割込みだけを受付けます。

(3)　　擬似割込み発生によるブレーク

擬似割込み発生時にブレークするか、しないかを選択できます。
[シミュレータシステム]ダイアログまたはEXEC_STOP_SETコマンドで設定してください。

【注】　擬似割込みでは、割込みを受け付けるかどうかの判定には、ベクタ番号ではなく割込み情報の[優先順位]を使用します。優先順位に H'8 **以上を指定した場合は、常に割込みを受け付けます。また、割込みコントローラはシミュレーションしていません。**

2.19　カバレジ

　シミュレータ・デバッガでは、ユーザが指定した測定範囲について命令実行中に命令カバレジ情報を収集できます。
　測定範囲は直接アドレスを指定して設定するほかに、ソースファイル名を指定してそのファイルに含まれる全関数を設定することができます。
　命令カバレジ情報を利用することで各命令の実行状態を観察できます。さらにプログラムのどの部分が未実行であるかを容易に特定できます。

　収集した命令カバレジ情報は[カバレジ]ウィンドウに表示します。
　命令カバレジ情報は命令実行済のソース行に対応するカラムを［エディタ]ウィンドウ上に強調表示します。
　また、測定対象のアドレス範囲または関数について、カバレジ統計情報をパーセント形式で表示します。これによりプログラムがどのくらい実行されているかを定量的に把握できます。

　命令カバレジ情報はファイルへの保存およびファイルからのロードが行えます。ロードできるのは".COV"ファイル形式のみです。

　詳しくは、「4.18　コードカバレジを測定する」を参照してください。

3. デバッグの準備をする

この章では、プログラムのデバッグを開始するための準備作業について説明します。デバッグを行うためのデバッギングプラットフォームの選択および構築方法、またユーザプログラムのロード、およびデバッグセッションについて学びます。

3.1 デバッグの前にビルドを行う

プログラムをC/C++ソースレベルでデバッグするには、C/C++プログラムをコンパイルして、Debugオプションをイネーブルの状態でリンクする必要があります。このオプションがイネーブルの場合、コンパイラは、C/C++コードのデバッグに必要な情報をすべてアブソリュートファイルまたはマネージメントファイルに入れます。これらのファイルはその後「デバッグオブジェクトファイル」とよばれます。プロジェクトを作成すると、通常の初期セットアップでデバッグを設定します。

【注】 デバッグ用のオブジェクトファイルを生成する際はコンパイラおよびリンカの[Debug]オプションを必ずイネーブルにしてください。
デバッグオブジェクトファイルにデバッグ情報(例えばSレコードフォーマットなど)がない場合でもデバッギングプラットフォームにロードできますが、アセンブリ言語レベルでのみデバッグすることができます。

3.2 デバッギングプラットフォームを選択する

デバッギングプラットフォームの選択はHEWのインストール方法に大きく依存します。HEWにツールチェインをインストールしている場合、アプリケーションプロジェクトジェネレータはツールチェインおよびデバッガのターゲットを同時にセットアップすることができます。これによってターゲットおよびツールチェインオプションを一致させ、矛盾が起こらないようにします。ツールチェインをインストールしていない場合には、デバッグ専用プロジェクトのみ選択することができます。HEWはデフォルトの設定で、[新規プロジェクトワークスペース]ダイアログボックスの中に、生成するそれぞれのCPUファミリのデバッグ専用プロジェクト生成タイプを表示します。

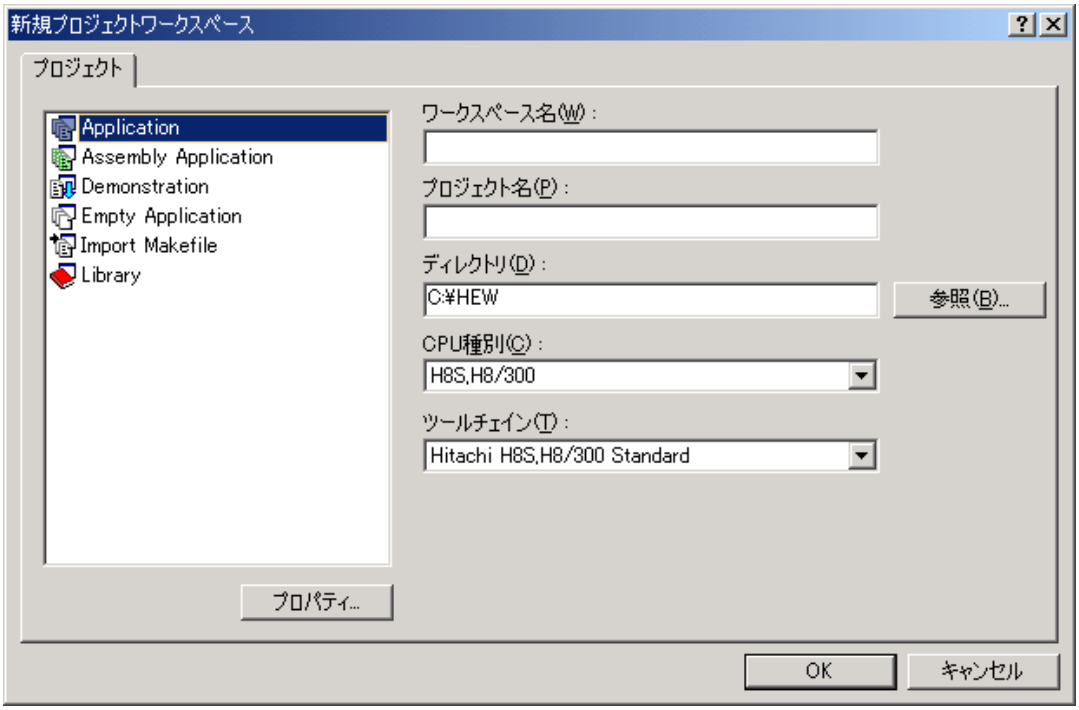


図 3-1 新規プロジェクトワークスペース ダイアログボックス

図 3-1に示すダイアログボックスを使用して、ターゲットCPUに合ったプロジェクト生成タイプを選択することができます。

[Application]	C/C++言語で記述された初期ルーチンファイルを含む実行プログラムを生成するためのプロジェクト
[Assembly Application]	アセンブリ言語で記述された初期ルーチンファイルを含む実行プログラムを生成するためのプロジェクト
[Demonstration]	C言語で記述さ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)
[Import Makefile]	既存のメイクファイルを取り込んで実行プログラムを作成するためのプロジェクト
[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)

【注】　デバッグ専用のプロジェクトを生成する場合は、[Empty Application]**を選択してください。**

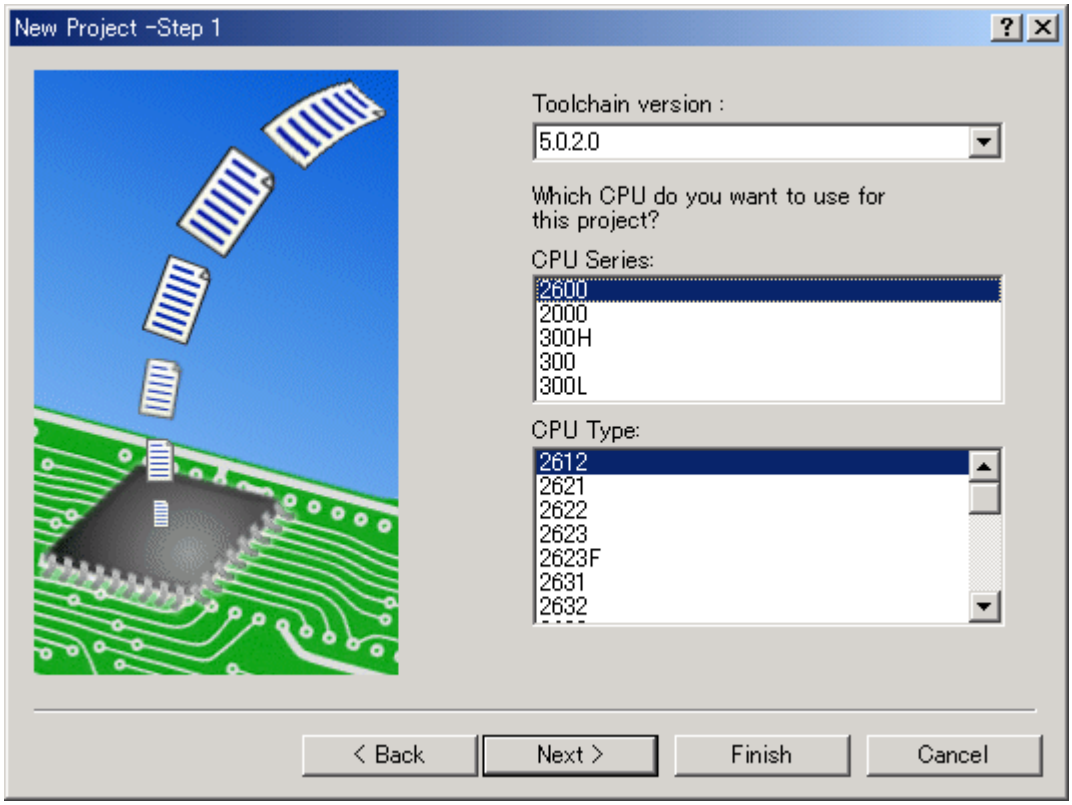


図 3-2 CPU **選択画面** (Step1)

1.　ステップ1ではツールチェインバージョンと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。ツールチェインバージョン[Toolchain version]、[CPU Type]、および[CPU Series]の選択により生成するファイルが異なりますので、開発するプログラムに対応した設定を選択してください。対応するCPUがない場合は、ハードウェア仕様の近いCPUまたは[Other]を選択してください。
ダイアログボックス下部のボタンは[New Project]ウィザードダイアログボックス共通です。

[Next>]	次の画面に移ります
[<Back]	前の画面に戻ります
[Finish]	以降の選択はデフォルトとなり、[Summary]ダイアログボックスを開きます
[Cancel]	[New Workspace]ダイアログボックスに戻ります

Step1画面で[Next>]をクリックすると、ステップ2へ進みます。

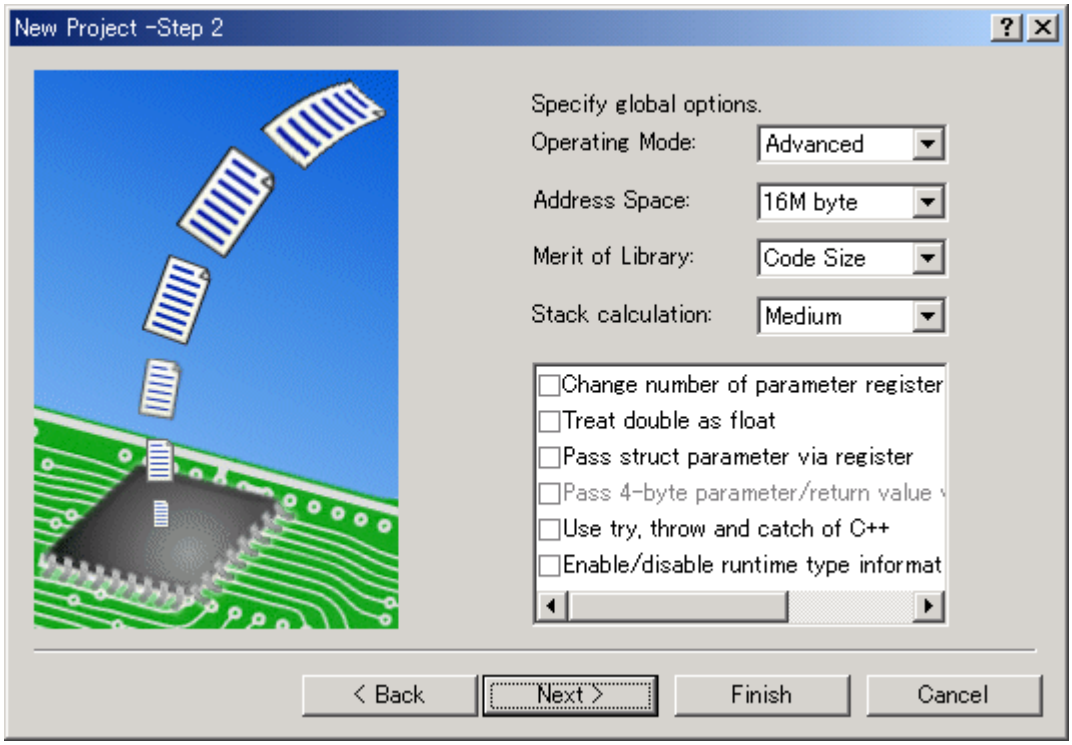


図 3-3 オプション設定画面 (Step2)

2.　ステップ2では全プロジェクトファイル共通のオプションを設定します。設定できる項目はステップ1で選択したCPUに依存します。

リストボックスでの指定項目

[Operating Mode]　作成するオブジェクトの動作モードを指定します。
- [Normal]　ノーマルモード
- [Advanced]　アドバンストモード
- [Middle]　ミドルモードを指定します。
- [Maximum]　マキシマムモードを指定します。

[Address Space]　作成するオブジェクトのアドレス空間を指定します。
- [1M byte]　アドレス空間は 1M byte
- [16M byte]　アドレス空間は 16M byte
- [256M byte]　アドレス空間は 256M byte
- [4G byte]　アドレス空間は 4G byte

[Multiple/Divide]　乗算器、除算器の有効／無効を指定します。
- [None]　乗算器、除算器ともに無効
- [Multiple]　乗算器のみ有効
- [Divide]　除算器のみ有効
- [Multiple and Divide]　乗算器、除算器ともに有効

[Merit of Library]	標準ライブラリを実行速度優先で作成するか、サイズ縮小優先で作成するかを指定します。	
	[Code Size]	サイズ縮小優先
	[Speed]	実行速度優先
[Stack calculation]	スタックアドレスの範囲を指定します。	
	[Small]	スタックアドレス計算を1byteで行う
	[Medium]	スタックアドレス計算を2byteで行う
	[Large]	スタックアドレス計算を4byteで行う
[Specify SBR address]	SBRの値を指定します。	
	[Default]	デフォルト値を使用する
	[Custom]	値を指定する 右横のエディットボックスに使用する値を入力します

チェックボックスでの指定項目

[Change number of parameter register s from 2(default) to 3]	引数格納用レジスタの本数を3本にします。デフォルトは2本です。
[Treat double as float]	倍精度浮動小数点数型の数値を単精度浮動小数点数型としてオブジェクトを生成します。
[Pass struct parameter via register]	構造体のパラメタをレジスタに割り付けるかどうかを指定します。チェックした場合、レジスタを使用して引数を渡します。チェックしなかった場合、レジスタを使用しないでメモリを使用して引数を渡します。
[Pass 4-byte parameter/return value via register]	4byteのパラメタやリターン値をレジスタに割り付けるかどうかを指定します。チェックした場合、レジスタを使用して引数を渡します。チェックしなかった場合、レジスタを使用しないでメモリを使用して引数を渡します。
[Use try, throw and catch of C++]	C++例外処理機能(try, catch, throw) の有無を指定します。
[Enable/disable runtime type information]	実行時型情報の有効/無効を指定します。 チェックした場合、dynamic_cast, typeidを使用できます。 チェックしない場合、dynamic_cast, typeidは使用できません。詳細はコンパイラマニュアルで確認してください。

Step2画面で[Next>]をクリックすると、ステップ3に進みます。

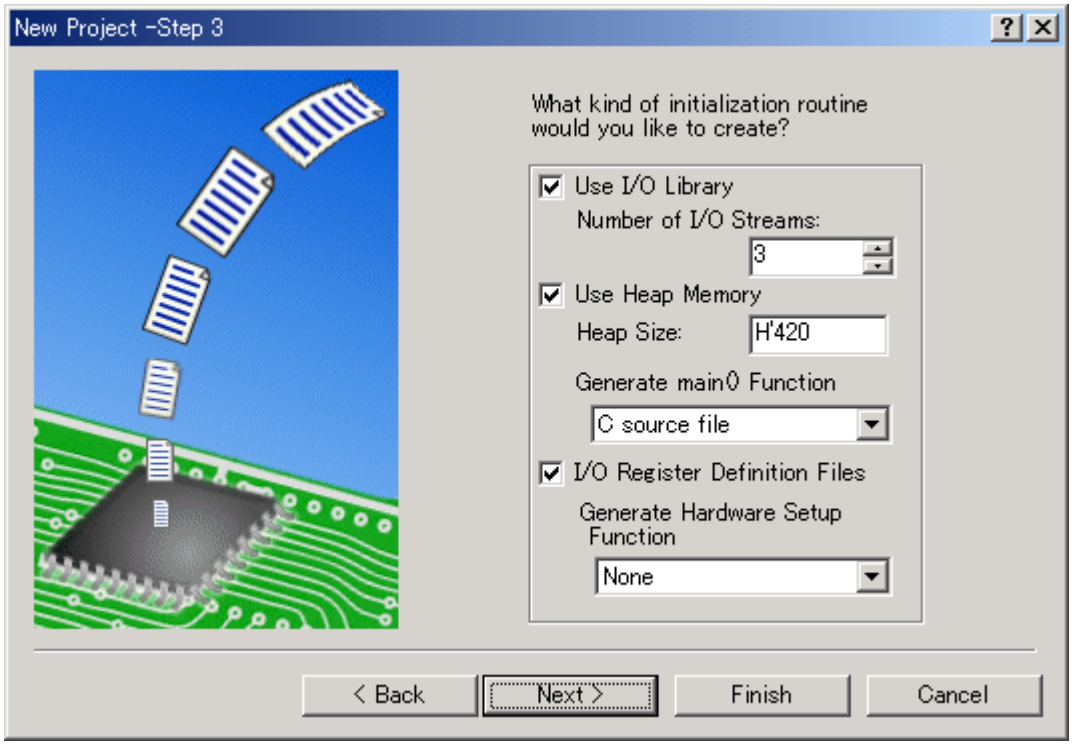


図 3-4 **生成ファイル設定画面** (Step3)

3.　ステップ3では生成するファイルを設定します。

[Use I/O Library]　チェックすると標準入出力ライブラリを活用できます。
- [Number of I/O streams]　同時に使用する入出力ストリームの本数を設定します。

[Use Heap Memory]　チェックするとヒープ領域管理用の関数sbrk()を使用できます。
- [Heap Size]　管理するヒープ領域サイズの単位を指定します。

[Generate main() Function]　main関数の雛型生成を選択できます。
メイン関数ファイル[(プロジェクト名).c(cpp)]を生成します。

[I/O Register Definition Files]　チェックするとC言語で記述したI/Oレジスタ定義ファイル(“iodefine.h”)を生成します。
- [Generate Hardware Setup Function]　I/Oレジスタ初期設定プログラムの雛型生成を選択します。ハードウェア設定用のファイル(“hwsetup.c(cpp)”または”hwsetup.src”)を生成します。

【注】　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 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は、resetprg.c の関数呼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。

Step3画面で[Next>]をクリックすると、ステップ4に進みます。

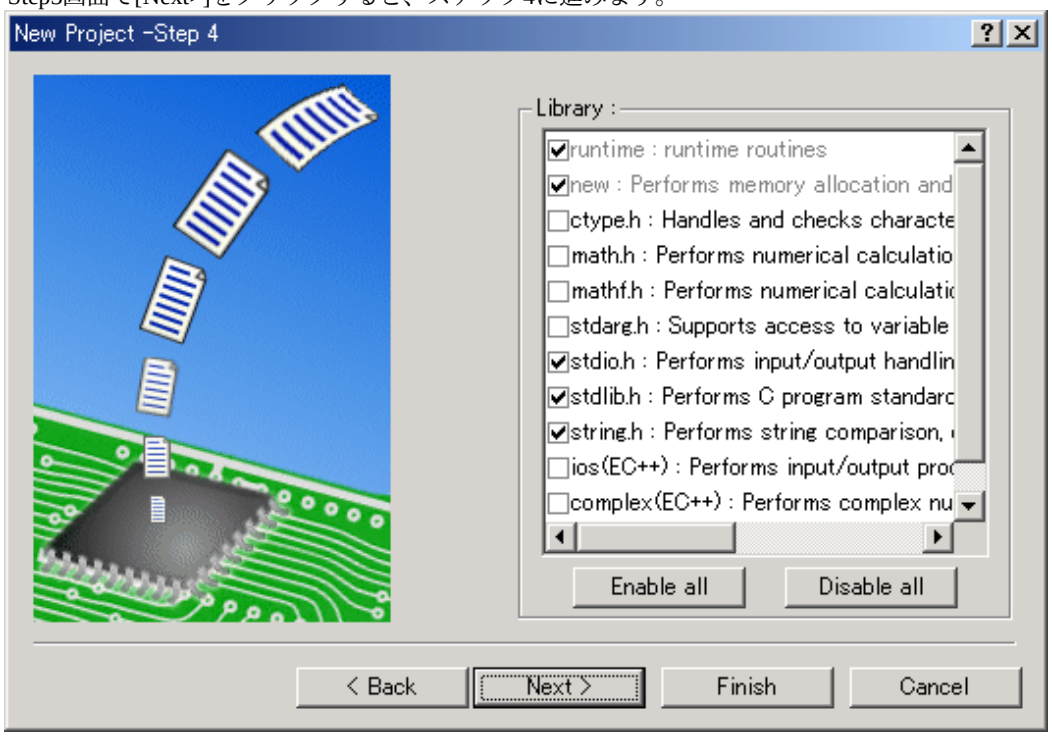


図 3-5 標準ライブラリ設定画面 (Step4)

4. ステップ4ではC/C++コンパイラで使用する標準ライブラリの構成を設定します。
チェックした項目で宣言されている関数と runtime 関数を組み込みます。
[Enable all] すべての標準ライブラリ関数を選択します。
[Disable all] すべての標準ライブラリ関数を選択しません。
ただし、最低限必要なruntime、newは選択します。

Step4画面で[Next>]をクリックすると、ステップ5に進みます。

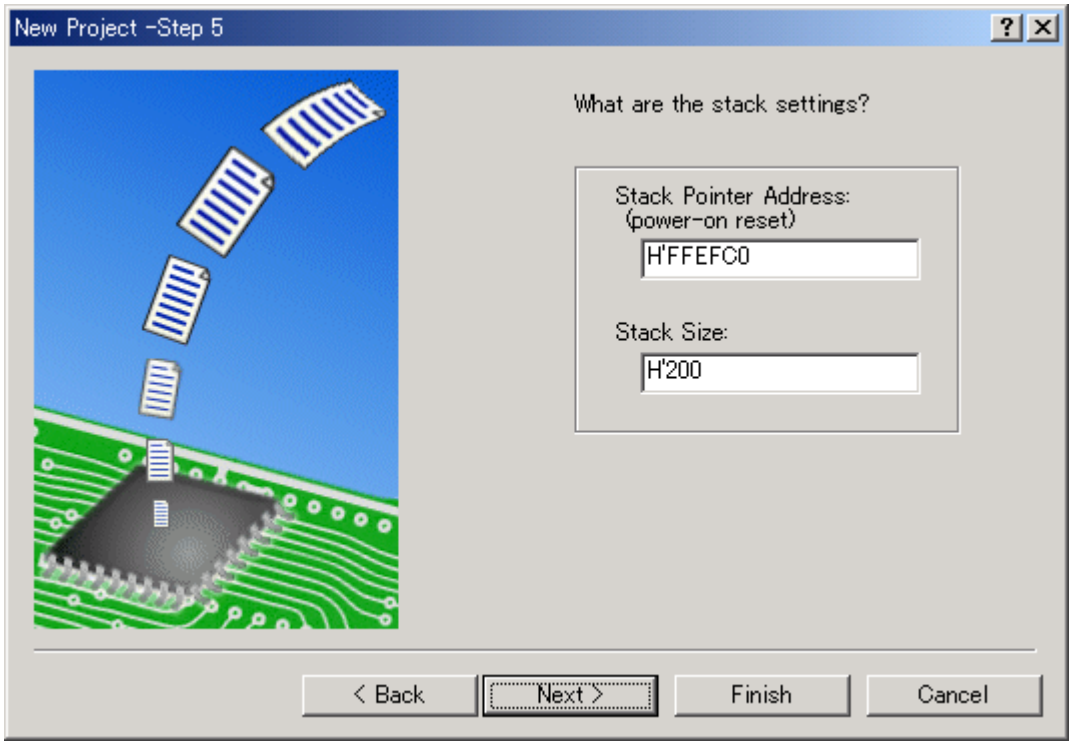


図 3-6 スタック領域設定画面 (Step5)

5.　ステップ5ではスタック領域を設定します。スタックポインタの初期値とスタックサイズを設定します。
スタック領域の初期値はステップ1で選択したCPUに依存します。

【注】　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は HEW の[プロジェクト->構成の編集]では変更できません。

Step5 で[Next>]をクリックするとステップ 6 に進みます。

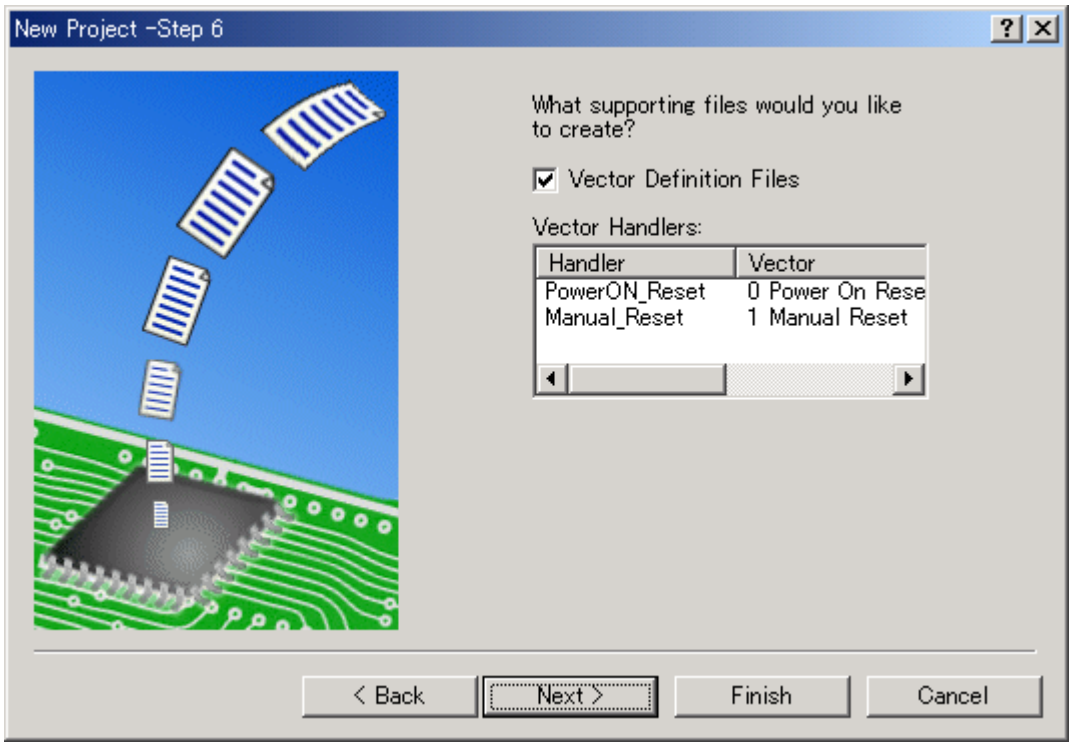


図 3-7 ベクタ設定画面 (Step6)

6.　ステップ6ではベクタを設定します。

[Vector Definition Files]		チェックするとベクタ定義ファイルをベクタテーブル設定関数の定義ファイルを生成します。
[Vector Handlers]	[Handler]	リセットベクタのハンドラプログラム名を表示します。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(“resetprg.c”)は生成しません。
	[Vector]	ベクタの説明を表示します。

【注】　生成されたリセットプログラム、および割込み関数はサンプルであるため、使用される前に CPU のハードウェアマニュアルをご確認ください。

Step6画面で[Next>]をクリックするとステップ7に進みます。

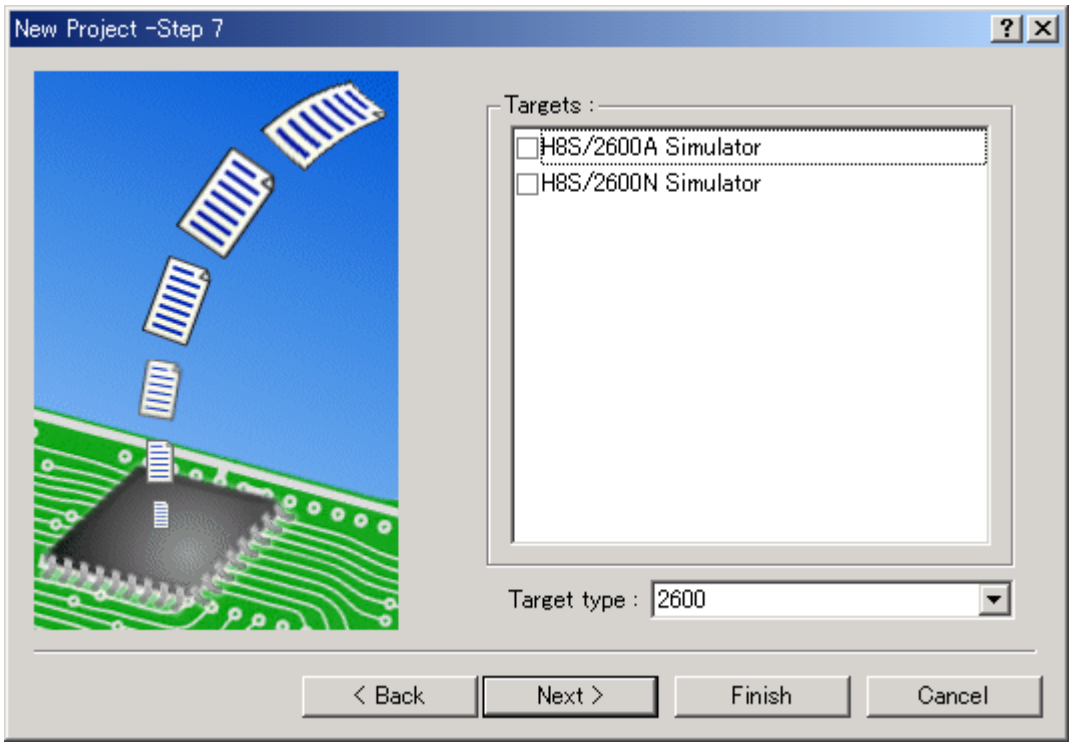


図 3-8 デバッガターゲット設定画面 (Step7)

7.　ステップ7ではデバッガターゲットを設定します。

[Targets]　デバッガターゲットを設定します。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でもかまいません。

[Target Type]　[Targets]に表示するターゲットの種類を指定します。

注：Step2で選択した[Operating Mode] と違う動作モードのデバッガターゲットを選択した場合、デバッグが出来ない場合があります。

Step7画面で[Next>]をクリックするとステップ8に進みます。

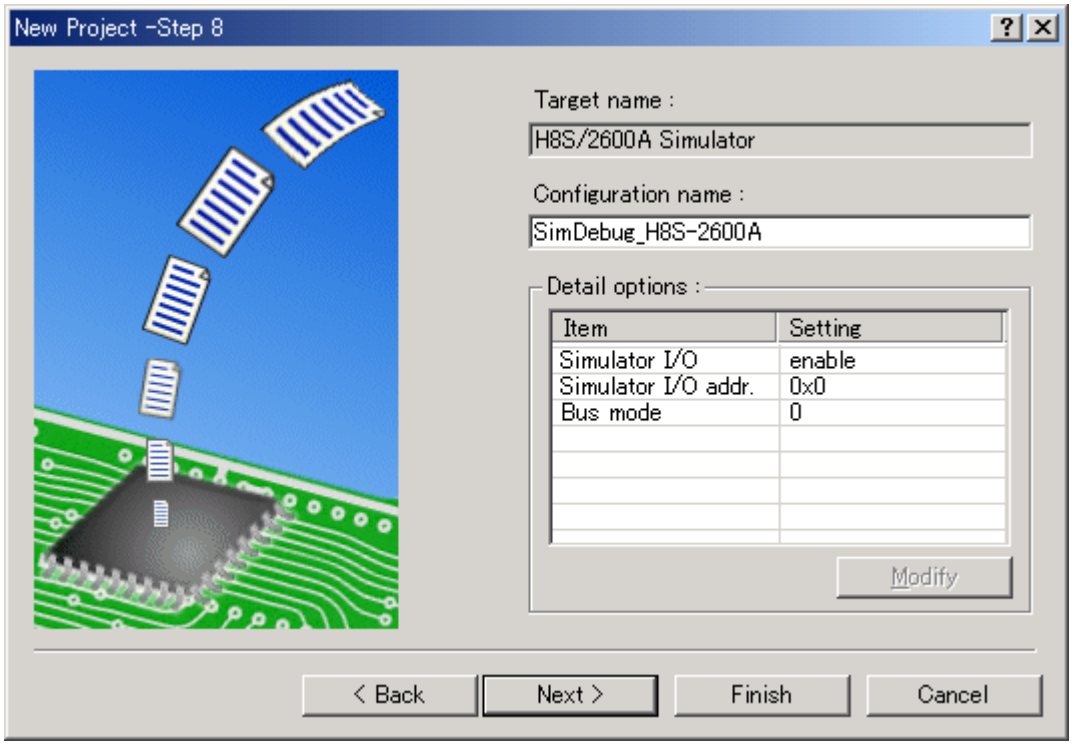


図 3-9 デバッガオプション設定画面 (Step8)

8.　ステップ8では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。

[Configuration name]　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。

[Detail options]　デバッガターゲットのオプションを設定します。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。

- [Simulator I/O]　ユーザプログラムから標準入出力またはファイル入出力を行うシステムコールは有効([Enable])または無効([Disable])
- [Simulator I/O addr.]　上記システムコールアドレス
- [Bus mode]　シミュレータ・デバッガでは現状未使用。
- [Fetch size]　フェッチサイズを指定します。[16 Bit]、[32 Bit]が指定できます。
- [Multiple]　乗算器の有効[Enable]／無効[Disable]を表示します。
- [Divide]　除算器の有効[Enable]／無効[Disable]を表示します。
- [Address space]　アドレス空間サイズを表示します。
- [Operating mode]　MCU動作モードを表示します。

Step8で[Next>]をクリックすると、ステップ9に進みます。

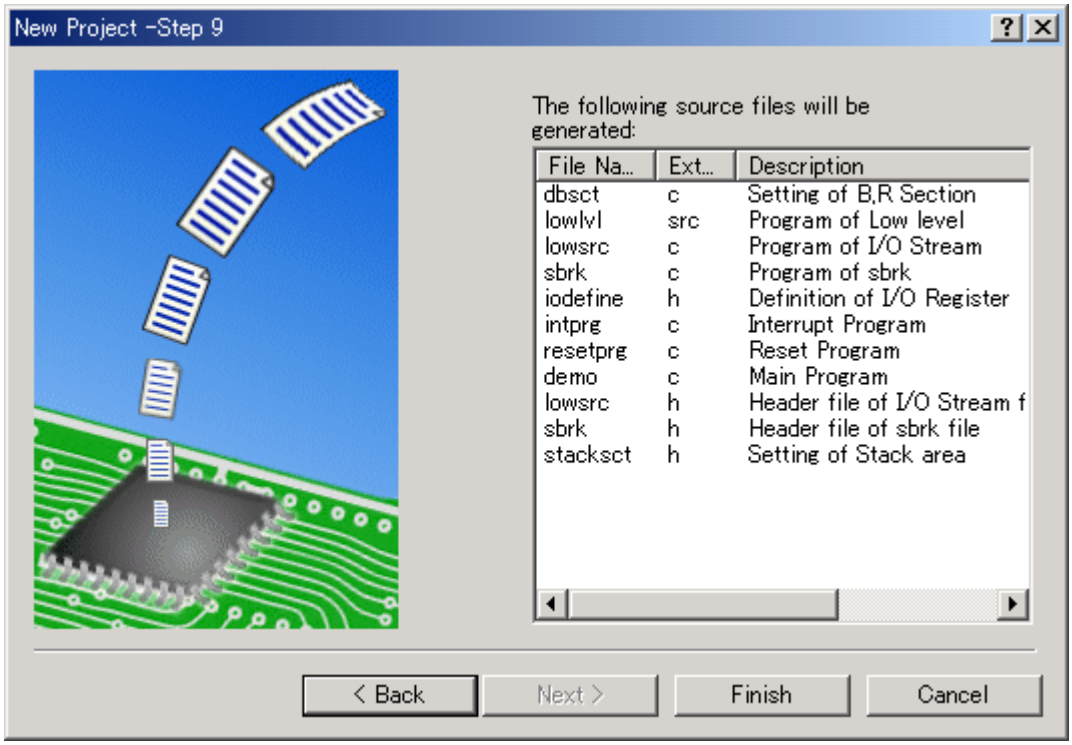


図 3-10 生成ファイル名変更画面 (Step9)

9.　ステップ9ではこれまでの設定によりHEWが生成するファイルをリスト表示します。
[File Name]　ファイル名
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。
[Extension]　拡張子
[Description]　ファイルの説明

Step9画面で[Finish]をクリックすると、[Summary]ダイアログボックスを表示します。

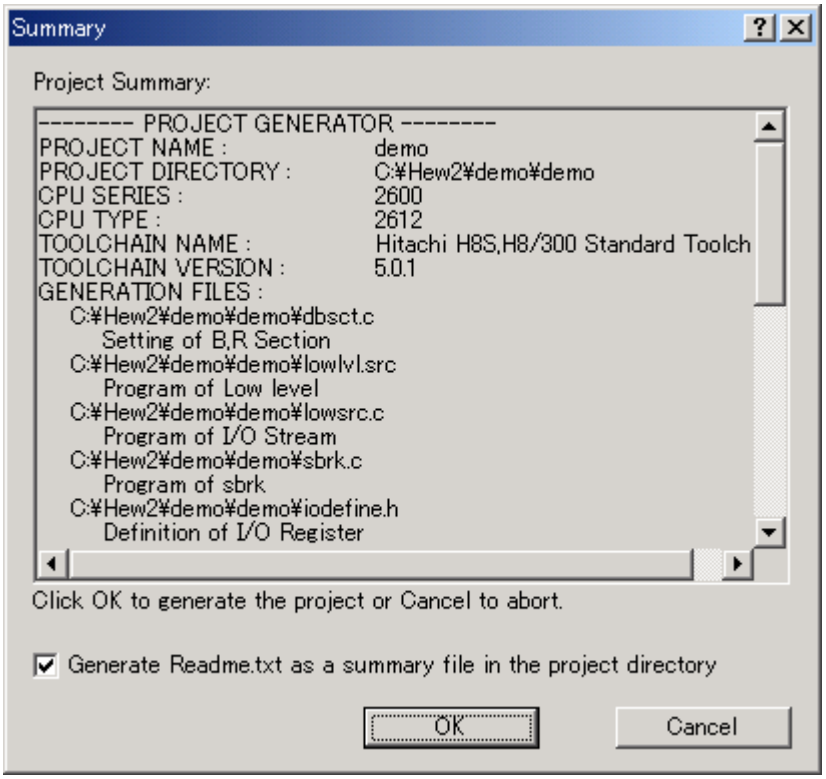


図 3-11 Summary **ダイアログボックス**

10.　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に表示しますので、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。

3.3　デバッギングプラットフォームを構築する

デバッギングプラットフォームにプログラムをロードする前に、アプリケーションシステムに合うようにプラットフォームを設定しなければなりません。必要な設定内容は、デバイスの種類、動作モード、クロックスピード、およびメモリマップなどです。その中でも特にメモリマップの設定が大切です。HEWでは、プロジェクトの生成処理でこの作業のほとんどが完了します。しかし、標準とは異なるボードのコンフィグレーションを使用する場合には、カスタマイズが必要になります。

3.3.1　メモリマップ

デバッギングプラットフォームは、デバイスのアドレス空間がROM、RAM、オンチップレジスタ、またはメモリのない領域なのかを知る必要があります。そのため、メモリマップを設定する必要があります。

プロジェクトジェネレータでデバイスの種類およびモードを選択すると、HEWは自動的にそのデ

バイスのマップおよびそのプロセッサが動作するモードを設定します。例えば、内部ROMおよびRAMを持つデバイスにおいて、これらのメモリがデバイスメモリマップ内で位置する領域をデフォルトで設定します。

また、外部メモリを使用する場合には、デバッギングプラットフォームに知らせる必要があります。

シミュレータを使用する場合、[シミュレータシステム]ダイアログボックスによりメモリマップの設定を変更することができます。詳細は「4.22.2　メモリマップおよびメモリリソースの設定を変更する」を参照下さい。

また、シミュレータを使用する場合、[Standard Toolchain]ダイアログボックスの[Debugger]タブで、設定内容を参照することができます。[Standard Toolchain]ダイアログボックス（図 3-12）の[View]ボタンをクリックすると、[CPU hardware map]ダイアログボックス（図 3-13）を表示します。

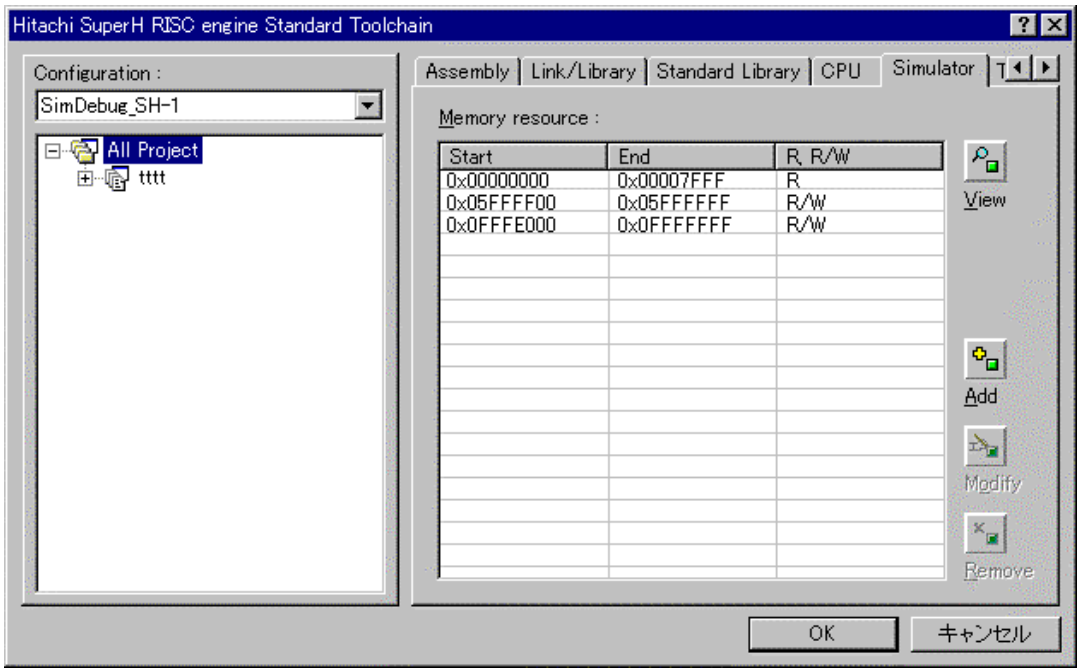


図 3-12 Standard Toolchain **ダイアログボックス**

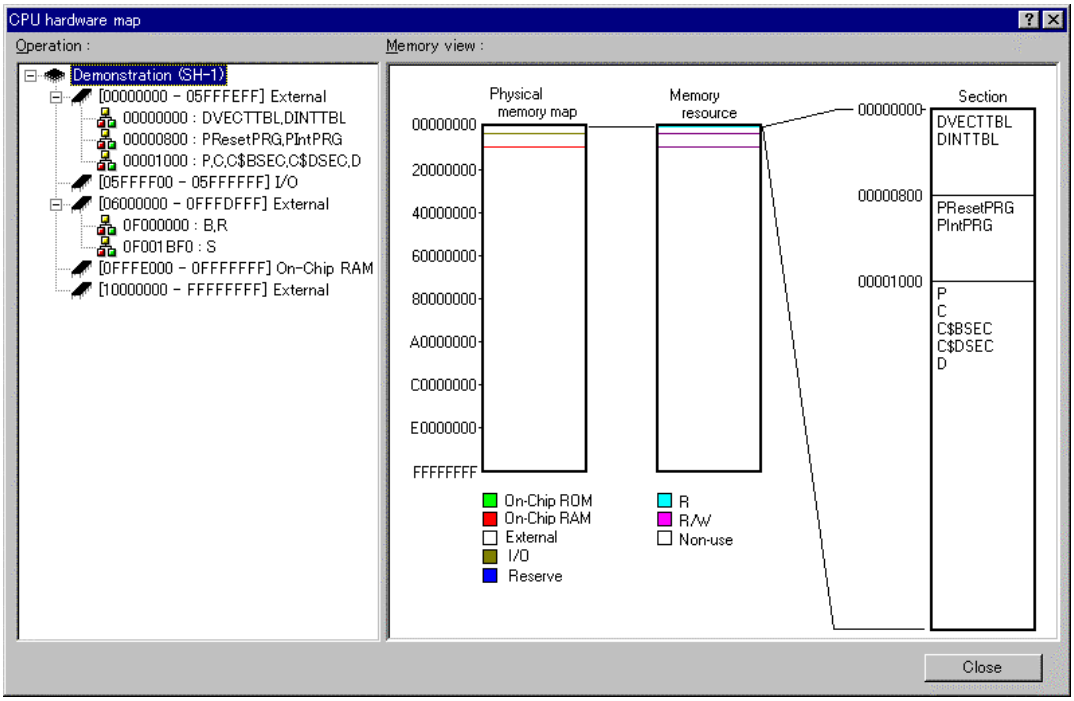


図 3-13 CPU hardware map **ダイアログボックス**

以下の情報を表示します。

[Operation]　　上段にアドレス範囲を表示します。
　　下段にリンケージマップを表示します。
[Memory view]　　メモリマップおよびメモリリソース情報を表示します。
　　確保済のメモリリソース(Allocated Area)は黄色で表示します。

3.3.2　メモリリソース

シミュレータを使用する場合、使用するアドレス範囲のメモリリソースを確保しなければ行けません。

また、シミュレータを使用する場合、図 3-12の[Standard Toolchain]ダイアログボックスの[Debugger]タブで、設定内容を参照することができます。[Debugger]タブには下記のボタンがあります。

[View]　　メモリマップ情報を表示します。

[Add]　　メモリリソースを追加します。

[Modify]　　選択したメモリリソースを変更します。

[Remove]　　選択したメモリリソースを削除します。

[Add]ボタンをクリックすると[Add memory resource]ダイアログボックス（図 3-14）を表示します。

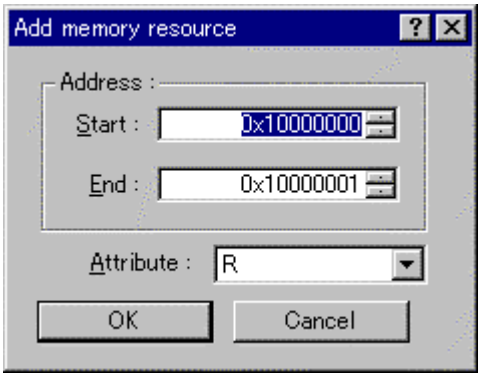


図 3-14 Add memory resource

[Add memory resource]ダイアログボックスには下記を設定してください。

[Start]　　先頭アドレス

[End]　　終了アドレス

[Attribute]　　属性（R:リードのみ、R/W:リード／ライト可）

メモリリソースは、[シミュレータシステム]ダイアログボックスの[メモリ]タブにより設定を変更することができます。詳細は「4.22.2　メモリマップおよびメモリリソースの設定を変更する」を参照下さい。

[シミュレータシステム]ダイアログボックスの[メモリ]タブで変更したメモリリソース情報は、[Standard Toolchain]ダイアログボックスに反映され、[Standard Toolchain]ダイアログボックスの変更も[シミュレータシステム]ダイアログボックスの[メモリ]タブに反映されます。

3.3.3　プログラムをダウンロードする

プログラムをダウンロードするためのメモリがシステムに十分あるならば、デバッグするプログラムをダウンロードすることができます。デフォルト設定では、現在のプロジェクトのリンカが出力したプログラムを自動的にダウンロードします。

また、プロジェクトを作成したあとにダウンロードモジュールを手動で選択することができます。これは、図 3-16に示す[デバッグの設定]ダイアログボックスを使用して行うことができます。このダイアログボックスでは、ワークスペース全体に渡りデバッグの設定を制御することができます。ダイアログボックスの左にあるツリーは、現在のプロジェクトをすべて含みます。このツリーでプロジェクトを選ぶと、そのプロジェクトの設定の表示、およびセッションドロップダウンリストには、セッションの選択を表示します。このリストボックスでは、複数のセッションを選択することもできますし、すべてのセッションを選択することもできます。複数のセッションを選択した場合には、１つま

たは複数のセッションの設定を同時に修正することができます。[デバッグの設定]ダイアログボックスは、以下のデバッグオプションを表示します。

- 現在のプロジェクトのための現在のデバッグターゲットおよびセッションの選択
- 現在のプロジェクトのためのダウンロードモジュールおよびセッションの選択

ダウンロードモジュールリストは、ターゲットにダウンロードするファイルの順番を表示します。このリストの中でモジュールを追加、削除、修正、上へ移動、下へ移動することができます。

➲新規のダウンロードモジュールを追加するには

1. [オプション->デバッグの設定...]を選択してください。図 3-16に示すダイアログボックスを表示します。
2. [ツリーコントロール]において、ダウンロードモジュールを追加するプロジェクトおよびコンフィグレーションを選択します。
3. [追加]ボタンをクリックしてください。図 3-15に示すダイアログボックスを表示します。
4. [フォーマット]リストボックスは、サポートしているオブジェクトフォーマットリーダのリストを含みます。これによってプロジェクトに追加するダウンロードモジュールのリーダを正しく選択することができます。
5. [ファイル名]フィールドを使うと、使用中のディスクにあるダウンロードモジュールをブラウズすることができます。また、まだモジュールをビルドしていなければ、[ファイル名を直接入力することができます。
6. [オフセット]フィールドにはオフセットアドレスを入力してください（一部のオブジェクトフォーマットに対してのみ入力できます）。
7. メモリアクセスサイズを指定します。アクセスサイズは1、2、4、および8が選択できます。
8. デバッグ情報のみをダウンロードしたい場合は、[デバッグ情報のみのダウンロード]チェックボックスをチェックします。
9. ダウンロード中にメモリベリファイを行う場合は、[ダウンロード時のメモリベリファイ]チェックボックスをチェックします。
10. ユーザが[OK]ボタンをクリックすると、デバッグダウンロードモジュールをリストの最後に追加します。そのモジュールを他のモジュールと関連した別の位置に変えたいときには、モジュールを選択して[上へ]および[下へ]ボタンを使用してモジュールの位置を正しく設定してください。

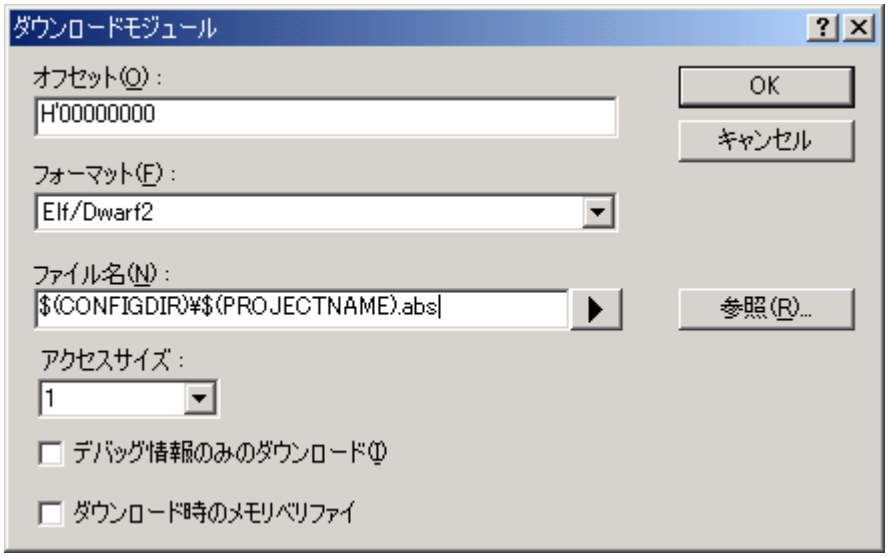


図 3-15 **ダウンロードダイアログボックス**

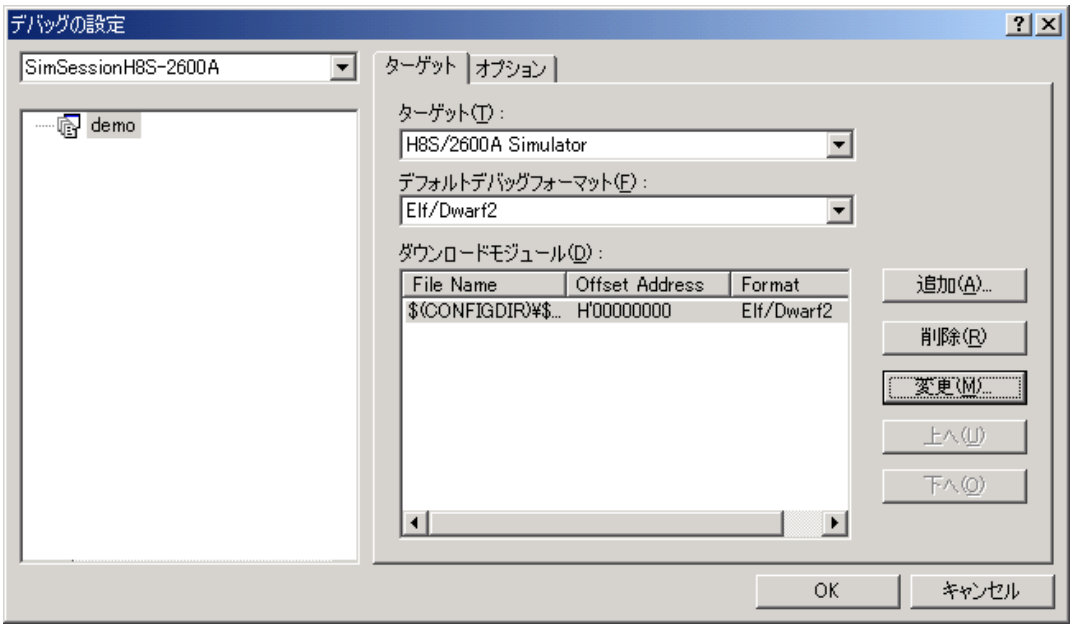


図 3-16 デバッグの設定ダイアログボックス(ターゲットタブ)

