

# User's Manual

# RX ファミリ シミュレータ/デバッガ V.1.01

# ユーザーズマニュアル

ルネサスマイクロコンピュータ開発環境システム



Rev.1.00 2010.04

#### 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。

# はじめに

# このマニュアルについて

このマニュアルは、High-performance Embedded Workshop (HEW)に付属しているシミュレータ・デバッガのデバッグ機能について説明します。

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

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

Microsoft, MS-DOS, Windows, Windows NT は米国 Microsoft 社の米国およびその他の国における登録商標です。 Visual SourceSafe は Microsoft 社の米国およびその他の国における商標です。 IBM は International Business Machines Corporation の登録商標です。 その他、記載されている製品名は各社の商標または登録商標です。

# このマニュアルの記号

このマニュアルで使われている記号の意味を説明します。

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

表 1: 記号一覧

# 製品の内容及び本書についてのお問い合わせ先

ルネサス エレクトロニクス株式会社 コンタクトセンタ <u>csc@renesas.com</u>

ホームページ <u>http://japan.renesas.com/tools</u>

# 目次

1.	はじめに	1			
2.	シミュレータ・デバッガの機能	3			
2.1	特長				
2.2	デバッグ対象プログラム	3			
2.3	シミュレーション範囲	4			
2.4	メモリ管理	4			
2.5	ァーテロー 命今実行リヤット処理				
2.6	例外如理				
2.7	エンディアン	5			
2.7	ーテントンフ	5			
	2.7.2 外部メモリ領域のエンディアン	5			
2.8	周辺機能シミュレーション	5			
	2.8.1 タイマ	5			
	2.8.2 シリアルコミュニケーションインタフェース	6			
	2.8.3 割り込みコントローラ	9			
	2.8.4 クロック 2.8.5 周辺機能を使用する	10 10			
2.9		11			
2.10	標準入出力およびファイル入出力処理	11			
2.11	ブレーク条件	11			
2.12	浮動小数点データ	13			
2.13	関数呼び出し履歴の表示	14			
2.14	パフォーマンス測定	14			
	2.14.1 プロファイラ	14			
	2.14.2 パフォーマンス解析	14			
2.15	擬似割込み	14			
2.16	カバレジ	15			
3.	デバッグ	17			
3.1	ワークスペースの作成	17			
	3.1.1 ターゲットの選択	17			
2.2	3.1.2 シミュレータ用ワークスペースの設定	19			
3.2	シミュレータ・テハッカの起動	20			
3.3	シミュレーダ・テハッカの設定を変更する	20			
	3.3.1 CPU のエンティアンと動作周波数を設定する	20 21			
	3.3.2 メモリマップおよびメモリリソースの設定を変更する	21			
	3.3.4 メモリマップ設定ダイアログボックス	23			
	3.3.5 メモリリソース設定ダイアログボックス	24			
3.4	周辺機能シミュレーションを設定する	24			
	3.4.1 周辺機能シミュレーションモジュールを登録する	25			
	<ul> <li>3.4.2 同辺 ( 限 ) アレイを 2 史 9 る</li></ul>	26 26			
	3.4.4 制御レジスタのメモリリソース	28			
	3.4.5 接続されている周辺機能を確認する	28			

	3.4.6 仮想	見ポートへのファイル入出力を行う	
3.5	メモリを操作	₣する	
	3.5.1 ウィ	インドウの表示内容を定期的に更新する	
	3.5.2 I/O s	領域の内容を表示、変更する	
3.6	シミュレータ	?・デバッガのブレークポイントを使用する	
	361 <b>ブ</b> レ	ノークポイントを一覧表示する	31
	3.62 JL	, ノボイントを設定する	
	3.63 <b>ブ</b> レ	/ ノボイントで設定りで	
	364 ブレ	ノークポイントを有効にする	
	365 ブレ	/ ノボイントを無効にする	38
	3.6.5 ブレ 3.6.6 ブレ	/ ノボイントを削除する	38
	3.67 ブレ	/ ノボイントをすべて削除する	38
	368 ブレ	/ ノボイントのソース行を表示する	38
	3.6.0 ΣP	ジークボークトのジークトロースのマクロームの「アイルを閉じる」 ドカファイルを閉じる	38
	3610 入出	37577777777777777777777777777777777777	38
3.7	トレース情報	375977 - 77 C 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	38
017	271 6		20
	3.7.1 FV	/一入り1 ノドリを用く	
	3.7.2 FV	/一人    報 収 侍 赤 叶 ど 政 と り る	
	3.7.3 FV	/一人1 ヘノトを設足 9 る	40
	3.7.4 FV	/一人    報 2 以 侍 9 つ	40
	3.7.5 FV	/一人 [[牧を快系 9 る	
	3.7.0 FV	/一人    報の ノイ ルタリノク	
		ノース情報をプリアする	
	3.7.8 FV	/一人情報をノアイルに休住する	
	3.7.9 <b>2</b> -	- ヘノアイルを衣小りる	
	3.7.10 21	ムスランフの衣小を切り目える	
29	3.7.11 戻政	X夫门腹症でな小する	
5.0		/ 同報を兄る	
	3.8.1 スタ	アック情報ファイル	
	3.8.2 <b>ス</b> タ	イック情報ノアイルのロード	
		Jノアイルを有効にする	
	3.8.4 測正	5.力法を指定 9 る	
	3.8.5 1-	- ワノロクフムを美行し結果を帷認 9 る	
	3.8.6 List		
	3.8./ Tree	3 ン - ト	
		コノアイルナヤートリイントリ	
	3.8.9 衣亦	『テーツの裡頬のよび用述	
	3.8.10 ノL 2.9.11 注音	J ノ パ 1 ル 恒 牧 ノ パ 1 ル を 1 F 成 9 る	
2.0	3.8.11 注思	まます。 パンティンティンティンティンティンティンティンティンティンティンティンティンティン	
3.9	ハノオーマノ	/ 人で 脾 们 9 つ	
	3.9.1 パフ	フォーマンス解析ウィンドウを開く	53
	3.9.2 評価	山関数を設定する	54
	3.9.3 デー	- 夕 収集を開始する	54
	3.9.4 デー	- タをリセットする	54
	3.9.5 評値	山関数を削除する	54
	3.9.6 J	べての評価関数を削除する	54
	3.9.7 現在	Eの表示内容をセーフする	
3.10	コードカバレ	/ンを測正する	55
	3.10.1 カハ	ヾレジウィンドウを開く	55
	3.10.2 カハ	ドレジ情報をすべて取得する	56
	3.10.3 カハ	ドレジ情報をすべてクリアする	56
	3.10.4 ソー	- スファイルを表示する	56
	3.10.5 新し	ノいカバレジ測定範囲を設定する	57
	3.10.6 カハ	ドレジ測定範囲を変更する	57
	3.10.7 カハ	<b>ドレジ測定範囲を削除する</b>	

	3.10.8	カバレジ情報を取得する	
	3.10.9	カバレジ情報をクリアする	58
	3.10.10	カバレジ情報をファイルに保存する	
	3.10.11	カバレジ情報をファイルからロードする	
	3.10.12	最新の情報に更新する	
	3.10.13	Confirmation Request ダイアログボックス	
	3 10 14		59
	3 10 15	エディタウィンドウへのカバレジ結里表示	60
	3 10 16	エティッシュティット シャックパンシ 加水 気がいいい しょう アンブリウィンドウへのカバレジ 往里 表示	
3 1 1	5.10.10 手動で歩	とりビジジジッキンキン、のパイレン。山木农小	
5.11			
	3.11.1	トリカウィンドウ	
	3.11.2	GUI I/O ウィンドウ	63
3.12	標準入出	出力およびファイル入出力を行う	65
	3.12.1	I/O シミュレーションウィンドウを開く	65
	3.12.2	入出力機能	
3.13	仮想入出	出力パネルを作成する	
	2 1 2 1		(7
	3.13.1	UUI ハ山ハフィントンで用て	
	3.13.2	ハウノで1FR29る	
	3.13.3	フペルを作成する	
	3.13.4	LED を作成する	
	3.13.5	テキスト文字列を作成する	72
	3.13.6	アイテムのサイズ/配置を変更する	72
	3.13.7	アイテムをコピーする	73
	3.13.8	アイテムを削除する	73
	3.13.9	グリッド線を表示する	73
	3.13.10	入出力パネル情報を保存する	73
	3.13.11	入出力パネル情報を読み込む	73
4.	ウィンド	<u> </u>	75
	2121		10
5.	コマンド	ライン	77
5.1	コマント	ヾ一覧(機能別)	77
	511	宝行関連	77
	512	メリアビード問連	, , ,
	5.1.2	ノ ノ ノ レ ロ 目 (月)年	י / רר
	5.1.5	レンスン保止街道	
	5.1.4	アモリ保証関連	
	5.1.5	アセノフル/逆アセノフル(浜)係	
	5.1.6	ノレーク設定関連	
	5.1.7	トレース 関連	
	5.1.8	カハレン計測関連	
	5.1.9	バフォーマンス測定関連	
	5.1.10	ウォッチ関連	
	5.1.11	スクリフト/ログファイル関連	79
	5.1.12	マップ関連	79
	5.1.13	シミュレーションの設定関連	80
	5.1.14	標準入出力およびファイル入出力関連	80
	5.1.15	ユーティリティ関連	80
	5.1.16	プロジェクト/ワークスペース関連	80
	5.1.17	テスト支援機能関連	
	5.1.18	リアルタイム OS 対応デバッグ機能関連	
	5.1.19	仮想ポートファイル入出力関連	
5.2	コマント	ドー覧(アルファベット順)	
- · -		、···································	
6.	メッセー	ソー頁	85
6.1	インフォ	ォメーションメッセージ	
6.2	エラーン	メッセージ	86

7.	チュートリアル	87
7.1	デバッグの準備	
	7.1.1 サンプルプログラム	
	7.1.2 <b>サンプルプログラムの</b> 作成	
7.2	デバッグのための設定	
	7.2.1 メモリリソースの確保	
	7.2.2 サンプルプログラムのダウンロード	
	7.2.3 <b>ソースプログラムの</b> 表示	
	7.2.4 PC ブレークポイントの設定	
	7.2.5 <b>プロファイラの</b> 設定	
	7.2.6 I/O シミュレーションの設定	
	7.2.7 トレース情報取得条件の設定	
	7.2.8 スタックポインタ、プログラムカウン	<b>'夕の設定</b> 94
7.3	デバッグ開始	
	7.3.1 <b>プログラムの</b> 実行	
	7.3.2 トレースバッファの使い方	
	7.3.3 トレース検索の実行	
	7.3.4 I/O シミュレーションの確認	
	7.3.5 ブレークポイントの確認	
	7.3.6 変数の参照	
	7.3.7 プログラムのステップ実行	
	7.3.8 プロファイラ情報の確認	

# 1. はじめに

シミュレータ・デバッガは、ルネサスエレクトロニクスのマイクロコンピュータの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され たプログラムを効率よくデバッグすることができます。

シミュレータ・デバッガは、High-performance Embedded Workshop (HEW)とともに動作します。HEWは、ルネサスエレクトロニクス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かれ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。

シミュレータ・デバッガを使用するにあたっては、シミュレータ・デバッガ、およびHEWのヘルプも参照ください。



1.はじめに



# 2. シミュレータ・デバッガの機能

本章では、RX600シリーズ シミュレータ・デバッガの機能について説明します。

# 2.1 特長

本シミュレータ・デバッガには、次のような特長があります。

- (1) ホスト計算機上で動作するので、実機がなくてもプログラムのデバッグを開始することができ、システム全体の 開発期間を短縮できます。
- (2) シミュレーション時にプログラムの命令実行サイクル数、および命令実行時間を計算します。これにより実機が なくても性能評価が行えます。
- (3) 擬似割込み機能、およびI/Oシミュレーション機能を持ち、簡易的なシステムレベルシミュレーションを行えます。
- (4) 下記のような機能を持ち、プログラムのテスト、およびデバッグを効率よく進めることができます。
  - RX600 シリーズの各 CPU に対応
  - ユーザプログラムの実行中に異常が発生した場合、異常を無視して続行するか、または停止するかを制御する機能
  - プロファイルデータ取得、および関数単位のパフォーマンス測定
  - 豊富なブレーク機能
  - メモリマップの設定・編集
  - 関数呼び出し履歴の表示
  - C/C++およびアセンブラソースレベルのカバレジ表示
  - イメージ表示、波形表示による視覚的デバッグ機能
- (5) Windows<sup>®</sup>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。RX600マイコンの各々のメモリマップに対応した環境設定もダイアログボックス上 で行うことができます。
- また、下記のような特徴を持ちます。
- 直観的なユーザインタフェース
- オンラインヘルプ
- 共通した表示と操作性

# 2.2 デバッグ対象プログラム

シミュレータ・デバッガでは、ELF/DWARF2フォーマットのロードモジュールがシンボリックデバッグ可能です。その他のフォーマットのロードモジュールについては、ダウンロードと命令実行はできますが、シンボリックデバッグはできません。詳しくは、「High-performance Embedded Workshop ユーザーズマニュアル」を参照してください。



# 2.3 シミュレーション範囲

- (1) RX600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、RX600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
- (3) シミュレータ・デバッガは、RX600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、RX600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。

項番	項目	備考
1	低消費電力状態	WAIT命令を実行するとシミュ
		レーションを停止します
2	ノンマスカブル割り込み(NMI)	
3	下記命令実行途中の割り込み受付け	命令実行完了で割り込みを受付
	( RMPA, SCMPU, SMOVF, SMOVB, SMOVU, SSTR,	けます
	SUNTIL, SWHILE )	
4	命令終了後に不定となるデータ、レジスタ値	
5	アキュムレータ(ACC)の下位16ビット	

## 2.4 メモリ管理

(1) メモリマップの設定

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

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

メモリマップでエンディアンを指定できるのは外部領域のみです。内蔵 ROM 領域、および内蔵 RAM 領域のエンディ アンはシミュレータ・デバッガ起動時に表示する[シミュレータの設定]ダイアログボックス[CPU の構成]タブで指定する [エンディアン]を適用します。

詳細は、「3.3.3 メモリマップおよびメモリリソースの設定を変更する」を参照してください。

(2) メモリリソースの設定

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

- 開始アドレス
- 終了アドレス
- アクセス種別

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

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

メモリリソース設定の詳細は、「3.3.3 メモリマップおよびメモリリソースの設定を変更する」を参照してください。

## 2.5 命令実行リセット処理

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

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 実行開始アドレスを指定したRunコマンドを実行した イニシャライズまたはプログラムをロードした

## 2.6 例外処理

シミュレータ・デバッガでは、RX600シリーズマイコンの例外発生を検出し、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) 専用レジスタ(高速割り込み時)、またはスタック領域(通常割り込み時)に PCと PSWを退避します。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) PSWの下記ビットをセットします。
   U=0
   I=0
   PM=0
- (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。

# 2.7 エンディアン

2.7.1 CPU のエンディアン

CPUのエンディアンはシミュレータ・デバッガ起動時に表示する[シミュレータの設定]ダイアログボックス[CPUの構成]タブで指定します。

CPUのエンディアンは内蔵 ROM、および内蔵 RAM に適用します。 指定の詳細は「3.3.1 CPUのエンディアンと動作周波数を設定する」を参照ください。

#### 2.7.2 外部メモリ領域のエンディアン

外部メモリ領域のエンディアンは[メモリマップ設定]ダイアログボックスで設定します。 指定の詳細は「3.3.4 メモリマップ設定ダイアログボックス」を参照ください。

# 2.8 周辺機能シミュレーション

#### 2.8.1 タイマ

(1) サポート範囲

RX600 シリーズシミュレータ・デバッガでは2 チャネルの16 ビットタイマにより構成されるコンペアマッチタイマ (CMT)を2 ユニット(ユニット0、ユニット1)、合計4 チャネルをサポートしています。

(2) 制御レジスタ

シミュレータ・デバッガでサポートしているCMTの制御レジスタを表2-1に示します。 表中サポート状況の はサポートしています。 制御レジスタは必ずレジスタサイズでアクセスしてください。



表2-1 シミュレータ・デバッガでサポートする CMT 制御レジスタ

周辺機能モ ジュール名	ユニット	制御レジスタ	サポート状況
CMT	ユニット0	CMSTR0	
		CMCR0	
		CMCNT0	
		CMCOR0	
		CMCR1	
		CMCNT1	
		CMCOR1	
	ユニット1	CMSTR1	
		CMCR2	
		CMCNT2	
		CMCOR2	
		CMCR3	
		CMCNT3	
		CMCOR3	

制御レジスタのアドレスは[周辺モジュールの構成]ダイアログボックスで参照・変更することができます。[周辺モジュールの構成]ダイアログボックスの詳細については、「3.4 周辺機能シミュレーションを設定する」を参照してください。

#### 2.8.2 シリアルコミュニケーションインタフェース

(1) サポート範囲

RX600 シリーズシミュレータ・デバッガでは7 チャネルのシリアルコミュニケーションインタフェース(SCI)をサポートしています。シミュレータ・デバッガでサポートするSCI機能を表 2-2に示します。 表中サポート状況の はサポート、- は未サポートです。

	サポート状況		
シリアル通信方式	調歩同期式/クロック同期		
	スマートカードインタフ	ェース	-
内蔵ボーレートジェネレータの	PCLK クロック		
クロックソース	PCLK/4、PCLK/16、お。	よび PCLK/64 クロック	-
全二重通信			
割り込み要因	送信終了、送信データエ		
調歩同期式モード	データ長	7ビット/8ビット	
	送信ストップビット	1 ビット / 2 ビット	
	パリティ機能	偶数 / 奇数 / なし	
	受信エラー検出機能	パリティエラー、オーバランエラー、 フレーミングエラー	
	ブレーク検出	-	
	クロックソース	内部クロック	
		外部クロック、TMR からの転送レー トクロック	-
クロック同期式モード	データ長	8ビット	
	受信エラーの検出	オーバランエラー	

(2) 制御レジスタ

シミュレータ・デバッガでサポートしているSCIの制御レジスタを表2-2に示します。

表中サポート状況の はサポート、 は「2.8.2 (1) サポート範囲」で説明した機能に関するビットのみサポートしています。

制御レジスタは必ずレジスタサイズでアクセスしてください。



周辺機能モ ジュール名	チャネル	制御レジスタ	サポート状況
SCI	0~6	SMR	
		BRR	
		SCR	
		TDR	
		SSR	
		RDR	
		SCMR	
		SEMR	

表2-3 シミュレータ・デバッガでサポートする SCI 制御レジスタ

制御レジスタのアドレスは[周辺モジュールの構成]ダイアログボックスで参照・変更することができます。[周辺モジュールの構成]ダイアログボックスの詳細については、「3.4 周辺機能シミュレーションを設定する」を参照してください。

#### (3) データ入出力

シミュレータ・デバッガでは端子の一部を仮想ポートとしてメモリ上に割付けています。この仮想ポートを介してデ バッグ対象プログラムやデバッガは端子にアクセスすることができます。

SCIの仮想ポートアドレスを 表 2-4に示します。

#### 表2-4 SCIの仮想ポートアドレス

チャネル	仮想ポート名	アドレス	アクセスサイズ	機能
0	RxD0	H'00088224	16	チャネル0受信データ
	TxD0	H'00088226	16	チャネル0送信データ
1	RxD1	H'00088228	16	チャネル 1 受信データ
	TxD1	H'0008822A	16	チャネル 1 送信データ
2	RxD2	H'0008822C	16	チャネル2受信データ
	TxD2	H'0008822E	16	チャネル2送信データ
3	RxD3	H'00088230	16	チャネル3受信データ
	TxD3	H'00088232	16	チャネル3送信データ
4	RxD4	H'00088234	16	チャネル 4 受信データ
	TxD4	H'00088236	16	チャネル4送信データ
5	RxD5	H'00088238	16	チャネル5受信データ
	TxD5	H'0008823A	16	チャネル5送信データ
6	RxD6	H'0008823C	16	チャネル6受信データ
	TxD6	H'0008823E	16	チャネル6送信データ

仮想ポートRxDの構成を表 2-5、仮想ポートTxDの構成を表 2-6、RxDとTxDのビット機能を表 2-7に示します。





ビット	ビット名	初期値	R/W	説明
0	D0	0	R/W	データビット。
1	D1	0	R/W	8 ビットデータの場合は、D7~D0 を使用します。
2	D2	0	R/W	7 ビットデータの場合は、D6~D0 を使用します。
3	D3	0	R/W	
4	D4	0	R/W	
5	D5	0	R/W	
6	D6	0	R/W	
7	D7	0	R/W	
12~8	-	すべて 0	-	予約ビット。
				常に0が読み出されます。書き込みは0としてください。
13	FE	0	R/W	フレーミングエラービット。
				本ビットが1にセットされたデータについて、SCI はフレーミングエラー を発生させます。
14	PE	0	R/W	パリティエラービット。
				本ビットに1がセットされたデータについて、SCIはパリティエラーを発 生させます。
15	SB	1	R/W	スタートビット。
				送信側が送信開始時に 1->0、送信終了時に 0->1 に変化させる。

表2-7 RxD、TxD ビット機能

シミュレータ・デバッガではデータ送受信動作を抽象化し、データを一斉に送受信します。シミュレータ・デバッガ の受信動作を 図 2-1に、送信動作を 図 2-2 に示します。



図 2-1 シミュレータ・デバッガの受信動作



図 2-2 シミュレータ・デバッガの送信動作

本シミュレータ・デバッガでは、仮想ポートへのファイル入出力が可能です。詳細は、「3.4.6 仮想ポートへのファイ ル入出力を行う」を参照してください。

#### 2.8.3 割り込みコントローラ

(1) サポート範囲

RX600 シリーズシミュレータ・デバッガでは CMT と SCI に関連する割り込みコントローラ (ICU) をサポートしています。

また、CPU への割り込みだけをサポートしており、DTC、および DMAC 起動はサポートしていません。

(2) 制御レジスタ

シミュレータ・デバッガでサポートしているICUの制御レジスタを表2-8に示します。

表中サポート状況の はサポート、 は「2.8.3 (1) サポート範囲」で説明した機能に関するビットのみサポートしています。

制御レジスタは必ずレジスタサイズでアクセスしてください。



周辺機能モ ジュール名	制御レジスタ	サポート状況
ICU	IRn (n=028 ~ 029、214 ~ 241)	
	ISELR028	
	ISELR029	
	ISELR030	
	ISELR031	
	ISELR215	
	ISELR216	
	ISELR219	
	ISELR220	
	ISELR223	
	ISELR224	
	ISELR227	
	ISELR228	
	ISELR231	
	ISELR232	
	ISELR235	
	ISELR236	
	ISELR239	
	ISELR240	
	IER03	
	IER1A	
	IER1B	
	IER1C	
	IER1D	
	IER1E	
	IPRm (m=04 ~ 07、80 ~ 86)	
	FIR	

表2-8 シミュレータ・デバッガでサポートする ICU 制御レジスタ

制御レジスタのアドレス、割り込みベクタ番号、および割り込み優先順位レジスタ位置は[周辺モジュールの構成]ダイ アログボックスで参照・変更することができます。[周辺モジュールの構成]ダイアログボックスの詳細については、「3.4 周辺機能シミュレーションを設定する」を参照してください。

(3) ICU 使用時の注意

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

2.8.4 クロック

シミュレータ・デバッガでは、メモリアクセスにかかわるシステムクロック、周辺機能用クロック、タイマを動かす クロックをサポートします。

メモリマップで指定するサイクル数は内部クロックになります。システムクロックと周辺機能用クロックの比は、[周辺機能シミュレーションの設定]ダイアログボックスで設定してください。

タイマを動かすクロックの分周率は、タイマ制御レジスタで指定してください。

#### 2.8.5 周辺機能を使用する

周辺機能を使用するにはシミュレータ・デバッガ起動時に表示される[周辺機能シミュレーションの設定]ダイアログ ボックスで使用するモジュールを登録する必要があります。

モジュール登録の詳細については、「3.4 周辺機能シミュレーションを設定する」を参照してください。

# 2.9 トレース

シミュレータ・デバッガは、実行結果をトレースバッファに書き込みます。トレース情報の取得条件は、[トレース取 得]ダイアログボックスで指定します。[トレース取得]ダイアログボックスは、[トレース]ウィンドウ上で右クリックして ポップアップメニューを表示し、[トレース設定...]を選択することによって表示できます。取得したトレース情報は、[ト レース]ウィンドウに表示します。

トレース情報は検索することができます。検索条件は、[検索]ダイアログボックスで設定します。[トレース検索]ダイ アログボックスは、[トレース]ウィンドウ上で右クリックしてポップアップメニューを表示し、[検索 -> 検索...]を選択す ることによって表示できます。

詳しくは、「3.7 トレース情報を見る」を参照してください。

# 2.10 標準入出力およびファイル入出力処理

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

r					
番号	機能コード	機能名	内容		
1	H'21	GETC	標準入力からの1バイト入力		
2	H'22	PUTC	標準出力への1バイト出力		
3	H'23	GETS	標準入力からの1行入力		
4	H'24	PUTS	標準出力への1行出力		
5	H'25	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'27	FGETC	ファイルからの1バイト入力		
8	H'28	FPUTC	ファイルへの1バイト出力		
9	H'29	FGETS	ファイルからの1行入力		
10	H'2A	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		

表2-9 入出力機能一覧

入出力機能の詳細は、「3.12 標準入出力およびファイル入出力を行う」を参照してください。

# 2.11 ブレーク条件

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

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

(1) ブレーク系コマンドの条件成立によるブレーク

ブレーク条件を設定するコマンドには次の9種類があります。

- BREAKPOINT : 命令実行位置によるブレーク
- BREAK\_ACCESS :メモリ範囲のアクセスによるブレーク
- BREAK\_CYCLE :実行サイクル数によるブレーク
- BREAK\_DATA :メモリ書き込みデータ値によるブレーク
- BREAK\_DATA\_DIFFERENCE :メモリのデータ値の変化量(差分)によるブレーク
- BREAK\_DATA\_INVERSE
   :メモリのデータ値の符号反転によるブレーク
- BREAK\_DATA\_RANGE : メモリのデータ範囲によるブレーク
- 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、「3.6 シミュレータ・デバッガのブレークポイントを使用する」を参照してください。

ユーザプログラム実行中にブレーク条件が成立しプログラムが中断した場合、ブレークポイントの命令を実行しない で停止するか、実行してから停止するかを表 2-10に示します。

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

表2-10 ブレーク条件成立時の処理

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

ユーザプログラム実行中にブレーク条件が成立すると、ブレーク条件成立のメッセージを[アウトプット]ウィンドウに 表示して、命令実行を中断します。

(2) ユーザプログラムの実行時エラー検出によるブレーク

シミュレータ・デバッガでは、CPUの例外発生機能では検出できないプログラムの誤りを検出するためにシミュレー ションエラーを設けています。これらのエラーが発生した場合に、シミュレーションを停止するか、続行するかを [シミュ レータシステム]ダイアログボックスにより選択できます。エラーの種類、エラーメッセージ、エラー発生要因、および 続行時のシミュレータ・デバッガの動作を表 2-11に示します。

