User's Manual

CS+ V3.00.00

統合開発環境

ユーザーズマニュアル RH850 デバッグ・ツール編

対象デバイス

RH850 ファミリ

本資料に記載の全ての情報は発行時点のものであり、ルネサス エレクトロニクスは、
予告なしに、本資料に記載した製品または仕様を変更することがあります。
ルネサス エレクトロニクスのホームページなどにより公開される最新情報をご確認ください。

ルネサス エレクトロニクス
www.renesas.com

Rev.1.00　2014.08

ご注意書き

1. 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・システムの設計において、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て、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。
2. 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。
3. 本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害に関し、当社は、何らの責任を負うものではありません。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。
4. 当社製品を改造、改変、複製等しないでください。かかる改造、改変、複製等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、
   各品質水準は、以下に示す用途に製品が使用されることを意図しております。

   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、産業用ロボット等

   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、各種安全装置等

   当社製品は、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（原子力制御システム、軍事機器等）に使用されることを意図しておらず、使用することはできません。たとえ、意図し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損害が生じても、当社は一切その責任を負いません。なお、ご不明点がある場合は、当社営業にお問い合わせください。
6. 当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の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。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、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、社会的損害等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、お客様の機器・システムとしての出荷保証を行ってください。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、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の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。
9. 本資料に記載されている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・システムに使用することはできません。また、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に使用しないでください。当社製品または技術を輸出する場合は、「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。
10. お客様の転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。
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禁じます。

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。

(2012. 4)

このマニュアルの使い方

このマニュアルは，RH850 ファミリ用アプリケーション・システムを開発する際の統合開発環境である CS+ について説明します。

CS+ は，RH850 ファミリの統合開発環境（ソフトウエア開発における，設計，実装，デバッグなどの各開発フェーズに必要なツールをプラットフォームである IDE に統合）です。統合することで，さまざまなツールを使い分ける必要がなく，本製品のみを使用して開発のすべてを行うことができます。

対象者	このマニュアルは，CS+ を使用してアプリケーション・システムを開発するユーザを対象としています。
目的	このマニュアルは，CS+ の持つソフトウエア機能をユーザに理解していただき，これらのデバイスを使用するシステムのハードウエア，ソフトウエア開発の参照要資料として役立つことを目的としています。
構成	このマニュアルは，大きく分けて次の内容で構成しています。 1. 概　　説 2. 機　　能 A. ウインドウ・リファレンス
読み方	このマニュアルを読むにあたっては，電気，論理回路，マイクロコンピュータに関する一般知識が必要となります。 凡例　データ表記の重み　：左が上位桁，右が下位桁 　　アクティブ・ロウの表記　：XXX （端子，信号名称に上線） 　　注　：本文中についた注の説明 　　注意　：気をつけて読んでいただきたい内容 　　備考　：本文中の補足説明 　　数の表記　：10 進数 ... XXXX 　　16 進数 ... 0xXXXX
関連資料	関連資料は暫定版の場合がありますが，この資料では「暫定」の表示をしておりません。あらかじめご了承ください。

注意　上記資料は，予告なしに内容を変更することがあります。設計などには，必ず最新の資料を使用してください。

目次

1. 概　　説

CS+ は，RH850 ファミリ，RX ファミリ，V850 ファミリ，RL78 ファミリ，78K0R マイクロコントローラ，78K0 マイクロコントローラ用の統合開発環境プラットフォームです。
CS+ では，設計／コーディング／ビルド／デバッグ／フラッシュ・プログラミングなど，プログラムの開発における一連の作業を行うことができます。
本マニュアルは，こうした一連のプログラムの開発工程のうち，デバッグ工程について説明します。

この章では，CS+ が提供するデバッグ機能の概要について説明します。

1.1 概　　要

CS+ が提供するデバッグ機能を使用することにより，RH850 ファミリ用に開発されたプログラムを，効率良くデバッグすることができます。

1.2 特　　長

次に，CS+ が提供するデバッグ機能の特長を示します。

- マルチコア対応版における同期実行／同期ブレーク
  使用するマイクロコントローラがマルチコア対応版の場合，同期実行／同期ブレークを行います。
  コアの選択を切り替えることにより，コアごとの情報がパネルに表示されます。

- 各種デバッグ・ツールとの接続
  フルスペック・エミュレータ（Full-spec emulator），オンチップ・デバッギング・エミュレータ（E1/E20），およびシミュレータと組み合わせて使用することにより，より快適な開発環境を実現できます。

- C ソース・テキストと逆アセンブル・テキストの混合表示
  1 つのパネル上で，C ソース・テキストと逆アセンブル・テキストを混合表示することができます。

- ソース・レベル・デバッグと命令レベル・デバッグ
  C ソース・プログラムに対して，ソース・レベル・デバッグ，または命令レベル・デバッグを行うことができます。

- フラッシュ・セルフ・プログラミング・エミュレーション機能の対応 （コード・フラッシュ）
  フラッシュ・セルフ・プログラミング・エミュレーション機能のフラッシュ・セルフ・ライブラリを使用し，コード・フラッシュの書き換えを行うことができます。

- リアルタイム表示更新機能
  プログラムの実行が停止した際に，表示情報を自動的に更新するだけでなく，プログラムが実行中の状態であっても，リアルタイムにメモリ／レジスタ／変数の値を表示更新することができます。

- デバッグ環境の保存／復元
  ブレークポイントやイベントの設定情報，ファイルのダウンロード情報，パネルの表示状態／位置などのデバッグ環境を保存することができます。

2. 機　　能

この章では，CS+ を使用したデバッグの手順，およびデバッグに関する主な機能について説明します。

2.1 概　　要

CS+ を使用した，プログラムの基本的なデバッグ手順は次のとおりです。

(1) CS+ を起動する
Windows の［スタート］メニューから CS+ を起動します。

備考　“CS+ を起動する ” についての詳細は，「CS+ 統合開発環境 ユーザーズマニュアル プロジェクト操作編」を参照してください。

(2) プロジェクトを設定する
プロジェクトの新規作成，または既存のプロジェクトの読み込みを行います。

備考　“ プロジェクトを設定する ” についての詳細は，「CS+ 統合開発環境 ユーザーズマニュアル プロジェクト操作編」を参照してください。

(3) ロード・モジュールを作成する
アクティブ・プロジェクトの設定，および使用するビルド・ツールの設定を行ったのち，ビルドを実行することにより，ロード・モジュールを作成します。

備考　CC-RH を使用して “ ロード・モジュールを作成する ” 場合についての詳細は，「CS+ 統合開発環境 ユーザーズマニュアル RH850 ビルド編」を参照してください。

(4) ホスト・マシンとの接続を確認する
ホスト・マシンに，使用するデバッグ・ツール（Full-spec emulator/E1/E20/ シミュレータ）を接続します。

(5) 使用するデバッグ・ツールを選択する
プロジェクトで使用するデバッグ・ツールを選択します。

(6) デバッグ・ツールの動作環境設定を行う
(5) で選択したデバッグ・ツールの動作環境を設定します。

- 【Full-spec emulator】の場合
- 【E1】の場合
- 【E20】の場合
- 【シミュレータ】の場合

(7) デバッグ・ツールを接続する
CS+ とデバッグ・ツールの通信を開始します。

(8) ダウンロードを実行する
(3) で作成したロード・モジュールを，デバッグ・ツールへダウンロードします。

(9) ソース・ファイルを表示する
ダウンロードしたロード・モジュールの内容（ソース・ファイル）をエディタ パネル，または逆アセンブル パネルで表示します。

(10) プログラムを実行する
目的に応じた実行方法により，プログラムを実行します。
なお，実行したプログラムを任意の箇所で停止する場合は，あらかじめブレークポイント／ブレーク・イベント[注]を設定しておきます（「2.9.3 任意の場所で停止する（ブレークポイント）」／「2.9.4 任意の場所で停止する（ブレーク・イベント）」／「2.9.5 変数 /I/O レジスタへのアクセスで停止する」参照）。

注　使用するデバッグ・ツールにイベントを設定することにより実現する機能です。イベントを設定する際には，「2.16.6 イベント設定に関する留意事項」を参照してください。

備考　選択しているマイクロコントローラがマルチコア対応版の場合，プログラムを実行する前に，デバッグ対象とするコア（PE：プロセッサ・エレメント）を選択してください（「2.7 コア（PE）の選択」参照）。

(11) プログラムの実行を手動で停止する
実行したプログラムを停止します。
ただし，(10) でブレークポイント／ブレーク・イベントを設定している場合，設定したブレーク条件が満たされると同時にプログラムの実行は自動的に停止します。

(12) プログラムの実行結果を確認する
プログラムを実行することにより取得した各種情報を確認します。

- メモリ，レジスタ，変数の表示／変更
- スタックからの関数呼び出し情報の表示
- 実行履歴の収集[注]
- 実行時間の計測[注]
- カバレッジの測定【シミュレータ】

注　使用するデバッグ・ツールにイベントを設定することにより実現する機能です。イベントを設定する際には，「2.16.6 イベント設定に関する留意事項」を参照してください。

以後，必要に応じて (9) ～ (12) を繰り返すことによりデバッグ作業を進めます。
なお，この際に，プログラムに変更を加えた場合は，(3)，および (8) の操作も繰り返す必要があります。

備考 1.　上記のほか，次の機能を利用して，プログラムの実行結果の確認を行うことができます。

- プログラム内へのアクションの設定
- フック処理を設定する

備考 2.　取得した各種情報をファイルに保存することができます。

- 逆アセンブル結果の表示内容を保存する
- メモリの表示内容を保存する
- CPU レジスタの表示内容を保存する
- I/O レジスタの表示内容を保存する
- ローカル変数の表示内容を保存する
- ウォッチ式の表示内容を保存する
- コール・スタック情報の表示内容を保存する
- 実行履歴の表示内容を保存する

(13) アップロードを実行する
必要に応じ，プログラム（メモリ内容）を任意のファイル形式（インテル拡張ヘキサ・フォーマット／モトローラ・S タイプ・フォーマット／バイナリ・フォーマットなど）で保存します。

(14) デバッグ・ツールを切断する
CS+ とデバッグ・ツールとの通信を終了します。

(15) プロジェクト・ファイルを保存する
プロジェクトの設定情報をプロジェクト・ファイルに保存します。

備考　“ プロジェクト・ファイルを保存する ” についての詳細は，「CS+ 統合開発環境 ユーザーズマニュアル プロジェクト操作編」を参照してください。

2.2 デバッグを始める前の準備

この節では，作成したプログラムのデバッグを開始するための準備について説明します。

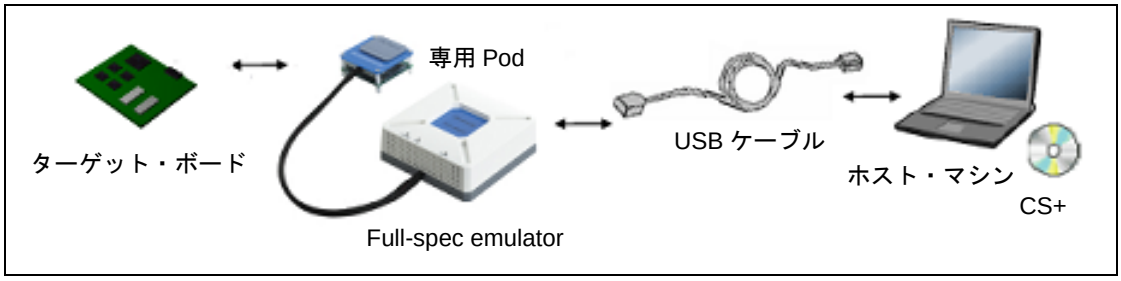
2.2.1 ホスト・マシンとの接続を確認する

使用するデバッグ・ツールごとに，ホスト・マシンとの接続例を示します。

2.2.1.1 【Full-spec emulator】の場合

ホスト・マシン，Full-spec emulator，および必要に応じてターゲット・ボードを接続します。
接続方法についての詳細は，Full-spec emulator のユーザーズ・マニュアルを参照してください。

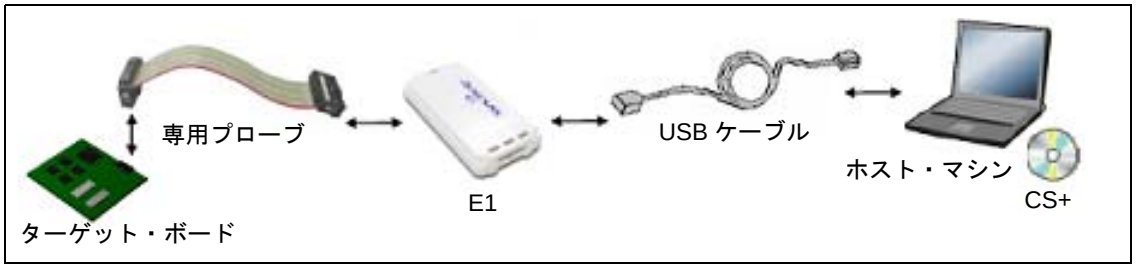
図 2.1　　ホスト・マシンとデバッグ・ツールとの接続例【Full-spec emulator】



2.2.1.2 【E1】の場合

ホスト・マシン，E1，および必要に応じてターゲット・ボードを接続します。
接続方法についての詳細は，E1 のユーザーズ・マニュアルを参照してください。

図 2.2　　ホスト・マシンとデバッグ・ツールとの接続例【E1】



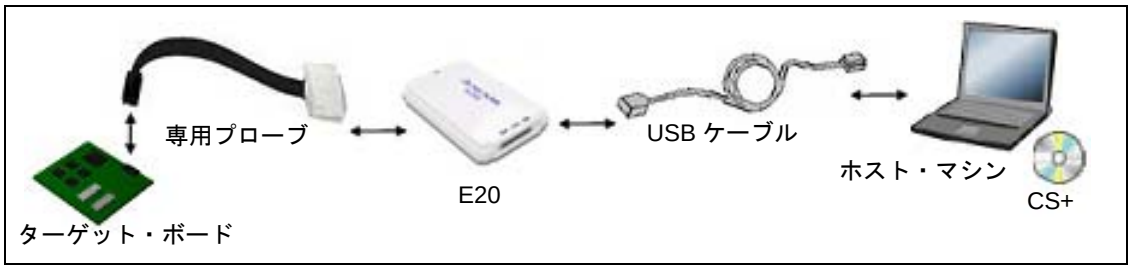
注意 1.　　ターゲット・ボードとの通信方式として，Low Pin Debug Interface（以降，LPD 通信方式と略します）のみをサポートしています。

注意 2.　　デバッグ MCU ボードを使用する場合の接続例については，デバッグ MCU ボードのユーザーズ・マニュアルを参照してください。

2.2.1.3 【E20】の場合

ホスト・マシン，E20，および必要に応じてターゲット・ボードを接続します。
接続方法についての詳細は，E20 のユーザーズ・マニュアルを参照してください。

図 2.3　　ホスト・マシンとデバッグ・ツールとの接続例【E20】



注意 1.　　ターゲット・ボードとの通信方式として，Low Pin Debug Interface（以降，LPD 通信方式と略します）のみをサポートしています。

注意 2.　　デバッグ MCU ボードを使用する場合の接続例については，デバッグ MCU ボードのユーザーズ・マニュアルを参照してください。

2.2.1.4 【シミュレータ】の場合

ホスト・マシンのみでデバッグ作業を行うことができます（エミュレータなどの接続は不要）。

図 2.4　　ホスト・マシンとデバッグ・ツールとの接続例【シミュレータ】

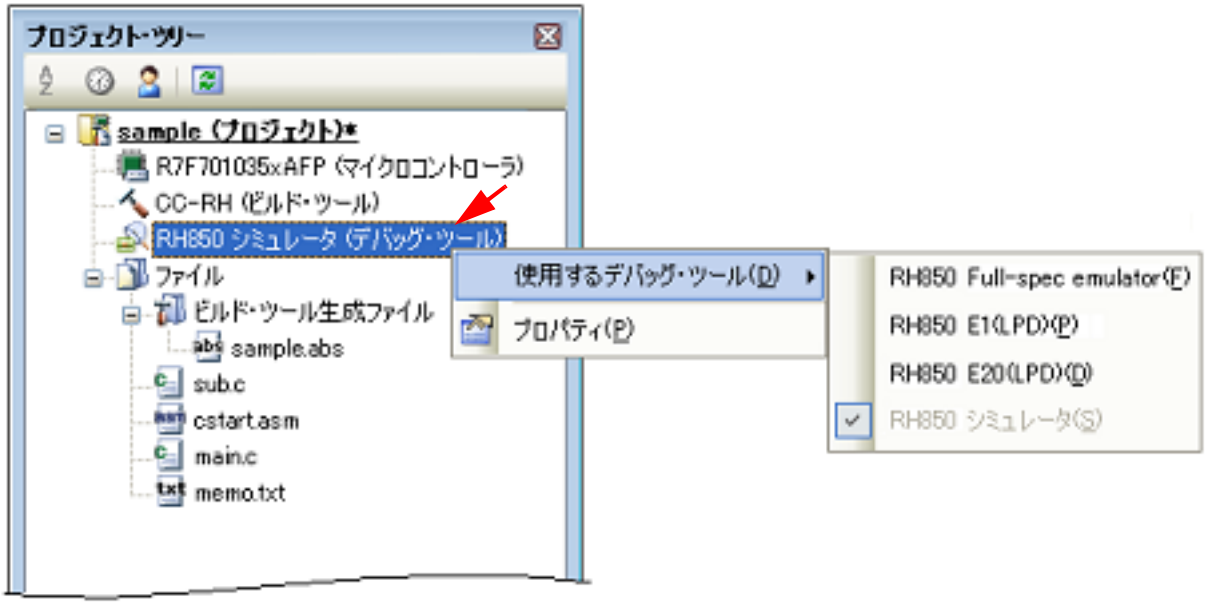
2.3 デバッグ・ツールの動作環境設定

この節では，各デバッグ・ツールの動作環境の設定方法について説明します。

2.3.1 使用するデバッグ・ツールを選択する

デバッグ・ツールの動作環境設定は，使用するデバッグ・ツールに対応したプロパティ パネルで行います。
そのため，まず，プロジェクト内で使用するデバッグ・ツールを選択します（使用するデバッグ・ツールはメイン・プロジェクト／サブプロジェクトごとに選択可）。
使用するデバッグ・ツールの選択／切り替えは，プロジェクト・ツリー パネル上の［RH850 *デバッグ・ツール名* (デバッグ・ツール)］ノードを右クリックすることで表示されるコンテキスト・メニューから行ってください。

図 2.5　　使用するデバッグ・ツールの選択／切り替え



注意　　表示されるコンテキスト・メニューは，プロジェクトで選択しているマイクロコントローラの種類により異なります。

すでにプロパティ パネルがオープンしている場合，再び［RH850 *デバッグ・ツール名* (デバッグ・ツール)］ノードをクリックすると，選択したデバッグ・ツールのプロパティ パネルに表示が切り替わります。
プロパティ パネルがオープンしていない場合では，同ノードをダブルクリックすることで，該当するプロパティ パネルがオープンします。

2.3.2 【Full-spec emulator】の場合

Full-spec emulator を使用する場合の動作環境の設定を次のプロパティ パネルで行います。

図 2.6 動作環境設定【Full-spec emulator】（プロパティ パネル）

プロパティ パネル上の該当するタブを選択し，次の設定を順次行ってください。

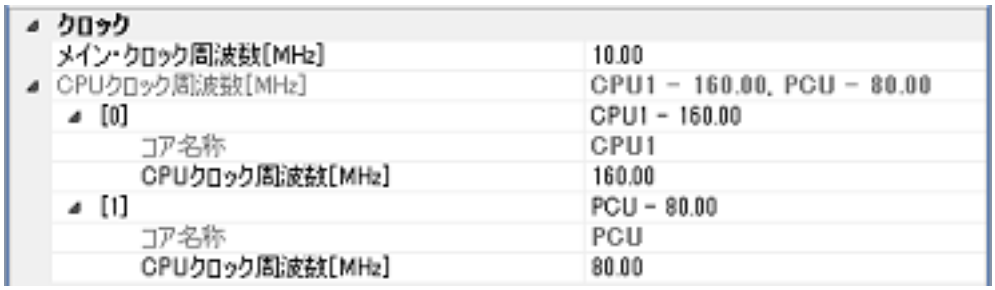
2.3.2.1 ［接続用設定］タブ

次に示すカテゴリごとに，デバッグ・ツールとの接続に関する設定を行います。

(1) ［クロック］
(2) ［ターゲット・ボードとの接続］
(3) ［フラッシュ］
(4) ［CPU 仮想化支援機能］

(1) ［クロック］
クロックに関する設定を行います。

図 2.7 ［クロック］カテゴリ【Full-spec emulator】



- ［メイン・クロック周波数 [MHz]］
メイン・クロック周波数（逓倍前）を指定します。
ドロップダウン・リストによる選択，または直接入力により，0.001 ～ 999.999（単位：MHz）の範囲の数値で指定してください（デフォルト：[10.00]）。

- [CPU クロック周波数 [MHz]]
CPU クロック周波数（逓倍後）をコアごとに指定します。
サブプロパティとして，選択しているマイクロコントローラが持つコア名称を表示します。