[デバッグの設定]ダイアログボックスで行った変更はすべて、[OK]ボタンをクリックするまで更新しません。

デフォルトのデバッグフォーマットは、リスト中の最初のダウンロードモジュールに設定しています。1セッションにつき、デフォルトのデバッグオブジェクトフォーマットを1つだけ指定することができます。現在インストールしているデバッガフォーマットはすべてここに表示します。

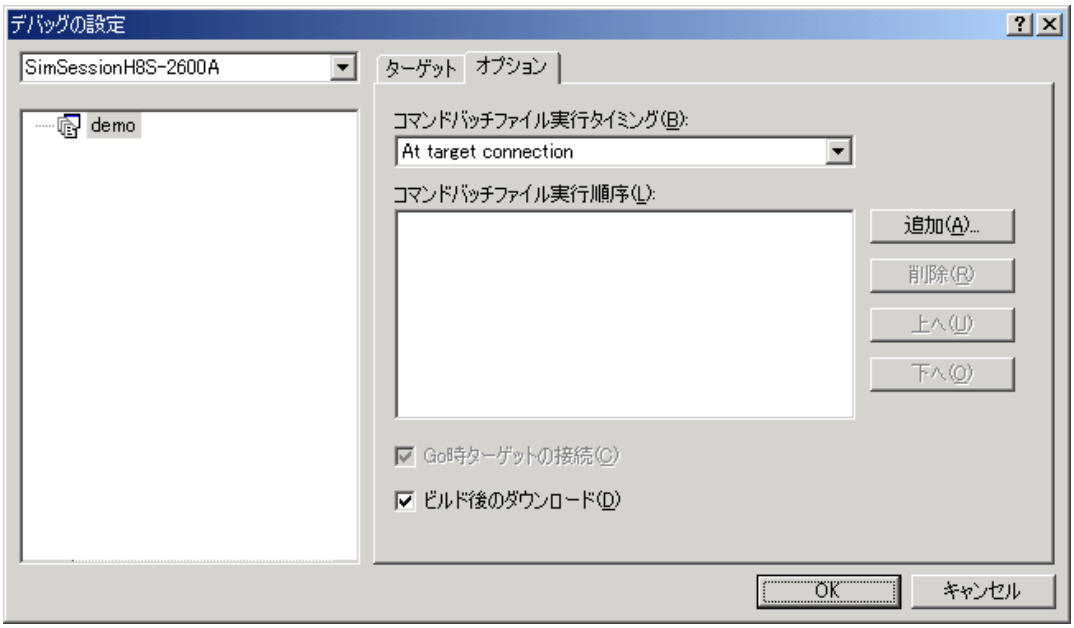


図 3-17 デバッグの設定ダイアログボックス(オプションタブ)

[デバッグの設定]ダイアログボックスの[オプション]タブ（図 3-17）では、ターゲットをいつ接続するのかを決定することができます。このオプションをチェックしていると、ユーザが[ビルド->デバッグ->実行]を選択するまでターゲットを接続しません。チェックしていないと、コンフィグレーションを開く度にターゲットを接続します。新しいワークスペースやプロジェクトを開いたときやツールバーや[ビルドコンフィグレーション]ダイアログボックスを使ってコンフィグレーションを切り替えたときにコンフィグレーションを開きます。

オプション[ビルド後にダウンロード]をチェックすると、ユーザプログラムはビルド後自動的にダウンロードされます。

オプション[ダウンロード後にリセット]をチェックすると、ユーザプログラムロード後自動的にデバッガターゲットをリセットします。

また、このタブの[コマンドバッチファイルロードタイミング]リストでは、コマンドラインバッチコマンドを指定できます。コマンドラインバッチコマンドは、コマンドラインを経由してデバッグを行います。このコマンドをいつ実行するか、[コマンドバッチファイルロードタイミング]ドロップダウンリストで選択することができます。[コマンドバッチファイルロードタイミング]ドロップダウンリストを選択してから、実行するバッチファイルを追加してください。リスト内の順番は、タイミングイベントが発生したときにコマンドラインバッチファイルを実行する順番です。

3.3.4 モジュールを手動でダウンロードする

いったんターゲットにダウンロードするモジュールを決定したら、接続したターゲット上のモジュールを手動で更新することができます。これは[デバッグ->ダウンロード]で行います。ダウンロードするモジュールは、１つまたはすべてのモジュールを選択できます。また、[ワークスペース]ウィンドウにある[Download Modules]フォルダの下のモジュールを右クリックしても更新できます。

3.3.5　モジュールを自動的にダウンロードする

[デバッグ->実行…] を選択すると、HEWは前回のダウンロードからモジュールに影響を与えるファイルまたはデバッガの設定に変更があったかどうかをチェックします。HEWは変更を見つけると、それぞれのモジュールをダウンロードする必要があるかどうかをユーザに問い合わせます。ユーザが[Yes]を選択すると、モジュールをターゲットにダウンロードします。

複数のモジュールを同じ開始アドレスに構築した場合、そのアドレスの最初のモジュールのみをデフォルトでダウンロードします。よって、[デバッグ->ダウンロード]および[デバッグ->アンロード]メニュー項目から、他のモジュールを手動でロードしたりアンロードしたりすることができます。

3.3.6　モジュールをアンロードする

[デバッグ->アンロード]メニュー項目から、ダウンロードしたモジュールを手動でアンロードすることができます。また、[ワークスペース]ウィンドウの[Download Module]ポップアップメニューからアンロードすることもできます。

モジュールをアンロードすると、そのモジュールのシンボルは、HEWデバッギングシステムからなくなりますが、ターゲットのメモリ内容は変更しません。一度アンロードしたモジュールは、再度ロードするまでデバッグすることはできません。

3.4　デバッガセッション

HEWは、ビルダオプションをコンフィグレーションへ保存することができます。同様に、HEWは、デバッガオプションをセッションに保存することもできます。セッションには、デバッギングプラットフォーム、ダウンロードするプログラム、各デバッギングプラットフォームのオプションを保存することができます。

セッションは、コンフィグレーションとは直接関連がありません。これは、複数のセッションが同じダウンロードモジュールを共有し、プログラムの不要なリビルドを避けられることを意味します。

各セッションのデータは、別々のファイルで HEW プロジェクトに保存します。詳細については、以下で説明します。

3.4.1　セッションを選択する

セッション選択するには、次の2通りの方法があります。

・ツールバーから選択する

1. ツールバーのドロップダウンリストボックス（図 3-18）からセッションを選んでください。

図 3-18 ツールバーの選択

・ダイアログボックスから選択する

1. [オプション->デバッグセッション...]を選んでください。[デバッグセッション]ダイアログボックスを表示します（図 3-19）。

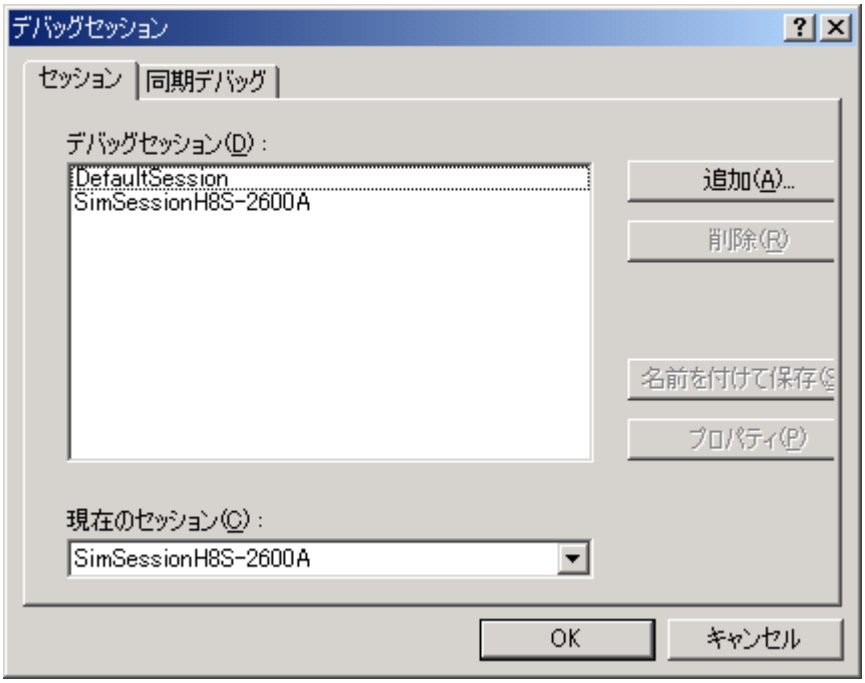


図 3-19 デバッグセッションダイアログボックス

2. [カレントセッション]ドロップダウンリストから使用したいセッションを選んでください。
3. [OK]ボタンをクリックして、セッションを設定してください。

3.4.2 セッションの追加と削除

別のセッションから設定をコピーしたり、セッションを削除したりして、新しいセッションを追加することができます。

・新しい空のセッションを追加する

1. [オプション->デバッグセッション...]を選んでください。[デバッグセッション]ダイアログボックスを表示します（図 3-19）。
2. [追加...]ボタンをクリックしてください。[新規セッション追加]ダイアログボックスを表示します（図 3-20）。
3. [新規セッションの追加]ラジオボタンをチェックしてください。
4. セッションの名前を入力してください。
5. [OK]ボタンをクリックし、[デバッグセッション]ダイアログボックスを閉じてください。
6. 入力したセッション名のファイルを新しく作成します。ファイルが既に存在する場合は、エラーを表示します。

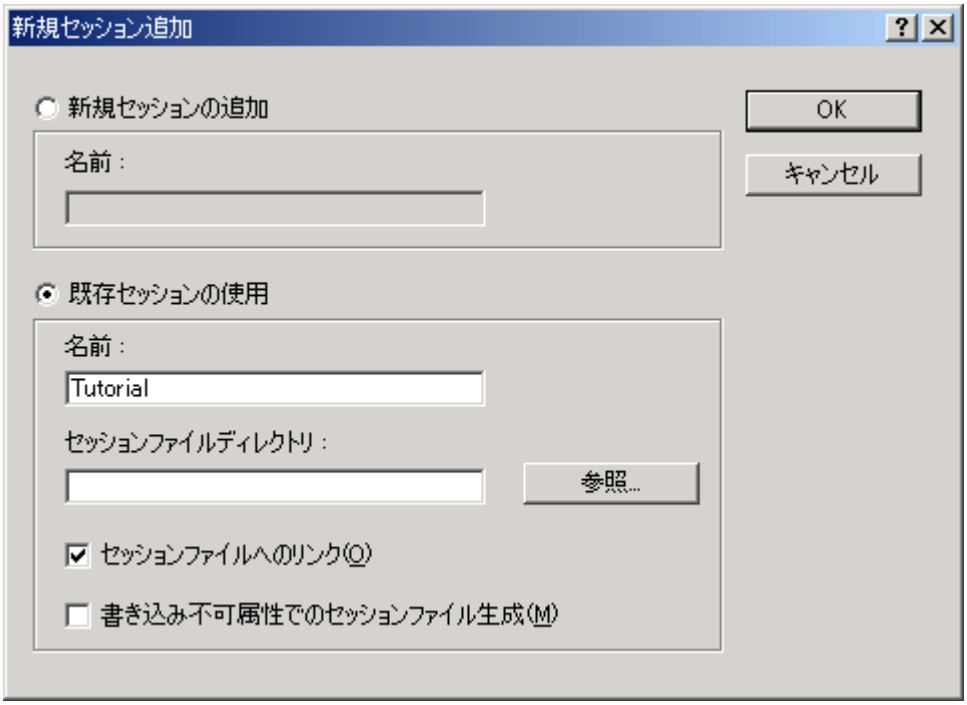


図 3-20 新規セッション追加ダイアログボックス

・既存のセッションを新しいセッションファイルにインポートする

1. [オプション->デバッグセッション...]を選んでください。[デバッグセッション]ダイアログボックスを表示します（図 3-19）。
2. [追加...]ボタンをクリックしてください。[新規セッション追加]ダイアログボックスを表示します（図 3-20）。
3. [既存セッションの使用]ラジオボタンをチェックしてください。
4. セッションの名前を入力してください。
5. 現在のプロジェクトにインポートしたい既存のセッションファイルをブラウズしてください。
   [セッションファイルへのリンク]チェックボックスをチェックしない場合、プロジェクトディレクトリにインポートした新しいセッションファイルを生成します。
   [セッションファイルへのリンク]チェックボックスをチェックした場合、プロジェクトディレクトリに新しいセッションファイルは生成せず、既存のセッションファイルにリンクします。
   [書き込み不可属性でのセッションファイル生成]チェックボックスをチェックした場合、リンクしたセッションファイルをリードオンリーで使用します。
6. [OK]ボタンをクリックし、[デバッグセッション]ダイアログボックスを閉じてください。

・セッションを削除する

1. [オプション->デバッグセッション...]を選んでください。[デバッグセッション]ダイアログボックスを表示します（図 3-19）。
2. 修正したいセッションを選んでください。
3. [削除]ボタンをクリックしてください。
   現在のセッションを削除することはできません。

4. [OK]ボタンをクリックし、[デバッグセッション]ダイアログボックスを閉じてください。

・セッションのプロパティを見る
1. [オプション->デバッグセッション...]を選んでください。[デバッグセッション]ダイアログボックスを表示します（図 3-19）。
2. 見たいプロパティのあるセッションを選んでください。
3. [プロパティ]ボタンをクリックしてください。[セッションプロパティ]ダイアログボックスを表示します（図 3-21）。

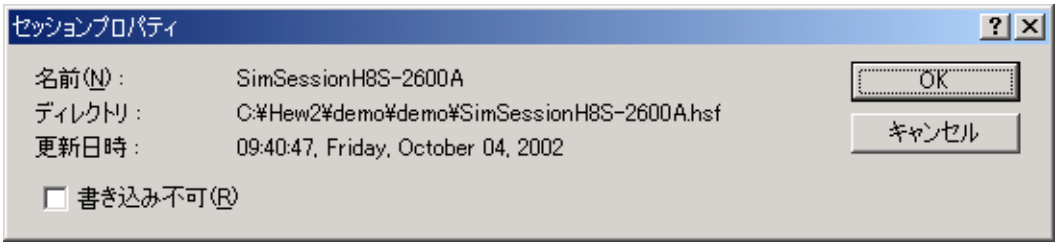


図 3-21 セッションプロパティダイアログボックス

・セッションをリードオンリーにする
1. [オプション->デバッグセッション...]を選んでください。[デバッグセッション]ダイアログボックスを表示します（図 3-19）。
2. リードオンリーにしたいセッションを選んでください。
3. [プロパティ]ボタンをクリックしてください。[セッションプロパティ]ダイアログボックスを表示します（図 3-21）。
4. [書き込み不可]チェックボックスをチェックしてください。リンクをリードオンリーにします。これは、デバッガ設定ファイルを共有する場合、およびデータを間違って修正したくない場合に便利です。
5. [OK]ボタンをクリックしてください。

・セッションを別名で保存する
1. [オプション->デバッグセッション...]を選んでください。[デバッグセッション]ダイアログボックスを表示します（図 3-19）。
2. 保存したいセッションを選んでください。
3. [名前を付けて保存]ボタンをクリックしてください。[セッションの保存]ダイアログボックスを表示します（図 3-22）。
4. 新しいファイルの場所をブラウズしてください。
5. セッションファイルを別の場所へエクスポートしたい場合は、[プロジェクトとのリンク]チェックボックスをチェックしないでください。現在のセッションの場所の代わりに、この場所をHEWで使用したい場合は、[プロジェクトとのリンク]チェックボックスをチェックしてください。
6. [OK]ボタンをクリックしてください。

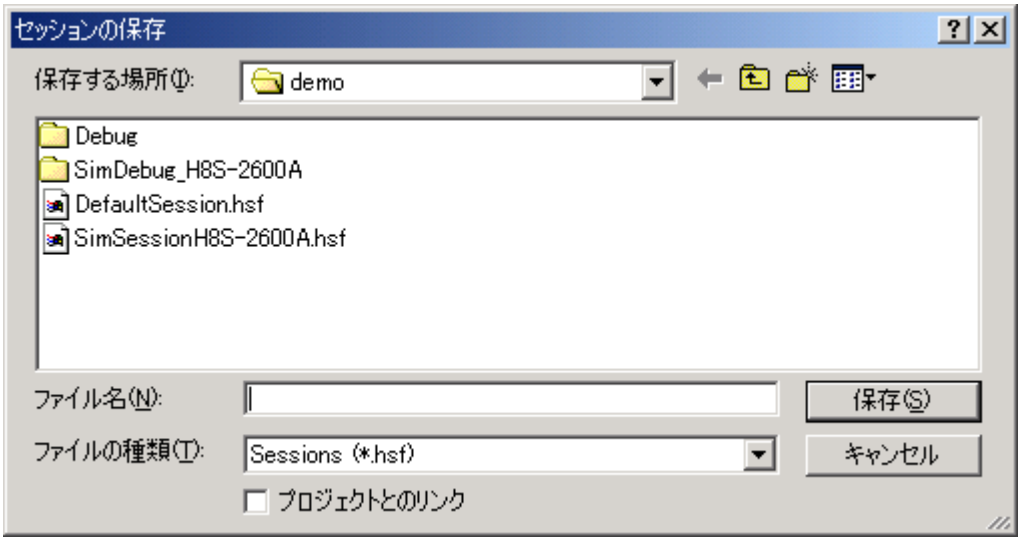


図 3-22 セッションの保存ダイアログボックス

3.4.3　セッション情報を保存する

➲セッションを保存するには

1. [ファイル->セッションの保存]を選んでください。標準のファイル保存ダイアログボックスを表示します。

【注】　セッション保存の前に、セッションを保存するかを確認するダイアログボックスを表示させることができます。ここで、[いいえ]を選択するとセッションへの変更を保存しません。このダイアログボックスは[ツール->オプション]メニューの[オプション]ダイアログボックス[ワークスペース]タブの[セッション保存前に確認]をチェックすると表示できます。

3.4.4　セッション情報の再ロード

➲セッションを再ロードするには

1. [ファイル->セッションの再ロード]を選んでください。これにより現在のセッションに加えられている変更を廃棄して、セッションを再ロードします。再ロードの前に確認ダイアログボックスを表示します。

3.4.5　複数ターゲットのデバッグ

複数ターゲットを同期してデバッグする方法は、「4.24 複数デバッギングプラットフォームを同期動作させる」を参照してください。

4. デバッグ

この章では、デバッグ操作と関連するウィンドウおよびダイアログボックス について説明します。

4.1 プログラムを表示する

この節では、プログラムをソースコードおよびアセンブリ言語ニモニックとして見る方法を説明します。

【注】 ブレークが起こると、HEW はプログラムカウンタ(PC)の場所をエディタ上に表示します。ELF/DWARF2 をベースにしたプロジェクトが、ビルド時のパスから移動していると、ソースファイルを自動的に見つけることができないことがあります。この場合、HEW はソースファイルブラウザダイアログボックスを開くので、ユーザは手動でファイルを探すことができます。このパスは、このデバッグプロジェクトのほかのソースファイルを更新するために使用します。

4.1.1 ソースコードを表示する

[ワークスペース]ウィンドウのソースファイル名を選択し、ポップアップメニューから[開く]を選択すると、HEWはエディタでソースファイルを表示します。[ワークスペース]ウィンドウのソースファイル名をダブルクリックすることによっても表示することができます。

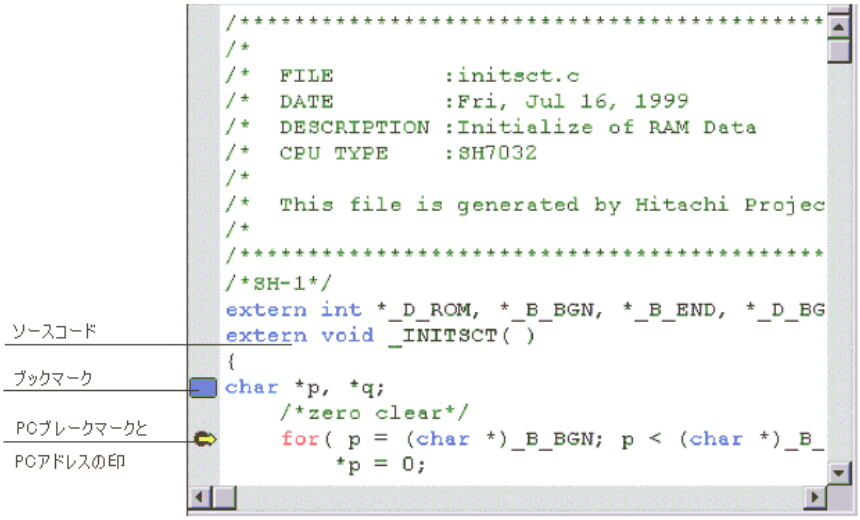


図 4-1 エディタウィンドウ

図 4-1に示すように、エディタの左側には、ブレークポイント、PCの場所などを表示する領域（カラム）があります。カラムの用途や表示する情報がわからない場合は、カーソルをカラム上に移動すると、ツールチップを表示します。

4.1.2 ソースアドレスカラム

プログラムをダウンロードすると、[エディタ]ウィンドウにソースファイルに対応するアドレスを表示することができます。（図 4-2）

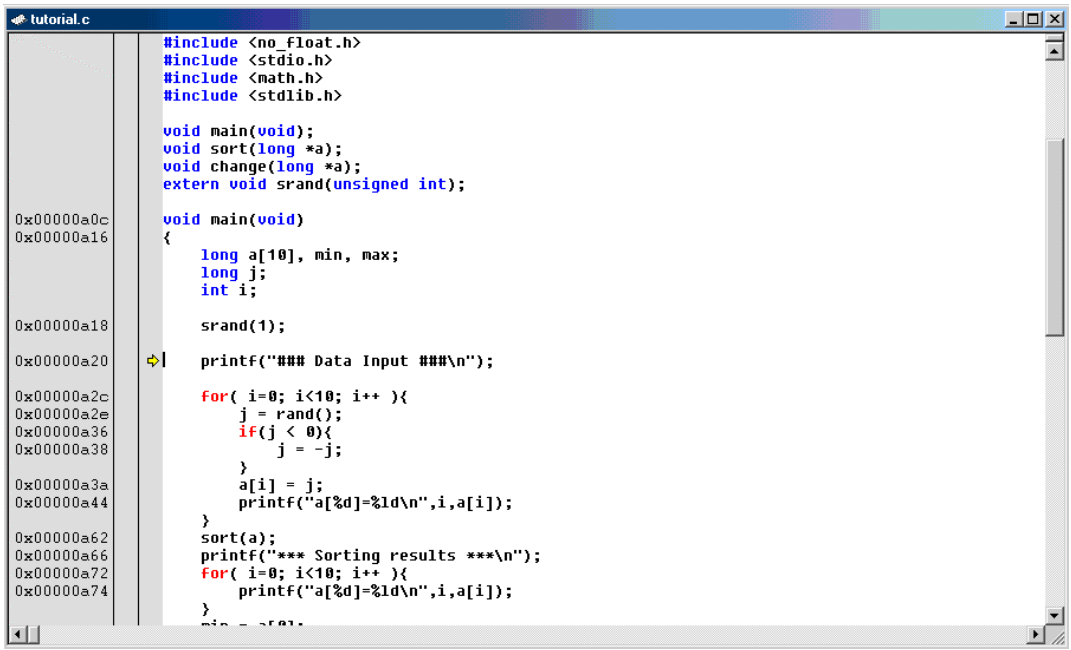


図 4-2 エディタウィンドウとアドレスカラム

4.1.3 カバレジカラム

デバッガの実行中にコードカバレージをグラフィカルに表示します。

4.1.4 エディタカラム

PC位置、PCブレークポイント位置、ブックマーク位置を表示します。
ダブルクリックによりPCブレークポイントを設定します。

4.1.5 カラムの表示／非表示を切り替える

以下の方法により、すべてのソースファイルのカラム表示を切り替えることができます。

1. [エディタ]ウィンドウを右クリックしてください。または、[編集]メニューを選択してください。
2. [表示カラムの設定...]メニュー項目をクリックしてください。
3. [エディタ全体のカラム状態]ダイアログボックスを表示します。（図4-3）
4. チェックボックスは、そのカラム表示が有効か無効かを示します。チェックしている場合は有効です。チェックボックスがグレー表示の場合、一部のファイルではカラムが有効で、別のファイルでは無効であることを意味します。
5. [OK]ボタンをクリックして、新しいカラム設定を有効にしてください。

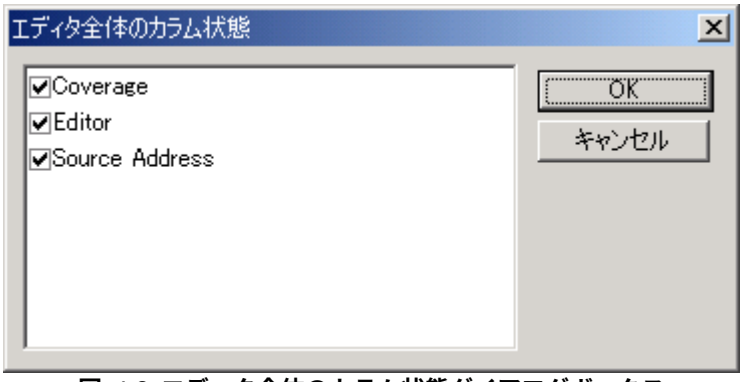


図 4-3 エディタ全体のカラム状態ダイアログボックス

[Coverage] ：カバレジカラム
[Editor] ：エディタカラム
[Source Address] ：ソースアドレスカラム

4.1.6 アセンブリ言語コードを表示する

[エディタ]ウィンドウのカーソル位置に対応するアドレス表示がある場合、ポップアップメニューから、 [逆アセンブリ]を選択することにより、[逆アセンブリ]ウィンドウを表示することができます。
[逆アセンブリ]ウィンドウの表示開始アドレスは、 [エディタ]ウィンドウのカーソル位置に対応するアドレスとなります。

ソースファイルが存在しない場合は、次のいずれかの方法でアセンブリ言語レベルでコードを表示できます。この場合、[逆アセンブリ]ウィンドウは現在のPCの場所で開きます。

- [表示->逆アセンブリ...]を選択する
- ”Ctrl+D”アクセラレータを使用する
- [逆アセンブリ]ツールバーボタン  をクリックする。

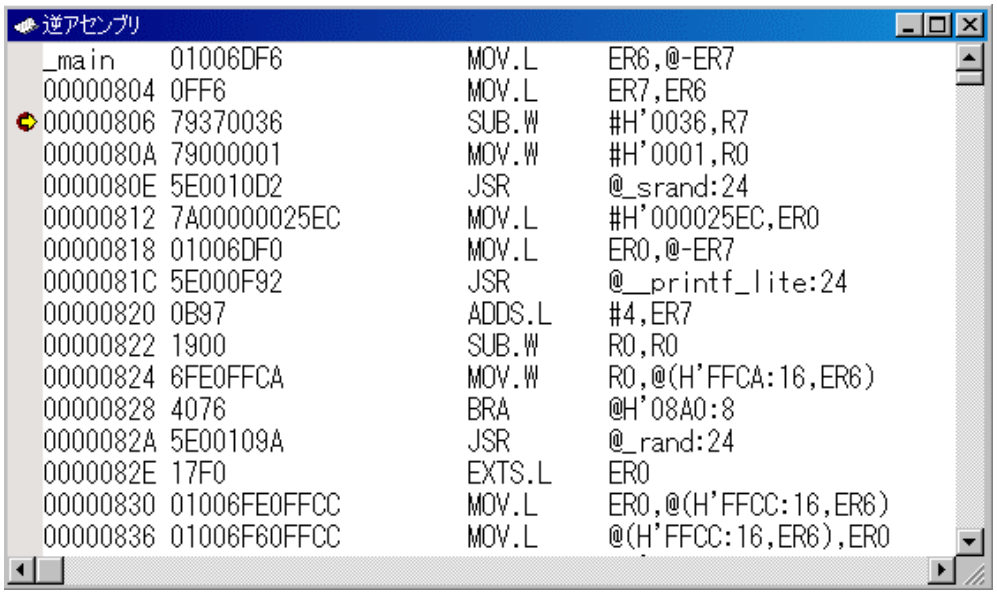


図 4-4 逆アセンブリウィンドウ

4.1.7 アセンブリ言語コードを修正する

修正したい命令をダブルクリックすることによって、アセンブリ言語コードを修正することができます。[アセンブル]ダイアログボックスが開きます。

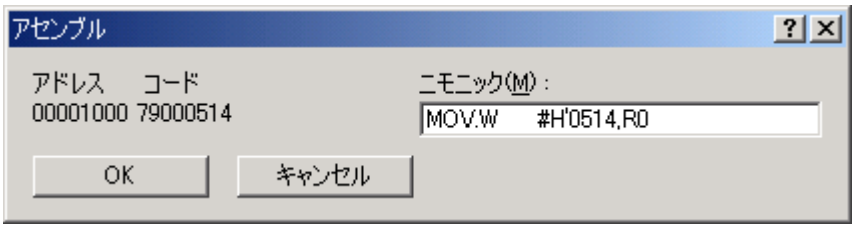


図 4-5 アセンブルダイアログボックス

アドレス、命令コード、およびニーモニックを表示します。

[ニモニック]フィールドに新しい命令を入力（または古い命令を編集）します。

'Enter'キーを押すと、メモリ内容を新しい命令コードに書き換えて、次の命令に移ります。

[OK]ボタンをクリックすると、メモリ内容を新しい命令コードに書き換えて、ダイアログボックスを閉じます。

[キャンセル]ボタンをクリックするか"Esc"キーを押すと、メモリ内容を書き換えずに、ダイアログボックスを閉じます。

【注】 アセンブリ言語コードは、現在のメモリ内容から表示しています。メモリ内容を修正すると、[逆アセンブリ]ウィンドウおよび[アセンブル]ダイアログボックスでは、新しいアセンブリ言語コードを表示します。しかし、[エディタ]ウィンドウに表示しているソースファイルは変更しません。これはソースファイルにアセンブラを含む場合も同じです。

4.1.8 特定のアドレスを見る

[逆アセンブリ]ウィンドウを使って作成したプログラムを見ているとき、プログラム内のほかのところも見たいときがあります。そのような場合、プログラム内のコードをスクロールせずに特定のアドレスに直接行くことができます。ポップアップメニューから[表示アドレス設定]を選択します。図4-6に示すダイアログボックスを表示します。

図 4-6 アドレス設定ダイアログボックス

[アドレス]エディットボックスにアドレスを入力して、[OK]ボタンをクリックするか'Enter'キーを押します。アドレスは、ラベル名で入力することも可能です。

[逆アセンブリ]ウィンドウを更新して新しいアドレスコードを表示します。オーバーロード関数またはクラス名を入力した場合、[関数選択]ダイアログボックスを開くので、関数を選択してください。これについては、このマニュアルの「4.11.3 複数ラベルをサポートする」で詳細を説明します。

4.1.9 現在のプログラムカウンタアドレスを見る

HEWでアドレスまたは値を入力できるところでは、式も入力することができます。先頭にハッシュ文字をつけたレジスタ名を入力すると、そのレジスタ内容を式の値として使用します。従って、[表示アドレス設定]ダイアログボックスで、"#pc"という式を入力すると、[ソース]または[逆アセンブリ]ウィンドウには、現在のPCアドレスを表示します。例えば、"#PC+0x100"といったPCレジスタおよびオフセットの式を入力することにより現在のPCのオフセットも表示することができます。

4.2 メモリを操作する

この節では、メモリ内容を操作する方法を説明します。操作の種類としては、色々なフォーマットでの表示、メモリブロックのフィル／移動、およびロードモジュールファイルのロード／ベリファイがあります。

4.2.1 メモリ領域を見る

メモリ領域を見るには、"Ctrl+M"アクセラレータを使用して[表示->CPU->メモリ…]を選択するか、[メモリ]ツールバーボタン  をクリックして[メモリ]ウィンドウを開きます。これにより、図 4-7に示す[表示形式]ダイアログボックスが開きます。

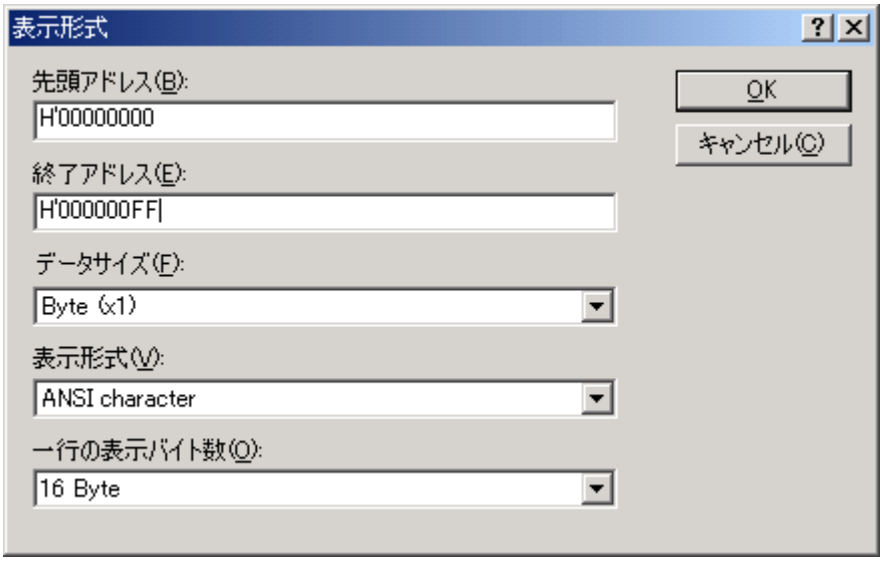


図 4-7 表示形式ダイアログボックス

[先頭アドレス] フィールドおよび[終了アドレス]フィールドに表示したい範囲をアドレス値またはシンボルで入力します。[データサイズ]および[表示形式]ドロップダウンリストから表示するデータサイズとフォーマットを選択します。また、[1行のバイト数]ドロップダウンリストから1行に表示するバイト数を選択します。[OK]ボタンをクリックするか、'Enter'キーを押すとダイアログボックスは閉じて[メモリ]ウィンドウが開きます。入力した表示開始および終了アドレス内でスクロールすることができます。

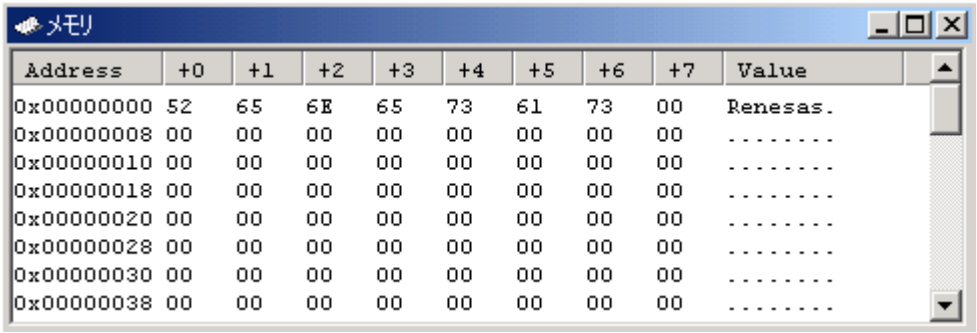


図 4-8 メモリウィンドウ

カラムを3つ表示します。

1. [Address] この行に表示したメモリデータの最初のアドレス
2. [+n] [Address]からのメモリデータで、nは行の最初のアドレスからのオフセット値
   デバッギングプラットフォーム物理メモリからアクセス幅でデータを読み出し、表示幅に変換します。
3. [Value] 他のフォーマットで表示するデータ

4.2.2 異なるフォーマットでデータを表示する

[メモリ]ウィンドウの表示フォーマットを変更する場合は、ポップアップメニューから[表示形式]を選択します。このダイアログボックスを図 4-7に示します。

メモリを異なる幅で表示して編集するには、[データサイズ]ドロップダウンリストを使用します。例えば、[Byte]オプションを選択すると、表示を更新して個々のバイトとしてメモリ領域を表示します。

データは、異なるフォーマットに変換することができます。これは3つ目の[表示形式]カラムに表示します。フォーマットのリストは、データ選択に依存します。

1行に表示するバイト数を変更するには、[1行のバイト数]ドロップダウンリストを使用します。例えば、[8 Byte]オプションを選択すると、1行に8バイト分のデータを表示します。

4.2.3 ウィンドウを分割表示する

[メモリ]ウィンドウを上下 2 分割で表示したいときは、ポップアップメニューから[分割]を選択後、分割バーを移動して分割します。

4.2.4 異なるメモリ領域を見る

[メモリ]ウィンドウの表示するメモリ領域を変更したいときは、スクロールバーを使用します。新しいアドレスをすぐに見たいときには、[表示形式]ダイアログボックスを使用します。これは、ポップアップメニューから[表示形式]を選択することによって開くことができます。

新しいアドレスを入力して[OK]ボタンをクリックするか'Enter'キーを押します。ダイアログボックスは閉じ、[メモリ]ウィンドウの表示が新しいアドレスのデータに更新します。オーバーロード関数またはクラス名を入力すると、[関数選択]ダイアログボックスが開くので、関数を選択します。

4.2.5 メモリの内容を修正する

メモリ内容は、[メモリ編集]ダイアログボックスで変更します。変更したいメモリユニット上にカーソルを移動します（[メモリ]ウィンドウ表示選択にしたがって）。メモリユニットをダブルクリックするか、'Enter'キーを押します。図 4-9のダイアログボックスを表示します。

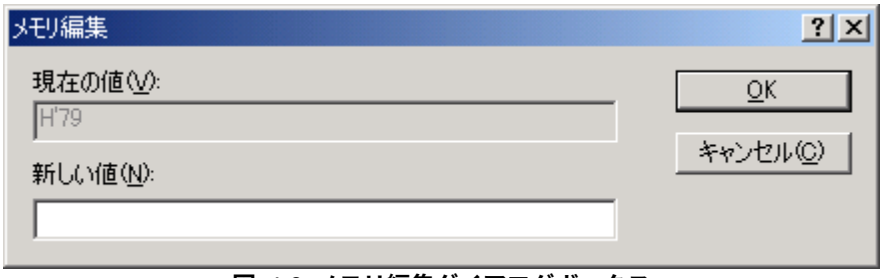


図 4-9 メモリ編集ダイアログボックス

[新しい値]フィールドには数字またはC/C++の式を入力することができます。新しい数字または式を入力したら、[OK]ボタンをクリックまたは'Enter'キーを押すと、新しい値をメモリに書き込みます。

また、メモリユニット上にマウスカーソルを移動し、キーボードより16進数を入力することにより、メモリの内容を変更することもできます。

4.2.6 メモリ範囲を選択する

メモリアドレス範囲が[メモリ]ウィンドウにある場合、最初のメモリユニット（[メモリ]ウィンドウディスプレイでの選択に従い）をクリックして最後のユニットまでマウスをドラッグすることによって領域を選択することができます。選択した領域はハイライト表示します。

4.2.7 メモリ内の値を探す

メモリ内の値を探すには、[メモリ -> 検索...]メニューを選択します。
図 4-10に示すように、[メモリ検索]ダイアログボックスを表示します。

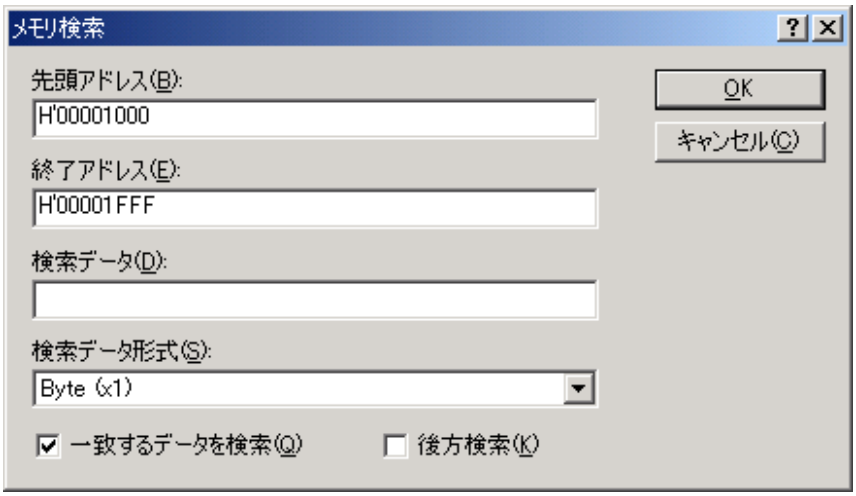


図 4-10 メモリ検索ダイアログボックス

検索するアドレス範囲の先頭および終了アドレス、および検索するデータ値を入力し、検索フォーマットを選択します。検索フォーマットに[パターン検索]を指定すると、最大256バイトのバイト列を検索できます。終了アドレスには先頭に'+'記号をつけることができ、この記号を入力すると先頭アドレス＋入力した値が終了アドレスになります。
また、パターン検索以外では検索条件として、データ一致／不一致、検索方向を指定できます。パターン検索はデータ一致および順方向のみの検索となります。

[OK]ボタンをクリックするか'Enter'キーを押します。ダイアログボックスは閉じてHEWは指定したデータの領域を検索します。データが見つかると、見つかったデータを反転表示します。

データを見つけることができなかった場合、[メモリ]ウィンドウの表示は以前と変わらず、データを見つけることができなかったことを知らせるメッセージボックスを表示します。

データが見つかった状態で、ポップアップメニューから[次を検索]を選択すると、次のアドレスから検索を続行します。

4.2.8 メモリ範囲に値をフィルする

メモリフィル機能を使って値をメモリアドレス範囲の内容に設定することができます。

同じ値をメモリ範囲に設定するには、[メモリ]ウィンドウのポップアップメニューの[フィル]を選択するか、[メモリ]ドロップダウンメニューの[フィル]を選択します。[メモリフィル]ダイアログボックスを図 4-11に示します。

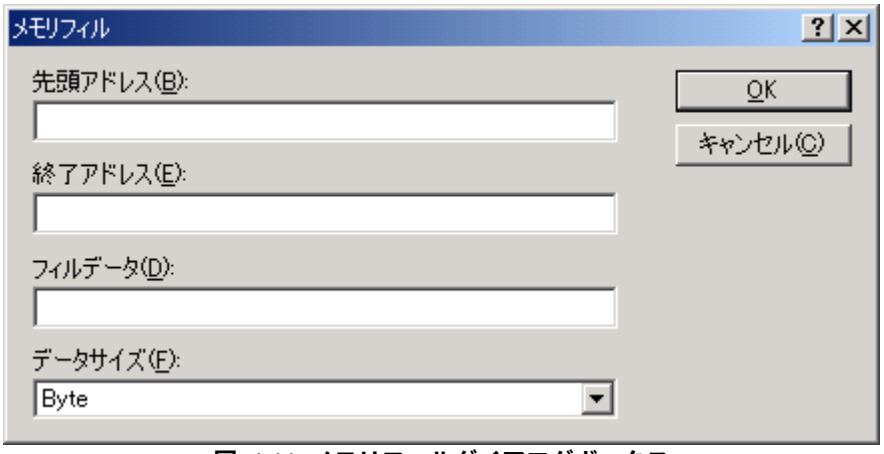


図 4-11 メモリフィルダイアログボックス

[メモリ]ウィンドウでアドレス範囲を選択した場合には、指定した開始および終了アドレスを表示します。

[データサイズ]ドロップダウンリストからフォーマットを選択して[フィルデータ]フィールドにデータ値を入力します。[ベリファイ]チェックボックスをチェックすることによりフィルするデータ値とフィル後のメモリデータを比較しながらフィルすることもできます。[OK]ボタンをクリックするか'Enter'キーを押すと、ダイアログボックスが閉じて新しい値をメモリ領域に書き込みます。

4.2.9 メモリ領域をコピーする

メモリコピー機能を使用してメモリ領域をコピーすることができます。メモリ領域を選択してポップアップメニューから[コピー...]を選択すると、[メモリコピー]ダイアログボックスを表示します。（図 4-12）

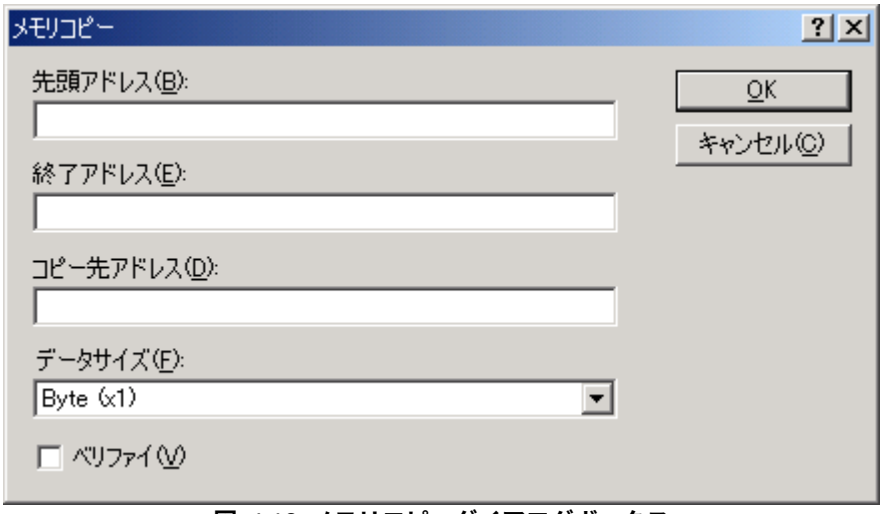


図 4-12 メモリコピーダイアログボックス

[メモリ]ウィンドウで選択したコピー元の開始アドレスおよび終了アドレスは、[先頭アドレス]および[終了アドレス]フィールドに表示します。[ベリファイ]チェックボックスをチェックすることによりコピー元とコピー先を比較しながらコピーすることもできます。[データサイズ]リストボックスでコピー単位を選択することもできます。コピー先の開始アドレスを[コピー先アドレス]フィールドに入力して[OK]ボタンをクリックするか、'Enter'キーを押すと、ダイアログボックスを閉じてメモリブロックを新しいアドレスにコピーします。

4.2.10 メモリ領域を保存、検証する

メモリ保存機能を使用してアドレス空間のメモリ領域をディスクファイルに保存することができます。
[ファイル->メモリの保存...]を選択して[名前を付けて保存(メモリ)]ダイアログボックスを開きます。

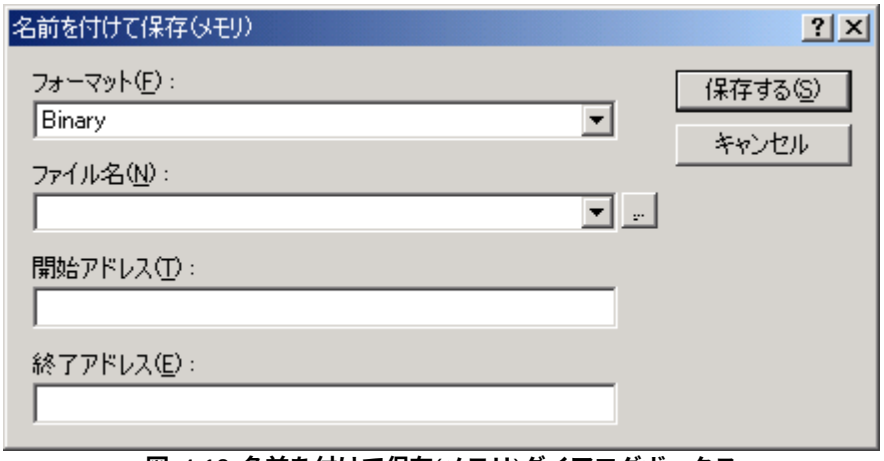


図 4-13 名前を付けて保存(メモリ)ダイアログボックス

保存するメモリブロックの開始および終了アドレス、ファイル名、ファイルフォーマットを入力します。[ファイル名]ドロップダウンリストには、メモリを保存するために使用した過去4つのファイル名を表示します。

また[参照...]ボタンをクリックすると、標準の[名前を付けて保存]ダイアログボックスを開きます。[OK]ボタンをクリックするか'Enter'キーを押すと、ダイアログボックスを閉じて、メモリブロックを指定フォーマットファイルとしてディスクに保存します。ファイルの保存が完了すると、確認のメッセージボックスを表示することがあります。

[アクセスサイズ]ドロップダウンリストから、保存時のアクセスサイズを選択できます。保存するメモリがリトルエンディアンの時は、アクセスサイズによってデータの並びが異なります。

メモリベリファイ機能を使用してアドレス空間のメモリ領域を検証することができます。[ファイル->メモリのベリファイ...]を選択して[メモリベリファイ]ダイアログボックスを開きます。

[アクセスサイズ]ドロップダウンリストから、ベリファイ時のアクセスサイズを選択できます。

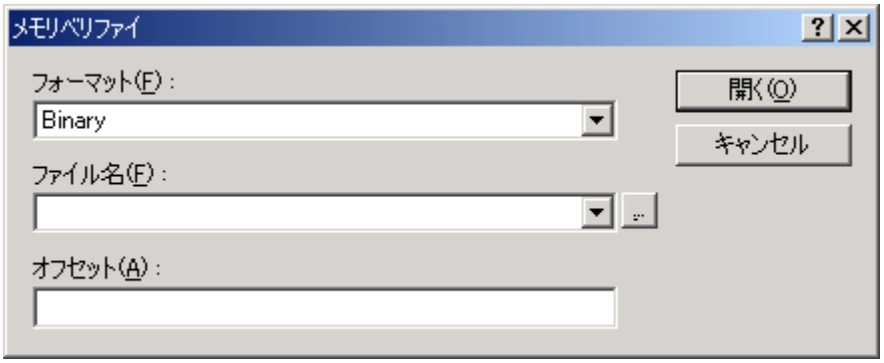


図 4-14 メモリベリファイダイアログボックス

4.2.11 ウィンドウ内容更新を抑止する

ユーザプログラム実行停止時などに、自動的に[メモリ]ウィンドウ内容を更新しないようにできます。

ポップアップメニューの[表示固定]をチェックします。[メモリ]ウィンドウの表示がグレー表示に変わります。

4.2.12 ウィンドウ内容を更新する

[メモリ]ウィンドウの内容を強制的にアップデートできます。

ポップアップメニューから[最新の情報に更新]を選択します。

4.2.13 メモリ内容を比較する

2つのメモリブロック内容を比較することができます。メインメニューから[メモリ->比較...]を選択するか、[メモリ]ウィンドウのポップアップメニューから[比較...]を選択して[メモリ比較]ダイアログボックスを開きます。

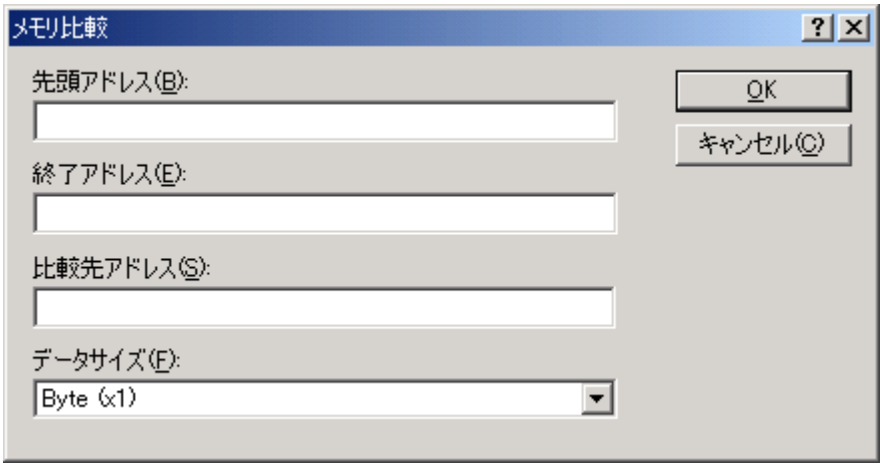


図 4-15 メモリ比較ダイアログボックス

比較フォーマット([データサイズ])、比較元メモリ領域の開始アドレス([先頭アドレス])と終了アドレス([終了アドレス])、および比較先メモリ領域の先頭アドレス([比較先アドレス])を入力します。[メモリ]ウィンドウでメモリブロックを反転表示していれば、ダイアログボックスを表示したときに開始アドレスと終了アドレスを自動的に設定します。

不一致個所があった場合は、そのアドレスをメッセージボックスに表示します。

4.2.14 メモリ領域をファイルからロードする

デバッグプラットフォームのメモリにファイルをロードできます。[メモリ]ウィンドウのポップアップメニューから[ロード...]を選択して、[ダウンロードプログラム]ダイアログボックスを開きます。