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
メモリアクセスエラー/ Memory Access Error (Address:H'nnnnnnn)	<ul> <li>確保していないメモリ領域をアクセスしようとした</li> <li>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</li> <li>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</li> <li>メモリが存在しない領域をアクセス</li> </ul>	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
	しようとした	

#### 表2-11 シミュレーションエラー一覧

停止モードの場合、シミュレーションエラーが発生するとシミュレータ・デバッガは、命令実行を中止してエラーメッ セージを表示後、コマンド待ち状態に戻ります。シミュレーションエラー停止後の PCの状態を表 2-12に示します。な お、シミュレーションエラー停止後 PSWの内容は変化しません。

エラーの種類	PC の内容
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

#### 表2-12 シミュレーションエラー停止時のレジスタ

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

- (a) 最初は停止モードで実行させて、意図している箇所以外にエラーがないかどうかを確認してください。
- (b) 確認が完了したら、続行モードで実行してください。
- 【注】 停止モードでエラーが発生して停止した状態から、モードを続行モードに変更してシミュレーションを再開する と、正しくシミュレーションできない場合があります。シミュレーションを再開する場合は、レジスタ内容、メ モリの内容をエラー発生前の状態に戻してから再実行するようにしてください。
- (3) トレースバッファ満杯によるブレーク

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

Trace Buffer Full

(4) WAIT 命令実行によるブレーク

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

WAIT Instruction

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

(5) [停止]ボタンによるブレーク

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

Stop

Go、Step コマンドにより実行を再開できます。

#### 2.12 浮動小数点データ

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

・[ブレーク種別の選択]ダイアログボックスにおいて、ブレーク種別を [ブレークデータ] や [ブレークレジスタ]

と指定したときのデータ

- ・[メモリ]ウィンドウにおけるデータ
- ・[メモリフィル]ダイアログボックスにおけるデータ
- ・[メモリ検索]ダイアログボックスにおけるデータ
- ・レジスタ値編集ダイアログボックスでの入力値

浮動小数点データフォーマットは、ANSIC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、浮動小数点数の 10 進->2 進変換で発生する丸めのモードに RN(最近値丸め)を使用します。

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、非正規化数のまま処理します。また、10 進->2 進変換時にオーバフローが発生した場合、無限大を設定します。



# 2.13 関数呼び出し履歴の表示

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

関数呼び出し履歴を更新するのは、以下のような場合です。

- 「2.11 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、「High-performance Embedded Workshop ユーザーズマニュアル」を参照してください。

# 2.14 パフォーマンス測定

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

#### 2.14.1 プロファイラ

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

プロファイル情報はリスト形式、ツリー形式、チャート形式で表示します。

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

詳しくは、「3.8 プロファイル情報を見る」を参照してください。

2.14.2 パフォーマンス解析

パフォーマンス解析はユーザプログラム内の指定関数について実行サイクル数、呼び出し回数を表示します。指定関数のみについてパフォーマンスデータを取得するため、プロファイラよりも高速なシミュレーションが可能です。詳しくは、「3.9 パフォーマンスを解析する」を参照してください。

## 2.15 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、「3.6 シミュレータ・デバッガのブレークポイントを使用する」を参照してください。

(2) ウィンドウによる擬似割込み発生

[トリガ]ウィンドウ、または[GUI I/O]ウィンドウのボタンをクリックすることにより、擬似割込みを発生させること ができます。

詳しくは、「3.11 手動で擬似割込みを発生させる」を参照してください。

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

(3) 擬似割込み発生によるブレーク

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



【注】 擬似割込みでは、割り込みベクタ番号と、割り込み優先順位を指定します。割り込み優先順位には0~8、また は0~H'10が指定できます。0~8の場合は8、0~H'10の場合はH'10を指定すると高速割り込みとして扱いま す。

## 2.16 カバレジ

シミュレータ・デバッガでは、ユーザが指定した測定範囲について命令実行中に命令カバレジ情報を収集できます。 測定範囲は直接アドレスを指定して設定するほかに、ソースファイル名を指定してそのファイルに含まれる全関数を 設定することができます。

命令カバレジ情報を利用することで各命令の実行状態を観察できます。さらにプログラムのどの部分が未実行である かを容易に特定できます。

収集した命令カバレジ情報は[カバレジ]ウィンドウに表示します。

命令カバレジ情報は命令実行済のソース行に対応するカラムを [エディタ]ウィンドウ上に強調表示します。

また、測定対象のアドレス範囲または関数について、カバレジ統計情報をパーセント形式で表示します。これにより プログラムがどのくらい実行されているかを定量的に把握できます。

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

詳しくは、「3.10 コードカバレジを測定する」を参照してください。





# 3. デバッグ

この章では、シミュレータ・デバッガ特有の操作と関連するウィンドウおよびダイアログボックスについて説明します。

HEWで共通な下記機能については、HEWのヘルプを参照ください。

- ・デバッグの準備
- ・プログラムを表示する
- ・メモリ内容を参照 / 設定する
- ・メモリ内容を波形形式で表示する
- ・メモリ内容を画像形式で表示する
- ・任意のアドレスのメモリ内容を参照 / 設定する
- ・I/Oレジスタを参照 / 設定する
- ・レジスタを参照 / 設定する
- ・プログラムを実行 / 停止、リセットする
- ・現在の状態の参照
- ・複数デバッギングプロットフォームの同期
- ・コマンドラインインタフェースのデバッグ
- ・Elf / Dwarf2のサポート
- ・ラベルを参照 / 設定する

# 3.1 ワークスペースの作成

シミュレータ・デバッガを使用するためには、まずHEWでワークスペースを作成する必要があります。ここでは、シ ミュレータ・デバッガ特有の説明のみをします。ワークスペース作成の詳細は、HEWのマニュアルを参照ください。

3.1.1 ターゲットの選択

HEWを起動し、新規プロジェクトワークスペースの作成を行う場合、下記ダイアログボックスを表示します。 ここで、シミュレータ・デバッガのターゲットを選択してください。



新規プロジェクト-8/10-デバッガ	<u>?×</u>
	- ゲット : ▼ RX600 Simulator
	ターケ <sup>5</sup> ットタイフ <sup>*</sup> : RX600 ▼ ターケ <sup>5</sup> ットCPU: All CPUs ▼
< 戻る( <u>B</u> )	次へ(N) > 完了 キャンセル

図 3-1 デバッガターゲット設定画面 (8/10)

[ターゲット]	デバッガターゲットを設定します。デバッガターゲットを選択(チェッ
	ク)してください。デバッガターゲットは未選択でも複数選択してもか
	まいません。

- [ターゲットタイプ] [ターゲット]に表示するターゲットの種類を指定します。
- [ターゲットCPU] [ターゲット]に表示するCPUの種類を指定します。

# 3.1.2 シミュレータ用ワークスペースの設定

新規プロジェクト-9/10では、シミュレータ用ワークスペースの設定を行うことができます。

新規プロジェクトー9/10ーデバッガオプション		?×
	ターケット名: RX600 Simulator コア: 〈single core〉 コンフィグレーション名: SimDebug_RX600 詳細オフジョン: Item Setting Simulator I/O addr. 0x0 Bus mode 0 Endian Little Patch Off	
A A A A A A A A A A A A A A A A A A A	変更(例)	
< 戻る( <u>B</u> )	次へ(N) > 完了 キャンt	274

図 3-2 デバッガオプション設定画面 (9/10)

[詳細オプション]	デバッガターゲット を選択して[変更]を [Item]を選択しても[	ーゲットのオプションを設定します。変更する場合は、[Item] 変更]をクリックしてください。なお、変更できない項目の場合、 {しても[変更]はグレーのままです。		
	[Simlator I/O]	ユーザプログラムから 出力を行うI/Oシミュ  は無効([Disable])	標準入出力またはファイル入 ィーションは有効([Enable])また	
	[Simulator I/O addr.]	上記I/Oシミュレーショ	ョン開始アドレス	
	[Bus mode]	現状未使用		
	[Endian]	CPUのエンディアンを	表示します	
	[Patch]	割り込み優先レベル、	およびMVTIPL命令の有効・無効	
		Off	割り込み優先レベルは0~15、	
			MVTIPL命令は有効	
		RX610	割り込み優先レベルは0~7、 MVTIPL命令は無効	

[詳細オプション]以外の項目については「High-performance Embedded Workshop ユーザーズマニュアル」を参照してください。



# 3.2 シミュレータ・デバッガの起動

シミュレータ・デバッガを使用する設定があらかじめ登録されているセッションファイルに切り替えることにより、 シミュレータ・デバッガを接続することができます。

プロジェクト作成時にターゲットを選択している場合は、その選択したターゲットの個数分のセッションファイルが 作成されています。

図 3-3に示すツールバーのドロップダウンリストから、接続するターゲットに対応したセッションファイルを選択して ください。



図 3-3 セッションファイルの選択

シミュレータ・デバッガが登録されているセッションファイルが選択されており、シミュレータ・デバッガが接続解 除状態の場合は、[デバッグ->接続]を選択するか、接続ツールバーボタン ም をクリックしてください。

シミュレータ・デバッガを接続解除する場合は、[デバッグ->接続解除]を選択するか、接続解除ツールバーボタン 🤻 を クリックしてください。

# 3.3 シミュレータ・デバッガの設定を変更する

本節ではシミュレータ・デバッガ起動後にシミュレータの設定を変更する方法について説明します。

#### 3.3.1 CPU のエンディアンと動作周波数を設定する

CPU のエンディアン、および動作周波数はシミュレータ・デバッガ起動時に表示する[シミュレータの設定]ダイアログボックス[CPU の構成]タブで設定します。

シミュレータの設定	?×
CPUの構成 周辺機能シミュレーション	
エンディアン(E):	
システムクロック(ICLK)周波数( <u>C</u> ): 100 MHz	
このダイアログを表示しない( <u>S</u> )	OK キャンセル

図 3-4 シミュレータの設定ダイアログボックス(CPUの構成)

本ダイアログボックスでは下記項目を設定します。 [エンディアン] CPU のエンディアンを設定します [Big] Big エンディアン

	[DIg]	DIg エノノイナノ
	[Little]	Little エンディアン
[システムクロック(ICLK)周波数]	CPU の動作	乍周波数を設定します(単位:MHz)
	指定範囲:	: 1~1000

[このダイアログを表示しない]チェックボタンをチェックすると、次回以降シミュレータ・デバッガ起動時に本ダイア



ログボックスを表示しなくなります。

#### 3.3.2 シミュレータシステムの設定を変更する

I/O シミュレーションの開始位置、実行モード等の設定変更は、[シミュレータシステム]ダイアログボックスの[システム]タブで行います。

[シミュレータシステム]ダイアログボックス[システム]タブを開くには、[基本設定->シミュレータ->システム…]を選択 するか、[シミュレータシステム]ツールバーボタン **14** をクリックします。

シミュレータシステム			? ×
システム Xモリ			
CPU( <u>C</u> ): RX600		<b>_</b>	
ビットサイズ( <u>B</u> ):  D'32	1/0シミュレーションアドレスΦ:  H'00000000	▼ 有効(N)	
エンディアン( <u>E</u> ): Little Endian	実行モード公: Stop	▼ I¥細(D)	
割り込み優先レベル( <u>T</u> ): D-7 (MVTIPL命令無効)	レスポンス( <u>P</u> ): D'40000		
▶ 命令デコード結果をキャッシュしてシミュレーション速度を	i向上する( <u>A</u> )		
	OK	キャンセル 道用(	( <u>A</u> )

図 3-5 シミュレータシステムダイアログボックス(システムタブ)

本ダイアログボックスでは下記項目を設定または表示します。

[CPU]	現在設定している CPU。
[ビットサイズ]	CPUのアドレス空間サイズ(ビット数)。
[エンディアン]	CPU のエンディアン。
[割り込み優先レベル]	割り込み優先レベル、および MVTIPL 命令の有効、無効を表示します。
	0-7 (MVTIPL 命令無効) 割り込み優先レベルは 0 ~ 7 です。
	0-15 (MVTIPL 命令有効) 割り込み優先レベルは 0 ~ 15 です。
[I/O シミュレーション	ユーザプログラムから標準入出力またはファイル入出力を行うための I/O シミュレー
アドレス]	ションの開始位置を指定します。
	[ <b>有効</b> ] チェックすると I/O シミュレーションが有効となります。
[レスポンス]	何命令ごとにウィンドウをリフレッシュするかを指定します。
	(1~D'2,147,483,647、デフォルトは D'40000)
[実行モード]	シミュレーションエラーが発生した場合のシミュレータ・デバッガ動作を規定します。
	割り込みはシミュレーションエラーに含み、本設定に従います。
	また、[詳細]ボタンで、割り込み発生時の動作を規定することもできます。
	[停止] シミュレーションを停止します。
	[続行] シミュレーションを続行します。
[命令デコード結果を キャッシュしてシミュ	命令実行時にデコード結果を保持し、同一アドレス実行時にデコード結果を利用する機 能の有効、無効を設定します。
レーション速度を向上 する]	チェックすると命令デコードキャッシュ機能が有効となり、シミュレーション速度が向 上します。

変更内容は、[OK]ボタンまたは[適用]ボタンをクリックすることにより設定します。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

【注】 命令デコードキャッシュ機能はデコード結果を再利用するため、自己書き換えコードを使用するプログラムでは 使用できません。また、プログラムの意図しない動作により命令が書き換わると、正しくエラーを検出できない 場合があります。



#### 3.3.3 メモリマップおよびメモリリソースの設定を変更する

メモリマップの設定および変更とメモリリソースの設定および変更は、[シミュレータシステム]ダイアログボックスの [メモリ]タブで行います。

XFU マップ(M):       XFU ワップ(M):       XFU リソース(B):       Image: Constraint of the second seco	ι <i>ν−φ</i> システ <i>μ</i>	1									?
Begin         End         Type         Size         Read         Write         Endian           00000000         0001FFFF         RAM          1         1         Little         0000000         00007FFF         U/O          1         1          0000000         00007FFF         U/O          1         1          0000000         00007FFF         U/O          1         1          FFFF8000         FFFFFFF         U/O          1         1         Little         0000000         00007FFF         U/O          1         1         Little         0000000         0007FFFF         U/O          1         1         Little         000000         0007FFFF         U/O          1         1          000000         007FFFF         U/O          1         1          000000         007FFFF         U/O          1         1          000000         007FFF         U/O          1         1         Little          0000000         00FFFFF         N/O          1         1         Little	メテム メモリ メモリ マップ( <u>1</u>	<u>v</u> ):			0		×	メモリ	リソース(	B):	• ∎ ≥ <sub>∎</sub> ×,
00000000         0001FFFF         RAM          1         1         Little           00000000         000FFFFF         I/O          1         1            00100000         0007FFFF         ROM          1         1            00100000         0007FFFF         ROM          1         1         Little           007F0000         007FC4FF         I/O          1         1            007FC000         007FFFFF         ROM          1         1            007FFC000         007FFFFF         ROM          1         1            002E00000         00FFFFFF         ROM          1         1            002E00000         0FFFFFFF         ROM          1         1         Little            FFFFE000         FFFFFFF         ROM          1         1         Little            FFFE00000         FFFFFFF         ROM          1         1         Little            FFE000000         FFFFFFFF         ROM         <	Begin	End	Туре	Size	Read	Write	Endian	Be	gin	End	Attribute
00080000         000FFFFF         I/O          1         1            00100000         00107FFF         ROM          1         1         Little           007F8000         007F9FFF         RAM          1         1         Little           007FC000         007FFFF         I/O          1         1            007FC000         007FFFFF         ROM          1         1            007FFC00         007FFFFF         ROM          1         1            007E00000         00FFFFFF         ROM          1         1            007E00000         00FFFFFF         ROM          1         1         Little           FEFFE000         FEFFFFF         ROM          1         1         Little           FFE000000         FFFFFFF         ROM          1         1         Little           FFE00000         FFFFFFF         ROM          1         1         Little	00000000	0001FFFF	RAM		1	1	Little	000	00000	00007FFF	リード/ライト
00100000         00107FFF         ROM          1         1         Little           007F8000         007F9FFF         RAM          1         1         Little           007F000         007FC4FF         I/O          1         1         Little           007FC000         007FFFFF         ROM          1         1            007FFC00         00FFFFFF         ROM          1         1            0020000         00FFFFFF         ROM          1         1         Little           FEFFE000         FEFFFFFF         ROM          1         1         Little           FF7FC000         FFFFFFF         ROM          1         1         Little           FF2E00000         FFFFFFF         ROM          1         1         Little           FFE000000         FFFFFFF         ROM          1         1         Little	00080000	000FFFFF	1/0		1	1		FFF	F8000	FFFFFFF	リード/ライト
007F8000         007F9FFF         RAM          1         1         Little           007FC000         007FC4FF         I/O          1         1            007FC000         007FFFF         I/O          1         1            007FFC00         007FFFFF         I/O          1         1            00EFFFFF         ROM          1         1         Little            FEFFE000         FEFFFFFF         ROM          1         1         Little           FF7FC000         FFFFFFF         ROM          1         1         Little           FFFE00000         FFFFFFF         ROM          1         1         Little           FFE00000         FFFFFFF         ROM          1         1         Little	00100000	00107FFF	ROM		1	1	Little				
007FC000         007FC4FF         I/O          1         1            007FFC00         007FFFFF         I/O          1         1            002FFC00         007FFFFF         I/O          1         1            00E00000         00FFFFFF         ROM          1         1         Little           FEFFE000         FEFFFFFF         ROM          1         1         Little           FF7FC000         FFFFFFF         ROM          1         1         Little           FFE00000         FFFFFFFF         ROM          1         1         Little	007F8000	007F9FFF	RAM		1	1	Little				
007FFC00         007FFFF         I/O          1         1            00E00000         00FFFFFF         ROM          1         1         Little           FEFFE000         FEFFFFFF         ROM          1         1         Little           FF7FC000         FF7FFFFF         ROM          1         1         Little           FFFE00000         FFFFFFF         ROM          1         1         Little           FFE00000         FFFFFFF         ROM          1         1         Little	007FC000	007FC4FF	1/0		1	1					
00E00000         00FFFFFF         ROM          1         1         Little           FEFFE000         FEFFFFF         ROM          1         1         Little           FF7FC000         FF7FFFF         ROM          1         1         Little           FF600000         FFFFFFF         ROM          1         1         Little	007FFC00	007FFFFF	I/O		1	1					
FEFFE000         FEFFFFF         ROM          1         1         Little           FF7FC000         FF7FFFF         ROM          1         1         Little           FFE00000         FFFFFFF         ROM          1         1         Little	00E00000	OOFFFFFF	ROM		1	1	Little				
FF7FC000         FF7FFFF         ROM          1         1         Little           FFE00000         FFFFFFF         ROM          1         1         Little	FEFFE000	FEFFFFF	ROM		1	1	Little				
FFE00000         FFFFFFF         ROM          1         1         Little	FF7FC000	FF7FFFFF	ROM		1	1	Little				
	FFE00000	FFFFFFF	ROM		1	1	Little				
	1	!		!			<u> </u>	-		!	-
									_		
OK キャンセル 近								OK		キャンセル	適用(色)

図 3-6 シミュレータシステムダイアログボックス(メモリタブ)

本ダイアログボックスでは以下の項目を設定します。

[メモリマップ]	メモリ情報として、メモリ種別とその先頭アドレス・終了アドレス、データ
	バス幅、アクセスステート数を表示します。
[メモリリソース]	現在設定しているメモリリソースのアクセス種別とその先頭アドレス・終了
	アドレスを表示します。

[メモリリソース]は次の各ボタンにより追加・変更・削除することができます。

【メモリリソース]の項目を追加します。クリックすることにより、[メモリリソース設定]ダイア
 ログボックスが開き、追加することができます。

2

[メモリリソース]の項目を変更します。変更したい項目をリストボックス上で選択後、ボタンを クリックします。クリックすることにより、[メモリリソース設定]ダイアログボックスが開き、 変更することができます。



[メモリリソース]の項目を削除します。削除したい項目をリストボックス上で選択後、ボタンを クリックします。

[メモリマップ]は、以下の各ボタンにより追加・変更・削除ができます。



×

[メモリマップ]の項目を追加します。クリックすると、[メモリマップ設定]ダイアログボックス (図 3-7 参照)が開き、追加することができます。

[メモリマップ]の項目を変更します。変更したい項目をリストボックス上で選択後、ボタンをク リックします。クリックすると、[メモリマップ設定]ダイアログボックス(図 3-7 参照)が開き、 変更することができます。

[メモリマップ]の項目を削除します。削除したい項目をリストボックス上で選択後、ボタンをク リックします。

なお、[メモリマップ]は、 岡ボタンによりデフォルト値にリセットすることができます。 変更内容は、[OK]ボタンまたは[適用]ボタンをクリックすることにより設定します。[キャンセル]ボタンをクリックす ると、設定しないでダイアログボックスを閉じます。

最適化リンケージエディタが出力するリンケージリストファイル(.map)がある場合、メモリマップおよびリンケージ マップ情報に基づきメモリリソースを自動的に確保することができます。 詳しくは「High-performance Embedded Workshop ユーザーズマニュアル」の「メモリリソースを自動的に確保する」を参 照してください。

## 3.3.4 メモリマップ設定ダイアログボックス

[メモリマップ設定]ダイアログボックスでは、対象 CPU のメモリマップを設定します。

各項目に表示する内容は、対象 CPU によって異なります。シミュレータ・デバッガはこれらの値をメモリアクセスの シミュレーションに使用します。

メモリマップ設定	? ×
メモリ種別( <u>M</u> ):	(COK
	キャンセル
H'00000000	
終了アドレス( <u>E</u> ):	
H'FFFFFFF 🗾 🗾	
データバスサイズ( <u>D</u> ): 32	
リードステート数( <u>R</u> ): 1	
ライトステート数( <u>W</u> ):	
エンディアン( <u>N</u> ):	
Little	

図 3-7 メモリマップ設定ダイアログボックス

本ダイアログボックスでは以下の項目を設定します。

[メモリ種別]	メモリ種別	
	[ROM]	内蔵 ROM
	[RAM]	内蔵 RAM
	[EXT]	外部メモリ
	[IO]	内蔵 I/O
[開始アドレス]	メモリ種別に	対応するメモリの先頭アドレス
[終了アドレス]	メモリ種別に	対応するメモリの終了アドレス
[データバスサイズ]	メモリのデー	・タバス幅
[リードステート数]	メモリのリー	・ドアクセスステート数
[ライトステート数]	メモリのライ	トアクセスステート数
[エンディアン]	当該メモリ領	域のエンディアン

変更内容は、[OK]ボタンをクリックすることにより設定します。[キャンセル]ボタンをクリックすると、設定しないで ダイアログボックスを閉じます。

#### 【注】

- (1) メモリリソースを確保している領域のメモリマップは、削除・変更することができません。あらかじめ、[シミュレー タシステム]ダイアログボックスの[メモリ]タブによりメモリリソースを削除してから、メモリマップを削除・変更し てください。
- (2) 外部メモリ以外のメモリ種別では、データバスサイズの表示・変更はできません。

- (3) データバスサイズ、リードステート数、およびライトステート数は命令シミュレーションには影響しません。メモリ アクセスステート数は常に1となります。
- (4) メモリマップは 16 バイト境界でのみ設定可能です。16 バイト境界以外での設定は、設定したメモリマップを含む 16 バイト境界に補正します。
- (5) 内蔵 I/O のエンディアンは、表示・変更することができません。
- (6) 本ダイアログボックスでは、内蔵ROM、および内蔵RAMのエンディアンの変更はできません。内蔵ROM、および内蔵RAMのエンディアンは、[シミュレータの設定]ダイアログボックスで変更することができます。[シミュレータの設定]ダイアログボックスの詳細については、「3.3.1 CPUのエンディアンと動作周波数を設定する」を参照してください。
- 3.3.5 メモリリソース設定ダイアログボックス

[メモリリソース設定]ダイアログボックスでは、メモリリソースの設定・変更を行います。

メモリリソース設定		? ×
開始アドレス( <u>B</u> ):  H'00000000	<b>•</b>	OK
終了アドレス( <u>E</u> ):  H'00007FFF	<b>.</b>	447.00
アクセス種別( <u>A</u> ):  Read/Write	-	

図 3-8 メモリリソース設定ダイアログボックス

本ダイアログボックスでは以下の項目を設定します。

- [開始アドレス] 確保するメモリ領域の先頭アドレス
- [終了アドレス] 確保するメモリ領域の終了アドレス
- [アクセス種別] アクセス種別

[Read]	読み出しのみ可能
[Write]	書き込みのみ可能
[Read/Write]	読み書き可能

各項目を指定後、[OK]ボタンをクリックすることによりメモリリソースの設定・変更を行います。[キャンセル]ボタン をクリックすると、設定しないでダイアログボックスを閉じます。

【注】

- (1) メモリリソースを設定すると、PCのメモリを使用します。したがって、メモリリソースを大きく取りすぎると、PC の動作が極端に遅くなる場合があります。
- (2) メモリリソースは16 バイト境界でのみ設定可能です。16 バイト境界以外での設定は、設定したメモリリソースを含む16 バイト境界に補正します。またアクセス種別に関しても16 バイト境界となります。

16 バイト以下で使用する場合は、ハードウェアマニュアルに沿った範囲内のメモリを使用してください。

(3) 命令による読み出しのみ許可メモリへの書き込み、および書き込みのみ許可メモリからの読み出しはメモリアクセス エラーとなります。

# 3.4 周辺機能シミュレーションを設定する

シミュレータ・デバッガは周辺機能シミュレーションを DLL 形式のモジュールで実現しています。

ここでは周辺機能シミュレーションを有効にするための周辺機能シミュレーションモジュールの登録方法および周辺 機能シミュレーション構成の設定方法を説明します。

#### 3.4.1 周辺機能シミュレーションモジュールを登録する

周辺機能シミュレーションモジュールの登録は[シミュレータの設定]ダイアログボックス[周辺機能シミュレーション] タブで行います。[シミュレータの設定]ダイアログボックスはシミュレータ・デバッガ起動時に表示されます。

本ダイアログボックスで周辺機能シミュレーションモジュールを登録すると、当該周辺機能シミュレーションモ ジュールが提供する周辺機能シミュレーションを利用可能となります。シミュレータ・デバッガ起動後は、周辺機能シミュ レーションモジュール登録内容を変更することはできません。利用する周辺機能シミュレーションモジュールを変更する 場合はシミュレータ・デバッガを再起動し、本ダイアログボックスを表示させてください。

シミュレータの設定		? ×
CPUの構成 周辺機能	きシミュレーション	
┌周辺機能( <u>F</u> ):		
Module Name	File Name	すべて登録( <u>E</u> )
СМТ	C:¥Program Files¥Renesas_Evaluation_RX¥RX¥Toc	± #7787784(p)
LICU	C:¥Program Files¥Renesas_Evaluation_RX¥RX¥Toc	タヘビ時期家団
		詳細(工)
」 周辺クロックレード( <u>P</u> )	: 1 <b>•</b>	
🗖 このダイアログを表示し	」ない(S) OK	キャンセル

図 3-9 シミュレータの設定ダイアログボックス(周辺機能シミュレーションタブ)