各コアの CPU クロック周波数を，ドロップダウン・リストによる選択，または直接入力により，0.001 ～ 999.999（単位：MHz）の範囲の数値で指定してください。
なお，表示されるコア名称の数，およびデフォルトの CPU クロック周波数は，選択しているマイクロコントローラの種類により異なります。

備考　CPU クロック周波数は，トレースのタイム・スタンプ情報を実時間に換算する際に使用されます。

(2)　［ターゲット・ボードとの接続］
Full-spec emulator とターゲット・ボードとの接続に関する設定を行います。

注意　Full-spec emulator が CS+ に接続している場合，このカテゴリ内のプロパティを変更することはできません。

図 2.8　［ターゲット・ボードとの接続］カテゴリ【Full-spec emulator】

- ［ターゲット・ボードを接続している］
Full-spec emulator にターゲット・ボードを接続しているか否かを選択します。
ターゲット・ボードと接続している場合は［はい］を選択してください（デフォルト：［いいえ］）。

(3)　［フラッシュ］
フラッシュ・セルフ・プログラミング機能に関する設定を行います。
ただし，このカテゴリは，選択しているマイクロコントローラがフラッシュ・セルフ・プログラミング機能をサポートしている場合のみ表示されます。

注意　Full-spec emulator が CS+ に接続している場合，このカテゴリ内のプロパティを変更することはできません。

図 2.9　［フラッシュ］カテゴリ

- ［Code Flash のセルフ・プログラミングを行う］
フラッシュ・セルフ・プログラミング機能のフラッシュ・セルフ・ライブラリを使用して，Code Flash の書き換えを行うか否かを選択します。
Code Flash の書き換えを行う場合は，［はい］を選択してください（デフォルト：［いいえ］）。
なお，［はい］を選択した場合，Code Flash のキャッシュは行われません。

- ［フラッシュ書き込み時にクロックを変更する］
デバッガ操作により，フラッシュ・メモリに書き込みを行う際に，一時的にクロック・スピードを変更するか否かを選択します。
フラッシュ・メモリへの書き込み性能を向上させるために，フラッシュ書き換え時にクロック・アップを行う場合は，［はい］を選択してください（デフォルト）。
［いいえ］を選択した場合，ユーザが設定したクロック・スピードでフラッシュ書き換えを行います。

注意　［はい］を選択した場合，CPU クロックだけではなく周辺クロックも変化するため，ブレーク中も動作している周辺システムに影響がある可能性があります。
［いいえ］を選択した場合，設定したクロック・スピードが低いと，デバッガ操作によるフラッシュ書き換え時間が長くなります。

(4)　［CPU 仮想化支援機能］
このカテゴリ内のプロパティは，常に無効です。

2.3.2.2　［デバッグ・ツール設定］タブ

次に示すカテゴリごとに，デバッグ・ツールの基本設定を行います。

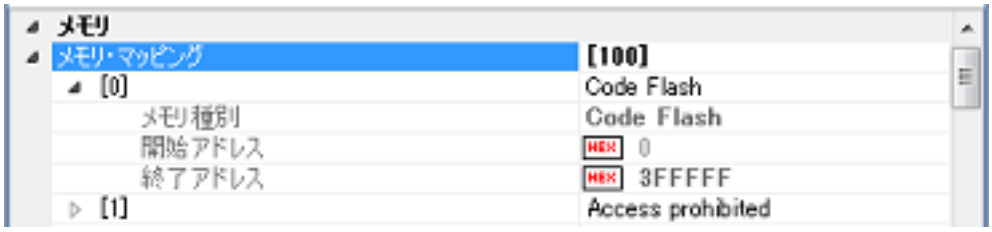
(1)　［メモリ］
(2)　［実行中のメモリ・アクセス］
(3)　［実行中のイベント設定］
(4)　［ブレーク］
(5)　［トレース］
(6)　［入力信号のマスク］

(1) ［メモリ］
メモリに関する設定を行います。

図 2.10 ［メモリ］カテゴリ【Full-spec emulator】

- ［メモリ・マッピング］
展開表示することにより，現在のメモリ・マッピングの状況が，メモリ領域の種別ごとに詳細表示されます。

図 2.11 メモリ・マッピング の詳細表示



注意 メモリ・マッピングの追加／削除はできません。

備考 選択しているマイクロコントローラがマルチコア対応版の場合，コア（PE）の選択を切り替えることにより，PE ごとのメモリ・マッピングの状況を表示します（「2.7 コア（PE）の選択」参照）。

- ［メモリ書き込み時にベリファイを行う］
メモリ書き込み時に，ベリファイを行うか否かを選択します。
［はい］を選択した場合，ダウンロードの際，またはメモリ パネル／ウォッチ パネルで値を変更した際にベリファイを行います（デフォルト）。

(2) ［実行中のメモリ・アクセス］
プログラム実行中におけるメモリ・アクセス（リアルタイム表示更新機能）に関する設定を行います。
リアルタイム表示更新機能についての詳細は，「2.10.1.4 プログラム実行中にメモリの内容を表示／変更する」を参照してください。

図 2.12 ［実行中のメモリ・アクセス］カテゴリ【Full-spec emulator】

- ［実行中にアクセスする］
プログラム実行中に内蔵 RAM 領域にアクセスするか否かを選択します。
アクセスを許可する場合は［はい］を選択してください（デフォルト：［いいえ］）。

- ［実行中に表示更新を行う］
プログラム実行中に，メモリ パネル／ウォッチ パネルの表示内容を更新するか否かを選択します。
表示内容の更新を行う場合は［はい］を選択してください（デフォルト）。

- ［表示更新間隔 [ms]］
このプロパティは，［実行中に表示更新を行う］プロパティにおいて［はい］を選択した場合のみ表示されます。
プログラム実行中に，メモリ パネル／ウォッチ パネルの表示内容を更新する間隔を指定します。
直接入力により，100 ～ 65500 の整数（100 ms 未満の端数切り上げ）を指定してください（デフォルト：［500］）。

(3) ［実行中のイベント設定］
このカテゴリでは，プログラム実行中におけるイベントの設定に関する設定を行います。

図 2.13 ［実行中のイベント設定］カテゴリ

- ［実行を一瞬停止してイベントを設定する］

プログラム実行中，またはトレーサ／タイマ動作中には設定することができないイベントを，プログラムの実行，またはトレーサ／タイマの動作を強制的に一瞬停止させることで設定を行うか否かを選択します。
このプロパティの対象となるイベント種別については，「2.16.6.2 実行中に設定／削除可能なイベント種別」を参照してください。
プログラム実行中，またはトレーサ／タイマ動作中に，イベントの設定を行う場合は［はい］を選択してください（デフォルト：［いいえ］）。

(4) ［ブレーク］
ブレーク機能に関する設定を行います。
ブレーク機能，およびこのカテゴリ内の設定についての詳細は，「2.9 プログラムの停止（ブレーク）」を参照してください。

(5) ［トレース］
トレース機能に関する設定を行います。
トレース機能，およびこのカテゴリ内の設定についての詳細は，「2.12 実行履歴の収集」を参照してください。

(6) ［入力信号のマスク］
入力信号のマスクに関する設定を行います。

図 2.14 ［入力信号のマスク］カテゴリ【Full-spec emulator】

- ［WAIT 信号をマスクする］
WAIT 信号をマスクするか否かを選択します。
WAIT 信号を Full-spec emulator に入力する場合は，［いいえ］を選択してください（デフォルト：［いいえ］）。

注意 ［接続設定用］タブ上の［ターゲット・ボードを接続している］プロパティにおいて［いいえ］を選択している場合，このプロパティは［はい］に固定となります（変更不可）。

- ［RESET 信号をマスクする］
RESET 信号をマスクするか否かを選択します。
RESET 信号を Full-spec emulator に入力する場合は，［いいえ］を選択してください（デフォルト：［いいえ］）。

注意 ［接続設定用］タブ上の［ターゲット・ボードを接続している］プロパティにおいて［いいえ］を選択している場合，このプロパティは［はい］に固定となります（変更不可）。

- ［マスクする RESET 信号の選択］
このプロパティは，［RESET 信号をマスクする］プロパティにおいて［はい］を選択した場合のみ表示されます。
マスクする RESET 信号を次のドロップダウン・リストより選択します。
  - TARGET RESET 信号（デフォルト）
  - TARGET RESET 信号と INTERNAL RESET 信号

2.3.2.3 ［ダウンロード・ファイル設定］タブ

デバッグ・ツールにダウンロードを実行する際の設定を行います。
各カテゴリ内の設定についての詳細は，「2.5.1 ダウンロードを実行する」を参照してください。

2.3.2.4 ［フラッシュ・オプション設定］タブ

マイクロコントローラに搭載されているフラッシュ・メモリのためのオプション設定を行います。
ただし，このタブは，選択しているマイクロコントローラがフラッシュ・オプションに対応している場合のみ表示されます。
なお，各オプションの設定はこのタブ上では行わず，［フラッシュ・オプション］カテゴリ内［フラッシュ・オプション］プロパティを選択すると欄内右端に表示される［...］ボタンをクリックすることによりオープンする，フラッシュ・オプションの設定 ダイアログ【Full-spec emulator】【E1】【E20】で行います（［...］ボタンは，デバッグ・ツールと接続している場合のみ表示されます）。
このダイアログにおいて，各項目の設定を完了したのち，［書き込み］ボタンをクリックしてください。
設定内容についての詳細は，フラッシュ・オプションの設定 ダイアログ【Full-spec emulator】【E1】【E20】を参照してください。

図 2.15　　フラッシュ・オプションの設定 ダイアログのオープン

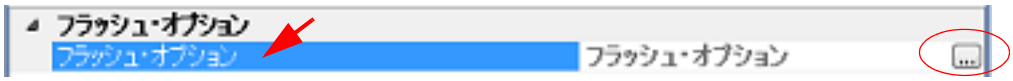
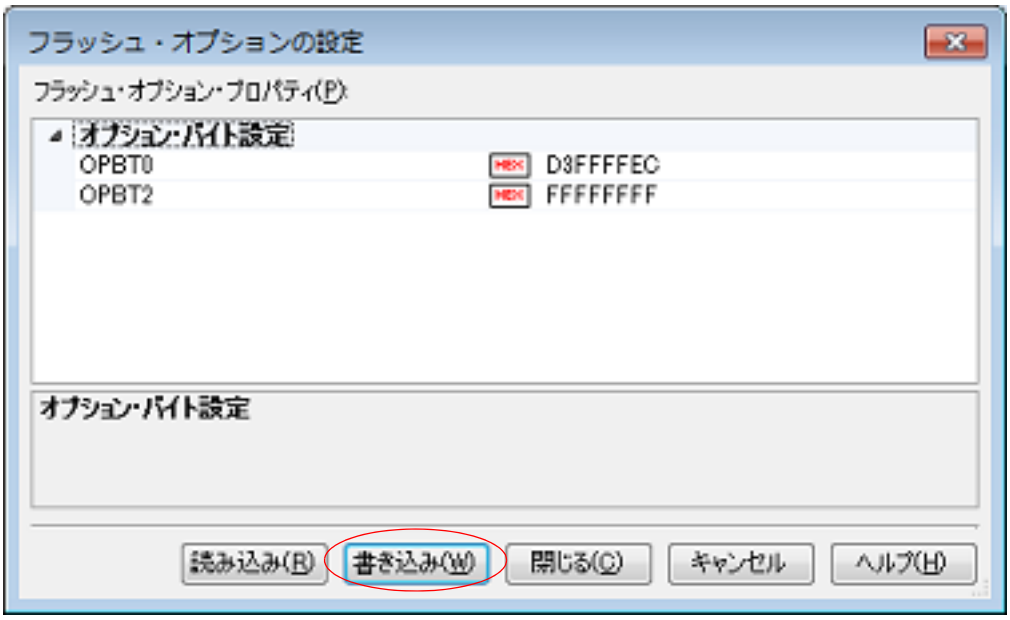


図 2.16　　フラッシュ・オプションの設定（フラッシュ・オプションの設定 ダイアログ）



2.3.2.5 ［フック処理設定］タブ

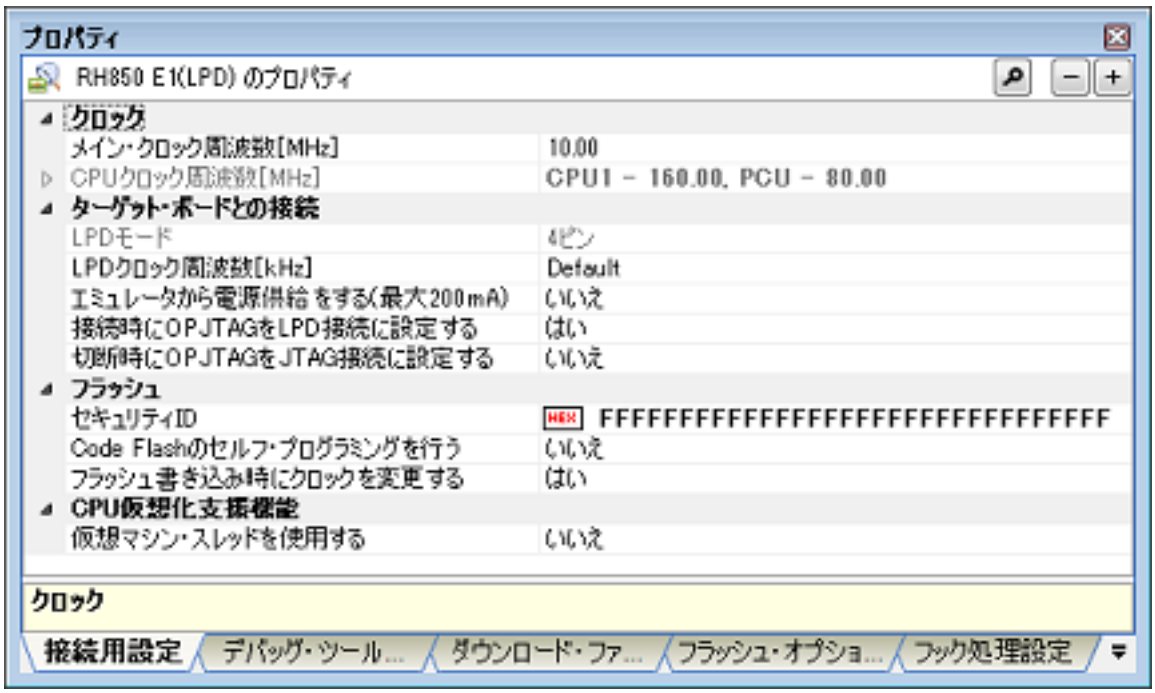
デバッグ・ツールにフック処理の設定を行います。
フック処理，および各カテゴリ内の設定についての詳細は，「2.17 フック処理を設定する」を参照してください。

2.3.3 【E1】の場合

E1 を使用する場合の動作環境の設定を次のプロパティ パネルで行います。

注意　　ターゲット・ボードとの通信方式として，LPD 通信方式のみをサポートしています。

図 2.17　　動作環境設定【E1】（プロパティ パネル）



プロパティ パネル上の該当するタブを選択し，次の設定を順次行ってください。

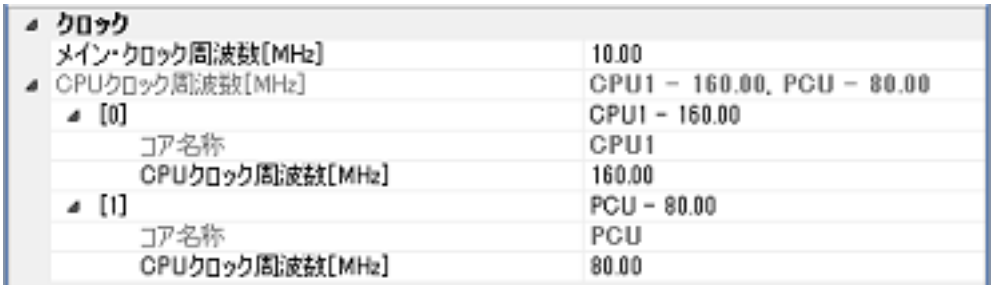
2.3.3.1 ［接続用設定］タブ

次に示すカテゴリごとに，デバッグ・ツールとの接続に関する設定を行います。

(1)　［クロック］

クロックに関する設定を行います。

図 2.18　　［クロック］カテゴリ【E1】



- [メイン・クロック周波数 [MHz]]
  メイン・クロック周波数（逓倍前）を指定します。
  ドロップダウン・リストによる選択，または直接入力により，0.001 ～ 999.999（単位：MHz）の範囲の数値で指定してください（デフォルト：[10.00]）。

- [CPU クロック周波数 [MHz]]
  CPU クロック周波数（逓倍後）をコアごとに指定します。
  サブプロパティとして，選択しているマイクロコントローラが持つコア名称を表示します。
  各コアの CPU クロック周波数を，ドロップダウン・リストによる選択，または直接入力により，0.001 ～ 999.999（単位：MHz）の範囲の数値で指定してください。
  なお，表示されるコア名称の数，およびデフォルトの CPU クロック周波数は，選択しているマイクロコントローラの種類により異なります。

  備考 CPU クロック周波数は，トレースのタイム・スタンプ情報を実時間に換算する際に使用されます。

(2) [ターゲット・ボードとの接続]
E1 とターゲット・ボードとの接続に関する設定を行います。

注意 E1 が CS+ に接続している場合，このカテゴリ内のプロパティを変更することはできません。

図 2.19 [ターゲット・ボードとの接続] カテゴリ【E1】

- [LPD モード]
  使用する LPD 通信方式のモードを選択します。
  ドロップダウン・リスト内に表示されるピン数は，選択しているマイクロコントローラの種類により異なります。
  ただし，選択できるモードが 1 つしか存在しない場合は，このプロパティを変更することはできません。

- [ボーレート [kbps]]
  このプロパティは，[LPD モード] プロパティにおいて，[1 ピン] を選択した場合のみ表示されます。
  LPD 通信方式の通信速度を選択します（デフォルト：[500]）。

- [LPD クロック周波数 [kHz]]
  このプロパティは，[LPD モード] プロパティにおいて，[4 ピン] を選択した場合のみ表示されます。
  LPD 通信方式のクロック周波数を選択します（デフォルト：[Default]）。
  なお，[Default] を選択した場合は，マイクロコントローラ固有のデフォルト値で接続処理を行います。

- [エミュレータから電源供給をする (最大 200mA)]
  E1 からターゲット・ボードへ電源を供給するか否かを選択します。
  電源を供給する場合は [はい] を選択してください（デフォルト：[いいえ]）。

- [供給電圧]
  このプロパティは，[エミュレータから電源供給をする (最大 200mA)] プロパティにおいて，[はい] を選択した場合のみ表示されます。
  ターゲット・ボードへ供給する電圧を選択します（デフォルト：[3.3V]）。

- [接続時に OPJTAG を LPD 接続に設定する]
  デバッグ・ツールとの接続時にシリアル・プログラミング・モードで起動し，オプション・バイトの設定を LPD 接続に変更するか否かを選択します。
  [はい] を選択した場合，接続時にシリアル・プログラミング・モードで起動し，OPJTAG をチェックします。この際に LPD 設定でない場合は LPD 設定に変更し，その後デバッグ・モードに移行します（デフォルト）。
  [いいえ] を選択した場合，接続時にデバッグ・モードで起動し，OPJTAG をチェックします。この際に LPD 設定でない場合はメッセージ・ダイアログを表示します。

- [切断時に OPJTAG を JTAG 接続に設定する]
  このプロパティは，[接続時に OPJTAG を LPD 接続に設定する] プロパティにおいて，[はい] を選択した場合のみ変更することができます。
  デバッグ・ツールとの接続を切断する際に，オプション・バイトの設定を JTAG 接続に変更するか否かを選択します。

切断時にオプション・バイトの設定を JTAG 接続に変更する場合は［はい］を選択してください。
なお，［いいえ］（デフォルト）を選択している場合，切断時にオプション・バイトを変更しないため，ピン・モードは LPD モード設定となります。

備考　　E1 との接続時に CS+ は，オプション・バイトが LPD モード設定になっていない場合，オプション・バイトの値を書き換えます。このため，マイクロコントローラのオプション・バイトが E1 との接続前と接続後で異なる可能性があります。

(3)　［フラッシュ］
フラッシュ書き換えに関する設定を行います。

注意　　E1 が CS+ に接続している場合，このカテゴリ内のプロパティを変更することはできません。

図 2.20　　［フラッシュ］カテゴリ【E1】

- ［セキュリティ ID］
このプロパティは，選択しているマイクロコントローラが，フラッシュ・メモリの ROM セキュリティ機能をサポートしている場合のみ表示されます。
内蔵 ROM，または内蔵フラッシュ・メモリ上のコードを読み出す際の ID コードを指定します。
直接入力により，32 桁の 16 進数（16 バイト）で指定します。
（デフォルト：[FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF]）

