お客様各位

カタログ等資料中の旧社名の扱いについて

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

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

2010年4月1日
ルネサスエレクトロニクス株式会社

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

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

ご注意書き

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

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

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

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

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

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

お客様各位

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

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

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

2003年4月1日
株式会社ルネサス テクノロジ
カスタマサポート部

RENESAS

ご注意

安全設計に関するお願い

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

本資料ご利用に際しての留意事項

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

User's Manual

H8S,H8/300シリーズ シミュレータ・デバッガ

ユーザーズマニュアル

HSS008SDIW4SJ

ルネサス エレクトロニクス
www.renesas.com

Rev.2.00　2002.05

ご注意

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

商標

Microsoft®および Windows®、 は、米国マイクロソフトコーポレーションの米国およびその他の国における登録商標です。

IBM PC は、米国 IBM 社により管理されている計算機の名称です。

ELF/DWARF2 は、the Tool Interface Standards Committee で開発された、オブジェクトフォーマットの名称です。

本マニュアルで使用されているすべての製品名またはブランド名は、それぞれの会社の商標または登録商標です。

本マニュアルを使用する際には、以下の点に留意してください。

1. 本マニュアルは、全体または一部が予告なしに変更される場合があります。
2. 本マニュアルの一部または全部を（株）日立製作所の許可を得ることなく、複製または複写することはできません。
3. 日立は、本マニュアルに従って操作を行っている間の事故または他の原因によるユーザの装置への損害については、責任を負いません。

まえがき

シミュレータ・デバッガの使用前に本マニュアルを必ずお読みください。
ユーザーズマニュアルは、なくさないよう保管してください。
本システムの使用方法を十分理解してから、使用してください。

本書について

本書では、日立のマイクロコンピュータの開発ツールに対応したHitachi Embedded Workshop (HEW) およびシミュレータ・デバッガの使用法について説明します。次章、「はじめに」では、デバッグインタフェースおよびシミュレータ・デバッガについて簡単に紹介し、おもな特長を示します。
次の各章、「システムの概要」、「シミュレータ・デバッガの機能」、「メニュー」、「ウィンドウおよびダイアログボックス」、「コマンドライン」および「メッセージ一覧」は、それぞれの操作方法や利便性についてのリファレンスです。

前提事項

本書では、読者の方々に、C/C++プログラミング言語およびデバッグ対象プロセッサ用のアセンブラニーモニックについての十分な知識があること、およびMicrosoft® Windows®アプリケーションの使用経験があることを前提としています。

マニュアルの規約

本書には、以下の表記上の規約があります。

表 1　表記上の規約

表　記	意　　味
[Menu->Menu Option]	'->'**の付いた太字の表記は、メニューオプションを指定する場合に使用します** (**例：** [File->Save As...])**。**
FILENAME.C	**大文字表記の名前は、ファイル名を指定する場合に使用します。**
"enter this string"	**入力必須のテキストを指定する場合に使用します(引用符" "は除く)。**
キー + **キー**	**必要なキーを押すよう指示する場合に使用します。たとえば、**CTRL+N **は、**CTRL **キーを押したまま** N **キーを押すことを意味します。**
➲ ("**手順**"**記号**)	**この記号は、必ず左端に記載されます。この記号が使われる場合は、すぐ右に実行手順が記載されていることを意味します。**

目次

1. **はじめに**

1.1 特長 1

1.2 デバッグ対象プログラム 2

1.3 シミュレーション範囲 3

2. **システムの概要**

2.1 ユーザインタフェース 5

2.2 データ入力 5

2.2.1 演算子 5

2.2.2 データ形式 5

2.2.3 精度 6

2.2.4 式の例 6

2.2.5 シンボル形式 6

2.2.6 シンボル指定の例 6

3. **シミュレータ・デバッガの機能**

3.1 シミュレータ・デバッガのメモリ管理 7

3.2 命令実行リセット処理 8

3.3 例外処理 8

3.4 H8S/2600CPU 特殊機能 8

3.5 制御レジスタ 9

3.6 トレース 9

3.7 標準入出力およびファイル入出力処理 10

3.8 命令実行サイクル数の計算 19

3.9 ブレーク条件 20

3.10 浮動小数点データ 22

3.11 関数呼び出し履歴の表示 23

3.12 プロファイラ 23

3.13 擬似割込み 23

3.14 カバレジ 24

4. **メニュー**

4.1 View 25

4.1.1 Workspace 25

4.1.2 Output 25

4.1.3 Breakpoints 26

4.1.4 Command Line 26

4.1.5 Disassembly 26
4.1.6 IO 26
4.1.7 Labels 26
4.1.8 Locals 26
4.1.9 Memory 26
4.1.10 Performance Analysis 26
4.1.11 Profile 26
4.1.12 Registers 26
4.1.13 Status 27
4.1.14 Trace 27
4.1.15 Watch 27
4.1.16 Localized Dump 27
4.1.17 Simulated I/O 27
4.1.18 Stack Trace 27
4.1.19 Coverage 27
4.1.20 Image 27
4.1.21 Waveform 27
4.1.22 Trigger 27
4.2 Options 27
4.2.1 Debug Settings 28
4.2.2 Radix 28
4.2.3 Simulator 28
4.3 Debug 28
4.3.1 Reset CPU 28
4.3.2 Go 28
4.3.3 Reset Go 28
4.3.4 Go To Cursor 28
4.3.5 Set PC To Cursor 29
4.3.6 Run 29
4.3.7 Step In 29
4.3.8 Step Over 29
4.3.9 Step Out 29
4.3.10 Step 29
4.3.11 Step Mode 29
4.3.12 Halt Program 29
4.3.13 Initialize 29
4.3.14 Disconnect 29
4.3.15 Download Modules 30
4.3.16 Unload Modules 30
4.4 Memory 30
4.4.1 Search 30
4.4.2 Copy 30
4.4.3 Compare 30
4.4.4 Fill 30
4.4.5 Refresh 30
4.4.6 Configure Overlay 30

5. ウィンドウおよびダイアログボックス

5.1 Break 31
5.1.1 Add... 32
5.1.2 Edit 32
5.1.3 Enable 32
5.1.4 Disable 32
5.1.5 Delete 32
5.1.6 Delete All 32
5.1.7 Go to Source 32
5.1.8 Close File 32
5.1.9 Close All Files 33
5.2 Set Break ダイアログボックス（Condition シート） 33
5.3 Set Break ダイアログボックス（Action シート） 35
5.4 Command Line 37
5.4.1 Set Batch File 37
5.4.2 Play 38
5.4.3 Stop 38
5.4.4 Set Log File 38
5.4.5 Logging 38
5.4.6 Browse… 38
5.4.7 Placeholder 38
5.4.8 Select All 38
5.4.9 Copy 39
5.4.10 Paste 39
5.5 Disassembly 40
5.5.1 View Source 40
5.5.2 Go to cursor 41
5.5.3 Set Address... 41
5.5.4 Set PC Here 41
5.5.5 Edit 41
5.5.6 Code Bytes 41
5.5.7 Toggle Breakpoint 41
5.6 IO 42
5.7 Label 43
5.7.1 Add... 43
5.7.2 Edit 43
5.7.3 Delete 44
5.7.4 Delete All 45
5.7.5 Load... 45
5.7.6 Save 46
5.7.7 Save As... 46
5.7.8 Find... 46
5.7.9 Find Next 46
5.7.10 View Source 46

5.8 Locals....47
5.8.1 Edit Value... 47
5.8.2 Radix....47
5.8.3 Copy 47
5.9 Memory....48
5.9.1 Lock Refresh....48
5.9.2 Refresh....48
5.9.3 Start Address....48
5.9.4 Format....49
5.9.5 Search... 49
5.9.6 Search Next....49
5.9.7 Copy....49
5.9.8 Compare....49
5.9.9 Fill....49
5.9.10 Save... 49
5.9.11 Load... 49
5.10 Performance Analysis 50
5.10.1 Add Range... 50
5.10.2 Edit Range 50
5.10.3 Reset Counts/Times 51
5.10.4 Enable Analysis 51
5.10.5 Delete Range....51
5.10.6 Delete All Ranges 51
5.11 Performance Option ダイアログボックス....51
5.12 Registers....52
5.12.1 Edit....52
5.13 Source 53
5.13.1 Toggle Breakpoint 53
5.13.2 Enable/Disable Breakpoint 53
5.13.3 Instant Watch… 53
5.13.4 Go To Cursor 54
5.13.5 Set PC Here....54
5.13.6 Go To Disassembly....54
5.14 Source address カラム....54
5.15 Debugger カラム....56
5.16 Status....56
5.17 Trace 57
5.17.1 Find... 58
5.17.2 Find Next 58
5.17.3 Acquisition....58
5.17.4 Clear....58
5.17.5 Save... 58
5.17.6 View Source....58
5.17.7 Trim Source 58
5.17.8 Statistic 58

5.18 Trace Acquisition ダイアログボックス 59
5.19 Trace Search ダイアログボックス 60
5.20 Trace Statistic ダイアログボックス 61
5.21 Trigger 62
5.21.1 Setting... 62
5.21.2 Size 62
5.22 Trigger Setting ダイアログボックス 63
5.23 Watch 64
5.23.1 Auto Update 64
5.23.2 Auto Update All 64
5.23.3 Delete Auto Update 65
5.23.4 Delete Auto Update All 65
5.23.5 Add Watch... 65
5.23.6 Edit Value... 65
5.23.7 Delete 65
5.23.8 Delete All 65
5.23.9 Radix 65
5.23.10 Copy 65
5.23.11 Save As... 65
5.23.12 Go To Memory... 65
5.24 Simulator System ダイアログボックス 66
5.25 Memory Map Modify ダイアログボックス 67
5.26 Simulator Memory Resource ダイアログボックス 68
5.27 System Memory Resource Modify ダイアログボックス 69
5.28 Simulated I/O 69
5.29 Stack Trace 70
5.29.1 Go to Source 70
5.29.2 View Setting... 71
5.29.3 Copy 71
5.30 Profile（List シート） 72
5.31 Profile（Tree シート） 73
5.31.1 View Source 73
5.31.2 View Profile-Chart 74
5.31.3 Enable Profiler 74
5.31.4 Not trace the function call 74
5.31.5 Find... 74
5.31.6 Find Data... 74
5.31.7 Clear Data 74
5.31.8 Output Profile Information Files... 75
5.31.9 Output Text File... 75
5.31.10 Setting 75
5.31.11 Properties... 75

5.32 Profile-Chart 76
5.32.1 View Source 76
5.32.2 View Profile-Chart 76
5.32.3 Enable Profiler 76
5.32.4 Clear Data 76
5.32.5 Multiple View 77
5.32.6 Output Profile Information File... 77
5.32.7 Expands Size 77
5.32.8 Reduces Size 77
5.33 Image View 78
5.33.1 Auto Refresh 78
5.33.2 Refresh Now 78
5.33.3 Property... 78
5.34 Image Properties ダイアログボックス 79
5.35 Pixel Information ダイアログボックス 81
5.36 Waveform 82
5.36.1 Auto Refresh 82
5.36.2 Refresh Now 82
5.36.3 Zoom In 82
5.36.4 Zoom Out 82
5.36.5 Reset Zoom 82
5.36.6 Zoom Magnification 83
5.36.7 Scale 83
5.36.8 Clear Cursor 83
5.36.9 Sample Information… 83
5.36.10 Property… 83
5.37 Waveform Properties ダイアログボックス 83
5.38 Sample Information ダイアログボックス 84
5.39 Coverage 85
5.39.1 View Source 85
5.39.2 Go to Address... 85
5.39.3 Set Range... 86
5.39.4 Enable Coverage 86
5.39.5 Clear Data... 86
5.39.6 Save Data... 86
5.39.7 Load Data 86
5.39.8 Refresh 86
5.39.9 Lock Refresh 86

5.40 Open Coverage ダイアログボックス 86
5.41 Go To Address ダイアログボックス 87
5.42 Coverage Range ダイアログボックス 87
5.43 Save Data ダイアログボックス 88
5.44 Load Data ダイアログボックス 88
5.45 Confirmation Request ダイアログボックス 89
5.46 Save Coverage Data ダイアログボックス 89

6. コマンドライン

6.1 ! (コメント) 93
6.2 ANALYSIS 93
6.3 ANALYSIS_RANGE 94
6.4 ANALYSIS_RANGE_DELETE 94
6.5 ASSEMBLE 95
6.6 ASSERT 95
6.7 BREAKPOINT 96
6.8 BREAK_ACCESS 97
6.9 BREAK_CLEAR 99
6.10 BREAK_CYCLE 99
6.11 BREAK_DATA 101
6.12 BREAK_DISPLAY 102
6.13 BREAK_ENABLE 103
6.14 BREAK_REGISTER 103
6.15 BREAK_SEQUENCE 105
6.16 CHANGE_CONFIGURATION 106
6.17 CHANGE_PROJECT 106
6.18 COVERAGE 107
6.19 COVERAGE_DISPLAY 107
6.20 COVERAGE_LOAD 108
6.21 COVERAGE_RANGE 108
6.22 COVERAGE_SAVE 108
6.23 DEFAULT_OBJECT_FORMAT 109
6.24 DISASSEMBLE 110
6.25 ERASE 110
6.26 EVALUATE 111
6.27 FILE_LOAD 112
6.28 FILE_SAVE 112
6.29 FILE_VERIFY 113
6.30 GO 113

6.31 GO_RESET 114
6.32 GO_TILL 115
6.33 HALT 115
6.34 INITIALIZE 116
6.35 LOG 116
6.36 MAP_DISPLAY 117
6.37 MAP_SET 117
6.38 MEMORY_DISPLAY 118
6.39 MEMORY_EDIT 119
6.40 MEMORY_FILL 120
6.41 MEMORY_MOVE 120
6.42 MEMORY_TEST 121
6.43 OPEN_WORKSPACE 121
6.44 PROFILE 122
6.45 PROFILE_DISPLAY 123
6.46 PROFILE_SAVE 124
6.47 QUIT 124
6.48 RADIX 125
6.49 REGISTER_DISPLAY 125
6.50 REGISTER_SET 126
6.51 RESET 126
6.52 RESPONSE 127
6.53 SLEEP 127
6.54 STEP 128
6.55 STEP_MODE 128
6.56 STEP_OUT 129
6.57 STEP_OVER 130
6.58 STEP_RATE 130
6.59 SUBMIT 131
6.60 SYMBOL_ADD 131
6.61 SYMBOL_CLEAR 132
6.62 SYMBOL_LOAD 132
6.63 SYMBOL_SAVE 133
6.64 SYMBOL_VIEW 133
6.65 TCL 134
6.66 TRACE 134
6.67 TRACE_ACQUISITION 135
6.68 TRACE_SAVE 135

6.69 TRACE_STATISTIC....136

7. メッセージ一覧

7.1 インフォメーションメッセージ....137

7.2 エラーメッセージ....137

付録 A GUI コマンド一覧

図版目次

図 1-1　デバッグ対象プログラム作成方法 2
図 3-1　入出力機能の説明形式 11
図 4-1　メニュー 25
図 5-1　Breakウィンドウ 31
図 5-2　Set Breakダイアログボックス（Conditionシート） 33
図 5-3　Set Breakダイアログボックス（Actionシート） 35
図 5-4　Command Lineウィンドウ 37
図 5-5　Set Batch Fileダイアログボックス 37
図 5-6　Open Log Fileダイアログボックス 38
図 5-7　Disassemblyウィンドウ 40
図 5-8　IO Area ウィンドウ 42
図 5-9　Label ウィンドウ 43
図 5-10　Add Labelダイアログボックス 43
図 5-11　Edit Labelダイアログボックス 44
図 5-12　ラベル削除確認メッセージボックス 44
図 5-13　Confirming All Label Deletionメッセージボックス 45
図 5-14　Load Symbols ダイアログボックス 45
図 5-15　Find Labelダイアログボックス 46
図 5-16　Localsウィンドウ 47
図 5-17　Memoryウィンドウ 48
図 5-18　Performance Analysisウィンドウ 50
図 5-19　Performance Optionダイアログボックス 51
図 5-20　Registersウィンドウ 52
図 5-21　Sourceウィンドウ 53
図 5-22　Instant Watchダイアログボックス 54
図 5-23　Sourceウィンドウとaddressカラム 55
図 5-24　 Global Editor Column Statesダイアログボックス 56
図 5-25　Statusウィンドウ 56
図 5-26　Traceウィンドウ 57
図 5-27　Trace Acquisitionダイアログボックス 59
図 5-28　Trace Searchダイアログボックス 60
図 5-29　Trace Statisticダイアログボックス 61
図 5-30　Triggerウィンドウ 62
図 5-31　Trigger Settingダイアログボックス 63
図 5-32　Watchウィンドウ 64
図 5-33　Simulator Systemダイアログボックス 66
図 5-34　Memory Map Modifyダイアログボックス 67
図 5-35　Simulator Memory Resourceダイアログボックス 68
図 5-36　System Memory Resource Modifyダイアログボックス 69
図 5-37　Simulated I/Oウィンドウ 69
図 5-38　Stack Traceウィンドウ 70

図 5-39 Stack Trace Settingダイアログボックス 71
図 5-40 Profileウィンドウ（Listシート） 72
図 5-41 Profileウィンドウ（Treeシート） 73
図 5-42 Find Dataダイアログボックス 74
図 5-43 Profile-Chartウィンドウ 76
図 5-44 Image Viewウィンドウ 78
図 5-45 Image Propertiesダイアログボックス 79
図 5-46 Pixel Informationダイアログボックス 81
図 5-47 Waveformウィンドウ 82
図 5-48 Waveform Propertiesダイアログボックス 83
図 5-49 Sample Informationダイアログボックス 84
図 5-50 Coverageウィンドウ 85
図 5-51 Open Coverageダイアログボックス 86
図 5-52 Go To Addressダイアログボックス 87
図 5-53 Coverage Rangeダイアログボックス 87
図 5-54 Save Dataダイアログボックス 88
図 5-55 Load Dataダイアログボックス 88
図 5-56 Confirmation Requestダイアログボックス 89
図 5-57 Save Coverage Dataダイアログボックス 89

1. はじめに

シミュレータ・デバッガは、H8S、H8/300 シリーズマイコンの CPU シミュレーション機能およびデバッグ機能を持っています。シミュレータ・デバッガを利用することによって、C/C++言語やアセンブリ言語で作成したプログラムを効率よくデバッグすることができます。

Hitachi Embedded Workshop (HEW)は、日立のマイクロコンピュータ用に、C/C++言語およびアセンブリ言語で書いたアプリケーションの開発およびデバッグを簡単に行うためのグラフィカルユーザインタフェースを提供します。アプリケーションを実行するシミュレータ・デバッガのアクセス、計測、および変更に関して、HEW は高機能でしかも直観的な手段を提供することを目的としています。
【注】HEW は Windows®3.1 では動作しません。

1.1 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） 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
- デバッグ対象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（擬似的な割込み動作も可能）
- 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

（4） 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

1.2 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、および S-type フォーマットのロードモジュールがデバッグ可能です。これらのロードモジュールをデバッグ対象プログラムと呼びます。