本ダイアログボックスでは以下の項目を設定します。

[周辺機能]	周辺機能シミュレーションモシュールの情報を表示しより。
	[Module Name] シミュレーションする周辺機能名
	[File Name] 周辺機能シミュレーションモジュールファイル名
	[Module name]欄のチェックボックスをチェックした周辺機能シミュレーショ
	ンモジュールが登録されて、利用可能となります。
[すべて登録]	すべての周辺機能シミュレーションモジュールを有効にします。
[すべて解除]	すべての周辺機能シミュレーションモジュールを無効にします。
[詳細]	周辺機能情報の表示、周辺機能の開始アドレス、および割り込み要因情報の変
	更を行うための[周辺モジュールの構成]ダイアログボックスを表示します。
[周辺クロックレート]	周辺クロックとシステムクロックの比(周辺1クロックがシステムクロックい
	くつに相当するか)を指定します。
	(1,2,3,4,6,8,12,16,24,32から選択)

各項目を指定後、[OK]ボタンをクリックすることにより周辺機能シミュレーションの設定、変更を行います。[キャン セル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

[このダイアログを表示しない]チェックボタンをチェックすると、次回以降シミュレータ・デバッガ起動時に本ダイア ログボックスを表示しなくなります。



#### 3.4.2 周辺機能のアドレスを変更する

周辺機能のアドレス変更は[周辺モジュールの構成]ダイアログボックスで行います。割り込み要因情報を持つ周辺機能 のアドレス変更は[周辺モジュールの構成]ダイアログボックス[アドレス]タブで行います。

[周辺モジュールの構成]ダイアログボックスを開くには、[シミュレータの設定]ダイアログボックス[周辺機能シミュレーション]タブの[周辺機能]欄で周辺機能を選択して、[詳細...]ボタンをクリックします。

周辺モジュールの構成	t	<u>?×</u>
アドレス 割り込み	-1	
モジュール( <u>M</u> ): ICU	<b>•</b>	
開始アドレス(レジ H'00087010	ジスタ)( <u>B</u> ): ・ マ)・	
Register IR028 IR029 IR030 IR031 ISELR028	\u00c0     \u00c	
ОК	キャンセル	適用( <u>A</u> )

図 3-10 周辺モジュールの構成ダイアログボックス (アドレスタブ)

本ダイアログボックスでは以下の項目を表示、設定します。

[モジュール] 選択した周辺機能シミュレーションモジュールでサポートしている周辺機能 名

[開始アドレス] [モジュール]で選択した周辺機能の開始アドレス

[レジスタアドレス] [モジュール]で指定した周辺機能のレジスタ名、レジスタアドレスを表示しま す。個々のレジスタアドレスは変更できません。

各項目を指定後、[OK]ボタンをクリックすることにより周辺機能のアドレスを設定します。[キャンセル]ボタンをク リックすると、設定しないでダイアログボックスを閉じます。

#### 3.4.3 周辺機能の割り込み要因情報を変更する

周辺機能の割り込み要因情報は[周辺モジュールの構成]ダイアログボックス[割り込み]タブで参照できます。 [周辺モジュールの構成]ダイアログボックスを開くには、[シミュレーションの設定]ダイアログボックス[周辺機能シ ミュレーション]タブの[周辺機能]欄で周辺機能を選択して、[詳細...]ボタンをクリックします。



周辺	周辺モジュールの構成						
7	アドレス割り込み						
	割り込み要因情	<b>韓辰(_])</b> :					
	Interrupt So	Vector Num	Priority Register				
	CMID	28	00087304/2-0				
	CMI	29	00087305/2-0				
	CMI2	30	00087306/2-0				
	CMB	31	00087307/2-0				
OK キャンセル 道用(A)							

図 3-11 周辺モジュールの構成ダイアログボックス (割り込みタブ)

本ダイアログボックスでは以下の項目を表示します。

割り込み要因情報

Interrupt Source Vector Number Priority Register Address/ Bit Field Position

周辺機能でサポートしている割り込み要因名 割り込みベクタ番号 割り込み優先順位レジスタアドレスとレジスタ 内のビット位置

割り込み要因情報を変更する場合は、変更したい割り込み要因をダブルクリックします。[割り込み要因情報の設定] ダイアログボックスが表示されます。

割り込み要因情報の設定	? ×
割り込み要因(S): CMID 割り込みベクタ番号(V):	<u> </u>
(夏)と//#12レンスクイレス(1)・ (○x00087304 ▼	
優先加則エレジスダザイス(Z): 8-bit	
優先順位レジスタビット位置(B): 2-0	

図 3-12 割り込み要因情報の設定ダイアログボックス

本ダイアログボックスでは以下の項目を表示、設定します。 割り込み要因 割り込み要因名 割り込みべクタ番号 割り込みベクタ番号 (接頭辞省略時は 10 進入力、10 進表示) 優先順位レジスタアドレス 割り込み優先順位レジスタのアドレス 優先順位レジスタサイズ 割り込み優先順位レジスタのサイズ 優先順位レジスタビット位置 割り込み優先順位レジスタ内ビット位置

各項目を指定後、[OK]ボタンをクリックすることにより割り込み要因情報を設定します。[キャンセル]ボタンをクリッ

3.デバッグ

クすると、設定しないでダイアログボックスを閉じます。

#### 3.4.4 制御レジスタのメモリリソース

制御レジスタ領域のメモリリソースは周辺機能シミュレーションモジュールが確保します。確保済の制御レジスタの メモリリソースを削除・変更しないでください。メモリリソース設定の詳細は、「3.3.3 メモリマップおよびメモリリソー スの設定を変更する」を参照してください。

#### 3.4.5 接続されている周辺機能を確認する

シミュレータ・デバッガ起動後は、 [ステイタス]ウィンドウの[Platform]シートの[Peripheral Modules]項目に接続されている周辺機能名を表示します。

#### 3.4.6 仮想ポートへのファイル入出力を行う

シミュレータ・デバッガでは端子の一部を仮想ポートとしてメモリ上に割付けています。これらの仮想ポートはファ イル入出力をサポートしています。本シミュレータ・デバッガでサポートしている仮想ポートについては、「2.8.2(3) デー タ入出力」を参照してください。

#### (1) 設定されているファイル入出力一覧を表示する

現在設定されているファイル入出力一覧を表示するには、[シミュレータシステム]ダイアログボックス[ポート入出力] タブを開きます。

[シミュレータシステム]ダイアログボックス[ポート入出力]タブを開くには、[基本設定->シミュレータ->システム...]を 選択し、[シミュレータシステム]ダイアログボックスの[ポート入出力]タブを選択します。

仮想ポートを持つモジュールが登録されていない場合は、[ポート入出力]タブは表示されません。

シミュレータシステム									
3	システム メモリ ポート入出力								
	Module	Port	File Name	I/O	Mode	Repeat Start	State		
	L								
	OK 作やンセル 道用(空)								

図 3-13 シミュレータシステムダイアログボックス (ポート入出力タブ)

#### 表示する項目は以下の通りです。

[Module]	モジュール名を表示します。				
[Port]	ポート名を表示します。				
[File Name]	ファイル名を表示します。				
[I/O]	ファイルの入力/出力を表示します。				
	[In]	:ファイル入力			
	[Out]	:ファイル出力			
[Mode]	ファイルのデータ入出力モードを表示します				
	[Repeat]	:繰り返し入力			
	[Once]	:1回のみ入力			
	[Overwrite]	: 上書き出力			
	[Append]	: 追記出力			


[Repeat Start] 繰り返し入力モード時の繰り返し開始番号を表示します。

- ファイルのオープン/クローズ状態を表示します。 [Open] :ファイルオープン状態
  - [Close] :ファイルクローズ状態

#### (2) ファイルを追加する

[State]

[ポート入出力]タブ上で右ボタンをクリックしてポップアップメニューから[追加]を選択するか、リスト項目をダブル クリックします。[ポート入出力指定]ダイアログボックスが開きます。

ポート入出力指定			? ×
ポート指定 モジュール( <u>M</u> ): SCI	•	ポート( <u>P)</u> :  R×D_0	<b>_</b>
- ファイル指定 ファイル( <u>F</u> ):			ブラウズ
- 入出力型	-入力モード(№) ● 繰り返し 開始行 ● 1回のみ	1	出力モード@ C 上書き C 追加
		OK	キャンセル

図 3-14 ポート入出力指定ダイアログボックス

設定	ģ	る項目は以	下の通り	です。

		*	
[ポート指定]	[モジュール]		入出力対象のポートが存在するモジュール名をリストから選択します。
	[ポート]		入出力対象のポート名をリストから選択します。
[ファイル指定]	[ファイル]		入出力ファイル名を指定します。
			ファイル型名を省略すると".csv"を付加します。
	[入出力]	[入力]	ファイル入力を行います。
		[出力]	ファイル出力を行います。
	[入力モード]	[繰り返し]	入力ファイルの最後に達した場合は、先頭に戻り繰り返し入力を行います。
			[開始行] 繰り返し入力モード時に、繰り返しを開始する行番号を指定しま
			す。 (1~65535)
		[1回のみ]	入力ファイルの最後に達した場合は、ファイル入力を終了します。
	[出力モード]	[上書き]	出力ファイルが既に存在する場合は、上書きします。
		[追加]	出力ファイルが既に存在する場合は、ファイルの最後から追記します。

1個のポートに割付けできるファイル数は、入出力各1ファイルです。同一ファイルを複数の入力ポートに指定することは可能です。

(3) ファイルを開く

[ポート入出力]タブ上で開きたいファイル行を選択後、ポップアップメニューから[開く]を選択すると、選択している ファイルを開きます。

(4) すべてのファイルを開く

[ポート入出力]タブ上でポップアップメニューから[すべて開く]を選択すると、すべてのファイルを開きます。

(5) ファイルを閉じる

[ポート入出力]タブ上で閉じたいファイル行を選択後、ポップアップメニューから[閉じる]を選択すると、選択してい

るファイルを閉じます。

(6) すべてのファイルを閉じる

[ポート入出力]タブ上でポップアップメニューから[すべて閉じる]を選択すると、すべてのファイルを閉じます。

(7) ファイル指定の編集

[ポート入出力]タブ上で編集したいファイル行を選択後、ポップアップメニューから[編集]を選択するか、ダブルクリックすると、[ポート入出力指定]ダイアログボックスが開き、ファイル指定を編集することができます。

(8) ファイルの削除

[ポート入出力]タブ上で削除したいファイル行を選択後、ポップアップメニューから[削除]を選択すると、選択しているファイル指定を削除します。

(9) 仮想ポートファイルフォーマット

仮想ポートファイルは CSV 形式です。

入力ファイルフォーマットを以下に示します。

<時間>、<データ>

入力ファイルは、入力する時間とデータを指定します。時間は前データからの差分をピコ秒(整数値)で指定します。時間には1以上の値を指定してください。また、データは16進数の整数値で指定してください。

出力ファイルフォーマットを以下に示します。

[Module] <モジュール名> [Port] <ポート名> [Length] <データビット長> [Data] <時刻>, <データ>

出力ファイルは、出力したモジュール名、ポート名、データビット長、時刻、およびデータを出力します。時刻は シミュレーション開始からデータを出力するまでの時間をピコ秒(整数値)で出力します。

# 3.5 メモリを操作する

#### 3.5.1 ウィンドウの表示内容を定期的に更新する

シミュレータ・デバッガでは[メモリ]ウィンドウのポップアップメニューから[自動更新]を選択すると、ユーザプログ ラム実行中に[メモリ]ウィンドウの表示内容を定期的に更新できます。

更新間隔のデフォルト値、および更新間隔の指定可能範囲は以下のとおりです。 更新間隔デフォルト値:100ミリ秒 更新間隔指定可能範囲:10ミリ秒~10,000ミリ秒

#### 3.5.2 I/O 領域の内容を表示、変更する

[メモリ]ウィンドウで I/O 領域を表示、変更する場合は、ハードウェアマニュアルに記載しているアクセスサイズに表示を切り替えてから行ってください。

ハードウェアマニュアルに記載しているアクセスサイズと異なるサイズでは、正しく表示、変更できない場合があり ます。



# 3.6 シミュレータ・デバッガのブレークポイントを使用する

シミュレータ・デバッガでは HEW 標準の PC ブレークポイントとは別により高度なブレークポイント機能を持っています。

これらブレークポイントについて、ブレーク条件の設定、ブレーク条件成立時の動作、および設定されているブレー クポイントの表示が行えます。

# 3.6.1 ブレークポイントを一覧表示する

現在設定されているブレークポイントを一覧表示するには[イベントポイント]ウィンドウを開きます。

[イベントポイント]ウィンドウは[表示->コード->イベントポイント]を選択するか、[イベントポイント]ツールバーボタン <mark>
[[]</mark>をクリックします。

イベントポイント	×
T S Condition	Action
BP Enable PC=FFFF90E4 (Tutorial.c/38)	Stop
	•
Software Break / Software Event /	

図 3-15 イベントポイントウィンドウ

#### 表示する項目は以下の通りです。

[Type]	ブレーク種別を表示します。
	[BP] : PC ブレーク
	[BA] :プレークアクセス
	[BD] :プレークデータ
	[BR] :プレークレジスタ
	[BS] :プレークシーケンス
	[BCY] :プレークサイクル
[State]	該当ブレークポイントの有効/無効を示します。
	[Enable] :有効
	[Disable] :無効
[Condition]	Break が成立する条件を表示します。表示内容はブレーク種別により異なります。
	ブレーク種別が BR の時はレジスタ名を、BCY の時はサイクル数を表示します。
	BP 時 : PC=プログラムカウンタ(対応するファイル名 / 行、シンボル名)
	BA 時 : Address=アドレス(シンボル名)
	BD 時 : Address=アドレス(シンボル名)
	BR 時 : Register=レジスタ名
	BS 時 : PC=プログラムカウンタ(対応するファイル名 / 行、シンボル名)
	BCY 時 : Cycle= <b>サイクル</b> 数(16 進表示)
[Action]	プ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 ] :ファイルヘ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
	[Trace Trigger] : トレース情報の取得開始

[Action]の設定で、[Stop]を指定した条件を[Software Break]タブに、[Stop]以外を指定した条件を[Software Event]タブに表示します。



3.6.2 ブレークポイントを設定する

[イベントポイント]ウィンドウのポップアップメニューで[設定…]を選択すると、[ブレーク種別の選択]ダイアログ ボックスが開きブレークポイントを設定できます。

[ブレーク種別の選択]ダイアログボックスからブレーク条件を設定する[条件の設定]ダイアログボックスとブレーク成 立時の動作を設定する[動作の設定]ダイアログボックスが開きます。[動作種別]として、[停止]を指定する場合は[Software Break]タブで、[停止]以外を指定する場合は[Software Event]タブでポップアップメニューを選択します。

(1) ブレーク種別を選択する

[イベントポイント]ウィンドウのポップアップメニューで[設定]を選択すると、[ブレーク種別の選択]ダイアログボックスが開きます。

ブレーク種別は[ブレーク種別の選択]ダイアログボックスの[ブレーク種別]フィールドで選択します。

ブレーク種別の選択			?×
ブレーク種別( <u>B</u> ):			OK
PCブレークポイント	-	詳細( <u>D</u> )	キャンセル
動作種別( <u>A</u> ):			
7ァイル入力	•	詳細( <u>E</u> )	

図 3-16 ブレーク種別の選択ダイアログボックス

選択内容を以下に示します。

ブレーク種別	内容
[PCブレークポイント]	実行命令位置によるブレークポイント
[ブレークアクセス]	メモリ範囲のアクセスによるブレーク
[ブレークデータ]	メモリのデータ値によるブレーク
[ブレークレジスタ]	レジスタのデータ値によるブレーク
[ブレークシーケンス]	実行順序を指定したブレークポイント
[ブレークサイクル]	サイクル数によるブレーク

(2) ブレーク条件を設定する

[ブレーク種別の選択]ダイアログボックスでブレーク種別を選択後[詳細]をクリックすると、各ブレーク種別の条件を 設定するダイアログボックスを表示します。

(a) PC ブレークポイント

PCブレークポイン	ト条件の設定		<u>? ×</u>
アドレス( <u>A</u> ):	H'00000000	- 2	OK
回数( <u>C</u> ):	D'1		キャンセル

図 3-17 PC ブレークポイント条件の設定ダイアログボックス

PCブレークポイント条件を設定します。

[PCブレークポイント] 1,024個まで指定可能

[アドレス] ブレークする命令の位置
[回数] 指定位置の命令をフェッチする回数
(接頭辞省略時は10進入力、10進表示)
(1~16,383、省略すると1となります)



(b) ブレークアクセス

ブレーク	リアクセス条件	の設定		? ×
開始	アドレス( <u>B</u> ):	H'00000000	•	(ÖK
終了	アドレス( <u>E</u> ):	H'00000000	• 🔊	キャンセル
アクセ	2.7種別( <u>A</u> ):	リード/ライト	•	

図 3-18 ブレークアクセス条件の設定ダイアログボックス

#### ブレークアクセス条件を設定します。

[ブレークアクセス] 1,024 個まで指定可能

[開始アドレス] アクセスするとブレークするメモリの開始位置
 [終了アドレス] アクセスするとブレークするメモリの終了位置
 (省略すると開始位置のみが範囲となります)
 [アクセス種別] アクセス種別

【注】ストリング命令。および積和演算命令は最終データアクセスのみがブレークアクセスチェック対象となります。

(c) ブレークレジスタ

ブレークレジスタ条件	の設定	<u>?×</u>
レジスタ( <u>R</u> ):	RO	<u>OK</u>
オプション( <u>O</u> ):	─致	キャンセル
データ( <u>D</u> ):		
□ データマスク(型):	H'FFFFFFF	
サイズ( <u>S</u> ):	Long word	

図 3-19 ブレークレジスタ条件の設定ダイアログボックス

ブレークレジスタ条件を設定します。

[ブレークレジスタ] 1,024 個まで指定可能

,	
[レジスタ]	ブレーク条件を設定するレジスタ名
[オプション]	データの一致/不一致
[データ]	ブレーク条件となるデータ値
	(省略するとレジスタへ書き込むたびにブレークします)
[データマスク]	マスク条件(0を指定したビットがマスクされます)
[サイズ]	データのサイズ

【注】

- (1) ストリング操作命令、および積和演算命令は最終レジスタアクセスのみがブレークレジスタチェック対象となります
- (2) スタックポインタレジスタをブレークレジスタに設定した場合のチェック対象レジスタは以下のとおりです

ブレークレジスタ指定レジスタ	アクセスレジスタ	
	ISP	USP
"R0"		
"ISP"		×
"USP"	×	
:ブレークチェックする、×:	ブレーク	?チェックしなl



(d) ブレークシーケンス

ブレークシーケンフ	条件の設定	? ×
アドレス(1):	H'00000000	• 🔊 🕅 🕅
アドレス(2):		<ul> <li></li></ul>
アドレス( <u>3</u> ):		
アドレス( <u>4</u> ):	·	
アドレス( <u>5</u> ):	<u> </u>	
アドレス( <u>6</u> ):	<u> </u>	
アドレス(7):	<u> </u>	
アドレス( <u>8</u> ):		

図 3-20 ブレークシーケンス条件の設定ダイアログボックス

# ブレークシーケンス条件を設定します。

[ブレークシーケンス] 1組のみ指定可能

[アドレス(1)]~[アドレス(8)] ブレークの発生条件となる通過アドレス (8 ポイントすべてを設定する必要はありません)

(e) ブレークサイクル

ブレークサイクル条件	<u>? ×</u>	
サイクル(公):	H'1	ОК
┌回数(⊆):		キャンセル
⊙ すべて		
<ul> <li>回数指定:</li> </ul>		

図 3-21 ブレークサイクル条件の設定ダイアログボックス

ブレークサイクル条件を設定します。

1,024 個まで指定可能 [ブレークサイクル]

[サイクル] ブレーク判定を行なうサイクル数(1~H'FFFFFFFF) [サイクル]×nのサイクルで条件が一致します ただし、指定したサイクルと実際に条件が一致するサイクルは ずれることがあります

#### [回数] ブレークが成立する回数 条件が一致するごとにブレークが成立します [すべて] 条件が一致した回数が[回数指定]以下の時だけ [回数指定] ブレークが成立します (接頭辞省略時は16進入力、16進表示)(1~65,535)



(f) ブレークデータ

ブレークデータ条件の影	定	? ×
アドレス( <u>A</u> ):	H'0000000	<b>Б</b>
オプション( <u>0</u> ):	一致	キャンセル
データ1 ( <u>D</u> ):	H'0	
データ2(2):		
<ul> <li>データマスク(M):</li> </ul>	H'FFFFFFF	
サイズ( <u>S</u> ):	Long word	
符号( <u>G</u> ):	符号あり	

図 3-22 ブレークデータ条件の設定ダイアログボックス

- ブレークデータ条件を設定します。
- [ブレークデータ] 1,024 個まで指定可能

[アドレス]	ブレーク判定を行うメモリの位置				
[オプション]	判定方法				
	一致	メモリのデータと指定値([データ])が一致で成立			
	不一致	メモリのデータと指定値([データ])が不一致で成立			
	符号反転 <sup>*1</sup>	前回メモリ書き込み値と今回メモリ書き込み値で符号が反			
		転した場合に成立			
	差分 <sup>*1</sup>	前回メモリ書き込み値と今回メモリ書き込み値の差分が指			
		定値(「データ])を超えた場合に成立			
	GT(>)	メモリ書き込み値が指定値(「データ])より大で成立			
	LT(<)	メモリ書き込み値が指定値([データ])より小で成立			
	GE(>=)	メモリ書き込み値が指定値([データ])以上で成立			
	LE(<=)	メモリ書き込み値が指定値(「データ」)以下で成立			
	範囲内	メモリ書き込み値が指定値(「データ1]と「データ2])の範囲内			
		([データ1]<= メモリ書き込み値<=[データ2])			
	範囲外	メモリ書き込み値が指定値(「データ1]と「データ2])の範囲外			
		(メモリ書き込み値 <[データ 1]  [データ 2] < メモリ書き			
[データ 1]	ブレーク条件	-となるデータ値			
	データ比較方	「法が「範囲内」、および「範囲外」の場合はブレーク条件となる開			
	始データ値				
[データ 2]	ブレーク条件	となる終了データ値			
	データ比較方	法が「範囲内」、および「範囲外1の場合に有効となります			
[データマスク]	マスク条件()を指定したビットがマスクされます)				
	[符号反転] [差分]を除くデータ比較方法オプション時に有効となります				
[サイズ]	データのサイズ				
[符号]	データの符号				
	下記の場合に有効となります				
	・データ比較	(方法が(差分)の場合			
	・データ比較	た法が[GT(>)]、[LT(<)]、[GE(>=)]、[LE(<=)]、[範囲内]、お			

よび[範囲外]で、データサイズが[Byte]、[Word]、[Long word]の場合

\*1: [符号反転]、および[差分]は前回書き込み値と比較するため、リセット後、およびブレーク成立後1回目の判定は常に不成立となります。

【注】ストリング命令。および積和演算命令は最終データアクセスのみがブレークアクセスチェック対象となります。



(3) 動作種別を選択する

各ブレーク条件設定ダイアログボックスで条件設定後[OK]をクリックすると、再度[ブレーク種別の選択]ダイアログ ボックスが開きます。

[ブレーク種別の選択]ダイアログボックスの[動作種別]フィールドで動作種別を選択します。

ブレーク種別の選択			? ×
ブレーク種別( <u>B</u> ): PCブレークポイント	•	〕 ■美約田( <u>D</u> )	OK キャンセル
17アイル入力	•	言羊糸田( <u>E</u> )	

図 3-23 ブレーク種別の選択ダイアログボックス

選択内容を以下に示します。

 動作種別 内容
 [停止] 条件成立時にユーザプログラムの実行を停止します
 [ファイル入力] 条件成立時に指定ファイルから読み込んだデータを指定メモリへ書き込みます
 [ファイル出力] 条件成立時に指定メモリの内容を指定ファイルへ書き込みます
 [割り込み] 条件成立時に割り込み処理を行います
 [トレーストリガ] 条件成立時にトレース情報の取得を開始します
 イベントトリガによるトレース情報の取得が有効となっている場合のみ、トレース情報の 取得を開始します

(4) 動作内容を設定する

[ブレーク種別の選択]ダイアログボックスで動作種別を選択後[詳細]をクリックすると、[停止]、および[トレーストリガ]を除く各動作種別の内容を設定するダイアログボックスを表示します。

(a) ファイル入力

7	ァイル入力動作の詞	定		<u>? ×</u>
	λカファイル⊉: 		▼ ブラウズ	OK キャンセル
	入力先———			
	アドレス( <u>A</u> ):	H'00000000	- 🔊	
	データサイズ( <u>S</u> ):	1	•	
	データ数( <u>C</u> ):	D'1		

図 3-24 ファイル入力動作の設定ダイアログボックス

ファイル入力動作内容を設定します。

[ファイル入力] 条件成立時に指定ファイルから読み込んだデータを指定メモリへ書き込みます
 [入力ファイル] 読み込むデータファイルを指定します
 ファイルの終端まで読み込んだら先頭から繰り返して読み込みます
 [アドレス] データを書き込むメモリのアドレスを指定します
 [データサイズ] 読み込むデータ1個のサイズ(バイト数)を指定します(1/2/4/8)
 [データ数] 読み込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)

(b) ファイル出力

ファイル出力動作の言	定		? ×
出力ファイル(2):	□追記(2)	ブラウズ…	ОК
・ 出力元 アドレス(A):	-		
データサイズ( <u>S</u> ):	1		
データ数( <u>C</u> ):	D'1		

図 3-25 ファイル出力動作の設定ダイアログボックス

ファイル出力動作内容を設定します。

[ファイル出力]	条件成立時に指定	メモリの内容を指定ファイルへ書き込みます
	[出力ファイル]	書き込むデータファイルを指定します
	[追記]	既存のファイルを[出力ファイル]で指定した場合にファイル
		の最後に追加出力するかを指定します
	[アドレス]	データを読み出すメモリのアドレスを指定します
	[データサイズ]	書き込むデータ1個のサイズ(バイト数)を指定します(1/2/4/8)
	[データ数]	書き込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)

(c) 割り込み

割り込み動作の設定		<u>?</u> ×
割り込み種別1①	H'0	ОК
割り込み種別2(N):		キャンセル
優先順位(P):	H'0	

図 3-26 割り込み動作の設定ダイアログボックス

割り込み動作内容を設定します。

[割り込み] 条件成立時に割り込み処理を行います。詳細は「2.15 擬似割込み」を参照してください。
 [割り込み種別 1] 割り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)
 [優先順位] 割り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)(0~8、または 0~H'10)
 0~8の場合は 8、0~H'10 の場合は H'10 を指定すると高速割り込みとして扱います。

## (d) 留意事項

複数の[ファイル入力]で同一ファイルを指定した場合、ブレーク成立順にファイルからデータを読み込みます。複数の [ファイル出力]で同一ファイルを指定した場合、ブレーク成立順にファイルへデータを書き込みます。ただし、[ファイル 入力]と[ファイル出力]で同一ファイルを指定した場合は、最初に成立した動作のみが有効となります。