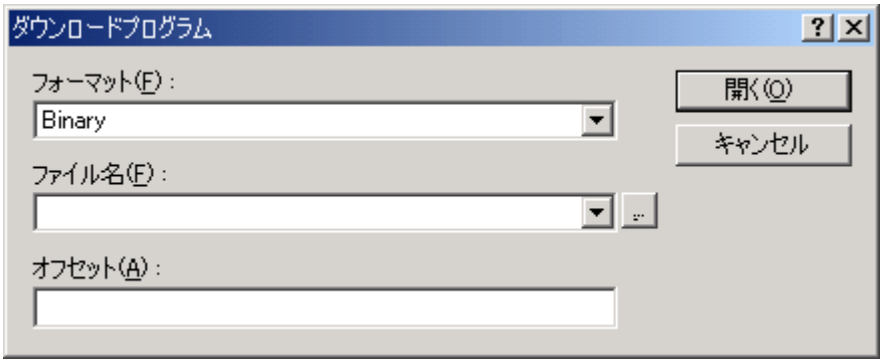


図 4-16 ダウンロードプログラムダイアログボックス

ファイルフォーマット([フォーマット])、ファイル名([ファイル名])を入力します。オフセット([オフセット])は、ロードアドレス値を変更するときはオフセット値、変更しないときは 0 を指定します。

4.3 メモリ内容を画像形式で表示する

[画像]ウィンドウを使用すると、メモリ内容を画像形式で表示することができます。

4.3.1 画像ウィンドウを開く

[表示->グラフィック->画像...]を選択するか、[画像]ツールバーボタン  をクリックすると、図4-17に示す[画像プロパティ]ダイアログボックスが開きます。

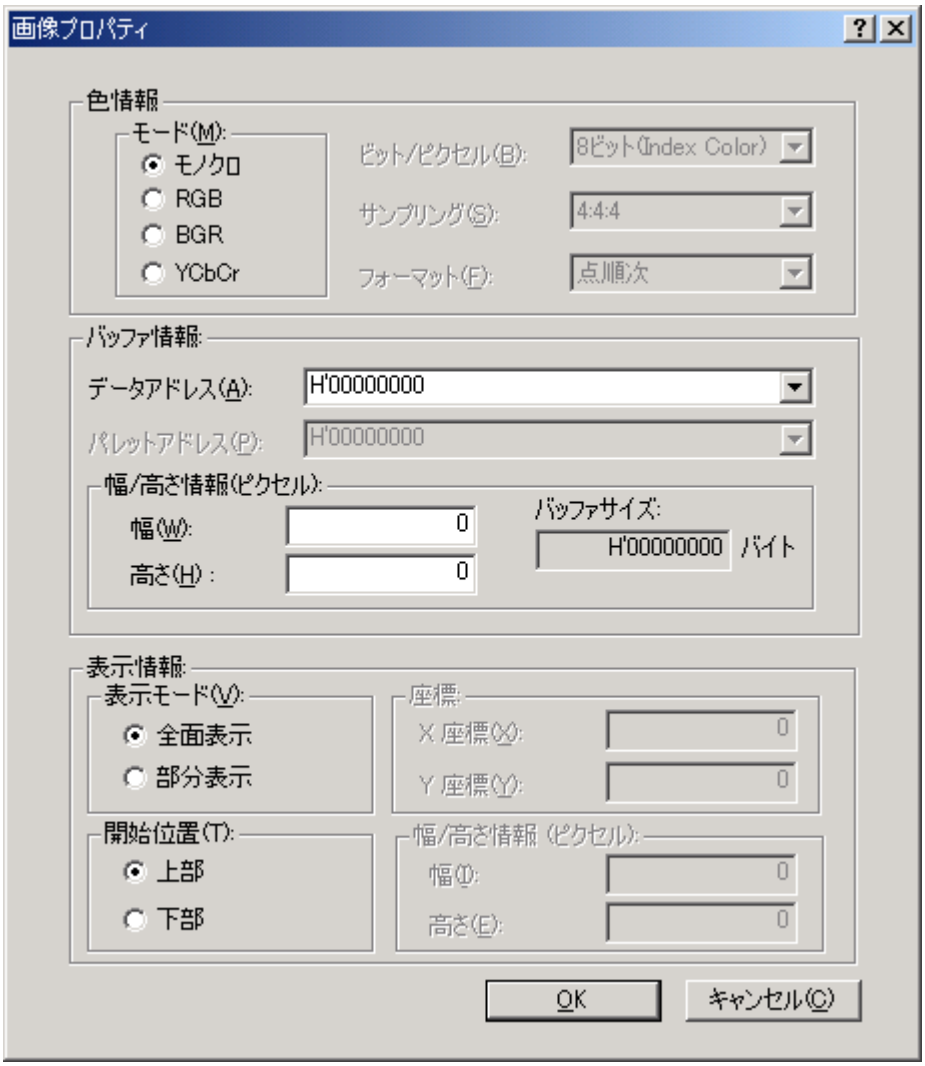


図 4-17 画像プロパティダイアログボックス

[画像プロパティ]ダイアログボックスでは[画像]ウィンドウの表示方法を指定します。

[色情報]	表示する画像のカラー情報を指定します。	
[モード]	フォーマットを指定します。	
	[モノクロ]	白黒で表現します。
	[RGB]	R(赤)、G(緑)、B(青)で表現します。
	[BGR]	B(青)、G(緑)、R(赤)で表現します。
	[YCbCr]	Y(輝度)、Cb(青色の色差)、Cr(赤色の色差)で表現します。
[ビット/ピクセル]	選択した[モード]によって、ビット/ピクセルを指定します。(RGB/BGR 選択時有効)	
[サンプリング]	サンプリングのフォーマットを指定します。(YCbCr 選択時有効)	
[フォーマット]	点順次/面順次を指定します。(YCbCr 選択時有効)	
[バッファ情報]	データの格納場所、サイズ、パレットのアドレスを指定します。	
[データアドレス]	表示する画像データのメモリ開始アドレスを指定します。(16 進表示)	
[パレットアドレス]	カラーパレットデータのメモリ開始アドレスを指定します。(16 進表示)(RGB/ BGR の 8Bit 選択時有効)	
[幅/高さ情報(ピクセル)]	画像の幅と高さを指定します。	
	[幅]	画像の幅を指定します。(接頭辞省略時は 10 進で入力、10 進表示)
	[高さ]	画像の高さを指定します。(接頭辞省略時は 10 進で入力、10 進表示)
	[バッファサイズ]	幅と高さから画像のバッファサイズを表示します。(16 進表示)
[表示情報]	画像全体中の表示部分の位置、サイズ、データ開始位置を指定します	
[表示モード]	画像の全体表示/部分表示を指定します。	
	[全面表示]	画像を全体表示します。
	[部分表示]	画像を部分表示します。
[開始位置]		
	[上部]	左上からデータを表示します。
	[下部]	左下からデータを表示します。
[座標]	部分表示する画像の開始位置を指定します。([部分表示]選択時有効)	
	[X 座標]	開始位置の X 座標を指定します。(接頭辞省略時は 10 進で入力、10 進表示)
	[Y 座標]	開始位置の Y 座標を指定します。(接頭辞省略時は 10 進で入力、10 進表示)
[幅/高さ情報(ピクセル)]	部分表示する画像の幅と高さを指定します。	
	[幅]	表示の幅を指定します。(接頭辞省略時は 10 進で入力、10 進表示)
	[高さ]	表示の高さを指定します。(接頭辞省略時は 10 進で入力、10 進表示)

[画像プロパティ]ダイアログボックスに設定後、[OK]ボタンをクリックすると[画像]ウィンドウが開きます。

[画像]ウィンドウ表示後もポップアップメニューの[プロパティ...]を選択することで本ダイアログボックスを表示して表示内容を変更できます。

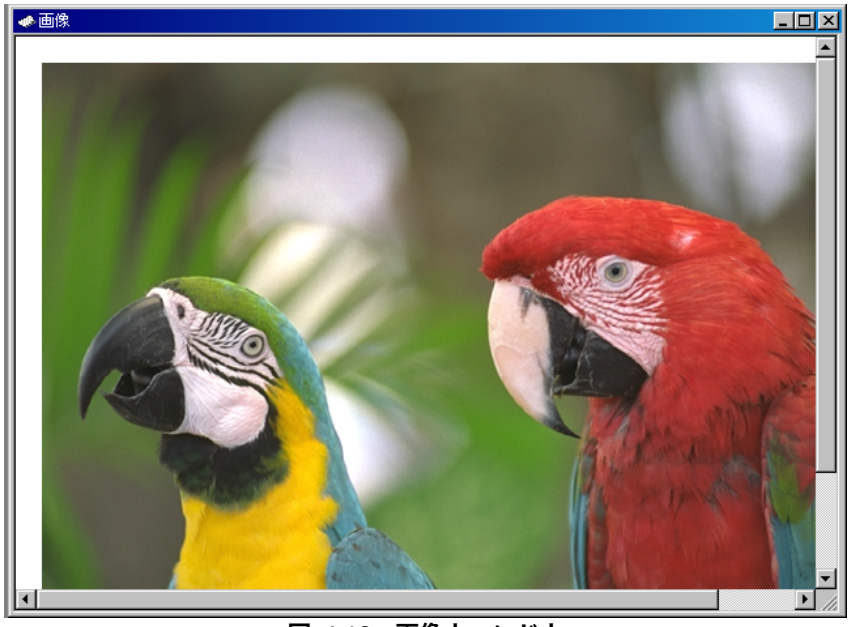


図 4-18　画像ウィンドウ

メモリの内容を画像で表示します。

4.3.2　ウィンドウを自動更新する

ポップアップメニューから[自動更新->更新しない]を選択すると、ウィンドウ内容を自動更新しません。

ポップアップメニューから[自動更新->停止]を選択すると、ユーザプログラム実行停止時にウインドウ内容を更新します。

ポップアップメニューから[自動更新->リアルタイム]を選択すると、[Response]コマンドで指定した命令間隔ごとにウィンドウ内容を更新します。

4.3.3　ウィンドウを更新する

ポップアップメニューから[更新]を選択すると、直ちにウィンドウ内容を更新します。

4.3.4　ピクセル情報を表示する

ウィンドウ内をダブルクリックするとマウスポインタの位置のピクセル情報を[ピクセル情報]ダイアログボックスに表示します。

図 4-19 ピクセル情報ダイアログボックス

カーソル位置のピクセル情報を表示します。

[カラーモード]	画像のフォーマットを表示します。	
[ピクセル]	カーソル位置のカラー情報を表示します。（10 進表示）	
[位置]	カーソル位置を X 座標、Y 座標で表示します。（10 進表示）	
	[X]	カーソル位置の X 座標を表示します。
	[Y]	カーソル位置の Y 座標を表示します。
[バッファサイズ]	バッファサイズを表示します。（10 進表示）	
	[幅]	バッファの幅を表示します。
	[高さ]	バッファの高さを表示します。
[画像サイズ]	表示の幅と高さを表示します。（10 進表示）	
	[幅]	幅を表示します。
	[高さ]	高さを表示します。

4.4 メモリ内容を波形形式で表示する

[波形]ウィンドウを使用すると、メモリ内容を波形形式で表示します。

4.4.1 波形ウィンドウを開く

[表示->グラフィック->波形...]を選択するか、[波形]ツールバーボタン  をクリックすると、図 4-20に示す[波形プロパティ]ダイアログボックスが開きます。

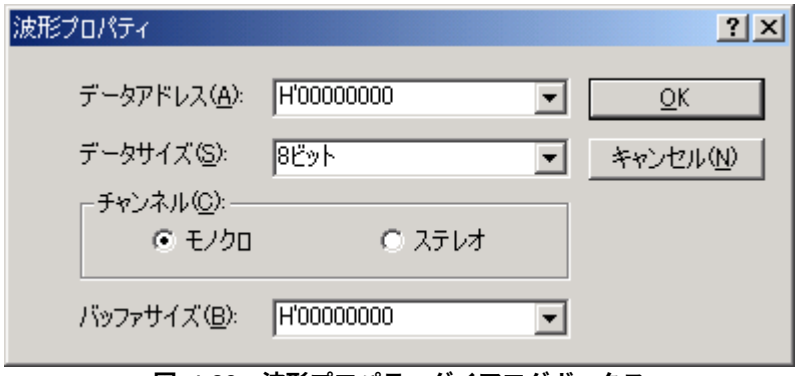


図 4-20　波形プロパティダイアログボックス

表示する波形形式を指定します。下記項目を指定できます。

[データアドレス]　データのメモリ開始アドレスを指定します。（16 進表示）
[データサイズ]　8 ビット / 16 ビットを指定します。
[チャンネル]　モノラル/ステレオを指定します。
[バッファサイズ]　データのバッファサイズを指定します。（16 進表示）

[波形プロパティ]ダイアログボックスに設定後、[OK]ボタンをクリックすると[波形]ウィンドウが開きます。

[波形]ウィンドウ表示後もポップアップメニューの[プロパティ...]を選択することで本ダイアログボックスを表示して表示内容を変更できます。

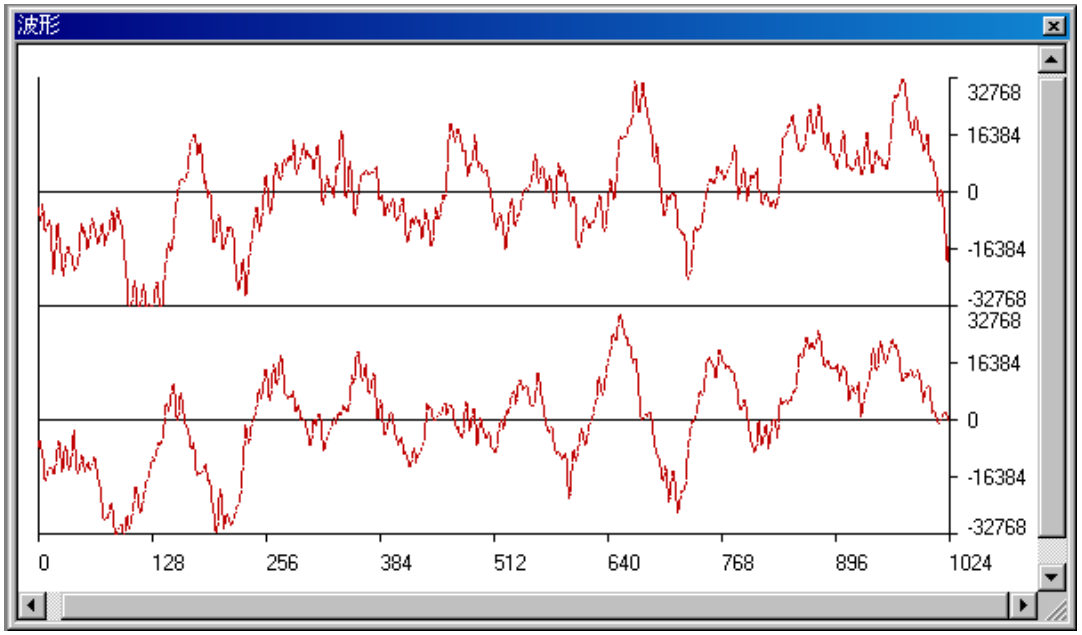


図 4-21　波形ウィンドウ

メモリ内容を波形で表示します。横軸(X)にサンプリングデータ数、縦軸(Y)にサンプリング値を表示します。

4.4.2 ウィンドウを自動更新する

ポップアップメニューから[自動更新->更新しない]を選択すると、ウィンドウ内容を自動更新しません。

ポップアップメニューから[自動更新->停止]を選択すると、ユーザプログラム実行停止時にウインドウ内容を更新します。

ポップアップメニューから[自動更新->リアルタイム]を選択すると、[Response]コマンドで指定した命令間隔ごとにウィンドウ内容を更新します。

4.4.3 ウィンドウを更新する

ポップアップメニューから[更新]を選択すると、直ちにウィンドウ内容を更新します。

4.4.4 拡大表示する

ポップアップメニューから[伸張]を選択すると、横軸を拡大して表示します。

4.4.5 縮小表示する

ポップアップメニューから[圧縮]を選択すると、横軸を縮小して表示します。

4.4.6 最初のサイズに戻す

ポップアップメニューから[元に戻す]を選択すると、最初のサイズに戻して表示します。

4.4.7 拡大/縮小倍率を設定する

ポップアップメニューの[圧縮・伸張倍率]サブメニューで拡大/縮小倍率を２、４、８倍から選択します。

4.4.8 横軸のサイズを設定する

ポップアップメニューの[スケール]サブメニューで横軸のサイズを１２８、２５６、５１２ピクセルから選択します。

4.4.9 カーソルを非表示にする

ポップアップメニューの[カーソル削除]をチェックすると、カーソルを非表示にします。

4.4.10 サンプリング情報を表示する

ポップアップメニューから[サンプリング情報...]を選択すると、[サンプリング情報]ダイアログボックスを表示します。

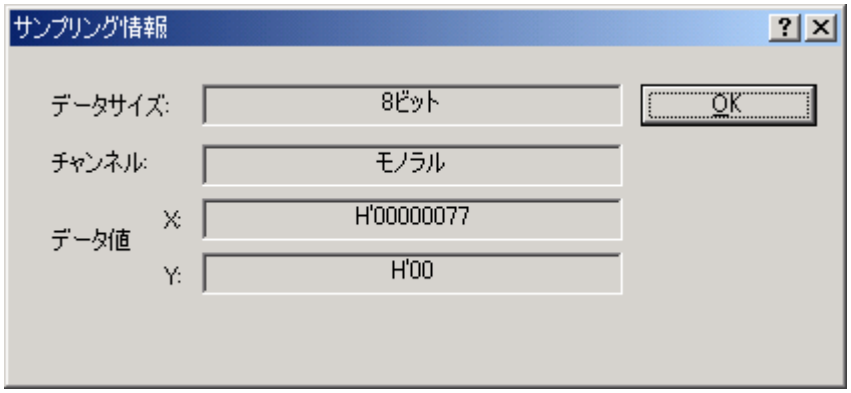


図 4-22 サンプリング情報ダイアログボックス

[波形]ウィンドウのカーソル位置のサンプリング情報を表示します。下記情報を表示します。

[データサイズ]	8 ビット / 16 ビットを表示します。	
[チャンネル]	データのチャンネルを表示します。	
[データ値]	[X]	カーソル位置の X 座標を表示します。
	[Y]	カーソル位置の Y 座標を表示します。 (ステレオ選択時は上下 2 つの Y 座標を表示します)

4.5 I/O レジスタを見る

CPUおよびROM/RAMと同様、マイクロコントローラには内蔵周辺モジュールがあります。デバイスによって周辺モジュールの数および型は異なりますが、代表的なモジュールとしては、DMAコントローラ、シリアルコミュニケーションインタフェース、A/Dコンバータ、インテグレーテッドタイマユニット、バスステートコントローラおよびウォッチドッグタイマなどがあります。マイクロコントローラのアドレス空間にマッピングしたアクセスレジスタは、内蔵周辺モジュールを制御します。

[メモリ]ウィンドウは、連続したメモリアドレスのデータをバイト、ワード、ロングワード、単精度浮動小数点、倍精度浮動小数点、またはASCII値として表示することがでます。これに対しI/Oレジスタは、非連続なメモリアドレスに異なるサイズでレジスタが割り付いているので、HEWはこれらのレジスタを簡単に確認したり設定したりすることができるように[IO]ウィンドウを提供します。

4.5.1 IO ウィンドウを開く

[IO]ウィンドウを開くには、[表示->CPU->IO]を選択するか、[IOの表示]ツールバーボタン  をクリックします。内蔵周辺と一致するモジュールがI/Oレジスタ情報を構成します。[IO]ウィンドウを最初に開くと、モジュール名の一覧表のみを表示します。

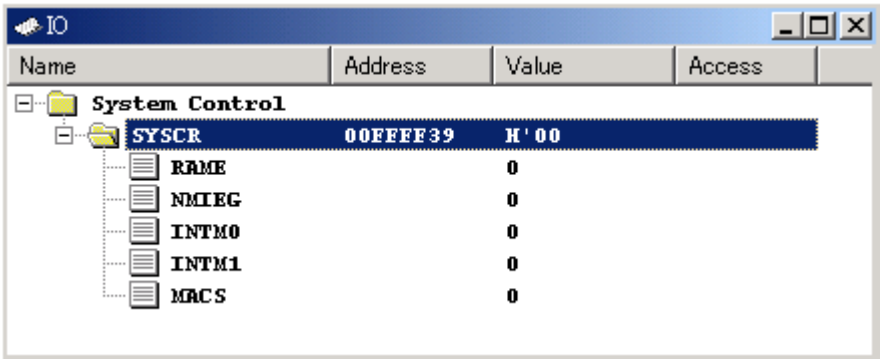


図 4-23 IO ウィンドウ

4.5.2 I/O Register 表示を拡張する

I/Oレジスタの名前、アドレス、および値を表示するには、モジュール名をダブルクリックするか、クリックまたはカーソルを使用することによってモジュール名を選択し、'Enter'キーを押します。モジュールの表示が拡張し、その周辺モジュールのそれぞれのレジスタおよびその名前、アドレス、および値を表示します。モジュール名を再びダブルクリックする（または'Enter'キーを押す）と、表示しているI/Oレジスタを閉じます。

ビットレベルで表示するには、[レジスタ]ウィンドウと同様のやり方でI/Oレジスタを拡張します。

4.5.3 I/O ファイルの手動ロード

I/O ファイルを手動でロードする場合は、[IO]ウィンドウ上での右クリックで表示されるポップアップメニューから[IO ファイルのロード...]を選択します。標準の[ファイルを探す]ダイアログボックスを表示します。ロードしたいファイルを選択して[OK]を選択します。これにより指定 I/O ファイルを[IO ウィンドウ]にロードします。I/O ファイルフォーマットは「付録 D I/O ファイルフォーマット」

を参照してください。

4.5.4 I/O レジスタの内容を修正する

I/Oレジスタの値を編集するには、ウィンドウに対して16進数を直接入力します。さらに複雑な式を入力するには、レジスタをダブルクリックするか'Enter'キーを押してレジスタの内容を修正するためのダイアログボックスを開きます。新しい数字または式を入力したら、[OK]ボタンをクリックするか'Enter'キーを押します。ダイアログボックスは閉じて新しい値をレジスタに書き込みます。

4.5.5 現在の表示内容をセーブする

現在ウィンドウに表示している内容をテキストファイルにセーブすることが出来ます。ポップアップメニューから[ファイルに保存]を選択してください。

4.6 メモリ内容の日本語表示

4.6.1 メモリ内容を UNICODE 形式で表示する

メモリ内容をUNICODE形式の日本語で表示する場合は[メモリ]ウィンドウを使用します。[メモリ]ウィンドウにUNICODE形式で表示するには、ポップアップメニューから[表示形式]を選択して[表示形式]ダイアログボックス(図 4-7参照)上で[データサイズ]に[Word]、[表示形式]に[Unicode character]を指定します。

4.6.2 メモリ内容を SJIS 形式または EUC 形式で表示する

メモリ内容をSJIS形式またはEUC形式の日本語で表示および変更する場合は、[日本語メモリダンプ]ウィンドウを使用します。

(1) 日本語メモリダンプウィンドウを開く

[日本語メモリダンプ]ウィンドウを開くには[表示->CPU->日本語メモリダンプ…]を選択するか、[日本語メモリダンプ]ツールバーボタンをクリックして[日本語メモリダンプ設定]ダイアログボックスを開きます。

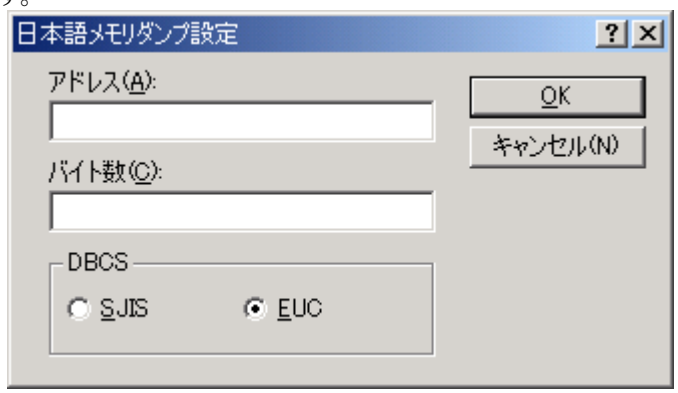


図 4-24 日本語メモリダンプ設定ダイアログボックス

本ダイアログでは下記項目を指定します。

[アドレス] データのメモリ開始アドレスを指定します。（16 進表示）

[バイト数]	データのバイト数を指定します。	
[DBCS]	データ形式を指定します。	
	[SJIS]	シフト JIS 形式で表示します。
	[EUC]	EUC 形式で表示します。

[日本語メモリダンプ設定]ダイアログボックスに設定後、[OK]ボタンをクリックすると[日本語メモリダンプ]ウィンドウが開きます。

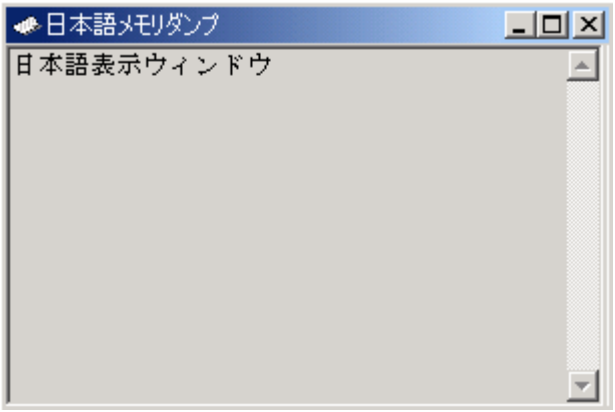


図 4-25 日本語メモリダンプウィンドウ

(2) 日本語データを編集する

ポップアップメニューから[編集...]を選択すると、[文字列編集]ダイアログボックスが開きます。メモリ内容の日本語データを編集できます。

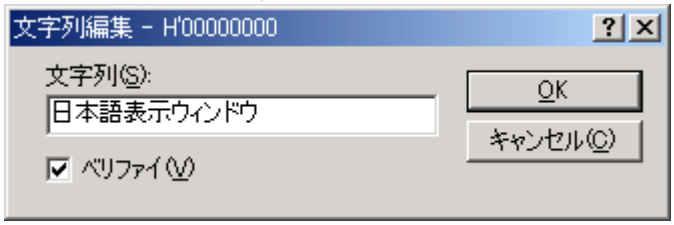


図 4-26 文字列編集ダイアログボックス

変更したい日本語データを入力して[OK]ボタンをクリックします。

(3) 日本語データを検索する

ポップアップメニューから[検索...]を選択すると、[検索]ダイアログボックスを開きます。日本語文字列を検索できます。

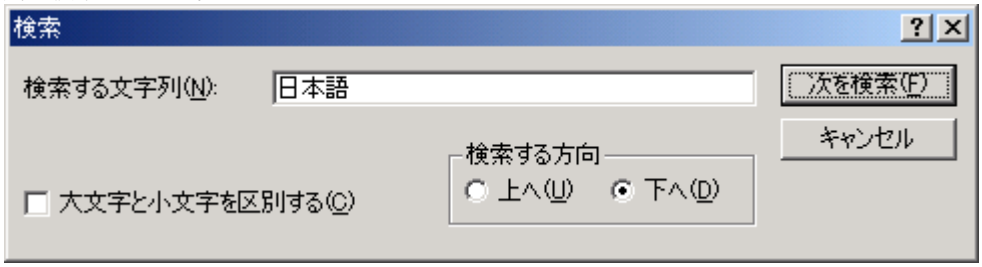


図 4-27 検索ダイアログボックス

検索したい日本語文字列をエディットボックスに入力し、[次を検索]ボタンまたは、'Enter'キーを押すと、ウィンドウ内で指定した文字列を検索します。一致した場合は強調表示します。
検索条件として文字列のほかに大文字／小文字の区別、および検索方向を指定できます。

(4) 表示アドレスを変更する

ポップアップメニューから[アドレス設定...]を選択すると、[アドレス設定]ダイアログボックスを開きます。ウィンドウに表示するメモリアドレスを変更できます。

(5) 表示バイト数を変更する

ポップアップメニューから[バイト数設定...]を選択すると、[バイト数設定]ダイアログボックスを開きます。ウィンドウに表示するバイト数を変更できます。

(6) 表示データ形式を変更する

ポップアップメニューから[EUC]または[SJIS]をチェックすると、ウィンドウに表示するデータ形式を変更できます。

4.7 ラベルを見る

デバッグ情報を含んだユーザプログラムをロードした時に、ラベルを同時に登録します。ラベルを追加することも可能です。
[逆アセンブリ]ウィンドウにおいては、対応するアドレスの代わりとして、また命令オペランドの一部として、ラベルの最初の8文字を表示します。

【注】 オペランドとラベルの値が一致すれば、命令のオペランドはラベル名に置き換わります。同じ値を持つラベルが 2 つ以上ある場合、アルファベット順で先に来るラベルを表示します。

ダイアログボックスで、アドレスまたは値を入力できる場合には、ラベルを使用することも出来ます。

4.7.1 ラベルを一覧にする

現在のデバッガセッションに定義したラベルのすべてを見るには、[表示->シンボル->ラベル]を選択するか、[ラベルの表示]ツールバーボタン をクリックします。

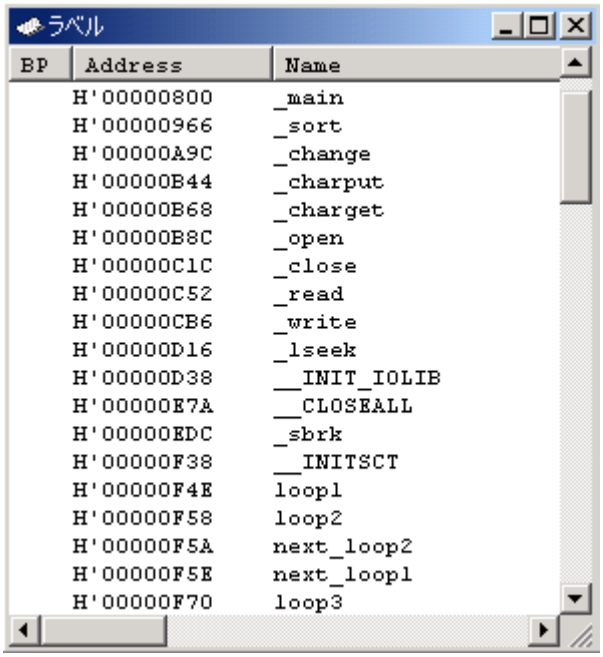


図 4-28 ラベルウィンドウ

それぞれのカラムのヘッダをクリックすることによりアルファベット順（ASCIIコードによって）またはアドレス値でソートしたシンボルを表示させることができます。

[BP]カラムをダブルクリックすることにより関数の入り口でソフトウェアブレークポイントをすばやく設定したり解除したりすることができます。

4.7.2 ラベルを追加する

ラベルを追加するには、ポップアップメニューから[追加...]を選択して、[ラベル追加]ダイアログボックスを表示します。

図 4-29 ラベル追加ダイアログボックス

新しいラベル名を[名前]フィールドに入力し、対応する値を[アドレス]フィールドに入力して[OK]

ボタンを押します。[ラベル追加]ダイアログボックスがクローズし、ラベルリストに新しいラベルを追加、更新します。多重定義関数やクラス名を入力したときは、[関数選択]ダイアログボックスが開くので、関数を選択して[アドレス]フィールドを設定します。詳細は「4.11.3　複数ラベルをサポートする」を参照してください。

4.7.3　ラベルを編集する

ラベルを編集するにはポップアップメニューから[編集...]を選択します。[ラベルの編集]ダイアログボックスを表示します。

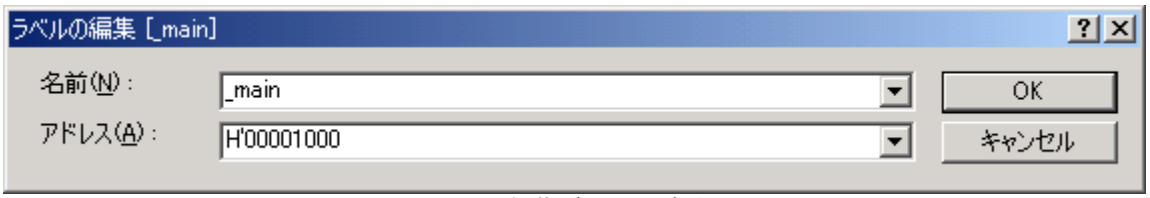


図 4-30　**ラベルの編集ダイアログボックス**

ラベル名と対応する値を編集して、[OK]ボタンを押すとラベルリストに編集を反映し、保存します。多重定義関数やクラス名を入力したときは、[関数選択]ダイアログボックスが開くので、関数を選択して[アドレス]フィールドを設定します。詳細は「4.11.3　複数ラベルをサポートする」を参照してください。

4.7.4　ラベルを削除する

削除したいラベルを選択した状態で、ポップアップメニューから[削除]を選択します。この際、図4-31に示す確認メッセージボックスを表示します。

図 4-31　**ラベル削除確認メッセージボックス**

[OK]ボタンを押すとラベルリストから削除し、ウィンドウを更新します。メッセージボックスの表示が不要のときは、 [今後表示しない]チェックボックスをチェックしてください。

4.7.5　すべてのラベルを削除する

ポップアップメニューから[すべてを削除]を選択すると、リストからすべてのラベルを削除します。この際、図 4-32に示す確認メッセージボックスを表示します。

図 4-32 全ラベル削除確認メッセージボックス

[OK]ボタンを押すと、すべてのラベルを HEW のシンボルテーブルから削除し、リスト表示もクリアします。メッセージボックスの表示が不要のときは、 [今後表示しない]チェックボックスをチェックしてください。

4.7.6 ラベルをファイルからロードする

シンボルファイルをロードして現在の HEW のシンボルテーブルに結合できます。ポップアップメニューから[ロード...]を選択すると、[ファイルを開く]ダイアログボックスをオープンします。

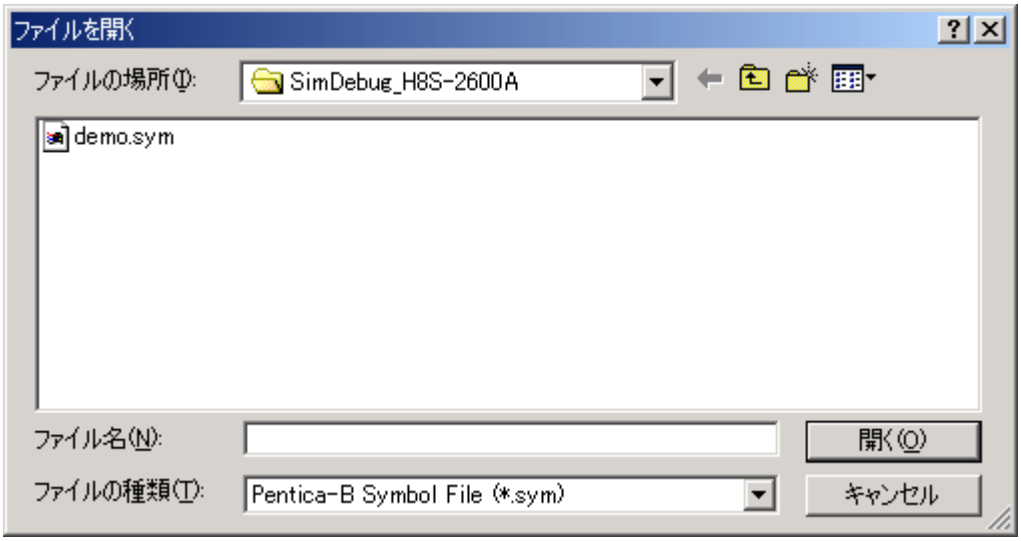


図 4-33 ファイルを開くダイアログボックス

ダイアログボックスは、Windows®標準の[ファイルを開く]ダイアログボックスと同様です。ファイルを選択し、[開く]ボタンを押すとロードを開始します。シンボルファイルの標準拡張子は".sym"です。

4.7.7　ラベルをファイルに保存する

ポップアップメニューから[名前を付けて保存...]を選択すると、[名前を付けて保存]ダイアログボックスを表示します。[名前を付けて保存]ダイアログボックスは Windows®標準の[名前を付けて保存]ダイアログボックスと同様に操作できます。[ファイル名]フィールドにファイル名を入力し[保存]ボタンを押すとシンボルファイルにラベルリストを保存します。標準ファイル拡張子は".sym"です。
フォーマットが「付録 E　シンボルファイルフォーマット」にあるので参照してください。

一度[名前を付けて保存...]メニューでファイルに保存すると、以後はポップアップメニューの[上書き保存]で、現在のシンボルテーブルを同一シンボルファイルに保存できます。

4.7.8　ラベルを検索する

ポップアップメニューから[検索...]を選択すると、[ラベルの検索]ダイアログボックスを表示します。

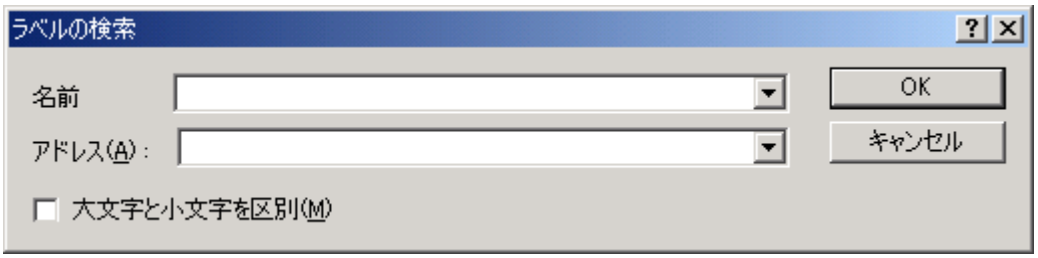


図 4-34　ラベルの検索ダイアログボックス

検索したいラベル名の一部、全部をエディットボックスに入力し、[OK]ボタンまたは、'Enter'キーを押すと、指定した文字列を含んでいるテキストファイルをサーチします。

【注】　ラベルは、はじめの 1024 文字分しか情報を保持していません。したがって、ラベル名のはじめの 1024 文字分は重複しないようにしてください。ラベルは、大文字、小文字を区別します。

4.7.9　次を検索する

ラベルが検索できた後、ポップアップメニューから[次を検索]を選択すると、検索条件に一致する次のラベルを検索します。

4.7.10　ラベルに対応するソースプログラムを表示する

ラベルを選択した状態で、ポップアップメニューから[ソースを表示]を選択すると、ラベルのアドレスに対応する[ソース]をオープンします。

4.8 レジスタ内容を見る

アセンブリ言語レベルでデバッグを行う場合に、CPUの汎用レジスタの内容を簡単に見ることができます。これは、[レジスタ]ウィンドウを使用して行います。

4.8.1 レジスタウィンドウを開く

[レジスタ]ウィンドウを開くには、[表示->CPU->レジスタ]を選択するか、[レジスタ]ツールバーボタン  をクリックします。[レジスタ]ウィンドウが開きCPU汎用レジスタおよびその値(16進数)を表示します。

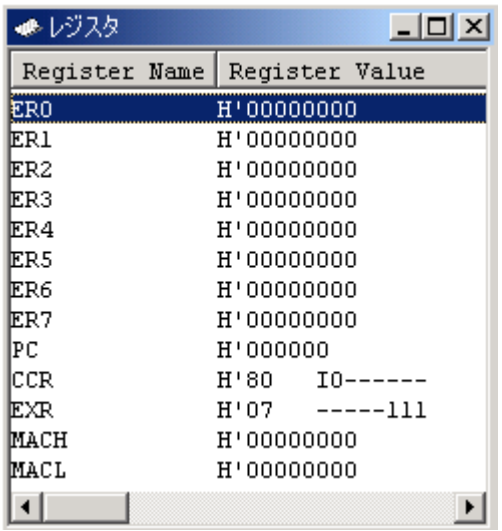


図 4-35 **レジスタウィンドウ**

4.8.2 ビットレジスタを拡張する

ビットレベルのフラグで制御するレジスタの場合、数値を表示するだけでなく、記号でもビットの状態を表示します。また、そのレジスタをダブルクリックするとレジスタ変更ダイアログを表示し、各ビット毎にオン／オフを設定できます。各ビットのチェックボックスをチェックしたときは１を、クリアしたときは０を設定します。

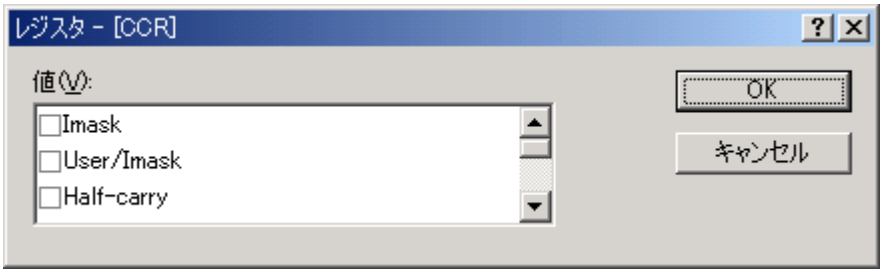


図 4-36 **ビットレジスタの拡張**

4.8.3 表示するレジスタを選択する

[レジスタ]ウィンドウに表示するレジスタを選択する場合は、ポップアップメニューから[設定...]を選択します。このダイアログボックスを図 4-37に示します。

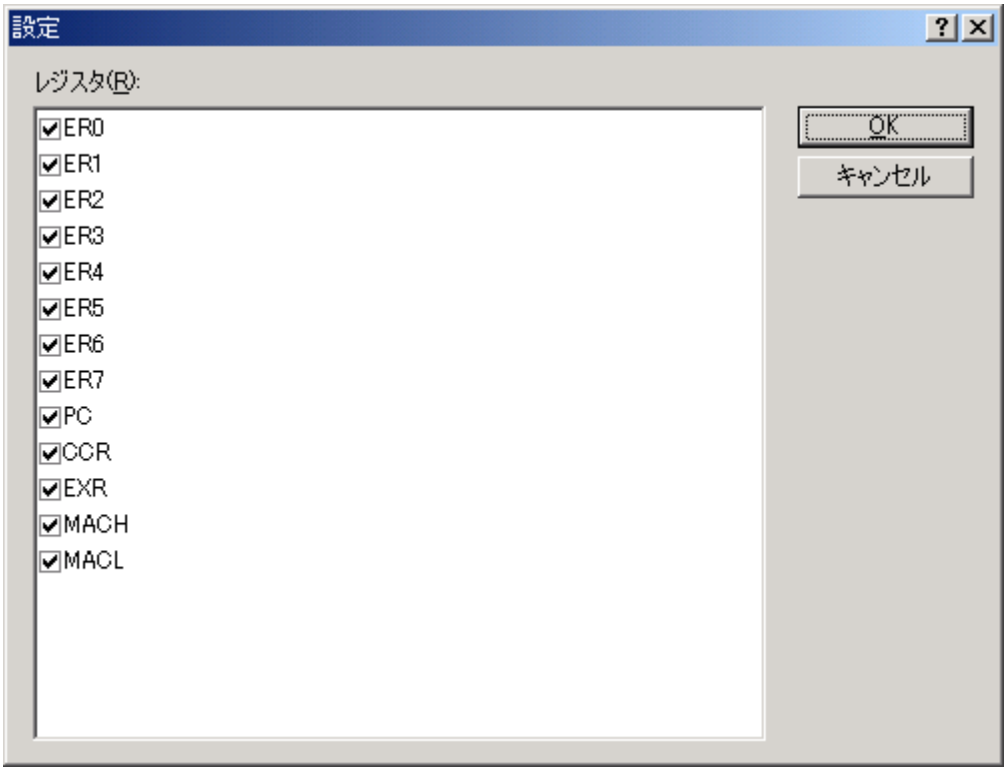


図 4-37 設定ダイアログボックス

4.8.4 ウィンドウを分割表示する

[レジスタ]ウィンドウを上下 2 分割で表示したいときは、ポップアップメニューから[分割]を選択後分割バーを移動して分割します。

4.8.5 レジスタの内容を修正する

レジスタの内容を修正するには、以下のいずれかの方法でレジスタ編集ダイアログボックスを開きます。

- ウィンドウに対して値を直接入力する。
- 修正したいレジスタをダブルクリックする。
- 修正したいレジスタを選択して、ポップアップメニューの[編集...]を選択する。

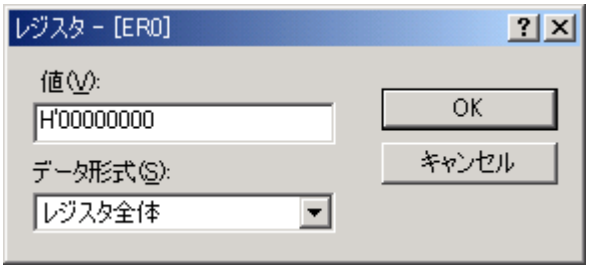


図 4-38 レジスタダイアログボックス

[値]フィールドには数字またはC/C++の式を入力することができます。ドロップダウンリストからオプションを選択して、レジスタの内容のすべて、マスクした領域、フローティングビットまたはフラグビットを修正することができます(このリスト内容は、CPUのモデルおよび選択したレジスタによって異なります)。

新しい数字および式を入力したら[OK]ボタンをクリックするか'Enter'キーを押します。ダイアログボックスは閉じて、新しい値をレジスタに書き込みます。

4.8.6 レジスタの内容を使用する

[逆アセンブリ]または[メモリ]ウィンドウのアドレス指定など、HEWの別のところで値を入力する場合、CPUレジスタの中にある値を使用するためには、"#R1"、"#PC"、"#R6L"、または"#ER3"などのように、レジスタ名の先頭に " # " 記号をつけてください。

4.8.7 現在の表示内容をセーブする

現在ウィンドウに表示している内容をテキストファイルにセーブすることが出来ます。ポップアップメニューから[ファイルに保存...]を選択してください。

4.9 プログラムを実行する

この節では、作成したプログラムコードの実行方法について説明します。ここでは、プログラムを連続して実行させたり、シングルステップ実行を行ったり、同時に複数の命令を実行させたりします。

4.9.1 リセットから実行を開始する

ターゲットマイコンをリセットしてリセットベクタアドレスプログラムを実行させるには、[デバッグ->リセット後実行]を選択するか、[リセット後実行]ツールバーボタン  をクリックします。

プログラムは、ブレークポイントにヒットするまで、またはブレーク条件が成立するまで実行を続けます。プログラムの実行は手動で停止することができます。その方法としては、[デバッグ->プログラムの停止]を選択するか[停止]ツールバーボタンをクリックします。

【注】 プログラムはリセットベクタ位置に格納したアドレスから実行を開始します。したがって、この位置に自分のスタートアップコードのアドレスを含んでいることを確認することが重要です。

4.9.2 実行を継続する

作成したプログラムが停止すると、HEW は CPU の現在のプログラムカウンタ（PC）アドレス値に対応する[エディタ]および[逆アセンブリ]ウィンドウの行の左余白に黄色の矢印を表示します。ステップ実行を行った場合、または実行を続けた場合、この命令を次に実行します。

現在の PC アドレスから実行を継続するには、[実行]ツールバーボタン  をクリックするか、[デバッグ->実行]を選択します。

前回停止時と異なるアドレスから実行を継続するには、下記の方法で PC を変更後に[実行]ツールバーボタン  をクリックするか、[デバッグ->実行]を選択します。

- [レジスタ]ウィンドウ上で変更する。詳しくは「4.8.5 レジスタの内容を修正する」を参照してください。
- [エディタ]ウィンドウまたは[逆アセンブリ]ウィンドウ上で、テキストカーソル(マウスカーソルではありません)を変更したい行に移動してポップアップメニューから[PC 値をカーソル位置に設定]を選択する。

4.9.3 カーソルまで実行する

アプリケーションを実行している途中で、シングルステップ実行を複数回行うだけの比較的小さいセクションコードのみを実行したいと考える場合があります。これは、[カーソル位置まで実行]機能を使用して行うことができます。

➲ [カーソル位置まで実行]を使用するには

1. [ソース]または[逆アセンブリ]ウィンドウが開いていて、プログラムを停止するアドレスを表示していることを確認します。
2. アドレスフィールドをクリックするかカーソルキーを使用してプログラムを停止するアドレス上にテキストカーソルを置きます。
3. ポップアップメニューから[カーソル位置まで実行]を選択します。

デバッギングプラットフォームは作成したコードを現在のPC値から実行し、カーソル位置が示すアドレスまで実行します。

【注】

1. **作成したプログラムがこのアドレスのコードを決して実行しない場合、プログラムは停止しません。その場合、コードの実行を中止するには、”Esc”キーを押すか、[デバッグ ->プログラムの停止]を選択するか、[停止]ツールバーボタン  をクリックします。**
2. **カーソル位置まで実行機能は、PC ブレークポイント機能を利用しています。そのため、すでに PC ブレークポイントが最大数設定してある場合は、本機能は使用できません。**

4.9.4 開始アドレスを指定して実行する

[プログラム実行]ダイアログボックスを利用すると、任意のアドレスから命令を実行することができます。

[プログラム実行]ダイアログボックスを開くには、[デバッグ->ラン…]を選択します。

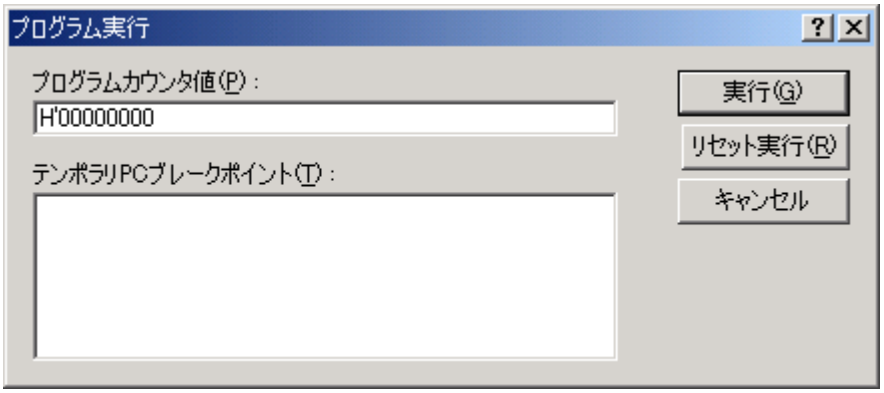


図 4-39 プログラム実行ダイアログボックス

本ダイアログボックスでは命令実行の条件を設定します。

[プログラムカウンタ値] 実行を開始する命令アドレスを設定します。
初期値は現在の PC 値となります。

[テンポラリ PC ブレークポイント] 一時的な PC ブレークポイントを設定します。
本ダイアログボックスによる命令実行が停止するとこのブレークポイントは解除されます。

【注】[テンポラリ PC ブレークポイント]は PC ブレークポイント機能を利用しています。既に PC ブレークポイントを最大数利用している場合は本機能を利用できません。

[実行]ボタンをクリックすると、設定した内容に従って命令を実行します。
[リセット実行]ボタンをクリックすると、リセットベクタから命令を実行します。
[キャンセル]ボタンをクリックすると、命令を実行しないで、本ダイアログボックスを閉じます。

4.9.5 シングルステップ

作成したコードをデバッグするために、一度に一行だけまたは一つの命令だけステップ実行して、この命令がシステムにどのように影響するかを確認したい場合、[エディタ]ウィンドウでは、ソースライン一行だけをステップ実行します。[逆アセンブリ]ウィンドウにおいては、アセンブリ言語命令単位にステップ実行します。命令が他の関数またはサブルーチンをコールした場合、オプションでその関数にステップインまたはステップオーバーすることができます。その命令コールを行わない場合には、いずれのオプションでも、デバッガに命令を実行させ、次の命令で停止させることができます。

(1) 関数にステップイン実行する

関数にステップイン実行することを選択した場合にはデバッガは関数の行または命令でコールを実行します。関数にステップインするには、[ステップイン]ツールバーボタン  をクリックするか、[デバッグ->ステップイン]を選択します。

(2) 関数コールをステップオーバー実行する

関数をステップオーバー実行することを選択した場合には、デバッガはコールおよび関数内のすべてのコード(および関数が行う可能性のある関数コールのすべて)を実行して、呼び出し元の関数の次の行または命令で停止します。関数をステップオーバーするには、[ステップオーバ]ツールバーボタン  をクリックするか、[デバッグ ->ステップオーバ]を選択します。

(3) 関数からステップアウト実行する

関数内の確認したい命令の実行が終了した場合や、誤って関数にステップインした場合に、ステップアウト機能を使用すると関数内の残りのコードをステップ実行せずに呼び出し元の関数に戻ることができます。

現在の関数からステップアウトするには、[ステップアウト]ツールバーボタン  をクリックするか、[デバッグ ->ステップアウト]を選択します。

4.9.6 複数のステップ

[プログラムステップ]ダイアログボックスを使用することにより、一度に複数のステップ実行ができます。このダイアログボックスでは、ステップ間の時間差を選択し、ステップ実行を自動的に行うよう設定できます。このダイアログボックスは、[デバッグ -> ステップ...]を選択して開きます。

[プログラムステップ]ダイアログボックスを表示します。

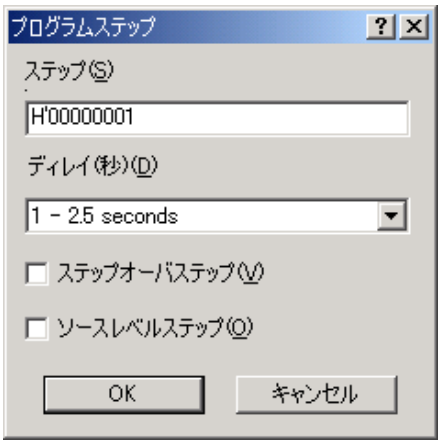


図 4-40 プログラムステップダイアログボックス

[ステップ]	連続実行するステップ数を設定します。
[ディレイ(秒)]	コードを自動的にステップ実行するときのステップ間の遅延を選択します。 0～３秒まで、0.5 秒単位で選択できます。
[ステップオーバステップ]	チェックすると関数呼出しをステップオーバーします。
[ソースレベルステップ]	チェックするとソースレベルでステップします。

[OK]ボタンをクリックするか、'Enter'キーを押してステップ実行を開始します。

4.9.7 複数ターゲットの実行

複数ターゲットを同時に実行してデバッグすることができます。詳細は「4.24 複数デバッギングプラットフォームを同期動作させる」を参照してください。

4.10 プログラムを停止する

この節では、作成したプログラムの実行を停止する方法を説明します。停止手段として、[停止]ボタンを使用して停止する方法、および作成したコードの特定の場所にブレークポイントを設定することによって停止する方法について説明します。