デバッグ対象プログラム作成方法を**図** 1-1に示します。

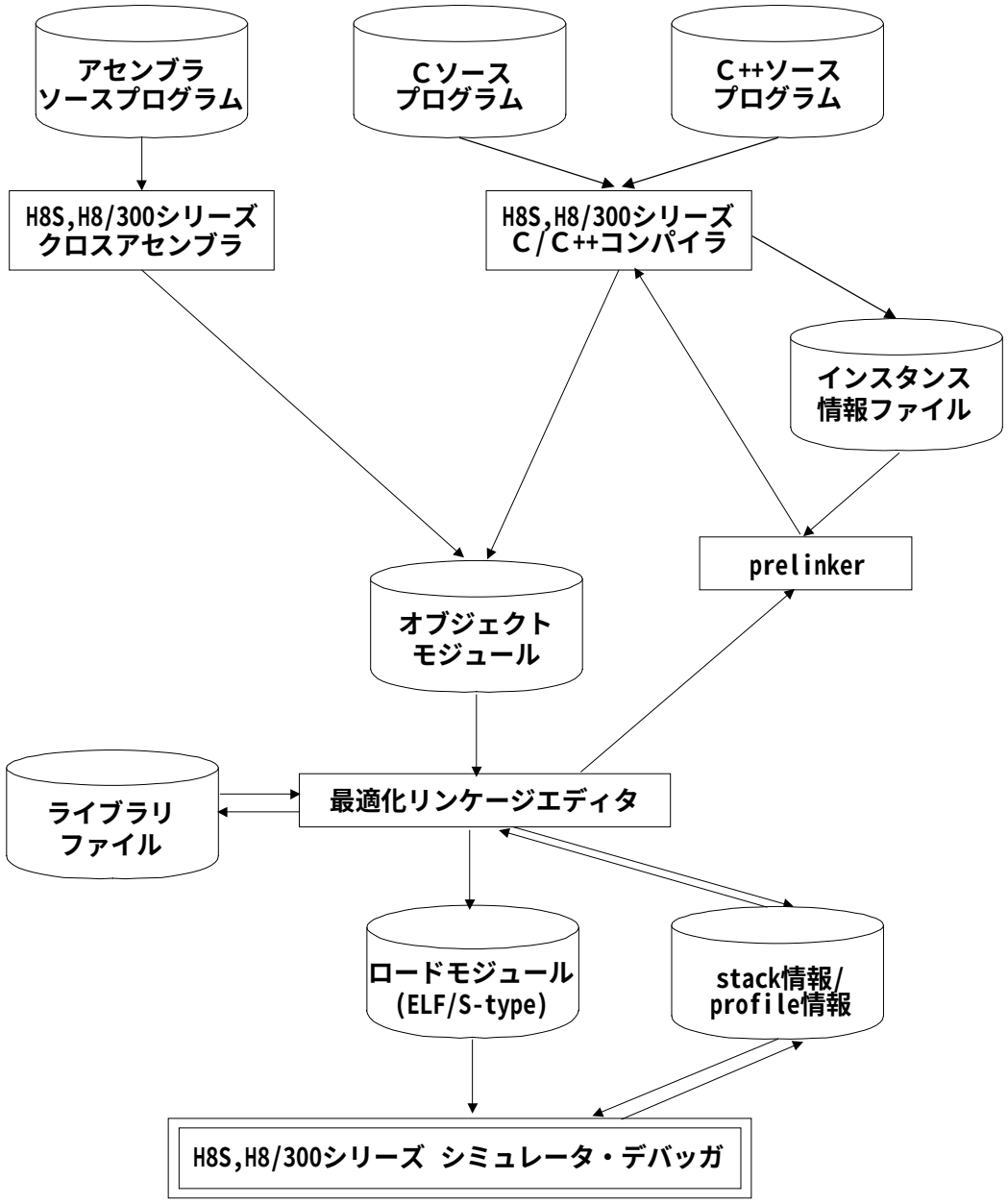


図 1-1 デバッグ対象プログラム作成方法

1.3 シミュレーション範囲

（1） H8/300、H8/300L、H8/300H、H8S/2600、および H8S/2000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 1-1 に示す各 CPU モード

表1-1 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**

（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. システムの概要

HEW はモジュール方式のソフトウェアシステムで、それぞれに独立したモジュールを使用します。これらのモジュールは汎用グラフィカルユーザインタフェースへリンクしており、これにより、システムを構成するモジュールに関係なく共通した操作性を提供します。

2.1 ユーザインタフェース

HEW グラフィカルユーザインタフェースは、ユーザにデバッグプラットフォームを提供し、システムの設定および変更を可能にする Windows®対応のアプリケーションです。Windows®アプリケーションの詳しい操作方法は、Windows®標準のユーザーズマニュアルを参照してください。

2.2 データ入力

ダイアログボックスまたはフィールドに数値を入力する場合には、単純な数値だけでなく式も入力できます。この式には、符号を含めたり、C/C++言語の演算子を使用することができます。C/C++言語のデバッグをサポートする ELF/DWARF2 フォーマットを使用している場合は、配列や構造体などの C/C++言語機能を使用できます。

一部のダイアログボックスでは、終了アドレスを入力する際に値の前に+符号を付けることによりアドレス範囲を入力することが可能です。
この場合、+符号を付けて入力した値と先頭アドレスの和が、終了アドレスになります。

2.2.1 演算子

以下の C/C++言語演算子を使用できます。

+, -, *, /, &, |, ^, ~, !, >>, <<, %, (,), <, >, <=, >=, ==, !=, &&, ||

2.2.2 データ形式

接頭辞のないデータ値は、[Options->Radix]メニューオプションで設定したデフォルトの基数を使用していると見なします。ただし、回数を入力するところは例外で 10 進数の値をデフォルトにとります。

名前にはシンボルを使用できます。また、一重引用符で囲めば、ASCII 文字列を入力できます (例：'demo')。

次の接頭辞を使用して、基数を指定できます。
B' 2 進
O' 8 進
D' 10 進
H' 16 進
0x 16 進

先頭コードに#文字を使用してレジスタ名を指定すると、そのレジスタの内容を使用できます。次に例を示します。

#R1, #E1

2.2.3 精度

式の評価では、すべての数理演算は 64 ビット(符号付き)を使用して行います。64 ビットを超える値はすべて切り捨てます。

2.2.4 式の例

```
Buffer_start + 0x1000
#R1 | B'10001101
((pointer + (2 * increment_size)) & H'FFFF0000) >> D'15
!(flag ^ #R4)
```

2.2.5 シンボル形式

シンボルには、C/C++ベースの指定ができます。これにより、C/C++言語ソースと同様な記述でシンボルの参照ができます。また、シンボルにはキャスト演算子をつけることができます。これにより、型変換後のデータを参照することができます。ただし、以下のような制限事項があります。

- ポインタ指定は 4 レベルまで可能です
- 配列指定は 3 次元まで可能です
- typedef 名は使用できません

2.2.6 シンボル指定の例

Object.value	：メンバの直接参照指定(C/C++)
p_Object->value	：メンバの間接参照指定(C/C++)
Class::value	：クラス指定付きメンバ参照指定(C++)
*value	：ポインタ指定(C/C++)
array[0]	：配列指定(C/C++)
Object.*value	：メンバへのポインタ参照指定(C++)
::g_value	：グローバル変数参照指定(C/C++)
Class::function(short)	：メンバ関数指定(C++)
(struct STR)*value	：キャスト指定(C/C++)

3. 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。

3.1 シミュレータ・デバッガのメモリ管理

(1) メモリマップの設定

メモリマップの設定は、シミュレーション時のメモリアクセスサイクル数計算に使用します。シミュレータ・デバッガでは、表 3-1に示すメモリ種別をサポートしています。

表3-1 **メモリ種別**

メモリ種別	デバッグ対象プログラムの実行
内蔵 ROM	**可能**
内蔵 RAM	**可能**
外部メモリ	**可能**
内蔵 I/O	**不可能**
EEPROM	**可能**

メモリマップは、Simulator System ダイアログボックス上で設定することができ、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

設定できるメモリ種別は、CPU によって異なります。詳細は、**「**5.24 Simulation System **ダイアログボックス」**を参照してください。なお、デバッグ対象プログラムは、内蔵 I/O 空間を除くすべてのメモリで実行可能です。

(2) メモリリソースの確保

デバッグ対象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。メモリリソースは、Simulator Memory Resource ダイアログボックスで設定できます。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。

デバッグ対象プログラムで、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。

3.2 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

3.3 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。

3.4 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、Simulator System 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、「5.24 Simulator System ダイアログボックス」を参照してください。

3.5 制御レジスタ

H8S/2600 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デバッグ対象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、Simulator System ダイアログボックスの [SYSCR Address]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、「5.24 Simulator Systemダイアログボックス」、「5.6 IO」を参照してください。

3.6 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、Trace Acquisition ダイアログボックスで指定します。Trace Acquisition ダイアログボックスは、Trace ウィンドウ上で右クリックしてポップアップメニューを表示し、[Acquisition...]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、Trace ウィンドウに表示します。Trace ウィンドウに表示する内容は、以下の通りです。

- 累計命令実行サイクル数
- 命令アドレス
- CCR
- 乗算器内部フラグ（H8S/2600 シリーズでのみ有効）
- 命令ニーモニック
- データアクセス情報（転送先および転送データ）
- C/C++またはアセンブラソース

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、Trace Search ダイアログボックスで設定します。Trace Search ダイアログボックスは、Trace ウィンドウ上で右クリックしてポップアップメニューを表示し、[Find...]を選択することによって表示できます。

詳しくは、「5.17 Trace」から「5.19 Trace Search ダイアログボックス」を参照してください。

3.7 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、デバッグ対象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 Simulated I/O 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。

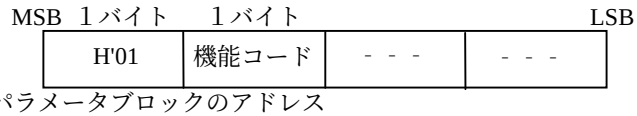
表3-2 入出力機能一覧

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

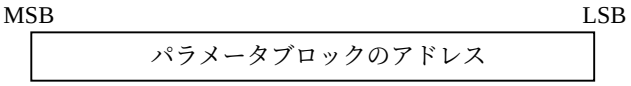
この機能を実現するためには、まず入出力用の特定の位置を Simulator System ダイアログボックスの [System Call Address]で指定し、[Enable]をチェック後、デバッグ対象プログラムを実行します。シミュレータ・デバッガでは、デバッグ対象プログラムの命令を実行中に、指定した位置へのサブルーチン分岐命令（BSR、JSR）すなわちシステムコール命令を検出すると、R0、R1（H8/300、H8/300L シリーズ）または ER1（H8/300H、H8S シリーズ）の内容をパラメータとして入出力処理を行います。

したがって、システムコールを行う前にデバッグ対象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 3-2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** 3-1の形式で説明します。