# 3.6.3 ブレークポイントの設定内容を変更する

変更したいブレークポイントを選択後ポップアップメニューから[編集…]を選択すると、[ブレーク種別の選択]ダイア ログボックスが開き、ブレーク条件を変更することができます。[編集…]メニューはブレークポイントを1個選択してい るときのみ有効となります。



### 3.6.4 ブレークポイントを有効にする

ブレークポイントを選択後ポップアップメニューから[有効]を選択すると、選択しているブレークポイントを有効にします。

### 3.6.5 ブレークポイントを無効にする

ブレークポイントを選択後ポップアップメニューから[無効]を選択すると、選択しているブレークポイントを無効にし ます。無効にした場合は、ブレークポイントはリストには残りますが、指定した条件が一致してもブレークは成立しませ ん。

#### 3.6.6 ブレークポイントを削除する

ブレークポイントを選択後ポップアップメニューから[削除]を選択すると、選択しているブレークポイントを削除しま す。ブレークポイントを削除しないで、詳細情報は保持したまま、条件が一致してもブレークを成立させないようにする には、[無効]オプションを使用します(「3.6.5 ブレークポイントを無効にする」参照)。

## 3.6.7 ブレークポイントをすべて削除する

ポップアップメニューから[すべて削除]を選択すると、すべてのブレークポイントを削除します。

## 3.6.8 ブレークポイントのソース行を表示する

ブレークポイントを選択後ポップアップメニューから[ソースファイル表示]を選択すると、ブレークポイントのある [ソース]または[逆アセンブリ]ウィンドウをオープンします。[ソースファイル表示]メニューはブレークポイントを1個選 択しているときのみ有効となります。

#### 3.6.9 入出力ファイルを閉じる

ブレークポイントを選択後ポップアップメニューから[ファイルを閉じる]を選択すると、選択した[ファイル入力] また は[ファイル出力]のデータファイルを閉じ、ファイル読み出し位置をリセットします。

#### 3.6.10 入出力ファイルをすべて閉じる

ポップアップメニューから[すべてのファイルを閉じる]を選択すると、すべての[ファイル入力] および[ファイル出力] のデータファイルを閉じ、ファイル読み出し位置をリセットします。

# 3.7 トレース情報を見る

シミュレータ・デバッガでは命令の実行結果をトレース情報として取得および表示することができます。 トレース情報は、[トレース]ウィンドウに表示します。トレース情報の取得条件は、[トレース取得]ダイアログボック スで設定します。

#### 3.7.1 トレースウィンドウを開く

[トレース]ウィンドウを開くには、[表示->コード->トレース]を選択するか、[トレース]ツールバーボタン<mark>早</mark>をクリッ クします。

#### 3.7.2 トレース情報取得条件を設定する

[トレース]ウィンドウを開いたら、トレース情報取得条件を設定します。 トレース情報取得条件は、[トレース取得]ダイアログボックスで設定します。 [トレース取得]ダイアログボックスを開くには、ポップアップメニューから[トレース設定...]を選択します。

トレ-	-ス取得			? ×
ŀ	レース機能(	D:	有効	<b>•</b>
ŀ	・レースバッファ	満杯時の動作(E):	続行	<b>_</b>
ŀ	レース容量(	_):	65536レコー	<b>ب</b>
耵	双得条件( <u>A</u> ):		すべて	▼
Г	- 	vk		
	Туре	Condition		追加( <u>D</u> )
				肖JI除(E)
				すべて削除(L)
				すべて有効( <u>N</u> )
				すべて無効( <u>S</u> )
			OK	キャンセル

図 3-27 トレース取得ダイアログボックス

本ダイアログボックスでは、トレース情報の取得条件を設定します。 [トレース機能] トレース情報の取得停止 [無効] [有効] トレース情報の取得開始 [トレースバッファ満杯時の動作] 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 [続行] [停止] トレース情報取得バッファが満杯になった場合、実行停止 [トレース容量] [65536]レコード トレースバッファのサイズは 64K レコード [131072]レコード トレースバッファのサイズは 128K レコード トレースバッファのサイズは 256K レコード [262144]レコード トレースバッファのサイズは 512K レコード [524288]レコード [1048576]レコード トレースバッファのサイズは 1M レコード [取得条件] [すべて] プログラム実行が停止するまでトレース情報を取得する トリガとなるイベントが成立する度に、トリガ発生前255 レコードとトリガ発生行、およびトリガ [イベントトリガ] 発生後 256 レコードの合計 512 レコードのトレース情報を取得する [トレースイベント] トレース情報の取得を開始するイベント情報を表示する 表示する項目は以下の通りです [Type] イベントの種別 [Condition] イベントの条件 [Type]欄のチェックボックスをチェックしたトレースイベントが有効となります [追加…] イベントを指定するダイアログボックスを表示します 指定したイベントを削除します [削除] すべてのイベントを削除します [すべて削除] [すべて有効] すべてのイベントを有効にします [すべて無効] すべてのイベントを無効にします

トレース取得ダイアログボックスの設定を変更した場合はトレース情報をクリアします。



指定した内容は、[OK]ボタンをクリックすることにより設定します。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

#### 3.7.3 トレースイベントを設定する

トレースイベントは、ブレーク条件を使用し、トレースイベント成立時の動作として成立したイベント前後のトレース情報を取得します。

トレースイベントは、[ブレーク種別の選択]ダイアログボックスで設定します。

[ブレーク種別の選択]ダイアログボックスを開くには、[トレース取得]ダイアログボックスの[追加]ボタンをクリック するか、[イベントポイント]ウィンドウの[Software event]タブ上でポップアップメニューから[設定...]を選択します。

トレースイベントの条件、条件成立時の動作の設定については、「3.6 シミュレータ・デバッガのブレークポイントを 使用する」を参照してください。

トレースイベントの条件を変更する場合は、[トレースイベント欄]で変更したいイベント条件をダブルクリックし、[ブ レーク種別の選択]ダイアログボックスを開きます。

#### 3.7.4 トレース情報を取得する

トレース情報の取得を開始した状態で命令を実行すると、トレースを取得できます。 取得したトレース情報は[トレース]ウィンドウに表示します。 トレース情報の表示は、バス表示、逆アセンブル表示、ソース表示とこれらの混在表示ができます。

(1) バス表示モード

ポップアップメニューから[表示モード -> BUS]を選択します。

- (a) すべて取得モード
  - シミュレーション開始からシミュレーション停止までのトレース情報を表示します。

🧆 トレース									×
• 🗸									
Range: -003	36672, 00000	100 File: 0	Oyole: -0035976  Address: FF	FF9071  Ti	me: 00:00:00.00	00.021.760			
PTR	Label	Address	Time Stamp	PSW	Instruction		Interrupt	Access Data	•
-0035976	_main	FFFF9071	00:00:00.000.021.760	0PUIC	ADD	#-30H, R0, R0	-	USP<-00001A74	-
-0035975		FFFF9074	00:00:00.000.021.770	0PUIC	MOY.L	#-00007BE4H,R5	-	R5<-FFFF841C	
-0035974		FFFF907A	00:00:00.000.021.780	OPUIC	SUB	#4H,R0	-	USP<-00001A70	
-0035973		FFFF907C	00:00:00.000.021.790	0PUIC	MOY.L	R5,[R0]	-	00001A70<-FFFF841C	
-0035972		FFFF907E	00:00:00.000.021.820	0PUIC	BSR.A	_printf	-	00001A6C<-FFFF9082	
-0035971	_printf	FFFF9349	00:00:00.000.021.830	0PUIC	MOY.L	#0H,R5	-	R5<-00000000	
-0035970		FFFF934B	00:00:00.000.021.840	0PUIC	PUSH.L	R5	-	00001A68<-00000000	
-0035969		FFFF934D	00:00:00.000.021.850	0PUI	ADD	#08H,R0,R4	-	R4<-00001A70	
-0035968		FFFF9350	00:00:00.000.021.860	0PUI	ADD	#7H,R4	-	R4<-00001A77	
-0035967		FFFF9352	00:00:00.000.021.870	OPUI	MOY.L	08H[R0],R3	-	R3<-FFFF841C	
-0035966		FFFF9354	00:00:00.000.021.880	0PUI	AND	#-04H,R4	-	R4<-00001A74	
-0035965		FFFF9357	00:00:00.000.021.890	0PUI	MOY.L	#000015A8H,R2	-	R2<-000015A8	
-0035964		FFFF935D	00:00:00.000.021.900	0PUI	MOY.L	#-00006CD6H,R1	-	R1<-FFFF932A	
-0035963		FFFF9363	00:00:00.000.021.930	0PUI	BSR.A	Printf	-	00001A64<-FFFF9367	
-0035962	Printf	FFFF96AC	00:00:00.000.021.970	0PUI	PUSHM	R6-R9	-	00001A60<-00000000	
-0035961		FFFF96AE	00:00:00.000.021.980	0PUIC	ADD	#-00A4H,R0,R0	-	USP<-000019B0	
-0035960		FFFF96B2	00:00:00.000.021.990	OPUIC	MOY.L	R2,98H[R0]	-	00001A48<-000015A8	•

図 3-28 すべて取得モードのトレースウィンドウ(バス表示モード)

#### 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)
[Label]	アドレスに対応するラベル(ラベルが設定されいる場合のみ表示します)
[Address]	命令アドレス
[Time Stamp]	累計命令実行時間(時:分:秒.ミリ秒.マイクロ秒.ナノ秒)
[PSW]	プロセッサステータスワード(PSW)の値をニモニックで表示
[Instruction]	命令ニモニック
[Interrupt]	割り込み(割り込み発生あり:"Interrupt"、割り込み発生なし:"-")
[Access Data]	データアクセス(転送先<-転送データの形式で表示) $^{*1}$

\*1:ストリング操作命令、および積和演算命令のデータアクセス表示は最終データのみとなります。



# (b) イベントトリガモード

イベント成立前後の情報を表示します。1回の表示は、1つのイベントで取得した512レコードです。表示する情報は、 トレースウィンドウのポップアップメニュー[トレースポイント->前のトレースポイントを表示]および[トレースポイン ト->次のトレースポイントを表示]で変更できます。シミュレーション停止後は、最も古いイベントのトレース情報を表示します。

	×۱۵−۶									
Ra										
No	. PTR Label	Address	Time Stamp	PSW	Instruction	ì	Interrupt	Access Data		
1	-0000008	FFFF803E	00:00:00.000.021.310	0I-S	PUSH.L	R1	-	00001B88<-FFFF8043		
1	-0000007	FFFF8040	00:00:00.000.021.370	OPUI	RTE		-	PC<-FFFF8043 ISP<-0	1	
1	-0000006	FFFF8043	00:00:00.000.021.380	0PUI	NOP		-			
1	-0000005	FFFF8044	00:00:00.000.021.410	0PUI	BSR.A	_main	-	00001A8C<-FFFF8048		
1	-0000004 _main	FFFF9042	00:00:00.000.021.420	OPUIC	ADD	#-30H,R0,R0	-	USP<-00001A5C		
1	-0000003	FFFF9045	00:00:00.000.021.430	OPUIC	MOV.L	#-00007BCCH,R5	-	R5<-FFFF8434		
1	-0000002	FFFF904B	00:00:00.000.021.440	OPUIC	SUB	#4H,R0	-	USP<-00001A58		
1	-0000001	FFFF904D	00:00:00.000.021.450	OPUIC	MOV.L	R5,[R0]	-	00001A58<-FFFF8434		
1	0000000	FFFF904F	00:00:00.000.021.480	OPUIC	BSR.A	_printf	-	00001A54<-FFFF9053		
1	0000001 _print	f FFFF9312	00:00:00.000.021.490	OPUIC	MOV.L	#0H,R5	-	R5<-00000000		
1	0000002	FFFF9314	00:00:00.000.021.500	OPUIC	PUSH.L	R5	-	00001A50<-00000000		
1	0000003	FFFF9316	00:00:00.000.021.510	0PUI	ADD	#08H,R0,R4	-	R4<-00001A58		
1	0000004	FFFF9319	00:00:00.000.021.520	0PUI	ADD	#7H,R4	-	R4<-00001A5F		
1	0000005	FFFF931B	00:00:00.000.021.530	0PUI	MOY.L	08H[R0],R3	-	R3<-FFFF8434		
1	0000006	FFFF931D	00:00:00.000.021.540	0PUI	AND	#-04H,R4	-	R4<-00001A5C		
1	0000007	FFFF9320	00:00:00.000.021.550	OPUI	MOV.L	#00001590H,R2	-	R2<-00001590		
1	0000008	FFFF9326	00:00:00.000.021.560	OPUI	MOV.L	#-00006D0BH,R1	-	R1<-FFFF92F5	-	

図 3-29 イベントトリガモードのトレースウィンドウ(バス表示モード)

#### 表示する項目は以下の通りです。

- \*1:ストリング操作命令、および積和演算命令のデータアクセス表示は最終データのみとなります。
- (2) 逆アセンブル表示モード

ポップアップメニューから[表示モード -> DIS]を選択します。 実行した命令を参照できます。

🧆 ኑレース						_	
• 🗸				r 🖸	9.0		
Range: -003	6672, 0000000	File:  Cycle: -0	0035976  Addres:	s: FFFF9071	Time: 00:00:00.000.02	1.760	
PTR	Label	Address	Object Code	Instruction	n	Time Stamp	
-0035976	_main	FFFF9071	7100D0	ADD	#-30H,R0,R0	00:00:00.000.021.760	
-0035975		FFFF9074	FB521C84FFFF	MOY.L	#-00007BE4H,R5	00:00:00.000.021.770	
-0035974		FFFF907A	6040	SUB	#4H,R0	00:00:00.000.021.780	
-0035973		FFFF907C	E305	MOY.L	R5,[R0]	00:00:00.000.021.790	
-0035972		FFFF907E	05CB0200	BSR.A	_printf	00:00:00.000.021.820	
-0035971	_printf	FFFF9349	6605	MOY.L	#0H, R5	00:00:00.000.021.830	
-0035970		FFFF934B	7EA5	PUSH.L	R5	00:00:00.000.021.840	
-0035969		FFFF934D	710408	ADD	#08H,R0,R4	00:00:00.000.021.850	
-0035968		FFFF9350	6274	ADD	#7H,R4	00:00:00.000.021.860	
-0035967		FFFF9352	A883	MOV.L	08H[R0],R3	00:00:00.000.021.870	
-0035966		FFFF9354	7524FC	AND	#-04H,R4	00:00:00.000.021.880	
-0035965		FFFF9357	FB22A8150000	MOY.L	#000015A8H,R2	00:00:00.000.021.890	
-0035964		FFFF935D	FB122A93FFFF	MOY.L	#-00006CD6H,R1	00:00:00.000.021.900	
-0035963		FFFF9363	05490300	BSR.A	Printf	00:00:00.000.021.930	
-0035962	Printf	FFFF96AC	6E69	PUSHM	R6-R9	00:00:00.000.021.970	
-0035961		FFFF96AE	72005CFF	ADD	#-00A4H,R0,R0	00:00:00.000.021.980	
-0035960		FFFF96B2	E70226	MOY.L	R2,98H[R0]	00:00:00.000.021.990	

図 3-30 トレースウィンドウ(逆アセンブルモード)



#### (3) ソース表示モード

ポップアップメニューから[表示モード -> SRC]を選択します。 ソースプログラムの実行経路を参照できます。

実行経路は、現在のトレースサイクルから順方向、または逆方向にトレースデータ内をソースステップして確認できます。

🐠 ዞレース				<b>I</b> ×
∎≣ <b>∀</b>	36672. 00000	<b>Z Z</b>	t Te fe	
Line	Address	Now	Source	
000022	FFFF9071	$\rightarrow$	void main(void)	
000023			{	
000024			long a[10];	
000025			long j;	
000026			int i;	
000027				
000028	FFFF9074	-	printf("### Data Input ###¥n");	
000029	55550004		6 ( 1-0- 1/10- 11) )[	
000030	FFFF9084	-	for( 1=0; 1<10; 1++ ){	
000031	FFFF8080	-	j = rand(j;	
000032	FFFF3033	-		
000033	FFFFJUAU		jj, 1	
000035	EEEE9048	-	a[i] = i•	
000036	EFEE90B2	-	printf("a[%d]=%ld¥n",i.a[i]):	
000037	11110002		}	
000038	FFFF90E4	-	sort(a);	<b>-</b>
		1	•••	

図 3-31 トレースウィンドウ(ソース表示モード)

(4) 混在表示モード

バス表示、逆アセンブル表示、ソース表示の混在表示ができます。

ポップアップメニューの[表示モード -> BUS]を選択した後、[表示モード -> DIS]を選択すると、バスと逆アセンブルの混在表示ができます。

同様の方法で、バスとソース、逆アセンブルとソース、バスと逆アセンブルとソースの混在表示ができます。 バスと逆アセンブルの混在表示にした後、バス表示のみの表示に戻すには、再度ポップアップメニューの[表示 -> DIS] を選択することでバス表示になります。

🐠 ኑレース									l ×		
Range: -00	Range: -0036672, 0000000 File: Cycle: -0035976 Address: FFFF9071 Time: 00:00:00.000.021.760										
PTR	Label	Address	Time Stamp	PSW	Instruction		Interrupt	Access Data			
-0035976	FFFF9071 _main	_main FFFF9071	ADD #-30H, R0, R0 00:00:00.000.021.760	OPUIC	ADD	#-30H,R0,R0	-	USP<-00001A74			
-0035975	FFFF0074	FFFF9074	00:00:00.000.021.770	OPUIC	MOV.L	#-00007BE4H,R5	-	R5<-FFFF841C			
	FFFF907A		SUB #4H, RO								
-0035974	CCCC0070	FFFF907A	00:00:00.000.021.780	OPUIC	SUB	#4H,R0	-	USP<-00001A70			
-0035973	FEEE907E	FFFF907C	00:00:00.000.021.790	OPUIC	MOV.L	R5,[R0]	-	00001A70<-FFFF841C			
-0035972	11110012	FFFF907E	00:00:00.000.021.820	0PUIC	BSR.A	_printf	-	00001A6C<-FFFF9082			
	FFFF9349	_printf	MOV.L #OH,R5								
-0035971	_printf	FFFF9349	00:00:00.000.021.830	OPUIC	MOV.L	<b>#</b> 0H, R5	-	R5<-00000000			
-0035970	FFFF004D	FFFF934B	00:00:00.000.021.840	0PUIC	PUSH.L	R5	-	00001A68<-00000000			
	FFFF934D		ADD #08H,R0,R4								
-0035969	FFFF0050	FFFF934D	00:00:00.000.021.850	OPUI	ADD	#08H,R0,R4	-	R4<-00001A70			
	LLLL99900		AUU #/H,R4								

図 3-32 トレースウィンドウ(混在表示モード)

# 3.7.5 トレース情報を検索する

トレース情報を検索するには[検索]ダイアログボックスを使用します。 [検索]ダイアログボックスを開くには、ポップアップメニューの[検索 -> 検索...]を選択します。

検索		?×
組合せ( <u>C</u> ):	検索項目:	
PTR Address Time Stamp Instruction Interrupt	トレースサイクル(Y): 「範囲指定(B) 0 - 0 「指定条件以外(X)	前方検索(火) 後方検索(E)
, 検索設定内容(E):		
		新規(₩)
		削除( <u>D</u> )
		すべて削除( <u>L</u> )
履歴( <u>0</u> ):		
		追加( <u>A</u> )
I		閉じる

図 3-33 検索ダイアログボックス

[組み合わせ]欄で、検索の対象とする条件を選択して、チェックボックスをチェックしてください。

[検索項目]欄で、選択した条件に対応する詳細項目を指定することができます。

[組み合わせ]欄で複数の条件を選択した場合は、それぞれの条件について詳細項目を指定してください。検索対象は複数条件の AND となります。

トレース検索の対象項目を以下に示します。

項目	内容	検索条件
[PTR]	トレースバッファ内ポインタ	数値指定(10進数)
		範囲指定可
		指定条件以外の検索可
[Address]	命令のアドレス	数値指定(16進数)
		範囲指定可
		指定条件以外の検索可
[Time Stamp]	累計命令実行時間	時間単位ごとのエディットボックスで指定
		範囲指定可
		指定条件以外の検索可
[Instruction]	命令ニモニック	文字列指定
		指定条件以外の検索可
[Interrupt]	割り込み発生	固定文字列"Interrupt"を検索
		指定条件以外の検索可

設定した検索条件は、[検索設定内容]リストボックスに表示されます。 検索条件を設定した後、[後方検索]ボタンまたは[前方検索]ボタンのクリックで検索を開始します。

検索の結果一致するレコードが見つかった場合は、当該レコードを強調表示します。

一致するレコードが見つからなかった場合は、メッセージダイアログボックスを表示します。

一致するレコードが見つかった場合は、ポップアップメニューで[後方検索]または[前方検索]を選択すると、次のレコードを検索できます。



### 3.7.6 トレース情報のフィルタリング

フィルタ機能を利用して取得したトレース情報から必要なレコードのみを抽出することができます。フィルタ機能を 使用するには[トレース]ウィンドウのポップアップメニューから[オートフィルタ]を有効にします。[オートフィルタ]を有 効にすると各カラムにオートフィルタ矢印 I を表示します。矢印 I をクリックしてドロップダウンリストから [Option...]を選択するとフィルタリングする条件を選択するオプションダイアログボックスを表示します。 フィルタリング可能な項目とフィルタリング条件はトレースレコード検索対象項目、検索条件と同じです。

【注】 イベントトリガ指定時は、フィルタ機能の表示は使用できません。

#### 3.7.7 トレース情報をクリアする

トレース情報はトレース情報取得後に命令シミュレーションを再実行するとクリアされます。

#### 3.7.8 トレース情報をファイルに保存する

[トレース]ウィンドウに表示しているトレース情報をテキスト形式で保存します。バイナリ形式での保存はできません。 トレース情報をファイルに保存するには、ポップアップメニューから[ファイル -> 保存…]を選択します。

[名前を付けて保存]ファイルダイアログボックスを表示します。トレースバッファの内容をテキストファイルとして保存します。保存する範囲を、[開始 – 終了サイクル]によって指定することができます。このファイルはトレースバッファ に再ロードできないことに注意してください。

# 3.7.9 ソースファイルを表示する

トレースレコードに対応するソースファイルを[エディタ]ウィンドウに表示するには、ソース表示モードとし、ポップ アップメニューから[ファイル -> ソースファイル編集]を選択します。

トレースウィンドウのソース表示モードで別のソースファイルを表示するには、[ソースファイル表示]ダイアログボックスを使用します。

[ソースファイル表示]ダイアログボックスを開くには、ポップアップメニューの[ファイル -> ソースファイル表示]を 選択します。

ソースファイル表示		? ×	:
ソースファイル( <u>S</u> ) 関数( <u>F</u> ):	Tutorial.c		
main sort change			
	OK	キャンセル	

図 3-34 ソースファイル表示ダイアログボックス

本ダイアログボックスは、トレースウィンドウに表示するソースファイルを選択します。条件設定後、[OK]ボタンを クリックすると、トレースウィンドウにソースファイルを表示し、選択した関数の先頭を強調表示します。

# 3.7.10 タイムスタンプの表示を切り替える

[トレース]ウィンドウに表示するタイムスタンプを絶対時間、差分時間、および相対時間に切り替えることができます。 初期状態では絶対時間で表示します。 ● 絶対時間

ポップアップメニューから[時間表示 -> 絶対時間]を選択するか、[絶対時間]ツールバーボタン 🔯 をクリック します。

- 相対時間

ポップアップメニューから[時間表示 -> 相対時間]を選択するか、[相対時間]ツールバーボタン <u>(</u>をクリックします。

#### 3.7.11 関数実行履歴を表示する

取得したトレース情報から関数実行履歴を表示することができます。関数実行履歴を表示するには、ポップアップメ ニューから[関数実行履歴表示 -> 関数実行履歴表示]を選択するか、[関数実行履歴表示]ツールバーボタン **た**をクリック します。

ポップアップメニューから[実行履歴解析]を選択するか、[実行履歴解析]ツールバーボタン ᢇ をクリックすると、トレース結果の最後尾から解析を開始し、結果をツリー構造で表示します。

🧆 トレース							
PowerON_Reset     PowerON_Reset     PowerON_Reset    INITSCT    INITSCT    INITSCT    INITSCT	PC> (FFF80 set_PC> (FFF (FFF8388)) LIB (FFFF8288) LIB (FFFF8288) <- FFF9384) <- FFF9384) <- FFF9384) <- FFF9384) <- FFF9384) <-	Image: Control of the second	* Q Q	Q			
	PPPP33437 (-	- FFFF300F	FFF0071 T	···· 00.00.00 00	20.001.760		<b>▼</b>
PTR         Label           PTR         Label           -0035376        main           -0035377        main           -0035373        main           -0035373        main           -0035373        main           -0035373        minn           -0035370        minn           -0035370        minn           -0035367        minn           -0035968        minn           -0035967        0035967           -0035965        0035965	Add File: C Address FFF9071 FFFF9074 FFFF9076 FFFF907C FFFF9348 FFFF9348 FFFF9348 FFFF9352 FFFF9352 FFFF9354 FFFF9357	ycle:	PSW           0PUIC           0PUIC	Instruction ADD MOV.L SUB MOV.L BSR.A MOV.L PUSH.L ADD ADD MOV.L AND MOV.L	<b>#-30H.R0.R0</b> <b>#-00007BE4H.R5</b> <b>#4H.R0</b> <b>#55.[R0]</b> printf <b>#0H.R5</b> <b>#08H.R0,R4</b> <b>#7H.R4</b> <b>08H[R0],R3</b> <b>#-04H.R4</b> <b>#00015A8H.R2</b>	Interrupt	Access Data USP<-00001A74 R5<-FFF841C 00001A70 00001A7000001A6C<-FFFF9082 R5<-00000000 00001A8C<-FFF9082 R5<-00000000 R4<-00001A70 R4<-00001A77 R4<-00001A74 R2<-000015A8

図 3-35 トレースウィンドウ(関数実行履歴表示)

下段のウィンドウには上段ウィンドウで選択された関数のレコードからトレース結果を表示します。

【注】 イベントトリガ指定時は、関数実行履歴の表示は使用できません。



# 3.8 プロファイル情報を見る

プロファイル機能は、アプリケーションプログラムの実行パフォーマンスを関数単位に測定します。アプリケーション プログラム中の性能劣化の原因となっている場所および要因を調査することができます。

HEWはプロファイルデータの参照方法、参照目的に応じて、3つのウィンドウでプロファイル測定結果を表示します。

# 3.8.1 スタック情報ファイル

プロファイル機能は、最適化リンカ(Ver.7.0以降)が出力するスタック情報ファイル(拡張子".SNI")を読み込むことが できます。このファイルには、ソースファイル上の(静的な)関数呼び出し関係の情報が入っています。HEWがスタック情 報ファイルを読み込むことで、ユーザアプリケーションが未実行(プロファイルデータの測定を行う前)でも、関数の呼 び出し関係を表示できるようになります。(但し、[プロファイル]ウィンドウのポップアップメニューで [表示設定->未実行関数を表示しない]をチェックしている場合を除きます。)