- ［Code Flash のセルフ・プログラミングを行う］
フラッシュ・セルフ・プログラミング機能のフラッシュ・セルフ・ライブラリを使用して，Code Flash の書き換えを行うか否かを選択します。
Code Flash の書き換えを行う場合は，［はい］を選択してください（デフォルト：［いいえ］）。
なお，［はい］を選択した場合，Code Flash のキャッシュは行われません。

- ［フラッシュ書き込み時にクロックを変更する］
デバッガ操作により，フラッシュ・メモリに書き込みを行う際に，一時的にクロック・スピードを変更するか否かを選択します。
フラッシュ・メモリへの書き込み性能を向上させるために，フラッシュ書き換え時にクロック・アップを行う場合は，［はい］を選択してください（デフォルト）。
［いいえ］を選択した場合，ユーザが設定したクロック・スピードでフラッシュ書き換えを行います。

注意　　［はい］を選択した場合，CPU クロックだけではなく周辺クロックも変化するため，ブレーク中も動作している周辺システムに影響がある可能性があります。
［いいえ］を選択した場合，設定したクロック・スピードが低いと，デバッガ操作によるフラッシュ書き換え時間が長くなります。

(4)　［CPU 仮想化支援機能］
このカテゴリ内のプロパティは，常に無効です。

2.3.3.2　［デバッグ・ツール設定］タブ

次に示すカテゴリごとに，デバッグ・ツールの基本設定を行います。

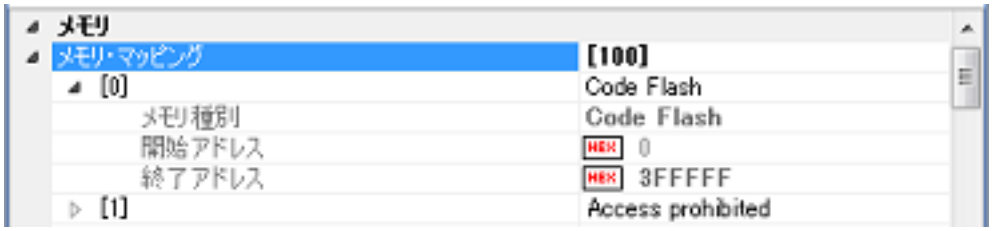
(1)　［メモリ］
(2)　［実行中のメモリ・アクセス］
(3)　［実行中のイベント設定］
(4)　［ブレーク］
(5)　［トレース］
(6)　［入力信号のマスク］

(1)　［メモリ］
メモリに関する設定を行います。

図 2.21　　［メモリ］カテゴリ【E1】

- [メモリ・マッピング]
展開表示することにより，現在のメモリ・マッピングの状況が，メモリ領域の種別ごとに詳細表示されます。

図 2.22 メモリ・マッピング の詳細表示



注意 メモリ・マッピングの追加／削除はできません。

備考 選択しているマイクロコントローラがマルチコア対応版の場合，コア（PE）の選択を切り替えることにより，PE ごとのメモリ・マッピングの状況を表示します（「2.7 コア（PE）の選択」参照）。

- [メモリ書き込み時にベリファイを行う]
メモリ書き込み時に，ベリファイを行うか否かを選択します。
[はい] を選択した場合，ダウンロードの際，またはメモリ パネル／ウォッチ パネルで値を変更した際にベリファイを行います（デフォルト）。

(2) [実行中のメモリ・アクセス]
プログラム実行中におけるメモリ・アクセス（リアルタイム表示更新機能）に関する設定を行います。
リアルタイム表示更新機能についての詳細は，「2.10.1.4 プログラム実行中にメモリの内容を表示／変更する」を参照してください。

図 2.23 [実行中のメモリ・アクセス] カテゴリ【E1】

- [実行中にアクセスする]
プログラム実行中に内蔵 RAM 領域にアクセスするか否かを選択します。
アクセスを許可する場合は [はい] を選択してください（デフォルト：[いいえ]）。

- [実行中に表示更新を行う]
プログラム実行中に，メモリ パネル／ウォッチ パネルの表示内容を更新するか否かを選択します。
表示内容の更新を行う場合は [はい] を選択してください（デフォルト）。

- [表示更新間隔 [ms]]
このプロパティは，[実行中に表示更新を行う] プロパティにおいて [はい] を選択した場合のみ有効となります。
プログラム実行中に，メモリ パネル／ウォッチ パネルの表示内容を更新する間隔を指定します。
直接入力により，100 ～ 65500 の範囲の整数（100 ms 未満の端数切り上げ）を指定してください（デフォルト：[500]）。

(3) [実行中のイベント設定]
このカテゴリでは，実行中のイベント設定に関する設定を行います。

図 2.24 [実行中のイベント設定] カテゴリ

- [実行を一瞬停止してイベントを設定する]
プログラム実行中，またはトレーサ／タイマ動作中には設定することができないイベントを，プログラムの実行，またはトレーサ／タイマの動作を強制的に一瞬停止させることで設定を行うか否かを選択します。
このプロパティの対象となるイベント種別については，「2.16.6.2 実行中に設定／削除可能なイベント種別」を参照してください。
プログラム実行中，またはトレーサ／タイマ動作中に，イベントの設定を行う場合は [はい] を選択してください（デフォルト：[いいえ]）。

(4) [ブレーク]
ブレーク機能に関する設定を行います。

ブレーク機能，およびこのカテゴリ内の設定についての詳細は，「2.9 プログラムの停止（ブレーク）」を参照してください。

(5) ［トレース］
トレース機能に関する設定を行います。
トレース機能，およびこのカテゴリ内の設定についての詳細は，「2.12 実行履歴の収集」を参照してください。

(6) ［入力信号のマスク］
入力信号のマスクに関する設定を行います。

図 2.25 ［入力信号のマスク］カテゴリ【E1】

- ［WAIT 信号をマスクする］
WAIT 信号をマスクするか否かを選択します。
WAIT 信号を E1 に入力しない場合は，［はい］を選択してください（デフォルト：［いいえ］）。

- ［RESET 信号をマスクする］
RESET 信号をマスクするか否かを選択します。
RESET 信号を E1 に入力しない場合は，［はい］を選択してください（デフォルト：［いいえ］）。

- ［マスクする RESET 信号の選択］
このプロパティは，［RESET 信号をマスクする］プロパティにおいて［はい］を選択した場合のみ表示されます。
マスクされる RESET 信号を表示します。
このプロパティを変更することはできません。

2.3.3.3 ［ダウンロード・ファイル設定］タブ

デバッグ・ツールにダウンロードを実行する際の設定を行います。
各カテゴリ内の設定についての詳細は，「2.5.1 ダウンロードを実行する」を参照してください。

2.3.3.4 ［フラッシュ・オプション設定］タブ

マイクロコントローラに搭載されているフラッシュ・メモリのためのオプション設定を行います。
ただし，このタブは，選択しているマイクロコントローラがフラッシュ・オプションに対応している場合のみ表示されます。
なお，各オプションの設定はこのタブ上では行わず，［フラッシュ・オプション］カテゴリ内［フラッシュ・オプション］プロパティを選択すると欄内右端に表示される［...］ボタンをクリックすることによりオープンする，フラッシュ・オプションの設定 ダイアログ【Full-spec emulator】【E1】【E20】で行います（［...］ボタンは，デバッグ・ツールと接続している場合のみ表示されます）。
このダイアログにおいて，各項目の設定を完了したのち，［書き込み］ボタンをクリックしてください。
設定内容についての詳細は，フラッシュ・オプションの設定 ダイアログ【Full-spec emulator】【E1】【E20】を参照してください。

図 2.26 フラッシュ・オプションの設定 ダイアログのオープン

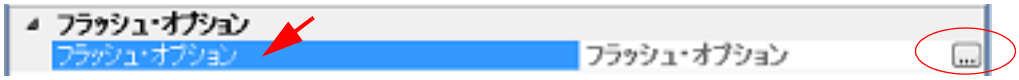
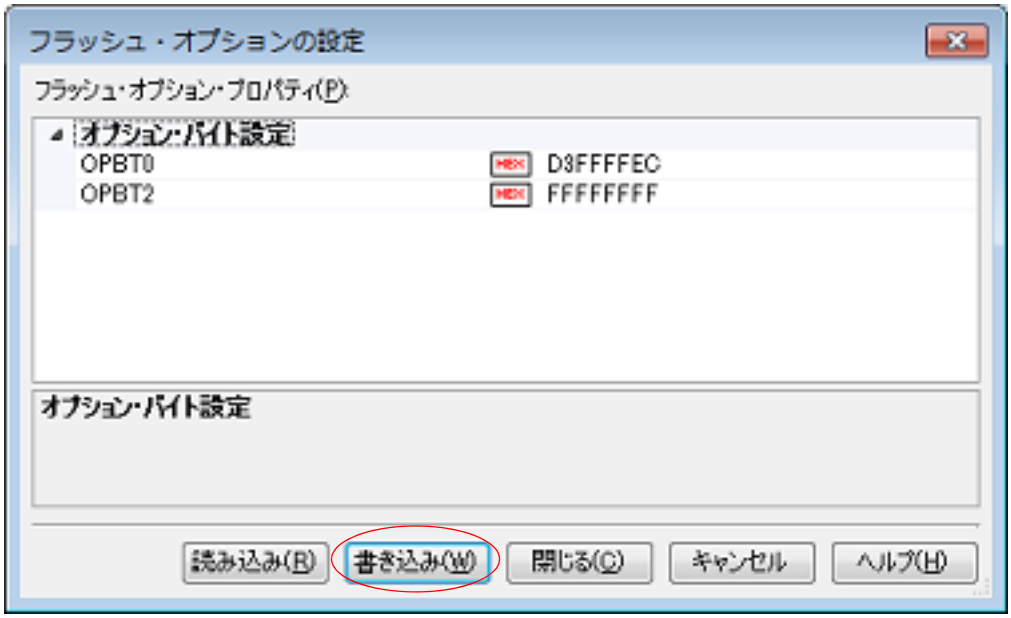


図 2.27　　フラッシュ・オプションの設定（フラッシュ・オプションの設定 ダイアログ）



2.3.3.5 ［フック処理設定］タブ

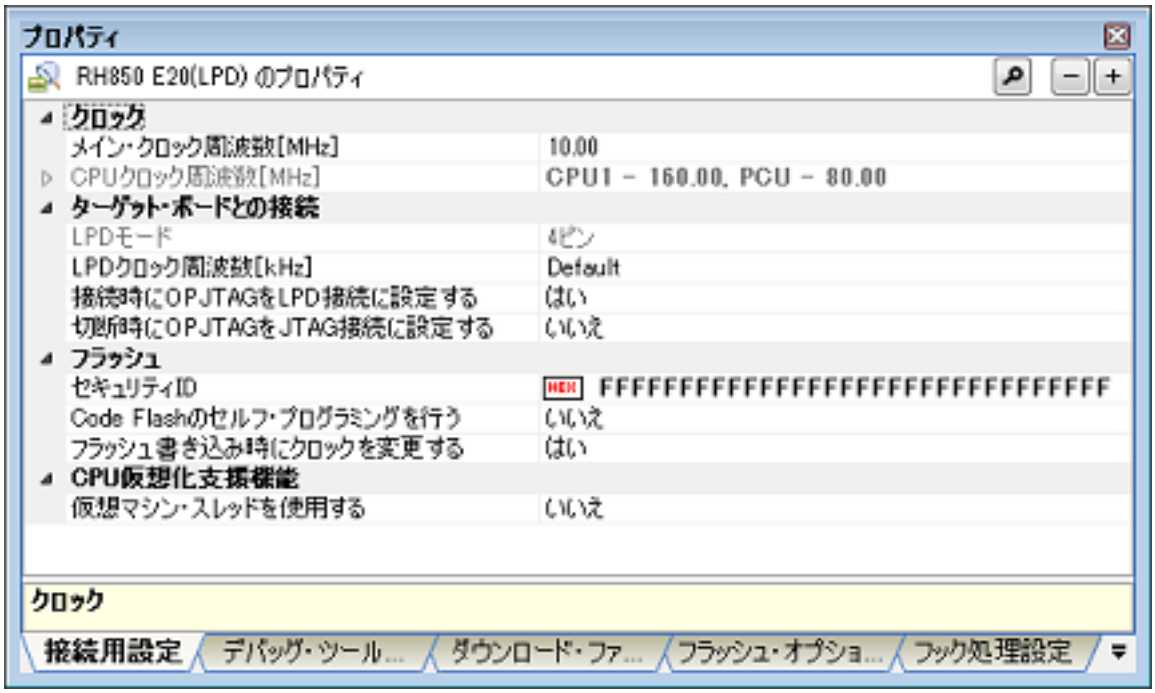
デバッグ・ツールにフック処理の設定を行います。
フック処理，および各カテゴリ内の設定についての詳細は，「2.17 フック処理を設定する」を参照してください。

2.3.4 【E20】の場合

E20 を使用する場合の動作環境の設定を次のプロパティ パネルで行います。

注意　　ターゲット・ボードとの通信方式として，LPD 通信方式のみをサポートしています。

図 2.28　　動作環境設定【E20】（プロパティ パネル）



プロパティ パネル上の該当するタブを選択し，次の設定を順次行ってください。

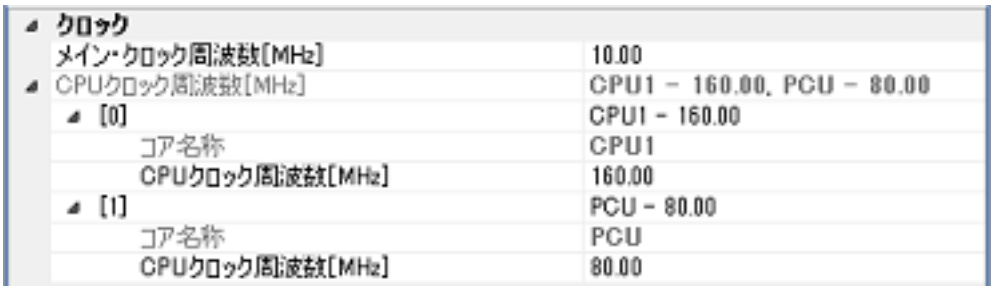
2.3.4.1 ［接続用設定］タブ

次に示すカテゴリごとに，デバッグ・ツールとの接続に関する設定を行います。

(1)　［クロック］

クロックに関する設定を行います。

図 2.29　［クロック］カテゴリ【E20】



- ［メイン・クロック周波数 [MHz]］
メイン・クロック周波数（逓倍前）を指定します。
ドロップダウン・リストによる選択，または直接入力により，0.001 ～ 999.999（単位：MHz）の範囲の数値で指定してください（デフォルト：[10.00]）。

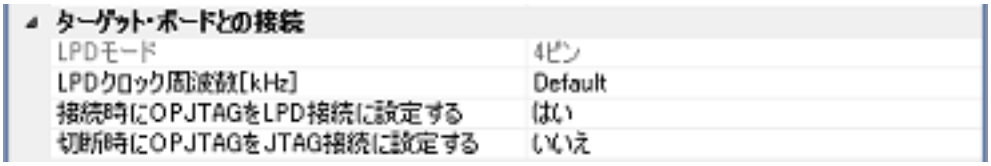
- ［CPU クロック周波数 [MHz]］
CPU クロック周波数（逓倍後）をコアごとに指定します。
サブプロパティとして，選択しているマイクロコントローラが持つコア名称を表示します。
各コアの CPU クロック周波数を，ドロップダウン・リストによる選択，または直接入力により，0.001 ～ 999.999（単位：MHz）の範囲の数値で指定してください。
なお，表示されるコア名称の数，およびデフォルトの CPU クロック周波数は，選択しているマイクロコントローラの種類により異なります。

備考　CPU クロック周波数は，トレースのタイム・スタンプ情報を実時間に換算する際に使用されます。

(2)　［ターゲット・ボードとの接続］
E20 とターゲット・ボードとの接続に関する設定を行います。

注意　E20 が CS+ に接続している場合，このカテゴリ内のプロパティを変更することはできません。

図 2.30　［ターゲット・ボードとの接続］カテゴリ【E20】



- ［LPD モード］
使用する LPD 通信方式のモードを選択します。
ドロップダウン・リスト内に表示されるピン数は，選択しているマイクロコントローラの種類により異なります。
ただし，選択できるモードが 1 つしか存在しない場合は，このプロパティを変更することはできません。

- ［ボーレート [kbps]］
このプロパティは，［LPD モード］プロパティにおいて，［1 ピン］を選択した場合のみ表示されます。
LPD 通信方式の通信速度を選択します（デフォルト：[500]）。

- ［LPD クロック周波数 [kHz]］
このプロパティは，［LPD モード］プロパティにおいて，［4 ピン］を選択した場合のみ表示されます。
LPD 通信方式のクロック周波数を選択します（デフォルト：[Default]）。
なお，［Default］を選択した場合は，マイクロコントローラ固有のデフォルト値で接続処理を行います。

- ［接続時に OPJTAG を LPD 接続に設定する］
デバッグ・ツールとの接続時にシリアル・プログラミング・モードで起動し，オプション・バイトの設定を LPD 接続に変更するか否かを選択します。
［はい］を選択した場合，接続時にシリアル・プログラミング・モードで起動し，OPJTAG をチェックします。この際に LPD 設定でない場合は LPD 設定に変更し，その後デバッグ・モードに移行します（デフォルト）。
［いいえ］を選択した場合，接続時にデバッグ・モードで起動し，OPJTAG をチェックします。この際に LPD 設定でない場合はメッセージ・ダイアログを表示します。

- ［切断時に OPJTAG を JTAG 接続に設定する］
このプロパティは，［接続時に OPJTAG を LPD 接続に設定する］プロパティにおいて，［はい］を選択した場合のみ変更することができます。

デバッグ・ツールとの接続を切断する際に，オプション・バイトの設定を JTAG 接続に変更するか否かを選択します。
切断時にオプション・バイトの設定を JTAG 接続に変更する場合は［はい］を選択してください。
なお，［いいえ］（デフォルト）を選択している場合，切断時にオプション・バイトを変更しないため，ピン・モードは LPD モード設定となります。

備考　E20 との接続時に CS+ は，オプション・バイトが LPD モード設定になっていない場合，オプション・バイトの値を書き換えます。このため，マイクロコントローラのオプション・バイトが E1 との接続前と接続後で異なる可能性があります。

(3)　［フラッシュ］
フラッシュ書き換えに関する設定を行います。

注意　E20 が CS+ に接続している場合，このカテゴリ内のプロパティを変更することはできません。

図 2.31　［フラッシュ］カテゴリ【E20】

- ［セキュリティ ID］
このプロパティは，選択しているマイクロコントローラが，フラッシュ・メモリの ROM セキュリティ機能をサポートしている場合のみ表示されます。
内蔵 ROM，または内蔵フラッシュ・メモリ上のコードを読み出す際の ID コードを指定します。
直接入力により，32 桁の 16 進数（16 バイト）で指定します（デフォルト：［FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF］）。

- ［Code Flash のセルフ・プログラミングを行う］
フラッシュ・セルフ・プログラミング機能のフラッシュ・セルフ・ライブラリを使用して，Code Flash の書き換えを行うか否かを選択します。
Code Flash の書き換えを行う場合は，［はい］を選択してください（デフォルト：［いいえ］）。
なお，［はい］を選択した場合，Code Flash のキャッシュは行われません。

- ［フラッシュ書き込み時にクロックを変更する］
デバッガ操作により，フラッシュ・メモリに書き込みを行う際に，一時的にクロック・スピードを変更するか否かを選択します。
フラッシュ・メモリへの書き込み性能を向上させるために，フラッシュ書き換え時にクロック・アップを行う場合は，［はい］を選択してください（デフォルト）。
［いいえ］を選択した場合，ユーザが設定したクロック・スピードでフラッシュ書き換えを行います。

備考　［はい］を選択した場合，CPU クロックだけではなく周辺クロックも変化するため，ブレーク中も動作している周辺システムに影響がある可能性があります。
［いいえ］を選択した場合，設定したクロック・スピードが低いと，デバッガ操作によるフラッシュ書き換え時間が長くなります。

(4)　［CPU 仮想化支援機能］
このカテゴリ内のプロパティは，常に無効です。

2.3.4.2　［デバッグ・ツール設定］タブ

次に示すカテゴリごとに，デバッグ・ツールの基本設定を行います。

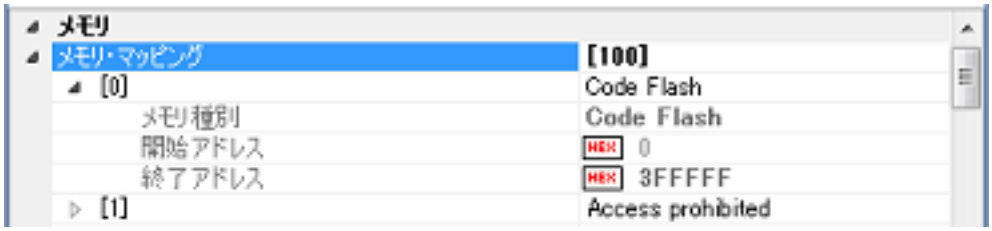
(1)　［メモリ］
(2)　［実行中のメモリ・アクセス］
(3)　［実行中のイベント設定］
(4)　［ブレーク］
(5)　［トレース］
(6)　［入力信号のマスク］