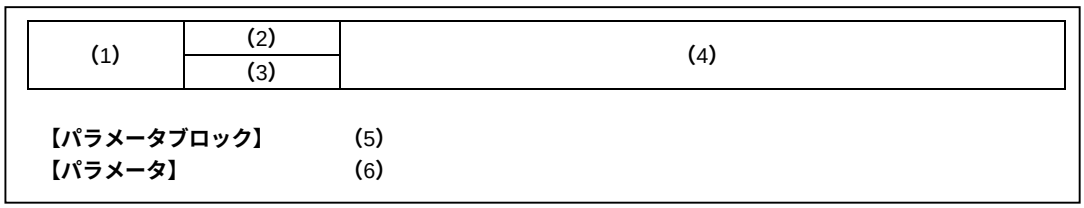


図 3-1 **入出力機能の説明形式**

各項目の内容は、以下の通りです。
（1） 表3-2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
```

3.8 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 Simulator System ダイアログボックスで設定したメモリ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。

3.9 ブレーク条件

デバッグ対象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
- デバッグ対象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
- [STOP]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_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、**「**5.1 Break**」**を参照してください。

デバッグ対象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** 3-3に示します。

表3-3 **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_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。

デバッグ対象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) デバッグ対象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 Simulator System 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** 3-4に示します。

表3-4 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 3-5に示します。なお、シミュレーションエラー停止後 SR の内容は変化しません。

表3-5 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

Trace Acquisition ダイアログボックスの [Trace Buffer Full Handling]で [Break]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 Output 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 Output 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) [STOP]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。

3.10 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。

・Set Break ダイアログボックスにおいて、ブレーク種別を [Break Data] や [Break Register]と指定したときのデータ
・Memory ウィンドウにおけるデータ
・Fill Memory ダイアログボックスにおけるデータ
・Search Memory ダイアログボックスにおけるデータ

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、Simulator System 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（Round to Nearest）
- RZ（Round to Zero）

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。

3.11 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 Stack Trace 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、Stack Trace 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 Source 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。

- 「3.9 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、「5.29 Stack Trace」を参照してください。

3.12 プロファイラ

シミュレータ・デバッガは、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。表示するプロファイルデータは、以下の通りです。

- Times（関数呼出し回数またはグローバル変数アクセス回数）
- 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）

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、「5.30 Profile (List シート)」～「5.32 Profile-Chart」および「最適化リンケージエディタマニュアル　４．２．３Optimize オプション　PROfile」を参照してください。

3.13 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、「5.1 Break」を参照してください。

(2) Trigger Window による擬似割込み発生

Trigger Window 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、「5.21 Trigger」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残します。

【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[Priority]**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。**

3.14 カバレジ

シミュレータ・デバッガでは、ユーザが指定したアドレス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。

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。表示内容は以下のとおりです。

- Times（命令実行回数）
- Pass（条件分岐命令の実行結果）
  T：分岐のみ実行
  F：未分岐のみ実行
  T/F：分岐および未分岐実行
  -：分岐命令でないかまたは命令未実行
- 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）

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強調表示することでエディタウィンドウにも表示します。

命令カバレジ情報はファイルに保存/回復できます。

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。

カバレジについての詳細は「5.39 Coverage」から「5.46 Save Coverage Data ダイアログボックス」を参照してください。

4. メニュー

このマニュアルでは、標準的な Microsoft®メニュー命名規約を使用しています。

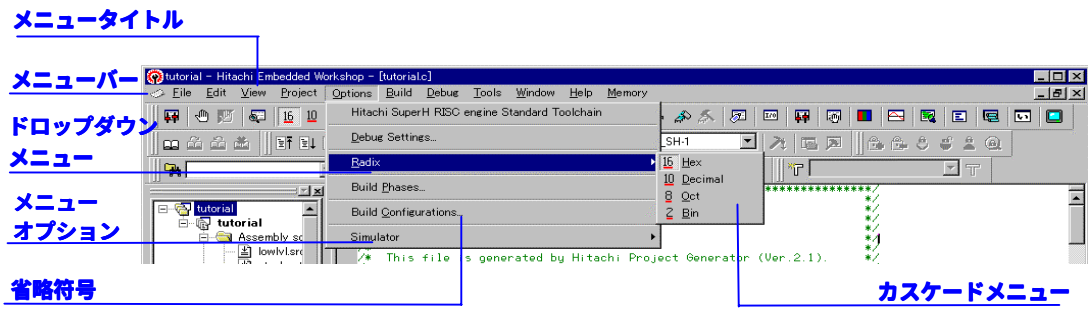


図 4-1 メニュー

チェックマークは、そのメニューオプションにより提供する機能を選択していることを示します。
省略符号は、そのメニューオプションを選択すると、追加情報の入力が必要なダイアログボックスを表示することを示します。

Windows®メニューシステムの使用法については、Windows®ユーザーズマニュアルを参照してください。

ここでは、デバッグのためのメニューを説明します。その他のメニューは HEW ユーザズマニュアルを参照下さい。

4.1 View

View メニューは、新規に各ウィンドウを開くために使用します。メニューオプションがグレー表示の場合、そのウィンドウが提供する機能は現在のデバッグプラットフォームで使用できません。

4.1.1 Workspace

Workspace ウィンドウを開きます。ソースファイルの一覧を表示します。

4.1.2 Output

Output ウィンドウを開きます。デバッガ使用時のメッセージを表示します。

4.1.3 Breakpoints

 Breakpoints ウィンドウを開きます。現在設定しているブレークポイントを表示して変更できるようにします。

4.1.4 Command Line

 Command Line ウィンドウを開きます。テキストベースのコマンドを使用して、デバッグプラットフォームを制御できます。これらのコマンドは、バッチファイルから読み込み、結果をログファイルに書き出すことができます。これにより、自動テストが実行できます。

4.1.5 Disassembly

 表示を開始するアドレスを入力できるように Set Address ダイアログボックスを表示します。

4.1.6 IO

 IO ウィンドウを開きます。制御レジスタの参照、変更ができます。

4.1.7 Labels

 プログラム中のシンボル・ラベルを操作できるよう Label ウィンドウを表示します。

4.1.8 Locals

 Locals ウィンドウを開きます。現在の関数において定義している変数の値を表示し、変更できるようにします。PC が C/C++ソースレベル関数の中になければ、ウィンドウは空白となります。

4.1.9 Memory...

 Set Address ダイアログボックスを表示します。Memory ウィンドウ内に表示するメモリブロック位置と表示フォーマットを指定できるようにします。

4.1.10 Performance Analysis

 Performance Analysis ウィンドウを開きます。ユーザプログラムの特定のセクションを呼び出した回数を計測・表示できるようにします。

4.1.11 Profile

 Profile ウィンドウを開きます。関数とグローバル変数のアドレス、サイズ、関数呼出し回数、およびプロファイルデータを表示します。

4.1.12 Registers

 Registers ウィンドウを開きます。現在の CPU のすべてのレジスタとその内容を見ることができます。

4.1.13 Status

 Status ウィンドウを開きます。デバッグプラットフォームの現在の状態、セッション、プログラム名を表示します。

4.1.14 Trace

 Trace ウィンドウを開きます。現在のトレース情報を表示します。

4.1.15 Watch

 Watch ウィンドウを開きます。C/C++ソースレベルの変数を入力し、その内容を表示・変更できるようにします。

4.1.16 Localized Dump…

 Open Localized Dump ダイアログボックスを開きます。Localized Dump ウィンドウ内に表示するメモリ内容の開始アドレス、表示バイト数、および DBSC の種類を入力できるようにします。

4.1.17 Simulated I/O

 Simulated I/O ウィンドウを開きます。標準入出力、ファイル I/O を使用できるようにします。

4.1.18 Stack Trace

 Stack Trace ウィンドウを開きます。現在のスタックトレース情報を表示できるようにします。

4.1.19 Coverage...

 Coverage ウィンドウを開きます。カバレジ情報を表示します。

4.1.20 Image...

 Image ウィンドウを開きます。メモリ内容を画像表示します。

4.1.21 Waveform...

 Waveform ウィンドウを開きます。メモリ内容を波形表示します。

4.1.22 Trigger

 Trigger ウィンドウを開きます。シミュレーション中に手動で割り込みを発生させるためのトリガボタンを表示します。

4.2 Options

Option メニューは、HEW のデバッグインタフェース部の設定変更と、デバッグプラットフォームの設定に使用します。

4.2.1 Debug Settings...

Debug Settings ダイアログボックスを表示します。HEW のデバッグインタフェース部の設定(デバッグプラットフォーム依存の設定ではありません)を変更できるようにします。

4.2.2 Radix

数値を表示したり入力する場合の、(基数の接頭辞を入力しなかった場合の)基数のデフォルトの設定をカスケードメニューとして表示します。現在選択している基数は左にボタンが付いていて、押し下げています。

たとえば現在の基数が Decimal ならば、10 進法の 10 は"10"と表示し、"10"、"H'A"、"0x0a"などと入力できます。現在の基数が Hexadecimal ならば、10 進法の 10 は"0A"と表示し、"A"、"D'10"などと入力できます。

4.2.3 Simulator

(1) System...


Simulator System ダイアログボックスを表示します。デバッグプラットフォームの設定を変更できるようにします。詳しくは**「**5.24 Simulator System **ダイアログボックス」**を参照してください。

(2) Memory Resource...


Simulator Memory Resource ウィンドウを開きます。デバッグプラットフォームの現在のメモリマップを表示し、変更できるようにします。

4.3 Debug

Debug メニューは、デバッグプラットフォームにおけるユーザプログラムの実行を制御するために使用します。

4.3.1 Reset CPU


ターゲットハードウェアをリセットし、PC をリセットベクタアドレスに設定します。

4.3.2 Go


現在の PC からユーザプログラムを実行します。

4.3.3 Reset Go


リセットベクタアドレスからユーザプログラムを実行します。

4.3.4 Go To Cursor


現在の PC からユーザプログラムの実行を開始し、PC が現在のテキストカーソル(マウスカーソルではありません)の位置によって示すアドレスに到達するまで継続します。

4.3.5 Set PC To Cursor

 PC を現在のテキストカーソル(マウスカーソルではありません)の位置によって示すアドレスに設定します。アドレスが有効でなければ、無効です。

4.3.6 Run...

Run Program ダイアログボックスを表示します。ユーザプログラムの実行開始前にブレークポイントを入力できます。

4.3.7 Step In

 ユーザプログラムの 1 ブロックを実行して停止します。このブロックのサイズは、通常は単一の命令ですが、ユーザが複数の命令または C/C++ソース行に設定することも可能です(4.3.10 Step...参照)。サブルーチンを呼び出した場合は、そのサブルーチンに入って実行を停止し、サブルーチンのコードを表示します。

4.3.8 Step Over

 ユーザプログラムの 1 ブロックを実行して停止します。このブロックのサイズは、通常は単一の命令ですが、ユーザが複数の命令または C/C++ソース行に設定することも可能です(4.3.10 Step...参照)。サブルーチンを呼び出す場合は、そのサブルーチンには入らず、現在の PC 位置が現在の表示の次行に移動するまでユーザプログラムを実行します。

4.3.9 Step Out

 現在の関数の終わりに到達するまでユーザプログラムを実行し、呼び出す関数の次の行に PC を設定して停止します。

4.3.10 Step...

Step Program ダイアログボックスを表示します。ステップ動作の設定を変更できるようにします。

4.3.11 Step Mode

Step Mode を設定します。ステップの単位を Auto（自動選択）、Assembly（アセンブラ命令単位）、Source（C/C++ソースレベル）から選択できます。

4.3.12 Halt Program

 ユーザプログラムの実行を停止します。

4.3.13 Initialize

デバッグプラットフォームを切断し、再接続します。

4.3.14 Disconnect

デバッグプラットフォームを切断します。

4.3.15 Download Modules

オブジェクトプログラムをロードします。

4.3.16 Unload Modules

オブジェクトプログラムをアンロードします。

4.4 Memory

Memory メニューは、ユーザプログラムがアクセスするメモリの設定に使用します。

4.4.1 Search...

メモリ領域の開始アドレス、終了アドレスおよび検索するデータ値を指定し、データ値を検索するための Search Memory ダイアログボックスを表示します。検索条件として、一致／不一致、検索方向を指定できます。

4.4.2 Copy...

 Copy Memory ダイアログボックスを表示します。デバッグプラットフォームの同一メモリスペース内で、メモリブロックを別の位置にコピーします。これらのブロックは重なってもかまいません。ダイアログボックスを表示したときに自動的に設定する開始、終了アドレスは、Memory ウィンドウ中の反転表示したメモリブロックです。コピー元とコピー先のデータを比較しながらコピーすることも指定できます。

4.4.3 Compare...

メモリ領域の開始アドレスと終了アドレスを指定し、別のメモリ領域と比較するための Compare Memory ダイアログボックスを表示します。ダイアログボックスを表示したときに自動的に設定する開始、終了アドレスは、Memory ウィンドウ中の反転表示したメモリブロックです。

4.4.4 Fill...

 Fill Memory ダイアログボックスを表示します。デバッグプラットフォームのメモリブロックに値を書き込みます。ダイアログボックスを表示したときに自動的に設定する開始、終了アドレスは、Memory ウィンドウ中の反転表示したメモリブロックです。

4.4.5 Refresh

すべてのオープンしている Memory ウィンドウの内容を強制的にアップデートします。

4.4.6 Configure Overlay...

 Overlay ダイアログボックスを表示します。オーバーレイ機能を利用した場合、優先するセクショングループを設定することができます。

5. ウィンドウおよびダイアログボックス

本章では、各ウィンドウおよびダイアログボックスの種類と、それぞれがサポートしている機能、および関連ポップアップメニューにより使用できるオプションについて説明します。

5.1 Break

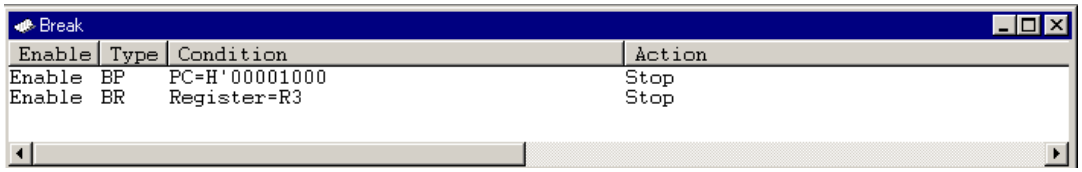


図 5-1 Break ウィンドウ

本ウィンドウは、設定している全ブレークポイントをリスト表示します。
表示する項目は以下の通りです。

[Enable] 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

[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

[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

また、本ウィンドウでブレークポイントをダブルクリックすると、Set Break 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。

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。

5.1.1 Add...

ブレークポイントを設定します。クリックすると、Set Break ダイアログボックスが開き、ブレーク条件を設定することができます。

5.1.2 Edit...

ブレークポイントを１つ選択している場合のみ有効です。変更したいブレークポイントを選択後クリックすると、Set Break 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。

5.1.3 Enable

選択しているブレークポイントを有効にします。

5.1.4 Disable

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。

5.1.5 Delete

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、Disable オプションを使用します(「5.1.4 Disable」参照)。

5.1.6 Delete All

全てのブレークポイントを削除します。

5.1.7 Go to Source

ブレークポイントを１つ選択している場合のみ有効です。ブレークポイントのある Source または Disassembly ウィンドウをオープンします。

5.1.8 Close File

選択した File Input または File Output 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。

5.1.9 Close All Files