4.10.1 [停止]ツールバーボタンによる停止

作成したプログラムが実行中の場合、[停止]ツールバーボタン  (赤い停止の印)を使用することができます。しかし、プログラムが停止している場合は、使用できません  (STOPの印が灰色になります)。[停止]ツールバーボタンをクリックするか、[デバッグ->プログラムの停止]を選択することによりプログラムが停止します。

[停止]によりプログラムが停止したとき、[アウトプット]ウィンドウの[Debug]タブに”Stop”というメッセージを表示します。

4.10.2 標準のブレークポイント(PC ブレークポイント)

作成したプログラムをデバッグする場合､PCブレークポイントにより指定した行または命令でプログラムの実行を停止させることができます｡標準的なPCブレークポイントを設定したり解除したりする方法を以下に示します。より複雑な設定をする場合には、[イベントポイント]ウィンドウを使用します。[イベントポイント]ウィンドウは  ボタンをクリックするか、
[表示->コード->ブレークポイント]を選択することにより表示します｡詳細は､「4.16 シミュレータ・デバッガのブレークポイントを使用する」を参照してください。

(1) [エディタ]ウィンドウ上で PC ブレークポイントを設定する

1. PCブレークポイントを設定する位置の[逆アセンブリ]または[エディタ]ウィンドウが開いていることを確認します。
2. プログラムを停止したい行でポップアップメニューの[ブレークポイントの挿入/削除] を選択するか、”F9”キーを押すかします。
3. 左余白に赤丸を表示します。これは、PC ブレークポイントを設定したことを示します。
4. ポップアップメニューの[ブレークポイントの有効化/無効化]を選択すると、現在設定しているブレークポイントの有効／無効の切り替えができます。

作成したプログラムを実行して PCブレークポイントを設定したアドレスに達すると､[アウトプット]ウィンドウの[Debug]タブに" PC Breakpoint"というメッセージを表示し、実行を停止し、[ソース]または[逆アセンブリ]ウィンドウを更新し、停止位置を左余白に矢印で表示します。

【注】 ブレーク発生時には、PC ブレークポイントを設定した行または命令を実行する直前で停止します。その PC ブレークポイントで停止した後に Go または Step を選択した場合、矢印で表示した行から実行します。また、複数ターゲットデバッグ時には一方のみ停止させるか両方停止させるかを指定することができます。詳細は「4.24 複数デバッギングプラットフォームを同期動作させる」を参照してください。

(2) [ブレークポイント]ダイアログボックスを使用して PC ブレークポイントを設定する

PCブレークポイントの設定を編集するには、[編集 -> ソースブレークポイント…]メニューを選択し、[ブレークポイント]ダイアログボックス図 4-41を表示します。

・このダイアログボックスに現在設定しているPCブレークポイントを表示します。
・[コードの編集]ボタンによりPCブレークポイントが存在するソースを見ることができます。
・[削除]、[すべて削除]ボタンにより、PCプレークポイント１つまたは全てを削除できます。
・各ブレークポイントのチェックボックスにより有効／無効の切り替えができます。

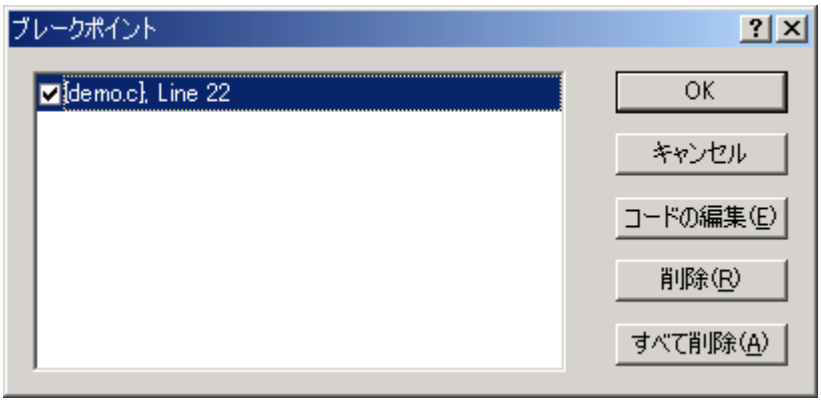


図 4-41 ブレークポイントダイアログボックス

(3) PC ブレークポイントを切り替える

PCブレークポイントを設定している行の[BP]カラムをダブルクリックするか、その行にカーソルを置いて”F9”キーを使用すると、[PCブレークポイント]の設定を切り替えることができます。切り替わる設定内容はデバッグプラットフォームによって異なります。

4.11 Elf/Dwarf2 のサポート

HEWは、C/C++およびアセンブリ言語で書いたアプリケーションのデバッグのためにElf/Dwarf2オブジェクトファイルフォーマットをサポートします。

主な特長

- ソースレベルデバッグ
- C/C++演算子
- C/C++式(キャスト、ポインタ、参照)
- あいまいな関数名
- オーバーレイメモリロード
- ウオッチ-ローカル、およびユーザ定義
- スタックトレース

4.11.1 C/C++演算子

以下のC/C++ 言語演算子を使用することができます。

```
+, -, *, /, &, |, ^, ~, !, >>, <<, %, (, ), <, >, <=, >=, ==, !=, &&, ||
Buffer_start + 0x1000
#R1 | B'10001101
```

```
((pointer + (2 * increment_size)) & H'FFFF0000) >> D'15
!(flag ^ #ER4)
```

4.11.2 C/C++の式

式の例

```
Object.value                        //メンバの直接参照を指定します(C/C++)
p_Object->value                     //メンバの間接参照を指定します(C/C++)
Class::value                        //クラスを持つメンバの参照を指定します(C++)
*value                              //ポインタを指定します(C/C++)
&value                              //参照を指定します(C/C++)
array[0]                            //アレイを指定します(C/C++)
Object.*value                       //ポインタを持つメンバの参照を指定します(C++)
::g_value                           //グローバル変数の参照を指定します(C/C++)
Class::function(short)              //メンバ関数を指定します(C++)
(struct STR) *value                 //キャスト動作を指定します(C/C++)
```

4.11.3 複数ラベルをサポートする

プログラム言語の中の、例えばC++オーバーロード関数などでは、1つのラベルが複数のアドレスを表す場合があります。各ダイアログボックスでこのようなラベル名を入力した場合、HEWは[関数選択]ダイアログボックスを使用してオーバーロード関数およびメンバ関数を表示します。

図 4-42 関数選択ダイアログボックス

[関数選択]ダイアログボックスでは、オーバーロード関数またはメンバ関数を選択します。通常、一度に一つの関数を選択します。ただし、ブレークポイントを設定する場合においてのみ、複数の関数を選択することができます。このダイアログボックスには3つの領域があります。

[関数名の選択] 同じ名前をもつ関数またはメンバ関数、およびその詳細情報を表示します。
[関数名の設定] 設定する関数およびそれらの詳細情報を表示します。
[カウンタ] [全関数] 同じ名前をもつ関数かたはメンバ関数を表示します。
[選択関数] [関数名の選択] リストボックスに表示する関数の数を表示します。
[指定関数] [関数名の設定]リストボックスに表示する関数の数を表示します。

(1) 関数を選択する

[関数名の選択]リストボックスから選択したい関数をクリックして、[>]ボタンをクリックします。選択した関数を[関数名の設定]リストボックスに表示します。[関数名の選択]リストの関数すべてを選択するには、[≫]ボタンをクリックします。

(2) 関数の選択を解除する

[関数名の設定]リストボックスから選択を解除する関数をクリックして、[<]ボタンをクリックします。すべての関数の選択を解除するには、[≪]ボタンをクリックします。選択を解除した関数は、[関数名の設定]リストボックスから[関数名の選択]リストボックスへ戻します。

(3) 関数を設定する

[OK]ボタンをクリックして、[関数名の設定]リストボックスに表示した関数を設定します。関数を設定し、[関数選択]ダイアログボックスを閉じます。

[キャンセル]ボタンをクリックすると、関数を設定せずにダイアログボックスを閉じます。

4.11.4 オーバレイプログラムのデバッグ

この章では、オーバレイ関数を使用するための設定について説明します。

(1) セクショングループを表示する

オーバレイ（いくつかのセクショングループを同じアドレス範囲に割り当てる）を使用すると、アドレス範囲およびセクショングループを[オーバレイ]ダイアログボックスに表示します。
[メモリ->オーバレイの構成]を選択して[オーバレイ]ダイアログボックスを開きます。

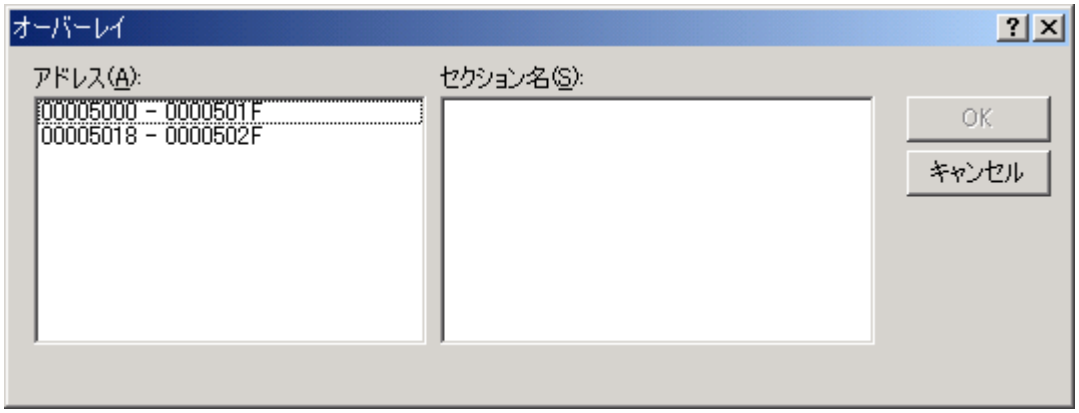


図 4-43 オーバレイダイアログボックス (開いたとき)

このダイアログボックスには2つの領域があります。[アドレス]リストボックスおよび[セクション名]リストボックスです。

[アドレス]リストボックスは、オーバレイプログラムが使用するアドレス範囲を表示します。アドレス範囲の１つをクリックして[アドレス]リストボックスのアドレス範囲を選択します。

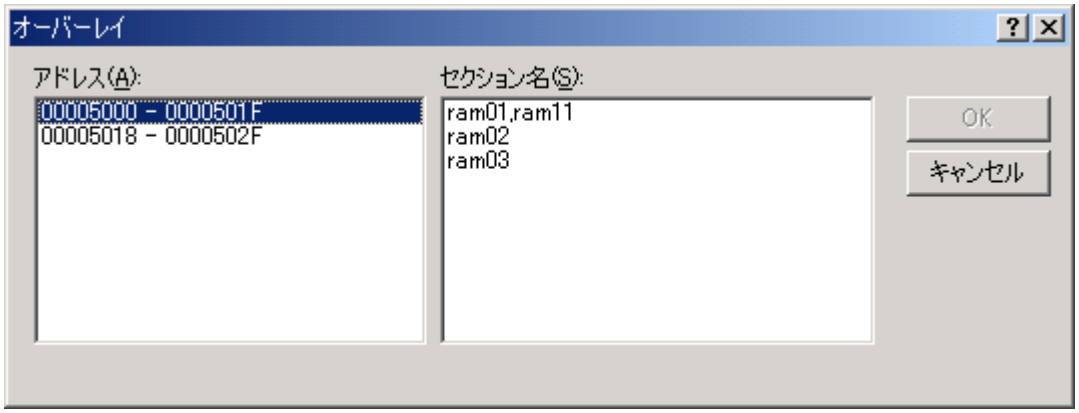


図 4-44 オーバレイダイアログボックス (アドレス範囲を選択)

[セクション名]リストボックスは、選択したアドレス範囲に割り当てたセクショングループを表示します。

➲セクショングループを設定するには
　オーバレイ関数を使用するときは、最も優先度の高いセクショングループを[オーバレイ]ダイアログボックスで選択していなければ、HEWは正しく動作しません。
　まず[アドレス]リストボックスに表示したアドレス範囲の一つをクリックします。選択したアドレス範囲に割り当てたセクショングループを[セクション名]リストボックスに表示します。
　表示しているセクショングループの中から最も優先度の高いセクショングループをクリックします。

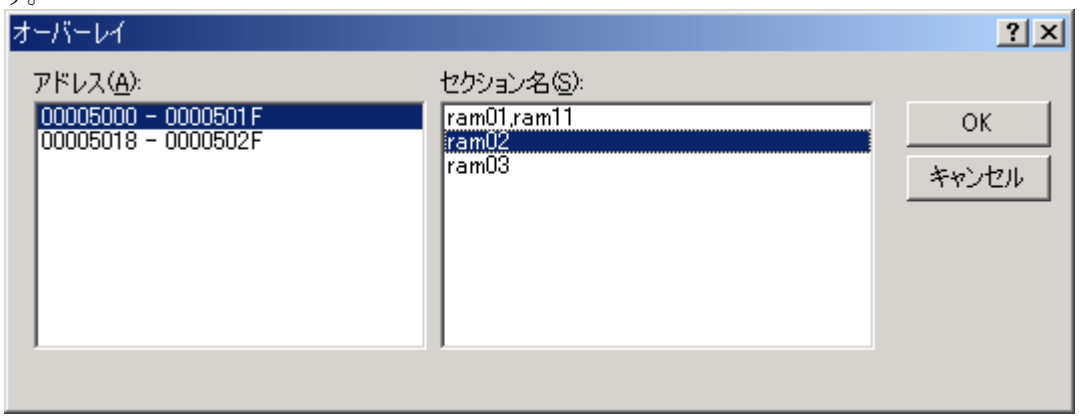


図 4-45 オーバレイダイアログボックス (最も優先度の高いセクショングループ選択時)

　セクショングループを選択したら、[OK]ボタンをクリックして優先度の設定を保存して、ダイアログボックスを閉じます。[キャンセル]ボタンをクリックすると、優先度の設定を保存せずにダイアログボックスを閉じます。

【注】 オーバレイ関数が使用するアドレス範囲内では[オーバレイ]ダイアログボックスに指定したセクションのデバッグ情報を参照します。したがって、現在ロードしているプログラムと同じセクションを[オーバレイ]ダイアログボックスで選択しなければなりません。

4.12 変数の表示

　本節では、ソースプログラム上の変数の値を表示する方法について説明します。

4.12.1 ツールチップウォッチ

　作成したプログラムの変数を最もすばやく見るには、ツールチップウォッチ 機能を使用します。

➲ツールチップウォッチ を使用するには
　確認したい変数を含むソースファイルを[エディタ]ウィンドウに表示します。
　確認したい変数名の上にマウスのカーソルを静止させます。変数の近くにツールチップを表示し、その変数の値を表示します。

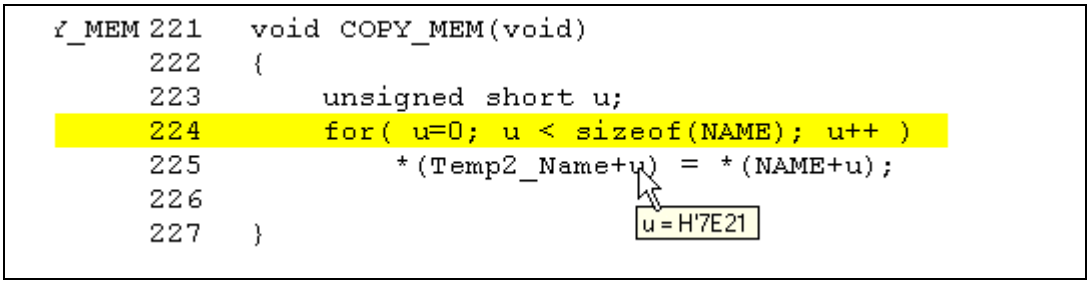


図 4-46 **ツールチップウォッチ**

4.12.2 インスタントウォッチ

確認したい変数を含むソースファイルを[エディタ]ウィンドウに表示します。

確認したい変数名の上にマウスのカーソルを置いてポップアップメニューから[インスタントウォッチ...]を選択します。

[インスタントウォッチ]ダイアログボックスが開き、カーソル上の変数を表示します。

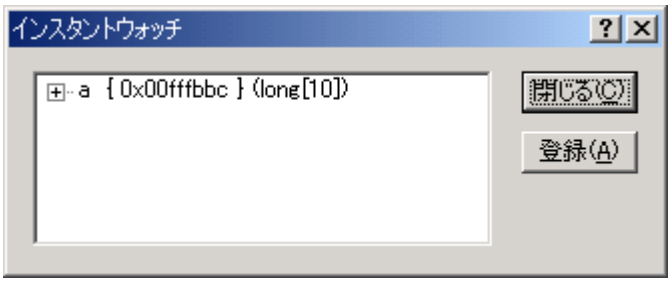


図 4-47 **インスタントウォッチダイアログボックス**

変数名の左側の'+'記号はクリックすれば情報を拡張表示できることを、'-'記号は情報を縮小表示できることを示します。[登録]ボタンを押すと、変数を[ウォッチ]ウィンドウに登録して、ダイアログボックスを閉じます。[閉じる]ボタンを押すと、変数を[ウォッチ]ウィンドウに登録しないで、ダイアログボックスを閉じます。

4.12.3 ウォッチウィンドウ

[ウォッチ]ウィンドウを開くことにより、任意の変数について値を参照することができます。

(1) ウォッチウィンドウを開く

[ウォッチ]ウィンドウを開くには、[表示->シンボル->ウォッチ]を選択するか、[ウォッチ]ツールバーボタン  が使用可能であれば、クリックします。[ウォッチ]ウィンドウが開きます。

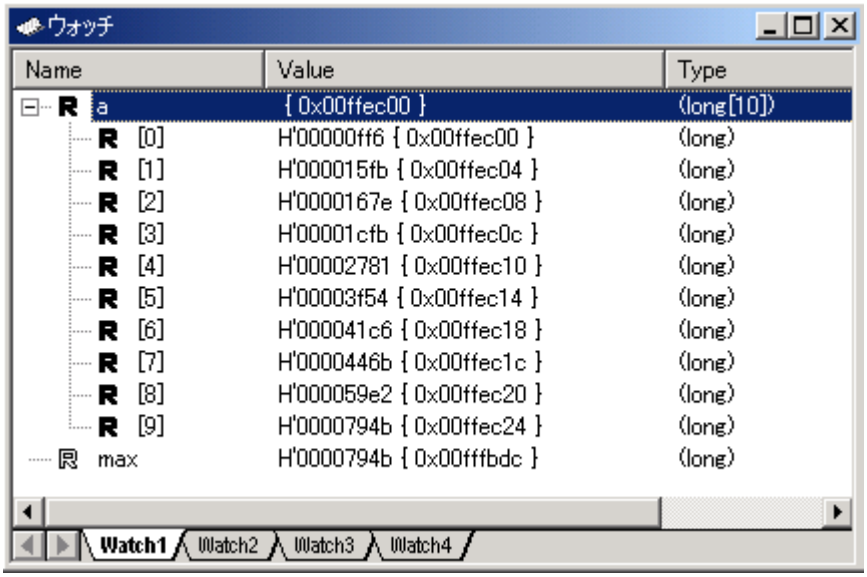


図 4-48 ウォッチウィンドウ

C/C++ソースレベルの変数を表示・変更することができるウィンドウです。本ウィンドウの内容は、アブソリュートファイル(*.abs)内のデバッグ情報から、C/C++ソースプログラムの情報がある場合にのみ表示します。コンパイラ等の最適化により、ソースプログラムの情報としてデバッグ情報にない場合は表示できません。また、マクロ宣言されたものについても表示できません。

表示する項目は以下の通りです。

[Name]	変数名を表示します
[Value]	変数の値、割付け位置を表示します 割付け位置は{}で囲んで表示します
[Type]	変数の型を表示します

Rマークはその変数がリアルタイムに更新できることを示します。Rマークが太字のとき、その変数の値をプログラムの実行時に[Response]コマンドで指定した命令間隔ごとに更新します。

(2) Watch アイテムを追加する

Watchアイテムを[ウォッチ]ウィンドウに追加するには、[ウォッチ]ウィンドウの[シンボル登録]ダイアログボックスを使用ます。

➲ [ウォッチ]ウィンドウから[シンボル登録]を使用するには
[ウォッチ]ウィンドウを開きます。
ポップアップメニューから[シンボル登録]を選択します。
[シンボル登録]ダイアログボックスが開きます。

図 4-49 シンボル登録ダイアログボックス

見たい変数名を入力して、[OK]ボタンをクリックします。その変数を[ウォッチ]ウィンドウに追加します。

また、[エディタ]ウィンドウ上のソースファイルから[ウォッチ]ウィンドウへ変数をドラッグアンドドロップしても追加できます。

【注】 追加した変数がローカル変数で現在範囲外の場合には、HEW はその変数を[ウォッチ]ウィンドウに追加しますが、値には"Not available now"を表示します。

(3) Watch アイテムを拡張する

Watchアイテムがポインタ、アレイ、または構造体のとき、その名前の左側にプラス記号(+)の拡張指示子を表示します。つまり、Watchアイテムを拡張できるという意味です。Watchアイテムを拡張するには、プラス記号(+)をクリックします。拡張したアイテムは、タブ１つ分インデントをつけて、その要素(構造体またはアレイの場合)またはデータ値(ポインタの場合)を表示し、プラス記号がマイナス記号に変わります。Watchアイテムが要素にポインタ、構造体、またはアレイを含む場合、その横に拡張指示子を表示します。

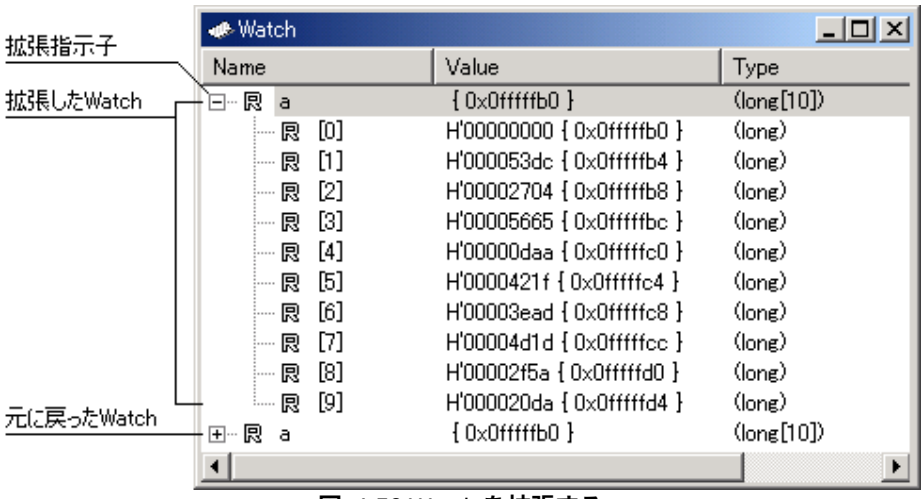


図 4-50 Watch を拡張する

拡張したWatchアイテムを元に戻すには、再び拡張指示子をクリックします。アイテムの要素は、元の単一のアイテムに戻り、マイナス記号はプラス記号に戻ります。

また、Watchアイテムを選択した状態で数字キー（１～９）を押すと、その数字の階層分Watchアイテムを拡張することができます。

(4) Watch アイテムの値を編集する

テストのためや、プログラムにバグがあるために値が正しくないときには、Watch変数の値を変更することができます。Watchアイテムの値を変更するには、値の編集機能を使用します。

➲ Watchアイテムの値を編集するには
ウィンドウに対して値を直接入力します。
または、ポップアップメニューから[値の編集]を選択します。
[値の編集]ダイアログボックスオプションが開きます。

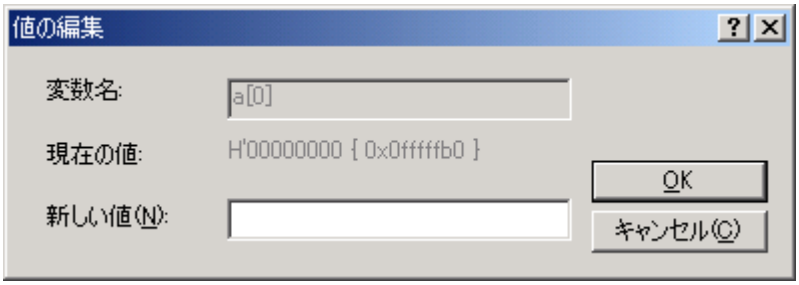


図 4-51 値の編集ダイアログボックス

[新しい値]フィールドに新しい値はまたは式を入力して[OK]ボタンをクリックします。[ウォッチ]ウィンドウを更新し、新しい値を表示します。

(5) Watch アイテムを削除する

特定の Watch アイテムを削除するには、そのアイテムを選択してポップアップメニューから[削除]を選択します。アイテムを削除し、[ウォッチ]ウィンドウを更新します。

すべての Watch アイテムを削除するには、ポップアップメニューから[全シンボル削除]を選択します。すべてのアイテムを削除し、[ウォッチ]ウィンドウを更新します。

(6) リアルタイム更新を設定する

Watch アイテムで変数名の左に表示する”R”マークは、その変数がリアルタイムに更新できることを示します。R マークが太字のとき、その変数の値をプログラムの実行時に従ってリアルタイムに更新します。

リアルタイム更新は[ウォッチ]ウィンドウのポップアップメニューで設定します。

(a) 自動更新有効化

選択している変数の”R”マークが太字になり、リアルタイム更新します。

(b) 全シンボル自動更新有効化

すべての”R”マークが太字になり、リアルタイム更新します。

(c) 自動更新無効化

選択している変数の”R”マークが中抜きになり、リアルタイム更新を解除します。

(d) 全シンボル自動更新無効化

すべての”R”マークが中抜きになり、リアルタイム更新を解除します。

(7) 表示基数を変更する

変数を選択してポップアップメニューから[基数]を選択するとサブメニューで変数の表示基数を変更できます。

(8) 表示内容をファイルに保存する

[ウォッチ]ウィンドウの表示内容をファイルに保存するには、ポップアップメニューの[保存…]を選択します。

[保存…]を選択すると、[名前を付けて保存]ダイアログボックスを表示します。ファイル名を指定し、[ウォッチ]ウィンドウに表示している内容をセーブします。[Append]チェックボックスにチェックすると追加書きこみ、チェックしないと上書きします。

(9) メモリウィンドウを表示する

選択している変数が割り付いているメモリ領域を[メモリ]ウィンドウに表示することができます。ポップアップメニューの[メモリウィンドウへ移動…]を選択すると、[表示形式]ダイアログボックスを開きます。ダイアログの初期値には選択している変数の情報(先頭アドレス、終了アドレス、およびデータサイズ)を設定します。[OK]ボタンをクリックすることで[メモリ]ウィンドウを開きます。

4.12.4 ローカルウィンドウ

[ローカル]ウィンドウを開くことにより、ローカル変数の一覧とそれらの値を参照することができます。

(1) ローカルウィンドウを開く

[ローカル]ウィンドウを開くには、[表示->シンボル->ローカル]を選択するか、[ローカル]ツールバーボタンをクリックします。

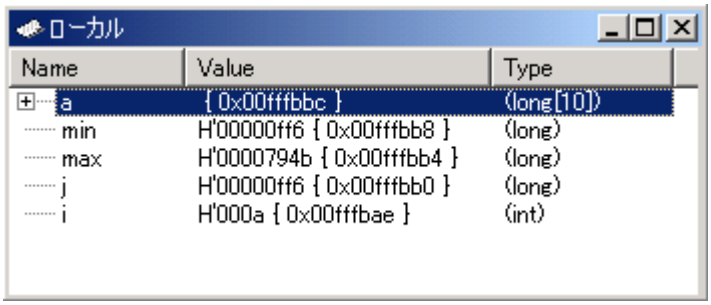


図 4-52 ローカルウィンドウ

プログラムをデバッグしていくに従い、[ローカル]ウィンドウを更新します。ローカル変数を定義した時点で初期化していないと、ローカル変数に値を代入するまで[ローカル]ウィンドウの値は不定値となります。

ローカル変数の値およびローカル変数の表示は、[ウォッチ]ウィンドウと同じ方法で修正することができます。

4.13 プロファイル情報を見る

プロファイル機能は、アプリケーションプログラムの実行パフォーマンスを関数単位に測定します。アプリケーションプログラム中の性能劣化の原因となっている場所および要因を調査することができます。

HEWはプロファイルデータの参照方法、参照目的に応じて、３つのウィンドウでプロファイル測定結果を表示します。

4.13.1 スタック情報ファイル

プロファイル機能は、最適化リンカ(Ver.7.0以降)が出力するスタック情報ファイル（拡張子".SNI"）を読み込むことができます。このファイルには、ソースファイル上の(静的な)関数呼び出し関係の情報が入っています。HEWがスタック情報ファイルを読み込むことで、ユーザアプリケーションが未実行（プロファイルデータの測定を行う前）でも、関数の呼び出し関係を表示できるようになります。(但し、[プロファイル]ウィンドウのポップアップメニューで
[表示設定->未実行関数を表示しない]をチェックしている場合を除きます。)

HEWがスタック情報ファイルを読み込まない場合、プロファイル機能で表示するデータは、プロファイルデータ測定中に実行した関数についてのみになります。

リンカでスタック情報ファイルを生成するには、[Standard Toolchain]ダイアログボックスの[Link/Library]タブで[Category]リストボックスを[Other]に指定し、[Stack information output]チェックボックスをチェックしてください。

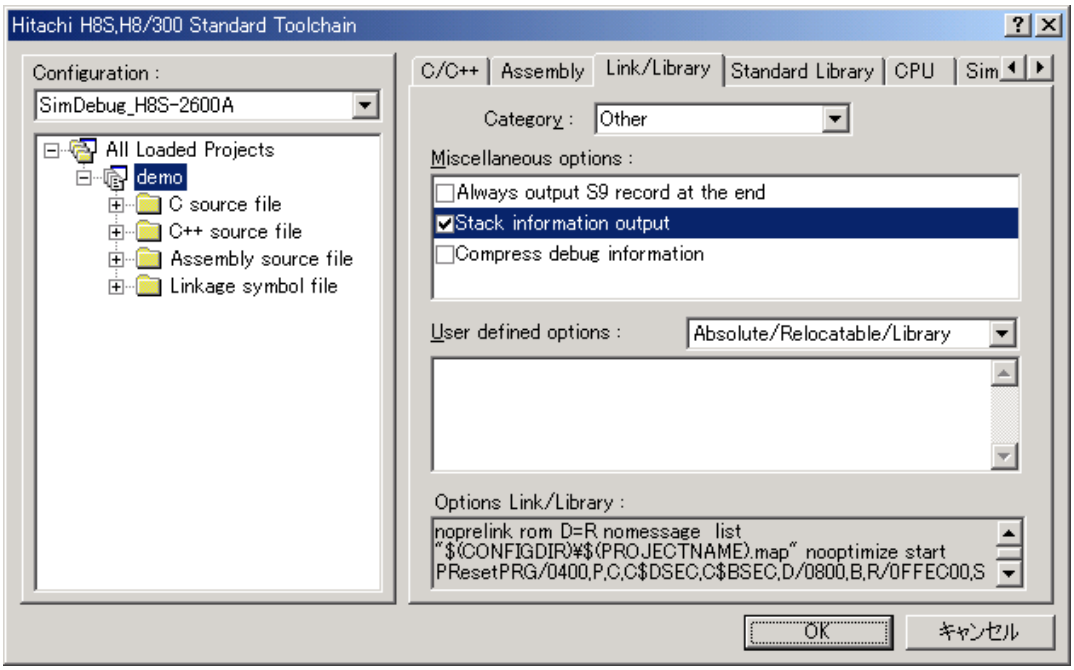


図 4-53 Standard Toolchain **ダイアログボックス**(1)

4.13.2 プロファイル情報ファイル

プロファイル情報ファイルを作成するためには、アプリケーションプログラムのプロファイルデータを測定後に、[プロファイル]ウィンドウのポップアップメニューで[プロファイル情報の保存...]メニューオプションを選択し、ファイル名を指定します。

プロファイル情報ファイルには、関数の呼び出し回数とグローバル変数のアクセス回数の情報が入っています。最適化リンカ(Ver.7.0以降)は、プロファイル情報ファイルを読み込み、関数および変数の配置を実際のプログラム動作状況に合わせた配置に最適化する機能を持っています。

プロファイル情報ファイルをリンカに入力するには、[Standard Toolchain]ダイアログの[Link/Library]タブで[Category]リストボックスを[Optimize]に指定し、[Include Profile]チェックボックスをチェックして、プロファイル情報ファイル名を指定してください。

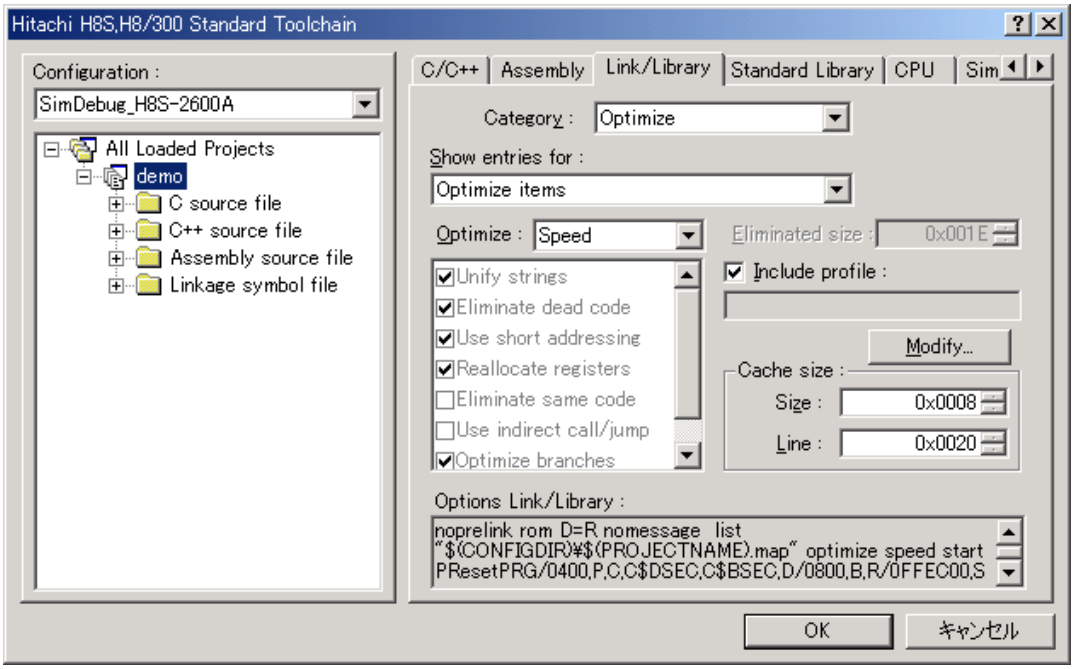


図 4-54 Standard Toolchain **ダイアログボックス**(2)

なお、[Include Profile]チェックボックスを有効にするには、[Optimize]リストボックスを[None]以外に設定する必要があります。

4.13.3 スタック情報ファイルのロード

スタック情報ファイルを読み込むかどうかは、ロードモジュールロード時に表示する、確認のメッセージボックスで指定できます。メッセージボックスの[OK]ボタンをクリックするとスタック情報ファイルをロードします。

確認のメッセージボックスは、次の場合に表示します。

- スタック情報ファイルが存在する時
- [オプション]ダイアログボックス(メインメニューの[ツール->オプション...]を選択すると開きます)の[確認]タブ（図 4-55）で[スタック情報ファイルをロードします (SNI ファイル)]チェックボックスをチェックしている場合

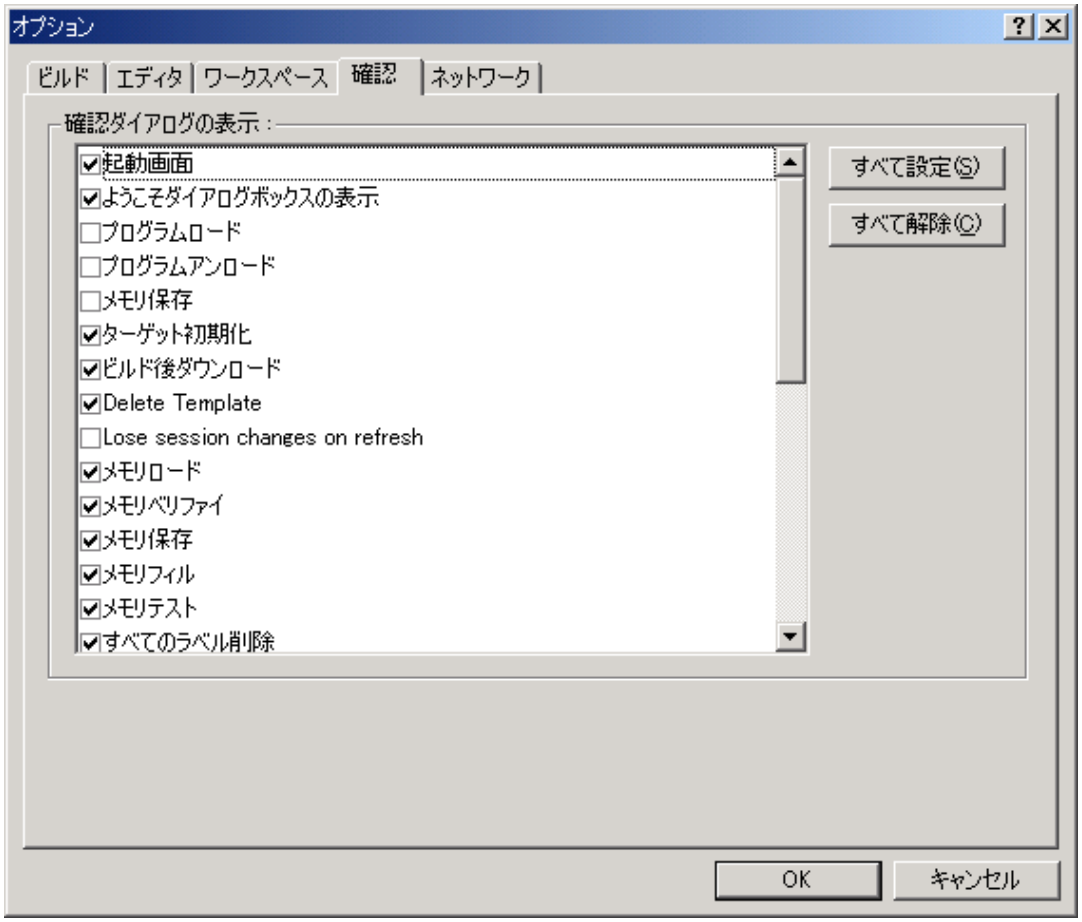


図 4-55 オプションダイアログボックス

4.13.4 プロファイルを有効にする

[表示->パフォーマンス->プロファイル]を選択し、[プロファイル]ウィンドウをオープンします。

[プロファイル]ウィンドウのポップアップメニューで[有効]メニューオプションを選択します（メニューにチェックマークが付きます）。

4.13.5 測定方法を指定する

プロファイルデータの測定時に、関数呼び出しをトレースするかどうかを指定できます。関数呼び出しをトレースすると、ユーザプログラム実行時の関数呼び出し関係をツリー形式で表示できるようになります。関数呼び出しをトレースしないと、関数呼び出し関係を表示できませんが、プロファイルデータの測定時間を短縮することができます。

関数呼び出しをトレースしないようにするためには、[プロファイル]ウィンドウのポップアップメニュー[関数呼び出しをトレースしない]を選択します。（メニューにチェックマークが付きます。）

また、OSによるタスクスイッチなど、通常の方法以外で関数を呼び出しているプログラムの場合、関数呼び出しを正しく表示できない場合がありますので、関数呼び出しをトレースせずにプロファイルデータを測定してください。

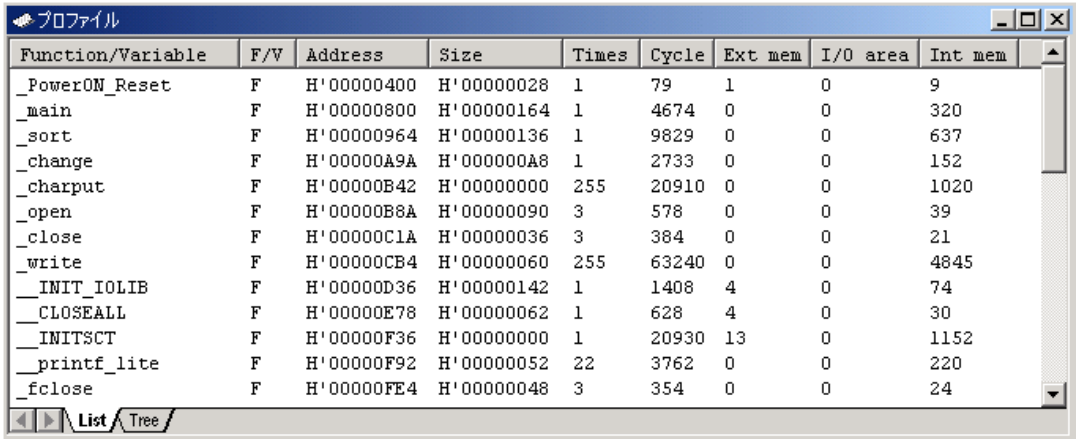
4.13.6 ユーザプログラムを実行し結果を確認する

ユーザプログラムを実行し、停止すると[プロファイル]ウィンドウに測定結果を表示します。

[プロファイル]ウィンドウには、[List]シートと[Tree]シートがあります。

4.13.7 List シート

関数とグローバル変数をリスト表示し、各関数/変数のプロファイルデータを表示します。



Function/Variable	F/V	Address	Size	Times	Cycle	Ext mem	I/O area	Int mem
_PowerON_Reset	F	H'00000400	H'00000028	1	79	1	0	9
_main	F	H'00000800	H'00000164	1	4674	0	0	320
_sort	F	H'00000964	H'00000136	1	9829	0	0	637
_change	F	H'00000A9A	H'000000A8	1	2733	0	0	152
_charput	F	H'00000B42	H'00000000	255	20910	0	0	1020
_open	F	H'00000B8A	H'00000090	3	578	0	0	39
_close	F	H'00000C1A	H'00000036	3	384	0	0	21
_write	F	H'00000CB4	H'00000060	255	63240	0	0	4845
__INIT_IOLIB	F	H'00000D36	H'00000142	1	1408	4	0	74
__CLOSEALL	F	H'00000E78	H'00000062	1	628	4	0	30
__INITSCT	F	H'00000F36	H'00000000	1	20930	13	0	1152
__printf_lite	F	H'00000F92	H'00000052	22	3762	0	0	220
_fclose	F	H'00000FE4	H'00000048	3	354	0	0	24

図 4-56 List **シート**

カラムヘッダをクリックすると、アルファベットまたは数値の昇降順にソートして表示します。

[Function/Variable]列または[Address]列をクリックすると、該当するアドレスに対応したソースプログラムを表示します。

ウィンドウ内でマウスの右ボタンをクリックするとポップアップメニューを表示します。このポップアップメニューについては「4.13.8 Tree シート」を参照してください。

4.13.8 Tree シート

関数の呼び出し関係を表示し、各呼び出し位置におけるプロファイルデータを表示します。

[Tree]シートは、[プロファイル]ウィンドウのポップアップメニュー[関数呼び出しをトレースしない]をチェックしていない時のみ有効です。

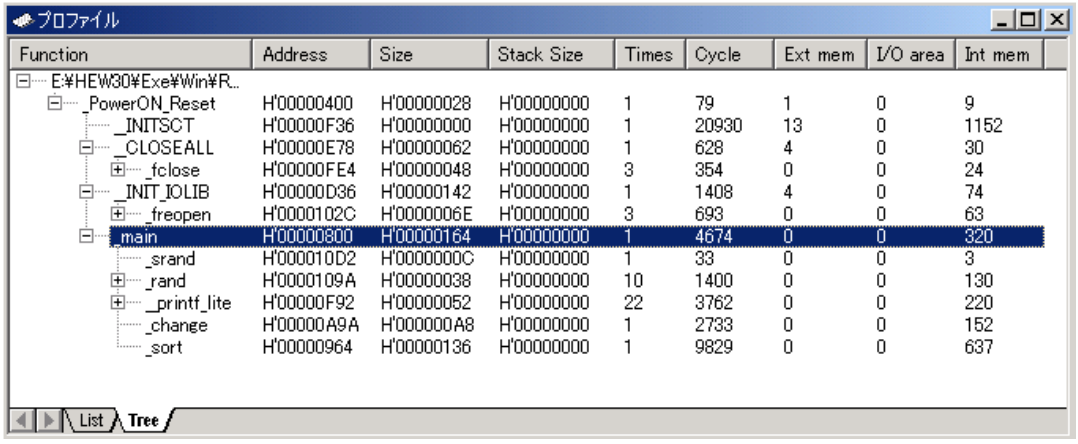


図 4-57 Tree **シート**

[Function]列の関数をダブルクリックすると、ツリー構造を拡張または収縮表示します。また、'+'／'-'キーでも拡張／収縮表示することができます。[Address]列をダブルクリックすると、該当するアドレスに対応したソースプログラムを表示します。

ウィンドウ内でマウスの右ボタンをクリックするとポップアップメニューを表示します。このメニューは以下のオプションを含みます。

(1) ソースファイル表示

選択している行の該当アドレスに対応したソースプログラムまたは逆アセンブルを表示します。

(2) チャート表示

選択している行の関数に着目した[プロファイルチャート]ウィンドウを表示します。

(3) 有効

プロファイルデータ測定のオン・オフを切り替えます。プロファイルデータ測定がオンのとき、メニューテキストの左にチェックマークを表示します。

(4) 関数呼び出しをトレースしない

本メニューをチェックすると、プロファイルデータ測定時に関数呼び出しをトレースしません。例えば、OS のタスクスイッチのように通常の方法以外で関数が呼び出されるプログラムのデータを測定する場合に使用します。

[プロファイル]ウィンドウの[Tree]シートで関数呼び出し関係を表示するためには、本メニューをチェックせずにプロファイルデータを測定してください。また、測定結果のプロファイル情報ファイルを使用して、最適化リンケージエディタによる最適化を行う場合も、本メニューをチェックしないでください。

(5) 検索...

[Function]列の文字列を検索する[テキスト検索]ダイアログボックスを表示します。検索したい文字列をエディットボックスに入力し、[次を検索]ボタンまたは、'Enter'キーを入力すると、検索を開始します。

(6) データ検索...

[データ検索]ダイアログボックスを表示します。

図 4-58 データ検索ダイアログボックス

[カラム]コンボボックスで検索カラムを、[検索データ]グループで検索方向を設定し、[次を検索]ボタンまたは、'Enter'キーを入力すると、検索を開始します。また、連続して[次を検索]ボタンまたは'Enter'キーを入力すると、次に大きいデータ（最小値の場合は小さいデータ）を検索します。

(7) データクリア

関数呼び出し回数のカウントおよびプロファイルデータをクリアします。[プロファイル]ウィンドウの[List]シートおよび[プロファイルチャート]ウィンドウのデータもクリアします。

(8) プロファイル情報の保存...

[プロファイル情報の保存]ダイアログボックスを表示します。プロファイル結果をプロファイル情報ファイル（拡張子は".pro"）に保存します。最適化リンケージエディタは、プロファイル情報を元に、ユーザプログラムの最適化を行うことが出来ます。プロファイル情報を使用した最適化についての詳細は、最適化リンケージエディタのマニュアルを参照してください。

【注】 関数呼び出しをトレースしないメニューをチェックして測定した結果のプロファイル情報では、最適化リンケージエディタによる最適化は行えません。

(9) テキスト形式で保存...

[プロファイルデータをテキスト形式で保存]ダイアログボックスを表示します。表示している状態をテキストファイルに保存します。

(10) 表示設定

このメニューには下記サブメニューがあります。（以下の説明には[List]シートのみのメニューも含みます）

(a) 関数と変数を表示

[Function/Variable]列で、関数およびグローバル変数の両方表示します。

(b) 関数を表示

[Function/Variable]列で、関数のみを表示します。

(c) 変数を表示

[Function/Variable]列で、グローバル変数のみを表示します。

(d) 未実行関数を表示しない

実行した関数のみ表示することができます。最適化リンケージエディタが出力するスタック使用量情報ファイル(拡張子：sni)がロードモジュールと同一ディレクトリに存在しない場合、このチェックボックスの設定に関わらず、実行関数のみ表示します。

(e) 子関数の実行結果を含んで表示

表示するプロファイルデータに、関数内で呼び出した子関数のプロファイルデータを含めるかどうかを設定します。

(11) プロパティ...