(1)　［メモリ］
メモリに関する設定を行います。

図 2.32 ［メモリ］カテゴリ【E20】

- ［メモリ・マッピング］
展開表示することにより，現在のメモリ・マッピングの状況が，メモリ領域の種別ごとに詳細表示されます。

図 2.33 メモリ・マッピングの詳細表示



注意 メモリ・マッピングの追加／削除はできません。

備考 選択しているマイクロコントローラがマルチコア対応版の場合，コア（PE）の選択を切り替えることにより，PE ごとのメモリ・マッピングの状況を表示します（「2.7 コア（PE）の選択」参照）。

- ［メモリ書き込み時にベリファイを行う］
メモリ書き込み時に，ベリファイを行うか否かを選択します。
［はい］を選択した場合，ダウンロードの際，またはメモリ パネル／ウォッチ パネルで値を変更した際にベリファイを行います（デフォルト）。

(2) ［実行中のメモリ・アクセス］
プログラム実行中におけるメモリ・アクセス（リアルタイム表示更新機能）に関する設定を行います。
リアルタイム表示更新機能についての詳細は，「2.10.1.4 プログラム実行中にメモリの内容を表示／変更する」を参照してください。

図 2.34 ［実行中のメモリ・アクセス］カテゴリ【E20】

- ［実行中にアクセスする］
プログラム実行中に内蔵 RAM 領域にアクセスするか否かを選択します。
アクセスを許可する場合は［はい］を選択してください（デフォルト：［いいえ］）。

- ［実行中に表示更新を行う］
プログラム実行中に，メモリ パネル／ウォッチ パネルの表示内容を更新するか否かを選択します。
表示内容の更新を行う場合は［はい］を選択してください（デフォルト）。

- ［表示更新間隔 [ms]］
このプロパティは，［実行中に表示更新を行う］プロパティにおいて［はい］を選択した場合のみ有効となります。
プログラム実行中に，メモリ パネル／ウォッチ パネルの表示内容を更新する間隔を指定します。
直接入力により，100 ～ 65500 の範囲の整数（100 ms 未満の端数切り上げ）を指定してください（デフォルト：［500］）。

(3) ［実行中のイベント設定］
このカテゴリでは，実行中のイベント設定に関する設定を行います。

図 2.35 ［実行中のイベント設定］カテゴリ

- ［実行を一瞬停止してイベントを設定する］
プログラム実行中，またはトレーサ／タイマ動作中には設定することができないイベントを，プログラムの実行，またはトレーサ／タイマの動作を強制的に一瞬停止させることで設定を行うか否かを選択します。

このプロパティの対象となるイベント種別については，「2.16.6.2 実行中に設定／削除可能なイベント種別」を参照してください。
プログラム実行中，またはトレーサ／タイマ動作中に，イベントの設定を行う場合は［はい］を選択してください（デフォルト：［いいえ］）。

(4) ［ブレーク］
ブレーク機能に関する設定を行います。
ブレーク機能，およびこのカテゴリ内の設定についての詳細は，「2.9 プログラムの停止（ブレーク）」を参照してください。

(5) ［トレース］
トレース機能に関する設定を行います。
トレース機能，およびこのカテゴリ内の設定についての詳細は，「2.12 実行履歴の収集」を参照してください。

(6) ［入力信号のマスク］
入力信号のマスクに関する設定を行います。

図 2.36 ［入力信号のマスク］カテゴリ【E20】

- ［WAIT 信号をマスクする］
WAIT 信号をマスクするか否かを選択します。
WAIT 信号を E20 に入力しない場合は，［はい］を選択してください（デフォルト：［いいえ］）。

- ［RESET 信号をマスクする］
RESET 信号をマスクするか否かを選択します。
RESET 信号を E20 に入力しない場合は，［はい］を選択してください（デフォルト：［いいえ］）。

- ［マスクする RESET 信号の選択］
このプロパティは，- ［RESET 信号をマスクする］プロパティにおいて［はい］を選択した場合のみ表示されます。
マスクされる RESET 信号を表示します。
このプロパティを変更することはできません。

2.3.4.3 ［ダウンロード・ファイル設定］タブ

デバッグ・ツールにダウンロードを実行する際の設定を行います。
各カテゴリ内の設定についての詳細は，「2.5.1 ダウンロードを実行する」を参照してください。

2.3.4.4 ［フラッシュ・オプション設定］タブ

マイクロコントローラに搭載されているフラッシュ・メモリのためのオプション設定を行います。
ただし，このタブは，選択しているマイクロコントローラがフラッシュ・オプションに対応している場合のみ表示されます。
なお，各オプションの設定はこのタブ上では行わず，［フラッシュ・オプション］カテゴリ内［フラッシュ・オプション］プロパティを選択すると欄内右端に表示される［...］ボタンをクリックすることによりオープンする，フラッシュ・オプションの設定 ダイアログ【Full-spec emulator】【E1】【E20】で行います（［...］ボタンは，デバッグ・ツールと接続している場合のみ表示されます）。
このダイアログにおいて，各項目の設定を完了したのち，［書き込み］ボタンをクリックしてください。
設定内容についての詳細は，フラッシュ・オプションの設定 ダイアログ【Full-spec emulator】【E1】【E20】を参照してください。

図 2.37 フラッシュ・オプションの設定 ダイアログのオープン

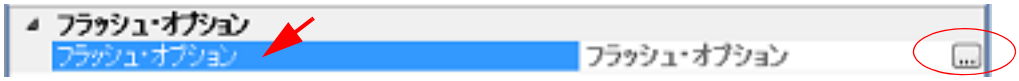
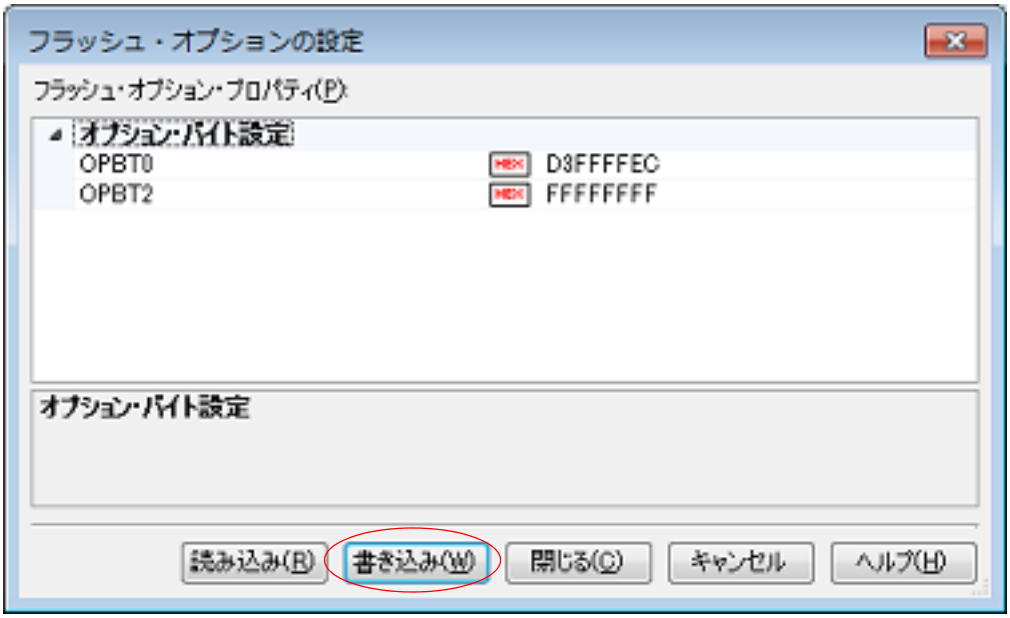


図 2.38 フラッシュ・オプションの設定（フラッシュ・オプションの設定 ダイアログ）



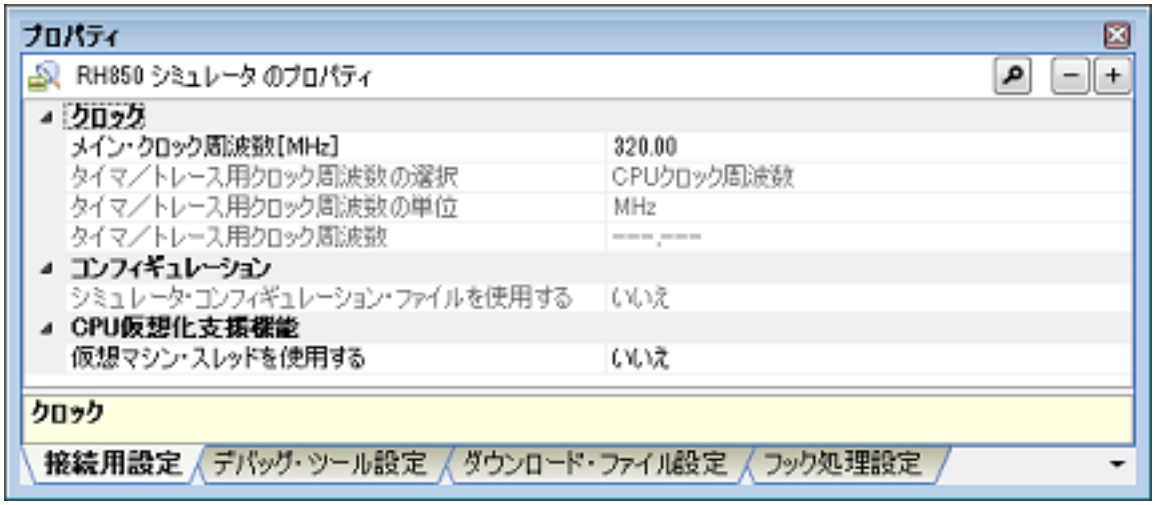
2.3.4.5 ［フック処理設定］タブ

デバッグ・ツールにフック処理の設定を行います。
フック処理，および各カテゴリ内の設定についての詳細は，「2.17 フック処理を設定する」を参照してください。

2.3.5 【シミュレータ】の場合

シミュレータを使用する場合の動作環境の設定を次のプロパティ パネルで行います。

図 2.39 動作環境設定【シミュレータ】（プロパティ パネル）



プロパティ パネル上の該当するタブを選択し，次の設定を順次行ってください。

2.3.5.1 ［接続用設定］タブ

次に示すカテゴリごとに，デバッグ・ツールとの接続に関する設定を行います。

(1) ［クロック］
(2) ［コンフィギュレーション］
(3) ［CPU 仮想化支援機能］

(1) ［クロック］
クロックに関する設定を行います。

図 2.40 ［クロック］カテゴリ【シミュレータ】

- ［メイン・クロック周波数 [MHz]］
メイン・クロック周波数を指定します。
ドロップダウン・リストによる選択，または直接入力により，0.001 ～ 999.999（単位：MHz）の範囲の周波数を指定します（デフォルト：[320.00]）。

注意 命令シミュレータ版では，CPU クロック周波数は，常にここで指定するメイン・クロック周波数の値となります。

- ［タイマ／トレース用クロック周波数の選択］
タイマ／トレース機能に使用するクロック周波数を表示します。
このプロパティ値を変更することはできません。

- ［タイマ／トレース用クロック周波数の単位］
タイマ／トレース機能に使用するクロック周波数の単位を表示します。
このプロパティ値を変更することはできません。

- ［タイマ／トレース用クロック周波数］
タイマ／トレース機能に使用するクロック周波数の値を表示します。
ただし，デバッグ・ツールと切断時は “---_---” を表示します。
このプロパティ値を変更することはできません。

(2) ［コンフィギュレーション］
このカテゴリ内のプロパティは，常に無効です。

(3) ［CPU 仮想化支援機能］
このカテゴリ内のプロパティは，常に無効です。

2.3.5.2 ［デバッグ・ツール設定］タブ

次に示すカテゴリごとに，デバッグ・ツールの基本設定を行います。

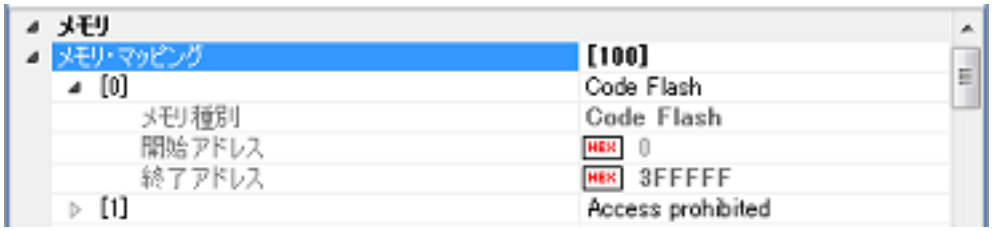
(1) ［メモリ］
(2) ［実行中のメモリ・アクセス］
(3) ［トレース］
(4) ［タイマ］
(5) ［カバレッジ］
(6) ［シミュレータ GUI］

(1) ［メモリ］
メモリに関する設定を行います。

図 2.41 ［メモリ］カテゴリ【シミュレータ】

- ［メモリ・マッピング］
展開表示することにより，現在のメモリ・マッピングの状況が，メモリ領域の種別ごとに詳細表示されます。

図 2.42　　メモリ・マッピングの詳細表示



注意　　メモリ・マッピングの追加／削除はできません。

備考　　選択しているマイクロコントローラがマルチコア対応版の場合，コア（PE）の選択を切り替えることにより，PE ごとのメモリ・マッピングの状況を表示します（「2.7 コア（PE）の選択」参照）。

(2)　［実行中のメモリ・アクセス］
プログラム実行中におけるメモリ・アクセス（リアルタイム表示更新機能）に関する設定を行います。
リアルタイム表示更新機能についての詳細は，「2.10.1.4 プログラム実行中にメモリの内容を表示／変更する」を参照してください。

図 2.43　　［実行中のメモリ・アクセス］カテゴリ【シミュレータ】

- ［実行中に表示更新を行う］
  プログラム実行中に，メモリ パネル／ウォッチ パネルの表示内容を更新するか否かを選択します。
  表示内容の更新を行う場合は［はい］を選択してください（デフォルト）。

- ［表示更新間隔 [ms]］
  このプロパティは，［実行中に表示更新を行う］プロパティにおいて［はい］を選択した場合のみ有効となります。
  プログラム実行中に，メモリ パネル／ウォッチ パネルの表示内容を更新する間隔を指定します。
  直接入力により，100 ～ 65500 の整数（100 ms 未満の端数切り上げ）を指定してください（デフォルト：[500]）。

(3)　［トレース］
トレース機能に関する設定を行います。
トレース機能，およびこのカテゴリ内の設定についての詳細は，「2.12 実行履歴の収集」を参照してください。

(4)　［タイマ］
タイマ機能に関する設定を行います。
タイマ機能についての詳細は，「2.13 実行時間の計測」を参照してください。

図 2.44　　［タイマ］カテゴリ

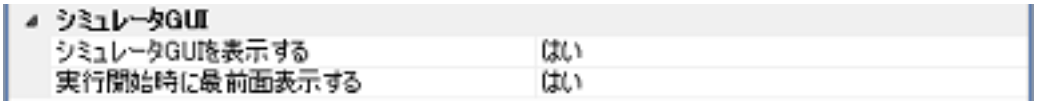
- ［タイマ機能を使用する］
  タイマ機能を使用するか否かを選択します。
  タイマ機能を使用する場合は［はい］を選択してください（デフォルト：[いいえ]）。

(5)　［カバレッジ］
カバレッジ機能に関する設定を行います。
カバレッジ機能，およびこのカテゴリ内の設定についての詳細は，「2.14 カバレッジの測定【シミュレータ】」を参照してください。

(6)　［シミュレータ GUI］
シミュレータ GUI 機能に関する設定を行います。

注意　　選択しているマイクロコントローラのシミュレータが周辺機能シミュレーションをサポートしていない場合（命令シミュレータ版の場合），このカテゴリ内のすべてのプロパティは無効となります。

図 2.45　［シミュレータ GUI］カテゴリ



- ［シミュレータ GUI を表示する］
  シミュレータ GUI ウインドウを表示するか否かを選択します。
  シミュレータ GUI の機能を使用する場合は［はい］を選択してください（デフォルト）。
  シミュレータ GUI の機能を使用しない場合は［いいえ］を選択することにより，シミュレータ GUI ウインドウがクローズします。
- ［実行開始時に最前面表示する］
  このプロパティは，［シミュレータ GUI を表示する］プロパティにおいて［はい］を選択した場合のみ表示されます。
  プログラムの実行開始時に，シミュレータ GUI ウインドウを最前面に表示するか否かを選択します。
  最前面に表示する場合は［はい］を選択してください（デフォルト）。

2.3.5.3 ［ダウンロード・ファイル設定］タブ

デバッグ・ツールにダウンロードを実行する際の設定を行います。
各カテゴリ内の設定についての詳細は，「2.5.1 ダウンロードを実行する」を参照してください。

2.3.5.4 ［フック処理設定］タブ

デバッグ・ツールにフック処理の設定を行います。
フック処理，および各カテゴリ内の設定についての詳細は，「2.17 フック処理を設定する」を参照してください。

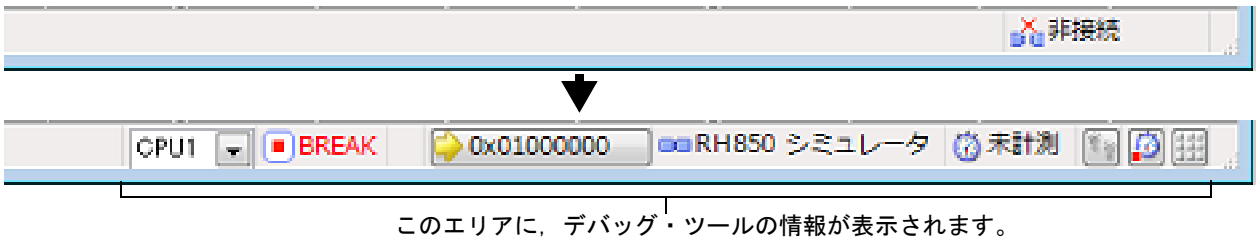
2.4 デバッグ・ツールとの接続／切断

この節では，CS+ とデバッグ・ツールの接続方法，および切断方法について説明します。

2.4.1 デバッグ・ツールを接続する

［デバッグ］メニュー→［デバッグ・ツールへ接続］を選択することにより，CS+ はアクティブ・プロジェクトで選択しているデバッグ・ツールと通信を開始します。
デバッグ・ツールとの接続に成功すると，メイン・ウインドウのステータスバーが，次のように変化します。
なお，ステータスバーに表示される各項目についての詳細は，メイン・ウインドウを参照してください。

図 2.46　デバッグ・ツールとの接続に成功したステータスバー



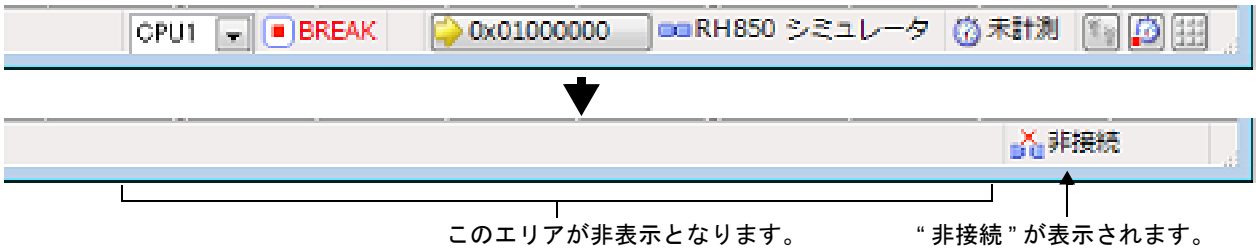
注意　CS+ のサポート範囲外のコンパイラを使用している場合、［デバッグ・ツールへ接続］は無効となります。

備考　デバッグ・ツールバーの  ボタンをクリックすることで，デバッグ・ツールと接続したのち，指定ファイルのダウンロードを実行します（「2.5.1 ダウンロードを実行する」参照）。
また，同ツールバーの  ボタンをクリックすることで，プロジェクトのビルドを行い，デバッグ・ツールと接続したのち，指定ファイルのダウンロードを実行します。

2.4.2 デバッグ・ツールを切断する

デバッグ・ツールバーの  ボタンをクリックすることにより，CS+ は接続しているデバッグ・ツールとの通信を切断します。
デバッグ・ツールとの通信が切断すると，メイン・ウインドウのステータスバーが，次のように変化します。

図 2.47　デバッグ・ツールを切断したステータスバー



注意　プログラム実行中にデバッグ・ツールを切断することはできません。

備考　デバッグ・ツールを切断すると，デバッグ・ツールと接続時のみ表示可能なパネル／ダイアログはすべてクローズします。

2.5 ダウンロード／アップロード

この節では，デバッグ対象となるプログラム（ロード・モジュール・ファイルなど）を CS+ へダウンロードする方法と，デバッグ中のメモリ内容を CS+ からファイルへアップロードする方法を説明します。