HEWがスタック情報ファイルを読み込まない場合、プロファイル機能で表示するデータは、プロファイルデータ測定 中に実行した関数についてのみになります。

リンカでスタック情報ファイルを生成するには、[Standard Toolchain]ダイアログボックスの[最適化リンカ]タブで[カテゴリ]リストボックスを[その他]に指定し、[スタック情報ファイル(sni)出力]チェックボックスをチェックしてください。

RX Standard Toolchain	? ×
JY74%b-%ay:         SimDebug_RX600         Image: All Loaded Projects         Image: Image: Image: All Loaded Projects         Image: Image	アセソフラ 最適化リンカ 標準ライフラリ CPU 全般         カテゴリ(Y):       その他         その他のオフション(M):          SPUコードを終端に出力          プメタック情報ファイル(sn()出力)          デバッグ情報販圧縮          コーザ指定オブション(U):       アブソリュート/リロケータブル/ライブラリー         ク       アブソリュート/リロケータブル/ライブラリー              アブソリュート/リロケータブル/ライブラリー                      アブソリュート/リロケータブル/ライブラリー <td< td=""></td<>
	OK キャンセル

図 3-36 Standard Toolchain ダイアログボックス(1)



# 3.8.2 スタック情報ファイルのロード

スタック情報ファイルを読み込むかどうかは、ロードモジュールロード時に表示する、確認のメッセージボックスで指 定できます。メッセージボックスの[OK]ボタンをクリックするとスタック情報ファイルをロードします。

確認のメッセージボックスは、次の場合に表示します。

- スタック情報ファイルが存在する時
- [オプション]ダイアログボックス(メインメニューの[基本設定->オプション...]を選択すると開きます)の[確認]タブ(図 3-37)で[スタック情報ファイルをロードします (SNIファイル)]チェックボックスをチェックしている場合

オプション	<u>? ×</u>
ビルド   エディタ   デバッグ   ワークスペース   確認   ネットワーク	
□ 確認ダイアログの表示:	
<ul> <li></li></ul>	▲ <u>すべて設定(S)</u> <u>すべて解除(C)</u>
<ul> <li>✓コマンドバッチファイルは存在しません。</li> <li>✓スタック情報ファイルをロードします (SNIファイル)</li> <li>✓すべてのラベル削除</li> <li>✓セーブしないで書き込み不可の属性のセッションを閉じます。</li> <li>✓ターゲット初期化</li> </ul>	
<ul> <li>●ダウンロードモジュールの更新</li> <li>●テンプレート削除</li> <li>●パフォーマンス解析範囲の削除</li> <li>●ビルド後ダウンロード</li> <li>□プログラムアンロード</li> </ul>	
□プログラムロード □プログラムをダウンロードせずに実行 □プロステムをダウンロードせずに実行	
	OK キャンセル

図 3-37 オプションダイアログボックス

# 3.8.3 プロファイルを有効にする

[表示->パフォーマンス->プロファイル]を選択し、[プロファイル]ウィンドウをオープンします。

[プロファイル]ウィンドウのポップアップメニューで[有効]メニューオプションを選択します(メニューにチェック マークが付きます)。

#### 3.8.4 測定方法を指定する

プロファイルデータの測定時に、関数呼び出しをトレースするかどうかを指定できます。関数呼び出しをトレースする と、ユーザプログラム実行時の関数呼び出し関係をツリー形式で表示できるようになります。関数呼び出しをトレースし ないと、関数呼び出し関係を表示できませんが、プロファイルデータの測定時間を短縮することができます。

関数呼び出しをトレースしないようにするためには、[プロファイル]ウィンドウのポップアップメニュー[関数呼び出し をトレースしない]を選択します。(メニューにチェックマークが付きます。)

また、OSによるタスクスイッチなど、通常の方法以外で関数を呼び出しているプログラムの場合、関数呼び出しを正 しく表示できない場合がありますので、関数呼び出しをトレースせずにプロファイルデータを測定してください。

# 3.8.5 ユーザプログラムを実行し結果を確認する

ユーザプログラムを実行し、停止すると[プロファイル]ウィンドウに測定結果を表示します。 [プロファイル]ウィンドウには、[List]シートと[Tree]シートがあります。

#### 3.8.6 List シート

関数とグローバル変数をリスト表示し、各関数/変数のプロファイルデータを表示します。

#JD771ル								
🗈 - 🗧 🎆 Show Functions/Variables 💽 🐹 🖉								
Function/Variable	F/V	Address	Size	Times	Cycle	Ext mem	I/O area	Int mem 🔺
_main	F	FFFF8C98	H'00000D1	1	738	0	0	271
sort	F	FFFF8D69	H'000000FD	1	1860	0	0	774
_change	F	FFFF8E66	H'0000006A	1	433	0	0	166
_charput	F	FFFF8EDO	н'00000000	249	2739	0	0	747
INIT_IOLIB	F	FFFF8F10	H'0000011F	1	89	0	0	31
CLOSEALL	F	FFFF902F	H'0000004F	1	510	0	0	144
_open	F	FFFF907E	н'0000009в	3	203	0	0	39
_close	F	FFFF9119	н'00000009	3	21	0	0	6
_write	F	FFFF9122	н'00000096	249	16932	0	0	5478
_freopen	F	FFFF92A9	H'0000002E	3	96	0	0	60
fclose	F	FFFF92D7	н'00000053	3	126	0	0	39
FFFF932A	F	FFFF932A	н'00000000	183	3700	0	0	2013
_printf	F	FFFF9349	н'00000021	22	384	0	0	176
_rand	F	FFFF936A	H'0000001C	10	110	0	0	30
INITSCT	F	FFFF9386	н'00000000	1	987	0	0	32
fwrite	F	FFFF93D0	H'000000CF	183	24459	0	0	6348 🗵
List / Tree /								

図 3-38 List シート

カラムヘッダをクリックすると、アルファベットまたは数値の昇降順にソートして表示します。 [Function/Variable]列または[Address]列をクリックすると、該当するアドレスに対応したソースプログラムを表示します。

ウィンドウ内でマウスの右ボタンをクリックするとポップアップメニューを表示します。このポップアップメニュー については「3.8.7 Treeシート」を参照してください。



# 3.8.7 Tree シート

関数の呼び出し関係を表示し、各呼び出し位置におけるプロファイルデータを表示します。 [Tree]シートは、[プロファイル]ウィンドウのポップアップメニュー[関数呼び出しをトレースしない]をチェックしてい ない時のみ有効です。

♥プロファイル								
🗈 🔫 Ena Show Functions/Variables	- 35	7						
Function	Address	Size	Stack Size	Times	Cycle	Ext mem	I/O area	Int mem 📥
_Expep_BRK	FFFFB1EC	H'00000004	н'00000000	1	0	0	0	2
CLOSEALL	FFFF902F	H'0000004F	н'00000000	1	510	0	0	144
INIT_IOLIB	FFFF8F10	H'0000011F	н'00000000	1	89	0	0	31
freopen	FFFF92A9	H'000002E	н'00000000	3	96	0	0	60
□ _fclose	FFFF92D7	н'00000053	н'00000000	3	126	0	0	39
fflush	FFFF949F	H'0000007E	н'00000000	3	57	0	0	12
Fofree	FFFF965B	H'00000051	н'00000000	3	66	0	0	36
_close	FFFF9119	н'00000009	н'00000000	3	21	0	0	6
+Foprep	FFFF951D	H'000000E8	н'00000000	3	415	0	0	87
main	FFFF8C98	H'000000D1	н'00000000	1	738	0	0	271
printf	FFFF9349	H'00000021	н'00000000	22	384	0	0	176
rand	FFFF936A	H'000001C	н'00000000	10	110	0	0	30
	FFFF8E66	H'0000006A	н'00000000	1	433	0	0	166
sort	FFFF8D69	H'000000FD	H'00000000	1	1860	0	0	774 🚽
List Tree								

図 3-39 Tree シート

[Function]列の関数をダブルクリックすると、ツリー構造を拡張または収縮表示します。また、'+' / '-'キーでも拡張 / 収 縮表示することができます。[Address]列をダブルクリックすると、該当するアドレスに対応したソースプログラムを表示 します。

ウィンドウ内でマウスの右ボタンをクリックするとポップアップメニューを表示します。このメニューは以下のオプ ションを含みます。

(1) ソースファイル表示

選択している行の該当アドレスに対応したソースプログラムまたは逆アセンブルを表示します。

(2) チャート表示

選択している行の関数に着目した[プロファイルチャート]ウィンドウを表示します。

(3) 有効

プロファイルデータ測定のオン・オフを切り替えます。プロファイルデータ測定がオンのとき、メニューテキストの 左にチェックマークを表示します。

(4) 関数呼び出しをトレースしない

本メニューをチェックすると、プロファイルデータ測定時に関数呼び出しをトレースしません。例えば、OSのタスク スイッチのように通常の方法以外で関数が呼び出されるプログラムのデータを測定する場合に使用します。

[プロファイル]ウィンドウの[Tree]シートで関数呼び出し関係を表示するためには、本メニューをチェックせずにプロファイルデータを測定してください。また、測定結果のプロファイル情報ファイルを使用して、最適化リンケージエディタによる最適化を行う場合も、本メニューをチェックしないでください。

(5) 検索...

[Function]列の文字列を検索する[テキスト検索]ダイアログボックスを表示します。検索したい文字列をエディットボックスに入力し、[次を検索]ボタンまたは、'Enter'キーを入力すると、検索を開始します。

(6) データ検索...

[データ検索]ダイアログボックスを表示します。



図 3-40 データ検索ダイアログボックス

[カラム]コンボボックスで検索カラムを、[検索データ]グループで検索方向を設定し、[次を検索]ボタンまたは、'Enter' キーを入力すると、検索を開始します。また、連続して[次を検索]ボタンまたは'Enter'キーを入力すると、次に大きいデー タ(最小値の場合は小さいデータ)を検索します。

(7) データクリア

関数呼び出し回数のカウントおよびプロファイルデータをクリアします。[プロファイル]ウィンドウの[List]シートおよび[プロファイルチャート]ウィンドウのデータもクリアします。

(8) プロファイル情報の保存...

[プロファイル情報の保存]ダイアログボックスを表示します。プロファイル結果をプロファイル情報ファイル(拡張子は".pro")に保存します。

#### (9) テキスト形式で保存...

[プロファイルデータをテキスト形式で保存]ダイアログボックスを表示します。表示している状態をテキストファイル に保存します。

#### (10) 表示設定

このメニューには下記サブメニューがあります。(以下の説明には[List]シートのみのメニューも含みます)

(a) 関数と変数を表示

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

(b) 関数を表示

[Function/Variable]列で、関数のみを表示します。

(c) 変数を表示

[Function/Variable]列で、グローバル変数のみを表示します。

(d) 未実行関数を表示しない

実行した関数のみ表示することができます。最適化リンケージエディタが出力するスタック使用量情報ファイル(拡張子:sni)がロードモジュールと同一ディレクトリに存在しない場合、このチェックボックスの設定に関わらず、実行関数のみ表示します。

(e) 子関数の実行結果を含んで表示

表示するプロファイルデータに、関数内で呼び出した子関数のプロファイルデータを含めるかどうかを設定します。

(11) プロパティ...

本シミュレータ・デバッガでは使用できません。

# 3.8.8 プロファイルチャートウィンドウ

[プロファイルチャート]ウィンドウは、特定の関数に着目した関数の呼び出し関係を表示します。本ウィンドウは、着 目する関数を中心に表示し、その左側には着目した関数を呼び出した関数、右側には、着目している関数が呼び出した関 数を、それぞれ表示します。また、各呼び出しを行った回数も表示します。



図 3-41 プロファイルチャートウィンドウ

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。

(1) ソースファイル表示

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。

(2) チャート表示

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[プロファイル-チャート]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。

(3) 有効

プロファイルデータ収集の有効 / 無効を切り替えます。プロファイルデータ測定が有効のとき、メニューテキストの左に チェックマークを表示します。

(4) データクリア

関数呼び出し回数のカウントをクリアします。[プロファイル]ウィンドウの[List]シートおよび[Tree]シートのデータも クリアします。

(5) チャートウィンドウを複数開く

[プロファイル-チャート]ウィンドウを表示する際、既に[プロファイル-チャート]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。

(6) プロファイル情報の保存...

[プロファイル情報の保存]ダイアログボックスを表示します。プロファイル結果をプロファイル情報ファイル(拡張子は".pro")に保存します。最適化リンケージエディタは、プロファイル情報を元に、ユーザプログラムの最適化を行うことが出来ます。プロファイル情報を使用した最適化についての詳細は、最適化リンケージエディタのマニュアルを参照してください。

(7) 拡大

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。



(8) 縮小

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。

### 3.8.9 表示データの種類および用途

プロファイル機能から下記情報を得ることができます。

(1) Address

関数を配置しているメモリ上の位置を知ることができます。アドレス順にソート表示することにより、メモリ上の配置 イメージで関数とグローバル変数を並べることができます。

(2) Size

サイズ順にソート表示すれば、サイズが小さくて頻繁に呼び出している関数を見つけることができます。そのような関数があればinline関数にすることで、関数呼び出しのオーバヘッドを減らせる場合があります。

#### (3) Stack Size

関数呼び出しのネストが深い場合、関数呼び出し経路をたどり、その経路上の全関数のスタックサイズを合計することで、おおよそのスタック使用量を見積もれます。

(4) Times

呼び出し(アクセス)回数順にソート表示すれば、頻繁に呼び出している関数や頻繁にアクセスしている変数を容易に 調べることができます。

- (5) プロファイルデータ
  - [Cycle] (実行サイクル数)
  - [Ext\_mem] (外部メモリアクセス回数)
  - [I/O\_area](内蔵 I/O アクセス回数)
  - [Int\_mem] (内部メモリアクセス回数)

実行サイクル数は、当該関数コール命令実行時の累計実行サイクル数と当該関数からのリターン命令実行時の累計実 行サイクル数の差から求めています。

ストリング操作命令、および積和演算命令のデータアクセス回数は最終データ1回のみとなります。

#### 3.8.10 プロファイル情報ファイルを作成する

プロファイル情報ファイルを作成する場合は、Pop-upメニューの[プロファイル情報の保存...]メニューオプションを選択します。[プロファイル情報の保存]ダイアログボックスを表示します。ファイル名を選択して[保存]ボタンを押すと、 選択したファイルにプロファイル情報を書きこみます。[全て保存]ボタンを押すと、全てのファイルにプロファイル情報 を書きこみます。



プ	コファイル情報の保存		? ×
	プロファイル情報ファイ. プログラムファイル	ル( <u>P)</u> プロファイル情報ファイル	閉じる( <u>C</u> )
	Tutorial	C:¥Workspace_Evaluation_RX¥Tutorial¥Tutc	保存( <u>5</u> )
			全て保存( <u>A</u> )
			参照( <u>B</u> )
	•		

図 3-42 プロファイル情報の保存ダイアログボックス

- 3.8.11 注意事項
  - (1)アプリケーションプログラムの実行サイクル数をプロファイル機能で測定した場合のデータには誤差があります。プロファイル機能では、アプリケーションプログラム全体の中で各関数が占める実行時間の比率を調べることができますが、より厳密に関数の実行サイクル数を測定したい場合には、パフォーマンス解析機能を使用してください。
  - (2) デバッグ情報が無いロードモジュールでプロファイル情報の測定を行った場合、関数名を表示しない場合があ ります。
  - (3) スタック情報ファイル(拡張子".SNI")はロードモジュールファイル(拡張子".ABS")と同一のディレクトリに置い てある必要があります。
  - (4) 測定結果の蓄積はできません。
  - (5) 測定結果の編集はできません。

# 3.9 パフォーマンスを解析する

関数名を指定してパフォーマンス解析する場合は、[パフォーマンス解析]ウィンドウを使用します。

# 3.9.1 パフォーマンス解析ウィンドウを開く

[パフォーマンス解析]ウィンドウを開くには、[表示->パフォーマンス->パフォーマンス解析]を選択するか、[パフォーマンス解析]ツールバーボタン E をクリックします。

≪パフォーマ	マンス解析				<u> –  –  ×</u>
•_ >_ >	Re Ena				
Index	Function	Cycle	Count	Histogram	
0	main	738	1	0%	
1	sort	1870	1	1%	
2	change	425	1	0%	

図 3-43 パフォーマンス解析ウィンドウ

本ウィンドウは、指定関数ごとの実行サイクル数を表示します。



実行サイクル数は、以下の計算で求めています。

実行サイクル数 = 指定関数からのリターン命令実行時累計実行サイクル数 - 指定関数コール命令実行時累計実行サ イクル数

表示する項目は以下の通りです。

[Index]	設定条件のインデックス番号
[Function]	測定対象の関数名(または関数の開始アドレス)
[Cycle]	当該関数の累計実行サイクル数
[Count]	当該関数の累計呼び出し回数
[Histogram]	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
	パーセンテージとヒストグラム形式で表示

#### 3.9.2 評価関数を設定する

[パフォーマンス解析]ウィンドウを開いたら、評価する関数を設定します。ポップアップメニューから[範囲の追加...] を選択すると、[パフォーマンスオプション]ダイアログボックスが開きます。また、'Insert'キーでも[パフォーマンスオプ ション]ダイアログボックスを開くことができます。

パフォーマンスオプ	ション			?×
関数名( <u>F</u> ):				ОК
			- 🔊	キャンセル
		 		<u> </u>

図 3-44 パフォーマンスオプションダイアログボックス

本ダイアログボックスでは、性能評価する関数(ラベルも可)を設定します。評価関数は255個まで設定可能です。

[OK]ボタンをクリックすることにより、評価関数を設定します。[キャンセル]ボタンをクリックすると、設定しないで ダイアログボックスを閉じます。

設定した評価関数を選択後、ポップアップメニューの[範囲の編集]を選択すると[パフォーマンスオプション]ダイアロ グボックスを表示して、評価関数を変更できます。また、'Enter'キーでも[パフォーマンスオプション]ダイアログボック スを開くことができます。

#### 3.9.3 データ収集を開始する

ポップアップメニューの[有効]をチェックして、プログラムを実行するとパフォーマンスデータを収集できます。

#### 3.9.4 データをリセットする

ポップアップメニューから[リセット]を選択すると、現在のプログラムのパフォーマンスデータをクリアします。

#### 3.9.5 評価関数を削除する

評価関数を選択した状態で、ポップアップメニューから[範囲の削除]を選択すると、選択された評価関数を削除し、他の範囲のデータを再計算します。また、'Delete'キーでも評価関数を削除することができます。

### 3.9.6 すべての評価関数を削除する

ポップアップメニューから[全ての範囲を削除]を選択すると、現在の評価関数をすべて削除し、パフォーマンスデータ をクリアします。

#### 3.9.7 現在の表示内容をセーブする

現在ウィンドウに表示している内容をテキストファイルにセーブすることが出来ます。ポップアップメニューから [ファイルに保存...]を選択してください。

# 3.10 コードカバレジを測定する

[カバレジ]ウィンドウは、ユーザが指定したアドレス範囲についてコードカバレジ情報(C0 カバレジおよび C1 カバレ ジ)を収集し、結果を表示します。

3.10.1 カバレジウィンドウを開く

[表示->コード->カバレジ…]を選択するか[カバレジ]ツールバーボタン 🔀 をクリックすると、[カバレジの設定]ダイア ログボックスが開きます。

カバレジの設定	? ×
オブション ● 新しいカバレジ測定範囲(小) ● 先頭アドレス(S): 00000000 ▼ 終了アドレス(E): 00000000 ▼ ◎ ● ファイル ● ● 医()…	<u>Q</u> K キャンセル( <u>C</u> )
<ul> <li>○ 最近使ったカバレジ情報ファイル(R)</li> <li>▼</li> <li>■</li> <li>■</li> <li>■</li> <li>■</li> </ul>	

図 3-45 カバレジの設定ダイアログボックス

[カバレジの設定]ダイアログボックスでは、カバレジ測定範囲を指定します。 新たなカバレジ範囲を設定するには、以下に示す2通りの方法があります。

• [新しいカバレジ測定範囲]で開始アドレスおよび終了アドレスを指定する 指定項目は以下のとおりです。

 [先頭アドレス]
 カバレジ情報表示の開始アドレスを指定します。(接頭辞省略時は 16 進で入力)

 [終了アドレス]
 カバレジ情報表示の終了アドレスを指定します。(接頭辞省略時は 16 進で入力)

• [新しいカバレジ測定範囲]でファイルを指定する

指定項目は以下のとおりです。

[ファイル] 現在のプロジェクト中の".C"または".CPP"を拡張子にもつソースファイルを指定します。 これにより指定ファイル内に存在する関数をカパレジ範囲として設定します。 拡張子を省略した場合は、".C"を補います。".C"または".CPP"以外の拡張子を持つファイル は指定できません。 プレースホルダまたは[参照...]ボタンが利用できます。

カバレジ情報ファイルに保存した内容を再設定する場合は、[最近使ったカバレジ情報ファイル]から選択するか、[別のカバレジファイルを参照]でファイルオープンダイアログボックスを表示して選択します。

[最近使ったカバレジ情報ファイル]では最近保存されたファイルを4個まで表示します。

[OK]ボタンをクリックすると[カバレジ]ウィンドウを表示します。

アドレス指定で[カバレジ]ウィンドウが既に表示されている場合は、そのウィンドウに設定が追加されます。



(1) カバレジウィンドウ(アドレス指定)



図 3-46 カバレジウィンドウ(アドレス指定)

カバレジ範囲とカバレジ統計情報を表示します。 表示する項目は以下の通りです。

 [Range]
 アドレス範囲

 [Statistic]
 C0 カバレジ値(パーセント表示)

 [Status]
 各カバレジ範囲の有効/無効状態

[カバレジ]ウィンドウをクローズすると取得したカバレジ情報と取得条件の設定をクリアします。

(2) カバレジウィンドウ(ソースファイル指定)

カバレジ		_ 🗆 🗵
🚮 🖪 🕅	N 8 😽	
Functions	Statistic	Status
-main	96%	Enable
-sort	97%	Enable
-change	100%	Enable
•		Þ

図 3-47 カバレジウィンドウ(ソースファイル指定)

カバレジ範囲とカバレジ統計情報を表示します。

表示する項目は以下の通りです。

[Functions] カバレジ対象の関数

[Statistic] C0 カバレジ値(パーセント表示)

[Status] 各関数の有効/無効状態

• カラムタブをクリックすることで関数名または C0 カバレジ値の降順または昇順に並べ替えることができます。

[カバレジ]ウィンドウをクローズすると取得したカバレジ情報と取得条件の設定をクリアします。

# 3.10.2 カバレジ情報をすべて取得する

ポップアップメニューで[すべて有効]をチェック後、命令を実行するとすべてのカバレジ情報を取得します。デフォルトで[すべて有効]はチェックされています。

# 3.10.3 カバレジ情報をすべてクリアする

ポップアップメニューから[すべてクリア]を選択すると、取得したすべてのカバレジ情報をクリアします。

### 3.10.4 ソースファイルを表示する

ポップアップメニューから[ソースファイル表示]を選択すると、[エディタ]ウィンドウを開いて、[カバレジ]ウィンド

ウ上のカーソル位置のアドレスに対応するソースファイルを表示します。

#### 3.10.5 新しいカバレジ測定範囲を設定する

ポップアップメニューで[測定範囲追加...]を選択すると、[カバレジの設定]ダイアログボックス(図 3-45)を表示しま す。[カバレジの設定]ダイアログボックスについては、「3.10.1 カバレジウィンドウを開く」を参照してください。

# 3.10.6 カバレジ測定範囲を変更する

#### (1) カバレジ測定範囲をアドレスで指定した場合

カバレジ測定範囲を選択し、ポップアップメニューで[測定範囲編集...]を選択すると、[カバレジ測定範囲]ダイアログ ボックスを表示します。

カバレジ測定範囲			? ×
◎ 先頭アドレス(S):	H'000800	- 🔊	<u>O</u> K
終了アドレス( <u>E</u> ):	H'000963	- 🗾	キャンセル( <u>C</u> )
O 771N		▶ 参照()	
			K (L)

図 3-48 カバレジ測定範囲ダイアログボックス(アドレス指定)

本ダイアログボックスで命令実行情報取得の条件を変更します。下記項目を指定できます。 [先頭アドレス] 先頭アドレス(接頭辞省略時は16進で入力) [終了アドレス] 終了アドレス(接頭辞省略時は16進で入力)

[OK]ボタンをクリックするとカバレジ測定範囲を変更します。

(2) カバレジ測定範囲をソースファイルで指定した場合

ポップアップメニューで[測定範囲編集...]を選択すると、[カバレジ測定範囲]ダイアログボックスを表示します。

カバレジ測定範囲			?×
<ul> <li>         ・         ・         ・</li></ul>			<u>0</u> K
終了アドレス(E):			キャンセル(の)
๏ ファイル	resetprg.c	参照()	.]