全ての File Input および File Output 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。

5.2 Set Break ダイアログボックス（Condition シート）

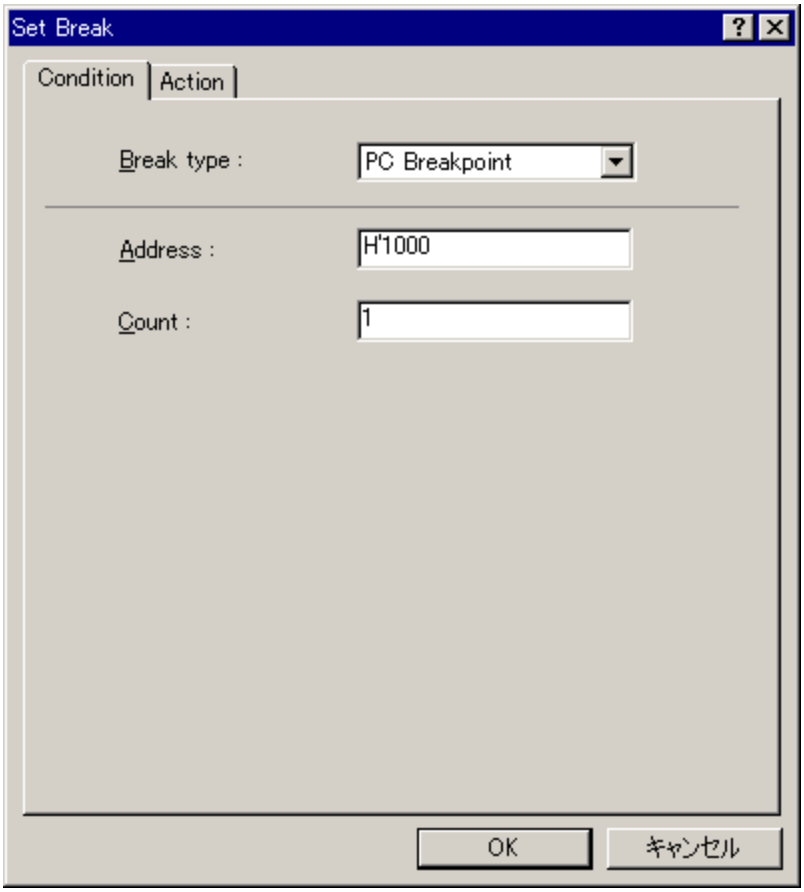


図 5-2 Set Break **ダイアログボックス**（Condition **シート**）

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 [Break type]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 Breakpoint]	255 個まで指定可能	
	[Address]	ブレークする命令の位置
	[Count]	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）
[Break Access]	2 個指定可能	
	[Start address]	アクセスするとブレークするメモリの開始位置
	[End address]	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）
	[Access type]	アクセス種別
[Break Data]	8 個まで指定可能	
	[Address]	ブレーク判定を行うメモリの位置
	[Data]	ブレーク条件となるデータ値
	[Size]	データのサイズ
	[Option]	データの一致/不一致
[Break Register]	8 個まで指定可能	
	[Register]	ブレーク条件を設定するレジスタ名
	[Size]	データのサイズ
	[Data]	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）
	[Option]	データの一致/不一致
[Break Sequence]	1 個のみ指定可能	
	[Address1]～[Address8]	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。）
[Break Cycle]	255 個まで指定可能	
	[Cycle]	ブレーク判定を行うサイクル数（1～H’FFFFFFFF） [Cycle]×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。
	[Count]	ブレークが成立する回数
	[ALL]	条件が一致するごとに、ブレークが成立します。
	[Times]	（接頭辞省略時は 10 進入力、10 進表示）（１~65535） 条件一致した回数が、[Times]以下の時だけブレークが成立します。

[PC Breakpoint]および[Break Sequence]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、Select Function 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、**「HEW デバッガ ユーザズマニュアル」**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。[Cancel]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。

5.3 Set Break ダイアログボックス（Action シート）

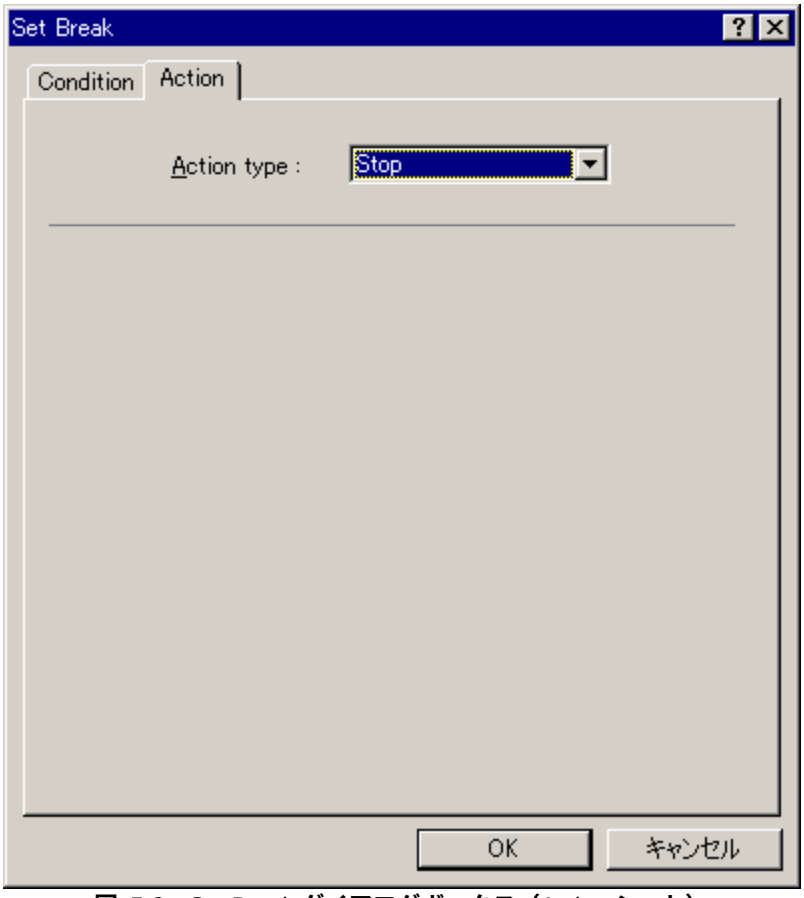


図 5-3 Set Break **ダイアログボックス（**Action **シート）**

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 [Action type]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。

[Stop]	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。	
[File Input]	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。	
	[Input file]	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。
	[Address]	データを書きこむメモリのアドレスを指定します。
	[Data size]	書きこむデータ１個のサイズを指定します。（1/2/4/8）
	[Count]	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）
[File Output]	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。	
	[Output file]	書きこむデータファイルを指定します。
	[Append]	既存のファイルを[Output File]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。
	[Address]	データを読みこむメモリのアドレスを指定します。
	[Data size]	読みこむデータ１個のサイズを指定します。（1/2/4/8）
	[Count]	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）
[Interrupt]	条件成立時に割込み処理を行います。詳しくは、「3.13 擬似割込み」を参照してください。	
	[Interrupt type1]	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）
	[Priority]	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 File Input 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 File Output 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、File Input と File Output 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。

5.4 Command Line

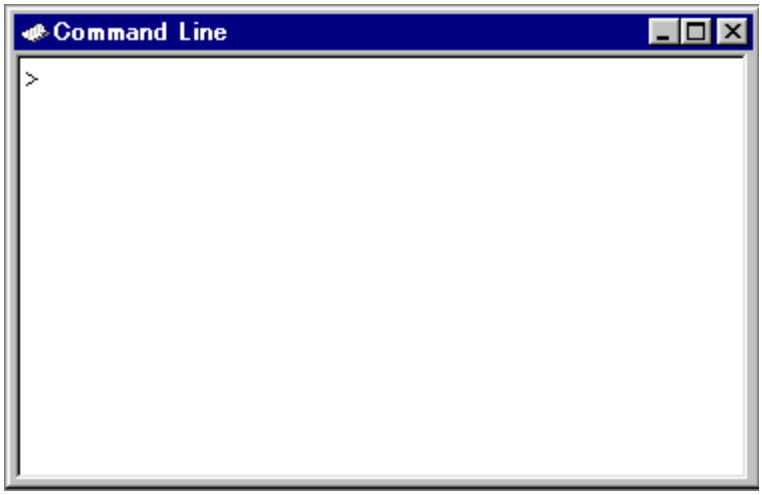


図 5-4 Command Line **ウィンドウ**

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プラットフォームを制御できるウィンドウです。あらかじめ定義した一連のコマンドをバッチファイルから呼び出してデバッグプラットフォームに送り、出力結果をログファイルに記録する必要がある場合に便利です。最終行に入力後、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、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。

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。

5.4.1 Set Batch File...

Set Batch File 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。

[Play]ボタンをクリックすることにより、ダイアログボックスを閉じて、設定したコマンドファイルを実行します。[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。[Cancel]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 5-5 Set Batch File **ダイアログボックス**

5.4.2 Play

Set Batch File ダイアログボックスで指定したコマンドファイルを実行します。バッチファイルの実行中はグレー表示となり、コマンドファイルの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。

5.4.3 Stop

コマンドを中断します。コマンド実行中に有効表示になります。

5.4.4 Set Log File...

Open Log File ダイアログボックスを表示します。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。

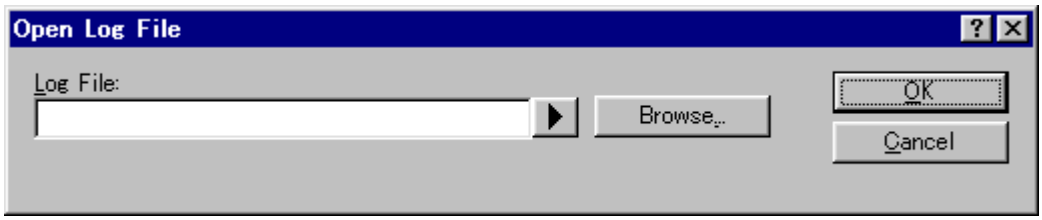


図 5-6 Open Log File **ダイアログボックス**

5.4.5 Logging

ファイルへのロギング処理を実行するか、停止するかを切り替えます。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。

5.4.6 Browse…

Browse ダイアログボックスを表示します。選択したファイルのフルパス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。

5.4.7 Placeholder

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。

5.4.8 Select All

Command Line ウィンドウ出力をすべて選択します。

5.4.9 Copy

テキストブロックを反転表示している場合にのみ使用できます。反転表示したテキストをWindows®クリップボードにコピーし、他のアプリケーションへ貼り付けられるようにします。

5.4.10 Paste

Windows®クリップボードの内容を現在の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。

5.5 Disassembly

本ウィンドウは、アセンブラレベルでプログラムを表示します。

アセンブラ情報はメモリ内容を逆アセンブルします。オブジェクトファイルからデバッグ情報を読まずにメモリ内容を直接編集、表示します。

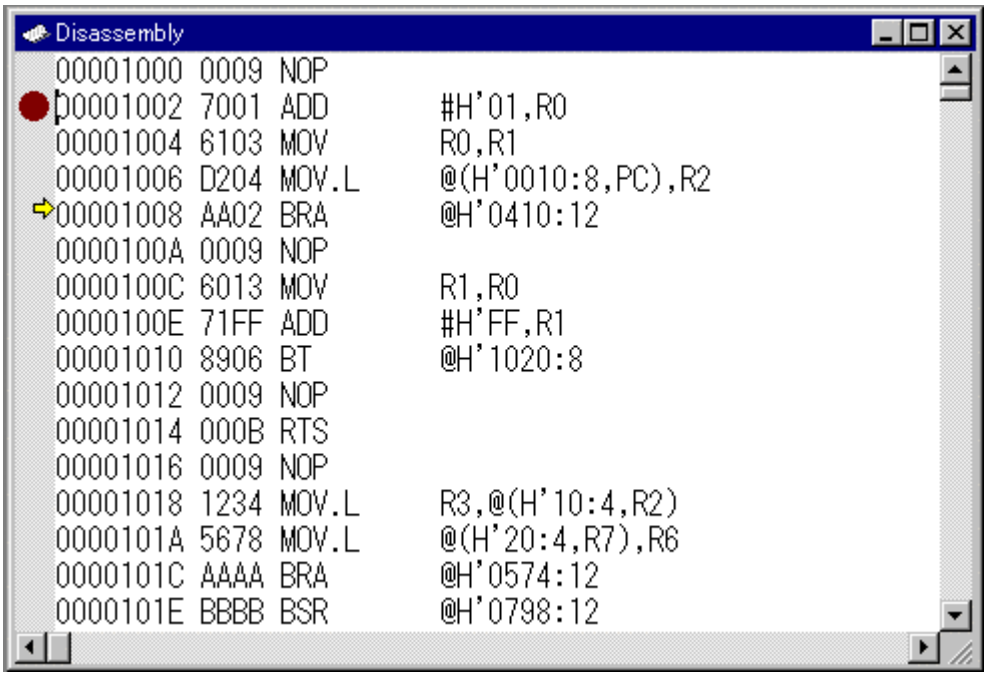


図 5-7 Disassembly **ウィンドウ**

本ウィンドウでは左より、アドレス情報、アドレス、命令コード、命令ニーモニックを表示します。

アドレス情報として下記を表示します。

- ブックマークを設定している
- PC Break を設定している
- PC 位置

ウィンドウ内で右クリックすると使用可能なオプションをポップアップメニューで表示します。

5.5.1 View Source

テキストカーソル (マウスカーソルではありません)の位置に対応する Source ウィンドウをオープンします。ソース行が有効なときのみ機能します。

5.5.2 Go to cursor

 現在の PC アドレスからプログラムを実行します。プログラムは、PC がテキストカーソル(マウスカーソルではありません)の位置によって指定したアドレスに達するか、別のブレーク条件が成立するまで実行します。本機能は PC ブレークポイントを使用しています。PC ブレークポイントがすでに 255 個設定してある場合は、本機能は使用できません。

5.5.3 Set Address...

Set Address ダイアログボックスを表示します。表示を開始するアドレスを指定します。

5.5.4 Set PC Here

PC の値をテキストカーソル(マウスカーソルではありません)の位置によって指定したアドレスに変更します。

5.5.5 Edit...

Assembler ダイアログボックスを表示します。任意のアドレスの命令を変更できます。機械語を変更しても、ソースファイルには反映しません。また、シミュレータ終了時には、変更した情報は保存しませんので注意してください。

5.5.6 Code Bytes

命令コードを表示するかを切り替えます。

5.5.7 Toggle Breakpoint

PC ブレークポイントの有効／無効を切り替えます。

5.6 IO

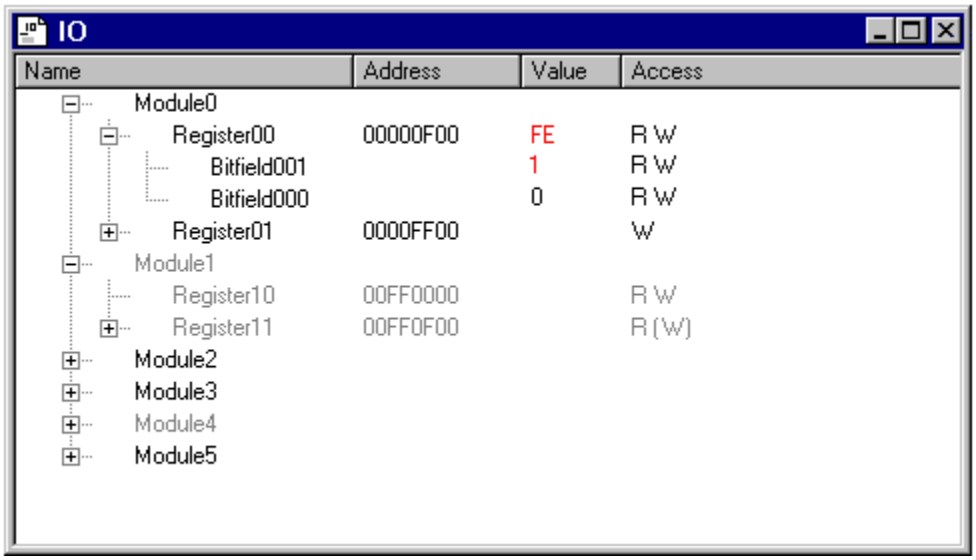


図 5-8 IO Area **ウィンドウ**

制御レジスタの参照、設定ができます。

‘+’記号および’-‘記号をダブルクリックするか、’+’キーおよび’-‘キーを入力すると制御レジスタの情報を拡張／縮小表示します。

名前をダブルクリックすると Edit Register ダイアログボックスを表示します。制御レジスタの値を変更できます。

5.7 Label

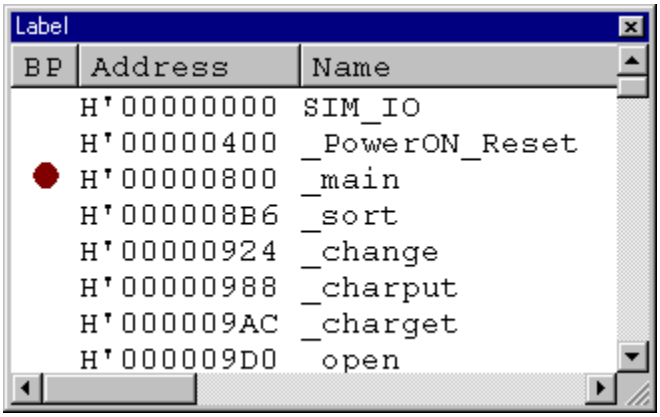


図 5-9 Label **ウィンドウ**

カラムヘッダをクリックすることで、シンボルをアルファベット順またはアドレス順にソートして表示します。

各列に対してダブルクリック動作をサポートしています。

- BP
  そのアドレスに対し、PCブレークポイントを設定または解除します。

ウィンドウ内で右クリックすると使用可能なオプションをポップアップメニューで表示します。

5.7.1 Add...

Add Label ダイアログボックスを表示します。

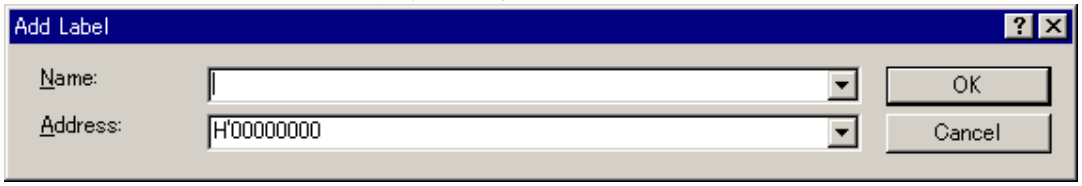


図 5-10 Add Label **ダイアログボックス**

新しいラベル名を Name フィールドに入力し、対応する値を Address フィールドに入力して[OK]を押します。Add Label 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、Select Function ダイアログボックスが開くので、関数を選択して Address フィールドを設定します。詳細は**「**HEW **デバッガ ユーザズマニュアル」**を参照してください。

5.7.2 Edit...

Edit Label ダイアログボックスを表示します。

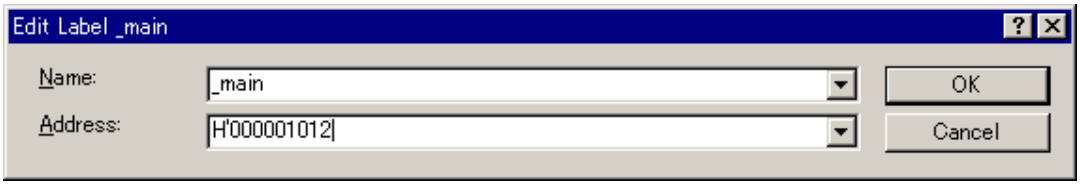


図 5-11　Edit Label **ダイアログボックス**

ラベル名と対応する値を編集して、[OK]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、Select Function ダイアログボックスが開くので、関数を選択して Address フィールドを設定します。詳細は**「HEW デバッガ ユーザズマニュアル」**を参照してください。

5.7.3　Delete

シンボルリストから選択したラベルを削除します。Delete キーでも同様の動作です。確認メッセージを表示します。

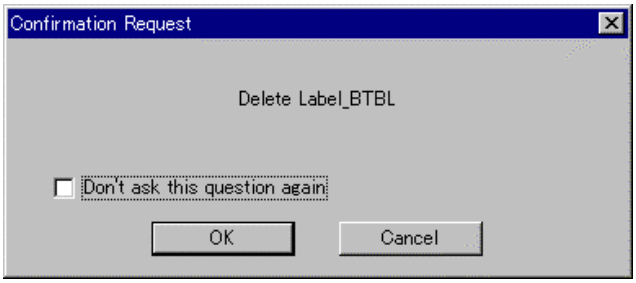


図 5-12　**ラベル削除確認メッセージボックス**

[OK]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不要のときは、HEW の Options ダイアログボックスの Confirmation シートの Delete Label オプションを選択しないでください。

5.7.4 Delete All

リストからすべてのラベルを削除します。確認メッセージボックスを表示します。

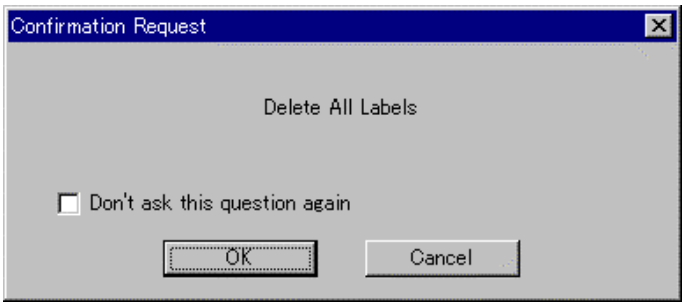


図 5-13 Confirming All Label Deletion **メッセージボックス**

[OK]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、HEW の Options ダイアログボックスの Confirmation シートの Delete All Labels オプションを選択しないでください。

5.7.5 Load...

現在の HEW のシンボルテーブルに結合します。Load Symbols ダイアログボックスをオープンします。

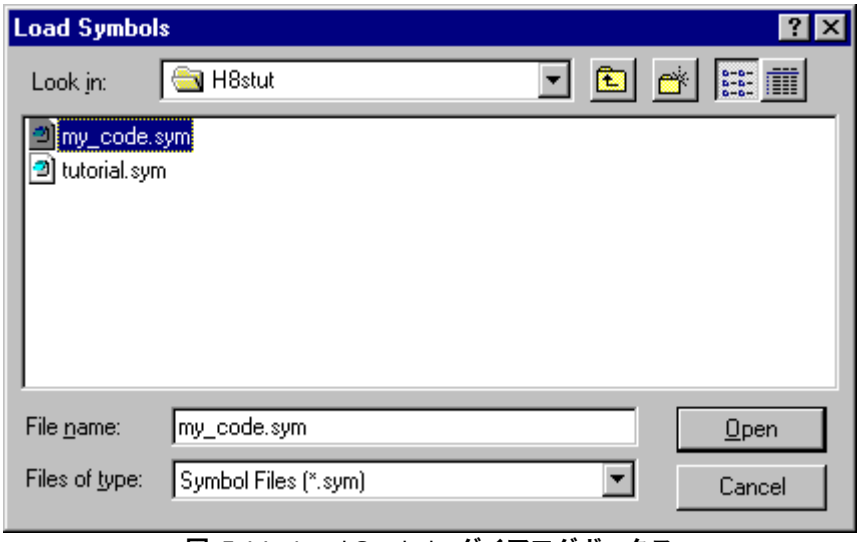


図 5-14 Load Symbols **ダイアログボックス**

ダイアログボックスは、Windows®標準の open file ダイアログボックスと同様です。ファイルを選択し、[Open] 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。シンボルのロードが完了するとロードしたシンボル数を表示したメッセージボックスを表示します。このメッセージボックスは、HEW Options ダイアログボックスの Confirmation シートで非表示にもできます。

5.7.6 Save

現在のシンボルテーブルをシンボルファイルに保存します。

5.7.7 Save As...

Save Symbols ダイアログボックスは Windows®標準の Save File As ダイアログボックスと同様に操作できます。File name フィールドにファイル名を入力し[Open] 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 HEW デバッガ ユーザーズマニュアル付録 B にあるので参照してください。

5.7.8 Find...

Find Label ダイアログボックスを表示します。

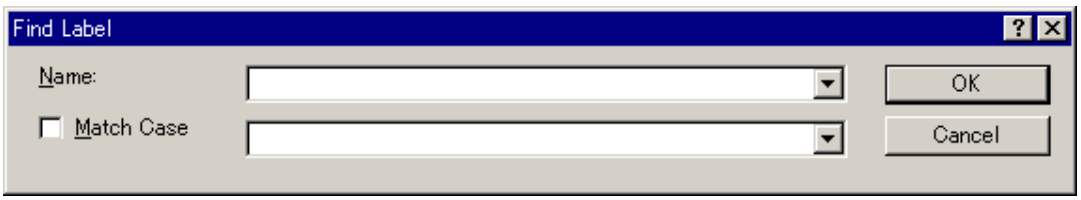


図 5-15 Find Label ダイアログボックス

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、ダイアログボックスはクローズし、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。

5.7.9 Find Next

ラベルが検索できた後、検索条件に一致する次のラベルを検索します。

5.7.10 View Source

ラベルのアドレスに該当する Source または、Disassembly ウィンドウをオープンします。

5.8 Locals

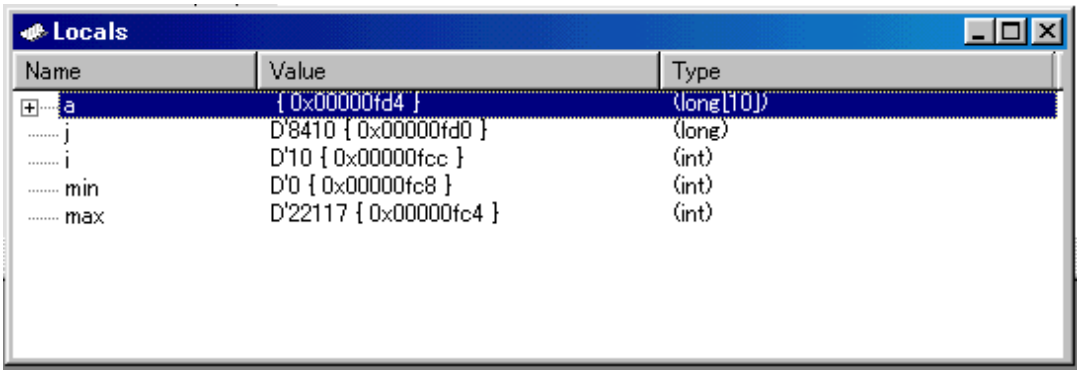


図 5-16 Locals **ウィンドウ**

すべてのローカル変数を表示･変更できるウィンドウです。このウィンドウは、アブソリュートファイル (*.abs)が含んでいるデバッグ情報によって、現在の PC 位置から関数内にあるローカル変数に関連づけることができない場合は空白となります。

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

変数をリスト表示し、'+'記号は変数名をダブルクリックすれば情報を拡張表示できることを、'-'記号は情報を収縮表示できることを示します。また、'+'／'-'キーでも拡張／収縮表示することができます。情報の表示については、**「**HEW **デバッガ ユーザズマニュアル」**を参照してください。

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。このメニューは次のオプションを含みます。

5.8.1 Edit Value...

選択しているローカル変数の値を変更するダイアログボックスを表示して、変数の値を変更できます。

5.8.2 Radix

選択しているローカル変数の表示基数を変更します。

5.8.3 Copy

テキストブロックを反転表示している場合にのみ使用できます。反転表示したテキストを Windows®クリップボードにコピーし、他のアプリケーションに貼り付けられるようにします。

5.9 Memory

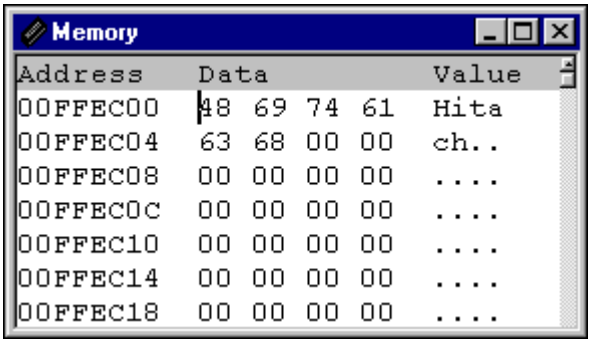


図 5-17 Memory **ウィンドウ**

デバッグプラットフォームのメモリ内容を表示・変更できるウィンドウです。メモリは ASCII、バイト、ワード、ロングワード、単精度浮動小数点、倍精度浮動小数点の各フォーマットで表示できます。タイトルバーは現在の表示スタイルと、直前のラベル(シンボル)からのオフセットで示したアドレスを示します。

メモリ内容は、データ項目をダブルクリックして変更することができます。データ項目をダブルクリックすると Edit ダイアログボックスを表示するので、複雑な式を使用して新しい値を入力できます。そのアドレスのデータが変更できない(すなわち ROM または未使用領域上にある)場合は、"Invalid address value"というメッセージを表示します。

Address 列でダブルクリックすると Set Address ダイアログボックスを表示して、新しい開始アドレスを入力できるようになります。[OK]をクリックするとウィンドウを更新して、入力したアドレスが、１行目になるように表示します。

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。このメニューには以下のオプションを含みます。

5.9.1 Lock Refresh

ユーザプログラム実行停止時などに、自動的に Memory ウィンドウ内容を更新しないようにします。

5.9.2 Refresh

Memory ウィンドウの内容を強制的にアップデートします。

5.9.3 Start Address...

Set Address ダイアログボックスを表示します。新しい開始アドレスと終了アドレスを入力すると、ウィンドウを更新し、ユーザが入力したアドレスが左上端のアドレスとなります。アドレスとして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、Select Function 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、**「**HEW **デバッガ ユーザーズマニュアル** 7.3 **複数ラベルをサポートする」**を参照してください。データ表示サイズを指定することができます。