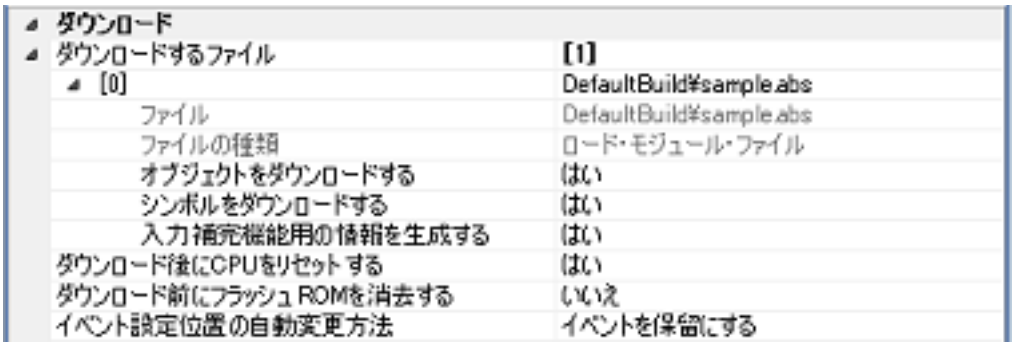
2.5.1 ダウンロードを実行する

デバッグ対象となるロード・モジュール・ファイルのダウンロードを実行します。
次に示す手順に従って，プロパティ パネルの［ダウンロード・ファイル設定］タブにおけるダウンロードのための設定を行ったのち，ダウンロードを実行してください。

注意　デフォルトの設定では，ダウンロード後に自動的に CPU をリセットし，指定シンボルまで実行します。この動作が不要な場合は，［ダウンロード後に CPU をリセットする］プロパティ，および［CPU リセット後に指定シンボル位置まで実行する］プロパティにおいて［いいえ］を選択してください。

(1)　［ダウンロード］カテゴリの設定

図 2.48　［ダウンロード］カテゴリ



(a)　［ダウンロードするファイル］
ダウンロードの対象となるファイル名，およびダウンロード条件を表示します（プロパティ値の “[ ]” 内の数値は，現在ダウンロードの対象に指定されているファイル数を示します）。
ダウンロードの対象となるファイルは，メイン・プロジェクト／サブプロジェクトでビルド対象に指定しているファイルが自動的に決定されます[注]。
ただし，ダウンロードの対象となるファイル，およびダウンロード条件は，手動で変更することができます。この場合は，「2.5.2 応用的なダウンロード方法」を参照してください。

注　外部ビルド・ツール（CS+ が提供するビルド・ツール以外のコンパイラ／アセンブラなど）により作成されたロード・モジュール・ファイルをダウンロードする場合，デバッグ専用プロジェクトを作成する必要があります。
デバッグ専用プロジェクトをデバッグの対象とする場合では，ユーザが，プロジェクト・ツリー上のダウンロード・ファイル・ノードにダウンロードするファイルを追加することで，ダウンロードの対象となるファイルがこのプロパティに反映されます。
なお，“ 外部ビルド・ツールの使用 ”，および “ デバッグ専用プロジェクト ” についての詳細は，「CS+ 統合開発環境 ユーザーズマニュアル プロジェクト操作編」を参照してください。

(b)　［ダウンロード後に CPU をリセットする］
ダウンロード完了後に CPU をリセットするか否かを指定します。
CPU をリセットする場合は［はい］を選択してください（デフォルト）。

(c)　［ダウンロード前にフラッシュ ROM を消去する］【Full-spec emulator】【E1】【E20】
ダウンロード実行前にフラッシュ ROM を消去するか否かを指定します。
フラッシュ ROM を消去する場合は［はい］を選択してください（デフォルト：［いいえ］）。
なお、[はい] を選択した場合、ダウンロード・データが存在する領域のみ消去の対象となります。

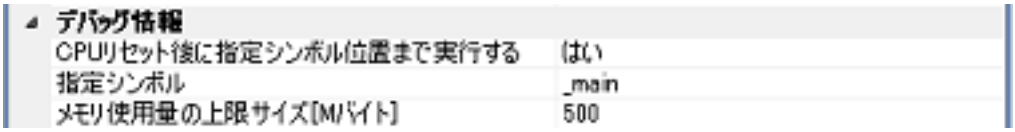
(d)　［イベント設定位置の自動変更方法］
デバッグ作業を進めることにより，変更を加えたプログラムを再ダウンロードした場合，現在設定されているイベントの設定位置（アドレス）が命令の途中になる場合があります。この場合の対象イベントの扱いをこのプロパティで指定します。
次のドロップダウン・リストによりどちらかを選択してください。

命令の先頭に移動する	命令の先頭アドレスに対象イベントを再設定します。
イベントを保留にする	対象イベントを保留状態にします（デフォルト）。

ただし，このプロパティでの指定は，デバッグ情報のないイベント設定位置に対してのみ適用されます。デバッグ情報があるイベント設定位置の場合は，常にソース・テキスト行の先頭に移動します。

(2) ［デバッグ情報］カテゴリの設定

図 2.49 ［デバッグ情報］カテゴリ



(a) ［CPU リセット後に指定シンボル位置まで実行する］
CPU リセット後，またはダウンロード完了後（［ダウンロード後に CPU をリセットする］プロパティで［はい］を選択している場合のみ）に，プログラムを指定シンボル位置まで実行するか否かを指定します。
プログラムを指定シンボル位置まで実行する場合は［はい］を選択してください（デフォルト）。

備考 ［ダウンロード後に CPU をリセットする］プロパティで［はい］を選択している場合では，このプロパティで［はい］を選択すると，ダウンロード完了後，［指定シンボル］プロパティで指定した位置のソース・テキストを表示した状態でエディタ パネルが自動的にオープンします。
また，［いいえ］を選択した場合では，リセット番地を表示した状態で同パネルがオープンします（リセット番地にソース・テキストが割り付けられていない場合は，逆アセンブル パネルで該当箇所を表示します）。

(b) ［指定シンボル］
このプロパティは，［CPU リセット後に指定シンボル位置まで実行する］プロパティにおいて［はい］を選択した場合のみ表示されます。
CPU リセット後にプログラムを実行して停止する位置を指定します。
直接入力により，0 ～ “ *アドレス空間の終了アドレス* ” の範囲のアドレス式で指定してください（デフォルト：[_main]）。
ただし，指定したアドレス式がアドレスに変換できない場合，プログラムは実行されません。

備考 通常，次を指定します。
アセンブリ・ソースの場合: メイン関数に相当する先頭ラベル
C ソースの場合： メイン関数名の先頭に付与したシンボル

(c) ［メモリ使用量の上限サイズ [M バイト]］
デバッグ情報を読み込む際に使用するメモリ・サイズの上限値を指定します。
使用したメモリ・サイズがここで指定した上限値を越えた場合，上限値の 1/2 以下のメモリ・サイズになるまで読み込んだデバッグ情報を破棄することでメモリを開放します（メモリ不足が発生する場合，上限値を小さくすることで改善される可能性があります）。
直接入力により，100 ～ 1000（単位：M バイト）の範囲の 10 進数値で指定してください（デフォルト：[500]）。

注意 上限値を小さくした場合，デバッグ情報の破棄と再読み込みが頻繁に行われるため，デバッグ・ツールの応答性が低下する場合があります。

(3) ダウンロードの実行
デバッグ・ツールバーの  ボタンをクリックします。
なお，デバッグ・ツールと切断時にこの操作が行われた場合は，自動的にデバッグ・ツールと接続したのち，ダウンロードを実行します。

備考 デバッグ作業を進めることにより，変更を加えたプログラムを再ダウンロードする場合は，メイン・ウインドウ上の［デバッグ］メニュー→［ビルド & デバッグ・ツールへダウンロード］を選択することにより，ビルド→ダウンロードを容易に行うことができます。

(4) ダウンロードの中断
ダウンロードの実行を中断する場合は，ダウンロードの進捗状況を表示する処理中表示 ダイアログの [キャンセル] ボタンをクリック，または [Esc] キーを押下します。

ロード・モジュール・ファイルのダウンロードが成功すると，エディタ パネルが自動的にオープンし，ダウンロードしたファイルのソース・テキストが表示されます。

備考 ダウンロードの実行前／実行後に，I/O レジスタ /CPU レジスタ値を指定した値に自動的に書き換える処理を設定することができます（「2.17 フック処理を設定する」参照）。

2.5.2 応用的なダウンロード方法

ダウンロードの対象となるファイル，およびダウンロード条件は変更することができます。
CS+ では，次のファイルをダウンロードすることができます。

表 2.1　　ダウンロード可能なファイル

ダウンロード可能なファイル	拡張子	ファイル形式
ロード・モジュール・ファイル	.abs	ロード・モジュール・フォーマット
	.out	
インテル拡張ヘキサ・ファイル	.hex	インテル拡張ヘキサ・フォーマット
モトローラ・S タイプ・ファイル	.mot	モトローラ・S タイプ・フォーマット - S0, S1, S9-16 ビット - S0, S2, S8-24 ビット - S0, S3, S7-32 ビット
バイナリ・ファイル	.bin	バイナリ・フォーマット

注　　GHS コンパイラ（米国 Green Hills Software, Inc. 製）使用上の注意点

- 対応バージョン
  - MULTI（Ver.6.1.4 以上），GHS コンパイラ（Ver.2012.5.5 以上）
- 対応オプション
  デバッグ・オプション　：　-G，-dual_debug，-cpu=rh850/-cpu=v850e3
  【注意】GHS コンパイラ（Ver.2013.1.5）では，rh850 オプションが次の名称に変更されました。
  rh850g3m，rh850g3k，rh850g3h
  詳細については，GHS 製品のリリースノートを参照してください。
  最適化オプション　：　-Odebug，-O，-Ospeed，-Onone
  その他　：　-prepare_dispose，-callt
- 非対応オプション
  リンカ最適化オプション　：　-shorten_loads，-code_factor，-delete
  （リンカによって実コードが変更され，デバッグ情報にその変更が反映されないため）
- デバッグ上の注意点
  - 作成したロード・モジュール・ファイルは，デバッグ専用プロジェクトに追加してください。
  - 次のプログラムのデバッグはできません。
    - C++ 言語
    - C99 言語固有の型／修飾子を使用したプログラム
    - GNU C 拡張仕様を使用したプログラム
  - **ステップ，および実行関連機能について**
    次の関数内からのリターン・アウト実行は失敗することがあります。この場合は，コール・スタック パネルの呼び出し履歴も正しく表示されません。
    - callt で呼び出された関数
    - 割り込み関数
  - 式を使用した変数の参照機能について
    - long long 型，または double 型の変数がレジスタに割り付いて場合，ウォッチ パネルの［アドレス］エリアには下位 4 バイト側のレジスタ名のみ表示されます。また，この時，CS+ は，上位 4 バイトの値を表示したレジスタの次のレジスタから値を取得します。
      例）［アドレス］エリアに “R4” が表示されている場合，CS+ は，上位 4 バイトの値を R5 レジスタから取得します。

- 構造体型の変数がレジスタに割り付いた場合，ウォッチ パネルでは構造体メンバの値を正しく表示できません。値は［アドレス］エリアに表示したレジスタを CPU レジスタ パネルで参照してください。
- スコープ指定付きの式を使用しても関数内に定義したスタティック変数は参照できません（PC カウンタ値が変数を定義した関数内にある場合のみ参照可能です）。

```
func(){
    static  sta = 100;
}
```

上記例の場合，
func() 関数をデバッグ中の場合，“sta” および “func()#sta” の両方で参照できます。
func() 関数以外をデバッグ中の場合，“sta”，または “func()#sta” のどちらでも参照できません。

- 関数の先頭位置ではスタック・フレームを形成していないため，スタック渡しの引数のアドレスが正しくありません。引数の値は，関数内へステップしてから参照してください。

- その他の機能について
  - シンボル名の入力補完機能を無効化することはできません（ダウンロード・ファイル ダイアログの［入力補完機能用の情報を生成する］項目の指定は無視されます）。

ダウンロード・ファイルの変更，およびその際のダウンロード条件の設定は，次のダウンロード・ファイル ダイアログにより行います。
ダウンロード・ファイル ダイアログは，プロパティ パネルの［ダウンロード・ファイル設定］タブ上の［ダウンロード］カテゴリ内［ダウンロードするファイル］プロパティを選択することで欄内右端に表示される［...］ボタをクリックするとオープンします。

図 2.50 ダウンロード・ファイル ダイアログのオープン

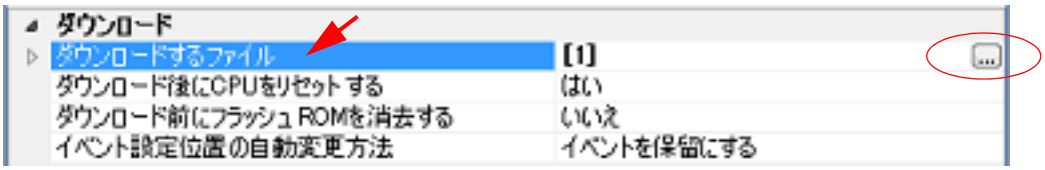
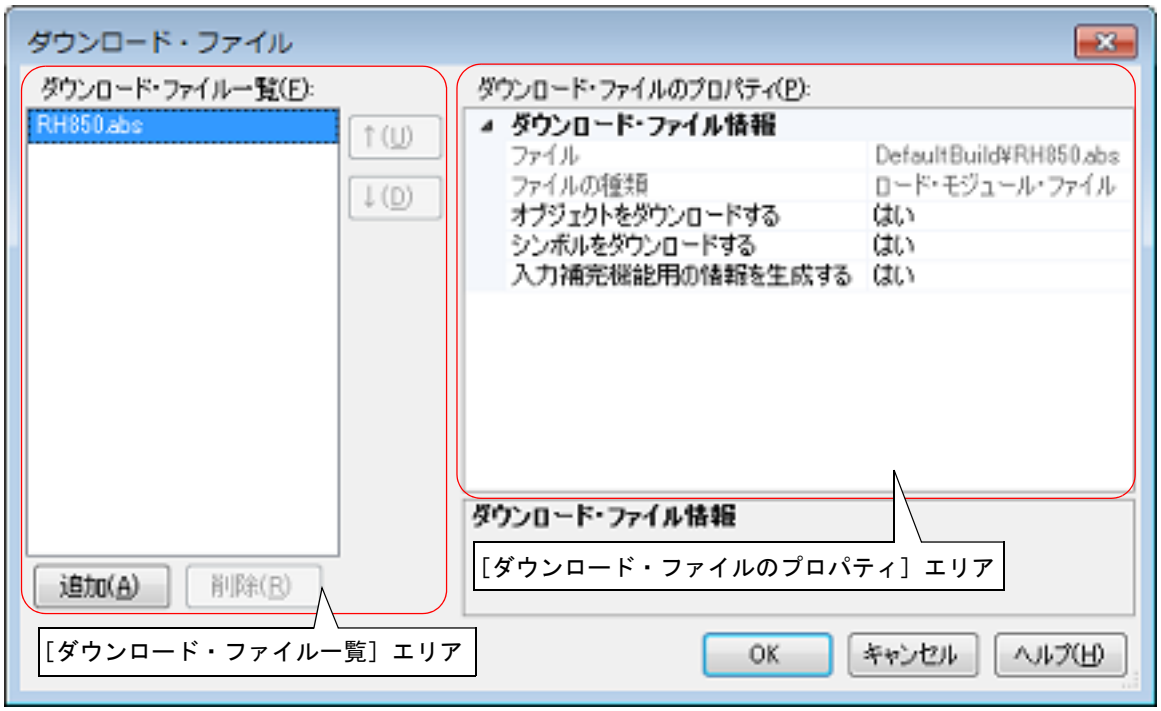


図 2.51 応用的なダウンロード方法（ダウンロード・ファイル ダイアログ）



ここでは，上記ダウンロード・ファイル ダイアログにおける，次の場合の設定方法を説明します。

2.5.2.1 ロード・モジュール・ファイルのダウンロード条件を変更する

ロード・モジュール・ファイルのダウンロード条件（オブジェクト情報／シンボル情報の読み込みなど）を変更する場合は，ダウンロード・ファイル ダイアログにおいて，次の手順の操作を行ってください。

(1) ロード・モジュール・ファイルの選択
［ダウンロード・ファイル一覧］エリアにおいて，ダウンロードするロード・モジュール・ファイルを選択します。

(2) ダウンロード条件の変更
［ダウンロード・ファイルのプロパティ］エリアでは，選択しているロード・モジュール・ファイルのダウンロード条件が表示されます。
表示される各項目において，設定の変更を行います。

オブジェクトをダウンロードする	指定したファイルからオブジェクト情報をダウンロードするか否かを選択します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	オブジェクト情報をダウンロードします。
		いいえ	オブジェクト情報をダウンロードしません。
シンボルをダウンロードする	指定したファイルからシンボル情報をダウンロードするか否かを選択します注1。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	シンボル情報をダウンロードします。
		いいえ	シンボル情報をダウンロードしません。
入力補完機能用の情報を生成する	ダウンロード時に，シンボル名の入力補完機能のための情報を生成するか否かを選択します注2。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	シンボル名の入力補完機能用の情報を生成します（入力補完機能を使用します）。
		いいえ	シンボル名の入力補完機能用の情報を生成しません（入力補完機能を使用しません）。

注 1. シンボル情報をダウンロードしない場合，ソース・レベル・デバッグを行うことはできません。

注 2. ［はい］を選択した場合，ダウンロード時間，およびホスト・マシンのメモリ消費量が増加します。シンボル名の入力補完機能を使用しない場合は，［いいえ］を選択することを推奨します。

(3) ［OK］ボタンのクリック
このダイアログでの設定を有効とし，ダウンロード条件が変更されます。

2.5.2.2 ダウンロード・ファイル（*.hex/*.mot/*.bin）を追加する

ロード・モジュール・フォーマット以外のファイル（インテル拡張ヘキサ・ファイル（*.hex）／モトローラ・S タイプ・ファイル（*.mot）／バイナリ・ファイル（*.bin））をダウンロード対象に追加する場合は，ダウンロード・ファイル ダイアログにおいて，次の手順の操作を行ってください。

(1) ［追加］ボタンのクリック
［追加］ボタンをクリックすると，［ダウンロード・ファイル一覧］エリアの最終行に空欄の項目（“-”）が表示されます。

(2) 追加するダウンロード・ファイルのプロパティ設定
［ダウンロード・ファイルのプロパティ］エリアにおいて，追加するダウンロード・ファイルの選択とダウンロード条件を設定します。
表示される各項目において，次の設定を行ってください。
設定を完了すると，［ダウンロード・ファイル一覧］エリア内の空欄の項目に，ここで指定したファイル名が反映されます。

ファイル	追加するダウンロード・ファイル（インテル拡張ヘキサ・ファイル（*.hex）／モトローラ・S タイプ・ファイル（*.mot）／バイナリ・ファイル（*.bin））を指定します（最大指定文字数：259 文字）。		
	デフォルト	空欄	
	変更方法	キーボードからの直接入力，またはこのプロパティを選択すると右端に表示される［...］ボタンのクリックによりオープンするダウンロードするファイルを選択 ダイアログによる指定	
	指定可能値	「表 2.1 ダウンロード可能なファイル」参照	
ファイルの種類	追加するダウンロード・ファイルのファイル形式を選択します。 ここでは，［ロード・モジュール・ファイル］以外を選択します。		
	デフォルト	ロード・モジュール・ファイル	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	ロード・モジュール・ファイル	ロード・モジュール・フォーマット（*.abs）を指定します。
		ヘキサ・ファイル	インテル拡張ヘキサ・フォーマット（*.hex）を指定します。
		S レコード・ファイル	モトローラ・S タイプ・フォーマット（*.mot）を指定します。
		バイナリ・データ・ファイル	バイナリ・フォーマット（*.bin）を指定します。
オフセット	指定したファイルのダウンロードを開始するアドレスからのオフセット値を指定します。 なお，この項目は，［ファイルの種類］に［ヘキサ・ファイル］，または［S レコード・ファイル］を選択している場合のみ表示されます。		
	デフォルト	0	
	変更方法	キーボードからの直接入力	
	指定可能値	0x0 ～ 0xFFFFFFFF の 16 進数	
開始アドレス	指定したファイルをダウンロードする開始アドレスを指定します。 なお，この項目は，［ファイルの種類］に［バイナリ・データ・ファイル］を選択している場合のみ表示されます。		
	デフォルト	0	
	変更方法	キーボードからの直接入力	
	指定可能値	0x0 ～ 0xFFFFFFFF の 16 進数	

備考　オブジェクト情報，およびシンボル情報をダウンロードするか否かの指定は，ダウンロードするファイルの種類がロード・モジュール・ファイルの場合のみ行うことができます。

(3) ダウンロードの際の実行順序の確認
［ダウンロード・ファイル一覧］エリアでの表示順序が，ダウンロードの際の実行順序となります。
順序を変更する場合は［↓］／［↑］ボタンで変更してください。

(4) ［OK］ボタンのクリック
このダイアログでの設定を有効とし，指定したファイルがダウンロード・ファイルとして追加されます（プロパティ パネルの［ダウンロード・ファイル設定］タブ上の［ダウンロード］カテゴリ内に追加したファイル名とダウンロード条件が表示されます）。