図 3-49 カバレジ測定範囲ダイアログボックス(ソースファイル指定)

本ダイアログボックスで命令実行情報取得の条件を変更します。下記項目を指定できます。

[ファイル] 現在のプロジェクト中の".C"または".CPP"を型名にもつソースファイルを指定します。 これにより指定ファイル内に存在する関数をカバレジ範囲として設定します。 ファイル型名を省略した場合は、".C"を補います。".C"または".CPP"以外のファイル型 名を持つファイルは指定できません。 プレースホルダまたは[参照...]ボタンが利用できます。

[OK]ボタンをクリックするとカバレジ測定範囲を変更します。

# 3.10.7 カバレジ測定範囲を削除する

カバレジ測定範囲を選択し、ポップアップメニューから[カバレジ測定範囲削除]を選択すると、選択したカバレジ測定 範囲を削除します。



#### カバレジ情報を取得する 3.10.8

カバレジ測定範囲を選択し、ポップアップメニューから[有効]をチェック後、命令を実行するとカバレジ情報を取得し ます。デフォルトで[有効]はチェックされています。

#### カバレジ情報をクリアする 3.10.9

カバレジ測定範囲を選択し、ポップアップメニューから[クリア]を選択すると、取得したカバレジ情報をクリアします。

#### 3.10.10 カバレジ情報をファイルに保存する

ポップアップメニューから[保存...]を選択すると、カバレジ情報をファイルに保存するための[カバレジ情報を保存]ダ イアログボックスを表示します。

ファイル名(E): <u>QK</u> 参照()     キャンセル( <u>C</u> )	カバレジ情報を保存		? ×
	ファイル名( <u>E</u> ):	▶ 参照()	<u>O</u> K キャンセル( <u>C</u> )

図 3-50 カバレジ情報を保存ダイアログボックス

保存するカバレジ情報ファイルの場所と名前を指定します。プレースホルダまたは[参照…]ボタンが使用できます。

ファイル拡張子の入力を省略すると、ファイル拡張子として".COV"を自動的に付加します。 ファイル拡張子として、".COV"または".TXT"以外を入力するとエラーメッセージを出力します。

#### 3.10.11 カバレジ情報をファイルからロードする

ポップアップメニューから[ロード...]を選択すると、カバレジ情報をファイルからロードするための[カバレジ情報ロー ド)ダイアログボックスを表示します。

カバレジ情報ロード		? ×
ファイル名( <u>E</u> ):	▶ 参照Q	<u>K</u> キャンセル( <u>©</u> )
₩ 3-51	カバレジ情報ロードダイアログボックフ	

図 3-51 カハレシ情報ロードタイアロクホックス

ロードするカバレジ情報ファイルの場所と名前を指定します。プレースホルダまたは[参照…]ボタンが使用できます。 ロードできるファイル拡張子は".COV"のみです。その他のファイル拡張子を入力するとエラーメッセージを出力しま す。

# 3.10.12 最新の情報に更新する

ポップアップメニューから[最新の情報に更新]を選択すると、[カバレジ]ウィンドウ内容を最新に更新します。

# 3.10.13 Confirmation Request ダイアログボックス

[すべてクリア] 、[クリア]、[測定範囲編集...]、[測定範囲削除]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。

Confirmation Request
カバレジ情報をクリアします
☑ カバレジ情報を保存(S)
<u> </u>

図 3-52 Confirmation Request ダイアログボックス

[OK]ボタンをクリックすると、カバレジデータをクリアします。 [カバレジ情報を保存]をチェックすると、「図 3-50」に示す[カバレジ情報を保存]ダイログボックスを表示し、クリアする 前にカバレジデータをファイルに保存できます。

# 3.10.14 カバレジ情報を保存ダイアログボックス

[ファイル->セッションの保存]メニューオプションをクリックすると、[カバレジ情報を保存]ダイログボックスを表示 します。[カバレジ]ウィンドウのデータを別々またはまとめて保存することができます。

カバレジ情報を保存	F	<u>?</u> ×
アドレス範囲	: H'00001000 – H'000010ff	
(#UIY)	<u>(いいえい)</u> すべていいえ( <u>1</u> )	すべて(はい(A)

図 3-53 カバレジ情報を保存ダイアログボックス

- [カバレジ]ウィンドウが複数開いているとウィンドウ数分の[カバレジ情報を保存]ダイアログボックスが開きます。
- [すべていいえ]ボタンをクリックすると、すべてのカバレジ情報を保存しないでダイアログボックスを閉じます。
- [すべてはい]ボタンをクリックすると、すべての[カバレジ]ウィンドウデータを1個のファイルに保存します。



# 3.10.15 エディタウィンドウへのカバレジ結果表示

命令実行済のソース行に対応する[カバレジ]カラムを強調表示することで[エディタ]ウィンドウにもカバレジ結果を表示します。[カバレジ]ウィンドウでカバレジに関する設定を変更すると、対応する[カバレジ]カラムの表示も更新します。

🚸 Tuto	orial.c				
	] 💭				
行番	ソースアドレス	カバレジー	S	ソース	
22	FFFF9071		Y	yoid main(void)	
23			i	long a[10]:	
25				long j;	
26				int i;	
28	FFFF9074			printf("### Data Input ###¥n");	
29					
30	FFFF9084			for( i=U; i<1U; i++ ){	
32	FFFF9099			$\frac{1}{1}$ $\frac{1}$	
33	FFFF90A0			j = -j;	
34	FFFF90A8			) a[i] = i•	
36	FFFF90B2			printf("a[%d]=%ld¥n",i,a[i]);	
37				}	
38	FFFF90E4			sort(a); printf("*** Sorting results ****o").	
40	FFFF90FB			for( i=0; i<10; i++ ){	
41	FFFF9107			printf("a[%d]=%ld¥n",i,a[i]);	
42	FFFF9139			} change(a):	
44			}	onango(a),	-

図 3-54 カバレジカラム (ソース)

# 3.10.16 逆アセンブリウィンドウへのカバレジ結果表示

命令実行済の逆アセンブリ行に対応する[カバレジ-ASM]カラムを強調表示することで[逆アセンブリ]ウィンドウにも カバレジ結果を表示します。[カバレジ]ウィンドウでカバレジに関する設定を変更すると、対応する[カバレジ-ASM]カラ ムの表示も更新します。

🐠 Tutorial.c - 🖄	きアセンブリ				_ 0	×
E & &						
カバレジー ASM	S 逆アセンブリアドレス	オブジェクトコード	ラベル	逆アセンブリ		
T <sub>F</sub> T	FFFF9132 FFFF9135 FFFF9137 FFFF9138 FFFF9138 FFFF9138 FFFF9142 FFFF9142 FFFF9145 FFFF9147 FFFF9147 FFFF9147 FFFF9146 FFFF9153 FFFF9158 FFFF9158 FFFF9158 FFFF9158 FFFF9158 FFFF9158 FFFF9158 FFFF9158 FFFF9158 FFFF9158 FFFF9158	ED0E0B 61AE 29D0 EF01 390401 710030 02 7100E8 A109 665E E70E04 ED0E04 610E 2A05 38E800 660E E70E03 ED0E04 A88D 47E5 2905 38BE00 ED0E04	_sort	MOV.L CMP BLT.B MOV.L BSR.W ADD RTS ADD MOV.L MOV.L MOV.L CMP BRA.W MOV.L MOV.L MOV.L MOV.L MOV.L BRA.W MOV.L CMP BLT.B BRA.W MOV.L	2CH[R0],R14 #0AH,R14 OFFFF9107H R0,R1 _chanse #30H,R0,R0 #-18H,R0,R0 R1,14H[R0] #5H,R14 R14,10H[R0],R14 #0H,R14 OFFFF9156H OFFFF923BH #0H,R14 R14,0CH[R0] 10H[R0],R14 OCH[R0],R5 R14,R5 OFFF9167H OFFF9222H 10H[R0],R14	LT III
•					•	

図 3-55 カバレジカラム (逆アセンブリ)



# 3.11 手動で擬似割込みを発生させる

ウィンドウ上のボタンを押下して手動で擬似割込みを発生させるには、[トリガ]ウィンドウ、または[GUI I/O]ウィンド ウを使用します。

3.11.1 トリガウィンドウ

[トリガ]ウィンドウを開くには、[トリガ->CPU->トリガ]を選択するか、[トリガ]ツールバーボタン<mark>型</mark>をクリックしま す。

🧆 わガ			<u> </u>
л≛			
1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

図 3-56 トリガウィンドウ

本ウィンドウは、手動で擬似割込みを発生させるためのトリガボタンを表示します。トリガボタン押下時に発生する 擬似割込みの内容は、[トリガ設定]ダイアログボックスで設定します。

トリガボタンは最大 256 個まで設定できます。

シミュレータ・デバッガの擬似割込み処理については、「2.15 擬似割込み」を参照してください。

(1) トリガボタンを設定する

各トリガボタン押下時に発生する擬似割込みの内容を設定するには[トリガ設定]ダイアログボックスを使用します。 [トリガ設定]ダイアログボックスを開くには、ポップアップメニューの[設定...]を選択します。

トリガ設定		? ×
トリガ番号( <u>N</u> ) [	1 💌	<u>O</u> K
□ 有効(E)		キャンセル(()
名前( <u>A)</u> :	1	
割り込み条件1(1):	H'00000000	
割り込み条件2(2):	H'00000000	
割り込み優先順位	( <u>P</u> ): 0 <b>•</b>	

図 3-57 トリガ設定ダイアログボックス

本ダイアログボックスでは、トリガボタン押下時に発生する割込みの内容を設定します。

[トリル留ち]	設定するトリカホタノを選択します。
[名前]	[トリガ]ウィンドウに表示するトリガボタンの名前を指定します。
[有効]	チェックするとトリガボタンが有効になります。
[割り込み条件 1]	割込みベクタ番号を指定します。

[割り込み優先順位] 割込み優先順位を指定します。(接頭辞省略時は16進入力、16進表示)(0~8、または0~H'10) 0~8の場合は8、0~H'10の場合はH'10を指定すると高速割り込みとして扱います。

指定した内容は、[OK]ボタンをクリックすることにより設定します。[キャンセル]ボタンをクリックすると、設定しないでダイアログボックスを閉じます。

- 【注】 複数のトリガボタンの設定を変更した後に[キャンセル]ボタンをクリックすると、すべてのトリガボタンの設定 変更が無効になります。
- (2) トリガボタンの数を変える

[トリガ]ウィンドウに表示するトリガボタンの数を変えるには、ポップアップメニューから [ボタンの数]のサブメニューで[4]、[16]、[64]、[256]を選択します。

(3) トリガボタンのサイズを変える

[トリガ]ウィンドウに表示するトリガボタンのサイズを変えるには、ポップアップメニューから[サイズ]のサブメ ニューで[大]、[中]、[小]を選択します。

3.11.2 GUI I/O ウィンドウ

[GUI I/O]ウィンドウを開くには、[表示->グラフィック->GUI I/O]を選択するか、[GUI I/O]ツールバーボタン <mark>强</mark>をクリックします。



図 3-58 GUI I/O ウィンドウ

本ウィンドウは、手動で擬似割込みを発生させるためのボタンを設定します。ボタン押下時に発生する擬似割込みの 内容は、[ボタンの設定]ダイアログボックスで設定します。

シミュレータ・デバッガの擬似割込み処理については、「2.15 擬似割込み」を参照してください。

(1) ボタンを設定する

擬似割込みを発生させるボタンを設定するには[ボタンの設定]ダイアログボックスを使用します。 まずポップアップメニューの[ボタンの作成]を選択するか、ツールバーの[ボタンの作成]ボタンへをクリックしてくだ さい。マウスカーソルの形状が" + "に変化します。

この状態で作成するボタンの左上位置から右下位置までドラッグしてください。





図 3-59 GUI I/O ウィンドウ (ボタンの作成)

#### 次にこのボタンをダブルクリックして[ボタンの設定]ダイアログボックスを開きます。

ボタン名: Button ボタン種別の選択 へ入カ用 ・ 割り込み用 ・ 入力と割り込みの両用 入力 種別: Address マ アドレス: マ
ボタン種別の選択       ○ 入力用     ● 割り込み用     ○ 入力と割り込みの両用       入力     ●     番別:     ▲ddress       アドレス:     ●     ●
<ul> <li>○入力用</li> <li>○ 入力用</li> <li>○ 入力と割り込みの両用</li> <li>○入力</li> <li>種別:</li> <li>△ Address</li> <li>アドレス:</li> <li>図 図</li> </ul>
入力       種別:       アドレス:         アドレス:
種別: Address ✓ アドレス: ✓
アドレス:
データ:
ビット単位のマスクーー
< <u>≺</u> X91E:
7 6 5 4 3 2 1 0 ビット番号:
ビットシンボル: 🔽 値: 🛛 🔽
「割り込み
割り込み条件1: H'000000
割0込み条件2: H'000000
割り込み優先順位: 0 ▼
OK + + + + + + + + + + + + + + + + + + +

図 3-60 ボタンの設定ダイアログボックス

本ダイアログボックスでは、ボタン押下時に発生する割込みの内容を設定します。 [ボタン名] [GUI I/O]ウィンドウに表示するボタンの名前を指定します。 [ボタン種別の選択] [割り込み条件 1] [割り込み条件 1] [割り込み優先順位 [割り込み] [割り込み優先順位] 割込みベクタ番号を指定します。 [割り込み優先順位] 割込み低先順位を指定します。(接頭辞省略時は 16 進入力、16 進表示) (0~8、または 0~H'10 ) 0~8 の場合は 8、0~H'10 の場合は H'10 を指定すると高速割り込みとし て扱います。


# 3.12 標準入出力およびファイル入出力を行う

ユーザプログラムから標準入出力およびファイル入出力の I/O シミュレーションを行うには、[I/O シミュレーション] ウィンドウを利用します。

#### 3.12.1 I/O シミュレーションウィンドウを開く

[I/O シミュレーション]ウィンドウを開くには、[表示->CPU->I/O シミュレーション]を選択するか、[I/O シミュレーション]ツールバーボタン <br/>
の<br/>
をクリックします。

→1/0 シミュレーション	
Simulated I/O	
	_
<u> </u>	•

図 3-61 I/O シミュレーションウィンドウ

ユーザプログラムからの標準出力は本ウィンドウに出力されます。本ウィンドウ上でのキー入力がユーザプログラムへの 標準入力となります。

#### 3.12.2 入出力機能

サポートしている入出力処理は以下の通りです。

番号	機能コード	機能名	内容
1	H'21	GETC	標準入力からの1バイト入力
2	H'22	PUTC	標準出力への1バイト出力
3	H'23	GETS	標準入力からの1行入力
4	H'24	PUTS	標準出力への1行出力
5	H'25	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'27	FGETC	ファイルからの1バイト入力
8	H'28	FPUTC	ファイルへの1バイト出力
9	H'29	FGETS	ファイルからの1行入力
10	H'2A	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

#### 表 3-1 入出力機能一覧

この機能を実現するためには、まず入出力用の特定の位置を [シミュレータシステム]ダイアログボックスの [I/O シ ミュレーションアドレス]で指定し、[有効]をチェック後、ユーザプログラムを実行します。シミュレータ・デバッガでは、 ユーザプログラムの命令を実行中に、指定した位置へのサブルーチン分岐命令(BSR、JSR)すなわち I/O シミュレーショ ン命令を検出すると、R1、R2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、I/O シミュレーションを行う前にユーザプログラムの中で次の設定をしておきます。



3.デバッグ



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、I/O シミュレーション命令の次の命令からシミュレーションを再開します。

各入出力機能の詳細は、シミュレータヘルプを参照ください。

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。I/O シミュレーションアドレスとしてラベル SYS\_CALLを指定します。

	MOV.L	#01210000h, R1
	MOV.L	#PARM, R2
	MOV.L	# SYS_CALL,R3
	JSR	R3
STOP	NOP	
SYS_CALL	NOP	
PARM	.LWORD	INBUF
	.SECTION	B,DATA
INBUF	.BLKB	2
	.END	

# 3.13 仮想入出力パネルを作成する

シミュレータ・デバッガにはユーザターゲットシステムの簡単なキー入力パネルおよび出力パネルをウィンドウ上で 模擬する GUI 入出力機能があります。この仮想的な入出力パネルを作成するには GUI 入出力ウィンドウを使用します。 GUI 入出力ウィンドウでは、ウィンドウ上に仮想のボタンを配置してデータを入力することや仮想の LED を配置して データを出力することができます。



図 3-62 GUI I/O ウィンドウ例

## 3.13.1 GUI 入出力ウィンドウを開く

[表示->グラフィック->GUI I/O]を選択するか、[GUI I/O]ツールバーボタン<mark>羅</mark>をクリックすると、GUI I/O ウィンドウが 開きます。\_\_\_\_\_\_



図 3-63 GUI I/O ウィンドウ

GUI 入出力ウィンドウでは、以下のアイテムが配置できます。

ボタン

ボタンを押下することにより、仮想ポートへのデータ入力や、仮想割り込みを発生させます。

- ラベル 指定したアドレス(もしくはビット)に指定した値が書き込まれた際に、文字列を表示/消去します。
- LED
- 指定したアドレス(もしくはビット)に指定した値が書き込まれた際に、指定した色で表示します(LED点灯の代 用)。
- テキスト
   テキスト文字列を表示します。

3.デバッグ

### 3.13.2 ボタンを作成する

ツールバーの 「ボタンをクリックするかポップアップメニューの[ボタンの作成]を選択してください。 マウスカーソ ルの形状が" + "に変化します。

この状態で作成するボタンの左上位置から右下位置までドラッグしてください。



図 3-64 GUI I/O ウィンドウ (ボタン作成)

(1) ボタンクリック時のイベントを指定する

ッールバーの ▶ ボタンをクリックした状態で、作成したボタンをダブルクリックしてください。以下のダイアログボックスが開きます。

ボタンの設定	×
ボタン名:	Set
「ボタン種別の選択	₹
● 入力用 (	)割り込み用 🔿 入力と割り込みの両用
<sub>「</sub> 入力———	
種別:	Address
アドレス:	3E0 💌 🗾
データ: 55	サイズ: Byte 💌
「ビット単位のマ	2.0
	/30)
マスタ1世:	
7 (四)	6 5 4 3 2 1 0
ビットシンホル	
「割り込みーーー	
割り込み条件1:	H'0000000
割り込み条件2:	H'0000000
割り込み優先順	
	OK キャンセル

図 3-65 ボタンの設定ダイアログボックス

ボタン名、入力ポートアドレスおよび入力データを指定してください。 ボタン名に空白文字を使用することはできません。

3.13.3 ラベルを作成する

ツールバーの
「ボタンをクリックするか、ポップアップメニューの[ラベルの作成]を選択してください。マウスカー

ソルの形状が"+"に変化します。

この状態で作成する位置の左上位置から右下位置までドラッグしてください。ラベル枠を表示します。



図 3-66 GUI I/O ウィンドウ (ラベル作成)

ツールバーの ▶ ボタンをクリックするか、ポップアップメニューの[アイテムの選択]を選択後、 ラベル枠をダブルク リックしてください。ラベル内容を設定するダイアログボックスが開きます。 イベント発生時の動作を指定してください。ラベル名に空白文字を使用することはできません。

(1) 指定ビットへの任意の値書き込みの場合

アドレス 0x3E0 のビット3が0 であれば"印刷中"、1 であれば"印刷可"と表示する設定を以下に示します。

ラベルの設定		×
アドレス: 3E0		- 🔊
ビット/データ © ビット © データ	ビット番号:	3
表示文字列1: 表示文字列2:	印刷中 印刷可	
動作 ● 正論理	○ 負論理	
- データ		
表示文字列2:		

図 3-67 ラベルの設定ダイアログボックス (ビット指定)

(2) 指定アドレスへの任意の値書き込みの場合

アドレス 0x3E0 にデータ 0x10 が書き込まれた時に"印刷中"、データ 0x20 が書き込まれた時に"印刷可"と表示する設定 を以下に示します。



ラベルの設定 🔀
アドレス: 3EO 💌 戻
- ビット/データー ○ ビット ○ データ
表示文字列1: 印刷中 表示文字列2: 印刷可
動作
データー 表示文字列1: 10
表示文字列2: 20
OK キャンセル

図 3-68 ラベルの設定ダイアログボックス (データ指定)

#### 3.13.4 LED を作成する

ッールバーの<br />
ペーパーの<br />
ペーパーの<br />
ペーパーの<br />
ペーパーの<br />
パーパーの<br />
ペーパーの<br />
パーパーの<br />
ペーパーの<br />
してください。<br />
マウスカーソ<br />
ルの形状が" + "に変化します。

この状態で作成する位置の左上位置から右下位置までドラッグしてください。LED 出力枠を表示します。



図 3-69 GUI I/O ウィンドウ (LED 作成)

ツールバーの ▶ ボタンをクリックするか、ポップアップメニューの[アイテムの選択]を選択後、 LED 出力枠をダブル クリックしてください。LED 出力を設定するダイアログボックスが開きます。イベント発生時の動作を指定してください。

(1) 指定ビットへの任意の値書き込みの場合

アドレス 0x3E0 のビット 2 が 0 であれば緑色、1 であれば赤色表示する設定を以下に示します。



LEDの設定 ×
アドレス: 3E0 💌 🗾
ビット/データー ・ ・ ビット ・ ビット番号: 2 ・ アータ
表示色1 表示色2 表示色2
- 動作 ・ 正論理 ・ 自論理
データー
表示色1:
表示色2:
表示色2:

(2) 指定アドレスへの任意の値書き込みの場合

アドレス 0x3E0 にデータ 0x10 が書き込まれた時に緑色表示、データ 0x20 が書き込まれた時に赤色表示する設定を以下に示します。

LEDの設定 <u>×</u>
アドレス: 3E0 💌 🗾
ビット/データー 〇 ビット 〇 データ
表示色1 表示色2 表示色2
動作 ⑥ 正論理 〇 負論理
-データ
表示色1: 10
表示色2: 20
OK キャンセル
🛿 3-71 LED の設定ダイアログボックス (データ指定

表示色1、表示色2ボタンをクリックするとカラー選択ダイアログボックスが開きます。



#### 3.13.5 テキスト文字列を作成する

ツールバーの<sup>11</sup>ボタンをクリックするか、ポップアップメニューの[テキストの作成]を選択してください。マウスカー ソルの形状が" + "に変化します。

この状態で作成するテキスト文字列の左上位置から右下位置までドラッグしてください。



図 3-72 GUI I/O ウィンドウ (文字列作成)

(1) テキスト文字列の表示形式を指定する

ツールバーの↓ボタンをクリックした状態で、作成したテキスト文字列をダブルクリックしてください。以下のテキスト文字列の表示形式を設定するためのダイアログボックスが開きます。

/10 C ID A / 01		
テキストの設定		×
テキスト: ┌─フォント	Text	
フォント名:	FixedSys	
サイズ:	11	フォント
テキ	·21	
背	景	
	OK	キャンセル

図 3-73 テキストの設定ダイアログボックス

[フォント]ボタンを押して、表示するテキスト文字列のフォントとサイズを設定してください。 [テキスト]ボタンを押して、表示するテキスト文字列の文字色を設定してください。 [背景]ボタンを押して、表示するテキスト文字列の背景色を設定してください。

### 3.13.6 アイテムのサイズ/配置を変更する

ツールバーの 
・ボタンをクリックした状態で、変更するアイテムをクリックしてください。 以下のようにアイテムが
選択された状態となります。



GUI I/O - 新規パネル *	×
<b></b>	
Button	
••••••••	-

図 3-74 GUI I/O ウィンドウ (アイテムの選択)

この状態でドラッグすることにより、アイテムサイズの変更や移動ができます。

#### 3.13.7 アイテムをコピーする

ッールバーの
・
ボタンをクリックするか、ポップアップメニューの[コピー]を選択してください。マウスカーソルの
形状が"+"に変化します。

この状態でコピー元のアイテムをクリックしてください。ツールバーの記述タンをクリックするか、ポップアップメニューの[貼り付け]を選択してください。コピー元と同サイズのアイテムが作成されます。

#### 3.13.8 アイテムを削除する

ッールバーの<mark>×</mark>ボタンをクリックするか、ポップアップメニューの[削除]を選択してください。マウスカーソルの形 状が" + "に変化します。

この状態で削除するアイテムをクリックしてください。

#### 3.13.9 グリッド線を表示する

ツールバーの

ボタンをクリックするか、ポップアップメニューの[グリッドの表示]を選択してください。背景にグリッドを表示します。

GUI I/O - 新	規パネル *	2
<b>► ×</b>	⊒, 🖳 ■, 🖷 🖨 🖨	
	Button	

図 3-75 GUI I/O ウィンドウ (グリッドの表示)

再度
ボタンをクリックすることにより、グリッドを非表示にします。

#### 3.13.10 入出力パネル情報を保存する

作成した入出<u>力パ</u>ネルは、その情報をファイルに保存することにより、再利用することができます。

ッールバーの<mark>||</mark>ボタンをクリックするか、ポップアップメニューの[保存]を選択してください。[GUII/O パネルファ イルの保存]ダイアログボックスが開きます。

保存先ディレクトリとファイル名を指定してください。

#### 3.13.11 入出力パネル情報を読み込む

ッールバーの🚰 ボタンをクリックするか、ポップアップメニューの[読み込み]を選択してください。 [GUII/Oパネル ファイルの読み込み]ダイアログボックスが開きます。

読み込むファイルを指定してください。読み込む前のパネル情報は削除されます。





# 4. ウィンドウ

ウィンドウー覧を表 4-1に示します。 ツールバーボタンについては、シミュレータ・デバッガヘルプを参照ください。

ウィンドウ名	機能
Ю	I/O レジスタを見る
I/O シミュレーション	標準入出力およびファイル入出力を行う
イベントポイント	シミュレータ・デバッガのブレークポイントを使用する
ウォッチ	変数を表示する(任意の変数)
エディタ	ソースコードを表示する
画像	メモリ内容を画像形式で表示する
カバレジ	コードカバレジを測定する
逆アセンブリ	アセンブリ言語コードを表示する
コマンドライン	コマンドラインインタフェースでデバッグ
スタックトレース	関数呼び出し履歴を見る
ステイタス	現在の状態の参照
トリガ	手動で擬似割り込みを発生させる
トレース	トレース情報を見る
波形	メモリ内容を波形形式で表示する
パフォーマンス解析	パフォーマンスを解析する
プロファイル / プロファイル チャート	プロファイル情報を見る
メモリ	メモリ領域を見る
ラベル	ラベルを参照 / 設定する
レジスタ	レジスタ内容を見る
ローカル	変数を表示する(ローカル変数)
GUI I/O	仮想入出力パネルを作成する
OS オブジェクト	タスクやセマフォなどの OS オブジェクトの状態を表示する
タスクトレース	リアルタイム OS を使用したプログラムの実行履歴を計測し、グラフィ カルに表示する
タスクアナライズ	CPU 占有状況を表示する

表 4-1 ウィンドウー覧





# 5. コマンドライン

# 5.1 コマンド一覧(機能別)

機能別のコマンド一覧を示します。 各コマンドのシンタックスは、ヘルプを参照ください。

### 5.1.1 実行関連

コマンド名	短縮形	説明
GO	GO	ユーザプログラムの実行
GO_RESET	GR	リセットベクタからのユーザプログラムの実行
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
HALT	HA	ユーザプログラムの停止
RESET	RE	CPUのリセット
STEP	ST	ステップ実行(命令単位またはソース行単位)
STEP_MODE	SM	ステップモードの設定
STEP_OUT	SP	PC 位置の関数を終了するまでのステップ実行
STEP_OVER	SO	ステップオーバ実行
STEP_RATE	SR	ステップ実行速度の設定、表示

## 5.1.2 ダウンロード関連

コマンド名	短縮形	説明
BUILD	BU	カレントプロジェクトのビルド
BUILD_ALL	BL	カレントプロジェクトのすべてをビルド
BUILD_FILE	BF	ファイルのコンパイル
BUILD_MULTIPLE	BM	複数プロジェクトのビルド
CLEAN	CL	ビルドの中間ファイルおよび出力ファイルの削除
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
FILE_LOAD	FL	オプジェクト(プログラム)ファイルのロード
FILE_LOAD_ALL	LA	すべてのオブジェクト(プログラム)ファイルのロード
FILE_SAVE	FS	メモリ内容のファイルセーブ
FILE_UNLOAD	FU	オブジェクト(プログラム)ファイルのアンロード
FILE_UNLOAD_ALL	UA	すべてのオブジェクト(プログラム)ファイルのアンロード
FILE_VERIFY	FV	ファイル内容とメモリ内容の比較
GENERATE_MAKE_FILE	GM	カレントワークスペースの make ファイル作成

# 5.1.3 レジスタ操作関連

コマンド名	短縮形	説明
REGISTER_DISPLAY	RD	レジスタ値の表示
REGISTER_SET	RS	レジスタ値の設定



## 5.1.4 メモリ操作関連

コマンド名	短縮形	説明
CACHE	-	メモリキャッシュの有効化 / 無効化
MEMORY_COMPARE	MC	メモリ内容の比較
MEMORY_DISPLAY	MD	メモリ内容の表示
MEMORY_EDIT	ME	メモリ内容の変更
MEMORY_FILL	MF	指定データによるメモリ内容の一括変更
MEMORY_FIND	MI	メモリ内容の検索
MEMORY_MOVE	MV	メモリプロックの移動
MEMORY_TEST	MT	メモリブロックのテスト