5.9.4 Format...

Format Memory Display ダイアログボックスを表示します。データ表示サイズ、表示形式、フォントを指定できます。

5.9.5 Search...

Search Memory ダイアログボックスを表示します。デバッグプラットフォームのメモリブロックの中で指定したデータ値を検索します。メモリブロックを反転表示している場合は、ダイアログボックスの開始フィールドと終了フィールドに、反転表示しているブロックに対応する開始アドレスと終了アドレスを自動的に設定します。検索条件として、一致／不一致、検索方向を指定できます。検索条件に合ったデータのアドレスから Memory ウィンドウを表示します。

5.9.6 Search Next

Search …で見つかった場合に本機能は有効になります。検索した次のアドレスから検索します。

5.9.7 Copy...

Copy Memory ダイアログボックスを表示します。デバッグプラットフォームのメモリブロックを同一メモリ空間内の別の場所にコピーします。ブロックはオーバラップしても構いません。開始フィールドと終了フィールドは、Search オプション(「5.9.5 Search...」も参照)と同様に設定します。コピー元とコピー先のデータを比較しながらコピーすることも指定できます。

5.9.8 Compare...

Compare Memory ダイアログボックスを表示します。メモリ領域の開始アドレスと終了アドレスを選択して別のメモリ領域と比較します。Memory 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。

メモリベリファイに似ていますが、本機能は、2 つのメモリブロックを比較します。

5.9.9 Fill...

Fill Memory ダイアログボックスを表示します。デバッグプラットフォームのメモリブロックに指定した値を書き込みます。開始フィールドと終了フィールドは、Search オプション(「5.9.5 Search...」も参照)と同様に設定します。

5.9.10 Save...

Save Memory As ダイアログボックスを表示します。デバッグプラットフォームのブロックを S-record ファイル(*.mot)でセーブします。開始フィールドと終了フィールドは、Search オプション(「5.9.5 Search...」も参照)と同様に設定します。

5.9.11 Load...

Load Memory ダイアログボックスを表示します。現在のデバッグ情報を削除せずにデバッグプラットフォームのメモリに S-record ファイル(*.mot)をロードします。offset フィールドは、アドレス値を変更するときに指定します。オプショナルベリファイフラグは正常にダウンロードしたかどうかをチェックします。

5.10 Performance Analysis

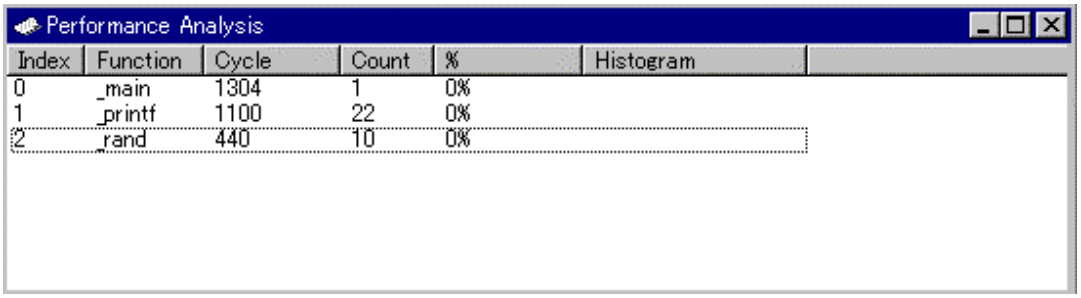


図 5-18 Performance Analysis **ウィンドウ**

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、指定関数コール命令実行時の累計実行サイクル数と指定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。

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

また、評価関数をダブルクリックすると、Performance Option ダイアログボックスが開き、関数を変更することができます。なお、関数は 255 個まで設定可能です。
表示領域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。

5.10.1 Add Range...

評価関数を追加します。クリックすると、Performance Option ダイアログボックスが開き、関数を追加することができます。また、'Insert'キーでも Performance Option ダイアログボックスを開くことができます。

5.10.2 Edit Range

反転表示カーソルバーがユーザ定義範囲にある場合にのみ有効です。Performance Option ダイアログボックスを表示して、範囲の設定を変更します。また、'Enter'キーでも Performance Option ダイアログボックスを開くことができます。

5.10.3 Reset Counts/Times

現在のプログラムの実行効率データをクリアします。

5.10.4 Enable Analysis

実行効率データ収集のオン・オフを切り替えます。実行効率測定が現在アクティブであれば、テキストの左にチェックマークを表示します。

5.10.5 Delete Range

反転表示カーソルバーがユーザ定義範囲にある場合にのみ有効です。範囲設定を削除し、他の範囲のデータを再計算します。また、'Delete'キーでも範囲設定を削除することができます。

5.10.6 Delete All Ranges

現在のユーザ定義範囲をすべて削除し、実行効率測定データをクリアします。

5.11 Performance Option ダイアログボックス

図 5-19 Performance Option **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。評価結果は Performance Analysis ウィンドウで表示します。

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、Select Function 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、**「**HEW **デバッガ ユーザーズマニュアル** 7.3 **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。[Cancel]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。

5.12 Registers

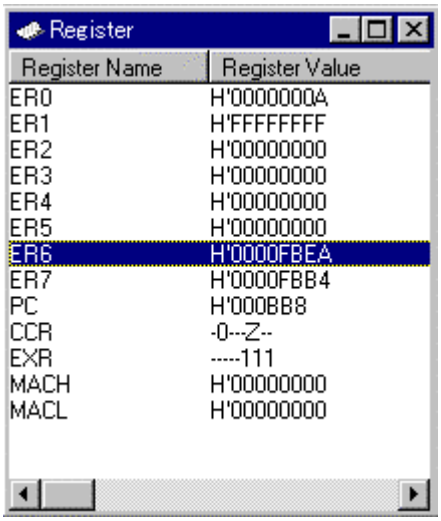


図 5-20 Registers **ウィンドウ**

現在のレジスタ値を表示・変更できるウィンドウです。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。

5.12.1 Edit...

Edit Register ダイアログボックスを表示します。テキストカーソル (マウスカーソルではありません)の位置によって示したレジスタの値を設定します。

5.13 Source

Source ウィンドウは、C/C++、アセンブラプログラムで、デバッグオプションを指定していれば表示できます。また、カバレジ情報も表示します。

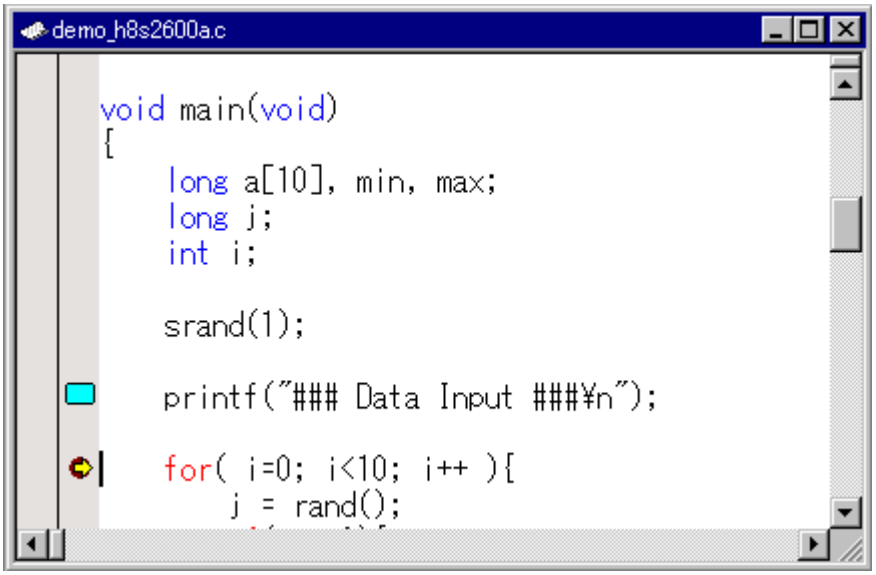


図 5-21 Source **ウィンドウ**

本ウィンドウでは左はじに行情報として下記を表示します。

- ブックマークを設定している
- PC Break を設定している
- PC 位置

ウィンドウ内で右クリックすると使用可能なオプションをポップアップメニューで表示します。

5.13.1 Toggle Breakpoint

PC ブレークポイントの設定／解除を切り替えます。

5.13.2 Enable/Disable Breakpoint

PC ブレークポイントの有効／無効を切り替えます。

5.13.3 Instant Watch…

Instant Watch 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。

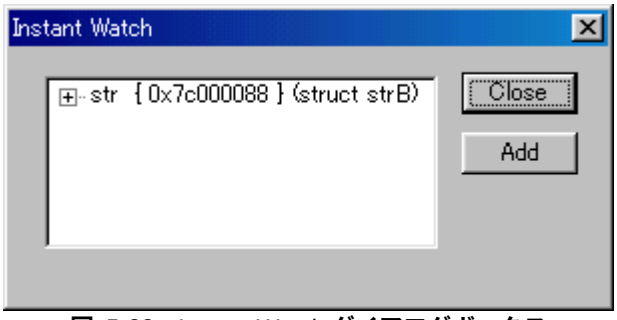


図 5-22 Instant Watch **ダイアログボックス**

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。[Add]を押すと、変数を Watch ウィンドウに登録して、ダイアログボックスを閉じます。[Close]を押すと、変数を Watch 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。

5.13.4 Go To Cursor

現在の PC アドレスからユーザプログラムの実行を開始します。プログラムは、PC がテキストカーソル(マウスカーソルではありません)の位置によって指定したアドレスに達するか、別のブレーク条件が成立するまで実行します。本機能は PC ブレークポイントを使用しています。PC ブレークポイントがすでに 255 個設定してある場合は、本機能は使用できません。

5.13.5 Set PC Here

PC の値をテキストカーソル(マウスカーソルではありません)の位置によって指定したアドレスに変更します。

5.13.6 Go To Disassembly

現在のソース行に対応したアドレスのDisassembly表示を行います。

5.14 Source address **カラム**

プログラムをダウンロードすると、現在のソースファイルに対して、Source ウィンドウがアドレスを更新したり表示したりします。アドレスは Source ウィンドウの左側に表示します。本機能は PC 値やブレークポイントをどこに設定するかを決めるときに便利です。図 5-23にSource ウィンドウと address カラムを示します。

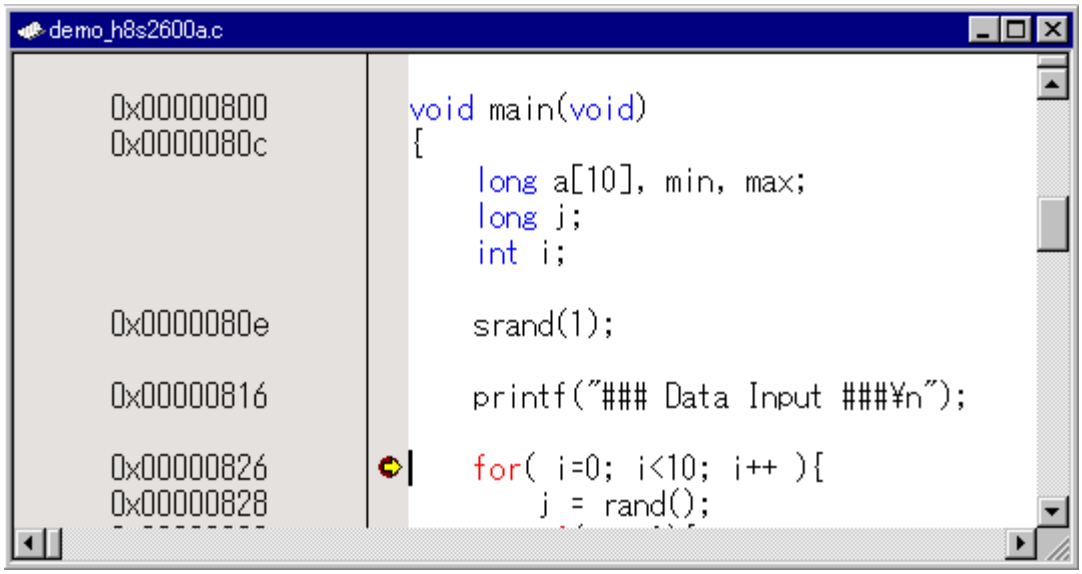


図 5-23　Source **ウィンドウと** address **カラム**

すべてのファイルについて address カラムを表示/非表示にする手順は以下のとおりです。

1. エディタウィンドウ上で右クリックしてください。
2. [Define Column Format...]メニュー項目をクリックしてください。
3. [Global Editor Column States]ダイアログボックスを表示します。
4. Check status はカラム表示/非表示を示します。チェックがある時にカラムを表示します。チェックボックスがグレーの場合、一部のファイルではカラム表示が有効で、他のファイルでは無効であることを意味します。
5. OK 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。

一つのファイルについて address カラムを表示/非表示にする手順は以下のとおりです。

1.カラム表示を変更したいエディタウィンドウ上で右クリックしてください。エディタポップアップメニューを表示します。
2.[Columns]メニュー項目をクリックしてください。メニュー項目を一覧表示します。各カラムは、このポップアップメニューに表示します。カラムが表示状態ならば名前の横にチェックマークが付いています。カラム名のクリックにより表示/非表示を切り替えてください。

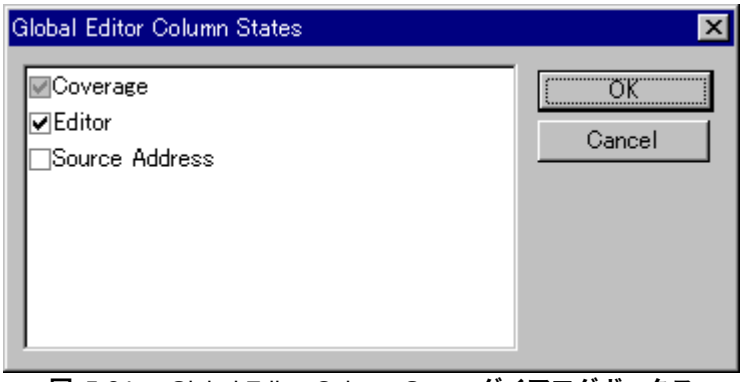


図 5-24 Global Editor Column States ダイアログボックス

5.15 Debugger カラム

どのコンポーネントも Disassembly および Source ウィンドウにカラムを追加することができます。カラムの例として、デバッガ実行中にコードカバレジをグラフィカルに表示するカバレジカラムや、ターゲットに設定しているハードウェアブレークポイントを表示するターゲットコンポーネントカラムがあります。

カラム上で右クリックすると、そのカラムに対するポップアップメニューを表示します。メニューはカラムごとに異なり、またカラム上でのダブルクリックもカラムにより異なった働きをします。例えば、ターゲットカラムではハードウェアブレークポイントの設定となります。

5.16 Status

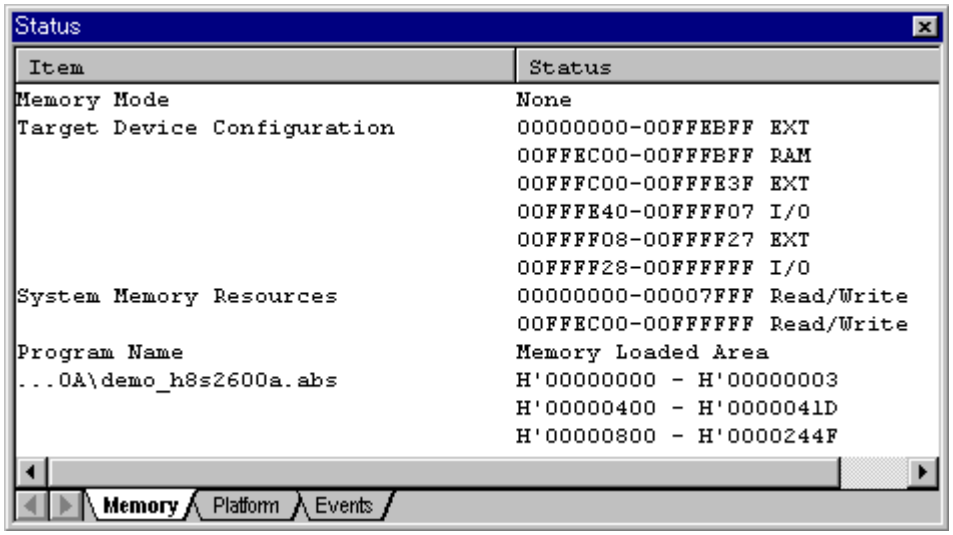


図 5-25 Status ウィンドウ

デバッグプラットフォームの現在のステータスを表示できるウィンドウです。
Status 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、現在のメモリステー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。

5.17 Trace

本ウィンドウは、トレース情報を表示します。なお、トレース情報の取得条件は、Trace Acquisition ダイアログボックスで設定します。

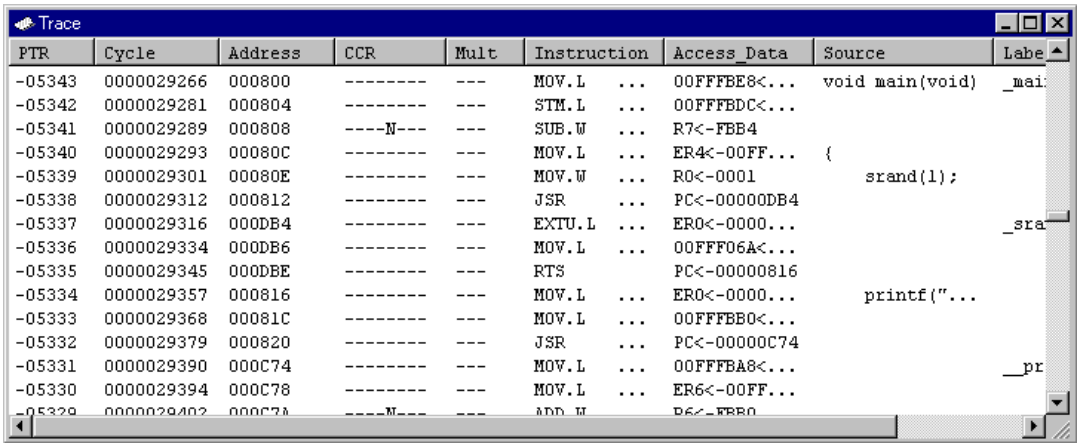


図 5-26 Trace ウィンドウ

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）の内容をニーモニックで表示します。
[Mult]	乗算器内部のフラグをニーモニックで表示します。（H8S/2600 シリーズのみ）
[Instruction]	命令ニーモニック
[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

また、Trace ウィンドウである行をダブルクリックすると、Source ウィンドウあるいは Disassembly ウィンドウを開いてソース表示し、選択した行をカーソルで示します。

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。

5.17.1 Find...

Trace Search ダイアログボックスを表示します。現在のトレースバッファ内の特定のトレースレコードを検索できます。

5.17.2 Find Next

トレースレコードが検索できた後、次の同様のトレースレコードをさらに検索できます。

5.17.3 Acquisition...

Trace Acquisition ダイアログボックスを表示します。トレース情報の取得条件を設定できます。

5.17.4 Clear

トレースバッファを空にします。複数の Trace 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、すべてのトレースバッファをクリアします。

5.17.5 Save...

Save As 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(すべてのバッファをセーブするには、数分かかることがあります)。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。

5.17.6 View Source

該当アドレスに対応したソースプログラムを表示します。

5.17.7 Trim Source

ソースプログラムの左側の空白を取り除きます。

5.17.8 Statistic

指定された条件で統計情報の解析を実行します。

5.18 Trace Acquisition ダイアログボックス

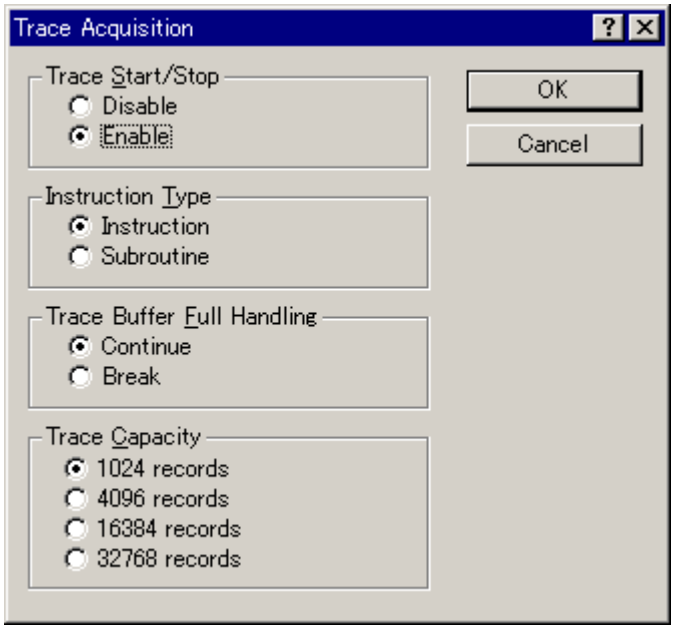


図 5-27 Trace Acquisition **ダイアログボックス**

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。

[Trace Start/Stop]

[Disable]	トレース情報の取得停止
[Enable]	トレース情報の取得開始

[Instruction Type]

[Instruction]	全命令のトレース情報を取得
[Subroutine]	サブルーチン命令のみトレース情報を取得

[Trace Buffer Full Handling]

[Continue]	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
[Break]	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、[Trace Capacity]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。[Cancel]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。

5.19 Trace Search ダイアログボックス

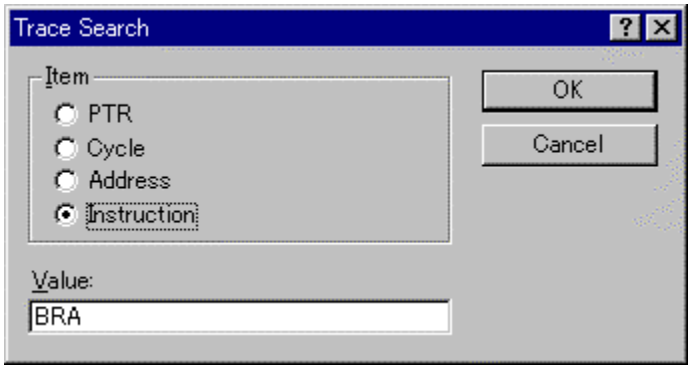


図 5-28 Trace Search **ダイアログボックス**

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。[Item]でサーチ対象項目を指定し、[Value]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。
[Cycle]	累計命令実行サイクル数
[Address]	命令アドレス
[Instruction]	命令ニーモニック

[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ます。[Cancel]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。

5.20 Trace Statistic ダイアログボックス

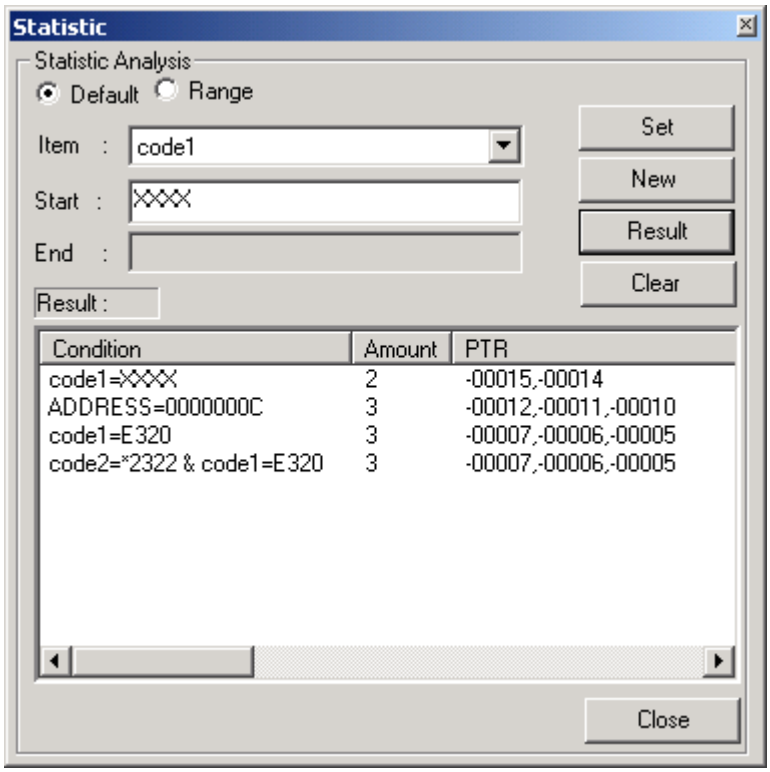


図 5-29 Trace Statistic **ダイアログボックス**

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。[Item]で解析対象項目を指定し、[Start]および[end]で入力値または文字列を指定します。

[Default]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[Range]を選択してください。

[Set]	現在の条件に追加設定します
[New]	新しい条件を指定します
[Result]	統計情報解析の結果を取得します
[Clear]	すべての条件と統計情報解析結果を削除します

[Close]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。

5.21 Trigger

本ウィンドウは、手動で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割込みの内容は、Trigger Setting ダイアログボックスで設定します。

トリガボタンは最大 16 個まで設定できます。

シミュレータ・デバッガの割込み処理については、**「3.13 擬似割込み」**を参照してください。

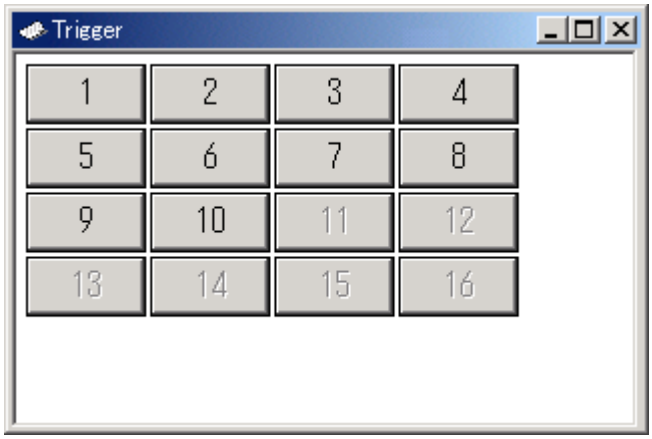


図 5-30 Trigger **ウィンドウ**

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。

5.21.1 Setting...

Trigger Setting ダイアログボックスを表示します。各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。

5.21.2 Size

Trigger ウィンドウに表示するトリガボタンのサイズを指定できます。サブメニューでトリガボタンサイズを Large、Normal、Small から選択します。

5.22 Trigger Setting ダイアログボックス

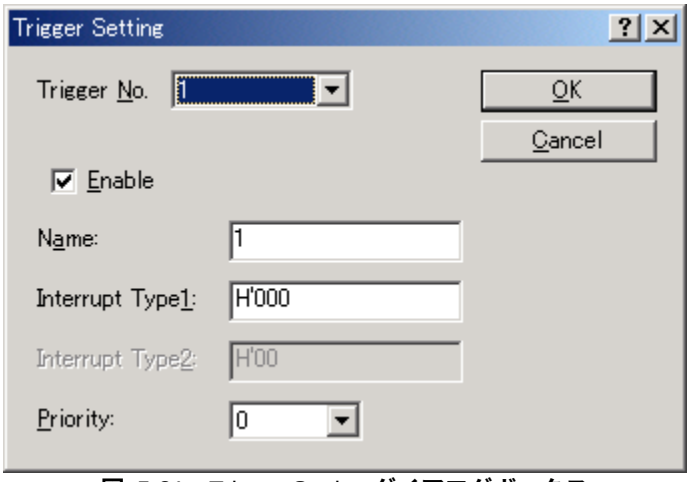


図 5-31 Trigger Setting **ダイアログボックス**

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。

[Trigger No.]	設定するトリガボタンを選択します。
[Name]	Trigger 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。
[Enable]	チェックするとトリガボタンが有効になります。
[Interrupt Type1]	割込みベクタ番号を指定します。（0~H'FF）
[Priority]	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。[Cancel]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[Cancel]ボタンをクリックすると、全てのトリガボタンの設定変更が無効になります。

5.23 Watch

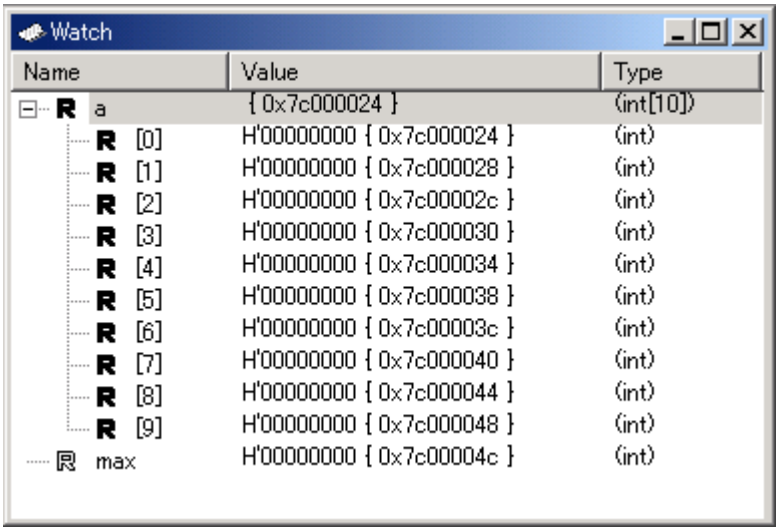


図 5-32 Watch **ウィンドウ**

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

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、'-'記号は情報を収縮表示できることを示します。また、'+'／'-'キーでも拡張／収縮表示することができます。
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。

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。

5.23.1 Auto Update

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。

5.23.2 Auto Update All