2.5.2.3 複数のロード・モジュール・ファイルをダウンロードする

複数のロード・モジュール・ファイルをダウンロードする場合は，ダウンロード・ファイル ダイアログにおいて，次の手順の操作を行ってください。

注意　複数のロード・モジュール・ファイルから構成されるプログラムをデバッグする際は，配置アドレスが重ならないよう注意が必要です。

(1) ［追加］ボタンのクリック
［追加］ボタンをクリックすると，［ダウンロード・ファイル一覧］エリアの最終行に空欄の項目（“-”）が表示されます。

(2) 追加するダウンロード・ファイルのプロパティ設定
［ダウンロード・ファイルのプロパティ］エリアにおいて，追加するロード・モジュール・ファイルの選択とダウンロード条件を設定します。
表示される各項目において，次の設定を行ってください。
設定を完了すると，［ダウンロード・ファイル一覧］エリア内の空欄の項目に，ここで指定したファイル名が反映されます。

ファイル	追加するロード・モジュール・ファイルを指定します（最大指定文字数：259 文字）。		
	デフォルト	空欄	
	変更方法	キーボードからの直接入力，またはこのプロパティを選択すると右端に表示される［...］ボタンのクリックによりオープンするダウンロードするファイルを選択 ダイアログによる指定	
	指定可能値	「表 2.1 ダウンロード可能なファイル」参照	
ファイルの種類	追加するダウンロード・ファイルのファイル形式を指定します。 ここでは，［ロード・モジュール・ファイル］を選択します（デフォルト）。		
オブジェクトをダウンロードする	指定したファイルからオブジェクト情報をダウンロードするか否かを選択します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	オブジェクト情報をダウンロードします。
		いいえ	オブジェクト情報をダウンロードしません。
シンボルをダウンロードする	指定したファイルからシンボル情報をダウンロードするか否かを選択します[注 1]。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	シンボル情報をダウンロードします。
		いいえ	シンボル情報をダウンロードしません。

入力補完機能用の情報を生成する	ダウンロード時に，シンボル名の入力補完機能のための情報を生成するか否かを選択します注2。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	シンボル名の入力補完機能用の情報を生成します（入力補完機能を使用します）。
		いいえ	シンボル名の入力補完機能用の情報を生成しません（入力補完機能を使用しません）。

注 1. シンボル情報をダウンロードしない場合，ソース・レベル・デバッグを行うことはできません。

注 2. ［はい］を選択した場合，ダウンロード時間，およびホスト・マシンのメモリ消費量が増加します。シンボル名の入力補完機能を使用しない場合は，［いいえ］を選択することを推奨します。

備考 シンボル情報が不要なロード・モジュール・ファイルの場合では，［シンボルをダウンロードする］項目を［いいえ］に設定することにより，メモリの使用量を軽減することができます（ただし，このファイルをソース・レベルでデバッグすることはできません）。

(3) ダウンロードの際の実行順序の確認
［ダウンロード・ファイル一覧］エリアでの表示順序が，ダウンロードの際の実行順序となります。
順序を変更する場合は［↓］／［↑］ボタンで変更してください。

(4) ［OK］ボタンのクリック
このダイアログでの設定を有効とし，指定したロード・モジュール・ファイルがダウンロード・ファイルとして追加されます（プロパティ パネルの［ダウンロード・ファイル設定］タブ上の［ダウンロード］カテゴリ内に追加したファイル名が表示されます）。

2.5.2.4 ロード・モジュール・フォーマット以外のファイルでソース・レベル・デバッグを行う

インテル拡張ヘキサ・ファイル（*.hex），モトローラ・S タイプ・ファイル（*.mot），またはバイナリ・ファイル（*.bin）をダウンロード対象のファイルと指定している場合でも，これらのファイルの作成元となったロード・モジュール・ファイルのシンボル情報を併せてダウンロードすることにより，ソース・レベル・デバッグを行うことができます。
この場合は，ダウンロード・ファイル ダイアログにおいて，次の手順の操作を行ってください。

(1) ［追加］ボタンのクリック
［追加］ボタンをクリックすると，［ダウンロード・ファイル一覧］エリアの最終行に空欄の項目（“-”）が表示されます。

(2) ロード・モジュール・ファイルのプロパティ設定
［ダウンロード・ファイルのプロパティ］エリアにおいて，各項目を次のとおりに指定します。

ファイル	ダウンロードするインテル拡張ヘキサ・ファイル（*.hex），モトローラ・S タイプ・ファイル（*.mot），またはバイナリ・ファイル（*.bin）を作成する基となったロード・モジュール・ファイルを指定します。 キーボードからの直接入力，または［...］ボタンのクリックによりオープンするダウンロードするファイルを選択 ダイアログにより指定してください。
ファイルの種類	［ロード・モジュール・ファイル］を選択します（デフォルト）。
オブジェクトをダウンロードする	［いいえ］を選択します。
シンボルをダウンロードする	［はい］を選択します（デフォルト）。

入力補完機能用の情報を生成する	ダウンロード時に，シンボル名の入力補完機能のための情報を生成するか否かを選択します注。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	シンボル名の入力補完機能用の情報を生成します（入力補完機能を使用します）。
		いいえ	シンボル名の入力補完機能用の情報を生成しません（入力補完機能を使用しません）。

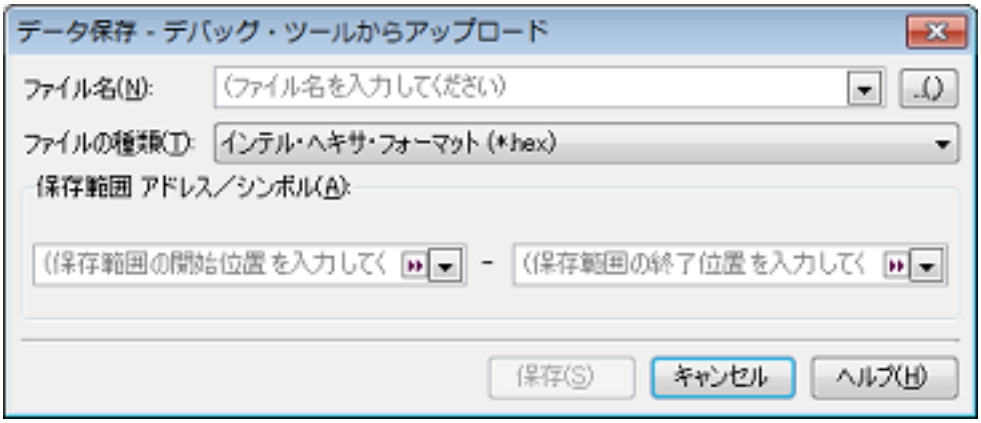
注　［はい］を選択した場合，ダウンロード時間，およびホスト・マシンのメモリ消費量が増加します。シンボル名の入力補完機能を使用しない場合は，［いいえ］を選択することを推奨します。

(3)　［OK］ボタンのクリック
このダイアログでの設定を有効とし，指定したロード・モジュール・ファイルがダウンロード・ファイルとして追加されます（ロード・モジュール・ファイル内に含まれるシンボル情報のみがダウンロードの対象となります）。

2.5.3 アップロードを実行する

現在接続しているデバッグ・ツールのメモリ内容を，任意のファイルに保存（アップロード）することができます。
アップロードは，［デバッグ］メニュー→［デバッグ・ツールからアップロード ...］を選択することによりオープンするデータ保存 ダイアログで行います。
このダイアログにおいて，次の手順で操作を行ってください。

図 2.52　メモリ内容のアップロード（データ保存 ダイアログ）



(1)　［ファイル名］の指定
保存するファイル名を指定します。
テキスト・ボックスに直接入力するか（最大指定文字数：259 文字），またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
また，［...］ボタンをクリックすることでオープンするデータ保存ファイルを選択 ダイアログにより，ファイルを選択することもできます。

(2)　［ファイルの種類］の指定
保存するファイルの形式を次のドロップダウン・リストにより選択します。
選択できるファイルの形式は次のとおりです。

表 2.2　アップロード可能なファイル形式

リスト表示	ファイル形式
インテル・ヘキサ・フォーマット (*.hex)	インテル拡張ヘキサ・フォーマット （常に拡張リニア・アドレス・レコードを使用）
モトローラ S フォーマット (*.mot)	モトローラ・S タイプ・フォーマット

リスト表示	ファイル形式
バイナリ・データ (*.bin)	バイナリ・フォーマット

(3) ［保存範囲 アドレス／シンボル］の指定
ファイルに保存する範囲を“開始アドレス”と“終了アドレス”で指定します。
それぞれのテキスト・ボックスに 16 進数の数値／アドレス式を直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。

備考　このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「シンボル名の入力補完機能」参照）。

(4) ［保存］ボタンのクリック
指定したファイルに指定した形式で，メモリ内容をアップロード・データとして保存します。

2.6 プログラムの表示と変更

この節では，デバッグ情報を持ったロード・モジュール・ファイルをデバッグ・ツールにダウンロードした場合のプログラムの表示方法，およびその変更方法について説明します。
ダウンロードしたプログラムの表示は，次の 2 つのパネルで行うことができます。

- エディタ パネル
  ソース・ファイルの表示／編集を行うほか，ソース・レベル・デバッグ／命令レベル・デバッグ（「プログラムをステップ実行する」参照），およびコード・カバレッジ測定結果の表示【シミュレータ】（「カバレッジの測定【シミュレータ】」参照）を行います。

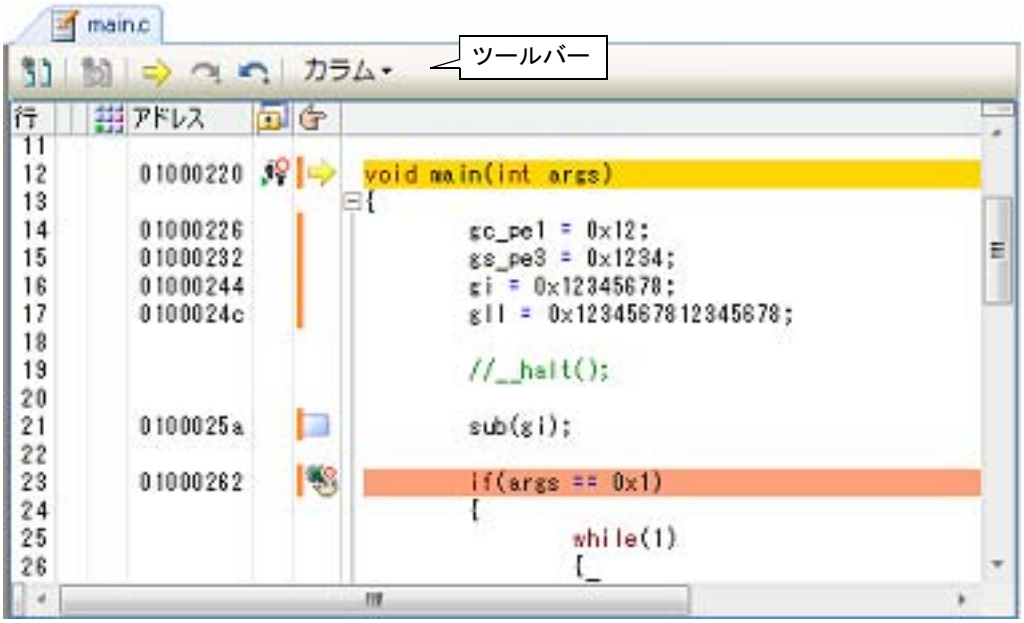
- 逆アセンブル パネル
  ダウンロードしたプログラム（メモリ内容）の逆アセンブル結果の表示／編集（ライン・アセンブル）を行うほか，命令レベル・デバッグ（「2.8.3 プログラムをステップ実行する」参照），およびコード・カバレッジ測定結果の表示【シミュレータ】（「2.14 カバレッジの測定【シミュレータ】」参照）を行います。
  なお，このパネルでは，逆アセンブル結果の表示とともに，対応するソース・テキストも表示することができます（デフォルト）。

備考　通常，ソース・レベル・デバッグを行うためには，デバッグ情報を持つロード・モジュール・ファイルをダウンロードする必要がありますが，インテル拡張ヘキサ・ファイル（*.hex），モトローラ・S タイプ・ファイル（*.mot），またはバイナリ・ファイル（*.bin）をダウンロード対象として，ソース・レベル・デバッグを行うことも可能です（「2.5.2.4 ロード・モジュール・フォーマット以外のファイルでソース・レベル・デバッグを行う」参照）。

2.6.1 ソース・ファイルを表示する

ソース・ファイルの表示は，次のエディタ パネルで行います。
エディタ パネルは，ロード・モジュール・ファイルが正常にダウンロードされると，指定された位置（「2.5.1 ダウンロードを実行する」参照）のソース・テキストを表示した状態で自動的にオープンします。
手動でエディタ パネルをオープンする場合は，プロジェクト・ツリー パネルにおいてソース・ファイルをダブルクリックしてください。
なお，各エリアの見方，および機能についての詳細は，「CS+ 統合開発環境 ユーザーズマニュアル エディタ編」を参照してください。

図 2.53　ソース・ファイルの表示（エディタ パネル）



2.6.2 逆アセンブル結果を表示する

ダウンロードしたプログラム（メモリ内容）の逆アセンブル結果（逆アセンブル・テキスト）の表示は，次の逆アセンブル パネルで行います。
［表示］メニュー→［逆アセンブル］→［逆アセンブル *1* ～ *4*］を選択してください。
逆アセンブル パネルは，最大 4 個までオープンすることができ，各パネルはタイトルバーの“逆アセンブル 1”，“逆アセンブル 2”，“逆アセンブル 3”，“逆アセンブル 4” の名称で識別されます。
なお，各エリアの見方，および機能についての詳細は，逆アセンブル パネルの項を参照してください。

図 2.54 逆アセンブル結果の表示（逆アセンブル パネル）

備考 ツールバーの［表示］→ ボタンをクリックすることによりオープンするスクロール範囲設定 ダイアログにより，このパネルの垂直スクロール・バーのスクロール範囲を設定することができます。

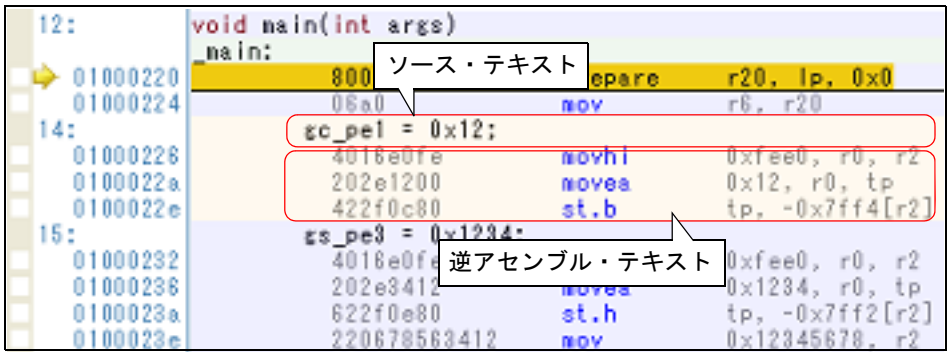
ここでは，次の操作方法について説明します。

2.6.2.1 表示モードを変更する

ツールバーの ボタン（トグル）をクリックすることにより，逆アセンブル パネルの表示モードを切り替えることができます。

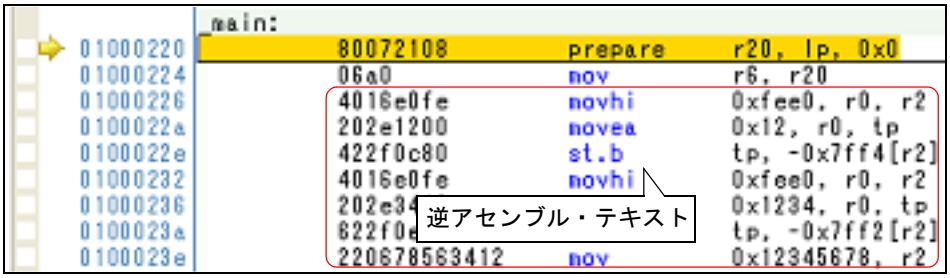
- 混合表示モード
  ソース・テキストと逆アセンブル・テキストを併せて表示する，デフォルトの表示モードです。

図 2.55 混合表示モード（逆アセンブル パネル）



- 逆アセンブル表示モード
ソース・テキストを非表示にし，逆アセンブル・テキストのみを表示します。

図 2.56 逆アセンブル表示モード（逆アセンブル パネル）



2.6.2.2 表示形式を変更する

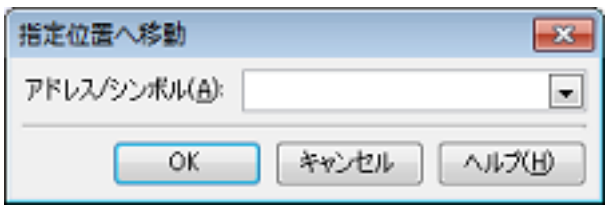
ツールバーの次のボタンにより，逆アセンブル結果の表示形式を変更することができます。

表示	表示形式を変更する次のボタンを表示します。
	ラベルのオフセット値を表示します。アドレスにラベルが定義されていない場合，一番近いラベルからのオフセット値を表示します。
	アドレス値を“シンボル＋オフセット値”で表示します（デフォルト）。 ただし，アドレス値にシンボルが定義されている場合は，シンボルのみを表示します。
	レジスタ名を機能名称で表示します（デフォルト）。
	レジスタ名を絶対名称で表示します。

2.6.2.3 指定アドレスへ移動する

逆アセンブル・テキスト上の指定アドレスへの移動は，コンテキスト・メニューの［移動 ...］を選択することによりオープンする指定位置へ移動 ダイアログで行います。
このダイアログにおいて，次の手順で操作を行ってください。

図 2.57 逆アセンブル結果内のアドレスへ移動（指定位置へ移動 ダイアログ）



(1) ［アドレス / シンボル］の指定
キャレットを移動したいアドレスを指定します。

テキスト・ボックスにアドレス式を直接入力するか（最大指定文字数：1024 文字），またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。

備考　このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

(2)　［OK］ボタンのクリック
指定したアドレスへキャレットを移動します。

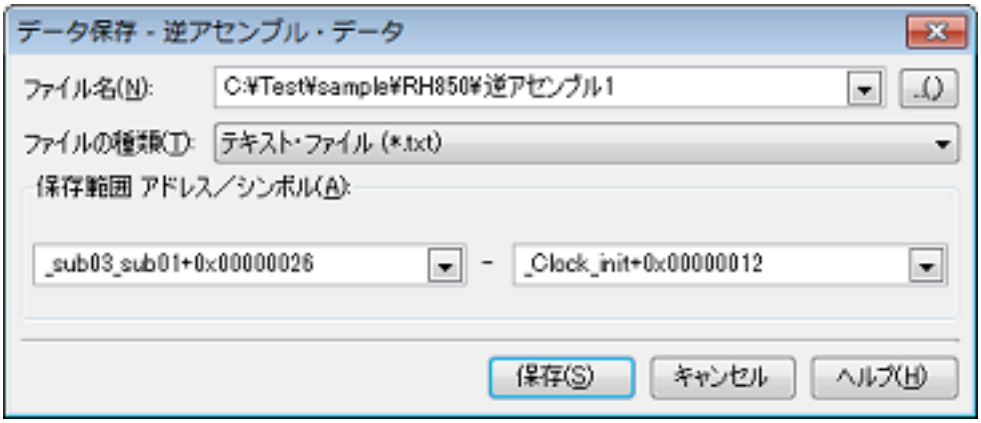
2.6.2.4 シンボル定義箇所へ移動する

シンボルが定義されているアドレスに，キャレット位置を移動することができます。
シンボルを参照している命令にキャレットを移動したのち，ツールバーの  ボタンをクリックしてください。
また，この操作に続き，ツールバーの  ボタンをクリックすると，キャレット移動前のシンボルを参照している命令にキャレット位置を戻します。

2.6.2.5 逆アセンブル結果の表示内容を保存する

逆アセンブル結果の内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
ファイルに保存する際は，デバッグ・ツールから最新の情報を取得し，このパネル上での表示形式に従ったデータで保存します。
［ファイル］メニュー→［名前を付けて逆アセンブル・データを保存 ...］を選択すると，次のデータ保存 ダイアログがオープンします（この際，パネル上で範囲選択した状態でこの操作を行うと選択範囲のみの逆アセンブル・データを保存することができます）。
このダイアログにおいて，次の手順で操作を行ってください。

図 2.58　逆アセンブル・データの保存（データ保存 ダイアログ）



(1)　［ファイル名］の指定
保存するファイル名を指定します。
テキスト・ボックスに直接入力するか（最大指定文字数：259 文字），またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
また，［...］ボタンをクリックすることでオープンするデータ保存ファイルを選択 ダイアログにより，ファイルを選択することもできます。

(2)　［ファイルの種類］の指定
保存するファイルの形式を次のドロップダウン・リストにより選択します。
選択できるファイルの形式は次のとおりです。

リスト表示	形式
テキスト・ファイル (*.txt)	テキスト形式（デフォルト）
CSV(カンマ区切り) (*.csv)	CSV 形式注

注　各データを “,” で区切り保存します。
なお，データ内に “,” が含まれている際の不正形式を避けるため，各データを "" （ダブルクォーテーション）で括り出力します。