本シミュレータ・デバッガでは使用できません。

4.13.9 プロファイルチャートウィンドウ

[プロファイルチャート]ウィンドウは、特定の関数に着目した関数の呼び出し関係を表示します。本ウィンドウは、着目する関数を中心に表示し、その左側には着目した関数を呼び出した関数、右側には、着目している関数が呼び出した関数を、それぞれ表示します。また、各呼び出しを行った回数も表示します。

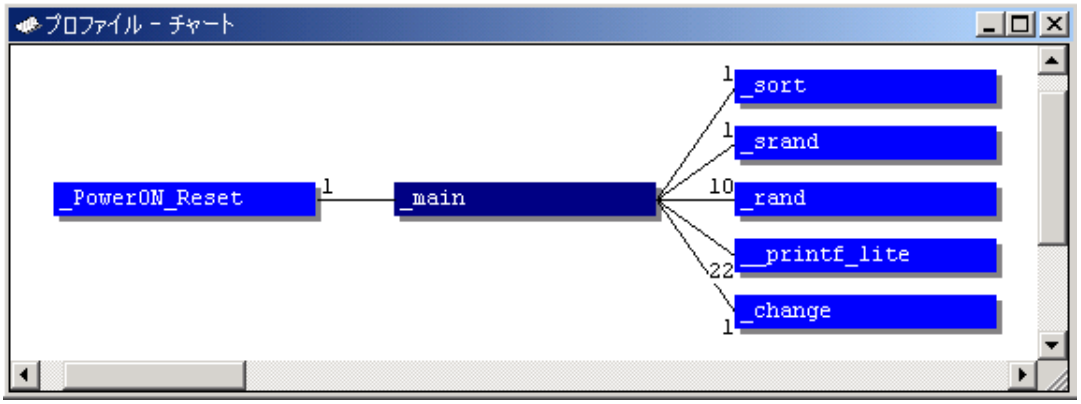


図 4-59 プロファイルチャートウィンドウ

4.13.10 表示データの種類および用途

プロファイル機能から下記情報を得ることができます。

(1) Address

関数を配置しているメモリ上の位置を知ることができます。アドレス順にソート表示することにより、メモリ上の配置イメージで関数とグローバル変数を並べることができます。

(2) Size

サイズ順にソート表示すれば、サイズが小さくて頻繁に呼び出している関数を見つけることができます。そのような関数があればinline関数にすることで、関数呼び出しのオーバヘッドを減らせる場合があります。

(3) Stack Size

関数呼び出しのネストが深い場合、関数呼び出し経路をたどり、その経路上の全関数のスタックサイズを合計することで、おおよそのスタック使用量を見積もれます。

(4) Times

呼び出し（アクセス）回数順にソート表示すれば、頻繁に呼び出している関数や頻繁にアクセスしている変数を容易に調べることができます。

(5) **プロファイルデータ**

- [Cycle]（実行サイクル数）
- [Ext_mem]（外部メモリアクセス回数）
- [I/O_area]（内蔵 I/O アクセス回数）
- [Int_mem]（内部メモリアクセス回数）

実行サイクル数は、当該関数コール命令実行時の累計実行サイクル数と当該関数からのリターン命令実行時の累計実行サイクル数の差から求めています。

4.13.11 プロファイル情報ファイルを作成する

プロファイル情報ファイルを作成する場合は、Pop-upメニューの[プロファイル情報の保存...]メニューオプションを選択します。[プロファイル情報の保存]ダイアログを表示します。ファイル名を選択して[保存]ボタンを押すと、選択したファイルにプロファイル情報を書きこみます。[全て保存]ボタンを押すと、全てのファイルにプロファイル情報を書きこみます。

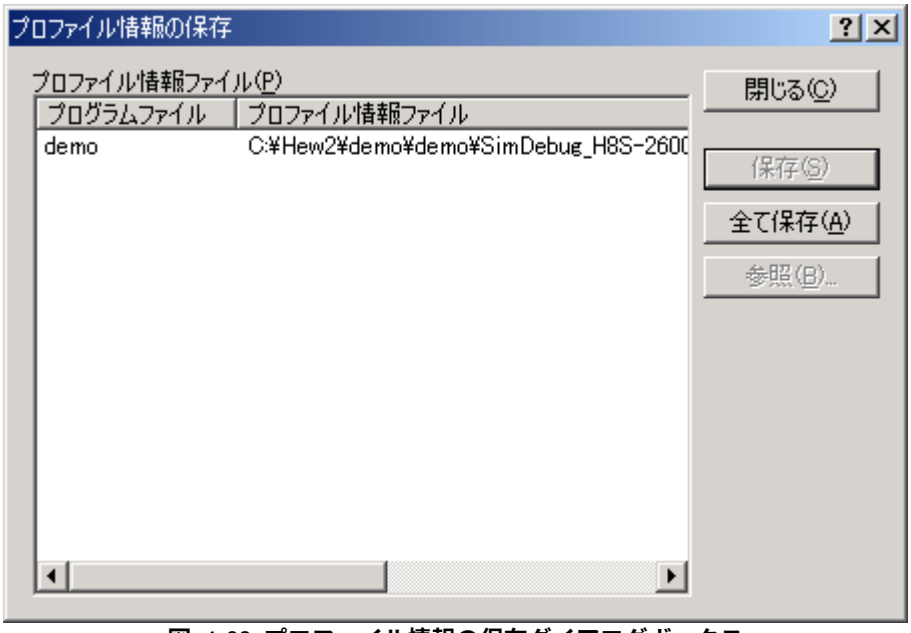


図 4-60 **プロファイル情報の保存ダイアログボックス**

4.13.12 注意事項

（1） アプリケーションプログラムの実行サイクル数をプロファイル機能で測定した場合のデータには誤差があります。プロファイル機能では、アプリケーションプログラム全体の中で各関数が占める実行時間の比率を調べることができますが、より厳密に関数の実行サイクル数を測定したい場合には、パフォーマンス解析機能を使用してください。
（2） デバッグ情報が無いロードモジュールでプロファイル情報の測定を行った場合、関数名を表示しない場合があります。
（3） スタック情報ファイル(拡張子".SNI")はロードモジュールファイル(拡張子".ABS")と同一のディレクトリに置いてある必要があります。
（4） 測定結果の蓄積はできません。
（5） 測定結果の編集はできません。

4.14 関数呼出し履歴を見る

関数呼出し履歴を表示する場合は、[スタックトレース]ウィンドウを使用します。

4.14.1 スタックトレースウィンドウを開く

[スタックトレース]ウィンドウを開くには、[表示->コード->スタックトレース]を選択するか、[スタックトレース]ツールバーボタンを  クリックします。

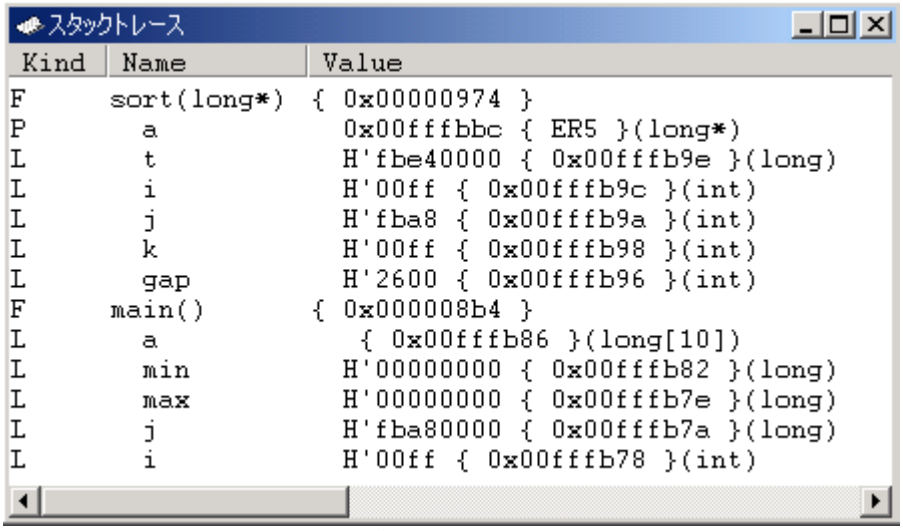


図 4-61 スタックトレースウィンドウ

表示する項目は以下の通りです。

[Kind]	該当シンボルのシンボル種別を示します。 F：関数 P：関数パラメータ L：ローカル変数
[Name]	シンボル名を示します。
[Value]	シンボルの値、アドレス、型を示します。

4.14.2 ソースプログラムを表示する

関数を選択した状態で、ポップアップメニューから[ソースファイル表示]を選択すると、[エディタ]ウィンドウを開いて選択した関数に該当するソースプログラムを表示します。

4.14.3 表示形式を設定する

ポップアップメニューから[表示設定...]を選択すると、[スタックトレース表示設定]ダイアログボックスを表示します。[スタックトレース]ウィンドウの表示形式を設定します。

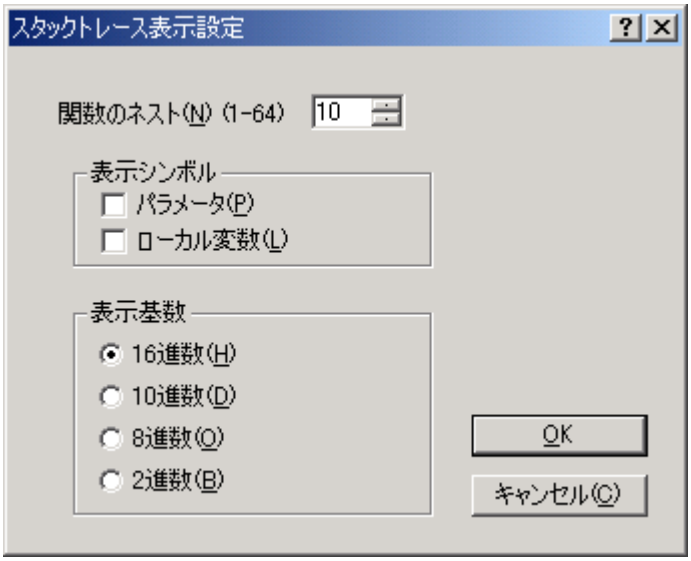


図 4-62 スタックトレース表示設定ダイアログボックス

[関数のネスト] [スタックトレース]ウィンドウに表示する関数コールネスト数を指定します。
[表示シンボル] 関数以外に表示するシンボルを指定します。
[表示基数] [スタックトレース]ウィンドウの表示基数を指定します。

4.15 トレース情報を見る

シミュレータ・デバッガでは命令の実行結果をトレース情報として取得および表示することができます。

トレース情報は、[トレース]ウィンドウに表示します。トレース情報の取得条件は、[トレース取得]ダイアログボックスで設定します。

4.15.1 トレースウィンドウを開く

[トレース]ウィンドウを開くには、[表示->コード->トレース]を選択するか、[トレース]ツールバーボタン をクリックします。

4.15.2 トレース情報取得条件を設定する

[トレース]ウィンドウを開いたら、トレース情報取得条件を設定します。
トレース情報取得条件は、[トレース取得]ダイアログボックスで設定します。
[トレース取得]ダイアログボックスを開くには、ポップアップメニューから[設定...]を選択します。

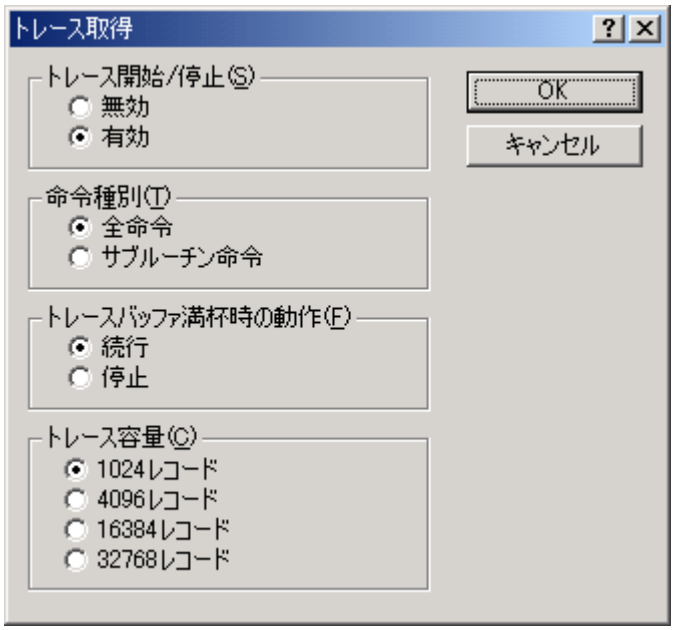


図 4-63 トレース取得ダイアログボックス

本ダイアログボックスでは、トレース情報の取得条件を設定します。

[トレース開始/停止]

[無効]	トレース情報の取得停止
[有効]	トレース情報の取得開始

[命令種別]

[全命令]	全命令のトレース情報を取得
[サブルーチン命令]	サブルーチン命令のみトレース情報を取得

[トレースバッファ満杯時の動作]

[続行]	トレース情報取得バッファが満杯になっても取得を続行
[停止]	トレース情報取得バッファが満杯になった場合、実行停止

また、トレースバッファの大きさは、[トレース容量]により 1024、4096、16384、32768 レコードの中から選択します。

指定した内容は、[OK]ボタンをクリックすることにより設定します。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

4.15.3 トレース情報を取得する

トレース情報の取得を開始した状態で命令を実行すると、トレースを取得できます。
取得したトレース情報は[トレース]ウィンドウに表示します。

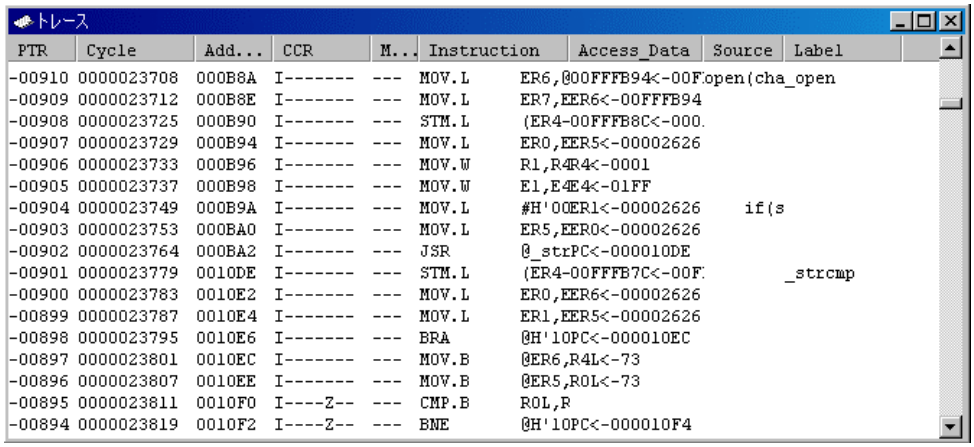


図 4-64 トレースウィンドウ

表示する項目は以下の通りです。

[PTR]	トレースバッファ内ポインタ（最後に実行した命令が 0 となります）
[Cycle]	累計命令実行サイクル数 （命令実行リセットによりクリアします）
[Address]	命令アドレス
[CCR]	コンディションコードレジスタ（CCR）の内容をニモニックで表示します。
[Mult]	乗算器内部のフラグをニモニックで表示します。（H8S/2600 シリーズのみ）
[Instruction]	命令ニモニック
[Access Data]	データアクセス（転送先<-転送データの形式で表示）
[Source]	C/C++またはアセンブラソース
[Label]	ラベル

4.15.4 Trace レコードを検索する

トレースレコードを検索するには[トレース検索]ダイアログボックスを使用します。
[トレース検索]ダイアログボックスを開くには、ポップアップメニューの[検索...]を選択します。

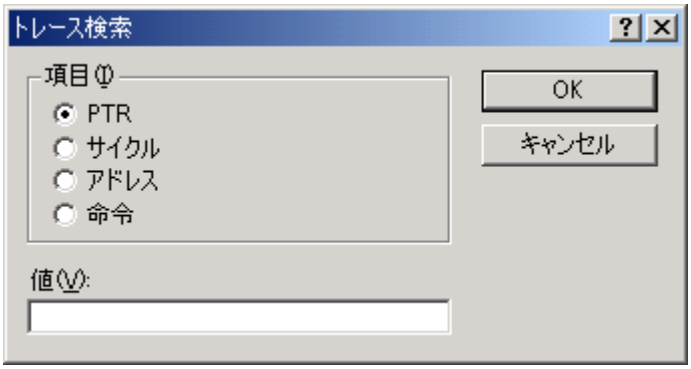


図 4-65 トレース検索ダイアログボックス

本ダイアログボックスは、トレース情報のサーチ条件を設定します。[項目]でサーチ対象項目を指定し、[値]で指定した内容をサーチします。

[PTR]	トレースバッファ内ポインタ。最後に実行した命令が 0 となります。 -nnn の形式で指定してください。
[サイクル]	累計命令実行サイクル数
[アドレス]	命令アドレス
[命令]	命令ニモニック

[OK]ボタンをクリックすることにより、サーチ条件を設定し、検索を開始します。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

検索の結果一致するトレースレコードが見つかった場合は当該レコード行を青色表示します。一致するトレースレコードが見つからなかった場合は、メッセージダイアログボックスを表示します。

トレースレコードが検索できた場合は、ポップアップメニューで[次を検索]を選択すると、次のトレースレコードを検索できます。

4.15.5 トレース情報をクリアする

トレース情報をクリアするには、ポップアップメニューから[クリア]を選択します。トレース情報を保持しているトレースバッファを空にします。複数の[トレース]ウインドウが開いているときは、それらは同じバッファをアクセスしているため、すべての[トレース]ウィンドウをクリアすることになります。

4.15.6 トレース情報をファイルに保存する

トレース情報をファイルに保存するには、ポップアップメニューから[保存...]を選択します。

[名前を付けて保存]ファイルダイアログボックスを表示します。トレースバッファの内容をテキストファイルとして保存します。保存する範囲を、[PTR]の範囲によって指定することができます。このファイルはトレースバッファに再ロードできないことに注意してください。

4.15.7 ソースファイルを表示する

トレースレコードに対応するソースファイルを[エディタ]ウィンドウに表示するには二通りの方法があります。

(1) **トレースレコードを選択した状態でポップアップメニューから[ソースファイル表示]を選択する**

(2) **トレースレコードをダブルクリックする**

上記の操作により、[エディタ]ウィンドウあるいは[逆アセンブリ]ウィンドウを開いて、選択した行をカーソルで示します。

4.15.8 ソース表示を整形する

ポップアップメニューで[ソーストリム]を選択すると、ソースプログラムの左側の空白を取り除き

ます。
取り除いた状態だと[ソーストリム]メニューの左にチェックが付きます。チェックありの状態で[ソーストリム]メニューを選択すると取り除いた空白を元に戻します。

4.15.9 統計情報を解析する

指定された条件で統計情報の解析を実行するには、ポップアップメニューから[統計]を選択します。[統計]ダイアログボックスが開きます。

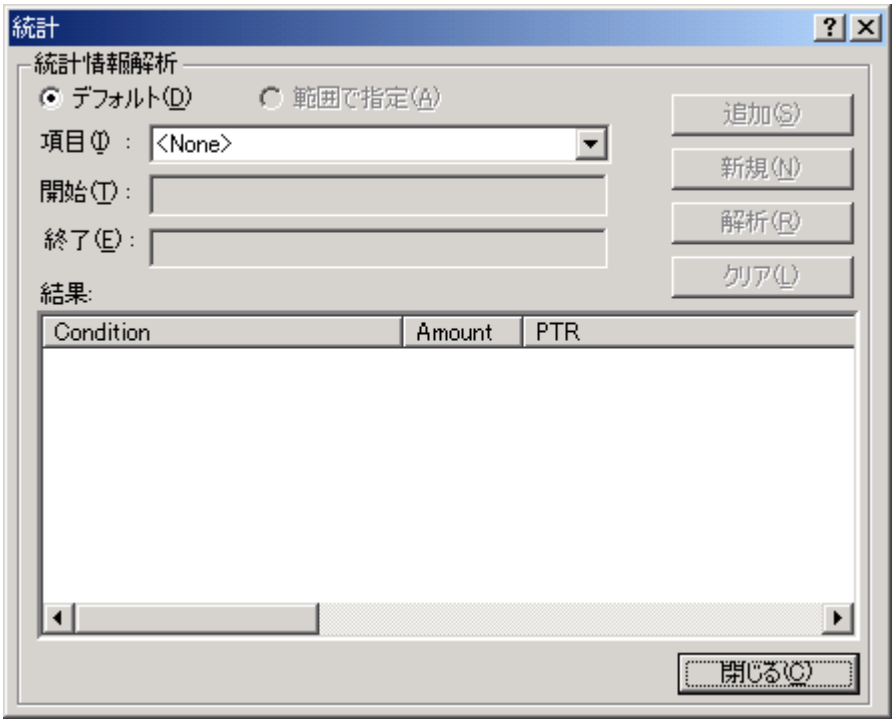


図 4-66 統計ダイアログボックス

本ダイアログボックスは、トレース情報の統計情報解析に使用します。[項目]で解析対象項目を指定し、[開始]および[終了]で入力値または文字列を指定します。
[デフォルト]を選択すると入力値または文字列を範囲で指定することはできません。範囲で指定する場合は[範囲で指定]を選択してください。

[追加]	現在の条件に追加設定します
[新規]	新しい条件を指定します
[解析]	統計情報解析の結果を取得します
[クリア]	すべての条件と統計情報解析結果を削除します

[閉じる]ボタンをクリックすると、ダイアログボックスを閉じます。

4.16　シミュレータ・デバッガのブレークポイントを使用する

シミュレータ・デバッガでは HEW 標準の PC ブレークポイントとは別により高度なブレークポイント機能を持っています。

これらブレークポイントについて、ブレーク条件の設定、ブレーク条件成立時の動作、および設定されているブレークポイントの表示が行えます。

4.16.1　ブレークポイントを一覧表示する

現在設定されているブレークポイントを一覧表示するには[イベントポイント]ウィンドウを開きます。

[イベントポイント]ウィンドウは[表示->コード->イベントポイント]を選択するか、[イベントポイント]ツールバーボタン をクリックします。

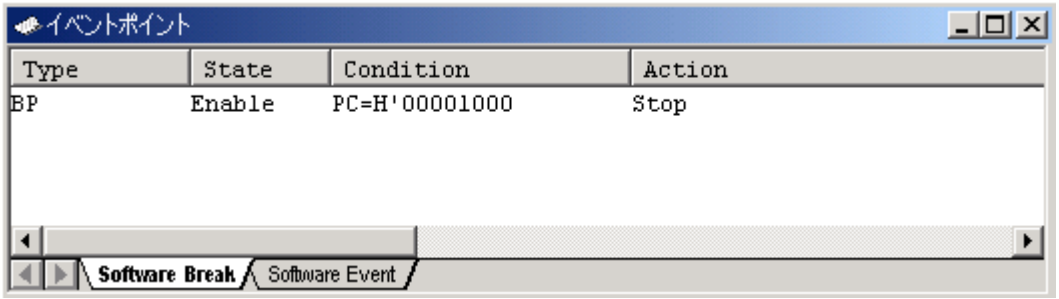


図 4-67　イベントポイントウィンドウ

表示する項目は以下の通りです。

[Type]　ブレーク種別を表示します。
- [BP]　：PC ブレーク
- [BA]　：ブレークアクセス
- [BD]　：ブレークデータ
- [BR]　：ブレークレジスタ
- [BS]　：ブレークシーケンス
- [BCY]　：ブレークサイクル

[State]　該当ブレークポイントの有効/無効を示します。
- [Enable]　：有効
- [Disable]　：無効

[Condition]　Break が成立する条件を表示します。表示内容はブレーク種別により異なります。
ブレーク種別が BR の時はレジスタ名を、BCY の時はサイクル数を表示します。
- BP 時　：PC=プログラムカウンタ（対応するファイル名／行、シンボル名）
- BA 時　：Address=アドレス（シンボル名）
- BD 時　：Address=アドレス（シンボル名）
- BR 時　：Register=レジスタ名
- BS 時　：PC=プログラムカウンタ（対応するファイル名／行、シンボル名）
- BCY 時　：Cycle=サイクル数（16 進表示）

[Action]　ブレーク条件成立時の動作を表示します。
- [Stop]　：実行停止
- [File Input]（ファイル名）［ファイルの状態］　：ファイルからのメモリデータ読みこみ
- [File Output]（ファイル名）［ファイルの状態］　：ファイルへメモリデータ書きこみ
- [Interrupt]（割込み種別／優先順位）　：割込み処理

[Action]の設定で、[Stop]を指定した条件を[Software Break]タブに、[Stop]以外を指定した条件を[Software Event]タブに表示します。

4.16.2 ブレークポイントを設定する

[イベントポイント]ウィンドウのポップアップメニューで[設定…]を選択すると、[ブレーク設定]ダイアログボックスが開きブレークポイントを設定できます。

[ブレーク設定]ダイアログボックスにはブレーク条件を設定する[条件]ページとブレーク成立時の動作を設定する[動作]ページがあります。[動作]の設定で、[Stop]を指定する場合は[Software Break]タブで、[Stop]以外を指定する場合は[Software Event]タブでポップアップメニューを選択します。

(1) ブレーク条件を設定する

ブレーク条件は図 4-68に示す[ブレーク設定]ダイアログボックスの[条件]ページで設定します。

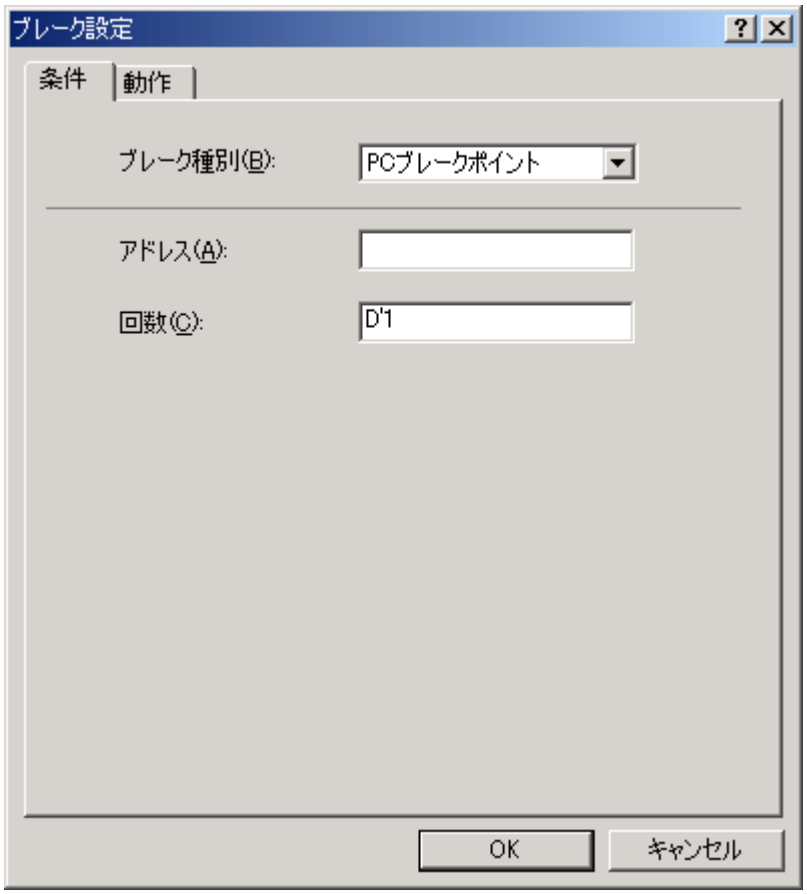


図 4-68 ブレーク設定ダイアログボックス（条件ページ）

本ダイアログボックスでは、ブレーク条件を設定します。

まず、設定するブレーク種別を [ブレーク種別]で指定します。各種別において設定可能な項目を以下に示します。

[PC ブレークポイント] 1024 個まで指定可能
- [アドレス] ブレークする命令の位置
- [回数] 指定位置の命令をフェッチする回数
  （接頭辞省略時は 10 進入力、10 進表示）（1～16383、省略すると 1 となります）

[ブレークアクセス] 1024 個まで指定可能
- [開始アドレス] アクセスするとブレークするメモリの開始位置
- [終了アドレス] アクセスするとブレークするメモリの終了位置
  （省略すると開始位置のみが範囲となります）
- [アクセス種別] アクセス種別

[ブレークデータ] 1024 個まで指定可能
- [アドレス] ブレーク判定を行うメモリの位置
- [データ] ブレーク条件となるデータ値
- [サイズ] データのサイズ
- [オプション] データの一致/不一致
- [データのマスク] マスク条件（0 を指定したビットがマスクされます）

[ブレークレジスタ] 1024 個まで指定可能
- [レジスタ] ブレーク条件を設定するレジスタ名
- [サイズ] データのサイズ
- [データ] ブレーク条件となるデータ値
  （省略するとレジスタへ書き込む度にブレークします）
- [オプション] データの一致/不一致
- [データのマスク] マスク条件（0 を指定したビットがマスクされます）

[ブレークシーケンス] 1 個のみ指定可能
- [アドレス 1]～[アドレス 8] ブレークの発生条件となる通過アドレス
  （８ポイントすべてを設定する必要はありません。）

[ブレークサイクル] 1024 個まで指定可能
- [サイクル] ブレーク判定を行うサイクル数（1～H’FFFFFFFF）
  [サイクル]×n のサイクルで条件が一致します。
  ただし、指定したサイクルと実際に条件が一致するサイクルはずれることがあります。
- [回数] ブレークが成立する回数
  - [すべて] 条件が一致するごとに、ブレークが成立します。
  - [回数指定] （接頭辞省略時は 16 進入力、16 進表示）（１~65535）
    条件一致した回数が、[回数指定]以下の時だけブレークが成立します。

[PC ブレークポイント]および[ブレークシーケンス]の設定時に、アドレスに多重定義関数あるいはメンバ関数を含むクラス名を入力した場合、[関数選択]ダイアログボックスが開くので設定する関数を選択します。詳細は、**「**4.11.3　複数ラベルをサポートする**」**を参照してください。

指定したブレーク条件は、[OK]ボタンをクリックすることにより設定します。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

(2) ブレーク条件成立時の動作を設定する

ブレーク条件成立時の動作は図 4-69に示す[ブレーク設定]ダイアログボックスの[動作]ページで設定します。

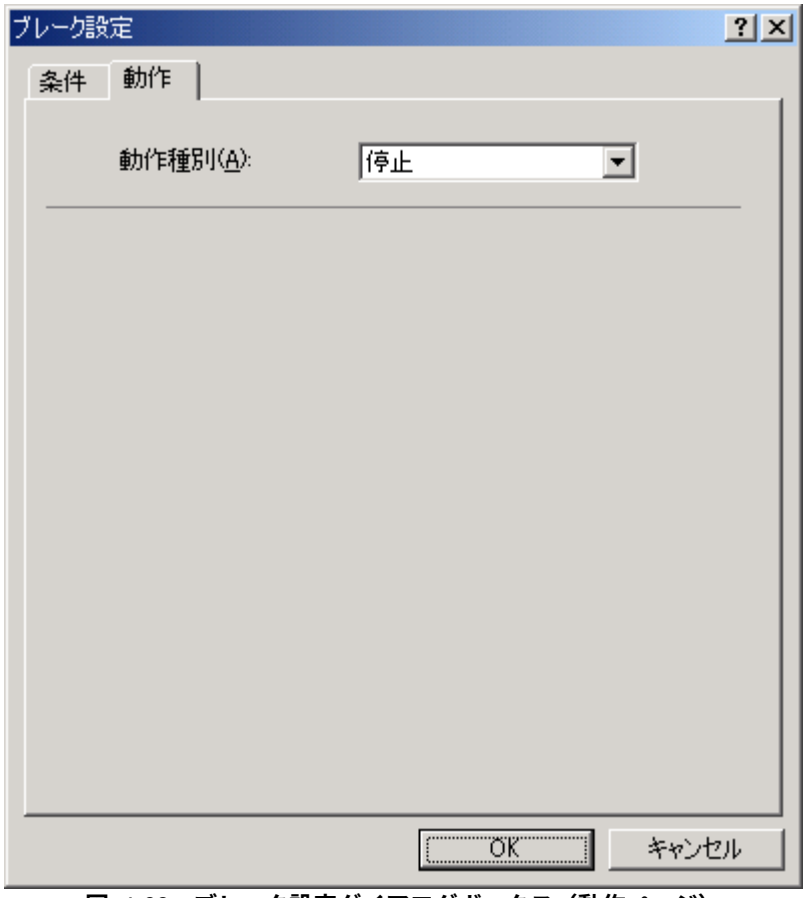


図 4-69 ブレーク設定ダイアログボックス（動作ページ）

本ダイアログボックスでは、ブレーク条件成立時の動作を設定します。

まず、設定する動作を [動作種別]で指定します。各動作において設定可能な項目を以下に示します。

[停止]　条件成立時にユーザプログラムの実行を停止します。
設定する項目は有りません。

[ファイル入力]　条件成立時に指定ファイルから読みこんだデータを指定メモリへ書きこみます。

- [入力ファイル]　読みこむデータファイルを指定します。
  ファイルの終了まで読みこんだら、先頭から繰り返して読みこみます。
- [アドレス]　データを書きこむメモリのアドレスを指定します。
- [データサイズ]　書きこむデータ１個のサイズを指定します。（1/2/4/8）
- [データ数]　書きこむデータの個数を指定します。(接頭辞省略時は 10 進入力、10 進表示）（１～H'FFFFFFFF）

[ファイル出力]　条件成立時に指定メモリの内容を指定ファイルへ書きこみます。

- [出力ファイル]　書きこむデータファイルを指定します。
- [追記]　既存のファイルを[出力ファイル]で指定した場合に、ファイルの最後に追加出力するかを指定します。
- [アドレス]　データを読みこむメモリのアドレスを指定します。
- [データサイズ]　読みこむデータ１個のサイズを指定します。（1/2/4/8）
- [データ数]　読みこむデータの個数を指定します。(接頭辞省略時は 10 進入力、10 進表示）（１～H'FFFFFFFF）

[割り込み]　条件成立時に割込み処理を行います。詳しくは、「2.16 擬似割込み」を参照してください。

- [割り込み種別 1]　割込みベクタ番号を指定します。（接頭辞省略時は 16 進入力、16 進表示）（0~H'FF）
- [優先順位]　割込み優先順位を指定します。（接頭辞省略時は 16 進入力、16 進表示）（0～H'11）
  割込みを受け付けるかどうかの判定は、選択されたデバッグプラットフォームの CPU の仕様に従います。ただし、優先順位に H'8 以上を指定した場合は、常に割込みを受け付けます。

【注】 複数の[ファイル入力]で同一ファイルを指定した場合、ブレーク成立順にファイルからデータを読み込みます。複数の[ファイル出力]で同一ファイルを指定した場合、ブレーク成立順にファイルへデータを書きこみます。ただし、[ファイル入力]と[ファイル出力]で同一ファイルを指定した場合は、最初に成立した動作のみが有効になります。

4.16.3 ブレークポイントの設定内容を変更する

変更したいブレークポイントを選択後ポップアップメニューから[編集…]を選択すると、[ブレーク設定]ダイアログボックスが開き、ブレーク条件を変更することができます。[編集…]メニューはブレークポイントを 1 個選択しているときのみ有効となります。

4.16.4 ブレークポイントを有効にする

ブレークポイントを選択後ポップアップメニューから[有効]を選択すると、選択しているブレークポイントを有効にします。

4.16.5 ブレークポイントを無効にする

ブレークポイントを選択後ポップアップメニューから[無効]を選択すると、選択しているブレーク

ポイントを無効にします。無効にした場合は、ブレークポイントはリストには残りますが、指定した条件が一致してもブレークは成立しません。

4.16.6 ブレークポイントを削除する

ブレークポイントを選択後ポップアップメニューから[削除]を選択すると、選択しているブレークポイントを削除します。ブレークポイントを削除しないで、詳細情報は保持したまま、条件が一致してもブレークを成立させないようにするには、[無効]オプションを使用します(「4.16.5 ブレークポイントを無効にする」参照)。

4.16.7 ブレークポイントをすべて削除する

ポップアップメニューから[すべて削除]を選択すると、すべてのブレークポイントを削除します。

4.16.8 ブレークポイントのソース行を表示する

ブレークポイントを選択後ポップアップメニューから[ソースファイル表示]を選択すると、ブレークポイントのある[ソース]または[逆アセンブリ]ウィンドウをオープンします。[ソースファイル表示]メニューはブレークポイントを 1 個選択しているときのみ有効となります。

4.16.9 入出力ファイルを閉じる

ブレークポイントを選択後ポップアップメニューから[ファイルを閉じる]を選択すると、選択した[ファイル入力] または[ファイル出力]のデータファイルを閉じ、ファイル読み出し位置をリセットします。

4.16.10 入出力ファイルをすべて閉じる

ポップアップメニューから[すべてのファイルを閉じる]を選択すると、すべての[ファイル入力] および[ファイル出力]のデータファイルを閉じ、ファイル読み出し位置をリセットします。

4.17 パフォーマンスを解析する

関数名を指定してパフォーマンスを解析する場合は、[パフォーマンス解析]ウィンドウを使用します。

4.17.1 パフォーマンス解析ウィンドウを開く

[パフォーマンス解析]ウィンドウを開くには、[表示->パフォーマンス->パフォーマンス解析]を選択するか、[パフォーマンス解析]ツールバーボタン  をクリックします。

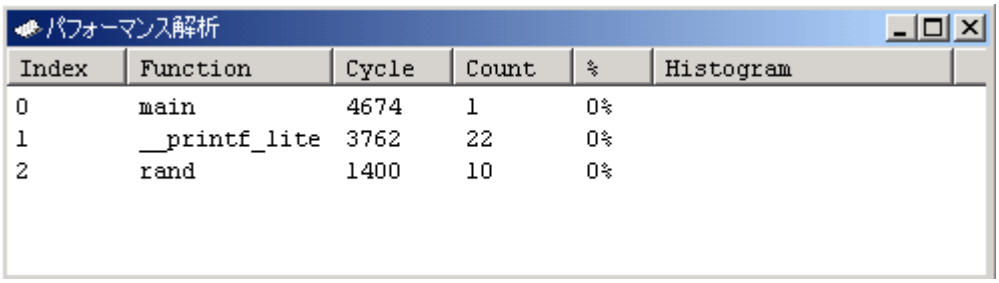


図 4-70 **パフォーマンス解析ウィンドウ**

本ウィンドウは、指定関数ごとの実行サイクル数を表示します。
実行サイクル数は、以下の計算で求めています。

実行サイクル数 = 指定関数からのリターン命令実行時累計実行サイクル数 - 指定関数コール命令実行時累計実行サイクル数

表示する項目は以下の通りです。

[Index]	設定条件のインデックス番号
[Function]	測定対象の関数名（または関数の開始アドレス）
[Cycle]	当該関数の累計実行サイクル数
[Count]	当該関数の累計呼び出し回数
[%]	プログラム全体の実行サイクル数に占める当該関数の実行サイクル数の割合
[Histogram]	上記割合のヒストグラム表示

4.17.2 評価関数を設定する

[パフォーマンス解析]ウィンドウを開いたら、評価する関数を設定します。ポップアップメニューから[範囲の追加...]を選択すると、[パフォーマンスオプション]ダイアログボックスが開きます。また、'Insert'キーでも[パフォーマンスオプション]ダイアログボックスを開くことができます。

図 4-71 **パフォーマンスオプションダイアログボックス**

本ダイアログボックスでは、性能評価する関数（ラベルも可）を設定します。評価関数は255個まで設定可能です。
多重定義関数あるいはメンバ関数を含むクラス名を入力した場合、[関数選択]ダイアログボックスが開くので設定する関数を選択します。詳細は、**「**4.11.3 複数ラベルをサポートする**」**を参照してください。

[OK]ボタンをクリックすることにより、評価関数を設定します。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

設定した評価関数を選択後、ポップアップメニューの[範囲の編集]を選択すると[パフォーマンスオプション]ダイアログボックスを表示して、評価関数を変更できます。また、'Enter'キーでも[パフォーマンスオプション]ダイアログボックスを開くことができます。

4.17.3 データ収集を開始する

ポップアップメニューの[有効]をチェックして、プログラムを実行するとパフォーマンスデータを収集できます。

4.17.4 データをリセットする

ポップアップメニューから[リセット]を選択すると、現在のプログラムのパフォーマンスデータをクリアします。

4.17.5 評価関数を削除する

評価関数を選択した状態で、ポップアップメニューから[範囲の削除]を選択すると、選択された評価関数を削除し、他の範囲のデータを再計算します。また、'Delete'キーでも評価関数を削除することができます。

4.17.6 すべての評価関数を削除する

ポップアップメニューから[全ての範囲を削除]を選択すると、現在の評価関数をすべて削除し、パフォーマンスデータをクリアします。

4.17.7 現在の表示内容をセーブする

現在ウィンドウに表示している内容をテキストファイルにセーブすることが出来ます。ポップアップメニューから[ファイルに保存...]を選択してください。

4.18 コードカバレジを測定する

[カバレジ]ウィンドウは、ユーザが指定したアドレス範囲についてコードカバレジ情報(C0 カバレジおよび C1 カバレジ)を収集し、結果を表示します。

4.18.1 カバレジウィンドウを開く

[表示->コード->カバレジ...]を選択するか[カバレジ]ツールバーボタン  をクリックすると、[カバレジウィンドウを開く]ダイアログボックスが開きます。

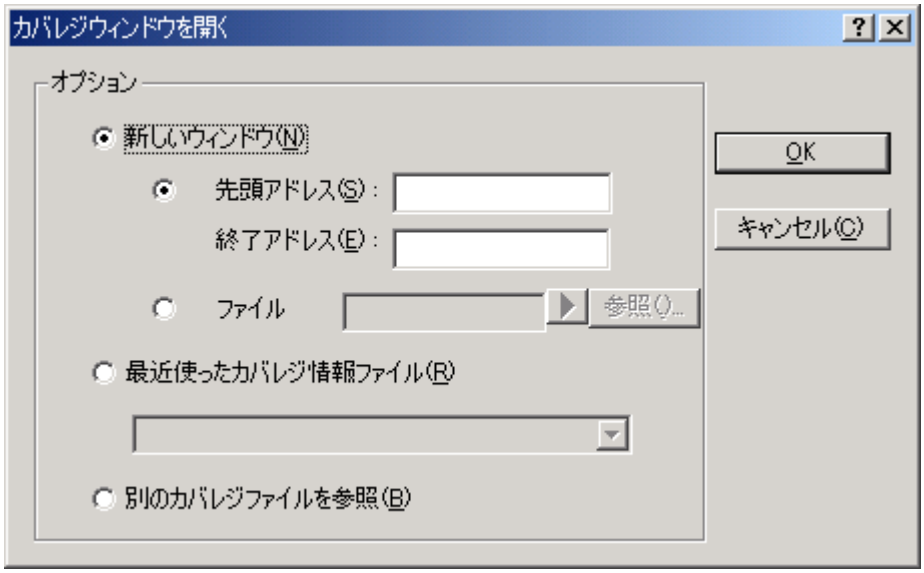


図 4-72 カバレジウィンドウを開くダイアログボックス

[カバレジウィンドウを開く]ダイアログボックスでは、カバレジ測定範囲を指定します。
新たなカバレジ範囲を設定するには、以下に示す 2 通りの方法があります。

- **新しいウィンドウで開始アドレスおよび終了アドレスを指定する**

指定項目は以下のとおりです。

[先頭アドレス] カバレジ情報表示の開始アドレスを指定します。（接頭辞省略時は 16 進で入力）
[終了アドレス] カバレジ情報表示の終了アドレスを指定します。（接頭辞省略時は 16 進で入力）

- **新しいウィンドウでファイルを指定する**

指定項目は以下のとおりです。

[ファイル] 現在のプロジェクト中の".C"または".CPP"を拡張子にもつソースファイルを指定します。
これにより指定ファイル内に存在する関数をカバレジ範囲として設定します。
拡張子を省略した場合は、".C"を補います。".C"または".CPP"以外の拡張子を持つファイルは指定できません。
プレースホルダまたは[Browse...]ボタンが利用できます。

カバレジ情報ファイルに保存した内容を再設定する場合は、[最近使ったカバレジ情報ファイル]から選択するか、[別のカバレジファイルを参照]でファイルオープンダイアログボックスを表示して選択します。
[最近使ったカバレジ情報ファイル]では最近保存されたファイルを 4 個まで表示します。
[OK]ボタンをクリックすると[カバレジ]ウィンドウを表示します。

(1) カバレジウィンドウ（アドレス指定）

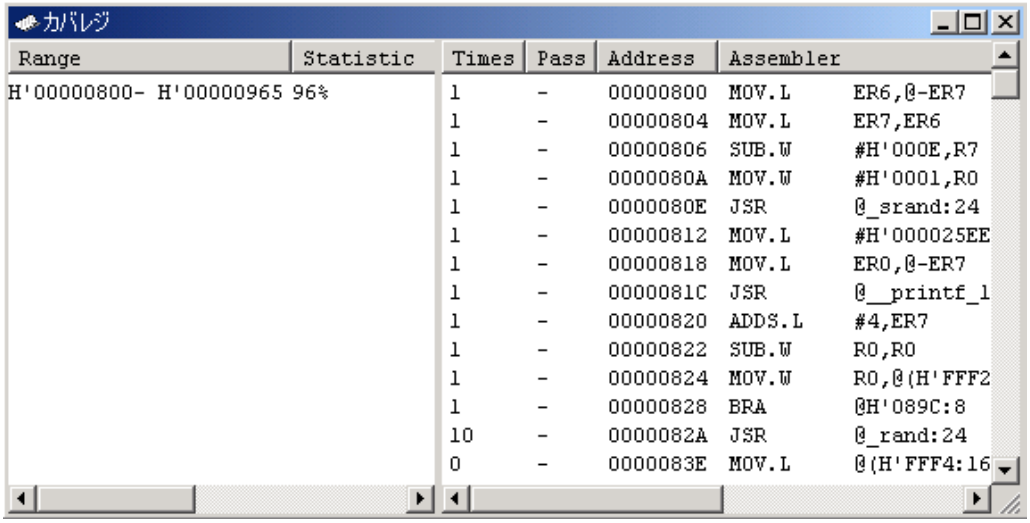


図 4-73 カバレジウィンドウ(アドレス指定)

本ウィンドウはスプリッタで2分割されています。

- **左側**

  カバレジ範囲とカバレジ統計情報を表示します。
  表示する項目は以下の通りです。

[Range]	アドレス範囲
[Statistic]	C0 カバレジ値(パーセント表示)
[Status]	各カバレジ範囲の有効/無効状態

- **右側**

C/C++およびアセンブラレベルでのカバレジ情報を表示します。

表示する項目は以下の通りです。

[Times]	命令を実行した回数
[Pass]	条件分岐命令の実行条件 T: 条件成立で分岐した F: 条件不成立で分岐しなかった
[Address]	命令アドレス
[Assembler]	逆アセンブル表示
[Source]	C/C++またはアセンブラソース

[カバレジ]ウィンドウをクローズすると取得したカバレジ情報と取得条件の設定をクリアします。

(2) カバレジウィンドウ（ソースファイル指定）

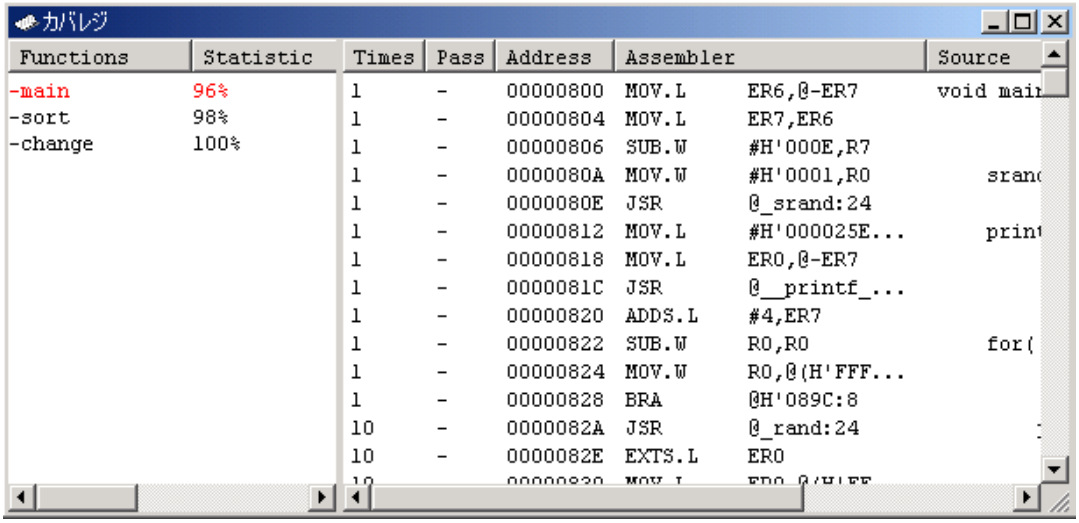


図 4-74 カバレジウィンドウ(ソースファイル指定)

本ウィンドウはスプリッタで2分割されています。

- **左側**

  カバレジ範囲とカバレジ統計情報を表示します。
  表示する項目は以下の通りです。

  [Functions] カバレジ対象の関数
  [Statistic] C0 カバレジ値(パーセント表示)
  [Status] 各関数の有効/無効状態

  - カラムタブをクリックすることで関数名または C0 カバレジ値の降順または昇順に並べ替えることができます。
  - 関数を選択して、ポップアップメニューから[Enable]を選択すると、その関数を有効または無効にできます。
  - ポップアップメニューから[Enable All]を選択すると、すべての関数を有効または無効にできます。
  - ポップアップメニューから[Clear All]を選択すると、すべての関数のカバレジデータをクリアすることができます。

- **右側**

  左側のウィンドウでクリックにより選択された関数について、C/C++およびアセンブラレベルでのカバレジ情報を表示します。

表示する項目は以下の通りです。

[Times] 命令を実行した回数

[Pass]	条件分岐命令の実行条件 T: 条件成立で分岐した F: 条件不成立で分岐しなかった
[Address]	命令アドレス
[Assembler]	逆アセンブル表示
[Source]	C/C++ソース

[カバレジ]ウィンドウをクローズすると取得したカバレジ情報と取得条件の設定をクリアします。

4.18.2 カバレジ情報を取得する

ポップアップメニューで[有効]をチェック後、命令を実行するとカバレジ情報を取得します。

4.18.3 ソースファイルを表示する

ポップアップメニューから[ソースファイル表示]を選択すると、[エディタ]ウィンドウを開いて、[カバレジ]ウィンドウ上のカーソル位置のアドレスに対応するソースファイルを表示します。

4.18.4 表示アドレスを変更する

ポップアップメニューから[表示アドレス...]を選択すると、[表示アドレス]ダイアログボックスを表示します。

図 4-75 表示アドレスダイアログボックス

本ダイアログボックスで[カバレジ]ウィンドウの表示アドレスを変更します。

4.18.5 カバレジ測定範囲を変更する

(1) カバレジ測定範囲をアドレスで指定した場合

ポップアップメニューで[測定範囲設定...]を選択すると、[カバレジ測定範囲]ダイアログボックスを表示します。

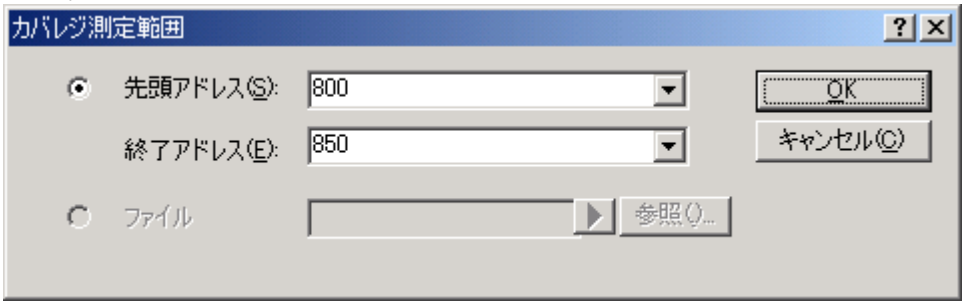


図 4-76 カバレジ測定範囲ダイアログボックス(アドレス指定)

本ダイアログボックスで命令実行情報取得の条件を変更します。下記項目を指定できます。

[先頭アドレス]　　先頭アドレス（接頭辞省略時は 16 進で入力）
[終了アドレス]　　終了アドレス（接頭辞省略時は 16 進で入力）

[OK]ボタンをクリックするとカバレジ測定範囲を変更します。

(2)　**カバレジ測定範囲をソースファイルで指定した場合**

ポップアップメニューで[測定範囲設定...]を選択すると、[カバレジ測定範囲]ダイアログボックスを表示します。

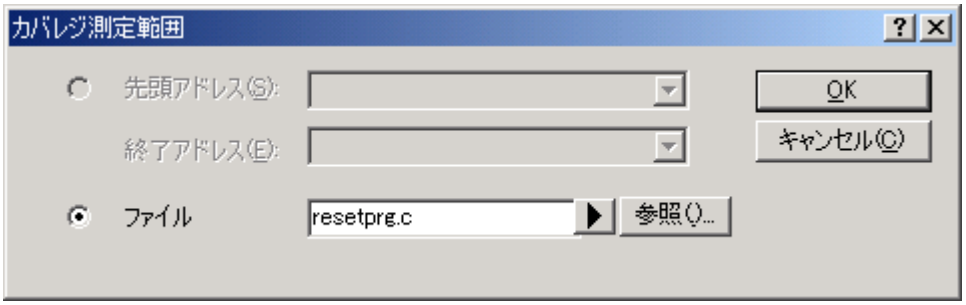


図 4-77　カバレジ測定範囲ダイアログボックス(ソースファイル指定)

本ダイアログボックスで命令実行情報取得の条件を変更します。下記項目を指定できます。

[ファイル]　　現在のプロジェクト中の".C"または".CPP"を型名にもつソースファイルを指定します。
これにより指定ファイル内に存在する関数をカバレジ範囲として設定します。
ファイル型名を省略した場合は、".C"を補います。".C"または".CPP"以外のファイル型名を持つファイルは指定できません。
プレースホルダまたは[参照...]ボタンが利用できます。

[OK]ボタンをクリックするとカバレジ測定範囲を変更します。

4.18.6　カバレジ情報をクリアする

ポップアップメニューから[クリア]を選択すると、取得したカバレジ情報をクリアします。

4.18.7　カバレジ情報をファイルに保存する

ポップアップメニューから[保存...]を選択すると、カバレジ情報をファイルに保存するための[カバレジ情報を保存]ダイアログボックスを表示します。

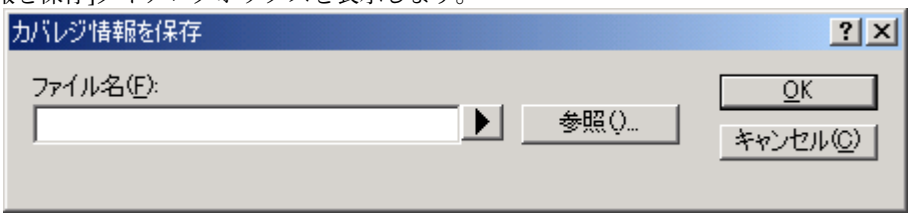


図 4-78　カバレジ情報を保存ダイアログボックス

保存するカバレジ情報ファイルの場所と名前を指定します。プレースホルダまたは[参照…]ボタンが使用できます。

ファイル拡張子の入力を省略すると、ファイル拡張子として".COV"を自動的に付加します。
ファイル拡張子として、".COV"または".TXT"以外を入力するとエラーメッセージを出力します。

4.18.8 カバレジ情報をファイルからロードする

ポップアップメニューから[ロード...]を選択すると、カバレジ情報をファイルからロードするための[カバレジ情報ロード]ダイアログボックスを表示します。

図 4-79 カバレジ情報ロードダイアログボックス

ロードするカバレジ情報ファイルの場所と名前を指定します。プレースホルダまたは[参照…]ボタンが使用できます。
ロードできるファイル拡張子は".COV"のみです。その他のファイル拡張子を入力するとエラーメッセージを出力します。

4.18.9 最新の情報に更新する

ポップアップメニューから[最新の情報に更新]を選択すると、[カバレジ]ウィンドウ内容を最新に更新します。

4.18.10 情報の更新を抑止する

ポップアップメニューの[表示固定]をチェックすると、プログラム実行停止時に[Times]および[Pass]のみ更新します。
これにより、[カバレジ]ウィンドウの命令コード部分更新のためのメモリアクセスを抑止することができます。

4.18.11 Confirmation Request ダイアログボックス

[クリア]、[測定範囲設定...]をクリックするか、[カバレジ]ウィンドウを閉じようとすると、確認のダイアログボックスを表示します。

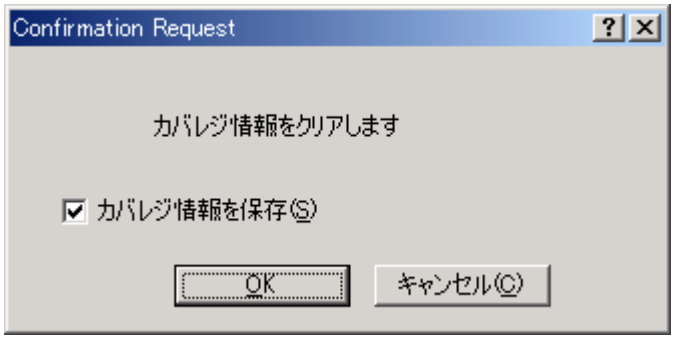


図 4-80 Confirmation Request ダイアログボックス

[OK]ボタンをクリックすると、カバレジデータをクリアします。
[カバレジ情報を保存]をチェックすると、「図 4-78」に示す[カバレジ情報を保存]ダイログボックスを表示し、クリアする前にカバレジデータをファイルに保存できます。

4.18.12 カバレジ情報を保存ダイアログボックス

[ファイル->セッションの保存]メニューオプションをクリックすると、[カバレジ情報を保存]ダイログボックスを表示します。[カバレジ]ウィンドウのデータを別々またはまとめて保存することができます。

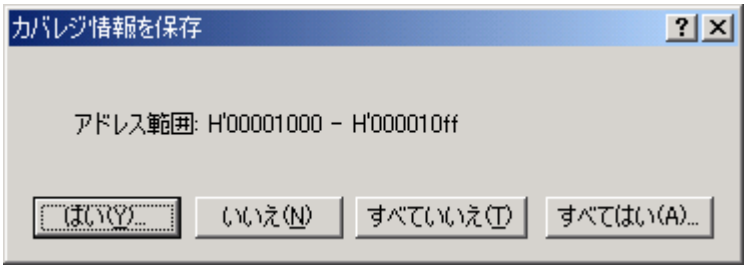


図 4-81 カバレジ情報を保存ダイアログボックス

- [カバレジ]ウィンドウが複数開いているとウィンドウ数分の[カバレジ情報を保存]ダイアログボックスが開きます。
- [すべていいえ]ボタンをクリックすると、すべてのカバレジ情報を保存しないでダイアログボックスを閉じます。
- [すべてはい]ボタンをクリックすると、すべての[カバレジ]ウィンドウデータを 1 個のファイルに保存します。