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。

5.23.3 Delete Auto Update

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。

5.23.4 Delete Auto Update All

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。

5.23.5 Add Watch...

Add Watch ダイアログボックスを表示します。監視する変数を入力します。

5.23.6 Edit Value...

Edit Value ダイアログボックスを表示して、変数の値を変更します。ポインタの値を変更する場合は、そのポインタが有効なデータを指し示さなくなる可能性があるため、特に注意が必要です。

5.23.7 Delete

選択している変数を、Watch ウィンドウから削除します。

5.23.8 Delete All

すべての変数を、Watch ウィンドウから削除します。

5.23.9 Radix

選択している変数の表示基数を変更します。

5.23.10 Copy

 テキストブロックを反転表示している場合にのみ使用できます。反転表示したテキストを Windows®クリップボードにコピーし、他のアプリケーションに貼り付けられるようにします。

5.23.11 Save As...

Save As ダイアログボックスを表示します。ファイル名を指定し、Watch 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。

5.23.12 Go To Memory...

選択している変数が割り付いているメモリ内容を Memory ウィンドウに表示します。

5.24 Simulator System ダイアログボックス

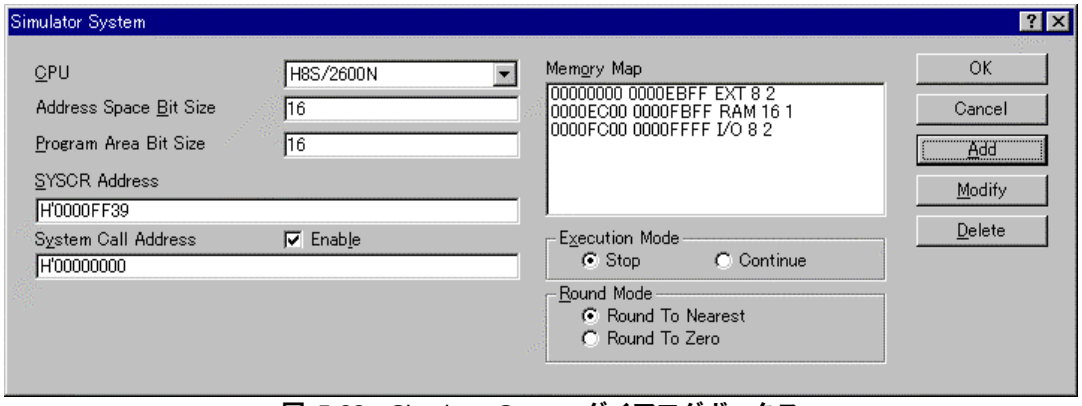


図 5-33 Simulator System **ダイアログボックス**

本ダイアログボックスでは、アドレス空間ビット数、プログラム領域ビット数、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、およびメモリマップの設定を行います。

[CPU] 現在設定している CPU。
（CPU は、Debug Setting ダイアログボックスにより設定します。）

[Address Space Bit Size] 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

[Program Area Bit Size] 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

[System Call Address] デバッグ対象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。
- [Enable] チェックするとシステムコールが有効となります。

[Execution Mode] 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。
- [Stop] シミュレーションを停止します。
- [Continue] シミュレーションを続行します。

[Round Mode] 浮動小数点数 10 進→2 進変換で発生する丸めのモードを指定します。
- [Round to nearest] 最近値丸め
- [Round to zero] ゼロ方向丸め

[Memory Map]には、メモリ情報として、先頭アドレス・終了アドレス、メモリ種別、データバス幅、アクセスサイクル数の順に表示します。

[Memory Map]は、以下の各ボタンにより設定・変更・削除ができます。

[Add]	[Memory Map]の項目を設定します。クリックすると、Memory Map Modify ダイアログボックスが開き、設定することができます。
[Modify]	[Memory Map]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、Memory Map Modify ダイアログボックスが開き、変更することができます。
[Delete]	[Memory Map]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。

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。[Cancel]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。

5.25 Memory Map Modify ダイアログボックス

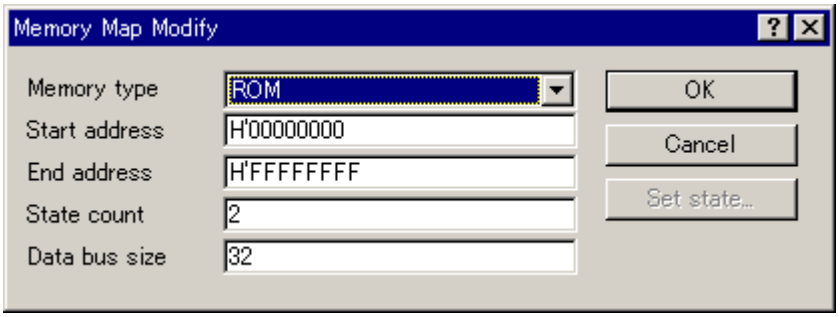


図 5-34 Memory Map Modify **ダイアログボックス**

本ダイアログボックスでは、シミュレータ・デバッガの対象 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。これらの値は、メモリアクセスサイクル数の算出に使用します。

[Memory type]	メモリ種別
[Start address]	メモリ種別に対応するメモリの先頭アドレス
[End address]	メモリ種別に対応するメモリの終了アドレス
[State count]	メモリのアクセスサイクル数
[Data bus size]	メモリのデータバス幅

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。[Cancel]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、Simulator Memory Resource **ダイアログボックス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。**

5.26 Simulator Memory Resource ダイアログボックス

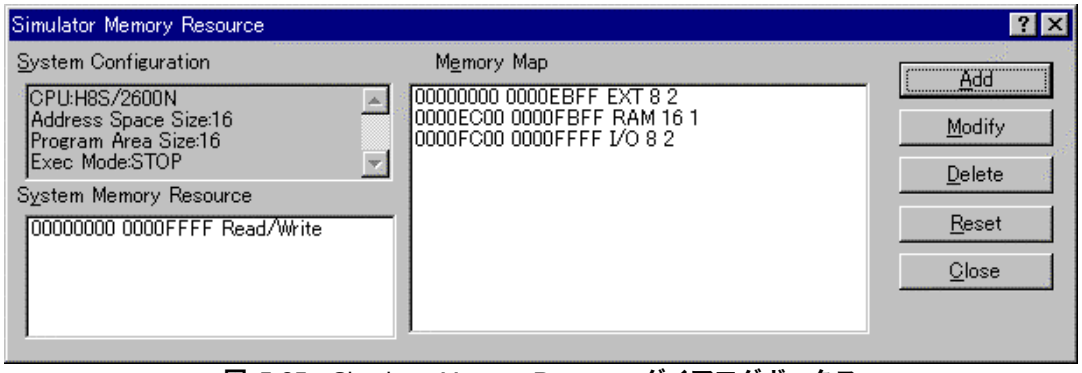


図 5-35 Simulator Memory Resource **ダイアログボックス**

本ダイアログボックスは、対象 CPU に関する情報とメモリマップの設定および変更を行います。

表示する項目は以下の通りです。

[System Configuration]	シミュレータ・デバッガの対象 CPU、アドレスバス幅、実行モードを表示します。
[System memory resource]	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。
[Memory map]	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。

[System memory resource]は次の各ボタンにより設定・変更・削除することができます。

[Add]	[System memory resource]の項目を設定します。クリックすることにより、System Memory Resource Modify ダイアログボックスが開き、設定することができます。
[Modify]	[System memory resource]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、System Memory Resource Modify ダイアログボックスが開き、変更することができます。
[Delete]	[System memory resource]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、[Memory map]・[System memory resource]は、[Reset]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。また、本ダイアログボックスは、[Close]ボタンをクリックすることにより閉じます。

[System memory resource]は、Hitachi 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。Hitachi H8S,H8/300 Standard Toolchain ダイアログについては、HEW デバッガユーザーズマニュアル 「2.3.1 メモリマップ」を参照ください。

5.27 System Memory Resource Modify ダイアログボックス

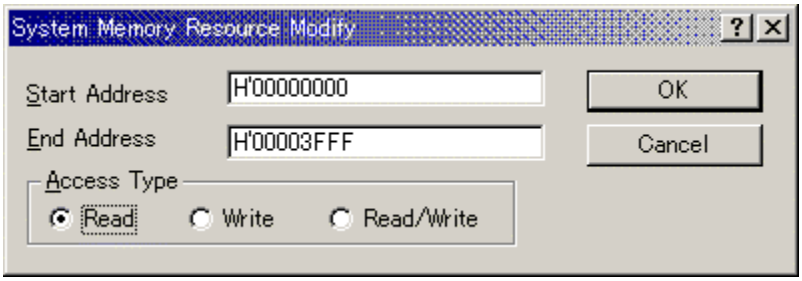


図 5-36 System Memory Resource Modify **ダイアログボックス**

本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。
設定する項目は以下の通りです。

[Start address]	確保するメモリ領域の先頭アドレス	
[End address]	確保するメモリ領域の終了アドレス	
[Access type]	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。[Cancel]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。

5.28 Simulated I/O

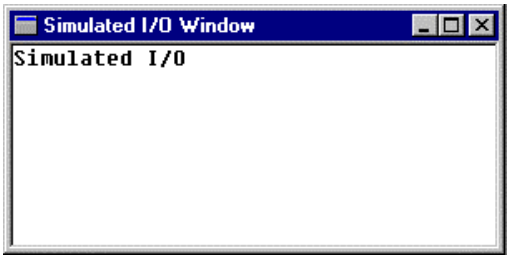


図 5-37 Simulated I/O **ウィンドウ**

本ウィンドウは、デバッグ対象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ためのウィンドウです。
本ウィンドウ上で右クリックすると、以下のポップアップメニューを表示します。

[Copy]	反転表示したテキストを Windows®クリップボードにコピーし、他のアプリケーションに貼り付けられるようにします。
[Paste]	Windows®クリップボードの内容を貼り付けます。
[Erace All]	本ウィンドウの内容をクリアします。

入出力を行うための手順は、**「3.7 標準入出力およびファイル入出力処理」**を参照してください。

5.29 Stack Trace

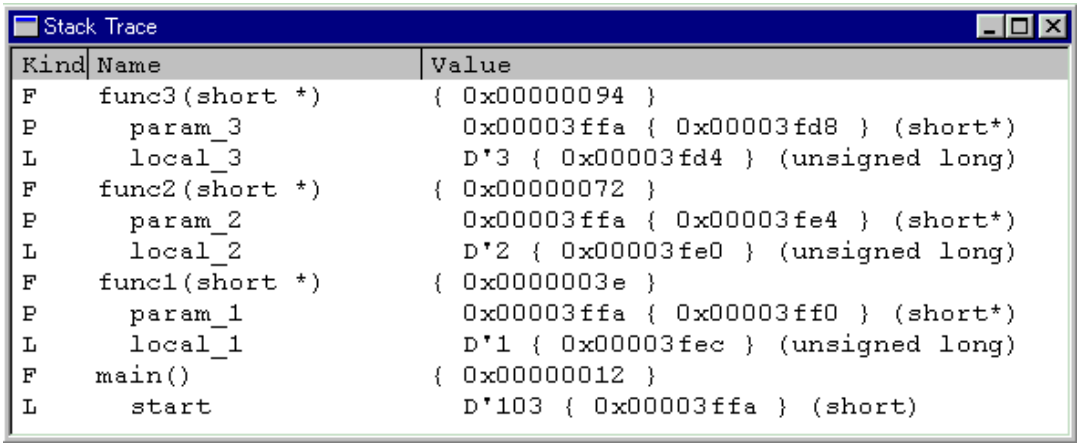


図 5-38 Stack Trace **ウィンドウ**

本ウィンドウは、関数呼び出し履歴を表示します。

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。

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。

5.29.1 Go to Source

選択した関数に該当するソースプログラムを Source ウィンドウ上に表示します。

5.29.2 View Setting...

Stack Trace Setting ダイアログボックスを表示します。Stack Trace ウィンドウの表示形式を設定します。

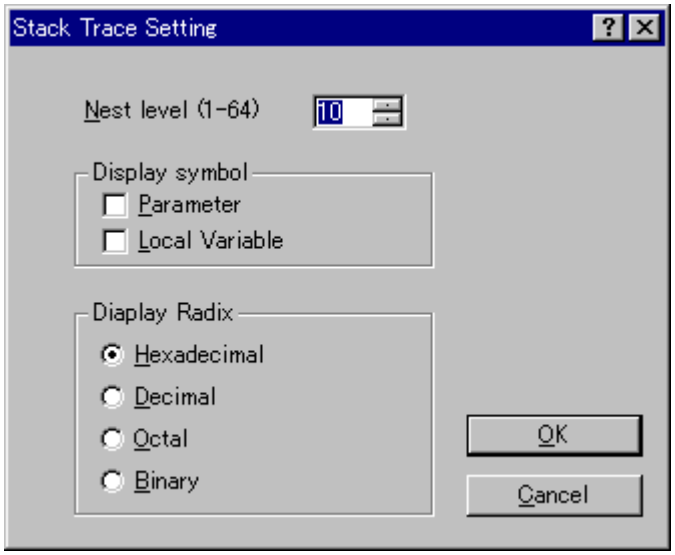


図 5-39 Stack Trace Setting **ダイアログボックス**

[Nest level] Stack Trace 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。
[Display symbol] 関数以外に表示するシンボルを指定します。
[Display Radix] Stack Trace ウィンドウの表示基数を指定します。

5.29.3 Copy

反転表示したテキストを Windows®クリップボードにコピーし、他のアプリケーションに貼り付けられるようにします。

5.30 Profile（List シート）

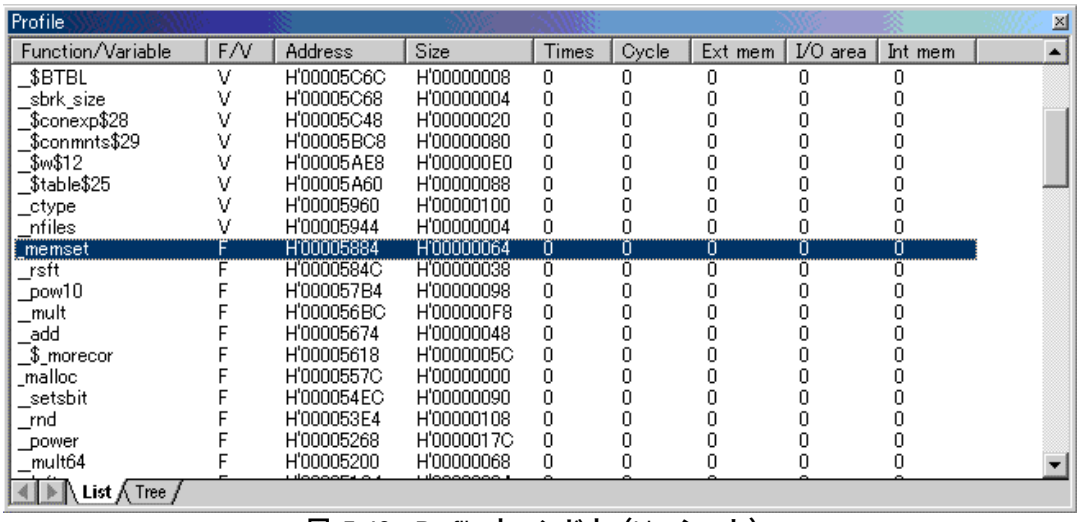


図 5-40　Profile **ウィンドウ**（List **シート**）

本ウィンドウは、関数とグローバル変数のアドレス、サイズ、関数呼出し回数、およびプロファイルデータを表示します。表示するプロファイルデータは以下の通りです。

- Times（関数呼出し回数またはグローバル変数アクセス回数）、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。

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。このポップアップメニューは「5.31 Profile（Tree シート）」を参照してください。

5.31 Profile（Tree シート）

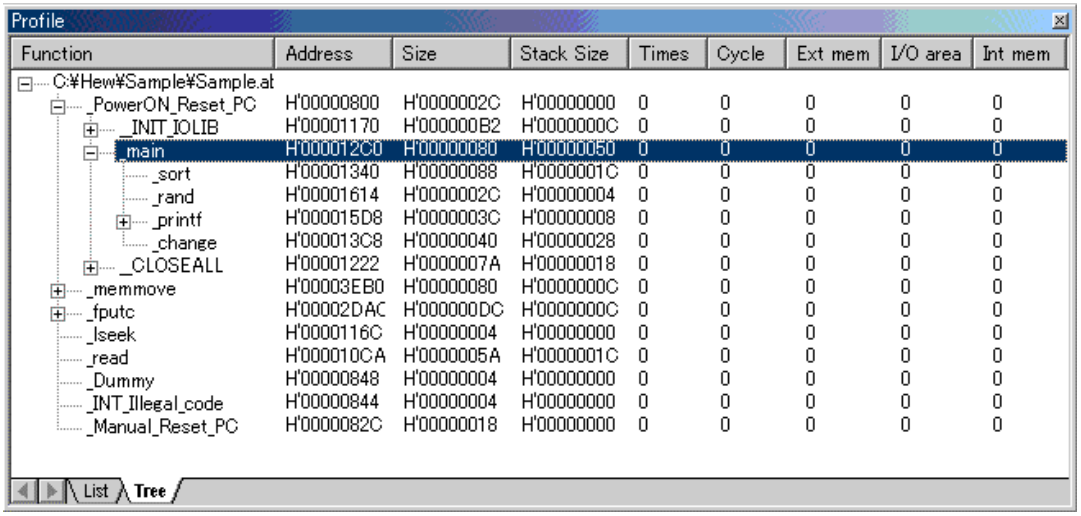


図 5-41 Profile **ウィンドウ**（Tree **シート**）

本ウィンドウは、関数の呼出し関係をツリー構造で表示します。また、各関数のアドレス、サイズ、スタックサイズ、関数呼出し回数、およびプロファイルデータを表示します。スタックサイズ、関数呼出し回数、およびプロファイルデータは、実際の関数を呼出した経路における値を表示します。

表示するプロファイルデータは以下の通りです。

- Times（関数呼出し回数）、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。

【注】 スタックサイズは実際の値とは異なります。関数の呼び出し経路における目安としてください。また、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（拡張子は ".sni"）が無い場合には、スタックサイズを表示しません。スタック使用量情報ファイルについては、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。

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。

5.31.1 View Source

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。

5.31.2 View Profile-Chart

選択している行の関数に着目した Profile-Chart ウィンドウを表示します。

5.31.3 Enable Profiler

プロファイルデータ収集のオン・オフを切り替えます。プロファイルデータ測定が ON 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。

5.31.4 Not trace the function call

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。

Profile 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。

5.31.5 Find...

Function 列の文字列を検索する Find Text 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、[Find Next]ボタンまたは、ENTER キーを入力すると、検索を開始します。

5.31.6 Find Data...

Find Data ダイアログボックスを表示します。

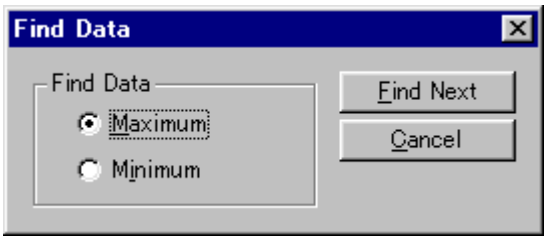


図 5-42 Find Data **ダイアログボックス**

Column コンボボックスで検索カラムを、Find Data グループで検索方向を設定し、[Find Next]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[Find Next]ボタンまたは ENTER キーを入力すると、次に大きいデータ（Minimum の場合は小さいデータ）を検索します。

5.31.7 Clear Data

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。Profile ウィンドウの List シートおよび Profile-Chart ウィンドウのデータもクリアします。

5.31.8 Output Profile Information Files...

Save Profile Information Files 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】 Not trace the function call **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。**

5.31.9 Output Text File...

Save Text of Profile Data 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。

5.31.10 Setting

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）

(1) Show Functions/Variables

Function/Variable 列で、関数およびグローバル変数の両方表示します。

(2) Show Functions

Function/Variable 列で、関数のみを表示します。

(3) Show Variables

Function/Variable 列で、グローバル変数のみを表示します。

(4) Only Executed Functions

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。

(5) Include Data of Child Functions

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。

5.31.11 Properties...

本シミュレータ・デバッガでは使用できません。

5.32 Profile-Chart

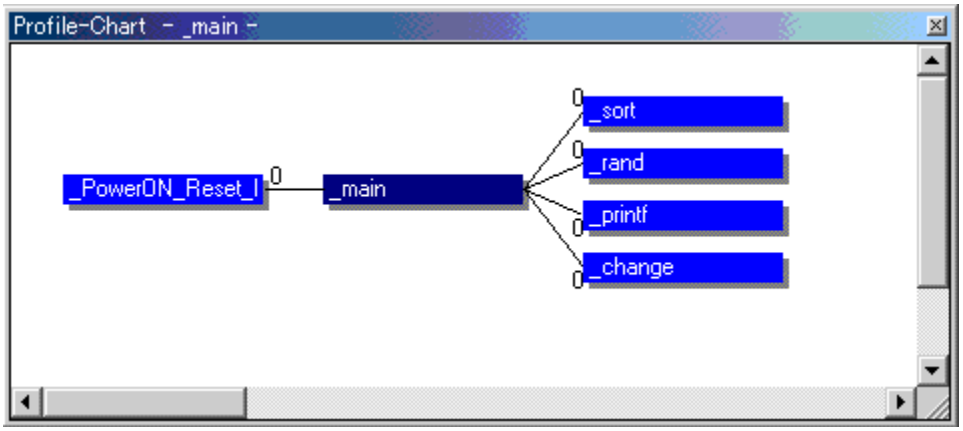


図 5-43 Profile-Chart **ウィンドウ**

ある関数に着目した呼び出し関係を表示します。Profile-Chart ウィンドウは、Profile ウィンドウの List シートまたは Tree シートから開きます。Profile ウィンドウの List シートまたは Tree シートで選択した関数に着目した呼び出し関係を表示します。着目した関数を中心に、左側に呼び出し元関数、右側に呼び出し先関数を表示します。呼び出し元関数および呼び出し先関数横の数値は、呼び出し回数を示します。

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。

5.32.1 View Source

マウスの右ボタンをクリックしたときの位置にある関数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。マウスの右ボタンをクリックしたときの位置が関数ではない場合、このメニューオプションはグレー表示となります。

5.32.2 View Profile-Chart

マウスの右ボタンをクリックしたときの位置にある関数に着目した Profile-Chart ウィンドウを表示します。マウスの右ボタンをクリックしたときの位置が関数ではない場合、このメニューオプションはグレー表示となります。

5.32.3 Enable Profiler

プロファイルデータ収集のオン・オフを切り替えます。プロファイルデータ測定が ON 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。

5.32.4 Clear Data

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。Profile ウィンドウの List シートおよび Tree シートのデータもクリアします。

5.32.5 Multiple View

Profile-Chart ウィンドウを表示する際、既に Profile-Chart ウィンドウが開いているとき、別のウィンドウを開くか同ウィンドウに表示するかを設定します。メニューテキストの左にチェックマークを表示していれば、別のウィンドウを開きます。

5.32.6 Output Profile Information File...

Save Profile Information File 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。

5.32.7 Expands Size

各関数の間隔を広げて表示します。また、'+'キーでも広げて表示することができます。

5.32.8 Reduces Size

各関数の間隔を縮めて表示します。また、'-'キーでも縮めて表示することができます。

5.33 Image View

図 5-44 Image View **ウィンドウ**

メモリの内容を画像で表示します。
表示方法は Image Properties ダイアログボックスで指定します。

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 pixel 情報を Pixel Information ダイアログボックスに表示します。

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。

5.33.1 Auto Refresh

本メニューをチェックすると、ユーザプログラム実行停止時に自動的にウィンドウ内容を更新します。