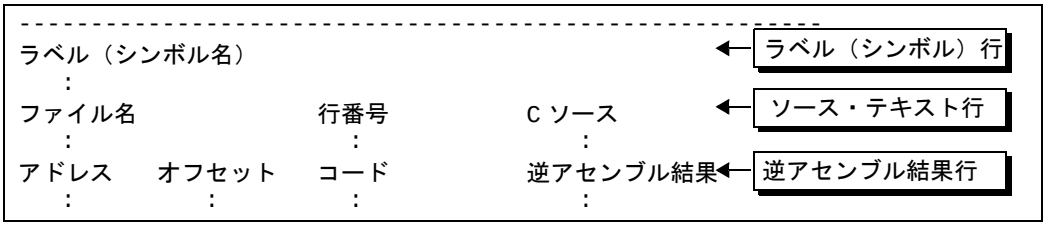
(3)　［保存範囲 アドレス／シンボル］の指定

ファイルに保存する範囲を“開始アドレス”と“終了アドレス”で指定します。
それぞれのテキスト・ボックスに 16 進数の数値／アドレス式を直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
なお，パネル上で範囲選択している場合は，デフォルトでその選択範囲がテキスト・ボックスに指定されます。範囲選択していない場合は，現在のパネルの表示範囲が指定されます。

備考　　このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

(4)　［保存］ボタンのクリック
指定したファイルに，指定した形式で逆アセンブル・データを保存します。

図 2.59　　逆アセンブル・データ保存の際の出力イメージ



備考 1.　　［ファイル］メニュー→［逆アセンブル・データを保存］の選択によりパネルの内容を上書き保存する場合，逆アセンブル パネル（逆アセンブル 1 ～ 4）はそれぞれ個別に扱われます。
また，保存範囲についても，前回指定したアドレス範囲で保存されます。

備考 2.　　［ファイル］メニュー→［印刷 ...］を選択することにより，現在このパネルで表示しているの画像イメージを印刷することができます。

2.6.3 他の処理と平行してビルドを実行する

CS+ では，次のタイミングでビルドを自動で開始する機能を提供しています（ラピッド・ビルド機能）。

- デバッグ専用プロジェクト以外の場合
  - プロジェクトに追加している C ソース・ファイル／アセンブリ・ソース・ファイル／ヘッダ・ファイル／シンボル情報ファイル／オブジェクト・モジュール・ファイル／ライブラリ・ファイルのいずれかを更新したとき
  - プロジェクトにビルド対象ファイルを追加，または削除したとき
  - オブジェクト・モジュール・ファイル，およびライブラリ・ファイルのリンク順を変更したとき
  - ビルド・ツール，およびビルド対象ファイルのプロパティを変更したとき

- デバッグ専用プロジェクトの場合
  - デバッグ専用プロジェクトに追加している C ソース・ファイル／アセンブリ・ソース・ファイル／ヘッダ・ファイルを編集して保存したとき
  - デバッグ専用プロジェクトに C ソース・ファイル／アセンブリ・ソース・ファイル／ヘッダ・ファイルを追加，または削除したとき
  - デバッグ専用プロジェクトのプロパティを変更したとき

ラピッド・ビルド機能を有効にすることにより，上記の操作と平行してビルドを行うことができます。
ラピッド・ビルド機能の有効／無効の設定は，［ビルド］メニュー→［ラピッド・ビルド］の選択により切り替えます（デフォルトで有効に設定されています）。

注意　　外部エディタを使用する場合，この機能を有効にするためには，オプション ダイアログの［ビルド／デバッグ］カテゴリの［登録されたファイルの変更を監視する］をチェックする必要があります。

備考 1.　　ソース・ファイル編集後，［Ctrl］＋［S］キーの押下により，こまめに上書き保存することを推奨します。

備考 2.　　ラピッド・ビルドの有効／無効は，プロジェクト全体（メイン・プロジェクト，およびサブプロジェクト）に対して設定されます。

備考 3.　ラピッド・ビルドの実行中に，ラピッド・ビルドを無効に切り替えた場合は，その場でラピッド・ビルドの実行を中止します。

2.6.4 ライン・アセンブルを行う

逆アセンブル パネルで表示されている命令／命令コードは，編集（ライン・アセンブル）することができます。
ここでは，次の操作方法について説明します。

2.6.4.1 命令を編集する

命令を編集する場合は，次の手順で操作を行ってください。

(1)　編集モードへの切り替え
対象命令をダブルクリックするか，または対象命令にキャレットを移動した状態でコンテキスト・メニューの［命令の編集］を選択すると，編集対象が編集モードに切り替わります。

(2)　命令の編集
キーボードから直接命令の文字列を編集します。

(3)　メモリへの書き込み
編集終了後，［Enter］キーを押下することにより，変更された命令が自動的にライン・アセンブルされ，コードがメモリに書き込まれます。
ただし，この際に，変更結果が不正な命令となる場合は，編集された文字列が赤色で表示され，メモリへの書き込みは行いません。

なお，表示されている逆アセンブル結果を別の命令で上書きすることによりメモリに空きが生じた場合，次の例のように自動的に nop 命令でバイト数を補います。

例 1.　3 行目の prepare 命令（8 バイト命令）を jarl 命令（4 バイト命令）で上書きした場合

編集前	0432	mov	0x4, r6
	1d38	mov	r29, r7
	8f071b0effff0000	prepare	r20, r21, r22, 0x1c, 0x0000ffff
	0132	mov	0x1, r6
編集後	0432	mov	0x4, r6
	1d38	mov	r29, r7
	bfffe265	jarl	0x100, lp
	0000	nop	
	0000	nop	
	0132	mov	0x1, r6

例 2.　2 行目の mov 命令（2 バイト命令）を jarl 命令（4 バイト命令）で上書きした場合

編集前	0432	mov	0x4, r6
	1d38	mov	r29, r7
	8f071b0effff0000	prepare	r20, r21, r22, 0x1c, 0x0000ffff
	0132	mov	0x1, r6
編集後	0432	mov	0x4, r6
	bfffe265	jarl	0x100, lp
	0000	nop	
	0000	nop	
	0000	nop	
	0132	mov	0x1, r6

注意　prepare 命令 /dispose 命令の扱いについて
prepare 命令 /dispose 命令の命令形式は，次のとおりで，オペランドの “list12” には，12 ビットの値が入り，各ビットごとに対応するレジスタが割り当てられます。

prepare 命令の命令形式	prepare　　list12, imm5
	prepare　　list12, imm5, sp/imm
dispose 命令の命令形式	dispose　　imm5, list12
	dispose　　imm5, list12, [reg1]

逆アセンブル パネルでは，prepare 命令 /dispose 命令の逆アセンブル結果を表示する場合，オペランドの “list12” は値ではなく，次の例のように対応するレジスタ名を表示します。

例 1. 命令コードが “0x91, 0x07, 0xe1, 0xff ”（prepare の 4 バイト命令）の場合

表示	prepare　　r20, r21, r22, r23, r24, r25, r26, r27,r28, r29, r30, r31, 0x20
正表記	prepare　　0xfff, 0x20

例 2. 命令コードが “0x90, 0x07, 0xbb, 0xaa 0xff, 0xff, 0xff, 0xff ”（prepare の 8 バイト命令）の場合

表示	prepare　　r20, r22, r24, r26, r28, r31, 0x20, 0x7fffffff
正表記	prepare　　0x555, 0x20, 0x7fffffff

例 3. 命令コードが “0x51, 0x06, 0xe0, 0xff ”（dispose の 4 バイト命令）の場合

表示	dispose　　0x20, r20, r21, r22, r23, r24, r25, r26, r27, r28, r29, r30, r31
正表記	dispose　　0x20, 0xfff

例 4. 命令コードが “0x50, 0x06, 0xaa, 0xaa ”（dispose の 4 バイト命令）の場合

表示	dispose　　0x20, r20, r22, r24, r26, r28, r31, [r10]
正表記	dispose　　0x20, 0x555, [r10]

ただし，prepare 命令 /dispose 命令をライン・アセンブルする場合は，オペランドの “list12” には，値／レジスタ名の両方の指定が可能です。

例 1. （1）と（2）の指定は，ライン・アセンブルの結果，同じ値の “0x91, 0x07, 0xe1, 0xff” となります。

（1）	prepare　　r20, r21, r22, r23, r24, r25, r26, r27,r28, r29, r30, r31, 0x20
（2）	prepare　　0xfff, 0x20

例 2. （1）と（2）の指定は，ライン・アセンブルの結果，同じ値の “0xbe, 0x07, 0xbb, 0xaa 0xff, 0xff, 0xff, 0x7f ” となります。

（1）	prepare　　r20, r22, r24, r26, r28, r31, 0x20, 0x7fffffff
（2）	prepare　　0x555, 0x20, 0x7fffffff

例 3. （1）と（2）の指定は，ライン・アセンブルの結果，同じ値の “0x51, 0x06, 0xe0, 0xff ” となります。

（1）	dispose　　0x20, r20, r21, r22, r23, r24, r25, r26, r27, r28, r29, r30, r31
（2）	dispose　　0x20, 0xfff

例 4.　（1）と（2）の指定は，ライン・アセンブルの結果，同じ値の “0x50, 0x06, 0xaa, 0xaa ” となります。

（1）	dispose　0x20, r20, r22, r24, r26, r28, [r10]
（2）	dispose　0x20, 0x555, [r10]

2.6.4.2 命令コードを編集する

命令コードを編集する場合は，次の手順で操作を行ってください。

(1) 編集モードへの切り替え
対象命令コードをダブルクリックするか，または対象命令コードにキャレットを移動した状態で表示されるコンテキスト・メニューの［コードの編集］を選択すると，編集対象が編集モードに切り替わります。

(2) 命令コードの編集
キーボードから直接命令コードの文字列を編集します。

(3) メモリへの書き込み
編集終了後，［Enter］キーを押下することにより，命令コードがメモリに書き込まれます。
ただし，この際に，変更結果が不正な命令となる場合は，編集された文字列が赤色で表示され，メモリへの書き込みは行いません。
命令コードがメモリに書き込まれた場合は，逆アセンブル結果も同時に更新されます。

2.7 コア（PE）の選択

この節では，選択しているマイクロコントローラがマルチコア対応版の場合における，デバッグ対象となるコア（PE：プロセッサ・エレメント）の選択方法について説明します。
CS+ では，デバッグ対象とするコア（PE）の選択を切り替えることにより（「2.7.1 コア（PE）を切り替える」参照），PE ごとの情報を表示します（PE ごとに複数のパネルの表示は行いません）。
なお，マルチコア対応版を対象とした CS+ の各機能の振る舞いは次のとおりです。

(1) プログラムの実行制御
全 PE において，原則として同期実行／同期ブレークを行います。
ただし，ステップ実行については，次の動作となります。

- 【Full-spec emulator】【E1】【E20】
命令レベル単位で 1 命令ずつ実行します。

- 【シミュレータ】
動作周波数に従い同期してステップ実行します。

注意　ステップ実行は，現在選択している PE でのみ行います。
ただし，ソース・レベル単位のステップ実行の場合は，選択外の PE が実行される場合があります。

(2) イベントの発生
イベントは全 PE で有効となるように自動設定されます。

注意 1.　【Full-spec emulator】【E1】【E20】
Local RAM self 領域については，現在選択している PE にブレークポイントを設定します。

注意 2.　【シミュレータ】
Local RAM self 領域へのアクセスであれば，それがどの PE からのアクセスであってもイベントが発生します。
ただし，Local RAM self 領域の実態に直接アクセスした場合，イベントは発生しません。

(3) メモリ／レジスタ／変数などの情報

(a) メモリ・マップ
現在選択している PE により，メモリ・マップが異なる場合があります。
この場合，プロパティ パネルの［デバッグ・ツール設定］タブ上［メモリ］カテゴリ，およびメモリ・マッピング ダイアログでは，PE を切り替えることによって対応したメモリ・マップを表示します。

(b) メモリ範囲と値
現在選択している PE にかかわらず，同じ値を表示／設定します。
ただし，Local RAM self 領域については，現在選択している PE の値を取得し表示／設定します。

(c) レジスタ（IOR/PC レジスタを含む）の値

現在選択している PE の値を取得し表示／設定します。

(d) シンボル（ウォッチ式／変数名を含む）
現在選択している PE の PC 値を基にアドレスと値を決定します（たとえば，シンボルが特定の PE のみで有効であった場合でも，現在選択している PE を基にアドレスと値を決定します）。

(e) コール・スタック情報
現在選択している PE の値を取得し表示／設定します。

(4) その他の機能

(a) 実行履歴の収集

-【Full-spec emulator】【E1】【E20】
動作は，プロパティ パネルの［デバッグ・ツール設定］タブの［トレース］カテゴリ内［トレースの取得対象設定］プロパティの指定に依存します。

- ［デバッグ対象のコアのみ］を選択している場合（デフォルト）
現在選択している PE を対象にトレース・データを収集します。
したがって，目的のトレース・データを収集するためには，プログラムを実行する前に，PE の選択を行う必要があります（トレース・データ収集後に PE を切り替えても，トレース パネルの表示内容は変化しません）。

- ［全てのコア］を選択している場合
全 PE を対象にトレース・データの収集を行います。
トレース・データ収集後，トレース パネルでは，PE を切り替えることによって対応したトレース・データを表示します。

-【シミュレータ】
現在選択している PE を対象にトレース・データを収集します。
したがって，目的のトレース・データを収集するためには，プログラムを実行する前に，PE の選択を行う必要があります（トレース・データ収集後に PE を切り替えても，トレース パネルの表示内容は変化しません）。

(b) 実行時間の計測
全 PE を対象に実行時間の計測を行います。
計測完了後，PE を切り替えることによって対応した測定結果を表示します。

(c) カバレッジ測定
全 PE のアクセスを対象にカバレッジ測定を行います。
ただし，Local RAM self 領域については，現在選択している PE のアクセスのみを対象に測定結果を表示します。

2.7.1 コア（PE）を切り替える

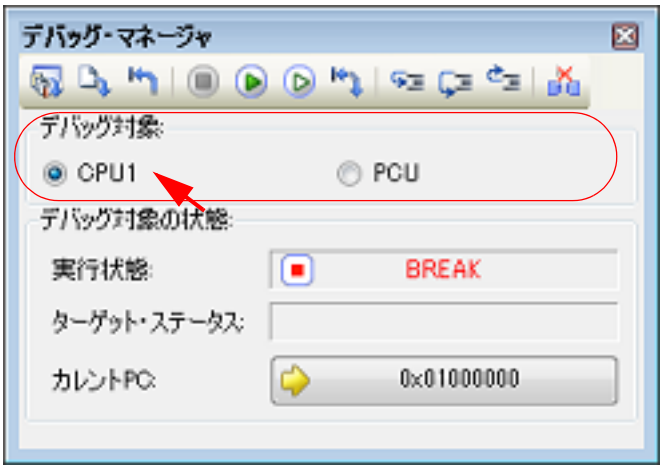
デバッグ対象とするコア（PE）の切り替えは，次のいずれかの方法により行うことができます。

(1) ステータスバーで切り替える
メイン・ウインドウのステータスバー上の次のドロップダウン・リストにより，任意の PE を選択します。

図 2.60　　メイン・ウインドウのステータスバー

(2) デバッグ・マネージャー パネルで切り替える
［表示］メニュー→［デバッグ・マネージャ］を選択することによりオープンするデバッグ・マネージャ パネルにおいて，任意の PE を選択します。

図 2.61　　デバッグ・マネージャ パネル



2.8 プログラムの実行

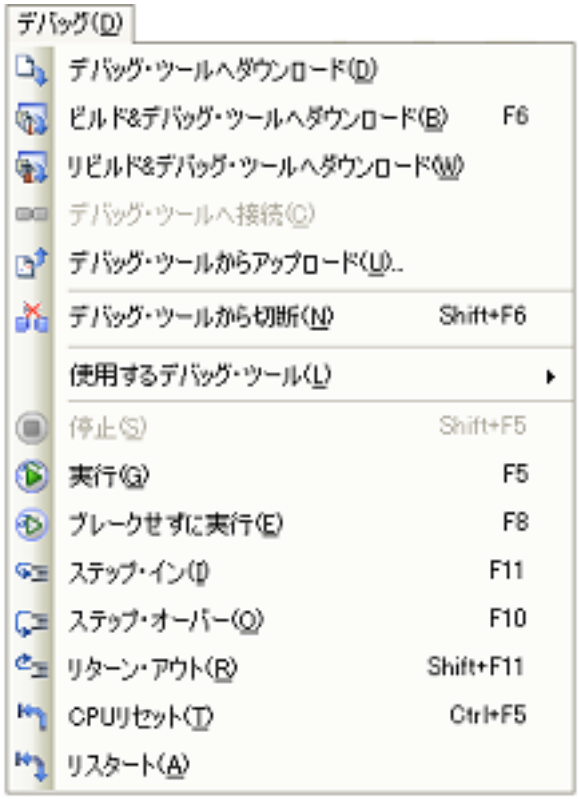
この節では，プログラムの実行方法について説明します。
なお，この節で説明する主な操作は，プログラムの実行を制御するためのコマンドをまとめたメイン・ウインドウ上のデバッグ・ツールバー，または［デバッグ］メニューより行います。

注意　デバッグ・ツールバー，および［デバッグ］メニューの各項目は，デバッグ・ツールと接続時のみ有効となります。

備考　マルチコア対応版を対象とした“プログラムの実行制御”については，「2.7 コア（PE）の選択」も参照してください。

図 2.62　デバッグ・ツールバー（フローティング状態）

図 2.63　［デバッグ］メニュー



2.8.1 マイクロコントローラ（CPU）をリセットする

デバッグ・ツールバーの  ボタンをクリックすることにより，CPU をリセットします。
CPU をリセットすることにより，カレント PC 値をリセット番地に設定します。

備考　ブレーク中の CPU リセット後に，I/O レジスタ /CPU レジスタの値を指定した値に自動的に書き換える処理を設定することができます（「2.17 フック処理を設定する」参照）。

2.8.2 プログラムを実行する

プログラムの実行方法には次の種類があります。
デバッグの目的に応じて実行方法を選択してください。
なお，実行中のプログラムの停止方法については，「2.9 プログラムの停止（ブレーク）」を参照してください。

2.8.2.3 PC 値を変更してから実行する

備考　プログラムの実行開始直前に，I/O レジスタ /CPU レジスタ値を指定した値に自動的に書き換える処理を設定することができます（「2.17 フック処理を設定する」参照）。

2.8.2.1 マイクロコントローラ（CPU）をリセットしてから実行する

CPU をリセットしたのち，リセット番地からプログラムの実行を開始します。
操作は，デバッグ・ツールバーの  ボタンをクリックします。
この操作によりプログラムの実行を開始した場合，次のいずれかの状態までその実行を続けます。

-  ボタンのクリック（「2.9.2 プログラムの実行を手動で停止する」参照）
- PC がブレークポイントに到達（「2.9.3 任意の場所で停止する（ブレークポイント）」参照）
- ブレーク・イベント条件の成立（「2.9.4 任意の場所で停止する（ブレーク・イベント）」／「2.9.5 変数 /I/O レジスタへのアクセスで停止する」参照）
- その他のブレーク要因の発生

備考　この操作は，  ボタンをクリックしたのち，  ボタンをクリックした場合と同等です。

2.8.2.2 現在のアドレスから実行する

現在のアドレス（カレント PC 値で示されるアドレス）からプログラムの実行を開始する方法には，次の種類があります。

(1) 通常の実行
デバッグ・ツールバーの  ボタンをクリックします。
この操作により実行を開始した場合，次のいずれかの状態までその実行を続けます。

-  ボタンのクリック（「2.9.2 プログラムの実行を手動で停止する」参照）
- PC がブレークポイントに到達（「2.9.3 任意の場所で停止する（ブレークポイント）」参照）
- ブレーク・イベント条件の成立（「2.9.4 任意の場所で停止する（ブレーク・イベント）」／「2.9.5 変数 /I/O レジスタへのアクセスで停止する」参照）
- その他のブレーク要因の発生

(2) ブレーク関連のイベントを無視した実行
デバッグ・ツールバーの  ボタンをクリックします。
この操作により実行を開始した場合，次のいずれかの状態までその実行を続けます。

-  ボタンのクリック（「2.9.2 プログラムの実行を手動で停止する」参照）
- その他のブレーク要因の発生

備考　この操作により実行を開始した場合，アクション・イベントの発生も無視されます。

(3) キャレット位置までの実行
エディタ パネル／逆アセンブル パネルにおいて，プログラムを停止させたい行／命令にキャレットを移動したのち，コンテキスト・メニューの［ここまで実行］を選択します。
この操作により実行を開始した場合，次のいずれかの状態までその実行を続けます。