# 5.1.5 アセンブル/逆アセンブル関係

コマンド名	短縮形	説明
ASSEMBLE	AS	アセンブルの実行
DISASSEMBLE	DA	逆アセンブル表示
SYMBOL_ADD	SA	シンボルの設定
SYMBOL_CLEAR	SC	シンボルの削除
SYMBOL_LOAD	SL	シンボル情報ファイルのロード
SYMBOL_SAVE	SS	シンボル情報のファイルセーブ
SYMBOL_VIEW	SV	シンボルの表示

# 5.1.6 ブレーク設定関連

コマンド名	短縮形	説明
BREAKPOINT	BP	実行命令位置によるプレークポイントの設定
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
BREAK_CLEAR	BC	ブレークポイントの削除
BREAK_CYCLE	BCY	サイクルによるプレーク条件の設定
BREAK_DATA	BD	メモリのデータ値によるブレーク条件の設定
BREAK_DATA_DIFFERENCE	BDD	メモリのデータ値の変化量(差分)によるブレーク条件の設定
BREAK_DATA_INVERSE	BDI	メモリのデータ値の符号反転によるブレーク条件の設定
BREAK_DATA_RANGE	BDR	メモリのデータ範囲によるブレーク条件の設定
BREAK_DISPLAY	BI	ブレークポイント一覧の表示
BREAK_ENABLE	BE	ブレークポイントの有効/無効の切換え
BREAK_REGISTER	BR	レジスタのデータ値によるブレーク条件の設定
BREAK_SEQUENCE	BS	実行順序を指定したプレークポイントの設定
SET_DISASSEMBLY_SOFT_BRE AK	SDB	逆アセンブリレベルのソフトウェアブレークポイントの設定 または解除
SET_SOURCE_SOFT_BREAK	SSB	ソースレベルのソフトウェアブレークポイントの設定または 解除
STATE_DISASSEMBLY_SOFT_B REAK	TDB	逆アセンブリレベルのソフトウェアブレークポイントの有効 化または無効化
STATE_SOURCE_SOFT_BREAK	TSB	ソースレベルのソフトウェアブレークポイントの有効化また は無効化

# 5.1.7 トレース関連

コマンド名	短縮形	説明
TRACE	TR	トレース情報の表示
TRACE_CONDITION_SET	TCS	トレース情報の取得条件設定
TRACE_SAVE	TV	トレース情報をファイルへ出力



## 5.1.8 カバレジ計測関連

コマンド名	短縮形	説明
COVERAGE	CV	カバレジ測定の有効化 / 無効化
COVERAGE_DISPLAY	CVD	カバレジ情報の表示
COVERAGE_LOAD	CVL	カバレジ情報のロード
COVERAGE_RANGE	CVR	カバレジ範囲の設定
COVERAGE_SAVE	CVS	カバレジ情報のセーブ

## 5.1.9 パフォーマンス測定関連

コマンド名	短縮形	説明
ANALYSIS	AN	性能分析機能の有効化 / 無効化
ANALYSIS_RANGE	AR	性能評価関数の設定、表示
ANALYSIS_RANGE_DELETE	AD	性能分析範囲の削除
PROFILE	PR	プロファイルの有効化 / 無効化
PROFILE_DISPLAY	PD	プロファイル情報の表示
PROFILE_SAVE	PS	プロファイル情報のファイル出力

## 5.1.10 ウォッチ関連

コマンド名	短縮形	説明
WATCH_ADD	WA	Watch アイテムの追加
WATCH_AUTO_UPDATE	WU	Watch アイテムの自動更新の設定または解除
WATCH_DELETE	WD	Watch アイテムの削除
WATCH_DISPLAY	WI	ウォッチウィンドウの内容の表示
WATCH_EDIT	WE	Watch アイテムの値の編集
WATCH_EXPAND	WX	Watch アイテムの展開または縮小
WATCH_RADIX	WR	Watch アイテムの表示基数の変更
WATCH_RECORD	WO	Watch アイテムの値更新履歴をファイルに出力
WATCH_SAVE	WS	ウォッチウィンドウの表示内容をファイルに保存

# 5.1.11 スクリプト/ログファイル関連

コマンド名	短縮形	説明
!	-	コメント
ASSERT	-	コンディションのチェック
AUTO_COMPLETE	AC	オートコンプリート機能の有効化 / 無効化
ERASE	ER	コマンドラインウィンドウの内容のクリア
EVALUATE	EV	式の計算
LOG	LO	ロギングファイルの操作
SLEEP	-	コマンド実行の遅延
SUBMIT	SU	コマンドファイルの実行
TCL	-	TCL の情報を表示

## 5.1.12 マップ関連

コマンド名	短縮形	説明
MAP_DISPLAY	MA	メモリリソース設定の表示
MAP_SET	MS	メモリリソースの設定



## 5.1.13 シミュレーションの設定関連

コマンド名	短縮形	説明
EXEC_MODE	EM	実行モードの設定、表示
EXEC_STOP_SET	ESS	割り込み発生時の実行モードの設定、表示

# 5.1.14 標準入出力およびファイル入出力関連

コマンド名	短縮形	説明
SIMULATEDIO_CLEAR	SIOC	I/O シミュレーションウィンドウの内容のクリア
TRAP_ADDRESS	TP	I/O シミュレーションアドレスの設定
TRAP_ADDRESS_DISPLAY	TD	I/O シミュレーションアドレス設定の表示
TRAP_ADDRESS_ENABLE	TE	I/O シミュレーション有効化 / 無効化

# 5.1.15 ユーティリティ関連

コマンド名	短縮形	説明
HELP	HE	コマンドラインヘルプの表示
INITIALIZE	IN	デバッグプラットフォームの初期化
QUIT	QU	HEW の終了
RADIX	RA	入力ラディックス(基数)の設定
RESPONSE	RP	ウィンドウリフレッシュ間隔の設定
STATUS	STA	デバッグプラットフォームの状況表示
TOOL_INFORMATION	то	現在登録されているツールの情報をファイルへ出力

## 5.1.16 プロジェクト/ワークスペース関連

コマンド名	短縮形	説明
ADD_FILE	AF	カレントプロジェクトへのファイル追加
CHANGE_CONFIGURATION	CC	コンフィギュレーションの設定
CHANGE_PROJECT	CP	プロジェクトの設定
CHANGE_SESSION	CS	セッションの設定
CHANGE_SUB_SESSION	СВ	同期デバッグが有効な場合のアクティブセッションの変更
CLEAR_OUTPUT_WINDOW	COW	アウトプットウィンドウの各タブの表示内容クリア
CLOSE_WORKSPACE	CW	ワークスペースを閉じる
OPEN_WORKSPACE	OW	ワークスペースのオープン
REFRESH_SESSION	RSE	セッション情報の再読み込み
REMOVE_FILE	REM	カレントプロジェクトからのファイル削除
SAVE_SESSION	SE	現在のセッションをセーブ
SAVE_WORKSPACE	SW	ワークスペースをセーブ
UPDATE_ALL_DEPENDENCIES	UD	カレントプロジェクトのすべての依存関係更新

## 5.1.17 テスト支援機能関連

コマンド名	短縮形	説明
CLOSE_TEST_SUITE	CTS	テストスウィートを閉じる
COMPARE_TEST_DATA	CTD	テストデータの比較
OPEN_TEST_SUITE	OTS	テストスイートを開く
RUN_TEST	RT	テストの実行



短縮形	説明
OAA	OS オブジェクトの追加(オブジェクト種別指定)
OAD	OS オブジェクトの削除(シート指定)
OAU	表示更新を「実行中」と「停止時」に変更
ODL	OS オブジェクトを 1 行下へ移動(単体指定)
ODS	OS オブジェクト表示内容をファイルに保存
ODU	OS オブジェクトを 1 行上へ移動(単体指定)
OD	OS オブジェクトの表示
ONU	更新表示を「更新しない」に変更
OOA	OS オブジェクトの追加(単体指定)
OOD	OS オブジェクトの削除(単体指定)
OOE	OS オブジェクトの編集(単体指定)
OSL	OS オブジェクト設定項目ファイルの読み込み
OSS	OS オブジェクト設定項目をファイルに保存
OSU	表示更新を「停止時のみ」に変更
	短縮形 OAA OAU ODL ODS ODU OD OD ONU OOA OOD OOA OOD OOE OSL OSS OSU

# 5.1.18 リアルタイム OS 対応デバッグ機能関連

## 5.1.19 仮想ポートファイル入出力関連

コマンド名	短縮形	説明
PORT_FILE_ADD	PFA	仮想ポートへのデータ入出力ファイルの追加
PORT_FILE_CLOSE	PFC	仮想ポートへのデータ入出力ファイルのクローズ
PORT_FILE_DELETE	PFD	仮想ポートへのデータ入出力ファイル指定の削除
PORT_FILE_OPEN	PFO	仮想ポートへのデータ入出力ファイルのオープン
PORT_FILE_STATUS	PFS	仮想ポートへのデータ入出力ファイルの状態表示

# 5.2 コマンド一覧(アルファベット順)

アルファベット順コマンド一覧を示します。

各コマンドのシンタックスは、ヘルプを参照ください。

項番	コマンド名	短縮形	説明
1	!	-	コメント
2	ADD_FILE	AF	カレントプロジェクトへのファイル追加
3	ANALYSIS	AN	性能分析機能の有効化 / 無効化
4	ANALYSIS_RANGE	AR	性能評価関数の設定、表示
5	ANALYSIS_RANGE_DELETE	AD	性能分析範囲の削除
6	ASSEMBLE	AS	アセンブルの実行
7	ASSERT	-	コンディションのチェック
8	AUTO_COMPLETE	AC	オートコンプリート機能の有効化 / 無効化
9	BREAKPOINT	BP	実行命令位置によるブレークポイントの設定
10	BREAK_ACCESS	BA	メモリ範囲のアクセスによるブレーク条件の設定
11	BREAK_CLEAR	BC	ブレークポイントの削除
12	BREAK_CYCLE	BCY	サイクルによるブレーク条件の設定
13	BREAK_DATA	BD	メモリのデータ値によるブレーク条件の設定
14	BREAK_DATA_DIFFERENCE	BDD	メモリのデータ値の変化量(差分)によるブレーク条件の設定
15	BREAK_DATA_INVERSE	BDI	メモリのデータ値の符号反転によるブレーク条件の設定
16	BREAK_DATA_RANGE	BDR	メモリのデータ範囲によるブレーク条件の設定
17	BREAK_DISPLAY	BI	ブレークポイント一覧の表示
18	BREAK_ENABLE	BE	ブレークポイントの有効/無効の切替
19	BREAK_REGISTER	BR	レジスタのデータ値によるブレーク条件の設定
20	BREAK_SEQUENCE	BS	実行順序を指定したブレークポイントの設定
21	BUILD	BU	カレントプロジェクトのビルド
22	BUILD_ALL	BL	カレントプロジェクトのすべてのビルド
23	BUILD_FILE	BF	ファイルのコンパイル
24	BUILD_MULTIPLE	BM	複数プロジェクトのビルド
25	CACHE	-	メモリキャッシュの有効化 / 無効化
26	CHANGE_CONFIGURATION	CC	コンフィギュレーションの設定
27	CHANGE_PROJECT	CP	プロジェクトの設定
28	CHANGE_SESSION	CS	セッションの設定
29	CHANGE_SUB_SESSION	СВ	同期デバッグが有効な場合のアクティブセッションの変更
30	CLEAN	CL	ビルドの中間ファイルおよび出力ファイルの削除
31	CLEAR_OUTPUT_WINDOW	COW	アウトプットウィンドウの各タブの表示内容クリア
32	CLOSE_TEST_SUITE	CTS	テストスウィートを閉じる
33	CLOSE_WORKSPACE	CW	ワークスペースを閉じる
34	COMPARE_TEST_DATA	CTD	テストデータの比較
35	COVERAGE	CV	カバレジ測定の有効化 / 無効化
36	COVERAGE_DISPLAY	CVD	カバレジ情報の表示
37	COVERAGE_LOAD	CVL	カバレジ情報のロード
38	COVERAGE_RANGE	CVR	カバレジ範囲の設定
39	COVERAGE_SAVE	CVS	カバレジ情報のセーブ
40	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
41	DISASSEMBLE	DA	逆アセンブル表示
42	ERASE	ER	[コマンドライン]ウィンドウの内容のクリア
43	EVALUATE	EV	式の計算
44	EXEC_MODE	EM	実行モードの設定、表示
45	EXEC_STOP_SET	ESS	割込み発生時の実行モードの設定、表示
46	FILE_LOAD	FL	オブジェクト(プログラム)ファイルのロード
47	FILE LOAD ALL	LA	すべてのオブジェクト(プログラム)ファイルのロード

表 5-1 コマンド一覧



項番	コマンド名	短縮形	説明
48	FILE_SAVE	FS	メモリ内容のファイルセーブ
49	FILE_UNLOAD	FU	オブジェクト(プログラム)ファイルのアンロード
50	FILE_UNLOAD_ALL	UA	すべてのオブジェクト(プログラム)ファイルのアンロード
51	FILE_VERIFY	FV	ファイル内容とメモリ内容の比較
52	GENERATE_MAKE_FILE	GM	カレントワークスペースの make ファイル作成
53	GO	GO	ユーザプログラムの実行
54	GO_RESET	GR	リセットベクタからのユーザプログラムの実行
55	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
56	HALT	HA	ユーザプログラムの停止
57	HELP	HE	コマンドラインのヘルプ表示
58	INITIALIZE	IN	デバッグプラットフォームの初期化
59	LOG	LO	ロギングファイルの操作
60	MAP_DISPLAY	MA	メモリリソース設定の表示
61	MAP_SET	MS	メモリリソースの設定
62	MEMORY_COMPARE	MC	メモリ内容の比較
63	MEMORY_DISPLAY	MD	メモリ内容の表示
64	MEMORY_EDIT	ME	メモリ内容の変更
65	MEMORY_FILL	MF	指定データによるメモリ内容の一括変更
66	MEMORY_FIND	MI	メモリ内容の検索
67	MEMORY_MOVE	MV	メモリブロックの移動
68	MEMORY_TEST	MT	メモリブロックのテスト
69	OPEN_TEST_SUITE	OTS	テストスイートを開く
70	OPEN_WORKSPACE	OW	ワークスペースのオープン
71	OSOBJECT_ALL_ADD	OAA	OS オブジェクトの追加(オブジェクト種別指定)
72	OSOBJECT_ALL_DELETE	OAD	OS オブジェクトの削除(シート指定)
73	OSOBJECT_AUTO_UPDATE	OAU	表示更新を「実行中」と「停止時」に変更
74	OSOBJECT_DATA_LOWLINE	ODL	OS オブジェクトを 1 行下へ移動(単体指定)
75	OSOBJECT_DATA_SAVE	ODS	OS オブジェクト表示内容をファイルに保存
76	OSOBJECT_DATA_UPLINE	ODU	OS オブジェクトを 1 行上へ移動(単体指定)
77	OSOBJECT_DISPLAY	OD	OS オブジェクトの表示
78	OSOBJECT_NO_UPDATE	ONU	更新表示を「更新しない」に変更
79	OSOBJECT_ONE_ADD	OOA	OS オブジェクトの追加(単体指定)
80	OSOBJECT_ONE_DELETE	OOD	OS オブジェクトの削除(単体指定)
81	OSOBJECT_ONE_EDIT	OOE	OS オブジェクトの編集(単体指定)
82	OSOBJECT_SETTING_LOAD	OSL	OS オブジェクト設定項目ファイルの読み込み
83	OSOBJECT_SETTING_SAVE	OSS	OS オブジェクト設定項目をファイルに保存
84	OSOBJECT_STOP_UPDATE	OSU	表示更新を「停止時のみ」に変更
85	PORT_FILE_ADD	PFA	仮想ポートへのデータ入出力ファイルの追加
86	PORT_FILE_CLOSE	PFC	仮想ポートへのデータ入出力ファイルのクローズ
87	PORT_FILE_DELETE	PFD	仮想ポートへのデータ入出力ファイルの削除
88	PORT_FILE_OPEN	PFO	仮想ポートへのデータ入出力ファイルのオープン
89	PORT_FILE_STATUS	PFS	仮想ポートへのデータ入出力ファイルの状態表示
90	PROFILE	PR	プロファイルの有効化 / 無効化
91	PROFILE_DISPLAY	PD	プロファイル情報の表示
92	PROFILE_SAVE	PS	プロファイル情報のファイル出力
93	QUIT	QU	HEW の終了
94	RADIX	RA	入力ラディックス(基数)の設定
95	REFRESH_SESSION	RSE	セッション情報の再読み込み
96	REGISTER_DISPLAY	RD	レジスタ値の表示
97	REGISTER_SET	RS	レジスタ値の設定
98	REMOVE_FILE	REM	カレントプロジェクトからのファイル削除
99	RESET	RE	
100	RESPONSE	RP	ワィンドウリフレッシュ間隔の設定



項番	コマンド名	短縮形	説明
101	RUN_TEST	RT	テストの実行
102	SLEEP	-	コマンド実行の遅延
103	SAVE_SESSION	SE	現在のセッションをセーブ
104	SAVE_WORKSPACE	SW	ワークスペースの保存
105	SET_DISASSEMBLY_SOFT_BR EAK	SDB	逆アセンブリレベルのソフトウェアブレークポイントの設定 または解除
106	SET_SOURCE_SOFT_BREAK	SSB	ソースレベルのソフトウェアプレークポイントの設定または 解除
107	SIMULATEDIO_CLEAR	SIOC	I/O シミュレーションウィンドウの内容のクリア
108	STATE_DISASSEMBLY_SOFT_ BREAK	TDB	逆アセンブリレベルのソフトウェアブレークポイントの有効 化または無効化
109	STATE_SOURCE_SOFT_BREAK	TSB	ソースレベルのソフトウェアブレークポイントの有効化また は無効化
110	STATUS	STA	デバッグプラットフォームの状況表示
111	STEP	ST	ステップ実行(命令単位またはソース行単位)
112	STEP_MODE	SM	ステップモードの設定
113	STEP_OUT	SP	PC 位置の関数を終了するまでのステップ実行
114	STEP_OVER	SO	ステップオーバ実行
115	STEP_RATE	SR	ステップ実行速度の設定、表示
116	SUBMIT	SU	コマンドファイルの実行
117	SYMBOL_ADD	SA	シンボルの設定
118	SYMBOL_CLEAR	SC	シンボルの削除
119	SYMBOL_LOAD	SL	シンボル情報ファイルのロード
120	SYMBOL_SAVE	SS	シンボル情報のファイルセーブ
121	SYMBOL_VIEW	SV	シンボルの表示
122	TCL	-	TCL の有効化 / 無効化
123	TOOL_INFORMATION	то	現在登録されているツールの情報をファイルへ出力
124	TRACE	TR	トレース情報の表示
125	TRACE_CONDITION_SET	TCS	トレース情報の取得条件設定
126	TRACE_SAVE	TV	トレース情報をファイルへ出力
127	TRACE_STATISTIC	TST	統計情報解析の実行
128	TRAP_ADDRESS	TP	I/O シミュレーションアドレスの設定
129	TRAP_ADDRESS_DISPLAY	TD	I/O シミュレーションアドレス設定の表示
130	TRAP_ADDRESS_ENABLE	TE	I/O シミュレーションコール有効化 / 無効化
131	UPDATE_ALL_DEPENDENCIES	UD	カレントプロジェクトのすべての依存関係更新
132	WATCH_ADD	WA	Watch アイテムの追加
133	WATCH_AUTO_UPDATE	WU	Watch アイテムの自動更新の設定または解除
134	WATCH_DELETE	WD	Watch アイテムの削除
135	WATCH_DISPLAY	WI	ウォッチウィンドウの内容の表示
136	WATCH_EDIT	WE	Watch アイテムの値の編集
137	WATCH_EXPAND	WX	Watch アイテムの展開または縮小
138	WATCH_RADIX	WR	Watch アイテムの表示基数の変更
139	WATCH_RECORD	WO	Watch アイテムの値更新履歴をファイルに出力
140	WATCH_SAVE	WS	ウォッチウィンドウの表示内容をファイルに保存



# 6. メッセージー覧

# 6.1 インフォメーションメッセージ

シミュレータ・デバッガは実行経過をユーザに知らせるため、インフォメーションメッセージを出力します。シミュ レータ・デバッガの出力するインフォメーションメッセージを 表 6-1に示します。

メッセージ	内容
Break Access (Access Address:H'nnnnnnn, Type:xxxx, Access Size:yyyy)	ブレークアクセス条件が成立して実行を中断しました。付加情報として成立し たブレークアクセス条件(アクセスアドレス(Access Address)、アクセス種別 (Type)、およびアクセスサイズ(Access Size))を表示します。
Break Cycle (Cycle:H'nnnnnnn)	プレークサイクル条件が成立して実行を中断しました。付加情報として成立し たプレークサイクル条件(サイクル数(Cycle))を表示します。
Break Data (Access Address:H'nnnnnnn, Data:H'mmmm)	ブレークデータ条件(符号反転、差分以外)が成立して実行を中断しました。付 加情報として成立したブレークデータ条件(アクセスアドレス(Access Address)、および値(Data))を表示します。
Break Data (Access Address:H'nnnnnnn, Previous Data:H'mmmm, Current Data:H'mmmm)	ブレークデータ条件(符号反転、または差分)が成立して実行を中断しました。 付加条件として成立したブレークデータ条件(アクセスアドレス(Access Address)、前回値(Previous Data)、および今回値(Current Data))を表示します。
Break Register (Register:XX, Value:H'mmmm)	ブレークレジスタ条件が成立して実行を中断しました。付加情報として成立し たブレークレジスタ条件(レジスタ名(Register)、および値(Value))を表示しま す。
Break Sequence (PC:H'nnnnnnn)	プレークシーケンス条件が成立して実行を中断しました。付加情報として成立 したブレークシーケンス条件(最後の命令アドレス(PC))を表示します。
I/O DLL Stop	周辺機能が停止しました。
PC Breakpoint (PC:H'nnnnnnn)	ブレークポイント条件が成立して実行を中断しました。付加情報として成立し たブレークポイント条件(命令アドレス(PC))を表示します。
Step Normal End	ステップ実行が正常に終了しました。
Stop	[STOP]ボタンにより実行を中断しました。
Trace Buffer Full	[トレース取得]ダイアログボックスの[トレースバッファ満杯時の動作]で[停 止]モードを選択しており、かつトレースバッファが満杯となったので実行を 中断しました。
WAIT Instruction	WAIT命令により命令実行を中断しました。

表 6-1 インフォメーションメッセージ一覧



# 6.2 エラーメッセージ

シミュレータ・デバッガはユーザプログラムや操作の誤りをユーザに知らせるため、エラーメッセージを出力します。 シミュレータ・デバッガの出力するエラーメッセージを表 6-2に示します。

メッセージ	内容・対策			
Undeifined Instruction Exception	未定義命令例外処理でエラーが発生しました。			
Privilege Insruction Exception	特権命令例外処理でエラーが発生しました。			
Floating-point Exception	浮動小数点例外処理でエラーが発生しました。			
Reset Exception	リセット例外処	リセット例外処理でエラーが発生しました。		
Interrupt Exception	割り込み処理で	エラーが発生しました。		
INT Instruction Exception	無条件トラッフ	<sup>1</sup> (INT 命令)例外処理でエラーが発生しました。		
BRK Instruction Execption	無条件トラッフ	<sup>(</sup> (BRK 命令)例外処理でエラーが発生しました。		
I/O area not exist	I/O 領域を削除	しました。		
	I/O 領域は必ず	設定してください。		
I/O DLL Illegal Interrupt Information (errNum=2xx)	割込み情報が不 てください。	「正です。詳細情報を[errNum]に示します。割込み情報を修正し		
	errNum	説明		
	200	指定したベクタが範囲外です		
	201	指定した優先度が範囲外です		
I/O DLL Memory Access Error (errNum=0xx, Address=0xXXXXXXXX)	周辺機能へのメモリアクセスでエラーが発生しました。詳細内容を[errNum]、 エラー発生アドレスを[Addess]に示します。エラー情報に従ってプログラムを 修正してください。			
	errNum	説明		
	001	指定したアドレスが範囲外です		
	002	指定した領域にメモリがありません		
	003	必要なメモリが確保できません		
	004	指定したデータサイズが範囲外です		
	005	指定したアドレスにアクセスできません		
I/O DLL Register Access Error (errNum=1xx, RegisterName=xxxx)	周辺機能へのレジスタアクセスでエラーが発生しました。詳細内容を [errNum]、エラー発生レジスタを[RegisterName]に示します。エラー情報に 従ってプログラムを修正してください。			
	errNum	説明		
	100	レジスタの記述に誤りがあります		
	101	指定したデータ値が不正です		
Memory Access Error (Address:H'nnnnnnn)	以下のいずれかの状態になりました。付加情報としてエラーの発生したアドレス(Address)を表示します。			
	(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			
	★モリの確保、属性変更を行うか、当該メモリアクセスが発生しないように ユーザプログラムを修正してください。			
System Call Error	1/0 シミュレー	ションエラーが発生しました。		
	レジスタR1,R2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。			
The memory resource has not	メモリマップ範	9囲外にメモリリソースを設定しました。		
been set up	エラーが発生しないようにメモリリソース設定を修正してください。			

表 6-2 エラーメッセージ一覧



# 7. チュートリアル

## 7.1 デバッグの準備

シミュレータ・デバッガの主な特徴をサンプルプログラムを用いて説明します。

注意

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

#### 7.1.1 サンプルプログラム

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

サンプルプログラムでは、以下の処理を行います。

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

サンプルプログラムは、HEWのデモンストレーションプログラムを用います。

#### 7.1.2 サンプルプログラムの作成

下記に注意して HEW のデモンストレーションプログラムを作成してください。

- 「新規ワークスペースを作成する」では[Project Type]に"Demonstration"を指定してください。
- [CPUシリーズ:] は"RX600"を、[ターゲット:] は"RX600 Simulator"を選択してください。
- コンフィギュレーションは、ツールバーより"SimDebug\_RX600"をビルド前に選択してください。
- セッションは、ツールバーより"SimSessionRX600"を選択してください。
- 本デモンストレーションプログラムでは周辺機能は利用しないので、セッション切換え時に表示される[周辺機 能シミュレーションの設定]ダイアログボックスは[このダイアログを表示しない]チェックボックスチェック 後に[OK]ボタンをクリックしてください。