5.33.2 Refresh Now

ウィンドウ内容を更新します。

5.33.3 Property...

Image Property ダイアログボックスを表示します。表示する画像データのフォーマットを指定します。

5.34 Image Properties ダイアログボックス

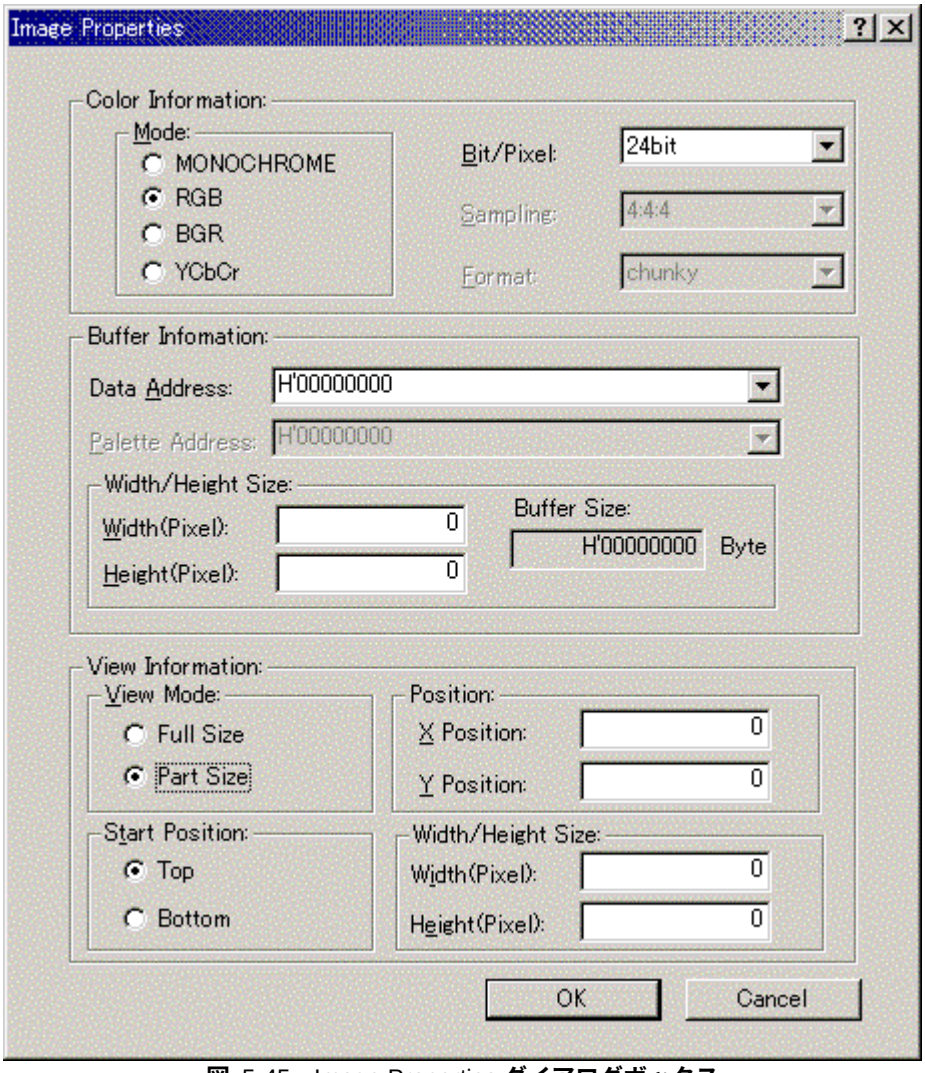


図 5-45 Image Properties **ダイアログボックス**

Image ウィンドウの表示方法を指定します。

[Color Information] 表示する画像のカラー情報を指定します。

【Mode】 フォーマットを指定します。

【MONOCHROME】	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。

【Bit/Pixel】 選択した【Mode】によって、Bit/Pixel を指定します。(RGB/BGR 選択時有効)

【Sampling】 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)

【Format】 Chunky(点順次)/planar(面順次)を指定します。(YCbCr 選択時有効)

[Buffer Information] 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。

【Data Address】 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)

【Palette Address】 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)

【Width/Height Size】 画像の幅と高さを指定します。

【Width(Pixel)】	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)
【Height(Pixel)】	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)
【Buffer Size】	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)

[View Information] 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

【View Mode】 画像の全体表示/部分表示を指定します。

【Full Size】	画像を全体表示します。
【Part Size】	画像を部分表示します。

【Start Position】

【Top】	左上からデータを表示します。
【Bottom】	左下からデータを表示します。

【Position】 部分表示する画像の開始位置を指定します。(【Part Size】選択時有効)

【X Position】	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 Position】	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)

【Width/Height Size】 部分表示する画像の幅と高さを指定します。

【Width(Pixel)】	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)
【Height(Pixel)】	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)

5.35 Pixel Information ダイアログボックス

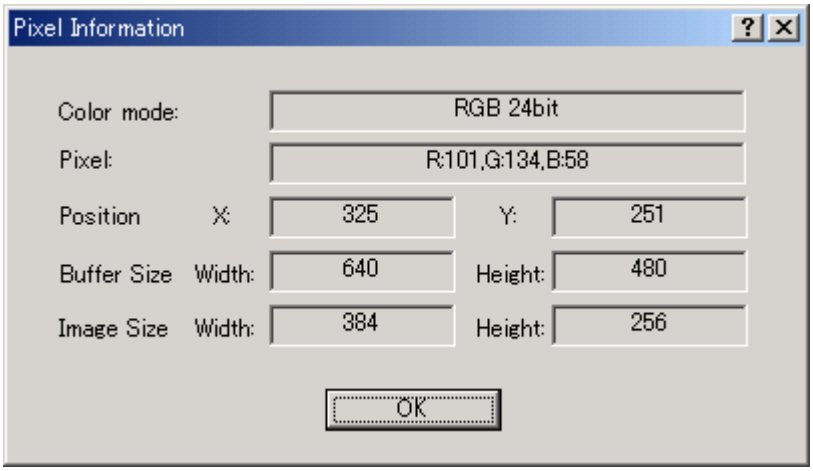


図 5-46 Pixel Information **ダイアログボックス**

カーソル位置の Pixel 情報を表示します。

【Color Mode】 画像のフォーマットを表示します。
【Pixel】 カーソル位置のカラー情報を表示します。（10 進表示）
【Position】 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。
【Buffer Size】 バッファサイズを表示します。（10 進表示）
　【Width】 バッファの幅を表示します。
　【Height】 バッファの高さを表示します。
【Image Size】 表示の幅と高さを表示します。（10 進表示）
　【Width】 幅を表示します。
　【Height】 高さを表示します。

5.36 Waveform

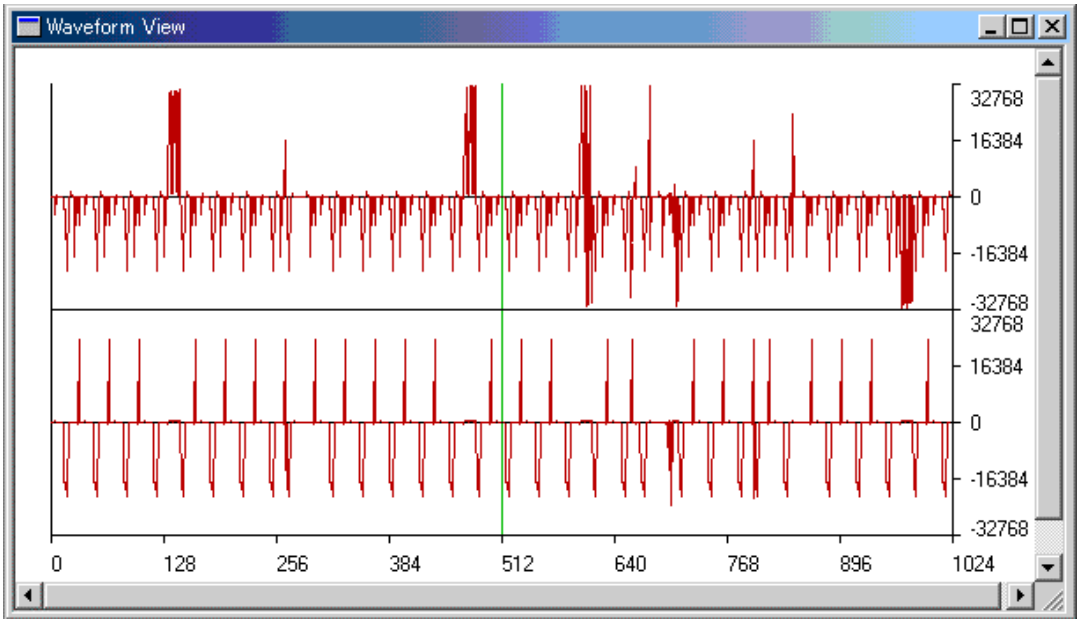


図 5-47 Waveform **ウィンドウ**

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。

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。

5.36.1 Auto Refresh

命令実行が停止したときに表示を更新します。

5.36.2 Refresh Now

表示を更新します。

5.36.3 Zoom In

横軸を拡大して表示します。

5.36.4 Zoom Out

横軸を縮小して表示します。

5.36.5 Reset Zoom

最初のサイズに戻して表示します。

5.36.6 Zoom Magnification

サブメニューでズーム倍率を２、４、８倍から選択します。

5.36.7 Scale

サブメニューで X 軸のサイズを１２８、２５６、５１２Pixel から選択します。

5.36.8 Clear Cursor

カーソルを非表示にします。

5.36.9 Sample Information…

Sample Information ダイアログボックスを表示します。サンプリング情報を表示します。

5.36.10 Property…

Waveform Property ダイアログボックスを表示します。波形データ形式を指定します。

5.37 Waveform Properties ダイアログボックス

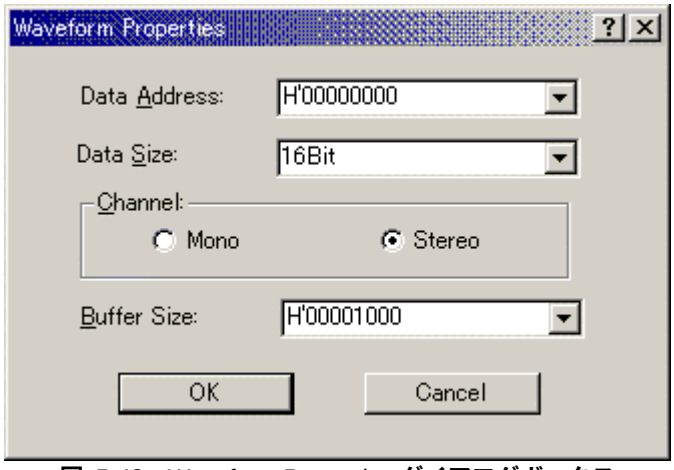


図 5-48 Waveform Properties **ダイアログボックス**

波形形式を指定します。下記項目を指定できます。

[Data Address]	データのメモリ開始アドレスを指定します。（16 進表示）
[Data Size]	8Bit / 16Bit を指定します。
[Channel]	Mono/Stereo　を指定します。
[Buffer Size]	データのバッファサイズを指定します。（16 進表示）

5.38 Sample Information ダイアログボックス

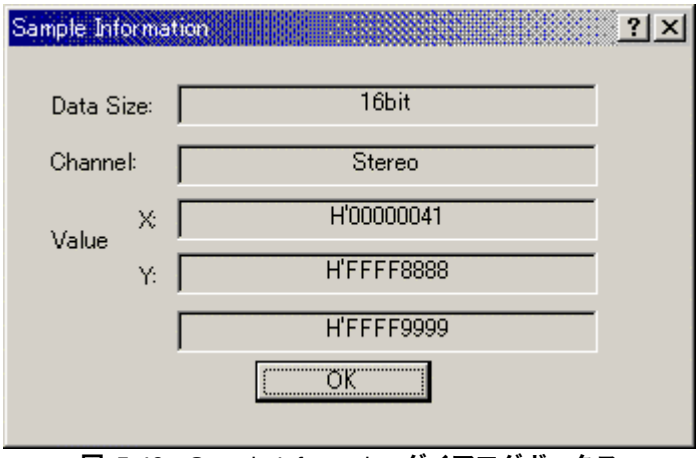


図 5-49 Sample Information **ダイアログボックス**

Waveform View 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。

[Data Size]	8bit / 16bit を表示します。	
[Channel]	データの Channel を表示します。	
[Value]	[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。 (Stereo 選択時は上下 2 つの Y 座標を表示します)

5.39 Coverage

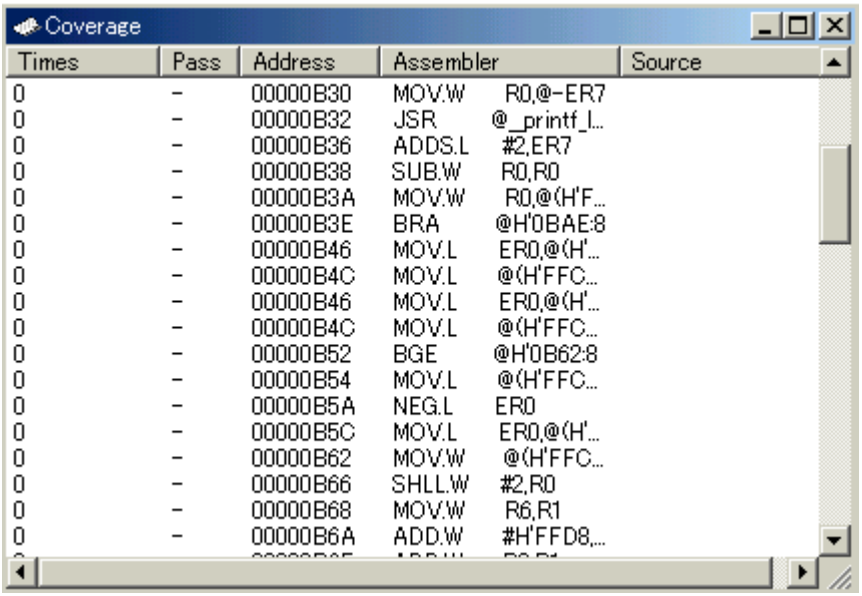


図 5-50 Coverage **ウィンドウ**

C/C++およびアセンブラレベルでの命令実行情報を表示します。なお、命令実行情報の取得条件は、Open Coverage ダイアログボックスまたは Coverage Range ダイアログボックスで設定します。また Coverage ウィンドウをクローズすると取得した命令実行情報と取得条件の設定をクリアします。

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

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。

5.39.1 View Source

Coverage ウィンドウ上のカーソル位置のアドレスに対応する Source ウィンドウを表示します。

5.39.2 Go to Address...

Coverage ウィンドウの表示アドレスを変更します。

5.39.3 Set Range...

Coverage Range ダイアログボックスを表示します。命令実行情報の取得条件を設定します。

5.39.4 Enable Coverage

命令実行情報取得を有効にするか無効にするかを設定します。

5.39.5 Clear Data...

取得した命令実行情報をクリアします。

5.39.6 Save Data...

カバレジ情報をファイルに保存するための Save Data ダイアログボックスを表示します。

5.39.7 Load Data...

カバレジ情報をファイルからロードするための Load Data ダイアログボックスを表示します。

5.39.8 Refresh

最新の命令実行情報を表示します。

5.39.9 Lock Refresh

[Assembler]および[Source]の内容を更新する/しないを指定します。

5.40 Open Coverage ダイアログボックス

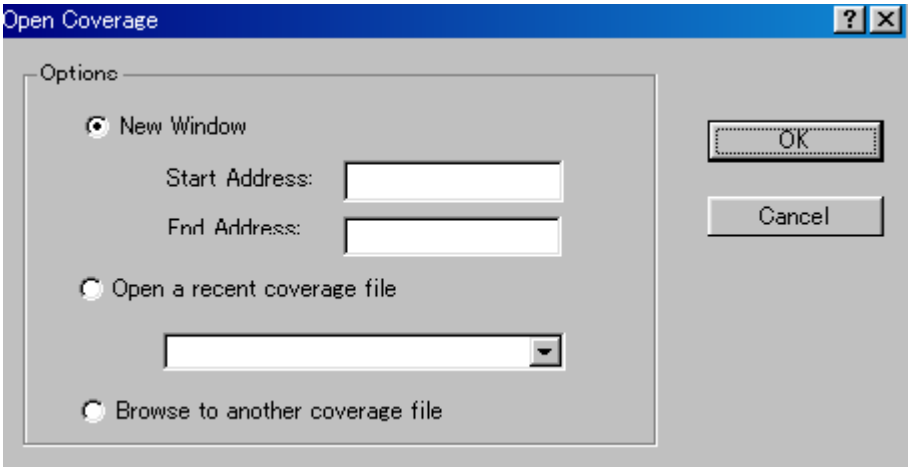


図 5-51 Open Coverage **ダイアログボックス**

カバレジアイコンをクリックすると、Open Coverage ダイアログボックスが開きます。

Open Coverage ダイアログボックスでは、[New Window]、[Open a recent coverage file]、[Browse to another coverage file]のいずれかを選択します。

[New Window]では、開始アドレスおよび終了アドレスを指定します。指定範囲についてカバレジ情報を表示します。指定項目は以下のとおりです。

[Start Address]	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）
[End Address]	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）

[Open a recent coverage file]では、最近保存されたファイルを 4 個まで表示します。

[Browse to another coverage file]では、カバレジ情報ファイルを選択するファイルオープンダイアログボックスを表示します。

5.41 Go To Address ダイアログボックス

図 5-52 Go To Address **ダイアログボックス**

Coverage ウィンドウの表示アドレスを変更します。

5.42 Coverage Range ダイアログボックス

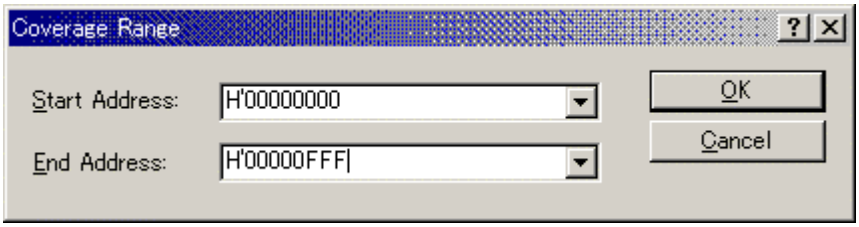


図 5-53 Coverage Range **ダイアログボックス**

命令実行情報の取得条件の指定します。下記項目を指定できます。

[Start Address]	先頭アドレス（接頭辞省略時は 16 進で入力）
[End Address]	終了アドレス（接頭辞省略時は 16 進で入力）

5.43 Save Data ダイアログボックス

図 5-54 Save Data **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。プレースホルダーまたはブラウズ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"および".TXT"以外を入力するとエラーメッセージを出力します。

5.44 Load Data ダイアログボックス

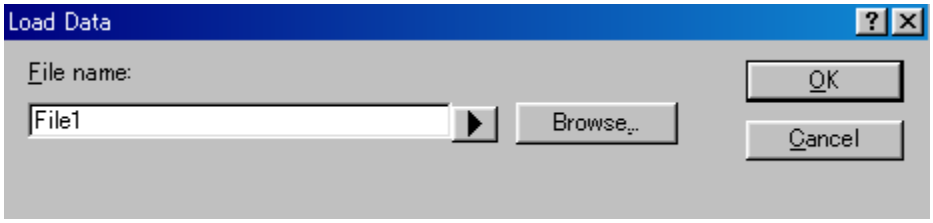


図 5-55 Load Data **ダイアログボックス**

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。プレースホルダーまたはブラウズ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。

5.45 Confirmation Request ダイアログボックス

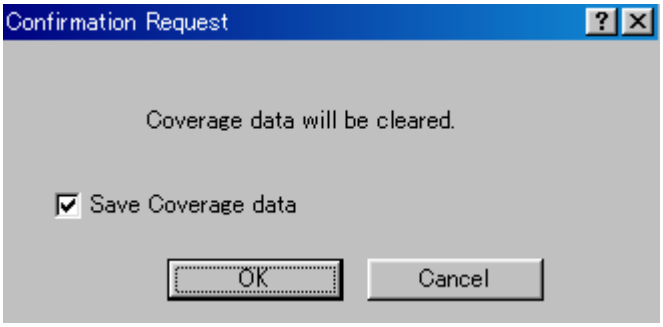


図 5-56 Confirmation Request ダイアログボックス

“Clear Data”、”Set Range...”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。

5.46 Save Coverage Data ダイアログボックス

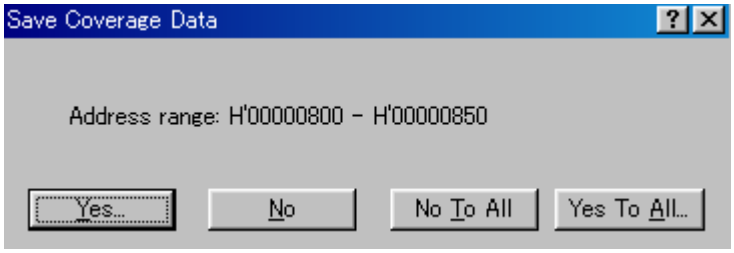


図 5-57 Save Coverage Data ダイアログボックス

[File->Save Session]メニューオプションをクリックすると、Save Coverage Data ダイログボックスが表示され、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。

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 Save Coverage Data ダイアログボックスが開きます。

”No To All”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。

“Yes To All”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。

6. コマンドライン

コマンド一覧を**表** 6-1に示します。

表 6-1 **コマンド一覧**

項番	コマンド名	短縮形	説　明
1	!	-	**コメント**
2	ANALYSIS	AN	**性能分析機能の有効化／無効化**
3	ANALYSIS_RANGE	AR	**性能評価関数の設定、表示**
4	ANALYSIS_RANGE_DELETE	AD	**性能分析範囲の削除**
5	ASSEMBLE	AS	**アセンブルの実行**
6	ASSERT	-	**コンディションのチェック**
7	BREAKPOINT	BP	**実行命令位置によるブレークポイントの設定**
8	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**
9	BREAK_CLEAR	BC	**ブレークポイントの削除**
10	BREAK_CYCLE	BCY	**サイクルによるブレーク条件の設定**
11	BREAK_DATA	BD	**メモリのデータ値によるブレーク条件の設定**
12	BREAK_DISPLAY	BI	**ブレークポイント一覧の表示**
13	BREAK_ENABLE	BE	**ブレークポイントの有効/無効の切換え**
14	BREAK_REGISTER	BR	**レジスタのデータ値によるブレーク条件の設定**
15	BREAK_SEQUENCE	BS	**実行順序を指定したブレークポイントの設定**
16	CHANGE_CONFIGURATION	CC	**コンフィギュレーションの設定**
17	CHANGE_PROJECT	CP	**プロジェクトの設定**
18	COVERAGE	CV	**カバレジ測定の有効化／無効化**
19	COVERAGE_DISPLAY	CVD	**カバレジ情報の表示**
20	COVERAGE_LOAD	CVL	**カバレジ情報のロード**
21	COVERAGE_RANGE	CVR	**カバレジ範囲の設定**
22	COVERAGE_SAVE	CVS	**カバレジ情報のセーブ**
23	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**
24	DISASSEMBLE	DA	**逆アセンブル表示**
25	ERASE	ER	Command Line **ウィンドウの内容のクリア**
26	EVALUATE	EV	**式の計算**
27	FILE_LOAD	FL	**オブジェクト(プログラム)ファイルのロード**
28	FILE_SAVE	FS	**メモリ内容のファイルセーブ**
29	FILE_VERIFY	FV	**ファイル内容とメモリ内容の比較**
30	GO	GO	**ユーザプログラムの実行**
31	GO_RESET	GR	**リセットベクタからのユーザプログラムの実行**
32	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**
33	HALT	HA	**ユーザプログラムの停止**

項番	コマンド名	短縮形	説　明
34	INITIALIZE	IN	デバッグプラットフォームの初期化
35	LOG	LO	ロギングファイルの操作
36	MAP_DISPLAY	MA	メモリマッピング情報の表示
37	MAP_SET	MS	メモリリソースの設定
38	MEMORY_DISPLAY	MD	メモリ内容の表示
39	MEMORY_EDIT	ME	メモリ内容の変更
40	MEMORY_FILL	MF	指定データによるメモリ内容の一括変更
41	MEMORY_MOVE	MV	メモリブロックの移動
42	MEMORY_TEST	MT	メモリブロックのテスト
43	OPEN_WORKSPACE	OW	ワークスペースのオープン
44	PROFILE	PR	プロファイルの有効化／無効化
45	PROFILE_DISPLAY	PD	プロファイル情報の表示
46	PROFILE_SAVE	PS	プロファイル情報のファイル出力
47	QUIT	QU	HEW の終了
48	RADIX	RA	入力ラディックス(基数)の設定
49	REGISTER_DISPLAY	RD	CPU レジスタ値の表示
50	REGISTER_SET	RS	CPU レジスタ値の設定
51	RESET	RE	CPU のリセット
52	RESPONSE	RP	ウィンドウリフレッシュ間隔の設定
53	SLEEP	-	コマンド実行の遅延
54	STEP	ST	ステップ実行(命令単位またはソース行単位)
55	STEP_MODE	SM	ステップモードの設定
56	STEP_OUT	SP	PC 位置の関数を終了するまでのステップ実行
57	STEP_OVER	SO	ステップオーバー実行
58	STEP_RATE	SR	ステップ実行速度の設定、表示
59	SUBMIT	SU	コマンドファイルの実行
60	SYMBOL_ADD	SA	シンボルの設定
61	SYMBOL_CLEAR	SC	シンボルの削除
62	SYMBOL_LOAD	SL	シンボル情報ファイルのロード
63	SYMBOL_SAVE	SS	シンボル情報のファイルセーブ
64	SYMBOL_VIEW	SV	シンボルの表示
65	TCL	-	TCL の有効化／無効化
66	TRACE	TR	トレース情報の表示
67	TRACE_ACQUISITION	TA	トレース情報取得の有効/無効の切換え
68	TRACE_SAVE	TV	トレース情報をファイルへ出力
69	TRACE_STATISTIC	TST	統計情報解析の実行

以下に各コマンドのシンタックスを示します。

6.1 ! (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　Command Line 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

6.2 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

6.3 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

6.4 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

6.5 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

6.6 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

6.7 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

6.8 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

6.9 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

6.10 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

6.11 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>] [<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
<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

6.12　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

6.13 BREAK_ENABLE

短縮形: BE

説明:

ブレークポイントの有効/無効を切換えます。

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