4.18.13 エディタウィンドウへのカバレジ結果表示

命令実行済のソース行に対応するカバレジカラムを強調表示することで[エディタ]ウィンドウにもカバレジ結果を表示します。[カバレジ]ウィンドウでカバレジに関する設定を変更すると、対応するカバレジカラムの表示も更新します。

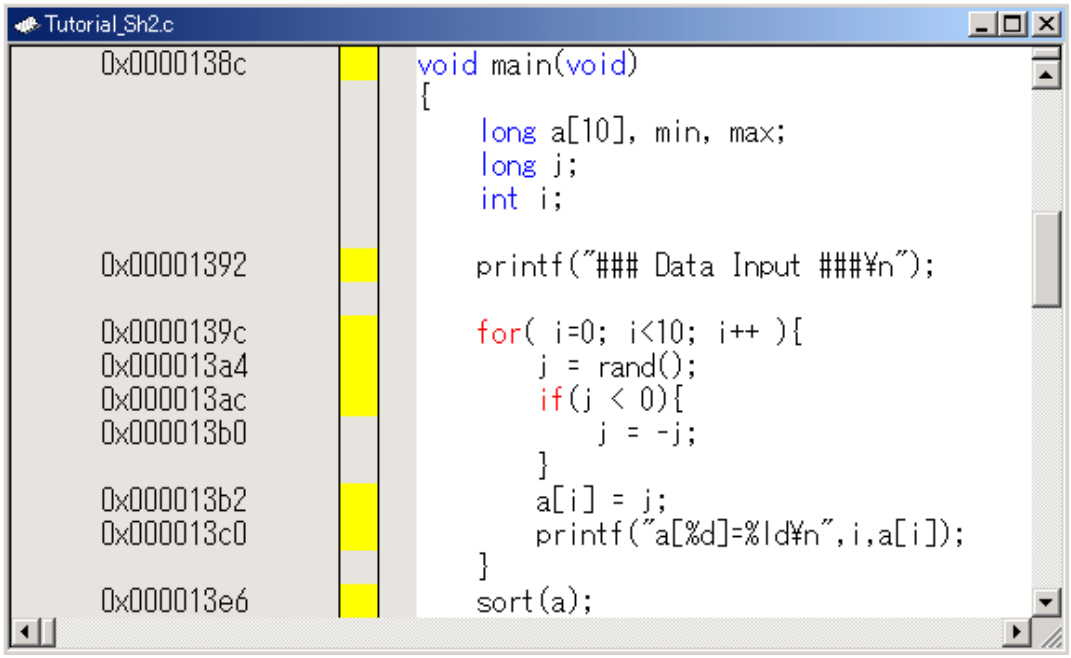


図 4-82 デバッガカラム（カバレジ）

4.19 現在の状態を表示する

デバッグプラットフォームの現在の状態を知るには[ステイタス]ウィンドウを表示します。

[ステイタス]ウィンドウを開くには、[表示->CPU->ステイタス]を選択するか、[ステイタスの表示]ツールバーボタン をクリックします。

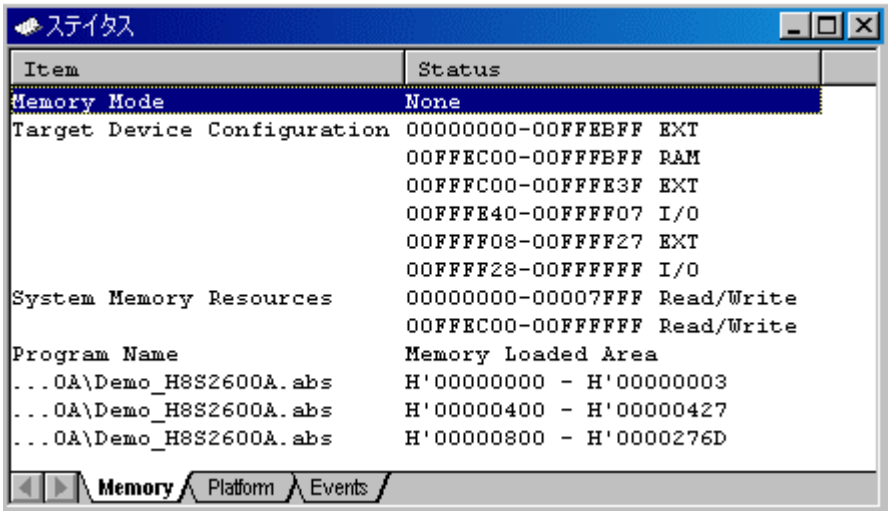


図 4-83 ステイタスウィンドウ

[ステイタス]ウィンドウには、3 枚のシートがあります。

- [Memory]シート
  メモリマッピングおよび現在ロードしたオブジェクト・ファイルが使用するメモリエリアなど、現在のメモリステイタスに関する情報を含んでいます。
- [Platform]シート
  CPU種別および動作モードなど、デバッグプラットフォームのステータス情報、実行状態および実行統計情報を含んでいます。
- [Events]シート
  リソース情報およびブレークポイント等のイベント情報に関する情報を含んでいます。

4.20 コマンドラインインタフェースでデバッグする

ウィンドウメニューやウィンドウコマンドを使用しないで、テキストベースのコマンドを入力してデバッグするには、[コマンドライン]ウィンドウを使用します。

4.20.1 コマンドラインウィンドウを開く

[コマンドライン]ウィンドウを開くには、[表示->コマンドライン]を選択するか、[コマンドライン]ツールバーボタン をクリックします。

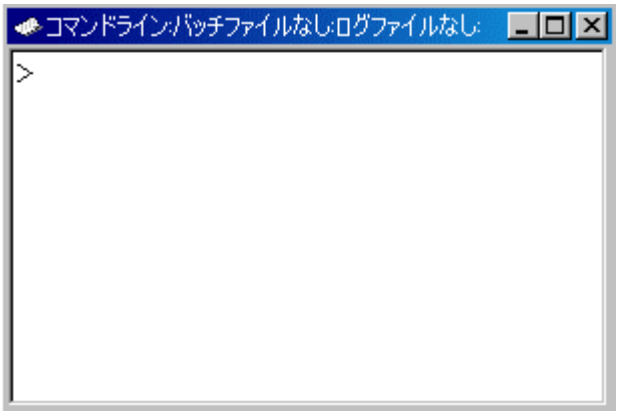


図 4-84 コマンドラインウィンドウ

テキストベースのコマンドを入力してデバッグプラットフォームを制御できるウィンドウです。コマンド行をファイルから呼び出して実行すること、および出力結果をファイルに記録することができます。

[コマンドライン]ウィンドウ最終行のコマンドプロンプト(‘>’)に続けて入力後、’Enter’キーを押すとコマンドを実行します。使用できるコマンドについては、「5　コマンドライン」およびオンラインヘルプを参照してください。

ウィンドウタイトルとしてバッチファイル名とログファイル名をコロンで区切って表示します。

最終行で”Ctrl+↑”または”Ctrl+↓”を押すと過去に実行したコマンド行を呼び出すことができます。

本ウィンドウで、HEW コマンドおよび、TCL コマンドが入力できます。

4.20.2 コマンドファイルを設定する

あらかじめ定義した一連のコマンド行を実行する場合は、コマンドファイルを利用すると便利です。

コマンドファイルはテキストエディタで作成し、実行すべきコマンドラインを記述しておきます。コマンドファイル型名のデフォルトは".hdc"です。

コマンドファイルを設定するには、[バッチファイルを指定]ダイアログボックスを使用します。

ポップアップメニューから[バッチファイル指定...]を選択すると、[バッチファイルを指定]ダイアログボックスを表示します。コマンドファイル名(*.hdc)を入力できます。 [OK]ボタンをクリックすると、設定したコマンドファイル名を、ウィンドウタイトルに表示します。[キャンセル]ボタンをクリックすると、設定を変更しないでダイアログボックスを閉じます。

図 4-85 バッチファイルを指定ダイアログボックス

4.20.3 コマンドファイルを実行する

コマンドファイルを実行するには、[バッチファイルを指定]ダイアログボックスで[実行]ボタンをクリックするか、ポップアップメニューから[バッチファイルの実行を開始]を選択します。[バッチファイルの実行を開始]メニューはコマンドファイル実行中にグレー表示となり、コマンドファイル実行が停止してユーザに制御が戻ったときに有効表示になります。

4.20.4 コマンド実行を中断する

コマンド実行を中断する場合は、ポップアップメニューから [バッチファイル停止]を選択します。[バッチファイル停止]メニューはコマンド実行中に有効表示になります。

4.20.5 ログファイルを設定する

コマンド実行結果を保存するログファイルは[ログファイルを開く]ダイアログボックスで設定します。

[ログファイルを開く]ダイアログボックスを開くには、ポップアップメニューから[ログファイル指定...]を選択します。

図 4-86 ログファイルを開くダイアログボックス

出力結果を記録するログファイル(*.log)の名前を入力します。ロギングオプションを自動的に設定し、ログファイル名をウィンドウのタイトルバーに表示します。

既に存在しているログファイル名を指定すると、ログを追加するか、以前のログを消去して、新しいログを上書きするかを確認します。

4.20.6 ログファイルへの出力を開始/停止する

ファイルへのロギング処理を実行するか、停止するかの切り替えは、ポップアップメニューの[ロギング開始/停止]で行います。ログファイルの内容は、ロギングが終了するか、チェックボックスをクリアしてロギングを一時的に停止しなければ表示できないことにご注意ください。ロギングを再び開始すると、ログファイルに追加します。

4.20.7 ファイルの full パスを入力する

カレントディレクトリは移動する可能性があるため、[コマンドライン]ウィンドウではファイル名をfullパスで指定することをお勧めします。しかし、fullパスのファイル名をキー入力するのは煩雑であるため、ファイルをブラウズして選択するだけでfullパスファイル名を入力できる機能をサポートしています。

ポップアップメニューで[参照...]を選択すると、[Browse]ダイアログボックスを表示します。ここでファイルを選択し[開く]をクリックすると、fullパスファイル名をカーソル位置に貼り付けます。カーソルが最終行にある場合のみ使用できます。

4.20.8 プレースホルダを入力する

ポップアップメニューから[プレースホルダ]のサブメニューで選択したプレースホルダをカーソル位置に貼り付けます。カーソルが最終行にある場合のみ使用できます。

4.21 手動で擬似割込みを発生させる

ウィンドウ上のボタンを押下して手動で擬似割込みを発生させるには、[トリガ]ウィンドウを使用します。

[トリガ]ウィンドウを開くには、[トリガ->CPU->トリガ]を選択するか、[トリガ]ツールバーボタン  をクリックします。

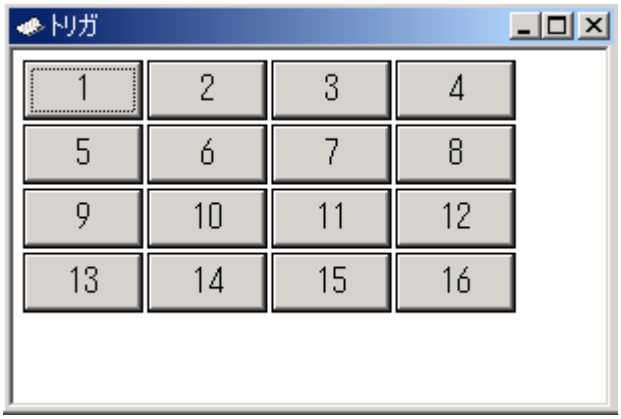


図 4-87　**トリガウィンドウ**

本ウィンドウは、手動で擬似割込みを発生させるためのトリガボタンを表示します。トリガボタン押下時に発生する擬似割込みの内容は、[トリガ設定]ダイアログボックスで設定します。

トリガボタンは最大 256 個まで設定できます。

シミュレータ・デバッガの擬似割込み処理については、**「2.16　擬似割込み」**を参照してください。

4.21.1　トリガボタンを設定する

各トリガボタン押下時に発生する擬似割込みの内容を設定するには[トリガ設定]ダイアログボックスを使用します。

[トリガ設定]ダイアログボックスを開くには、ポップアップメニューの[設定...]を選択します。

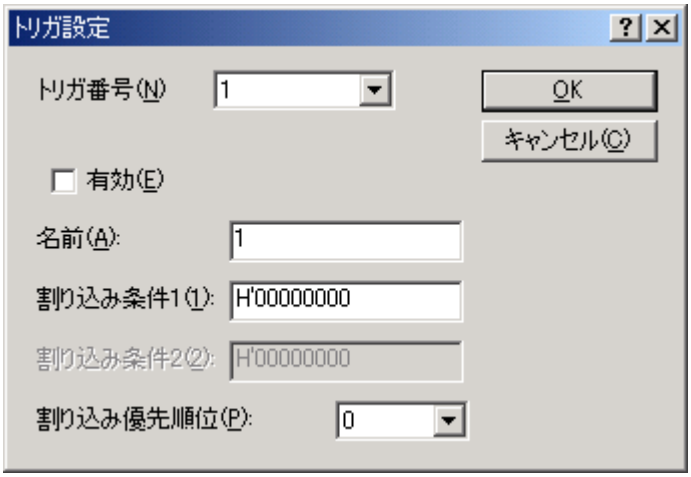


図 4-88　**トリガ設定ダイアログボックス**

本ダイアログボックスでは、トリガボタン押下時に発生する割込みの内容を設定します。

[トリガ番号]	設定するトリガボタンを選択します。
[名前]	[トリガ]ウィンドウに表示するトリガボタンの名前を指定します。
[有効]	チェックするとトリガボタンが有効になります。
[割り込み条件 1]	割込みベクタ番号を指定します。（0~H'FF）
[割り込み優先順位]	割込み優先順位を指定します。（接頭辞省略時は 16 進入力、16 進表示）（0～H'11） 割込みを受付けるかどうかの判定は、選択されたデバッグプラットフォームの CPU の仕様に従います。ただし、優先順位に H'8 以上を指定した場合は、常に割込みを受付けます。

指定した内容は、[OK]ボタンをクリックすることにより設定します。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

【注】 複数のトリガボタンの設定を変更した後に[キャンセル]ボタンをクリックすると、すべてのトリガボタンの設定変更が無効になります。

4.21.2 トリガボタンの数を変える

[トリガ]ウィンドウに表示するトリガボタンの数を変えるには、ポップアップメニューから[ボタンの数]のサブメニューで[4]、[16]、[64]、[256]を選択します。

4.21.3 トリガボタンのサイズを変える

[トリガ]ウィンドウに表示するトリガボタンのサイズを変えるには、ポップアップメニューから[サイズ]のサブメニューで[大]、[中]、[小]を選択します。

4.22 シミュレータ・デバッガの設定を変更する

本節ではシミュレータ・デバッガ起動後にシミュレータシステム、メモリマップ、およびメモリリソースを変更する方法について説明します。

4.22.1 シミュレータシステムの設定を変更する

システムコールの開始位置、実行モード、浮動小数点の丸めモード等の設定変更は、[シミュレータシステム]ダイアログボックスの[システム]タブで行います。

[シミュレータシステム]ダイアログボックス[システム]タブを開くには、[オプション->シミュレータ->システム...]を選択するか、[シミュレータシステム]ツールバーボタン  をクリックします。

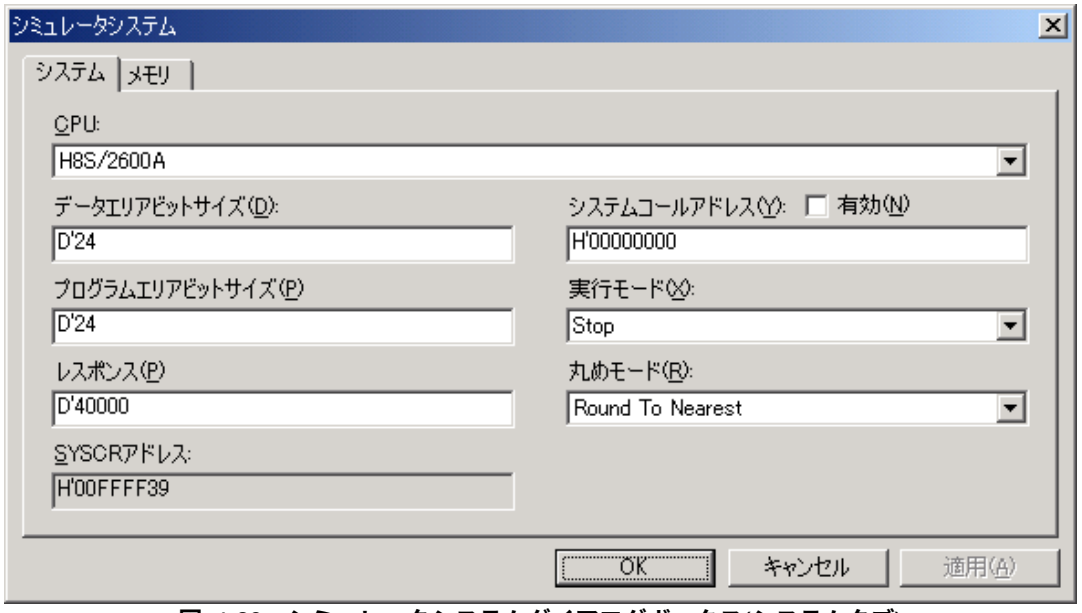


図 4-89 シミュレータシステムダイアログボックス(システムタブ)

本ダイアログボックスでは下記項目を設定または表示します。

[CPU] 現在設定している CPU。
（CPU は、[デバッグの設定]ダイアログボックスにより設定します。）

[データエリアビットサイズ] アドレス空間のビット数を指定します。各 CPU で指定できる値は以下の通りです。

H8/300、H8/300L、H8/300HN、H8S/2600N、H8S/2000N：	16
H8/300HA：	17～24
H8S/2600A、H8S/2000A：	17～32

[プログラムエリアビットサイズ] プログラム領域のビット数を指定します。各 CPU で指定できる値は以下の通りです。

H8/300、H8/300L、H8/300HN、H8S/2600N、H8S/2000N：	16
H8/300HA：	アドレス空間ビット数と同一
H8S/2600A、H8S/2000A：	17～24

[SYSCR アドレス] SYSCR レジスタのアドレスを指定します。

[Simulated I/O システムコールアドレス] ユーザプログラムから標準入出力またはファイル入出力を行うためのシステムコールの開始位置を指定します。

[有効] チェックするとシステムコールが有効となります。

[レスポンス] 何命令ごとにウィンドウをリフレッシュするかを指定します。
(1~D’2,147,483,647、デフォルトは D’40000)

[実行モード] シミュレーションエラーが発生した場合のシミュレータ・デバッガ動作を規定します。
割り込みはシミュレーションエラーに含み、本設定に従います。
また、[Detail]ボタンで、割り込み発生時の動作を規定することもできます。

[停止] シミュレーションを停止します。

[続行] シミュレーションを続行します。

[丸めモード]　浮動小数点数 10 進->2 進変換で発生する丸めのモード、および非正規化数の扱いを指定します。
- [最近値丸め]　最近値丸め。非正規化数は 0 に変換しない。
- [ゼロ丸め]　ゼロ方向丸め。非正規化数は 0 に変換する。

変更内容は、[OK]ボタンまたは[適用]ボタンをクリックすることにより設定します。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

4.22.2 メモリマップおよびメモリリソースの設定を変更する

メモリマップの設定および変更とメモリリソースの設定および変更は、[シミュレータシステム]ダイアログボックスの[メモリ]タブで行います。

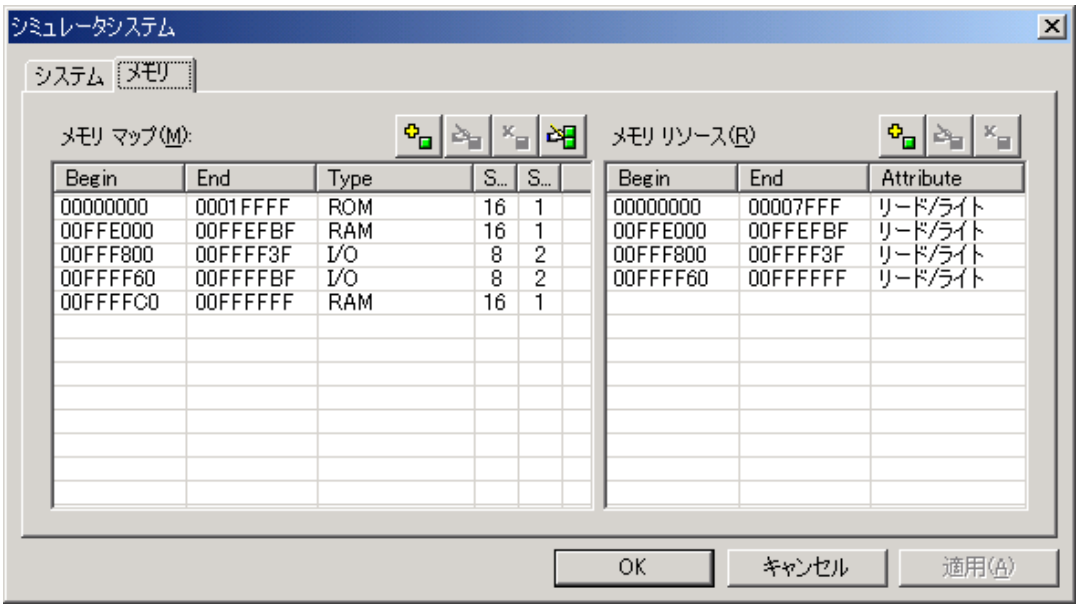


図 4-90　シミュレータシステムダイアログボックス(メモリタブ)

本ダイアログボックスでは以下の項目を設定します。

[メモリマップ]　メモリ情報として、メモリ種別とその先頭アドレス・終了アドレス、データバス幅、アクセスサイクル数を表示します。

[メモリリソース]　現在設定しているメモリリソースのアクセス種別とその先頭アドレス・終了アドレスを表示します。

[メモリリソース]は次の各ボタンにより追加・変更・削除することができます。

[メモリリソース]の項目を追加します。クリックすることにより、[メモリリソース設定]ダイアログボックスが開き、追加することができます。

[メモリリソース]の項目を変更します。変更したい項目をリストボックス上で選択後、ボタンをクリックします。クリックすることにより、[メモリリソース設定]ダイアログボックスが開き、変更することができます。

[メモリリソース]の項目を削除します。削除したい項目をリストボックス上で選択後、ボタンをクリックします。

[メモリリソース]は、[H8S,H8/300 Standard Toolchain]ダイアログの[Simulator]シートの[Memory Resource]と同一の設定情報です。おたがいの変更が反映されます。[H8S,H8/300 Standard Toolchain]ダイアログについては、「3.3.1　メモリマップ」を参照ください。

[メモリマップ]は、以下の各ボタンにより追加・変更・削除ができます。

[メモリマップ]の項目を追加します。クリックすると、[メモリマップ設定]ダイアログボックス(**図** 4-91 参照)が開き、追加することができます。

[メモリマップ]の項目を変更します。変更したい項目をリストボックス上で選択後、ボタンをクリックします。クリックすると、[メモリマップ設定]ダイアログボックス(**図** 4-91 参照)が開き、変更することができます。

[メモリマップ]の項目を削除します。削除したい項目をリストボックス上で選択後、ボタンをクリックします。

なお、[メモリマップ]・[メモリリソース]は、  ボタンによりデフォルト値にリセットすることができます。

変更内容は、[OK]ボタンまたは[適用]ボタンをクリックすることにより設定します。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

4.22.3　メモリマップ設定ダイアログボックス

[メモリマップ設定]ダイアログボックスでは、対象 CPU のメモリマップを設定します。

各項目に表示する内容は、対象 CPU によって異なります。シミュレータ・デバッガはこれらの値をメモリアクセスサイクル数の算出に使用します。

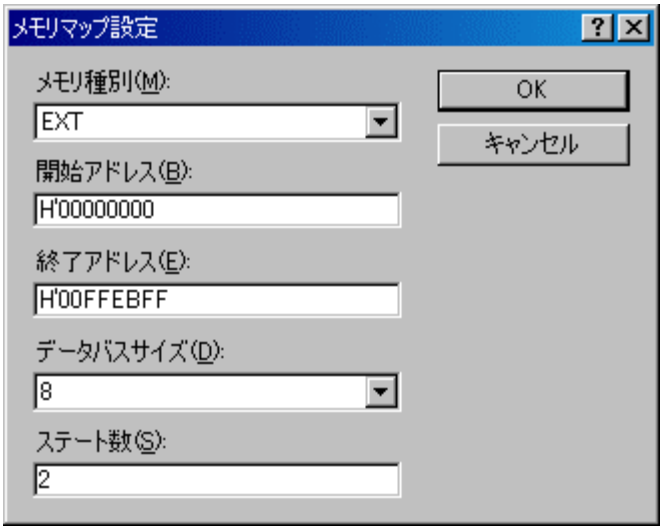


図 4-91　**メモリマップ設定ダイアログボックス**

本ダイアログボックスでは以下の項目を設定します。

[メモリ種別]	メモリ種別	
	[ROM]	**内蔵** ROM
	[RAM]	**内蔵** RAM
	[EXT]	**外部メモリ**
	[IO]	**内蔵** I/O
	[EEPROM]	EEPROM
[開始アドレス]	メモリ種別に対応するメモリの先頭アドレス	
[終了アドレス]	メモリ種別に対応するメモリの終了アドレス	
[データバスサイズ]	メモリのデータバス幅	
[リードサイクル数]	メモリのリードサイクル数	
[ライトサイクル数]	メモリのライトサイクル数	

変更内容は、[OK]ボタンをクリックすることにより設定します。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

【注】 メモリリソースを確保している領域のメモリマップは、削除・変更することができません。あらかじめ、[シミュレータシステム]ダイアログボックスの[メモリ]タブによりメモリリソースを削除してから、メモリマップを削除・変更してください。

4.22.4 メモリリソース設定ダイアログボックス

[メモリリソース設定]ダイアログボックスでは、メモリリソースの設定・変更を行います。

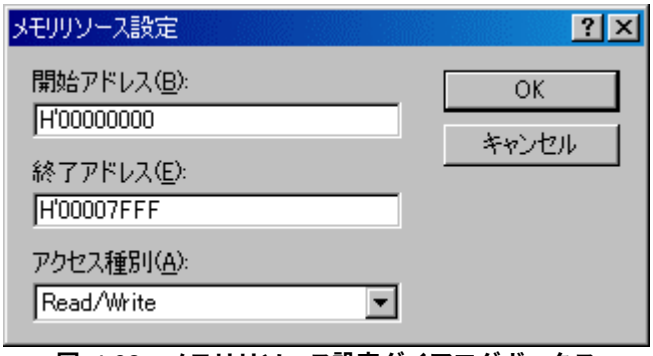


図 4-92 メモリリソース設定ダイアログボックス

本ダイアログボックスでは以下の項目を設定します。

[開始アドレス]	確保するメモリ領域の先頭アドレス	
[終了アドレス]	確保するメモリ領域の終了アドレス	
[アクセス種別]	アクセス種別	
	[Read]	読み出しのみ可能
	[Write]	書き込みのみ可能
	[Read/Write]	読み書き可能

各項目を指定後、[OK]ボタンをクリックすることによりメモリリソースの設定・変更を行います。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

注：1.メモリリソースの設定すると、PCのメモリを使用します。したがって、メモリリソースを大きく取りすぎると、PCの動作が極端に遅くなる場合があります。

4.23 標準入出力およびファイル入出力を行う

ユーザプログラムから標準入出力およびファイル入出力のシステムコールを行うには、[I/O シミュレーション]ウィンドウを利用します。

4.23.1 I/O シミュレーションウィンドウを開く

[I/O シミュレーション]ウィンドウを開くには、[表示->CPU->I/O シミュレーション]を選択するか、[I/O シミュレーション]ツールバーボタン をクリックします。

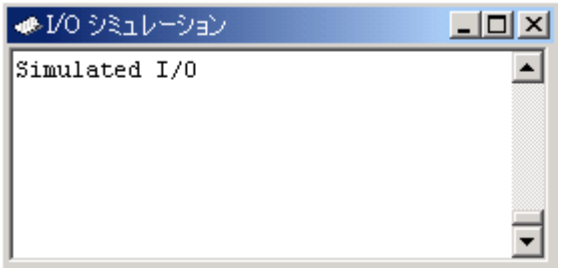


図 4-93 I/O シミュレーションウィンドウ

ユーザプログラムからの標準出力は本ウィンドウに出力されます。本ウィンドウ上でのキー入力がユーザプログラムへの標準入力となります。

4.23.2 入出力機能

サポートしている入出力処理は以下の通りです。機能コードには、16 ビットアドレス版、24 ビットアドレス版、32 ビットアドレス版があります。使用する CPU に合わせて選択してください。

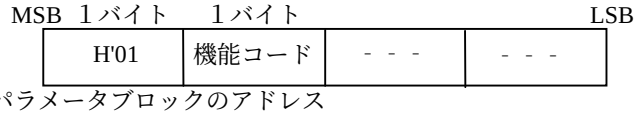
表4-1 入出力機能一覧

番号	機能コード	機能名	内 容
1	H'01 (16 ビットアドレス) H'11 (24 ビットアドレス) H'21 (32 ビットアドレス)	GETC	標準入力からの１バイト入力
2	H'02 (16 ビットアドレス) H'12 (24 ビットアドレス) H'22 (32 ビットアドレス)	PUTC	標準出力への１バイト出力
3	H'03 (16 ビットアドレス) H'13 (24 ビットアドレス) H'23 (32 ビットアドレス)	GETS	標準入力からの１行入力
4	H'04 (16 ビットアドレス) H'14 (24 ビットアドレス) H'24 (32 ビットアドレス)	PUTS	標準出力への１行出力
5	H'05 (16 ビットアドレス) H'15 (24 ビットアドレス) H'25 (32 ビットアドレス)	FOPEN	ファイルのオープン
6	H'06	FCLOSE	ファイルのクローズ
7	H'07 (16 ビットアドレス) H'17 (24 ビットアドレス) H'27 (32 ビットアドレス)	FGETC	ファイルからの１バイト入力
8	H'08 (16 ビットアドレス) H'18 (24 ビットアドレス) H'28 (32 ビットアドレス)	FPUTC	ファイルへの１バイト出力
9	H'09 (16 ビットアドレス) H'19 (24 ビットアドレス) H'29 (32 ビットアドレス)	FGETS	ファイルからの１行入力
10	H'0A (16 ビットアドレス) H'1A (24 ビットアドレス) H'2A (32 ビットアドレス)	FPUTS	ファイルへの１行出力
11	H'0B	FEOF	エンドオブファイルのチェック
12	H'0C	FSEEK	ファイルポインタの移動
13	H'0D	FTELL	ファイルポインタの現在位置を得る

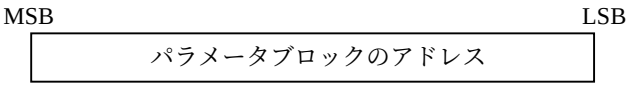
この機能を実現するためには、まず入出力用の特定の位置を [シミュレータシステム]ダイアログボックスの [システムコールアドレス]で指定し、[有効]をチェック後、ユーザプログラムを実行します。シミュレータ・デバッガでは、ユーザプログラムの命令を実行中に、指定した位置へのサブルーチン分岐命令（BSR、JSR）すなわちシステムコール命令を検出すると、R0、R1（H8/300、H8/300L シリーズ）または ER1（H8/300H、H8S シリーズ）の内容をパラメータとして入出力処理を行います。

したがって、システムコールを行う前にユーザプログラムの中で次の設定をしておきます。

・R0 レジスタ：表 4-1に示す機能コード



・R1 レジスタ：パラメータブロックのアドレス
（パラメータブロックの内容は各機能の説明を参照してください。）



・パラメータブロックおよび入出力バッファ領域の確保

なお、パラメータブロックの各パラメータにアクセスする場合は、該当するパラメータのサイズでアクセスしてください。
入出力処理が終了すると、システムコール命令の次の命令からシミュレーションを再開します。

各入出力機能を図 4-94の形式で説明します。

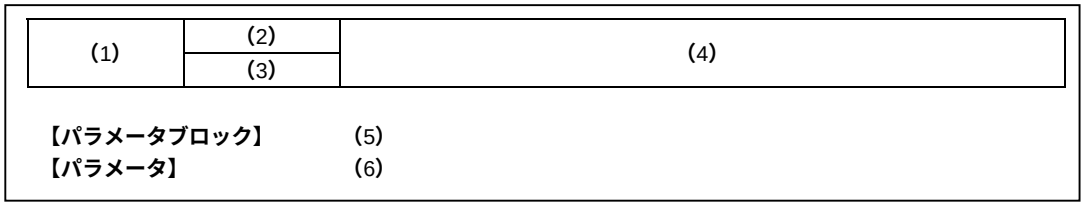


図 4-94 入出力機能の説明形式

各項目の内容は、以下の通りです。
（1） 表4-1に対応する番号
（2） 機能名
（3） 機能コード
（4） 入出力の機能
（5） 入出力のパラメータブロック
（6） 入出力のパラメータ

1	GETC H'01、H'11、H'21	標準入力からの１バイト入力

【パラメータブロック】

・機能コード：H'01 (16 **ビットアドレス**)

	１バイト	**１バイト**
+0	**入力バッファ先頭アドレス**	

・機能コード：H'11 (24 **ビットアドレス**)、H'21 (32 **ビットアドレス**)

	１バイト	**１バイト**
+0 +2	**入力バッファ先頭アドレス**	

【パラメータ】

入力バッファ先頭アドレス（入力）
入力データを書き込むバッファの先頭アドレス

2	PUTC H'02、H'12、H'22	標準出力への１バイト出力

【パラメータブロック】

・機能コード：H'02 (16 **ビットアドレス**)

	１バイト	**１バイト**
+0	**出力バッファ先頭アドレス**	

・機能コード：H'12 (24 **ビットアドレス**)、H'22 (32 **ビットアドレス**)

	１バイト	**１バイト**
+0 +2	**出力バッファ先頭アドレス**	

【パラメータ】

出力バッファ先頭アドレス（入力）
出力データを格納しているバッファの先頭アドレス

3	GETS H'03、H'13、H'23	標準入力からの１行入力

【パラメータブロック】

・機能コード：H'03 (16 **ビットアドレス**)

	１バイト	**１バイト**
+0	**入力バッファ先頭アドレス**	

・機能コード：H'13 (24 **ビットアドレス**)、H'23 (32 **ビットアドレス**)

	１バイト	**１バイト**
+0 +2	**入力バッファ先頭アドレス**	

【パラメータ】

入力バッファ先頭アドレス（入力）
入力データを書き込むバッファの先頭アドレス

4	PUTS H'04、H'14、H'24	標準出力への１行出力

【パラメータブロック】　**・機能コード：**H'04 (16 **ビットアドレス**)

	１バイト	**１バイト**
+0	**出力バッファ先頭アドレス**	

・機能コード：H'14 (24 **ビットアドレス**)、H'24 (32 **ビットアドレス**)

	１バイト	**１バイト**
+0 +2	**出力バッファ先頭アドレス**	

【パラメータ】　**出力バッファ先頭アドレス（入力）**

出力データを格納しているバッファの先頭アドレス

5	FOPEN H'05、H'15、H'25	ファイルのオープン

[FOPEN]によってファイルをオープンすると、ファイル番号を返します。以後のファイル入出力、ファイルクローズ等ではこのファイル番号を用います。同時にオープンできる最大ファイル数は256です。

【パラメータブロック】

・機能コード：H'05 (16 **ビットアドレス**)

	1バイト	**1バイト**
+0	**実行結果**	**ファイル番号**
+2	**オープンモード**	**未使用**
+4	**ファイル名先頭アドレス**	

・機能コード：H'15 (24 **ビットアドレス**)、H'25 (32 **ビットアドレス**)

	1バイト	**1バイト**
+0	**実行結果**	**ファイル番号**
+2	**オープンモード**	**未使用**
+4 +6	**ファイル名先頭アドレス**	

【パラメータ】

実行結果（出力）

0 **正常終了**

−1 **エラー**

ファイル番号（出力）

オープン処理以降のファイルアクセスで使用する番号

オープンモード（入力）

H'00	"r"
H'01	"w"
H'02	"a"
H'03	"r+"
H'04	"w+"
H'05	"a+"
H'10	"rb"
H'11	"wb"
H'12	"ab"
H'13	"r+b"
H'14	"w+b"
H'15	"a+b"

各モードの内容は以下の通りです。

"r"	**読み出し用にオープンする。**
"w"	**空ファイルを書き込み用にオープンする。**
"a"	**ファイルの最後から書き込み用にオープンする。**
"r+"	**読み出し，書き込み用にオープンする。**
"w+"	**空ファイルを読み出し，書き込み用にオープンする。**
"a+"	**読み出し追加用にオープンする。**
"b"	**バイナリモードでオープンする。**

ファイル名先頭アドレス（入力）

ファイル名を格納している領域の先頭アドレス

6	FCLOSE H'06	ファイルのクローズ

【パラメータブロック】

	1バイト	1バイト
+0	実行結果	ファイル番号

【パラメータ】

実行結果（出力）

0 正常終了

–1 エラー

ファイル番号（入力）

ファイルオープン時に返す番号

7	FGETC H'07、H'17、H'27	ファイルから 1 バイトのデータ読み出し

【パラメータブロック】

・機能コード：H'07 (16 **ビットアドレス**)

	1バイト	1バイト
+0	実行結果	ファイル番号
+2	入力バッファ先頭アドレス	

・機能コード：H'17 (24 **ビットアドレス**)、H'27 (32 **ビットアドレス**)

	1バイト	1バイト
+0	実行結果	ファイル番号
+2 +4	入力バッファ先頭アドレス	

【パラメータ】

実行結果（出力）

0 正常終了

–1 EOF 検出

ファイル番号（入力）

ファイルオープン時に返す番号

入力バッファ先頭アドレス（入力）

入力データを書き込むバッファの先頭アドレス

8	FPUTC H'08、H'18、H'28	ファイルへ 1 バイトのデータ書き込み

【パラメータブロック】

・機能コード：H'08 (16 **ビットアドレス**)

	1バイト	**1バイト**
+0	**実行結果**	**ファイル番号**
+2	**出力バッファ先頭アドレス**	

・機能コード：H'18 (24 **ビットアドレス**)、H'28 (32 **ビットアドレス**)

	1バイト	**1バイト**
+0	**実行結果**	**ファイル番号**
+2 +4	**出力バッファ先頭アドレス**	

【パラメータ】

実行結果（出力）

0 **正常終了**

–1 **エラー**

ファイル番号（入力）

ファイルオープン時に返す番号

出力バッファ先頭アドレス（入力）

出力データを格納しているバッファの先頭アドレス

9	FGETS H'09、H'19、H'29	ファイルから文字列データの読み出し

改行コードまたは”NULL”コードを検出するまで，またはバッファサイズに達するまでファイルから文字列データを読み出します。

【パラメータブロック】

・機能コード：H'09 (16 **ビットアドレス**)

	１バイト	**１バイト**
+0	**実行結果**	**ファイル番号**
+2	**バッファサイズ**	
+4	**入力バッファ先頭アドレス**	

・機能コード：H'19 (24 **ビットアドレス**)、H'29 (32 **ビットアドレス**)

	１バイト	**１バイト**
+0	**実行結果**	**ファイル番号**
+2	**バッファサイズ**	
+4 +6	**入力バッファ先頭アドレス**	

【パラメータ】

実行結果（出力）

０ **正常終了**

－１ EOF **検出**

ファイル番号（入力）

ファイルオープン時に返す番号

バッファサイズ（入力）

データを格納する領域のサイズ

（バイト単位で最大 256 **バイトまで）**

入力バッファ先頭アドレス（入力）

入力データを書き込むバッファの先頭アドレス

10	FPUTS H'0A、H'1A、H'2A	ファイルへ文字列データ書き込み

ファイルへ文字列データ書き込みます。文字列終端記号の”NULL”コードはファイルには書き込みません。

【パラメータブロック】

・機能コード：H'0A (16 **ビットアドレス**)

	1バイト	**1バイト**
+0	**実行結果**	**ファイル番号**
+2	**出力バッファ先頭アドレス**	

・機能コード：H'1A (24 **ビットアドレス**)、H'2A (32 **ビットアドレス**)

	1バイト	**1バイト**
+0	**実行結果**	**ファイル番号**
+2 +4	**出力バッファ先頭アドレス**	

【パラメータ】

実行結果（出力）

0 **正常終了**

–1 **エラー**

ファイル番号（入力）

ファイルオープン時に返す番号

出力バッファ先頭アドレス（入力）

出力データを格納しているバッファの先頭アドレス

11	FEOF H'0B	エンドオブファイルのチェック

【パラメータブロック】

	1バイト	**1バイト**
+0	**実行結果**	**ファイル番号**

【パラメータ】

実行結果（出力）

0 EOF **でない**

–1 EOF **検出**

ファイル番号（入力）

ファイルオープン時に返す番号

12	FSEEK H'0C	指定位置にファイルポインタを移動

【パラメータブロック】

	１バイト	１バイト
+0	実行結果	ファイル番号
+2	ディレクション	未使用
+4	オフセット上位ワード	
+6	オフセット下位ワード	

【パラメータ】

実行結果（出力）
- 0　正常終了
- −1　エラー

ファイル番号（入力）
- ファイルオープン時に返す番号

ディレクション（入力）
- 0　オフセットはファイルの先頭からのバイト数
- 1　オフセットは現在のファイルポインタからのバイト数
- 2　オフセットはファイルの最後尾からのバイト数

オフセット（入力）
- ディレクションで指定した位置からのバイト数

13	FTELL H'0D	ファイルポインタの現在位置を調査

【パラメータブロック】

	１バイト	１バイト
+0	実行結果	ファイル番号
+2	オフセット上位ワード	
+4	オフセット下位ワード	

【パラメータ】

実行結果（出力）
- 0　正常終了
- −1　エラー

ファイル番号（入力）
- ファイルオープン時に返す番号

オフセット（出力）
- 現在のファイルポインタの位置（ファイル先頭からのバイト数）

以下に標準入力（キーボード）から１文字入力する例を示します。システムコールアドレスとしてラベル SYS_CALL を指定します。

```
                MOV.W     #H'0101, R0
                MOV.W     #PARM, R1
                JSR       @SYS_CALL
STOP            NOP
SYS_CALL        NOP
PARM            .DATA.W   INBUF
INBUF           .RES.B    2
                .END
```

4.24 複数デバッギングプラットフォームを同期動作させる

HEW では複数のデバッギングプラットフォームを同期動作させることができます。

これを実現するために 2 通りの方法を提供しています。HEW アプリケーションでの同期は HEW2.x から利用可能な方法です。HEW デバッガターゲットの同期は HEW3.0 の複数ターゲットデバッグに適した新しい方法です。

4.24.1 HEW アプリケーションでの同期

複数デバッギングプラットフォームの同期動作を HEW から HEW を起動することで実現しています。ここで起動する側をマスタ HEW、起動された側をスレーブ HEW と呼びます。

スレーブ HEW を起動するには、[ツール->従属 HEW の起動…]を選択するか、[従属 HEW の実行]ツールバーボタン  をクリックします。

スレーブHEWはマスタHEWと同様の操作が可能です。

マスタHEWでの下記動作はスレーブHEWに通知されます。これによりスレーブHEWをマスタHEWと同期させて動作させることができます。

- リセット後実行
- 実行
- プログラムの停止

【注】 マスタ HEW から複数のスレーブ HEW アプリケーションを起動することはできますが、スレーブ HEW アプリケーションのネスト(スレーブ HEW からスレーブ HEW を起動する)はできません。

4.24.2 HEW デバッガターゲットの同期

HEW では、内部の複数ターゲットデバッグもサポートしています。これにより、複数のターゲットコンポーネントを同じ HEW アプリケーション内で接続することが出来ます。これらのターゲットは同時にデバッグすることができます。ユーザは異なるターゲットに複数のセッションを設定することができます。そしてデバッグ時に、一つのセッションのイベントに同期して他のセッションで同じイベントを引き起こすことができます。これは HEW アプリケーションでの同期と同様です。さらに、セッションを切り換えて同じアプリケーション内で何が起きているかを見ることが出来ます。

➲ HEW内デバッガターゲットの同期を設定するには

1. [オプション->デバッグセッション...]メニューを選択します。
2. ダイアログボックスの[同期デバッグ]タブを選択します｛図 4-95 参照)。
3. ワークスペース内で現在利用可能な全てのセッションを表示するので、同期させたいセッションを選択します。
4. [同期セッション]ボタンをクリックします。選択したセッションのアイコンがチェック無しからチェックありに変わります。
5. [同期デバッグ有効化]チェックボックスをチェックして本機能を有効にします。
6. 変更を確認するために[OK]をクリックします。

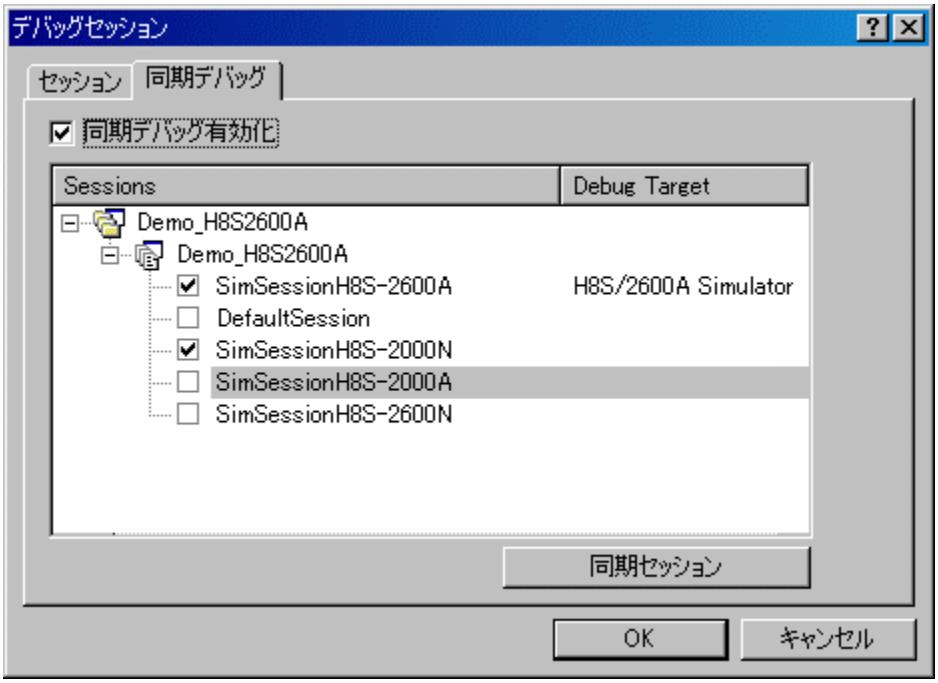


図 4-95 デバッグセッションダイアログボックス

➲ HEW内デバッガターゲットの同期を利用するには

1. 標準ツールバーの[セッション]コンボボックスをクリックします。[Synch. Session]選択肢を選択します(図 4-96参照)。このオプションはシステムに同期デバッグが追加されている場合のみ有効となります。
2. 一度選択するとHEWで同期デバッグ機能が利用可能となります。これにより[シンクロナイズドセッション]という別のツールバーが追加されます(図 4-96 参照)。
3. [シンクロナイズセッション]ツールバー上の有効／無効ツールバーボタンで、設定を変更しないで一時的に同期設定を切り替えることができます。
4. [シンクロナイズセッション]ツールバーでのセッション切替で現在見えるセッションを切り替えることができます。通常のＨＥＷデバッグ状態ではこれはセッションを閉じることを意味しますが、同期セッションでは、複数セッションを開くことができ、現在このツールバーで選択されているセッションを見ることができます。
5. これにより複数のターゲットまたはCPUコアを同時にデバッグすることができます。セッションを変えると画面に表示されるセッションやデータが変わります。

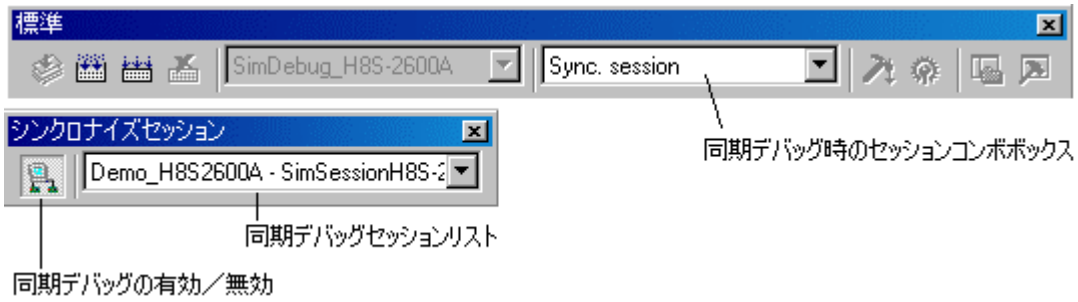


図 4-96 同期デバッグツールバー

【注】 同期機能を有効にすると多くのことが同期して実行できます。以下の表で、2 個の同期したセッションを例に同期デバッグ有効／無効時の動作を示します。

表 4-2 同期デバッグ有効時の動作

ユーザ操作	ターゲットデバッガセッション1	ターゲットデバッガセッション2
いずれかのセッションでの[実行]実行	“実行”	“実行”
いずれかのセッションでの[ステップ]実行	“ステップ”	“ステップ”
いずれかのセッションでの’ESC’押下	“停止”	“停止”
-	ブレークポイントまたはユーザプログラム不正による ”停止”	実行停止（’ESC’押下による停止と同じ結果）
-	実行停止（’ESC’押下による停止と同じ結果）	ブレークポイントまたはユーザプログラム不正による ”停止”
いずれかのセッションでの[CPUリセット]実行	“CPUリセット”	“CPUリセット”

表 4-3 同期デバッグ無効時の動作

ユーザ操作	ターゲットデバッガセッション1	ターゲットデバッガセッション2
セッション1での[実行]実行	“実行”	何もしない。[実行]実行はユーザの入力が必要
セッション2での[実行]実行	何もしない。[実行]実行はユーザの入力が必要	“実行”
セッション1での[ステップ]実行	“ステップ”	何もしない。[ステップ]実行はユーザの入力が必要
セッション2での[ステップ]実行	何もしない。[ステップ]実行はユーザの入力が必要	“ステップ”
セッション1での’ESC’押下	“停止”	ユーザプログラム実行中ならば、実行継続
セッション2での’ESC’押下	ユーザプログラム実行中ならば、実行継続	“停止”
-	ブレークポイントまたはユーザプログラム不正による”停止”	ユーザプログラム実行中ならば、実行継続
-	ユーザプログラム実行中ならば、実行継続	ブレークポイントまたはユーザプログラム不正による”停止”
セッション1での[CPUリセット]実行	“CPUリセット”	何もしない
セッション2での[CPUリセット]実行	何もしない	“CPUリセット”

【注】 標準のデバッグ時とのもう 1 つの相違点は、全ての同期セッションのダウンロードモジュールをワークスペースウィンドウで表示できることです。これにより、任意の使用可能なセッションにモジュールを容易にダウンロードすることができるようになります。

5. コマンドライン

コマンド一覧を表 5-1に示します。

表 5-1 コマンド一覧

項番	コマンド名	短縮形	説 明
1	!	-	コメント
2	ADD_FILE	AF	カレントプロジェクトへのファイル追加
3	ANALYSIS	AN	性能分析機能の有効化／無効化
4	ANALYSIS_RANGE	AR	性能評価関数の設定、表示
5	ANALYSIS_RANGE_DELETE	AD	性能分析範囲の削除
6	ASSEMBLE	AS	アセンブルの実行
7	ASSERT	-	コンディションのチェック
8	AUTO_COMPLETE	AC	オートコンプリート機能の有効化／無効化
9	BREAKPOINT	BP	実行命令位置によるブレークポイントの設定
10	BREAK_ACCESS	BA	メモリ範囲のアクセスによるブレーク条件の設定
11	BREAK_CLEAR	BC	ブレークポイントの削除
12	BREAK_CYCLE	BCY	サイクルによるブレーク条件の設定
13	BREAK_DATA	BD	メモリのデータ値によるブレーク条件の設定
14	BREAK_DISPLAY	BI	ブレークポイント一覧の表示
15	BREAK_ENABLE	BE	ブレークポイントの有効/無効の切替
16	BREAK_REGISTER	BR	レジスタのデータ値によるブレーク条件の設定
17	BREAK_SEQUENCE	BS	実行順序を指定したブレークポイントの設定
18	BUILD	BU	カレントプロジェクトのビルド
19	BUILD_ALL	BL	カレントプロジェクトのすべてのビルド
20	CACHE	-	メモリキャッシュの有効化／無効化
21	CHANGE_CONFIGURATION	CC	コンフィギュレーションの設定
22	CHANGE_PROJECT	CP	プロジェクトの設定
23	CHANGE_SESSION	CS	セッションの設定
24	COVERAGE	CV	カバレジ測定の有効化／無効化
25	COVERAGE_DISPLAY	CVD	カバレジ情報の表示
26	COVERAGE_LOAD	CVL	カバレジ情報のロード
27	COVERAGE_RANGE	CVR	カバレジ範囲の設定
28	COVERAGE_SAVE	CVS	カバレジ情報のセーブ
29	DEFAULT_OBJECT_FORMAT	DO	デフォルトオブジェクト（プログラム）フォーマットの設定
30	DISASSEMBLE	DA	逆アセンブル表示
31	ERASE	ER	[コマンドライン]ウィンドウの内容のクリア
32	EVALUATE	EV	式の計算
33	EXEC_MODE	EM	実行モードの設定、表示
34	EXEC_STOP_SET	ESS	割込み発生時の実行モードの設定、表示
35	FILE_LOAD	FL	オブジェクト(プログラム)ファイルのロード