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

## 7.2 デバッグのための設定

#### 7.2.1 メモリリソースの確保

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

[基本設定]メニューから [シミュレータ → メモリリソース...]を選択し、現在のメモリリソースを表示してください。



Ð٩	ロレータシステム	<b>v</b>								? ×
Ð	ステム メモリ									
	メモリ マップ()	<u>v</u> ):			0		× <sub>=</sub> è	አモリ リソース(	<u>R</u> ):	
	Begin	End	Туре	Size	Read	Write	Endian	Begin	End	Attribute
	00000000	0001FFFF	RAM		1	1	Little	00000000	00007FFF	リード/ライト
	00080000	000FFFFF	I/O		1	1		FFFF8000	FFFFFFF	リード/ライト
	00100000	00107FFF	ROM		1	1	Little			
	007F8000	007F9FFF	RAM		1	1	Little			
	007FC000	007FC4FF	I/O		1	1				
	007FFC00	007FFFFF	I/O		1	1				
	00E00000	00FFFFFF	ROM		1	1	Little			
	FEFFE000	FEFFFFFF	ROM		1	1	Little			
	FF7FC000	FF7FFFFF	ROM		1	1	Little			
	FFE00000	FFFFFFF	ROM		1	1	Little			
	-									
	OK キャンセル 適用(益)									

図 7-1 シミュレータシステムダイアログボックス(メモリページ)

プログラム領域として H'FFFF8000 から H'FFFFFFF、データ領域として H'00000000 から H'00007FFF を読み出し/書き 込み可能領域として確保しています。

• [OK]ボタンをクリックしてダイアログボックスを閉じてください。

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



## 7.2.2 サンプルプログラムのダウンロード

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

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

デバッグの設定			? ×
SimSessionRX600	ターゲット オプション		
Tutorial	ターゲット(I): R×600 Simulator コア(2): Single Core Target デバッグフォーマット(E): [Elf/Dwarf2 ダウンロードモジュール(D): Filename Offset Address Format \$(CONFIGDIR)¥\$(PR 00000000 Elf/Dwarf2	<b>x</b> <b>x</b> <b>x</b>	<u>追加(A)</u> 交更( <u>M)</u> 削除(I) 上へ(U) 下へ(Q)
		ОК	

図 7-2 デバッグの設定ダイアログボックス

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



# 7.2.3 ソースプログラムの表示

HEW では、ソースレベルでプログラムをデバッグできます。ソースプログラム("Tutorial.c")を表示してください。

• [ワークスペース]ウィンドウの Tutorial.c をダブルクリックして[ソース]ウィンドウを開いてください。

🚸 Tuto	prial.c			. 🗆 🗵				
行番	ソースアドレス	カバレジー:	S ソース					
23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50	FFFF9042 FFFF9045 FFFF9055 FFFF9064 FFFF9064 FFFF9071 FFFF9079 FFFF9083 FFFF9083 FFFF9088 FFFF9088 FFFF9088 FFFF9084 FFFF9004 FFFF9104 FFFF910D		<pre>void main(void) {     long a[10];     long j;     int i;     printf("### Data Input ####n");     for( i=0; i&lt;10; i++ ){         j = rand();         if(j &lt; 0){             j = -j;         }         a[i] = j;         printf("a[%d]=%ld¥n",i,a[i]);     }     sort(a);     printf("*** Sorting results ****¥n");     for( i=0; i&lt;10; i++ ){         printf("a[%d]=%ld¥n",i,a[i]);     }     change(a); } void sort(long *a) {     long t;     int i, j, k, gap; } </pre>					
Ľ								

図 7-3 ソースウィンドウ(ソースプログラムの表示)



### 7.2.4 PCブレークポイントの設定

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

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

🧼 Tuto	orial.c			
6	] 52			
行番	ソースアドレス	カバレジ	S/Wブレークポイント	ソース
23 24 25 26 27 28 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50	FFFF9042 FFFF9045 FFFF9055 FFFF9061 FFFF9064 FFFF9071 FFFF9073 FFFF9083 FFFF9083 FFFF9083 FFFF9088 FFFF9084 FFFF9004 FFFF9104 FFFF9104		•	<pre>void main(void) {     long a[10];     long j;     int i;     printf("### Data Input ####n");     for( i=0; i&lt;10; i++ ){         j = rand();         if(j &lt; 0){             j = -j;         }         a[i] = j;         printf("a[%d]=%Id¥n",i,a[i]);     }     sort(a);     printf("a[%d]=%Id¥n",i,a[i]);     for( i=0; i&lt;10; i++ ){         printf("a[%d]=%Id¥n",i,a[i]);     }     change(a); } void sort(long *a) {     long t;     int i, j, k, gap; </pre>

図 7-4 ソースウィンドウ (ブレークポイントの設定)

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

## 7.2.5 プロファイラの設定

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

<b>∞ブ</b> ロファイル										_	
🗈 - 🤄 📴 Show Function	ons/Var	ables		0							
Function/Variable	F/V	Address	Size	Times	Cycle	Ext :	mem	I/0	area	Int	mem
List Tree											

#### 図 7-5 プロファイルウィンドウ

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



### 7.2.6 I/Oシミュレーションの設定

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

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

シミュレータシステム			? ×
システムメモリ			
CPU( <u>C</u> ):			
RX600		▼	
ビットサイズ(B):	1/0シミュレーションアドレスΦ:	☑ 有効(№)	
D'32	H'00000000	▼ 🔊	
エンディアン(E):	実行モード⊗∷		
Little Endian	Stop	▼ 詳細(型)	
割り込み優先レベル(T):	レスポンス( <u>P)</u> :		
0-7 (MVTIPL命令無効)	D'40000		
🥅 命令デコード結果をキャッシュしてシミュレーション速度を	:向上する( <u>A</u> )		
	ОК	キャンセル 適用	( <u>A</u> )

図 7-6 シミュレータシステムダイアログボックス(システムタブ)

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



図 7-7 I/O シミュレーションウィンドウ

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

以下の[トレース取得]ダイアログボックスを表示します。



トレース取得		?×
トレース機能(II):	有効	•
トレースバッファ満杯時の動作( <u>F</u> ):	続行	•
トレース容量( <u>C</u> ):	65536 V I -	- 14-
取得条件( <u>A</u> ):	বশ্ব	•
- トレースイベント	1	
Type Condition		<u>追加(D)</u>
		削除( <u>E</u> )
		すべて削除(L)
		すべて有効( <u>N</u> )
		すべて無効⑤)
	OK	キャンセル

図 7-8 トレース取得ダイアログボックス

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

#### 7.2.8 スタックポインタ、プログラムカウンタの設定

プログラムを実行するために、プログラムカウンタをリセットベクタから設定してください。サンプルプログラムの リセットベクタには PC 値として H'FFFF8000 が書いてあります。

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

リセットベクタからプログラムカウンタに H'FFFF8000 を設定します。



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

プログラムはブレークポイントを設定したところまで実行します。プログラムが停止した位置を示すために[ソース] ウィンドウに を表示します。また 停止要因として[アウトプット]ウィンドウに"PC Breakpoint (PC:H'FFFF90E4)"を表示

94

します。

「行番… ソースアドレス カバレジ S… ソース 23 FFFF9042 24 25 しのg a[10];	22
行番…     ソースアドレス     カバレジ     S…     ソース       23     FFFF9042     void main(void)       24     25     long a[10];	
23 FFFF9042 void main(void) 24 { 25   long a[10];	
26 27 28	•
29       FFFF9045       printf("### Data Input ###¥n");         30       31       FFF9055       for( i=0; i<10; i++ ){	
39       FFFF90B3         40       FFFF90B8         41       FFFF90C8         42       FFFF90D4         43       Printf("a[%d]=%Id¥n",i,a[i]);         44       FFFF9104         45       Change(a);         46       Printf("a[%d]=%Id¥n",i,a[i]);         48       Void sort(long *a)         49       Iong t;         50       Int i, j, k, gap;	

図 7-11 ソースウィンドウ(ブレーク状態)



[ステイタス]ウィンドウで停止要因が確認できます。

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

🧆 ステイタス	
Item	Status
Connected To	RX600 Simulator
СРО	RX600
Exec Mode	Stop
Run Status	Ready
Break Cause	PC Breakpoint(PC:H'FFFF90B3)
Execute From	Reset
Exec Instructions	35898
Cycles	58246
Run Time Count	00:00:00.000.582.460
CPU Frequency	100 MHz
▲ ► \ Memory \ Platform	Events /

図 7-12 ステイタスウィンドウ

これは、以下の実行内容を表示します。

ブレークの原因は PCブレーク リセットからの実行 リセットからの実行命令数は35,898命令 リセットからの実行サイクル数は58,246サイクル リセットからの実行時間は582.46マイクロ秒

CPU**動作周波数は**100MHz



[レジスタ]ウィンドウでレジスタの値が確認できます。

• [表示]メニューから [CPU-> レジスタ]を選択してください。

🧈 レジスタ	2	
Name	Value	
RO	00001A5C	
R1	000000A	
R2	00001008	
R3	0000000	
R4	00001001	
R5	0000000	
R6	0000000	
R7	0000000	
R8	0000000	
R9	0000000	
R10	0000000	
R11	0000000	
R12	0000000	
R13	0000000	
R14	A000000A	
R15	0000198C	
USP	00001A5C	
ISP	00001в90	
PSW	000000000010011000000000000011	OPUIZC
PC	FFFF90B3	
INTB	FFFF858C	
BPSW	0000000	
BPC	0000000	
FINTV	0000000	
FPSW	00000100	
ACC	00000000000000	

図 7-13 レジスタウィンドウ

プログラム停止時の各レジスタの値を確認することができます。

## 7.3.2 トレースバッファの使い方

トレースバッファを使って、命令実行の履歴を知ることができます。

• [表示]メニューから [コード -> トレース]を選択し、[トレース]ウィンドウを開いてください。main()関数先頭ま でスクロールアップしてください。



🐠 トレース								_ [	IJ×
Image: -0035897, 0000000       File:       Cycle: -0035207       Address: FFFF9042       Time: 00:00:00.000.021.420									
PTR	Label	Address	Time Stamp	PSW	Instruction	)	Interrupt	Access Data	
-0035207	_main	FFFF9042	00:00:00.000.021.420	0PUIC	ADD	#-30H,R0,R0	-	USP<-00001A5C	
-0035206		FFFF9045	00:00:00.000.021.430	0PUIC	MOY.L	#-00007BCCH,R5	-	R5<-FFFF8434	
-0035205		FFFF904B	00:00:00.000.021.440	0PUIC	SUB	#4H, RO	-	USP<-00001A58	
-0035204		FFFF904D	00:00:00.000.021.450	0PUIC	MOV.L	R5,[R0]	-	00001A58<-FFFF8434	
-0035203		FFFF904F	00:00:00.000.021.480	0PUIC	BSR.A	_printf	-	00001A54<-FFFF9053	
-0035202	_printf	FFFF9312	00:00:00.000.021.490	OPUIC	MOV.L	#0H, R5	-	R5<-00000000	
-0035201		FFFF9314	00:00:00.000.021.500	OPUIC	PUSH.L	R5	-	00001A50<-00000000	
-0035200		FFFF9316	00:00:00.000.021.510	OPUI	ADD	#08H,R0,R4	-	R4<-00001A58	
-0035199		FFFF9319	00:00:00.000.021.520	OPUI	ADD	#7H,R4	-	R4<-00001A5F	
-0035198		FFFF931B	00:00:00.000.021.530	OPUI	MOV.L	08H[RO],R3	-	R3<-FFFF8434	
-0035197		FFFF931D	00:00:00.000.021.540	OPUI	AND	#-04H,R4	-	R4<-00001A5C	
-0035196		FFFF9320	00:00:00.000.021.550	0PUI	MOY.L	#00001590H,R2	-	R2<-00001590	
-0035195		FFFF9326	00:00:00.000.021.560	OPUI	MOV.L	#-00006D0BH,R1	-	R1<-FFFF92F5	
-0035194		FFFF932C	00:00:00.000.021.590	0PUI	BSR.A	Printf	-	00001A4C<-FFFF9330	
-0035193	Printf	FFFF9675	00:00:00.000.021.640	0PUI	PUSHM	R6-R10	-	00001A48<-00000000	
-0035192		FFFF9677	00:00:00.000.021.650	OPUIC	ADD	#-00A4H,R0,R0	-	USP<-00001994	
-0035191		FFFF967B	00:00:00.000.021.660	OPUIC	MOV.L	R2,98H[R0]	-	00001A2C<-00001590	•

図 7-14 トレースウィンドウ(トレース情報の表示)

## 7.3.3 トレース検索の実行

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

検索		<u> </u>
組合せ( <u>C</u> ):		
<ul> <li>PTR</li> <li>Address</li> <li>Time Stamp</li> <li>✓ Instruction</li> <li>Interrupt</li> </ul>	文字列⑤):  BRA □ 指定条件以外⊗	前方検索(⊻) 【後方検索(E)】
人 検索設定内容(E):	J	
[Instruction] BRA		新規(₩)
		削除( <u>D</u> )
		すべて削除(L)
履歴(0):		
		追加(益)
		閉じる

図 7-15 トレース検索ダイアログボックス

[組合せ]欄で検索の対象とする条件を選択して、チェックボックスをチェックしてください。 [検索項目]欄で選択した条件に対応する詳細項目を指定します。

設定した検索条件は[検索設定内容]リストボックスに表示します。 検索条件を設定後、[前方検索] または[後方検索]ボタンのクリックで検索を開始します。 検索の結果一致するレコードが見つかった場合は、当該レコードを強調表示します。

ー致するレコードが見つかった場合は、ポップアップメニューで[前方検索] または[後方検索]を選択すると、次のレ コードを検索できます。

🐠 トレース									<u>- 🗆 ×</u>
Image:         Image:									
PTR	Label	Address	Time Stamp	PSW	Instruct	ion	Interrupt	Access Data	
-0035886		FFFF935F	00:00:00.000.000.200	0	BRA.B	next_loop1	-	PC<-FFFF936F	
-0035885	next_loop1	FFFF936F	00:00:00.000.000.210	0C	CMP	R4,R5	-		
-0035884		FFFF9371	00:00:00.000.000.240	0C	BGTU.B	loop1	-	PC<-FFFF9361	
-0035883	loop1	FFFF9361	00:00:00.000.000.250	0C	MOV.L	[R4+],R1	-	R4<-FFFF8578 R1	<-000
-0035882		FFFF9364	00:00:00.000.000.270	0C	MOV.L	[R4+],R3	-	R4<-FFFF857C R3	<-000
-0035881		FFFF9367	00:00:00.000.000.280	0C	CMP	R1,R3	-		
-0035880		FFFF9369	00:00:00.000.000.290	0C	BLEU.B	next_loop1	-		
-0035879		FFFF936B	00:00:00.000.000.300	0C	SUB	R1,R3	-	R3<-00000458	
-0035878		FFFF936D	00:00:00.000.003.100	0C	SSTR.B		-	0000152F<-00 R	1<-00
-0035877	next_loop1	FFFF936F	00:00:00.000.003.110	0C	CMP	R4,R5	-		
-0035876		FFFF9371	00:00:00.000.003.140	0C	BGTU.B	loop1	-	PC<-FFFF9361	
-0035875	loop1	FFFF9361	00:00:00.000.003.150	0C	MOV.L	[R4+],R1	-	R4<-FFFF8580 R1	<-000
-0035874		FFFF9364	00:00:00.000.003.170	0C	MOV.L	[R4+],R3	-	R4<-FFFF8584 R3	<-000
-0035873		FFFF9367	00:00:00.000.003.180	0ZC	CMP	R1,R3	-		
-0035872		FFFF9369	00:00:00.000.003.200	0ZC	BLEU.B	next_loop1	-	PC<-FFFF936F	
-0035871	next_loop1	FFFF936F	00:00:00.000.003.210	0C	CMP	R4,R5	-		
-0035870		FFFF9371	00:00:00.000.003.240	0C	BGTU.B	loop1	-	PC<-FFFF9361	-

図 7-16 トレースウィンドウ(検索結果)

### 7.3.4 I/Oシミュレーションの確認

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

### Data Input ###	٦				
a[0]=0					
a[1]=21468					
a[2]=9988					
a[3]=22117					
a[4]=3498					
a[5]=16927					
a[6]=16045					
a[7]=19741					
a[8]=12122					
a[9]=8410					

図 7-17 I/O シミュレーションウィンドウ

• [I/O シミュレーション]ウィンドウは閉じないでください。



7.チュートリアル

#### 7.3.5 ブレークポイントの確認

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

• [表示]メニューから [コード -> イベントポイント]を選択してください。

<b>B</b> a	$\mathbb{Z} \times  _{\mathbb{Z}}$	<b>Z</b>			
т	s	Condition Action			
BP Enable PC=FFFF90B3 (Tutorial.c/39)			Stop		
•				F	
Software Break / Software Event /					

図 7-18 イベントポイントウィンドウ

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

[イベントポイント]ウィンドウを閉じてください。

#### 7.3.6 変数の参照

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

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

以下の [シンボル登録]ダイアログボックスを表示します。

?×
<u>O</u> K
キャンセル(©)

図 7-19 シンボル登録ダイアログボックス

• 配列 "a" をタイプし、[OK]ボタンをクリックします。

[ウォッチ]ウィンドウに、long型の配列 a を表示します。 配列 a の前にあるシンボル+をクリックすると、配列を拡張して表示します。
🧆 ট্বস্ট					_ 🗆 ×
RR	- 1	/ 🐴 🗙 🛃	<b>r</b> P <b>r</b> .		
Name		Value	Address	Туре	Scope
⊡… R a			{ 00001A5C }	(long[10])	[Current Scope]
R	[0]	н'00000000	{ 00001A5C }	(long)	
R	[1]	H'000053dc	{ 00001A60 }	(long)	
R	[2]	н'00002704	{ 00001A64 }	(long)	
R	[3]	н'00005665	{ 00001A68 }	(long)	
R	[4]	H'00000daa	{ 00001A6C }	(long)	
R	[5]	H'0000421f	{ 00001A70 }	(long)	
R	[6]	H'00003ead	{ 00001A74 }	(long)	
R	[7]	H'00004d1d	{ 00001A78 }	(long)	
R	[8]	H'00002f5a	{ 00001A7C }	(long)	
R	[9]	H'000020da	{ 00001A80 }	(long)	
<b>▲</b> ► \ Wa	tch1 🗸 🕅	/atch2 👌 Watch3 👌 ۱	Watch4 /		

図 7-20 ウォッチウィンドウ

[ウォッチ]ウィンドウを閉じてください。



7.チュートリアル

## 7.3.7 プログラムのステップ実行

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

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

- (1) [ステップイン]の実行
- [ステップイン]はコール関数の中に入り、コール関数の先頭のステートメントで停止します。
  - sort 関数の中に入るために、[デバッグ]メニューから [ステップイン]を選択するか、またはツールバーの [ステッ プイン]ボタンをクリックしてください。

		図 7-21 ステップイブホタブ
📣 Tutorial.c		
	カバレジ S.	
31         FFFF9055           32         FFFF9061           33         FFFF906A           34         FFFF9071           35         36           36         FFFF9079           37         FFFF9083           38         39           40         FFFF9083           41         FFFF90C8           42         FFFF90C4           43         44           45           46	•	<pre>for( i=0; i&lt;10; i++ ){     j = rand();     if(j &lt; 0){         j = -j;     }     a[i] = j;     printf("a[%d]=%ld¥n",i,a[i]);     }     sort(a);     printf("*** Sorting results ***¥n");     for( i=0; i&lt;10; i++ ){         printf("a[%d]=%ld¥n",i,a[i]);     }     change(a); }</pre>
47 FFFF910D 48 49 50 51 52 FFFF9112 53 FFFF9117 54 FFFF9121 55 FFFF9132 56 FFFF9146 57 FFFF9159 58 FFFF9178 59 FFFF9185 ◀		<pre>void sort(long *a) {     long t;     int i, j, k, gap;     gap = 5;     while( gap &gt; 0 ){         for( k=0; k<gap; ){="" for(="" for(j="i-gap;" i="i+gap" i<10;="" j="" k++){="">=k; j=j-gap){</gap;></pre>

- 図 7-22 ソースウィンドウ(ステップイン)
- [ソース]ウィンドウの PC 位置表示 ( )が、sort 関数の先頭のステートメントに移動します。

(2) [ステップアウト]の実行

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

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

🚸 Tutorial.c		
行番 ソースアドレス カバレジ	S ソース	
31         FFFF9055           32         FFF9061           33         FFF9064           34         FFF9071           35         36           38         FFF9083           39         FFF9083           39         FFF9084           40         FFF9083           40         FFF9084           41         FFF9084           42         FFF9084           43         FFF9084           44         FFF9084           45         46           47         FFF9004           43         FFF9104           45         50           51         52           52         FFFF9104           45         46           47         FFFF9104           48         49           50         51           52         FFFF9104           48         49           50         51           52         FFFF9112           53         FFFF9132           56         FFFF9146           57         FFF9159           58         FFFF9178           59	<pre>for( i=0; i&lt;10; i++ ){     j = rand();     if(j &lt; 0){         j = -j;     }     a[i] = j;     printf("a[%d]=%Id¥n",i,a[i]);     sort(a);     printf("#** Sorting results ****¥n");     for( i=0; i&lt;10; i++ ){         printf("a[%d]=%Id¥n",i,a[i]);     }     change(a); } void sort(long *a) {     long t;     int i, j, k, gap;     gap = 5;     while( gap &gt; 0 ){         for( i=k+gap; i&lt;10; i=i+gap ){             for(j=i-gap; j&gt;=k; j=j-gap){</pre>	
•		• //

⑦ 7-23 ステップアウトボタン

図 7-24 ソースウィンドウ(ステップアウト)

(3) [ステップオーバ]の実行

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

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

🚸 Tuto	orial.c				
	] 💭				
行番	ソースアドレス	カバレジ	S	ソース	
31 32 334 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58	FFFF9055 FFFF9061 FFFF9064 FFFF9071 FFFF9083 FFFF9083 FFFF9083 FFFF9088 FFFF9088 FFFF9088 FFFF9088 FFFF9088 FFFF9088 FFFF9088 FFFF9088 FFFF9088 FFFF9088 FFFF9104 FFFF9104 FFFF9107 FFFF9121 FFFF9122 FFFF9128 FFFF9178		•	<pre>for( i=0; i&lt;10; i++ ){     j = rand();     if(j &lt; 0){         j = -j;     }     a[i] = j;     printf("a[%d]=%ld¥n",i,a[i]);     }     sort(a);     printf("*** Sorting results ***¥n");     for( i=0; i&lt;10; i++ ){         printf("a[%d]=%ld¥n",i,a[i]);     }     change(a); } void sort(long *a) {     long t;     int i, j, k, gap;     gap = 5;     while( gap &gt; 0 ){         for( i=k+gap; i&lt;10; i=i+gap ){             for( j=i-gap; j&gt;=k; j=j-gap){</pre>	
1	FFFF9185			a[J] = a[J+gap];	

⑦→
図 7-25 ステップオーバボタン

図 7-26 ソースウィンドウ(ステップオーバ)

printf 関数を実行すると、[I/O シミュレーション]ウィンドウに\*\*\* Sorting results \*\*\*を表示します。



## 7.3.8 プロファイラ情報の確認

プロファイラ情報を[プロファイル]ウィンドウで確認することができます。

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

#### (1) [List]シート

プロファイラ情報をリスト形式で表示します。

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

אראקלטל 💶 🔍								
E -  Bar Show Functions/Variables								
Function/Variable	F/V	Address	Size	Times	Cycle	Ext mem	I/O area	Int mem 📃 📥
fwrite	F	FFFF9399	H'000000CF	183	24459	0	0	6348
INITSCT	F	FFFF934F	н'00000000	1	969	0	0	32
_rand	F	FFFF9333	H'0000001C	10	110	0	0	30
printf	F	FFFF9312	н'00000021	22	374	0	0	198
FFFF92F5	F	FFFF92F5	н'00000000	183	3477	0	0	2013
fclose	F	FFFF92A2	н'00000053	3	120	0	0	39
_freopen	F	FFFF9274	H'0000002E	3	93	0	0	60
_change	F	FFFF920A	H'0000006A	1	424	0	0	166
_sort	F	FFFF910D	H'000000FD	1	1869	0	0	774
_main	F	FFFF9042	Н'000000СВ	1	717	0	0	271
_write	F	FFFF8EE2	Н'0000008В	249	15438	0	0	5478
_close	F	FFFF8ED9	н'00000009	3	21	0	0	6
_open	F	FFFF8E44	н'00000095	3	192	0	0	39
CLOSEALL	F	FFFF8DF7	H'0000004D	1	470	0	0	144
INIT_IOLIB	F	FFFF8CD8	H'0000011F	1	89	0	0	31
_charput	F	FFFF8C98	н'00000000	249	2739	0	0	747
List Tree								

図 7-27 List シート (プロファイルウィンドウ)

\_\_fclose 関数を 3 回コールし、実行サイクルは 120 サイクル、内部メモリを 39 回アクセスしたことがわかります。 コール回数が多い関数や遅いメモリを多くアクセスする関数など、プログラム性能のクリティカルパスを探すことが できます。

### (2) **[Tree]シート**

プロファイラ情報をツリー形式で表示します。

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

♣วือวะ4ม								_ 🗆 ×
🗈 🗲 🔚 Show Functions/Variables	V	0						
Function	Address	Size	Stack Size	Times	Cycle	Ext mem	I/O area	Int mem 🔺
⊨main	FFFF9042	н'000000св	н'00000000	1	717	0	0	271
printf	FFFF9312	H'00000021	н'00000000	22	374	0	0	198
rand	FFFF9333	H'000001C	н'00000000	10	110	0	0	30
	FFFF920A	H'0000006A	н'00000000	1	424	0	0	166
sort	FFFF910D	H'000000FD	н'00000000	1	1869	0	0	774
CLOSEALL	FFFF8DF7	H'0000004D	н'00000000	1	470	0	0	144
INIT_IOLIB	FFFF8CD8	H'0000011F	н'00000000	1	89	0	0	31
⊟freopen	FFFF9274	H'0000002E	н'00000000	3	93	0	0	60
⊨fclose	FFFF92A2	н'00000053	н'00000000	3	120	0	0	39
fflush	FFFF9468	H'0000007E	н'00000000	3	54	0	0	12
	FFFF9624	н'00000051	н'00000000	3	66	0	0	36
_close	FFFF8ED9	н'00000009	н'00000000	3	21	0	0	6
⊡Foprep	FFFF94E6	H'000000E8	н'00000000	3	421	0	0	87
Fopen	FFFF9FA7	н'00000032	н'00000000	3	84	0	0	9
List Tree								

図 7-28 Tree シート(プロファイルウィンドウ)

\_\_close 関数が\_fclose 関数から 3 回コールされ、その時の実行サイクルは 21 サイクル、内部メモリを 6 回アクセスした ことがわかります。



(3) プロファイル - チャートウィンドウ

[プロファイル - チャート]ウィンドウで関数の呼び出し関係を表示します。

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



図 7-29 プロファイル - チャートウィンドウ

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

以上で、シミュレータ・デバッガを使用したチュートリアルは終了です。





RX ファミリ シミュレータ/デバッガ V.1.01 ユーザーズマニュアル

- 発行年月日: 2010年4月1日 Rev.1.00
- 発行: 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
- 編集:株式会社ルネサス ソリューションズ

© 2010 Renesas Electronics Corporation, All rights reserved. Printed in Japan.

# RX ファミリ シミュレータ/デバッガ V.1.01 ユーザーズマニュアル