6.14 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>] [<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。）
	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
<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

6.15 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

6.16 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 <config name>

パラメータ	型	説明
<config name>	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

6.17 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 name>

パラメータ	型	説明
<project name>	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

6.18 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 [<state>]

パラメータ	型	説明
なし		カバレジの状態を表示します
<state>	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 enable	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

6.19 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

6.20 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。

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

6.21 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>]

パラメータ	型	説明
なし		カバレジ測定の範囲を表示します
<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

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

6.22 COVERAGE_SAVE

短縮形: CVS

説明:

カバレジ情報を.COV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

6.23 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

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

6.24 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

6.25 ERASE

短縮形: ER

説明:

Command Line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		Command Line ウィンドウをクリアします

例:

ER　　Command Line ウィンドウをクリアします

6.26 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

6.27 FILE_LOAD

短縮形: FL

説明:

指定したオフセットで、メモリに指定したファイルをロードします。 現在のシンボルはクリアし、新しいシンボルを定義します。 オフセットを指定するとシンボルに加算します。ファイルの拡張子はデフォルトとして".MOT"を設定します。

シンタックス

fl [<format>] <filename> [<offset>] [<state>]

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）
<state>	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:

FILE_LOAD 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
FL ANOTHER.MOT H’200	モトローラ S レコードファイル "another.mot" をオフセット H'200 バイトからロードします

6.28 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。ファイルの拡張子のデフォルトは、".MOT"です。シンボルは自動的にセーブしません。

シンタックス
fs [<format>] <filename> <start> <end>

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**

例:

FILE_SAVE TESTFILE 0 H’2013	アドレス範囲 0 - H'2013 をモトローラ S レコードファイル "TESTFILE.MOT" として保存します
FS 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

6.29 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。 ファイルデータはモトローラ S レコードフォーマットです。ファイルの拡張子はデフォルトとして".MOT"を設定します。

シンタックス
fv <filename> [<offset>]

パラメータ	型	説明
<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**）**

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
FV ANOTHER 200	メモリに対してモトローラ S レコードファイル "ANOTHER.MOT" にオフセット H'200 バイトを加えたアドレスからベリファイします

6.30 GO

短縮形: GO

説明:

ユーザプログラムを実行します。ユーザプログラムを実行しているとき、Performance Analysis ウィンドウを更新します。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 [<state>] [<address>]

パラメータ	型	説明
<state>	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)

6.31 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、Performance Analysisウィンドウを更新します。

シンタックス
gr [<state>]

パラメータ	型	説明
<state>	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)

6.32 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 [<state>] <address>...

パラメータ	型	説明
<state>	**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**	**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

6.33 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

6.34 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

6.35 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

6.36 MAP_DISPLAY

短縮形: MA

説明:

現在のメモリマップ構成を表示します。

シンタックス

ma

パラメータ	型	説明
なし		現在のメモリマップ構成を表示します

例:

MA　　現在のメモリマップ構成を表示します

6.37 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

6.38 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

6.39 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>] [<state>]

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**
<state>	**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)

6.40 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>] [<state>]

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**
<state>	**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
MF H’5000 H’7FFF H’21	H'5000 から H'7FFF までデータ H'21 に変更します

6.41 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> <dest> [<state>]

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）**
<dest>	**数値**	**移動先開始アドレス**
<state>	**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_MOVE H’1000 H’1FFF H’2000	領域 H'1000 - H'1FFF を H'2000 へ移動します
MV H’FB80 H’FF7F H’3000	領域 H'FB80 - H'FF7F を H'3000 へ移動します

6.42 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
MT H’4000 H’5000	H'4000 から H'5000 までテストします

6.43 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

6.44 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、切り替え、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

6.45 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

6.46 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

6.47 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

6.48 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 [<mode>]

パラメータ	型	説明
なし		**現在の基数を表示します**
<mode>	**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

6.49 REGISTER_DISPLAY

短縮形: RD

説明:

CPUレジスタ値を表示します。

シンタックス

rd

パラメータ	型	説明
なし		**全レジスタの内容を表示します**

例:

RD 全レジスタの内容を表示します

6.50 REGISTER_SET

短縮形: RS

説明:

CPU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

6.51 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

6.52 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**	1~65535（**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

6.53 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

6.54 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 [<mode>] [<count>]

パラメータ	型	説明
<mode>	**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

6.55 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**
	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

Source 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

Disassembly 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

6.56 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

6.57 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 [<mode>] [<count>]

パラメータ	型	説明
<mode>	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

6.58 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。 6レートは最大限に速くステップします。 値が0の場合は最も遅いステップです。

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

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

6.59 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

6.60 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 <symbol> <value>

パラメータ	型	説明
<symbol>	文字列	シンボル名
<value>	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

6.61 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 [<symbol>]

パラメータ	型	説明
なし		**すべてのシンボルを削除します**
<symbol>	**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

6.62 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

SL MY_CODE.SYM ファイル MY_CODE.SYM をロードします

6.63 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
SS MY_CODE.SYM	MY_CODE.SYM にシンボルテーブルを保存します

6.64 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

6.65 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

6.66 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

6.67 TRACE_ACQUISITION

短縮形: TA

説明:

トレース情報取得の有効/無効を切換えます。

シンタックス

ta <mode>

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**

例:

TRACE_ACQUISITION E	トレース情報の取得を有効にします
TA D	トレース情報の取得を無効にします

6.68 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>

パラメータ	型	説明
<filename>	**文字列**	**ファイル名**

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

6.69 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

7. メッセージ一覧

7.1 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 7-1に示します。

表 7-1 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	Trace Acquisition ダイアログボックスの Trace buffer full handling で Break 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。

7.2 エラーメッセージ

シミュレータ・デバッガはデバッグ対象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 7-2に示します。

表 7-2 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デバッグ対象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デバッグ対象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。

メッセージ	内容・対策
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デバッグ対象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デバッグ対象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デバッグ対象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。

付録 A　GUI コマンド一覧

メニュー	メニューオプション	ショートカットキー	ツールバーボタン
View	Workspace	Alt+K	
	Output	Alt+U	
	Breakpoints	Shift+Ctrl+B	
	Coverage...	Shift+Ctrl+O	
	Command Line	Ctrl+L	
	Disassembly	Ctrl+D	
	IO	Ctrl+I	
	Image...	Shift+Ctrl+G	
	Labels	Shift+Ctrl+A	
	Locals	Shift+Ctrl+D	
	Memory...	Ctrl+M	
	Performance Analysis	Shift+Ctrl+P	
	Profile	Shift+Ctrl+F	
	Registers	Ctrl+R	
	Status	Ctrl+U	
	Trace	Ctrl+T	
	Watch	Ctrl+W	
	Waveform...	Shift+Ctrl+V	
	Localized Dump...	Shift+Ctrl+D	
	Simulated I/O	Shift+Ctrl+I	
	Trigger	Shift+Ctrl+R	
	Stack Trace	Ctrl+K	

メニュー	メニューオプション	ショートカットキー	ツールバーボタン
Options	Debug Settings...		
	Radix ➢		
	Hex		
	Decimal		
	Oct		
	Bin		
	Simulator ➢		
	System...		
	Memory Resource...		
Debug	Reset CPU		
	Go	F5	
	Reset Go	Shift+F5	
	Go to Cursor		
	Set PC To Cursor		
	Run...		
	Step In	F11	
	Step Over	F10	
	Step Out	Shift+F11	
	Step...		
	Step Mode ➢		
	Auto		
	Assembly		
	Source		
	Halt Program	Esc	
	Initialize		
	Disconnect		
	Download Modules		
	Unload Modules		

メニュー	メニューオプション	ショートカットキー	ツールバーボタン
Memory	Search...		
	Copy...		
	Compare...		
	Fill...		
	Refresh		
	Configure Overlay...		

H8S,H8/300 シリーズ シミュレータ・デバッガ
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

ADJ-702-355A