項番	コマンド名	短縮形	説　明
36	FILE_SAVE	FS	メモリ内容のファイルセーブ
37	FILE_UNLOAD	FU	オブジェクト(プログラム)ファイルのアンロード
38	FILE_VERIFY	FV	ファイル内容とメモリ内容の比較
39	GENERATE_MAKE_FILE	GM	カレントワークスペースの make ファイル作成
40	GO	GO	ユーザプログラムの実行
41	GO_RESET	GR	リセットベクタからのユーザプログラムの実行
42	GO_TILL	GT	テンポラリブレークポイントまでのユーザプログラムの実行
43	HALT	HA	ユーザプログラムの停止
44	HELP	HE	コマンドラインのヘルプ表示
45	INITIALIZE	IN	デバッグプラットフォームの初期化
46	LOG	LO	ロギングファイルの操作
47	MAP_DISPLAY	MA	メモリリソース設定の表示
48	MAP_SET	MS	メモリリソースの設定
49	MEMORY_COMPARE	MC	メモリ内容の比較
50	MEMORY_DISPLAY	MD	メモリ内容の表示
51	MEMORY_EDIT	ME	メモリ内容の変更
52	MEMORY_FILL	MF	指定データによるメモリ内容の一括変更
53	MEMORY_FIND	FI	メモリ内容の検索
54	MEMORY_MOVE	MV	メモリブロックの移動
55	MEMORY_TEST	MT	メモリブロックのテスト
56	OPEN_WORKSPACE	OW	ワークスペースのオープン
57	P_CLOCK_RATE	PCR	周辺クロック比の設定（H8SX のみ）
58	PROFILE	PR	プロファイルの有効化／無効化
59	PROFILE_DISPLAY	PD	プロファイル情報の表示
60	PROFILE_SAVE	PS	プロファイル情報のファイル出力
61	QUIT	QU	HEW の終了
62	RADIX	RA	入力ラディックス(基数)の設定
63	REGISTER_DISPLAY	RD	レジスタ値の表示
64	REGISTER_SET	RS	レジスタ値の設定
65	REMOVE_FILE	REM	カレントプロジェクトからのファイル削除
66	RESET	RE	CPU のリセット
67	RESPONSE	RP	ウィンドウリフレッシュ間隔の設定
68	SLEEP	-	コマンド実行の遅延
69	SAVE_SESSION	SE	現在のセッションをセーブ
70	STATUS	STA	デバッグプラットフォームの状況表示
71	STEP	ST	ステップ実行(命令単位またはソース行単位)
72	STEP_MODE	SM	ステップモードの設定
73	STEP_OUT	SP	PC 位置の関数を終了するまでのステップ実行
74	STEP_OVER	SO	ステップオーバー実行
75	STEP_RATE	SR	ステップ実行速度の設定、表示
76	SUBMIT	SU	コマンドファイルの実行
77	SYMBOL_ADD	SA	シンボルの設定
78	SYMBOL_CLEAR	SC	シンボルの削除
79	SYMBOL_LOAD	SL	シンボル情報ファイルのロード

項番	コマンド名	短縮形	説　明
80	SYMBOL_SAVE	SS	シンボル情報のファイルセーブ
81	SYMBOL_VIEW	SV	シンボルの表示
82	TIMER	TMR	擬似タイマの有効化／無効化（H8SX のみ）
83	TCL	-	TCL の有効化／無効化
84	TOOL_INFORMATION	TO	現在登録されているツールの情報をファイルへ出力
85	TRACE	TR	トレース情報の表示
86	TRACE_ACQUISITION	TA	トレース情報取得の有効/無効の切替
87	TRACE_SAVE	TV	トレース情報をファイルへ出力
88	TRACE_STATISTIC	TST	統計情報解析の実行
89	TRAP_ADDRESS	TP	システムコールアドレスの設定
90	TRAP_ADDRESS_DISPLAY	TD	システムコールアドレス設定の表示
91	TRAP_ADDRESS_ENABLE	TE	システムコール有効化／無効化
92	UPDATE_ALL_DEPENDENCIES	TE	カレントプロジェクトのすべての依存関係更新

以下に各コマンドのシンタックスを示します。

5.1 ! (コメント)

短縮形: なし

説明:

ログファイル等への記録に便利な、コメントを出力することができます。

シンタックス

! <text>

パラメータ	型	説明
<text>	テキスト	出力するテキスト

例:

! Start of test routine　[コマンドライン]ウィンドウ（ログが有効の場合はログファイル）にコメント "Start of test routine" を出力します

5.2 ADD_FILE

短縮形: AF

説明:

カレントプロジェクトにファイルを追加します。

シンタックス

af <filename>

パラメータ	型	説明
<filename>	**文字列**	**ファイル名**

例:

add_file $(PROJDIR)¥¥sbrk.c	プロジェクトディレクトリ内の sbrk.c ファイルを追加します ファイル指定にプレースホルダを使用できます

5.3 ANALYSIS

短縮形: AN

説明:

性能分析を有効、または無効にします。分析数は、実行前に自動的にリセットしません。

シンタックス

an [<state>]

パラメータ	型	説明
なし		**分析状況を表示します**
<state>	**キーワード**	**分析を設定します**
	enable	**分析を可能にします**
	disable	**分析を無効にします**
	reset	**分析数をリセットします**

例:

ANALYSIS	分析状況を表示します
AN enable	分析を可能にします
AN disable	分析を無効にします
AN reset	分析数をリセットします

5.4 ANALYSIS_RANGE

短縮形: AR

説明:

性能評価を行う関数を設定するか、またはパラメータなしで性能評価を行う関数を表示します。

シンタックス

ar [<関数名>]

パラメータ	型	説明
なし		**すべての性能評価を行う関数を表示します**
<関数名>	**文字列**	**性能評価を行う関数名**

例:

```
ANALYSIS_RANGE   sort      関数 sort の性能評価を行います
AR                         性能評価を行う関数を表示します
```

5.5 ANALYSIS_RANGE_DELETE

短縮形: AD

説明:

指定した項目番号の関数、またはパラメータを何も指定しなかった場合すべての関数を削除します(削除時に確認はしません)。

シンタックス

ad [<index>]

パラメータ	型	説明
なし		**すべての関数を削除します**
<index>	**数値**	**削除する関数の番号**

例:

```
ANALYSIS_RANGE_DELETE 6    項目番号 6 の関数を削除します
AD                         すべての関数を削除します
```

5.6 ASSEMBLE

短縮形: AS

説明:

アセンブルし、メモリに書き込みます。
アセンブルモードでは "." は終了、"^" は2バイト戻り、”Enter”キーを押すと先に進みます。

シンタックス
as <address>

パラメータ	型	説明
<address>	数値	アセンブルを開始するアドレス

例:

AS H’1000 H'1000 からアセンブルします

5.7 ASSERT

短縮形: なし

説明:

式が真であることを調べます。 式が偽のときはバッチファイルを終了するため、バッチファイルで使えます。 式が偽のときエラーを返します。 このコマンドは、サブルーチンのテストハーネスを記述するために使うことができます。

シンタックス
assert <expression>

パラメータ	型	説明
<expression>	式	判定する式

例:

ASSERT #R0 == 0x100 R0 が 0x100 を含んでいないときエラーを返します

5.8 AUTO_COMPLETE

短縮形: AC

説明:

コマンドラインウィンドウでのオートコンプリート機能をONまたはOFFに切り替えます。ON状態のときは、ユーザの入力がコマンドのリストと合致するようにします。また有効なコマンドが入力された場合には、パラメータのリストを表示し、ユーザの入力を補助します。

シンタックス

auto_complete [<state>]

パラメータ	型	説明
なし		現在の設定を表示します
<state>	キーワード	オートコンプリート機能の有効／無効を設定します
	enable	オートコンプリートを有効にします
	disable	オートコンプリートを無効にします

例:

AC	オートコンプリートの情報を表示します
AC enable	オートコンプリートを有効にします
AC d	オートコンプリートを無効にします

5.9 BREAKPOINT

短縮形: BP

説明:

実行命令位置によるブレークポイントを設定します。

シンタックス

bp <address> [<count>] [<Action>]

パラメータ	型	説明
<address>	**数値**	**ブレークポイントのアドレス**
<count>	**数値**	**指定アドレスの命令をフェッチする回数（1〜16383、デフォルト=1）**
<Action>	**キーワード**	**条件成立時の動作（任意、デフォルト＝Stop）**
	Stop（**短縮形**:P）	**ユーザプログラム実行を停止する**
	Input（**短縮形**:I）	**ファイルからデータを入力する**
	Output（**短縮形**:O）	**ファイルにデータを出力する**
	Interrupt（**短縮形**:T）	**擬似割込みを発生する**

書式:

各Actionの指定方法を下記に示します。

<”Stop”>

<”Input”> <filename> <addr> <size> <count>

パラメータ	型	説明
<filename>	文字列	入力するファイル名
<addr>	数値	データを読み込むアドレス
<size>	数値	データ１個のサイズ（1/2/4/8）
<count>	数値	データ数（1〜H’FFFFFFFF）

<”Output”> <filename> <addr> <size> <count> [<option>]

パラメータ	型	説明
<filename>	文字列	出力するファイル名
<addr>	数値	データを出力するアドレス
<size>	数値	データ１個のサイズ（1/2/4/8）
<count>	数値	データ数（1〜H’FFFFFFFF）
<option>	キーワード	新規／追加指定（任意、省略時は新規ファイル作成）
	A	既存ファイルにデータを追加

<”Interrupt”> <interrupt type1> [<priority>]

パラメータ	型	説明
<interrupt type1>	数値	割込み種別 割込みベクタ番号（0〜H’FF）
<priority>	数値	割込み優先順位（任意、デフォルト＝0） 0〜17

例:

BREAKPOINT 0 2	0 番地の命令を２回目に実行しようとしたとき、ブレークするように設定します
BP C0 Input in.dat 100 2 8	H'C0 番地の命令を実行しようとしたとき、ファイル in.dat から２バイトデータを 8 個、H'100 番地からへ書きこみます

5.10 BREAK_ACCESS

短縮形: BA

説明:

メモリ範囲のアクセスによるブレーク条件を設定します。

シンタックス

ba <start address> [<end address>] [<mode>] [<Action>]

パラメータ	型	説明
<start address>	**数値**	**ブレークポイントの開始アドレス**
<end address>	**数値**	**ブレークポイントの終了アドレス（任意、デフォルト=開始アドレス）**
<mode>	**キーワード**	**アクセス種別（任意、デフォルト**=RW）
	R	**リードした場合にブレーク**
	W	**ライトした場合にブレーク**
	RW	**リードまたはライトした場合にブレーク**
<Action>	**キーワード**	**条件成立時の動作（任意、デフォルト**＝Stop）
	Stop（**短縮形**:P）	**ユーザプログラム実行を停止する**
	Input（**短縮形**:I）	**ファイルからデータを入力する**
	Output（**短縮形**:O）	**ファイルにデータを出力する**
	Interrupt（**短縮形**:T）	**擬似割込みを発生する**

書式:

各Actionの指定方法を下記に示します。

<”Stop”>

<”Input”> <filename> <addr> <size> <count>

パラメータ	型	説明
<filename>	**文字列**	**入力するファイル名**
<addr>	**数値**	**データを読み込むアドレス**
<size>	**数値**	**データ１個のサイズ**（1/2/4/8）
<count>	**数値**	**データ数**（1～H’FFFFFFFF）

<”Output”> <filename> <addr> <size> <count> [<option>]

パラメータ	型	説明
<filename>	文字列	出力するファイル名
<addr>	数値	データを出力するアドレス
<size>	数値	データ１個のサイズ（1/2/4/8）
<count>	数値	データ数（1〜H’FFFFFFFF）
<option>	キーワード	新規／追加指定（任意、省略時は新規ファイル作成）
	A	既存ファイルにデータを追加

<”Interrupt”> <interrupt type1> [<priority>]

パラメータ	型	説明
<interrupt type1>	数値	割込み種別 割込みベクタ番号（0〜H’FF）
<priority>	数値	割込み優先順位（任意、デフォルト＝0） 0〜17

例:

BREAK_ACCESS 0 1000 W	0 番地から H’1000 番地の範囲でライトアクセスが発生したときブレークするように設定します
BA FFFF	H’FFFF 番地でリードまたはライトアクセスが発生したときブレークするように設定します

5.11 BREAK_CLEAR

短縮形: BC

説明:

ブレークポイントを削除します。

シンタックス

bc [<index>]

パラメータ	型	説明
<index>	数値	削除するブレークポイントのインデックス(省略した場合は全てのブレークポイントを削除します。)

例:

BREAK_CLEAR 0	１番目のブレークポイントを削除します
BC	全てのブレークポイントを削除します

5.12 BREAK_CYCLE

短縮形: BCY

説明:

サイクル数によるブレーク条件を設定します。

シンタックス

bcy <cycle> [<count>] [<Action>]

パラメータ	型	説明
<cycle>	数値	<cycle>×n のサイクルで条件が一致します
<count>	キーワード	条件が成立する回数（任意、デフォルト＝ALL）
	ALL	条件一致するごとに、すべてブレーク条件が成立します
	数値	１～H'FFFF 条件一致した回数が指定回数以下の時だけブレーク条件が成立します
<Action>	キーワード	条件成立時の動作（任意、デフォルト＝Stop）
	Stop（短縮形:P）	ユーザプログラム実行を停止する
	Input（短縮形:I）	ファイルからデータを入力する
	Output（短縮形:O）	ファイルにデータを出力する
	Interrupt（短縮形:T）	擬似割込みを発生する

書式:

各Actionの指定方法を下記に示します。

<"Stop">

<"Input"> <filename> <addr> <size> <count>

パラメータ	型	説明
<filename>	文字列	入力するファイル名
<addr>	数値	データを読み込むアドレス
<size>	数値	データ１個のサイズ（1/2/4/8）
<count>	数値	データ数（1～H'FFFFFFFF）

<"Output"> <filename> <addr> <size> <count> [<option>]

パラメータ	型	説明
<filename>	文字列	出力するファイル名
<addr>	数値	データを出力するアドレス
<size>	数値	データ１個のサイズ（1/2/4/8）
<count>	数値	データ数（1～H'FFFFFFFF）
<option>	キーワード	新規／追加指定（任意、省略時は新規ファイル作成）
	A	既存ファイルにデータを追加

<”Interrupt”> <interrupt type1> [<priority>]

パラメータ	型	説明
<interrupt type1>	数値	割込み種別 割込みベクタ番号（0〜H'FF）
<priority>	数値	割込み優先順位（任意、デフォルト＝0） 0〜17

例:

BREAK_CYCLE 1000 20　　H'1000 サイクルごとに H'20 回ブレークするように設定します
BCY 5000　　H'5000 サイクルごとにブレークするように設定します

5.13 BREAK_DATA

短縮形: BD

説明:

メモリのデータ値によるブレーク条件を設定します。

シンタックス

bd <address> <data> [<size>] [<data mask>] [<option>] [<Action>]

パラメータ	型	説明
<address>	数値	ブレーク条件の判定を行うアドレス
<data>	数値	アクセスデータ
<size>	キーワード	アクセスサイズ（任意、デフォルト=B）
	Byte	バイトデータ
	Word	ワードデータ
	Long	ロングワードデータ
	Single	単精度浮動小数点データ
	Double	倍精度浮動小数点データ
<data mask>	数値	<data>をマスクする値、0 はマスク、1 は非マスク
<option>	キーワード	データの一致/不一致（任意、デフォルト=EQ）
	EQ	データが一致したときブレーク
	NE	データが不一致となったときブレーク
<Action>	キーワード	条件成立時の動作（任意、デフォルト＝Stop）
	Stop（短縮形:P）	ユーザプログラム実行を停止する
	Input（短縮形:I）	ファイルからデータを入力する
	Output（短縮形:O）	ファイルにデータを出力する
	Interrupt（短縮形:T）	擬似割込みを発生する

書式:

各Actionの指定方法を下記に示します。

<”Stop”>

<”Input”> <filename> <addr> <size> <count>

パラメータ	型	説明
<filename>	文字列	入力するファイル名
<addr>	数値	データを読み込むアドレス
<size>	数値	データ１個のサイズ（1/2/4/8）
<count>	数値	データ数（1～H’FFFFFFFF）

<”Output”> <filename> <addr> <size> <count> [<option>]

パラメータ	型	説明
<filename>	文字列	出力するファイル名
<addr>	数値	データを出力するアドレス
<size>	数値	データ１個のサイズ（1/2/4/8）
<count>	数値	データ数（1～H’FFFFFFFF）
<option>	キーワード	新規／追加指定（任意、省略時は新規ファイル作成）
	A	既存ファイルにデータを追加

<”Interrupt”> <interrupt type1> [<priority>]

パラメータ	型	説明
<interrupt type1>	数値	割込み種別 割込みベクタ番号（0～H’FF）
<priority>	数値	割込み優先順位（任意、デフォルト＝0） 0～17

例:

BREAK_DATA 0 100 L EQ	0 番地のメモリにロングワードサイズで H’100 を書き込んだときブレークするように設定します
BD C0 FF B NE	H’C0 番地のメモリにバイトサイズの H’FF 以外の値を書き込んだときブレークするように設定します
BD 4000 10	H’4000 番地のメモリにバイトサイズで H’10 を書き込んだときブレークするように設定します

5.14 BREAK_DISPLAY

短縮形: BI

説明:

ブレークポイント一覧を表示します。

シンタックス

bi

パラメータ	型	説明
なし		ブレークポイント一覧を表示します

例:

BREAK_DISPLAY　　ブレークポイントの一覧を表示します
BI　　ブレークポイントの一覧を表示します

5.15 BREAK_ENABLE

短縮形: BE

説明:

ブレークポイントの有効/無効を切り替えます。

シンタックス

be <flag> [<index>]

パラメータ	型	説明
<flag>	キーワード	有効/無効
	E	有効
	D	無効
<index>	数値	有効／無効を切りかえるブレークポイントのインデックス（省略した場合は全てのブレークポイントを対象とします。）

例:

BREAK_ENABLE D 0　　１番目のブレークポイントを無効にします
BE E　　全てのブレークポイントを有効にします

5.16 BREAK_REGISTER

短縮形: BR

説明:

レジスタのデータ値によるブレーク条件を設定します。

シンタックス

br <register> [<data> <size> [<data mask>]] [<option>] [<Action>]

パラメータ	型	説明
<register>	文字列	レジスタ名
<data>	数値	アクセスデータ
<size>	キーワード	アクセスサイズ（任意、省略した場合は指定レジスタのサイズとします。ただし、データ値設定時は省略不可となります。）
	B	バイトデータ
	W	ワードデータ
	L	ロングワードデータ
	S	単精度浮動小数点データ
	D	倍精度浮動小数点データ
<data mask>	数値	<data>をマスクする値、0 だとマスク
<option>	キーワード	データの一致/不一致（任意、デフォルト=EQ）
	EQ	データが一致したときブレーク
	NE	データが不一致となったときブレーク
<Action>	キーワード	条件成立時の動作（任意、デフォルト＝Stop）
	Stop（短縮形:P）	ユーザプログラム実行を停止する
	Input（短縮形:I）	ファイルからデータを入力する
	Output（短縮形:O）	ファイルにデータを出力する
	Interrupt（短縮形:T）	擬似割込みを発生する

書式:

各Actionの指定方法を下記に示します。

<”Stop”>

<”Input”> <filename> <addr> <size> <count>

パラメータ	型	説明
<filename>	文字列	入力するファイル名
<addr>	数値	データを読み込むアドレス
<size>	数値	データ１個のサイズ（1/2/4/8）
<count>	数値	データ数（1～H'FFFFFFFF）

<”Output”> <filename> <addr> <size> <count> [<option>]

パラメータ	型	説明
<filename>	文字列	出力するファイル名
<addr>	数値	データを出力するアドレス
<size>	数値	データ１個のサイズ（1/2/4/8）
<count>	数値	データ数（1〜H’FFFFFFFF）
<option>	キーワード	新規／追加指定（任意、省略時は新規ファイル作成）
	A	既存ファイルにデータを追加

<”Interrupt”> <interrupt type1> [<priority>]

パラメータ	型	説明
<interrupt type1>	数値	割込み種別 割込みベクタ番号（0〜H’FF）
<priority>	数値	割込み優先順位（任意、デフォルト＝0） 0〜17

例:

BREAK_REGISTER R0 FFFF W EQ	R0 レジスタの下位２バイトが H’FFFF になったときブレークします
BR R10	R10 レジスタにライトアクセスが発生したときブレークします

5.17 BREAK_SEQUENCE

短縮形: BS

説明:

実行順序を指定したブレークポイントを設定します。

シンタックス

bs <address1> [<address2> [<address3> […]]] [<Action>]

パラメータ	型	説明
<address1> 〜 <address8>	数値	シーケンシャルブレークポイントとなるアドレス（最大８個まで指定できます。）
<Action>	キーワード	条件成立時の動作（任意、デフォルト＝Stop）
	Stop（短縮形:P）	ユーザプログラム実行を停止する
	Input（短縮形:I）	ファイルからデータを入力する
	Output（短縮形:O）	ファイルにデータを出力する
	Interrupt（短縮形:T）	擬似割込みを発生する

書式:

各Actionの指定方法を下記に示します。

<”Stop”>

<”Input”> <filename> <addr> <size> <count>

パラメータ	型	説明
<filename>	文字列	入力するファイル名
<addr>	数値	データを読み込むアドレス
<size>	数値	データ１個のサイズ（1/2/4/8）
<count>	数値	データ数（1〜H’FFFFFFFF）

<”Output”> <filename> <addr> <size> <count> [<option>]

パラメータ	型	説明
<filename>	文字列	出力するファイル名
<addr>	数値	データを出力するアドレス
<size>	数値	データ１個のサイズ（1/2/4/8）
<count>	数値	データ数（1〜H’FFFFFFFF）
<option>	キーワード	新規／追加指定（任意、省略時は新規ファイル作成）
	A	既存ファイルにデータを追加

<”Interrupt”> <interrupt type1> [<priority>]

パラメータ	型	説明
<interrupt type1>	数値	割込み種別 割込みベクタ番号（0〜H’FF）
<priority>	数値	割込み優先順位（任意、デフォルト＝0） 0〜17

例:

BREAK_SEQUENCE 1000 2000	通過位置が H’1000 番地、H’2000 番地のときブレークします
BS 1000	H’1000 番地を実行するときブレークするように設定します

5.18 BUILD

短縮形: BU

説明:

カレントプロジェクトのビルド処理を開始します。

シンタックス

bu

パラメータ	型	説明
なし		カレントプロジェクトのビルド処理を開始します

例:

build	カレントプロジェクトのビルド処理を開始します

5.19 BUILD_ALL

短縮形: BL

説明:

カレントプロジェクトのすべてのビルド処理を開始します。

シンタックス

bl

パラメータ	型	説明
なし		**カレントプロジェクトのすべてのビルド処理を開始します**

例:

build_all	カレントプロジェクトのすべてのビルド処理を開始します

5.20 CACHE

短縮形: なし

説明:

メモリキャッシュの有効／無効設定、または現在のキャッシュ状態を表示します。

シンタックス

cache [<state>]

パラメータ	型	説明
なし		**カレントプロジェクトのビルド処理を開始します**
<state>	**キーワード**	**メモリキャッシュ状態**
	On	**メモリキャッシュを有効にします**
	Off	**メモリキャッシュを無効にします**

例:

cache	現在のメモリキャッシュ状態を表示します
cache On	メモリキャッシュを有効にします
cache Off	メモリキャッシュを無効にします

5.21 CHANGE_CONFIGURATION

短縮形: CC

説明:
現在のコンフィギュレーションを設定します。

シンタックス
cc <configuration>

パラメータ	型	説明
<configuration>	文字列	コンフィギュレーション名

例:
CC Debug 現在のコンフィギュレーションを”Debug”に設定します

5.22 CHANGE_PROJECT

短縮形: CP

説明:
現在のプロジェクトを設定します。

シンタックス
cp <project>

パラメータ	型	説明
<project>	文字列	プロジェクト名

例:
CP PROJ2 現在のプロジェクトを” PROJ2”に設定します

5.23 CHANGE_SESSION

短縮形: CS

説明:
現在のセッションを設定します。

シンタックス
cs <session>

パラメータ	型	説明
<session>	文字列	セッション名

例:
CS DefaultSession 現在のセッションを”Default Session”に変更します。

5.24 COVERAGE

短縮形: CV

説明:

カバレジ測定の有効／無効設定、およびカバレジ情報をクリアします。

シンタックス

cv [<function number>] [<state>]

パラメータ	型	説明
<function number>	数値	表示、有効化、無効化、リセットする関数の番号を指定します 番号は COVERAGE_RANGE コマンドで表示します <function number>省略時は全関数が対象となります
<state>	キーワード	<state>省略時は、ガバレジの状態を表示します
	enable	カバレジ測定を有効にします
	disable	カバレジ測定を無効にします
	reset	カバレジ測定結果をリセットします

例:

COVERAGE	カバレジの状態を表示します
CV 1 enable	番号１の関数のカバレジ測定を有効にします
CV r	カバレジ測定結果をリセットします

5.25 COVERAGE_DISPLAY

短縮形: CVD

説明:

カバレジ情報を表示します。

シンタックス

cvd

パラメータ	型	説明
なし		カバレジ情報を表示します

例:

COVERAGE_DISPLAY	カバレジ情報を表示します

5.26 COVERAGE_LOAD

短縮形: CVL

説明:

カバレジ情報を.COVファイルからロードします。
異なるファイルフォーマットの入力や指定ファイルが存在しない場合はウォーニングメッセージを表示します。

【注】 カバレジ測定範囲をソースファイル名で指定した場合は、保存した.COV ファイルをロードすることはできません。

シンタックス

cvl <filename>

パラメータ	型	説明
<filename>	文字列	ファイル名

例:

COVERAGE_LOAD TEST TEST.COV ファイルからカバレジ情報をロードします
CVL COVERAGE.COV COVERAGE.COV ファイルからカバレジ情報をロードします

5.27 COVERAGE_RANGE

短縮形: CVR

説明:

カバレジの範囲を設定するか、またはパラメータなしでカバレジ測定の範囲を表示します。

シンタックス

cvr [[<start> <end>] | [<file name>]]

パラメータ	型	説明
なし		カバレジ測定の関数番号、関数名、アドレス範囲を表示します
<start>	数値	カバレジ測定範囲の開始アドレス
<end>	数値	カバレジ測定範囲の終了アドレス
<file name>	文字列	カバレジ取得するソースファイル

例:

COVERAGE_RANGE H'1000 H'10FF H'1000 から H'10FF のカバレジ測定を行います
CVR カバレジ測定の範囲を表示します

5.28 COVERAGE_SAVE

短縮形: CVS

説明:

カバレジ情報を.COVファイルまたは.TXTファイルに保存します。
.COVまたは.TXT以外のファイル拡張子を入力するとエラーメッセージが出力されます。

シンタックス
cvs <filename>

パラメータ	型	説明
<filename>	**文字列**	**ファイル名**

例:

COVERAGE_SAVE TEST　TEST.COV ファイルにカバレジ情報を保存します
CVS COVERAGE.COV　COVERAGE.COV ファイルにカバレジ情報を保存します

5.29 DEFAULT_OBJECT_FORMAT

短縮形: DO

説明:

オブジェクト（プログラム）ファイルをロードするときのデフォルトフォーマットを設定します。FILE_LOADコマンドでフォーマットの指定を省略した場合に、このコマンドで設定したフォーマットが有効になります。

シンタックス
do [<format>]

パラメータ	型	説明
なし		**デフォルトフォーマットの設定を表示します**
<format>	**キーワード**	**オブジェクトフォーマット**
	Binary	**バイナリタイプ**
	Elf/Dwarf2	Elf/Dwarf2 **タイプ**
	IntelHex	Intel Hex **タイプ**
	S-Record	S **タイプ**

例:

DEFAULT_OBJECT_FORMAT　デフォルトフォーマットの設定を表示します
DO binary　デフォルトフォーマットをバイナリに設定します

5.30 DISASSEMBLE

短縮形: DA

説明:

メモリ内容を逆アセンブル表示します。 逆アセンブル表示は完全なシンボリックです。

シンタックス

da <address> [<length>]

パラメータ	型	説明
<address>	数値	開始アドレス
<length>	数値	命令数（任意、デフォルト=16）

例:

DISASSEMBLE H’100 5　　アドレス H'100 からコード 5 行を逆アセンブル表示します
DA H’3E00 20　　アドレス H'3E00 からコード 20 行を逆アセンブル表示します

5.31 ERASE

短縮形: ER

説明:

[コマンドライン]ウィンドウをクリアします。

シンタックス

er

パラメータ	型	説明
なし		[コマンドライン]ウィンドウをクリアします

例:

ER　　[コマンドライン]ウィンドウをクリアします

5.32 EVALUATE

短縮形: EV

説明:

簡単な式、そして括弧、混合基数とシンボルを持つ複雑な式を評価することができます。 すべての演算子は同じ優先順位を持っていますが、括弧は評価の順序を変更することができます。 演算子はC/C++と同じ意味を持っています。 式は数値を要求するコマンドで使うことができます。 レジスタ名を使うことができますが、先頭に"#"文字を付けなければいけません。結果は、16進、10進、8進、2進で表示します。

シンタックス

ev <expression>

パラメータ	型	説明
<expression>	**式**	**評価する式**

有効な演算子:

&& **論理** AND	|| **論理** OR	<< **左算術シフト**	>> **右算術シフト**
+ **加算**	- **減算**	* **乗算**	/ **除算**
% **剰余**	| **ビット毎の** OR	& **ビット毎の** AND	~ **ビット毎の** NOT
^ **ビット毎の排他的** OR	! **論理** NOT	== **等しい**	!= **等しくない**
> **より大きい**	< **より小さい**	>= **以上**	<= **以下**

例:

EV H’123 + (D’73 | B’10)　　結果: H'16E D'366 O'556
B'00000000000000000000000101101110

EV #R1 * #R2　　結果: H'121 D'289 O'441
B'00000000000000000000000100100001

5.33 EXEC_MODE

短縮形: EM

説明:

シミュレーションエラーが発生した場合のシミュレータ・デバッガ動作を指定します。

シンタックス

em [<mode>]

パラメータ	型	説明
なし		現在の実行モードを表示します
<mode>	キーワード	実行モード(シミュレーションエラー時のシミュレータ・デバッガ動作)
	S	停止
	C	続行

例:

EXEC_MODE	現在の実行モードを表示します
EM c	実行モードを「続行」に設定します

5.34 EXEC_STOP_SET

短縮形: ESS

説明:

擬似割込みおよびタイマー割り込みが発生した場合のシミュレータ・デバッガ動作を指定します。ただし、EXEC_MODEの設定がS（停止）の時のみ設定が有効になり、C（続行）の場合は、常に続行になります。ただし、タイマー割り込みをサポートしているのはH8SXのみです。

シンタックス

ess INTERRUPT <break cause> [<mode>]

パラメータ	型	説明
<break cause>	キーワード	発生した割り込みの種別を指定します 現在は、INTERRUPT のみ指定可能です
	Interrupt	擬似割込みを指定します
<mode>	キーワード	割込み発生時の実行モード <mode>を省略した場合、現在の実行モードを表示します
	Stop	停止
	Continue	続行（デフォルト）

例:

EXEC_STOP_SET INTERRUPT	現在の割込み発生時の実行モードを表示します
ESS INTERRUPT C	割込み発生時の実行モードを「続行」に設定します

5.35 FILE_LOAD

短縮形: FL

説明:

指定したファイルをメモリにロードします。
<offset>パラメータを指定するとシンボルにも加算します。
<style>パラメータを省略した場合、HEWに登録しないため、ロードしたファイルに関してはシンボリックデバッグを行うことはできません。シンボリックデバッグを行う場合は、R（置き換え）またはP（追加）を指定してください。
シミュレータ・デバッガでは、<mode>パラメータを使用しないことをお勧めします。リトルエンディアンでビルドしたファイルは、byteサイズでロードするように作られています。リトルエンディアンでは、byteサイズ以外でロードした場合の動作を保証できません。
主に、エミュレータをご使用の場合で、決められたサイズでしかライトできないメモリへロードする時に使用するパラメータです。

シンタックス

fl [<format>] <filename> [<offset>] [<verify>] [<style>] [<mode>]

パラメータ	型	説明
<format>	キーワード	オブジェクトフォーマット （任意、デフォルト=DEFAULT_OBJECT_FORMAT の設定）
	Binary	バイナリタイプ
	Elf/Dwarf2	Elf/Dwarf2 タイプ
	IntelHex	Intel Hex タイプ
	S-Record	S タイプ
<filename>	文字列	ファイル名
<offset>	数値	ロードアドレスに加えるオフセットを設定します（任意、デフォルト=0）
<verify>	キーワード	ベリファイフラグ（任意、デフォルト=V）
	V	ベリファイあり
	N	ベリファイなし
<style>	キーワード	ロードするファイルを HEW に登録するかどうかを指定します。 （デフォルト =HEW に登録せず、デバッグ情報をロードしません）
	R	HEW の登録をクリアし、指定したファイルを新たに登録します。デバッグ情報も読み込みます。
	P	指定したファイルを追加登録します。デバッグ情報も読み込みます。
<mode>	キーワード	メモリライト時のサイズ（任意、デフォルト=byte）
	byte	バイト
	word	ワード
	long	ロングワード
	ascii	ASCII
	single	単精度浮動小数点
	double	倍精度浮動小数点

例:

FILE_LOAD S-RECORD A:¥¥BINARY¥¥TESTFILE.A22　モトローラ S レコードファイル "testfile.a22" をロードします

5.36 FILE_SAVE

短縮形: FS

説明:

メモリ内容をファイルへ保存します。すでに存在するファイル名を指定した場合は、上書きするかどうかユーザに確認します。シンボルはセーブしません。

シンタックス

fs [<format>] <filename> <start> <end> [<mode>]

パラメータ	型	説明
<format>	**キーワード**	**オブジェクトフォーマット（任意、デフォルト**=DEFAULT_OBJECT_FORMAT **の設定）**
	Binary	**バイナリタイプ**
	IntelHex	Intel Hex **タイプ**
	S-Record	S **タイプ**
<filename>	**文字列**	**ファイル名**
<start>	**数値**	**開始アドレス**
<end>	**数値**	**終了アドレス**
<mode>	**キーワード**	**メモリ内容をリードする時のフォーマット（任意、デフォルト**=byte**）**
	byte	**バイト**
	word	**ワード**
	Long	**ロングワード**
	Ascii	ASCII
	Single	**単精度浮動小数点**
	Double	**倍精度浮動小数点**

例:

FS S-RECORD D:¥¥USER¥¥ANOTHER.A22 H’4000 H’4FFF　アドレス範囲 H'4000 - H'4FFF をモトローラ S レコードフォーマットファイル "ANOTHER.A22" として保存します

5.37 FILE_UNLOAD

短縮形: FU

説明:

メモリから指定オフセットのファイルをアンロードします。オフセットの指定がない場合は、メモリを検索して最初に一致したファイルをアンロードします。

シンタックス

fu [<format>] <filename> [<offset>]

パラメータ	型	説明
<format>	**キーワード**	**オブジェクトフォーマット（任意、デフォルト**=DEFAULT_OBJECT_FORMAT **の設定）**
	Binary	**バイナリタイプ**
	Elf/Dwarf2	Elf/Dwarf2 **タイプ**
	IntelHex	Intel Hex **タイプ**
	S-Record	S **タイプ**
<filename>	**文字列**	**ファイル名**
<offset>	**数値**	**アンロードするファイルのオフセットを指定します（任意）**

例:

FILE_UNLOAD TEST.A22 ファイル ”TEST.A22” をアンロードします

5.38 FILE_VERIFY

短縮形: FV

説明:

メモリとファイル内容をベリファイします。

シンタックス

fv [<format>] <filename> [<offset>] [<mode>]

パラメータ	型	説明
<format>	**キーワード**	**オブジェクトフォーマット（任意、デフォルト**=DEFAULT_OBJECT_FORMAT **の設定）**
	Binary	**バイナリタイプ**
	IntelHex	Intel Hex **タイプ**
	S-Record	S **タイプ**
<filename>	**文字列**	**ファイル名**
<offset>	**数値**	**ファイルアドレスへ加えるオフセットを設定します（任意、デフォルト**=0）
<mode>	**キーワード**	**メモリ内容リード時のフォーマット（任意、デフォルト**=byte）
	byte	**バイト**
	word	**ワード**
	long	**ロングワード**
	ascii	ASCII
	single	**単精度浮動小数点**
	double	**倍精度浮動小数点**

例:

FILE_VERIFY A:¥¥BINARY¥¥TEST.A22　　メモリに対して モトローラ S レコードファイル "TEST.A22" をベリファイします

5.39 GENERATE_MAKE_FILE

短縮形: GM

説明:

HEW外でビルドするためのmakeファイルを生成します。

シンタックス

gm [<mode>] [<scan>] [<sub>] [<format>]

パラメータ	型	説明
<mode>	**キーワード**	
	cccp	make **ファイルをカレントプロジェクトのカレントコンフィグレーションから生成します**
	accp	make **ファイルをカレントプロジェクトのすべてのコンフィグレーションから生成します**
	acap	make **ファイルをすべてのプロジェクトのすべてのコンフィグレーションから生成します**
<scan>	**キーワード**	
	scan	**依存関係をスキャンする（デフォルト）**
	noscan	**依存関係をスキャンしない**
<sub>	**キーワード**	
	sub	**サブコマンドファイルを使用します**
	nosub	**サブコマンドファイルを使用しません（デフォルト）**
<format>	**キーワード**	
	hmake	Hmake make **ファイル互換の** make **ファイルを生成します（デフォルト）**
	nmake	Nmake make **ファイル互換の** make **ファイルを生成します。**
	gnumkae	Gmake make **ファイル互換の** make **ファイルを生成します。**

例:

Generate_make_file cccp scan　　make ファイルをカレントプロジェクトのカレントコンフィグレーションから生成します
make ファイル作成前に依存関係をスキャンします
この例では、サブコマンドファイルを使用せずに
Hmake make ファイル互換の make ファイルを生成します。

5.40 GO

短縮形: GO

説明:

ユーザプログラムを実行します。ユーザプログラムが停止すると、PC値を表示します。

シンタックス

go [<wait key>] [<address>]

パラメータ	型	説明
<wait key>	**キーワード**	**プログラム実行中でのコマンド処理の有効／無効（任意、デフォルト**= wait **）**
	wait	**コマンド処理を続けることができません**
	continue	**コマンド処理を続けることができます**
<address>	**数値**	PC **のための開始アドレス（任意、デフォルト**=PC **値）**

wait はデフォルトで、プログラムが実行を停止するまでコマンド処理ができません。

continue は、コマンド処理を続けることができます（デバッグプラットフォームの機能により動作しないものもあります）。

例:

GO	現在の PC からユーザプログラムを実行します (コマンド処理を続けることができません)
GO CONTINUE H’1000	H'1000 からユーザプログラムを実行します (コマンド処理を続けることができます)

5.41 GO_RESET

短縮形: GR

説明:

リセットベクタで指定しているアドレスから始まるユーザプログラムを実行します。ユーザプログラムを実行しているとき、[パフォーマンス解析]ウィンドウを更新します。

シンタックス

gr [<wait key>]

パラメータ	型	説明
<wait key>	**キーワード**	**プログラム実行中でのコマンド処理の有効／無効（任意、デフォルト**= wait **）**
	wait	**コマンド処理を続けることができません**
	continue	**コマンド処理を続けることができます**

wait はデフォルトで、プログラムが実行を停止するまでコマンド処理ができません。

continue は、コマンド処理を続けることができます（デバッグプラットフォームの機能により動作しないものもあります）。

例:

GR　　リセットベクタで指定しているアドレスから始まる、ユーザプログラムを実行します (コマンド処理を続けることができません)

5.42 GO_TILL

短縮形: GT

説明:

テンポラリブレークポイントを設定し、現在の PC からプログラムを実行します。 テンポラリブレークポイントとして、複数のアドレスを指定することができます（テンポラリブレークポイントは、コマンド実行中のみ有効です）。

シンタックス

gt　[<wait key>]　<breakpoints>...

パラメータ	型	説明
<wait key>	**キーワード**	**プログラム実行中でのコマンド処理の有効／無効（任意、デフォルト**= wait **）**
	wait	**コマンド処理を続けることができません**
	continue	**コマンド処理を続けることができます**
<breakpoints>...	**数値**	**テンポラリブレークポイントのアドレス（複数指定可）**

wait はデフォルトで、プログラムが実行を停止するまでコマンド処理ができません。

continue は、コマンド処理を続けることができます（デバッグプラットフォームの機能により動作しないものもあります）。

例:

GO_TILL H’1000　　PC がアドレス H'1000 に到達するまでユーザプログラムを実行します

5.43 HALT

短縮形: HA

説明:

ユーザプログラムを停止します（"go continue"コマンドの後で、使うことができます）。

シンタックス

ha

パラメータ	型	説明
なし		**ユーザプログラムを停止します**

例:

HA　　ユーザプログラムを停止します

5.44 HELP

短縮形: HE

説明:

コマンドのシンタックスを表示します

シンタックス

help [<command name>]

パラメータ	型	説明
なし		**全コマンドのシンタックスを表示します。**
<command name>		**表示するコマンドを指定します。** **コマンドの指定は短縮形でも可能です。**

例:

help go go コマンドのシンタックスを表示します

5.45 INITIALIZE

短縮形: IN

説明:

デバッグプラットフォームを初期化（ターゲットライブラリを再選択する）します。すべてのブレークポイント、メモリマッピングなどもリセットします。

シンタックス

in

パラメータ	型	説明
なし		**デバッグプラットフォームを初期化します**

例:

IN デバッグプロットフォームを初期化します

5.46 LOG

短縮形: LO

説明:

ファイルへのコマンド出力のログを制御します。 パラメータなしでは、現在のログ状況を表示します。 存在するファイルを指定すると、追加するかをユーザに確認します。No と答えるとデータをファイルに上書きし、Yes と答えるとファイルに追加します。 ロギングはコマンドラインインタフェースでのみサポートします。

シンタックス

lo [<state> | <filename>]

パラメータ	型	説明
なし		**ロギング状態を表示します**
<state>	**キーワード**	**ロギングを再開／停止します**
	+	**ロギングを再開します**
	-	**ロギングを停止します**
<filename>	**文字列**	**ロギングを出力するファイル名を指定します**

例:

LOG TEST	ファイル TEST への出力をロギングします
LO -	ロギングを停止します
LOG +	ロギングを再開します
LOG	現在のロギング状態を表示します

5.47 MAP_DISPLAY

短縮形: MA

説明:

現在のメモリリソース設定を表示します。

シンタックス

ma

パラメータ	型	説明
なし		**現在のメモリリソース設定を表示します**

例:

MA	現在のメモリリソース設定を表示します

5.48 MAP_SET

短縮形: MS

説明:

メモリリソースを設定します。

シンタックス

ms <start address> [<end address>] [<mode>]

パラメータ	型	説明
<start address>	**数値**	**開始アドレス**
<end address>	**数値**	**終了アドレス（任意、デフォルト＝開始アドレス）**
<mode>	**キーワード**	**アクセス種別（任意、デフォルト**=RW**）**
	R	**リードのみ可**
	W	**ライトのみ可**
	RW	**リード／ライト可**

例:

MAP_SET 0000 3FFF RW	H’0000 番地から H’3FFF 番地をリード/ライト可能な領域として確保します
MS 5000	H’5000 番地をリード/ライト可能な領域として確保します

5.49 MEMORY_COMPARE

短縮形: MC

説明:

メモリ内容を比較します。

シンタックス

mc <start> <end> <destination> [<mode>]

パラメータ	型	説明
<start>	**数値**	**開始アドレス**
<end>	**数値**	**終了アドレス（この値を含む）**
<destination>	**数値**	**比較先開始アドレス**
<mode>	**キーワード**	**フォーマット（任意、デフォルト**=byte**）**
	byte	**バイト**
	word	**ワード**
	long	**ロングワード**
	ascii	ASCII
	single	**単精度浮動小数点**
	double	**倍精度浮動小数点**

例:

MEMORY_COMPARE H’1000 H’1FFF H’2000 LONG	H’1000 番地から H’1FFF 番地のメモリ内容と H’2000 番地から始まるメモリ内容をロングワードサイズで比較します

5.50 MEMORY_DISPLAY

短縮形: MD

説明:

メモリ内容を表示します。

シンタックス

md <address> [<length>] [<mode>]

パラメータ	型	説明
<address>	**数値**	**開始アドレス**
<length>	**数値**	**長さ（任意、デフォルト**=H'100 **バイト）**
<mode>	**キーワード**	**フォーマット（任意、デフォルト**=byte**）**
	byte	**バイトとして表示します**
	word	**ワードとして表示します（**2 **バイト）**
	long	**ロングワードとして表示します（**4 **バイト）**
	ascii	ASCII **として表示します**
	single	**単精度浮動小数点として表示します**
	double	**倍精度浮動小数点として表示します**

例:

MEMORY_DISPLAY H’C000 H’100 WORD	H'C000 から始まるメモリを H'100 バイト分ワードフォーマットで表示します
MEMORY_DISPLAY H’1000 H’FF	H'1000 から始まるメモリを H'FF バイト分バイトフォーマットで表示します

5.51 MEMORY_EDIT

短縮形: ME

説明:

メモリ内容を変更します。 メモリを変更するとき、現在位置は ASSEMBLE コマンドで記述したときと同じような方法で変更することができます。"." を入力すると変更モードを終了し、"^" は1データ分戻り、空白は変更せずに進みます。

シンタックス

me <address> [<mode>] [<verify>]

パラメータ	型	説明
<address>	**数値**	**変更するアドレス**
<mode>	**キーワード**	**フォーマット（任意、デフォルト**=byte**）**
	byte	**バイトとして変更します**
	word	**ワードとして変更します**
	long	**ロングワードとして変更します**
	ascii	ASCII **として変更します**
	single	**単精度浮動小数点として表示します**
	double	**倍精度浮動小数点として表示します**
<verify>	**キーワード**	**ベリファイフラグ（任意、デフォルト**=V)
	V	**ベリファイあり**
	N	**ベリファイなし**

例:

ME H’1000 WORD H'1000 から word フォーマットでメモリ内容を変更します(ベリファイあり)

5.52 MEMORY_FILL

短縮形: MF

説明:

メモリ領域を指定したデータ値に変更します。

シンタックス

mf <start> <end> <data> [<mode>] [<verify>]

パラメータ	型	説明
<start>	**数値**	**開始アドレス**
<end>	**数値**	**終了アドレス**
<data>	**数値**	**データ値**
<mode>	**キーワード**	**データサイズ（任意、デフォルト**=byte**）**
	byte	**バイト**
	word	**ワード**
	long	**ロングワード**
	single	**単精度浮動小数点**
	double	**倍精度浮動小数点**
<verify>	**キーワード**	**ベリファイフラグ（任意、デフォルト**=V)
	V	**ベリファイあり**
	N	**ベリファイなし**

例:

MEMORY_FILL H’C000 H’C100 H’55AA WORD　　H'C000 から H'C0FF までワードデータ H'55AA に変更します

5.53 MEMORY_FIND

短縮形: MI

説明:

メモリ範囲内でデータを検索します。

シンタックス

mi <start> <end> <string> [<mode>]

パラメータ	型	説明
<start>	**数値**	**開始アドレス**
<end>	**数値**	**終了アドレス**
<string>	**数値**	**検索データ**
<mode>	**キーワード**	**データサイズ（任意、デフォルト**=byte**）**
	byte	**バイト**
	word	**ワード**
	long	**ロングワード**

例:

MEMORY_FIND H’1000 H’1FFF H’12　　H’1000-H’1FFF 内でデータ H’12 をバイトサイズで検索します

5.54 MEMORY_MOVE

短縮形: MV

説明:

指定したメモリ内容を移動します。

シンタックス

mv <start> <end> <destination> [<verify>] [<mode>]

パラメータ	型	説明
<start>	**数値**	**開始アドレス**
<end>	**数値**	**終了アドレス（この値を含む）**
<destination>	**数値**	**移動先開始アドレス**
<verify>	**キーワード**	**ベリファイフラグ（任意、デフォルト**=V)
	V	**ベリファイあり**
	N	**ベリファイなし**
<mode>	**キーワード**	**メモリリード・ライト時のフォーマット（任意、デフォルト**=byte）
	byte	**バイト**
	word	**ワード**
	long	**ロングワード**
	ascii	ASCII
	single	**単精度浮動小数点**
	double	**倍精度浮動小数点**

例:

MEMORY_MOVE H’1000 H’1FFF H’2000　　領域 H'1000 - H'1FFF を H'2000 へ移動します

5.55 MEMORY_TEST

短縮形: MT

説明:

指定したアドレス範囲で、リード・ライト・ベリファイのテストを行います。このときデータを書き換えます。マップ設定に従ってメモリテストを行います。本シミュレータ・デバッガではサポートしていません。

シンタックス

mt <start> <end>

パラメータ	型	説明
<start>	**数値**	**開始アドレス**
<end>	**数値**	**終了アドレス（この値を含む）**

例:

MEMORY_TEST H’8000 H’BFFF　　H'8000 から H'BFFF までテストします

5.56 OPEN_WORKSPACE

短縮形: OW

説明:

ワークスペースを開きます。

シンタックス

ow <filename>

パラメータ	型	説明
<filename>	文字列	ワークスペースファイル名

例:

OW WKSP.HWS ” WKSP.HWS”を開きます

5.57 P_CLOCK_RATE

短縮形: PCR

説明:

外部クロックと周辺クロックの比を設定します。H8SX のみサポートしています。

シンタックス

pcr [<rate>]

パラメータ	型	説明
なし		現在の周辺クロック比を表示します
<rate>	数値	外部クロックを 1 とした時の内部クロック比 設定できる値は 1、2、3、4、6、8、12、16、24、32 です

例:

P_CLOCK_RATE 6 周辺クロック比を 1：6 に設定します

5.58 PROFILE

短縮形: PR

説明:

プロファイラの有効／無効表示、設定、およびプロファイラ情報をクリアします。

シンタックス

pr [<state>]

パラメータ	型	説明
なし		**プロファイラ情報を表示します**
<state>	**キーワード**	**プロファイラの有効／無効表示、切替、クリアを行います**
	enable	**プロファイラを有効にする**
	tree-off	**プロファイラを有効にしますが、プロファイラ情報測定時に関数呼び出しをトレースしません**
	disable	**プロファイラを無効にする**
	reset	**プロファイラ情報をクリアする**

例:

PROFILE ENABLE	プロファイラを有効にします
pr r	プロファイラ情報をクリアします

5.59 PROFILE_DISPLAY

短縮形: PD

説明:

プロファイラ情報を表示します。

シンタックス

pd [<mode>] [<state1>] [<state2>] [<count>]

パラメータ	型	説明
<mode>	**キーワード**	**プロファイラ情報の表示方法を指定します（任意、デフォルト＝**list**）**
	tree	**ツリー形式で表示します**
	list	**リスト形式で表示します**
<state1>	**キーワード**	**親関数のサイクル情報に子関数の情報を含むかを指定します（任意、デフォルト＝**n**）**
	i	**子関数のサイクルも含めて表示します**
	n	**子関数のサイクルは含めずに表示します**
<state2>	**キーワード**	**未実行関数の表示を抑制するかを指定します（任意、デフォルト＝**a**）**
	e	**実行関数のみを表示します**
	a	**すべての関数を表示します**
<count>	**数値**	**表示する関数呼び出しネストレベルを指定します。**<mode>**パラメータが**'tree'**の場合のみ指定可能です（任意、デフォルト＝**16**）**

例:

PROFILE_DISPLAY TREE I	ツリー形式で、子関数の情報を含めてプロファイラ情報を表示します
pd	リスト形式で、子関数の情報を含めずにプロファイラ情報を表示します

5.60 PROFILE_SAVE

短縮形: PS

説明:

プロファイラ情報をファイルに保存します。ファイル拡張子は".PRO"をデフォルトとします。

シンタックス

ps [<filename>]

パラメータ	型	説明
なし		すべてのダウンロードモジュールのプロファイラ情報をファイルにセーブします
<filename>	文字列	プロファイラ情報を出力するファイル名を指定します

例:

PROFILE_SAVE PR_INFO　　プロファイラ情報を PR_INFO.PRO というファイルに出力します

5.61 QUIT

短縮形: QU

説明:

HEW を終了します。 オープンしていたログファイルはクローズします。

シンタックス

qu

パラメータ	型	説明
なし		HEW を終了します

例:

QU　　HEW を終了します

5.62 RADIX

短縮形: RA

説明:

デフォルトの基数を設定、または表示します。 パラメータなしで基数を表示します。
基数は数値データの前の B'/H'/D'/O' を使って変更できます。

シンタックス

ra [<radix>]

パラメータ	型	説明
なし		**現在の基数を表示します**
<radix>	**キーワード**	**基数指定子**
	H	16 **進数**
	D	10 **進数**
	O	8 **進数**
	B	2 **進数**

例:

RADIX 現在の基数を表示します
RA H 基数を 16 進数にします

5.63 REGISTER_DISPLAY

短縮形: RD

説明:

レジスタ値を表示します。

シンタックス

rd [<register>]

パラメータ	型	説明
なし		**全レジスタの内容を表示します**
<register>	**キーワード**	**レジスタ名**

例:

RD 全レジスタの内容を表示します
RD R0 R0 の内容を表示します

5.64 REGISTER_SET

短縮形: RS

説明:

レジスタ値を変更します。

シンタックス

rs <register> <value> [<mode>]

パラメータ	型	説明
<register>	**キーワード**	**レジスタ名**
<value>	**数値**	**レジスタ値**
<mode>	**キーワード**	**データサイズ（任意、デフォルト=レジスタサイズ）**
	byte	**バイト**
	word	**ワード**
	long	**ロングワード**
	single	**単精度浮動小数点**
	double	**倍精度浮動小数点**

例:

RS PC _StartUp　　プログラムカウンタをシンボル_StartUp に設定します
RS R0 H'1234 WORD　　R0 にワードデータ H'1234 に設定します

5.65 REMOVE_FILE

短縮形: REM

説明:

指定ファイルをカレントプロジェクトから削除します。

シンタックス

rem <filename>

パラメータ	型	説明
<filename>	**文字列**	**ファイル名**

例:

REMOVE_FILE $(PROJDIR)¥¥sbrk.c　　sbrk.c ファイルをプロジェクトから削除します
ファイル指定にプレースホルダが使用できます

5.66 RESET

短縮形: RE

説明:

プロセッサをリセットします。 すべてのレジスタ値はデバイスの初期化状態となります。
メモリマッピングとブレークポイントには影響しません。

シンタックス

re

パラメータ	型	説明
なし		**プロセッサをリセットします**

例:

RE　　プロセッサをリセットします

5.67 RESPONSE

短縮形: RP

説明:

ウィンドウをリフレッシュするタイミングを指定します。
リフレッシュ間隔を長くするとシミュレーション速度は上がりますが、ブレークボタン等の反応が遅くなります。お使いのマシンに合わせて設定してください。

シンタックス

rp　[<instruction number>]

パラメータ	型	説明
<instruction number>	**数値**	1~D'2,147,483,647**（任意、デフォルト**=40000） **何命令実行ごとにウィンドウをリフレッシュするかを指定します**

例:

RESPONSE 9　　9 命令実行ごとにウィンドウをリフレッシュします

5.68 SAVE_SESSION

短縮形: SE

説明:

現在のセッションをセーブします。

シンタックス

se [<session>]

パラメータ	型	説明
なし		**カレントプロジェクトのすべてのセッションをセーブします**
<session>	**文字列**	**セッション名**

例:

SE **カレントプロジェクトのすべての**セッションをセーブします

SE DefaultSession ”Default Session”をセーブします

5.69 SLEEP

短縮形: なし

説明:

指定したミリ秒の間コマンド実行を遅延します。

シンタックス

sleep <milliseconds>

パラメータ	型	説明
<milliseconds>	**数値**	**遅延時間（ミリ秒）**

基数は、10 進数固定です。

例:

SLEEP D’9000 9 秒間遅延します

5.70 STATUS

短縮形: STA

説明:

デバッグプラットフォームの現在の状態を表示します。表示内容は[ステイタス]ウィンドウの[Platform]シートと同じです。

シンタックス

sta

パラメータ	型	説明
なし		現在のデバッグプラットフォーム状態を表示します

例:

STA 現在のデバッグプラットフォーム状態を表示します

5.71 STEP

短縮形: ST

説明:

シングルステップ（ソース行、または命令）を行います。現在の PC から指定した命令数分ステップします。ソースデバッグが有効な時、デフォルトのステップはソース行となります。ステップ数のデフォルトは1です。

シンタックス

st [<instruction>] [<count>]

パラメータ	型	説明
<instruction>	キーワード	ステップの種類（任意）
	instruction	アセンブラの 1 命令を基準にステップします
	line	ソースコードの 1 行を基準にステップします
<count>	数値	ステップ数（任意、デフォルト=1）

例:

STEP 9 9 ステップコードをステップします

5.72 STEP_MODE

短縮形: SM

説明:

ステップモードを選択します。

シンタックス

sm [<mode>]

パラメータ	型	説明
<mode>	**キーワード**	**ステップモードの選択（デフォルト =ステップモードを表示）**
	Auto	**自動選択**
	Assembly	**アセンブラの 1 命令を基準にステップします**
	Source	**ソースコードの 1 行を基準にステップします**

例:

STEP_MODE auto ステップモードを自動選択に設定します

[エディタ]ウィンドウが現在アクティブならば、ソースコードの 1 行を基準にステップ実行します

[逆アセンブリ]ウィンドウが現在アクティブならば、アセンブラの 1 命令を基準にステップ実行します

5.73 STEP_OUT

短縮形: SP

説明:

現在の関数の外へプログラムをステップします（即ち、ステップアップ）。 アセンブラとソースレベルデバッグの両方に有効です。

シンタックス

sp

パラメータ	型	説明
なし		**ステップアップします**

例:

SP 現在の関数の外へプログラムをステップします

5.74 STEP_OVER

短縮形: SO

説明:

現在の PC から指定した命令数分ステップします。
このコマンドはサブルーチン、または割り込みルーチンの中でステップしないという点でSTEPとは異なります。フルスピードで実行します。

シンタックス

so [<instruction>] [<count>]

パラメータ	型	説明
<instruction>	**キーワード**	**ステップの種類（任意）**
	instruction	**アセンブラの** 1 **命令を基準にステップします**
	line	**ソースコードの** 1 **行を基準にステップします**
<count>	**数値**	**ステップ数（任意、デフォルト**=1**）**

例:

SO 1 ステップコードをステップオーバーします

5.75 STEP_RATE

短縮形: SR

説明:

STEPとSTEP_OVERコマンドでステップの速度をコントロールします。

シンタックス

sr [<rate>]

パラメータ	型	説明
なし		**ステップレートを表示します**
<rate>	**数値**	**ステップレート** 0 **から** 6 **で** 6 **が最も速くなります** 0 : 3 seconds 1 : 2.5 seconds 2 : 2 seconds 3 : 1.5 seconds 4 : 1 seconds 5 : 0.5 seconds 6 : 0 seconds

例:

SR 現在設定しているステップレートを表示します
SR 6 ステップレートを最速にします

5.76 SUBMIT

短縮形: SU

説明:

コマンドファイルを処理します。 処理するファイル中でもこのコマンドを使用できます。
エラーが発生するとファイルの処理を中止します。

シンタックス

su <filename>

パラメータ	型	説明
<filename>	文字列	ファイル名

例:

SUBMIT COMMAND.HDC　　COMMAND.HDC ファイルを処理します
SU A:SETUP.TXT　　ドライブ A: の SETUP.TXT ファイルを処理します

5.77 SYMBOL_ADD

短縮形: SA

説明:

新しいシンボルを追加するか、または存在しているシンボルを変更します。

シンタックス

sa <name> <address>

パラメータ	型	説明
<name>	文字列	シンボル名
<address>	数値	値

例:

SYMBOL_ADD start H'1000　　H'1000 に start を定義します
SA END_OF_TABLE 1FFF　　H'1FFF に END_OF_TABLE を定義します

5.78 SYMBOL_CLEAR

短縮形: SC

説明:

シンボルを削除します。 パラメータを指定しないとすべてのシンボルを削除します。

シンタックス

sc [<name>]

パラメータ	型	説明
なし		すべてのシンボルを削除します
<name>	文字列	シンボル名

例:

SYMBOL_CLEAR　　すべてのシンボルを削除します
SC start　　シンボル start を削除します

5.79 SYMBOL_LOAD

短縮形: SL

説明:

ファイルからシンボルをロードします。 ファイルは XLINK Pentica-b フォーマット（即ち "XXXXH name"）である必要があります。 シンボルをシンボルテーブルに加えます。

シンタックス
sl <filename>

パラメータ	型	説明
<filename>	文字列	ファイル名

例:

SYMBOL_LOAD TEST.SYM　　ファイル TEST.SYM をロードします

5.80 SYMBOL_SAVE

短縮形: SS

説明:

すべてのシンボルをファイルへ保存します。ファイルは XLINK Pentica-b フォーマットで、シンボルファイル拡張子は".SYM" をデフォルトとします。

シンタックス
ss <filename>

パラメータ	型	説明
<filename>	文字列	ファイル名

例:

SYMBOL_SAVE TEST　　TEST.SYM にシンボルテーブルを保存します

5.81 SYMBOL_VIEW

短縮形: SV

説明:

定義したすべてのシンボル、または指定した文字列（大文字／小文字は区別する）を含んでいるシンボルを表示します。

シンタックス

sv [<pattern>]

パラメータ	型	説明
なし		**すべてのシンボルを表示します**
<pattern>	**文字列**	**表示するシンボルに含まれる文字列**

例:

SYMBOL_VIEW BUFFER	BUFFER を含んでいるすべてのシンボルを表示します
SV	すべてのシンボルを表示します

5.82 TCL

短縮形: なし

説明:

TCLを有効、または無効にします。

シンタックス

tcl [<state>]

パラメータ	型	説明
なし		TCL の情報を表示します
<state>	キーワード	TCL の有効／無効を設定します
	enable	TCL を有効にします
	disable	TCL を無効にします

例:

TCL	TCL の情報を表示します
TCL enable	TCL を有効にします
TCL d	TCL を無効にします

5.83 TIMER

短縮形: TMR

説明:

擬似タイマを有効、または無効にします。H8SX のみサポートしています。

シンタックス

tmr [<mode>]

パラメータ	型	説明
なし		擬似タイマの情報を表示します
<mode>	キーワード	擬似タイマの有効／無効を設定します
	E	擬似タイマを有効にします
	D	擬似タイマを無効にします

例:

TMR	擬似タイマの情報を表示します
TMR E	擬似タイマを有効にします
TMR D	擬似タイマを無効にします

5.84 TOOL_INFORMATION

短縮形: TO

説明:

現在登録されているツールの情報をファイル（HTML形式）で出力します。

シンタックス

to <file name>

パラメータ	型	説明
<file name>	文字列	出力するファイル名

例:

to information.htm	ツール情報を information.htm ファイルに出力します。

5.85 TRACE

短縮形: TR

説明:

トレースバッファの内容を表示します。バッファでの最初の（最も以前に実行した）レコードは0で、それ以降のレコードは正のオフセット値を持っています。

シンタックス

tr [[<start rec> [<count>]] | [<clear>]]

パラメータ	型	説明
<start rec>	**数値**	**表示を始めるレコード（任意、デフォルト=最も新しいレコード-9）**
<count>	**数値**	**表示するレコード数（任意、デフォルト**= 10**）**
<clear>	**キーワード**	**すべてのトレースレコードクリア（任意）**
	clear	**すべてのトレースレコードクリア**

<start rec> に負の値(-0 は指定できません)を指定した場合は、PTR 値での指定になります。

例:

TR 0 20	トレースバッファ先頭から 20 行の内容を表示します
TR	トレースバッファ最後の 10 行(最近実行した 10 行)の内容を表示します
TR –20 10	トレースバッファの、PTR が–20 から-11 の内容を表示します
TR C	すべてのトレースレコードをクリアします

5.86 TRACE_ACQUISITION

短縮形: TA

説明:

トレース情報取得の有効/無効を切り替えます。

シンタックス

ta <mode> [<trace full handling>] [<trace capacity>]

パラメータ	型	説明
<mode>	**キーワード**	**トレース情報取得の有効/無効**
	E	**有効**
	D	**無効**
<trace full handling>	**キーワード**	**トレースバッファ満杯時の処理**
	C	**続行 (デフォルト)**
	B	**停止**
<trace capacity>	**キーワード**	**トレースバッファ容量**
	BUF=1	D'1024 **レコード**
	BUF=4	D'4096 **レコード**
	BUF=16	D'16384 **レコード**
	BUF=32	D'32768 **レコード**

【注】 トレースバッファ容量はトレースバッファが空のときのみ変更できます。

例:

TRACE_ACQUISITION E B BUF=32	トレース情報の取得を有効にします トレースバッファ満杯時は停止します トレースバッファサイズを 32768 とします
TA D	トレース情報の取得を無効にします

5.87 TRACE_SAVE

短縮形: TV

説明:

トレース情報をファイルに保存します。ファイルはテキスト形式で、ファイル拡張子は".TXT"をデフォルトとします。

シンタックス

tv <filename> [<mode>]

パラメータ	型	説明
<filename>	**文字列**	**ファイル名**
<mode>	**キーワード**	**ファイル書き込みモード**
	A	**ファイルに追加書き込み**
	O	**ファイルに上書き (デフォルト)**

例:

TRACE_SAVE TEST	TEST.TXT にトレース情報を保存します
TV TRACE.TXT	TRACE.TXT にトレース情報を保存します

5.88 TRACE_STATISTIC

短縮形: TST

説明:

指定された条件で統計情報解析を行います。

シンタックス

tst <item> <string>

パラメータ	型	説明
<item>	文字列	統計情報解析項目
<string>	文字列	条件文字列

例:

TST CODE1 E630	条件 CODE1=E630 で統計情報解析を行います

5.89 TRAP_ADDRESS

短縮形: TP

説明:

ユーザプログラムから標準入出力およびファイル入出力を利用するためのシステムコールアドレスを設定します。詳細は、「4.23　標準入出力およびファイル入出力を行う」を参照してください。

シンタックス

tp <address>

パラメータ	型	説明
<address>	数値	システムコールアドレス

例:

TP 1000	システムコールアドレスとして 1000 を設定します。

5.90 TRAP_ADDRESS_DISPLAY

短縮形: TD

説明:

ユーザプログラムから標準入出力およびファイル入出力を利用するためのシステムコールアドレスの現在の値を表示します。詳細は、「4.23 標準入出力およびファイル入出力を行う」を参照してください。

結果の表示形式は以下のとおりです。

H'xxxxxxxx Y

ここで、

H'xxxxxxxx システムコールアドレス (16 進数 8 桁表示)

Y システムコール有効時：'E'、無効時：'D'

シンタックス

td

パラメータ	型	説明
なし		システムコールアドレスの現在の値、および システムコール有効／無効を表示します

例:

TD システムコールアドレス値とシステムコール有効／無効を表示します。

5.91 TRAP_ADDRESS_ENABLE

短縮形: TE

説明:

ユーザプログラムから標準入出力およびファイル入出力を利用するためのシステムコールの有効／無効を設定します。詳細は、「4.23 標準入出力およびファイル入出力を行う」を参照してください。

シンタックス

te <mode>

パラメータ	型	説明
<mode>	キーワード	システムコールの有効／無効を指定します
	E	システムコールを有効にします
	D	システムコールを無効にします

例:

TE E システムコールを有効にします。

5.92 UPDATE_ALL_DEPENDENCIES

短縮形: UD

説明:

カレントプロジェクトの依存関係を更新します。

シンタックス

ud

パラメータ	型	説明
なし		**カレントプロジェクトのすべての依存関係を更新します**

例:

UD	カレントプロジェクトのすべての依存関係更新処理を開始します

6. メッセージ一覧

6.1 インフォメーションメッセージ

シミュレータ・デバッガは実行経過をユーザに知らせるため、インフォメーションメッセージを出力します。シミュレータ・デバッガの出力するインフォメーションメッセージを表 6-1に示します。

表 6-1 インフォメーションメッセージ一覧

メッセージ	内容
Break Access	**ブレークアクセス条件が成立して実行を中断しました。**
Break Cycle	**ブレークサイクル条件が成立して実行を中断しました。**
Break Data	**ブレークデータ条件が成立して実行を中断しました。**
Break Register	**ブレークレジスタ条件が成立して実行を中断しました。**
Break Sequence	**ブレークシーケンス条件が成立して実行を中断しました。**
PC Breakpoint	**ブレークポイント条件が成立して実行を中断しました。**
Sleep	SLEEP **命令により実行を中断しました。**
Step Normal End	**ステップ実行が正常に終了しました。**
Stop	[STOP]**ボタンにより実行を中断しました。**
Trace Buffer Full	[**トレース取得**]**ダイアログボックスの**[**トレースバッファ満杯時の動作**]**で**[**停止**]**モードを選択しており、かつトレースバッファが満杯となったので実行を中断しました。**

6.2 エラーメッセージ

シミュレータ・デバッガはユーザプログラムや操作の誤りをユーザに知らせるため、エラーメッセージを出力します。シミュレータ・デバッガの出力するエラーメッセージを表 6-2に示します。

表 6-2 エラーメッセージ一覧

メッセージ	内容・対策
Address Error	以下のいずれかの状態になりました。 （1）PC 値が奇数である （2）内蔵 I/O 空間から命令読み出しを行おうとした （3）ワードデータを（2n）番地以外からアクセスしようとした （4）ロングワードデータを（2n）番地以外からアクセスしようとした エラーが発生しないようにユーザプログラムを修正してください。
Exception Error	例外処理でエラーが発生しました。 エラーが発生しないようにユーザプログラムを修正してください。
File Open Error	Break の FileInput/Output アクションでファイルオープンに失敗しました。 ファイル指定を見直してください。
File Input Error	Break の File Input アクションでファイル読込みに失敗しました。 ファイル指定を見直してください。
File Output Error	Break の File Output アクションでファイル書出しに失敗しました。 ファイル指定を見直してください。
Illegal Instruction	以下のいずれかの状態になりました。 （1）命令ではないコードを実行しようとした （2）MOV.B Rn,@-SP または、MOV.B @SP+,Rn を実行しようとした エラーが発生しないようにユーザプログラムを修正してください。
Illegal Operation	以下のいずれかの状態になりました。 （1）DAA 命令、DAS 命令で CCR の C フラグ、H フラグと補正前の値の関係が不正である （2）DIVXU 命令、DIVXS 命令でゼロ除算または、オーバフローが発生した エラーが発生しないようにユーザプログラムを修正してください。
Memory Access Error	以下のいずれかの状態になりました。 （1）確保していないメモリ領域をアクセスしようとした （2）書き込み不可属性を持つメモリへの書き込みを行おうとした （3）読み出し不可属性を持つメモリからの読み出しを行おうとした （4）メモリが存在しない領域をアクセスしようとした （5）EEPMOV 命令以外で EEPROM へ書き込みを行おうとした メモリの確保、属性変更を行うか、当該メモリアクセスが発生しないようにユーザプログラムを修正してください。
System Call Error	システムコールエラーが発生しました。 レジスタ R0,R1 およびパラメータブロックの内容の誤りを修正してください。
I/O area not exist	I/O 領域を削除しました。 I/O 領域は必ず設定してください。
The memory resource has not been set up	メモリマップ範囲外にメモリリソースを設定しました。 エラーが発生しないようにメモリリソース設定を修正してください。

付録 A　トラブルシューティング

? “ビルド中止”ボタンをクリックするか [ビルド->ビルドの中止]を選んでビルドを中止しようとしたが、ビルドの実行が中止できない

HEWは現在のファイルをビルド実行（または現在のフェーズを実行）後、ビルド実行を中止します。もしも、プロジェクトビルダから長い間応答がない場合、 [ビルド->ツールの終了] を選んでください。現在の処理を強制終了します。強制終了したツールから出力されたファイルは無効です。それらの出力ファイルを削除してフェーズを再実行してください。

? エディタにテキストファイルが表示されているが、シンタックス色付けが表示されない

ファイルに名前が付いている（保存した）ことを確認してください。また、[ツール->オプション...]を選んで“オプション” ダイアログボックスを開き、“エディタ”タブの“シンタックスカラーリング” チェックボックスがチェックされていることを確認してください。HEWではファイルの拡張子の属するファイルグループを調べてファイルを色付けするかどうか判断します。現在定義されている拡張子とファイルグループを参照するには、 [プロジェクト ->ファイルの拡張子...]を選んで“ファイル拡張子”ダイアログボックスを表示してください。色付け情報を確認するには、[ツール-> 表示形式...]を選んで“表示形式”ダイアログボックスの“カラー” タブを参照してください。詳細は、4章, 「エディタの使用」のシンタックスの色付けに関する説明をお読みください。

? ツールの設定を変えたいが、[ツール->アドミニストレーション…] メニューオプションを選べない

ワークスペースを開いている間は[ツール->アドミニストレーション...] を選ぶことはできません。“ツールアドミニストレーション”ダイアログボックスを開くには、現在のワークスペースを閉じてください。

? 日本語版 Windows® 98/Me 、Windows NT®、Windows® 2000 上で日本語入力ができない、または日本語の文字が正しく表示されない

[ツール->表示形式…] を選んで“フォント”タブをクリックして、“フォント”フィールドで日本語のフォントを選んでください。

? プロジェクトファイルの編集をしていないのに、[ビルド-> ビルド] を選んだらいくつかのファイルが再ビルドされた

ファイルは以下のいずれかの条件で再ビルドされます。

- ビルド後に現在のファイルのオプションが変更されたとき
- 出力ファイルのいずれかがないとき
- いずれかのソースファイルの日付がそのフェーズの出力ファイルの日付より新しいとき
- いずれかの依存ファイルの日付がそのフェーズの出力ファイルの日付より新しいとき
- ユーザ定義のビルドフェーズで、“実行前に入力ファイルが存在するか否かのチェックを行わない”チェックボックスがチェックされているとき。このチェックボックスを参照するには、[オプション->ビルドフェーズ…]を選び、“ビルド順序”タブの“ビルドフェーズの順序”リストからフェーズを選び、“変更…”ボタンをクリックしてください。すると、“コマンド”タブで“実行前に入力ファイルが存在するか否かのチェックを行わない”チェックボックスを参照できます。
- 最適化リンケージエディタでサブコマンドファイルが選択されているとき

? プロジェクトのファイルを一時的にビルドから除外したい。

ワークスペースウィンドウの“Projects”タブのファイル上でマウスの右ボタンを押下し、[ビルドから除外 <file>]を選択してください。すると、そのファイルがビルドから除外されます。再びファイルをビルドに戻すには、ワークスペースウィンドウの“Projects”タブの当該ファイル上でマウスの右ボタンを押下し、[ビルドから除外の解除 <file>]を選択してください。

? 自分の PC でワークスペースを開いた。同時に、他の人が他の PC から同じワークスペースを開いた。自分でワークスペースの設定を変えて保存した。その後、他の人がワークスペースの設定を変えて保存した。自分が再びワークスペースを開くと、設定が自分の行った設定とは異なっていた。

最後に保存した設定が有効です。HEWはワークスペースを開くとメモリ内で更新します。ユーザが意識的に設定をファイルに保存しない限り、設定はファイルに保存されません。

? プロジェクトにファイルを追加するとき、ファイルに自動的にデフォルトオプションを指定したい

フェーズが複数ビルドフェーズであれば、フェーズにデフォルトオプションを指定できます。[オプション] メニューからフェーズを選んでください。フェーズが複数ビルドフェーズの場合、オプションダイアログボックスの左側にファイルのリストが表示されます（図 A.1）。 ファイルリストで、デフォルトオプションを指定したいファイルグループのフォルダを開いてください。 フォルダ内に“Default Options” アイコンが表示されます。アイコンを選択して、オプションダイアログボックスの右側でオプションを指定して“OK ” ボタンをクリックしてください。このオプションは、プロジェクトにそのファイルグループのファイルを初めて追加するときに適用されます。

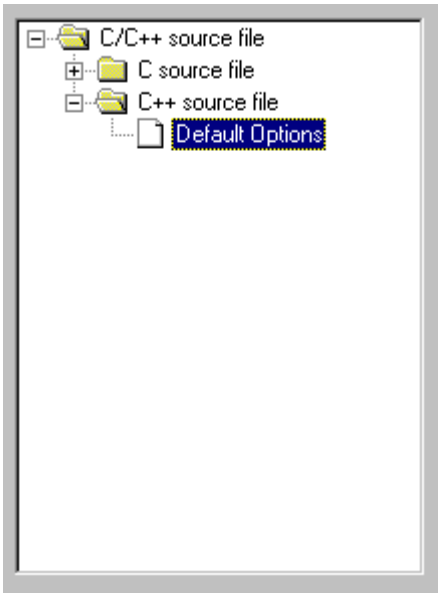


図 A.1: **オプションダイアログボックス、ファイルリスト**

? [ビルド>ビルド]を選択するとワークスペースウィンドウの “Projects”タブのファイル依存関係が更新されるが、[ビルド-> すべてをビルド]を選択してもファイル依存関係が更新されない。

[ビルド->すべてをビルド]はファイルの依存関係を更新しません。ファイル保存関係を更新するには[ビルド->すべての依存関係を更新]を選択してください。

付録 B　正規表現

HEW エディタでは検索・置換操作の際、文字列に特殊文字を指定できます。指定できる特殊文字を表 B.1 に示し、その詳細を以下に示します。

表 B.1: 正規表現の文字

文字	意味
+	1 個以上の指定文字（ブラケット表現を除く）に一致します。例えば、a+は a、aa、aaa などに一致します。
*	0 個以上の指定文字（ブラケット表現を除く）に一致します。例えば、a*は空白文字列、a、aa などに一致します。
?	0 個以上の指定文字（ブラケット表現を除く）に一致します。例えば、a?は空白文字列と a に一致します。
{ }	{m,n}の形式で濃度の範囲を指定します。この構文は、m 個から n 個の指定文字に一致します。例えば、a{2,3}は aa と aaa に一致します。 この構文はまた、{m.}や{m}を使用して構成することもできます。1 つ目は m 個以上の指定文字に一致します。例えば、a{2.}は aa、aaa、aaaa などに一致します。2 つ目はちょうど m 個の指定文字に一致します。例えば、a{2}は aa に一致します。
[ ]	ブラケット表現を作成します。ブラケット表現は、どれもが一致するような文字セットを作成します。例えば、[abc]は a、b、または c に一致します。 ブラケット表現では、以下を除き、あらゆる正規表現の特殊文字は通常の文字として扱われます。 - は、ビットパターンに基づき文字値の範囲を指定します。例えば、[A-Za-z]はすべての大文字および小文字の英文字に一致します。- をブラケット表現の文字として表すには、文字セットの最初または最後の文字でなくてはなりません（例：[-a-z]、[A-Z-]）。 ^ は、かっこ内の最初の文字にあるときだけ特別です。^ を最初の位置に置くと、文字セットを補って一致させます。例えば、[^a-z]は小文字の英文字を除くすべての文字に一致します。 かっこ内の文字として] を含めるには、文字セットの最初の文字として含めなくてはなりません（例：[]abc]または[^]abc]）。
( )	正規表現を、一単位として扱うことのできる部分正規表現に分類します。例えば、ab*は a、ab、abb などに一致しますが、(ab)*は空白文字列、ab、abab などに一致します。(と) は、ブラケット表現の中では特殊文字として扱われません。
¥	正規表現文字を無視し、通常の文字として扱うようにします。例えば、(ab)は ab から成る部分正規表現を表しますが、¥(ab¥)は連続した文字（ 、a、b、 ）を示します。 注意：C++ソースプログラムで¥の文字を指定する場合、¥¥と指定する必要があります。これは、C++コンパイラが¥を C++ソースプログラムに実装したエスケープシーケンスの始まりを示す特殊文字として扱うためです。ただし、データファイルまたはダイアログボックスのテキストコントロールでは、二重のバックスラッシュは必要ありません。
^	入力文字列の先頭に正規表現または部分正規表現があることを示します。例えば、^ab は ab と abc に一致しますが、cab には一致しません。^ の扱いがブラケット表現では異なることに注意してください。
$	入力文字列の最後に正規表現または部分正規表現があることを示します。例えば、ab$は ab と cab に一致しますが、abc には一致しません。
|	同等に有効な選択肢である表現または部分表現の交代、または作成を示し、それぞれが一致します。例えば、ab|cd は ab または cd に一致します。
.	行の論理の終わりを示すものを除き、どのようなコード単位にも一致します。

付録 C　プレースホルダ

プレースホルダは、HEW の複数のツールによって提供される機能です。この章ではプレースホルダの使い方を説明します。

C.1　プレースホルダとは?

プレースホルダとは一時的にテキストに挿入される特殊文字列です。後に実際の値に置き換えます。例えば、HEW のプレースホルダのひとつに、$(FULLFILE)があります。これは、すべてのパス付きのファイルを示します。パラメータとしてファイルを編集できるエディタが c:¥myedit¥myeditor.exe であるとします。c:¥files ディレクトリのファイル FILE1.C を開き、このエディタを起動するには、以下のように直接指定することもできます。

```
c:¥myedit¥myeditor.exe c:¥files¥file1.c
```

しかし、このエディタを介して任意のファイルを開きたいとき、上記コマンドは“c:¥files¥file1.c”を開くだけのものであるため、問題が起きます。指定したエディタを使うときにその時点で選んだファイルを開くことができるようにするには、特定のファイル名を一般的なプレースホルダに置き換えます。以下に例を示します。

```
c:¥myedit¥myeditor.exe $(FULLFILE)
```

これで、HEW がエディタでファイルを開くとき、$(FULLFILE)を選ばれたファイル名で置き換えます。

C.2　プレースホルダを挿入する

プレースホルダは HEW の 3 つの編集フィールドで指定できます (図 C.1 、C.2、C.3)。プレースホルダの指定方法には 4 種類あります。

例 1

プレースホルダを挿入したい場所にカーソルを置いてください。次に、必要なプレースホルダを編集フィールドの右のポップアップメニューから選んでください。

図 C.1: **プレースホルダポップアップメニュー**

例 2

“Custom directory”以外のプレースホルダをドロップダウンリストボックスから選び、プレースホルダによって示されたディレクトリからの相対サブディレクトリを指定してください。 “Custom directory”を選んだ場合、 “Sub-Directory”フィールドに絶対ディレクトリパスを指定してください。

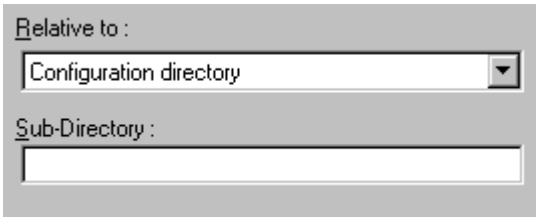


図 C.2: **プレースホルダドロップダウンリストと** Sub-Directory **フィールド**

例 3

プレースホルダを挿入したい場所にカーソルを置いてください。次に、必要なプレースホルダをドロップダウンリストボックスから選んでください。そして、“Insert”ボタンをクリックしてください。

図 C.3: **プレースホルダドロップダウンリストボックス**

例 4

フィールドにプレースホルダを直接入力してください。大文字で入力して、“$(” で始めて“) ”で終わってください。

正しい

```
$(FILEDIR)
```

誤り

```
$(Filedir)
$(  FILEDIR   )
$FILEDIR
```

C.3　使用できるプレースホルダ

表 C.1 にプレースホルダと意味を示します。

表 C.1: プレースホルダ

プレースホルダ	意味
$(FULLFILE)	ファイル名（フルパスを含む）
$(FILEDIR)	ファイルディレクトリ
$(FILENAME)	ファイル名（パスを除き拡張子を含む）
$(FILELEAF)	ファイル名（パスと拡張子を除く）
$(EXTENSION)	ファイルの拡張子
$(WORKSPDIR)	ワークスペースディレクトリ
$(WORKSPNAME)	ワークスペース名
$(PROJDIR)	プロジェクトディレクトリ
$(PROJECTNAME)	プロジェクト名
$(CONFIGDIR)	コンフィグレーションディレクトリ
$(CONFIGNAME)	コンフィグレーション名
$(HEWDIR)	HEW インストールディレクトリ
$(TCINSTALL)	ツールチェインインストールディレクトリ（オプションダイアログ上）
$(TOOLDIR)	ツールインストールディレクトリ（Tools Administration 上）
$(TEMPDIR)	テンポラリディレクトリ
$(WINDIR)	Windows® ディレクトリ
$(WINSYSDIR)	Windows® システムディレクトリ
$(EXEDIR)	コマンドディレクトリ
$(USERNAME)	ユーザログイン（バージョン管理）
$(PASSWORD)	ユーザパスワード（バージョン管理）
$(VCDIR)	「仮想的」バージョン管理ディレクトリ
$(COMMENT)	コメント（バージョン管理）
$(LINE)	エラー/ウォーニングの行番号

プレースホルダの使用例を以下に示します。

表 C.2: プレースホルダの展開（例）

プレースホルダ	プレースホルダの展開例
$(FULLFILE)	c:¥hew¥workspace¥project¥file.src
$(FILEDIR)	c:¥hew¥workspace¥project
$(FILENAME)	file.src
$(FILELEAF)	file
$(EXTENSION)	src
$(WORKSPDIR)	c:¥hew¥workspace
$(WORKSPNAME)	workspace
$(PROJDIR)	c:¥hew¥workspace¥project
$(PROJECTNAME)	project
$(CONFIGDIR)	c:¥hew¥workspace¥project¥debug
$(CONFIGNAME)	debug
$(HEWDIR)	c:¥hew
$(TCINSTALL)	c:¥hew¥toolchains¥renesas¥sh¥511
$(TOOLDIR)	c:¥hew¥toolchains¥renesas¥sh¥511
$(TEMPDIR)	c:¥Temp
$(WINDIR)	c:¥Windows
$(WINSYSDIR)	c:¥Windows¥System
$(EXEDIR)	v:¥vc¥win32
$(USERNAME)	JHARK
$(PASSWORD)	214436
$(VCDIR)	“c:¥project” **は** “x:¥vc¥project” **へマッピングされている**
$(COMMENT)	“Please Enter Comment” **ダイアログボックスが表示される**
$(LINE)	12

表 C.2 では以下のことを仮定しています。

- ファイルパスは “c:¥hew¥workspace¥project¥file.src”
- ワークスペース名 “workspace”の位置は “c:¥hew¥workspace”
- プロジェクト名 “project” の位置は “c:¥hew¥workspace¥project”
- コンフィグレーション名 “debug” にはコンフィグレーションディレクトリがあり、位置は “c:¥hew¥workspace¥project¥debug”
- HEW.EXE が“c:¥hew”にインストールされている
- ツールチェイン（コンパイラ、アセンブラ、リンケージエディタ）の*.HRF ファイル の位置は “c:¥hew¥toolchain¥renesas¥sh¥511”
  このディレクトリは[オプション]メニューのオプション設定ダイアログボックス上では $(TCTINSTALL)として参照され、“ツールアドミニストレーション”ダイアログボックス上では$(TOOLDIR)として参照される
- Windows® オペレーティングシステムが c:¥Windows にインストールされており、

Windows® システムファイルが c:¥Windows¥System にインストールされている
- バージョン管理実行可能パスが v:¥vc¥win32¥ss.exe である。バージョン管理システムにログインするユーザ名は JHARK でパスワードが 214436 であり、バージョン管理実行可能ファイルへのコマンドラインには$(COMMENT)が指定されている。c:¥project は[ツール->バージョン管理->構成...]で選ぶと表示される“Version Control Setup”ダイアログボックスの“Projects”タブの x:¥vc¥project にマッピングされている
- コンパイラまたはアセンブラのエラーが 12 行目で起きた

注意　どのフィールドでもすべてのプレースホルダを使用できるとは限りません。例えば、プレースホルダ$(LINE) **は依存ファイル位置を指定するときには意味をもちません。プレースホルダ**$(USERNAME)**、**$(PASSWORD)**、**$(VCDIR)**、**$(COMMENT)**はバージョン管理でのみ受け付けられます。各編集フィールドで、使用できないプレースホルダを指定すると、警告メッセージが表示される場合があります。**

C.4　プレースホルダを使うにあたって

プレースホルダによって、システムが使用する様々なファイルへのパスを任意に指定できます。

- パスまたはファイル名を入力する編集フィールドのとなりにプレースホルダのポップアップメニュー () がある場合、プレースホルダをどのように使ってパスやファイルの指定をフレキシブルにできるかご考慮ください。
- いくつかの構成を使うとき、プレースホルダ$(CONFIGDIR) を使うと、現在の構成のディレクトリからファイルへの書き込みやそのファイルから現在の構成のディレクトリへの書き込みができるので、便利です。
- できるだけプレースホルダを利用してください。プレースホルダは後で削除したり追加したりできるので、気軽に試すことができます。

付録 D　I/O ファイルフォーマット

HEWは、I/Oレジスタ定義ファイルで取得する情報に基づいて、[IO]ウィンドウをフォーマットします。デバッギングプラットフォームを選択すると、HEWは、選択したデバイスに対応する“<device>.IO”ファイルを検索し、存在する場合にはこのファイルをロードします。これは、I/Oモジュール、およびそのレジスタのアドレスやサイズを記述するフォーマット済みテキストファイルです。ユーザはテキストエディタでこのファイルを編集し、ユーザアプリケーションに特有のメモリマップレジスタや周辺レジスタ（例えば、マイコンのアドレス空間にマップしたASICデバイスのレジスタ）のサポートを追加することができます。

D.1　ファイルフォーマット

各モジュール名を[Modules]定義セクションで定義し、モジュールの番号を、順番に付けていなければなりません。各モジュールはレジスタ定義セクションに対応しており、セクション内のエントリは、I/Oレジスタを定義します。

[BaseAddress]はデバイスのための定義であり、そのデバイスでは、CPUモードによってアドレス空間のI/Oレジスタの場所が移動します。この場合、[BaseAddress]値は、ある特有モードのI/Oレジスタのベースアドレスです。また、レジスタ定義で使用するアドレスは、同じモードにおけるレジスタのアドレス位置です。I/Oレジスタファイルを実際に使用する場合、定義したレジスタアドレスから[BaseAddress]値を引き、その結果のオフセットを選択したモードのベースアドレスに加算します。

各モジュールにはセクションがあり、オプションの依存性によって形成するレジスタを定義します。依存性は、モジュールが有効かどうかを確認するためにチェックします。各レジスタ名をセクションで定義し、レジスタの番号を、順番に付けていなければなりません。依存性は、dep=<reg> <bit> <value>のようにセクションに入力します。

1. <reg>は依存性のレジスタIDです。
2. <bit>はレジスタのビット位置です。
3. <value>は値で、ビットは、有効であるモジュールに使用しなければなりません。

[Register]定義エントリは、id=<name> <address> [<size>[<absolute>[<format>[<bitfields>]]]]のフォーマットで入力します。

1. <name>は表示するレジスタ名です。
2. <address>はレジスタのアドレスです。
3. <size>は、Bがバイトサイズ、Wがワードサイズ、Lがロングワードサイズを意味します（デフォルトはバイトです）。
4. <absolute>は、レジスタが絶対アドレスにある場合、Aと設定します。これは、異なるモードのCPUによってI/O空間アドレス範囲が移動する場合のみ関連します。レジスタが絶対アドレスにあると定義すると、ベースアドレスオフセットは計算せず、指定したアドレスを直接使用します。

5. <format>はレジスタを出力するためのフォーマットです。有効な値は、16進数の場合はH、10進数はD、2進数はBです。
6. <bitfields>セクションは、レジスタのビットを定義します。

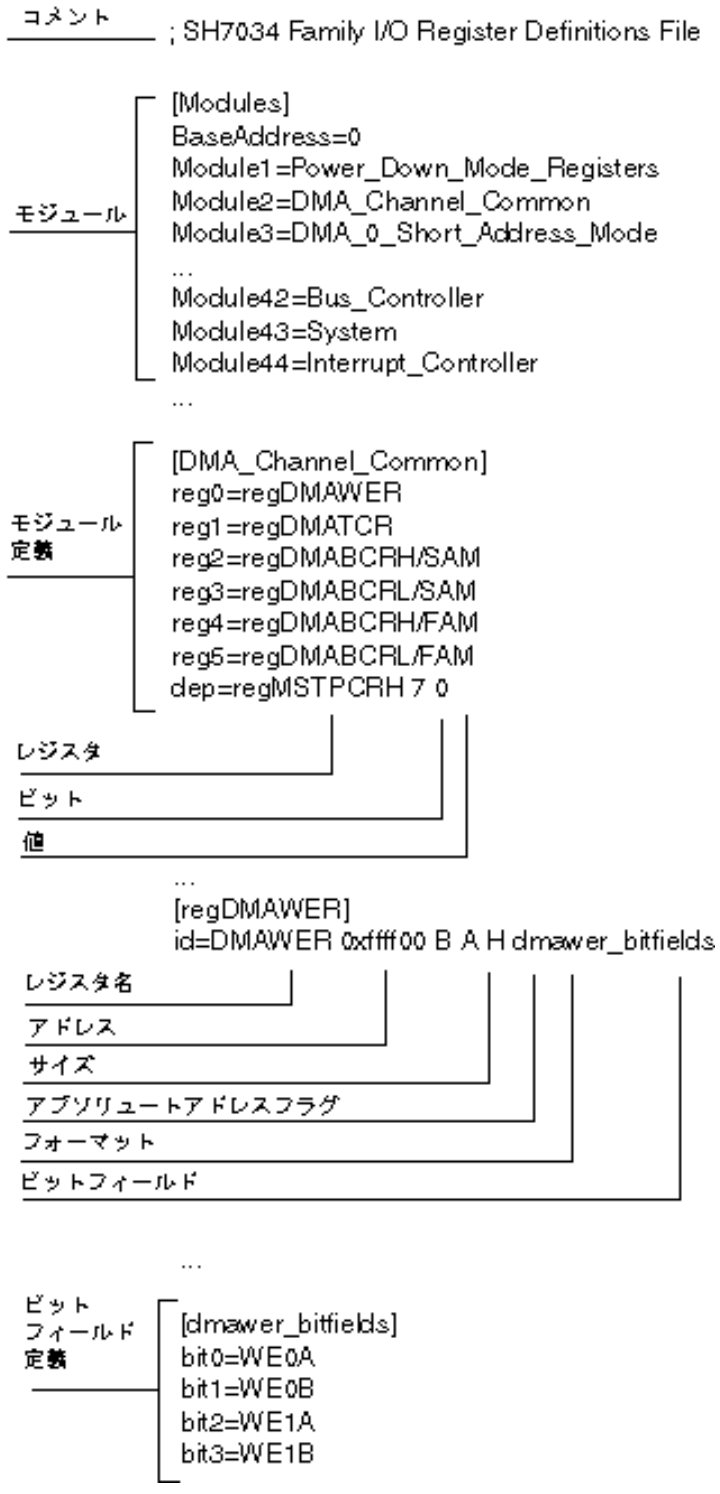
ビットフィールドセクションは、各エントリがbit<no>=<name>タイプのレジスタ内のビットを定義します。

1. <no>はビット番号です。
2. <name>はビットのシンボル名です。

コメント行を入れる場合、“;”で始めなければなりません。

次に例を示します。

例:


コメント
; SH7034 Family I/O Register Definitions File
モジュール
[Modules]
BaseAddress=0
Module1=Power_Down_Mode_Registers
Module2=DMA_Channel_Common
Module3=DMA_0_Short_Address_Mode
…
Module42=Bus_Controller
Module43=System
Module44=Interrupt_Controller
…
モジュール定義
[DMA_Channel_Common]
reg0=regDMAWER
reg1=regDMATCR
reg2=regDMABCRH/SAM
reg3=regDMABCRL/SAM
reg4=regDMABCRH/FAM
reg5=regDMABCRL/FAM
dep=regMSTPCRH 7 0
レジスタ
ビット
値
…
[regDMAWER]
id=DMAWER 0xffff00 B A H dmawer_bitfields
レジスタ名
アドレス
サイズ
アブソリュートアドレスフラグ
フォーマット
ビットフィールド
…
ビットフィールド定義
[dmawer_bitfields]
bit0=WE0A
bit1=WE0B
bit2=WE1A
bit3=WE1B

付録 E　シンボルファイルフォーマット

HEW を理解し、シンボルファイルを正確にデコードするためには、ファイルを Pentica-B ファイルとしてフォーマットしなければなりません。

1. ファイルは、簡単なASCIIテキストファイルでなければなりません。
2. ファイルは、ワード[BEGIN]で始めなければなりません。
3. 各シンボルは、個々の行で、まず、“H”で終了する16進数の値から始まり、次にスペース、シンボルテキストの順でなければなりません。
4. ファイルは、ワード[END]で終了しなければなりません。

例：

```
BEGIN
11FAH Symbol_name_1
11FCH Symbol_name_2
11FEH Symbol_name_3
1200H Symbol_name_4
END
```

付録 F　HMAKE ユーザガイド

F.1　コマンドライン

次の節では、用意されたオプションの使用または不使用にかかわらず、ファイルで hmake プログラムを実行するときに使うコマンドラインを説明します。

F.1.1　基本構成

コマンドラインのシンタックスを以下に示します。

hmake <メイクファイル> <パラメータリスト>

拡張子なしでファイルを指定した場合、.mak が付加されます。パラメータリストには、次の節に示す複数のパラメータを指定できます。また、パラメータは省略できます。パラメータリストはメイクファイル名の前に指定してもかまいません。各パラメータは、1 つ以上の空白文字で区切ってください。パラメータは、大文字と小文字を区別しません。パラメータとファイル名を指定しなかった場合、ヘルプ情報が表示されます。

F.1.2　Exit コード

実行中のメイクファイルでシンタックスエラー発生時、またはメイクファイル実行中の処理で不当エラーコードが返されたとき、hmake はコード 1 で終了します。その他の場合、hmake はコード 0 で終了します（ファイルのシンタックスや exit コード条件の指定方法は、以下を参照してください）。

F.1.3　パラメータ

表 F.1 に指定できるパラメータとその機能を示します。

表 F.1 hmake **のパラメータとその機能**

パラメータ	機能
/A	入出力ファイル状態にかかわらず全コマンドを実行する。全ビルドと同じ
/N	入出力ファイルの使用状態により、（ノーマルで）実行すべきコマンドを判断して、表示するが、実行はしない
/?	ヘルプ情報の表示

F.2　ファイルのシンタックス

hmake ファイルでは、基本的に 4 種類の文（変数宣言、説明部分、コメント、メッセージコメント）が使用されます。hmake ファイルの作成にあたって、これらの文はどのような順序でも使用できます。ただし、変数を説明部分や他の変数宣言に使用する前に、変数宣言をしてください。nmake ファイルで使用する最初の“ all ”文は、hmake ファイルでは必要ありません。メイクファイルでは、コマンドは出現する順序で実行されます。“ → ”文字は、メイクファイルのシンタックスを正しくするためにタブ文字を使用すべきところを示すことに注意してください。

F.2.1　変数宣言

変数宣言をした後、その変数は、hmake ファイルのすべての部分ですべての文で使うことができます。変数宣言のシンタックスを以下に示します。

```
<変数名> = <値>
```

変数名と‘ = ’符号の間や、値と‘ = ’符号の間に、複数の空白文字を入れることができます。値は、‘ ¥ ’文字で区切って、数行にわたって書くことができます。メインテキスト内で値に‘ ¥ ’文字を含む場合は文字どおり解釈されます。‘ ¥ ’文字の後に新しい行が続くときのみ、値が次の行にわたっているとみなされます。変数宣言の例をいくつか示します。

```
EXECUTABLE = c:¥dir¥prog.exe
OUTPUT = c:¥dir2¥file1.out
INPUT = c:¥dir2¥file1.c
DEPEND = c:¥dir2¥file2.h ¥
         c:¥dir2¥file3.h ¥
         c:¥dir2¥file4.h
```

hmake ファイルの後半で変数を使うには、変数名の前に“ $(”を、後ろに“) ”を加えてください。“ $() ”文字付きの変数名は、変数の値で置き換えられます。次の説明部分の節にその例を示します。変数名には、半角英数文字と半角下線を使用することができます。別の変数宣言のなかで変数を使うことができますが、すべての変数は、使う前に宣言しなければなりません。

F.3　記述部

F.3.1　概要

説明部分には、1 つ以上のターゲット、0 以上の依存ファイル、そして最新のターゲットが最新の依存ファイルより新しい場合に実行するコマンド一覧を指定します。ターゲット、依存ファイルのどちらか、またはいずれもない場合、コマンドは常に実行されます。コマンドを常に実行したいときには、依存ファイルを指定する必要はありません。説明部分のシンタックスを以下に示します。

```
<ターゲット1> <ターゲット2> … : <依存ファイル1> <依存ファイル2> …
→<コマンド1>
→<コマンド2>
→…
→<コマンドn>
```

最後のターゲットと‘ : ’文字の間や、最初の依存ファイルと‘ : ’文字の間に、複数の空白文字を入れることができます。最初のターゲットの前に空白文字を入れないでください。各ターゲットと各依存ファイルは、少なくとも 1 つの空白文字で区切ってください。タブ文字はコマンドを含む行のはじめに必要です。変数宣言の後で、変数を上記シンタックスで説明部分に使うことができます。

記述部の例（変数宣言の後に変数を使う例）を以下に示します。

```
c:¥dir1¥file1.obj : c:¥dir1¥file1.c c:¥dir1¥file1.h
→gcc c:¥dir1¥file1.c
$(OUTPUT) : $(INPUT) $(DEPEND)
→$(EXECUTABLE) $(INPUT)
```

F.3.2　特殊なコマンド

記述ブロックで使用できる特殊なコマンドが 2 つあります。cd コマンドはカレントディレクトリを変更し、set コマンドはメイクファイル実行の間に使用する環境変数を設定します。両方とも、相当する DOS コマンドと同じ方法で使用されます。

これらのコマンドを使用した有効な記述ブロックの例を、以下に示します。

```
CHANGEDIR :
→ cd c:¥dir1¥dir2
SETENV:
→ set VAR1=value1
→ set VAR2=value2
→ set VAR3=value3
```

$(EXECUTABLE)を置き換える CHANGEDIR と SETENV がファイル名でなくてもかまいません。これらは存在しないファイルとして扱われてコマンドは常に実行されます。

F.3.3　サブコマンドファイル

hmake でサブコマンドファイルを生成するには、説明部分のコマンド部分を以下のように指定してください(上記の<コマンド n>を置き換えます)。

```
→<コマンド開始> <<
→<サブコマンド1>,
→<サブコマンド2>,
→…
→<サブコマンドn>
→<<<コマンド終了>
```

これで、ウィンドウズの一時的ファイルに、<サブコマンド 1>、<サブコマンド 2>などの行を含むサブコマンドファイルが生成されます。このコマンドファイルはメイク処理が終了すると削除されます。コマンドファイル名は 2 つの”<<”の間のすべてのテキストで置き換えられます。サブコマンドファイル名は、自動的に hmake により付けられます。

例：

```
c:¥dir1¥file1.obj : c:¥dir1¥file1.c c:¥dir1¥file1.h
→gcc @”<<
→-c -o c:¥dir1¥file1.obj c:¥dir1¥file1.c
<<”
```

生成されたサブコマンドファイル名が“ c:¥temp¥hmk111.cmd ”の場合、hmake は以下を実行します。(c:¥dir1¥file1.obj が更新されるとき)：

```
gcc @”c:¥temp¥hmk111.cmd”
```

サブコマンドファイル(c:¥temp¥hmk111.cmd)には以下を含みます。

```
-c -o c:¥dir1¥file1.obj c:¥dir1¥file1.c
```

記述部に 1 つ以上のコマンドを含むことができ、標準とサブコマンドファイルコマンドを混在して使用できます。

F.4　コメント

‘ # ’文字はコメントを示します。この文字が行の最初にあると、（次の改行文字があるまで）その行は無視されます。コメントの例を次に示します。

```
# My hmake file

# Variable declaration
OUTPUT= c:¥dir1¥file1.obj
# Descriptor
$(OUTPUT) : c:¥dir1¥file1.c c:¥dir1¥file1.h
→set VAR1=value1
→gcc c:¥dir1¥file1.c
```

hmake ファイルではコメントには専用の行が必要です。コメントを他の文の最後に付けることはできません。

F.5　メッセージコマンド

メッセージコマンドは、メイクファイル実行中にテキストの行を標準出力に出力します。これらのテキスト行は、適切に実行されている実行ファイルの出力のなかで、メイクファイルに出現する順序で出力されます。出力テキストのバッファリングは行いません。メッセージコマンドのシンタックスを以下に示します。

```
!MESSAGE <出力するテキスト>
```

改行文字が<出力するテキスト>の最後の文字の後にあると想定します。!MESSGE と<出力するテキスト>の間の空白文字は無視されます。メッセージコマンドの例を以下に示します。

```
!MESSAGE Executing C Compiler
```

H8S,H8/300シリーズ
High-performance Embedded Workshop 3　ユーザーズマニュアル
(Windows® 98/Me、Windows NT® 4.0、Windows® 2000およびWindows® XP用)

発行年月　2003 年 7 月 22 日　Rev.1.00
　　2004 年 6 月 23 日　Rev.2.00
発　　行　株式会社ルネサス テクノロジ 営業企画統括部
　　〒100-0004 東京都千代田区大手町 2-6-2
編　　集　株式会社ルネサス小平セミコン 技術ドキュメント部

株式会社ルネサス テクノロジ　営業企画統括部　〒100-0004 東京都千代田区大手町2-6-2 日本ビル

営業お問合せ窓口
株式会社ルネサス販売

http://www.renesas.com

本社	〒100-0004	千代田区大手町2-6-2 (日本ビル)	(03) 5201-5350
京浜支社	〒212-0058	川崎市幸区鹿島田890-12 (新川崎三井ビル)	(044) 549-1662
西東京支社	〒190-0023	立川市柴崎町2-2-23 (第二高島ビル2F)	(042) 524-8701
札幌支店	〒060-0002	札幌市中央区北二条西4-1 (札幌三井ビル5F)	(011) 210-8717
東北支社	〒980-0013	仙台市青葉区花京院1-1-20 (花京院スクエア13F)	(022) 221-1351
いわき支店	〒970-8026	いわき市平小太郎町4-9 (損保ジャパンいわき第二ビル3F)	(0246) 22-3222
茨城支店	〒312-0034	ひたちなか市堀口832-2 (日立システムプラザ勝田1F)	(029) 271-9411
新潟支店	〒950-0087	新潟市東大通1-4-2 (新潟三井物産ビル3F)	(025) 241-4361
松本支社	〒390-0815	松本市深志1-2-11 (昭和ビル7F)	(0263) 33-6622
中部営業本部	〒460-0008	名古屋市中区栄3-13-20 (栄センタービル4F)	(052) 261-3000
浜松支店	〒430-7710	浜松市板屋町111-2（浜松アクトタワー10F）	(053) 451-2131
西部営業本部	〒541-0044	大阪市中央区伏見町4-1-1 (明治安田生命大阪御堂筋ビル)	(06) 6233-9500
北陸支社	〒920-0031	金沢市広岡3-1-1 (金沢パークビル8F)	(076) 233-5980
広島支店	〒730-0036	広島市中区袋町5-25 (広島袋町ビルディング8F)	(082) 244-2570
鳥取支店	〒680-0822	鳥取市今町2-251 (日本生命鳥取駅前ビル)	(0857) 21-1915
九州支社	〒812-0011	福岡市博多区博多駅前2-17-1 (ヒロカネビル本館5F)	(092) 481-7695
鹿児島支店	〒890-0053	鹿児島市中央町12-2 (明治安田生命鹿児島中央町ビル)	(099) 284-1748

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：カスタマサポートセンタ　E-Mail: csc@renesas.com

Colophon 3.0

H8S,H8/300 シリーズ
High-performance Embedded Workshop 3
ユーザーズマニュアル

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

RJJ10B0029-0200H