- PC がキャレット位置のアドレスに到達
-  ボタンのクリック（「2.9.2 プログラムの実行を手動で停止する」参照）
- その他のブレーク要因の発生

注意　キャレット位置の行に対応するアドレスが存在しない場合は，下方向の有効な行までプログラムを実行します（有効な行が存在しない場合は，エラーとなります）。

備考　この操作により実行を開始した場合，アクション・イベントの発生も無視されます。

2.8.2.3 PC 値を変更してから実行する

カレント PC 値を任意のアドレスに強制的に変更したのち，プログラムを実行します。
この操作を行うには，まず，エディタ パネル／逆アセンブル パネルにおいて，プログラムの実行を開始したい行／命令にキャレットを移動したのち，コンテキスト・メニューの［PC をここに設定］を選択します（カレント PC 値が現在キャレットのある行／命令のアドレスに変更されます）。
次に，「2.8.2.2 現在のアドレスから実行する」で示した，いずれかの実行方法を行います。

2.8.3 プログラムをステップ実行する

次のいずれかの操作を行うと，現在のアドレス（カレント PC 値で示されるアドレス）から，ソース・レベル単位（ソース・テキスト 1 行分），または命令レベル単位（1 命令分）でプログラムをステップ実行したのち，自動的に停止します。
プログラムの停止後は逐一各パネルの内容が自動的に更新されるため，ステップ実行は，プログラムの実行遷移をソース・レベル単位／命令単位でデバッグする場合に有効な実行方法です。
なお，ステップ実行を行う際の実行単位は，次の設定に依存します。

- エディタ パネルのツールバーの  ボタンを無効にしている場合（デフォルト）
  ソース・レベル単位によるステップ実行を行います。
  ただし，逆アセンブル パネルにフォーカスがある場合，またはカレント PC 値で示されるアドレスに行情報が存在しない場合は，命令レベル単位によるステップ実行を行います。

- エディタ パネルのツールバーの  ボタンを有効にしている場合
  命令レベル単位によるステップ実行を行います。

  注意  ボタンは，エディタ パネルを混合表示モードに設定している場合のみ有効となります。

ステップ実行には，次の種類があります。

2.8.3.1 関数内にステップ・インする（ステップ・イン実行）
2.8.3.2 関数をステップ・オーバする（ステップ・オーバ実行）
2.8.3.3 関数内でリターンが完了するまで実行する（リターン・アウト実行）

注意 1. ステップ実行中は，設定されているブレークポイント／ブレーク・イベント／アクション・イベントを発生しません。

注意 2. 関数のプロローグ／エピローグ処理中，および戻りアドレスが取得できない場合は，エラー・メッセージを表示します。

注意 3. 【Full-spec emulator】【E1】【E20】

- ステップ実行中は，割り込みが禁止されます。
- ステップ実行中は，スタンバイ・モードに移行しません。

注意 4. 【シミュレータ】
ステップ実行中に割り込みハンドラに飛ぶことがあります。

2.8.3.1 関数内にステップ・インする（ステップ・イン実行）

関数呼び出しの場合，呼び出された関数内の先頭で停止するステップ実行です。
操作は，デバッグ・ツールバーの  ボタンをクリックします。

注意 1. デバッグ情報がない関数へのステップ・イン実行はできません。

注意 2. longjmp 関数へのステップ・イン実行は，実行処理が完了せずタイムアウト待ちになることがあります。

注意 3. 関数の入口の処理（プロローグ処理）はスキップされません。プロローグ処理をスキップさせたい場合は，再度ステップ・イン実行してください。

2.8.3.2 関数をステップ・オーバする（ステップ・オーバ実行）

jarl 命令による関数呼び出しの場合，その関数内のソース行／命令すべてを 1 ステップとみなして実行し，関数から戻った箇所で停止するステップ実行です（jarl 命令を実行したときと同じネストになるまで，ステップ実行します）。
操作は，デバッグ・ツールバーの  ボタンをクリックします。
なお，jarl 命令以外の場合は，  ボタンのクリックと同じ動作となります。

注意　longjmp 関数のステップ・オーバ実行は，実行処理が完了せずタイムアウト待ちになることがあります。

2.8.3.3 関数内でリターンが完了するまで実行する（リターン・アウト実行）

現在の関数から，呼び出し元関数に戻った箇所で停止するステップ実行します。
ある関数内において確認が必要なソース行／命令の実行が終了した際などに，この命令によるステップ実行を行うと，残りの関数内の命令をステップ実行せずに呼び出し元の関数に戻ることができます。
操作は，デバッグ・ツールバーの  ボタンをクリックします。

注意 1.　main 関数内でのリターン・アウト実行は，スタートアップ・ルーチン内でブレークします。

注意 2.　関数にステップ・インした直後にリターン・アウト実行はできません。

注意 3.　関数のプロローグ／エピローグ処理中からリターン・アウト実行はできません。

注意 4.　longjmp 関数の呼び出し元関数内でリターン・アウト実行すると，ブレークしないことがあります。

注意 5.　再帰関数からリターン・アウト実行を行うと，フリーラン状態となります。

2.9 プログラムの停止（ブレーク）

この節では，実行中のプログラムを停止する方法について説明します。
CS+ では，次のブレーク機能を使用して任意の箇所でプログラムを停止させることができます。

(1) 強制ブレーク機能
強制的にプログラムの実行を停止する機能です。

(2) ハードウエア・ブレーク機能
デバッグ・ツールが，ハードウエアの資源を使用してプログラム実行中にブレーク条件を逐次確認し，条件を満たした際にプログラムを停止させる機能です。
ハードウエア・ブレーク・イベントには，任意の箇所でプログラムの実行を停止させる“実行系”と，任意の変数などに指定したアクセスがあった際にプログラムの実行を停止させる“アクセス系”があります。
なお，ハードウエア・ブレーク・イベント（実行系）を設定すると，指定したアドレスの命令実行前にプログラムがブレークします（実行前ブレーク）。

備考 ハードウエア・ブレーク・イベント（アクセス系）を使用する場合では（「2.9.5.1 ブレーク・イベント（アクセス系）を設定する」参照），次の場合のみ“実行後ブレーク”となります。

- コンテキスト・メニューの［ブレークの設定］→［読み込みブレークを設定］／［読み書きブレークを設定］の選択によるブレーク・イベントの設定において，データ条件を設定した場合
- コンテキスト・メニューの［ブレークの設定］→［書き込みブレークを設定］／［読み書きブレークを設定］の選択によるブレーク・イベントの設定において，リードモディファイライト系の命令のライト・アクセスを検出した場合

(3) ソフトウエア・ブレーク機能【Full-spec emulator】【E1】【E20】
指定したアドレスの命令コードを一時的にブレーク用の命令に書き換え，その命令を実行した際にプログラムを停止させる機能です。
ソフトウエア・ブレーク・イベントを設定すると，指定したアドレスの命令実行前にプログラムがブレークします（実行前ブレーク）。

注意 命令コードをブレーク用の命令に書き換えるため，ソフトウエア・ブレーク・イベントの設定／削除を行うたびに，次のタイミングでフラッシュ・メモリの書き換えが行われます。

- プログラムの実行開始時（［デバッグ］メニュー→［ブレークせずに実行］の選択を含む）
- デバッグ・ツールと切断時

注意 1. スタンバイ・モード（HALT/STOP/IDLE）中に強制ブレークを行った場合，カレント PC 値はスタンバイ・モード命令以降の次命令のアドレスとなります。
また，使用するデバッグ・ツールによって，次のように動作が異なります。

- 【Full-spec emulator】**【E1】【E20】**
  強制ブレークによりスタンバイ・モードを解除します。
- 【シミュレータ】
  強制ブレークによりスタンバイ・モードを解除しません。
  スタンバイ・モードが解除されているように見えますが，スタンバイ・モードが解除されているか否かは，メイン・ウインドウのステータス・バー上の CPU 状態で確認してください。

注意 2. 【Full-spec emulator】【E1】【E20】
ブレーク時にターゲット・システムの電圧を下げないようにしてください。ブレーク中に低電圧検出回路（LVI），またはパワーオン・クリア（POC）によるリセットが発生した場合，CS+ の不正動作や通信エラーの原因となる場合があります。
なお，ターゲット電源 OFF のエミュレーション中でのブレークもこれに該当します。

備考 1. マルチコア対応版を対象とした“プログラムの実行制御”，または“イベントの発生”については，「2.7 コア（PE）の選択」も参照してください。

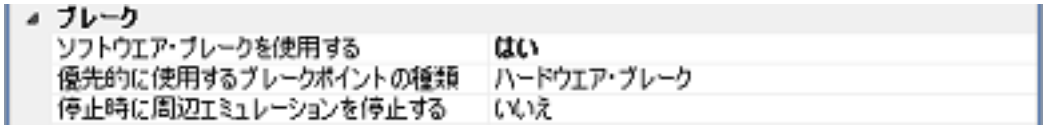
備考 2. 実行中のプログラムが停止すると，その原因（ブレーク要因）がメイン・ウインドウのステータスバーに表示されます。

2.9.1 ブレーク動作の設定をする【Full-spec emulator】【E1】【E20】

ブレーク機能を使用するためには，あらかじめブレーク動作に関する設定を行う必要があります。
ブレーク動作の設定は，プロパティ パネルの［デバッグ・ツール設定］タブ上の［ブレーク］カテゴリ内で行います。

備考　【シミュレータ】
ブレーク動作の設定は必要ありません。

図 2.64　［ブレーク］カテゴリ



(1)　［ソフトウエア・ブレークを使用する］
ソフトウエア・ブレーク機能【Full-spec emulator】【E1】【E20】を使用するか否かを選択します。
ソフトウエア・ブレーク機能を使用する場合は［はい］を選択してください（デフォルト：［いいえ］）。

注意 1.　1 度ソフトウエア・ブレーク機能を使用したのち［いいえ］を選択した場合，それまで設定していたすべてのソフトウエア・ブレーク・イベント，および Printf イベントは無効状態となります。この場合，このプロパティを［はい］に再設定しても自動的に有効状態には戻りません（手動で設定を行う必要があります）。

注意 2.　プログラム実行中は，このプロパティを変更することはできません。

(2)　［優先的に使用するブレークポイントの種類］
このプロパティは，［ソフトウエア・ブレークを使用する］プロパティにおいて［はい］を選択した場合のみ表示されます。
エディタ パネル／逆アセンブル パネルにおいて，マウスのワンクリック操作で設定するブレークポイントの種別を選択します。ブレークポイントの用途に合わせて，次のドロップダウン・リストから選択します。

ハードウエア・ブレーク	ハードウエア・ブレーク機能を使用した，ハードウエア・ブレークポイントを優先的に設定します（デフォルト）。 設定すると，ハードウエア・ブレーク・イベント（実行系）として扱われます。
ソフトウエア・ブレーク	ソフトウエア・ブレーク機能【Full-spec emulator】【E1】【E20】を使用した，ソフトウエア・ブレークポイントを優先的に設定します。 設定すると，ソフトウエア・ブレーク・イベントとして扱われます。

注意　指定した種類のブレークポイントの設定数が制限を越える場合（「2.16.6.1 有効イベント数の制限」参照），もう一方の種類のブレークポイントが使用されます。

(3)　［停止時に周辺エミュレーションを停止する］
ブレーク時に，エミュレータの周辺エミュレーション機能を停止（Peripheral Break）するか否かを選択します。
停止する場合は［はい］を選択してください（デフォルト：［いいえ］）。

2.9.2 プログラムの実行を手動で停止する

デバッグ・ツールバーの  ボタンをクリックすることにより，現在実行中のプログラムを強制的に停止します（強制ブレーク機能）。

2.9.3 任意の場所で停止する（ブレークポイント）

ブレークポイントは，マウスのワン・クリック操作で設定することができるブレーク・イベントの 1 つです。
ブレークポイントを設定することにより，任意の箇所でプログラムの実行を容易に停止させることができます。
ここでは，次の操作方法について説明します。

2.9.3.1 ブレークポイントを設定する

操作は，ソース・テキスト／逆アセンブル・テキストを表示しているエディタ パネル／逆アセンブル パネルで行います。
アドレス表示のあるメイン・エリア（エディタ パネル）／イベント・エリア（逆アセンブル パネル）において，ブレークポイントを設定したい箇所をクリックしてください。［優先的に使用するブレークポイントの種類］プロパティで選択している種別のブレークポイントが，クリックした行に対応する先頭アドレスの命令に設定されます。

ブレークポイントが設定されると，設定した箇所に次のイベント・マークが表示され，ソース・テキスト行／逆アセンブル・テキスト行が強調表示されます。
また，対象アドレスにブレーク・イベント（ハードウエア・ブレーク・イベント／ソフトウエア・ブレーク・イベント）が設定されたとみなされ，イベント パネルで管理されます（「2.16 イベントの管理」参照）。

表 2.3 ブレークポイントのイベント・マーク

ブレークポイント種別	イベント種別	イベント・マーク
ハードウエア・ブレークポイント	ハードウエア・ブレーク・イベント注	
ソフトウエア・ブレークポイント【Full-spec emulator】【E1】【E20】	ソフトウエア・ブレーク・イベント注	

注 イベント パネルにおける［名前］エリアでは，イベント種別名が“ブレーク”として表示されます。

図 2.65 ブレークポイントの設定例（逆アセンブル パネルの場合）

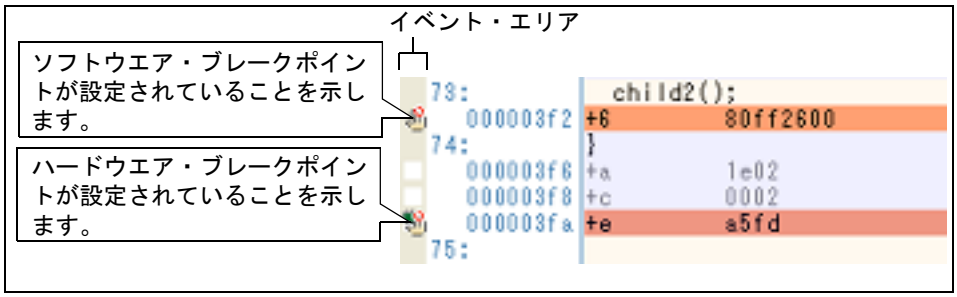
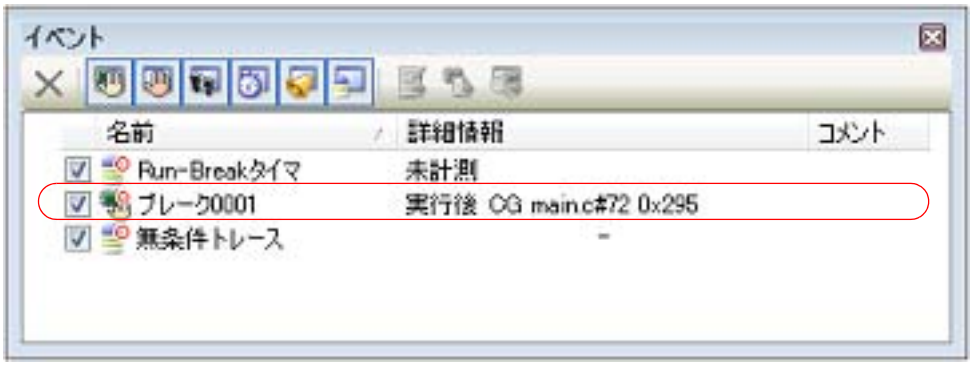


図 2.66 イベント パネルのブレークポイントの設定例



注意 1. ブレークポイントはブレーク・イベントとして設定され，イベントとして管理されるため，設定数に制限があります。ブレークポイントの設定に関しては（有効イベント数の制限など），「2.16.6 イベント設定に関する留意事項」も参照してください。

注意 2. ブレークポイントは，アドレス表示がない行に設定することはできません。

注意 3. 【Full-spec emulator】【E1】【E20】
ソフトウエア・ブレークポイントは，コード・フラッシュ領域にのみ設定することができます。

備考 1. イベントの設定状態によりイベント・マークは異なります（「2.16.1 設定状態（有効／無効）を変更する」参照）。
また，すでにイベントが設定されている箇所で，新たにイベントを設定した場合は，複数のイベントが設定されていることを示すイベント・マーク（  ）が表示されます。

備考 2. 【Full-spec emulator】【E1】【E20】
次に示す操作により，［優先的に使用するブレークポイントの種類］プロパティの選択に依存することなく，ハードウエア・ブレークポイント／ソフトウエア・ブレークポイントを設定することができます。
ただし，“操作方法 1”は，逆アセンブル パネルでのみ有効です。

種別	操作方法 1	操作方法 2
ハードウエア・ブレークポイント	［Ctrl］キー + クリック	コンテキスト・メニューの［ブレークの設定］→［ハード・ブレークを設定］を選択
ソフトウエア・ブレークポイント	［Shift］キー + クリック	コンテキスト・メニューの［ブレークの設定］→［ソフト・ブレークを設定］を選択

備考 3.　【シミュレータ】
設定できるブレークポイントは，ハードウエア・ブレークポイント固定です。

2.9.3.2 ブレークポイントを削除する

設定したブレークポイントを削除するには，エディタ パネル／逆アセンブル パネルで表示されているイベント・マークを再度クリックします（イベント・マークが消失します）。

2.9.4 任意の場所で停止する（ブレーク・イベント）

ブレーク・イベント（実行系）を設定することにより，任意の箇所でプログラムの実行を停止させることができます。
ここでは，次の操作方法について説明します。

2.9.4.1 ブレーク・イベント（実行系）を設定する

操作は，ソース・テキスト／逆アセンブル・テキストを表示しているエディタ パネル／逆アセンブル パネルで行います。
各パネルのアドレス表示のある行にキャレットを移動したのち，目的のイベント種別に従って，コンテキスト・メニューより次の操作を行います。

イベント種別	操作方法	説明
ハードウエア・ブレーク	［ブレークの設定］→［ハード・ブレークの設定］を選択	ハードウエア・ブレーク機能を使用してブレーク・イベントを設定します。
ソフトウエア・ブレーク【Full-spec emulator】【E1】【E20】	［ブレークの設定］→［ソフト・ブレークの設定］を選択	ソフトウエア・ブレーク機能【Full-spec emulator】【E1】【E20】を使用してブレーク・イベントを設定します。

ブレーク・イベント（実行系）は，キャレット位置の行に対応する先頭アドレスの命令に設定されます。
ブレーク・イベント（実行系）が設定されると，設定した箇所に次のイベント・マークが表示され，ソース・テキスト行／逆アセンブル・テキスト行が強調表示されます。
また，イベント パネルにおいて，ハードウエア・ブレーク・イベント（実行系）／ソフトウエア・ブレーク・イベント（実行系）として管理されます（「イベントの管理」参照）。

表 2.4　ブレーク・イベント（実行系）のイベント・マーク

イベント種別	イベント・マーク
ハードウエア・ブレーク	
ソフトウエア・ブレーク【Full-spec emulator】【E1】【E20】	

図 2.67 ブレーク・イベント（実行系）の設定例（逆アセンブル パネルの場合）

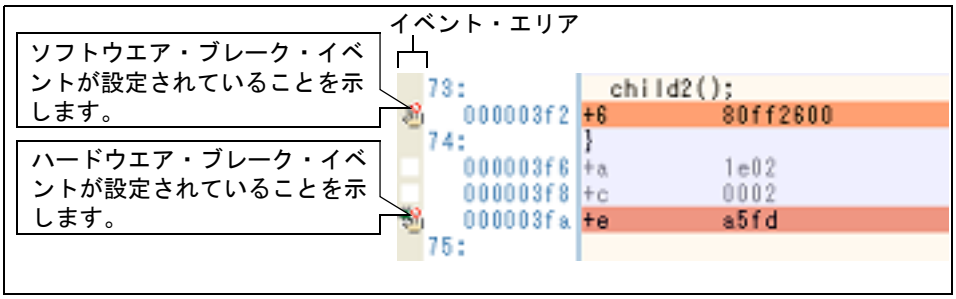
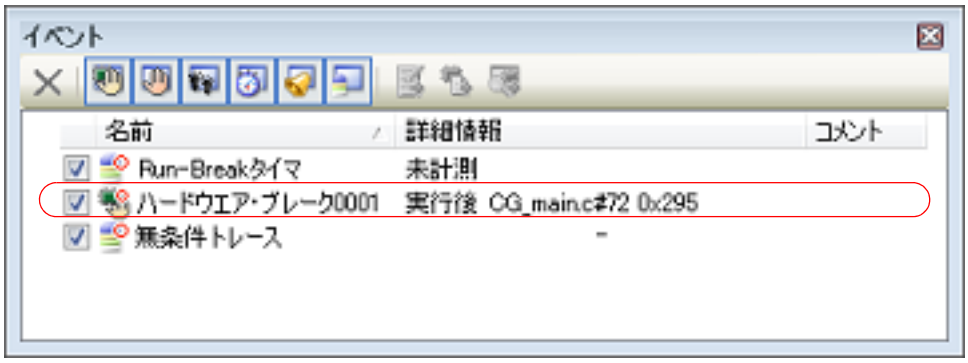


図 2.68 イベント パネルのハードウエア・ブレーク・イベント（実行系）の設定例



注意 1. ブレーク・イベント（実行系）の設定に関しては（有効イベント数の制限など），「イベント設定に関する留意事項」も参照してください。

注意 2. 【Full-spec emulator】【E1】【E20】
ソフトウエア・ブレーク・イベントは，コード・フラッシュ領域にのみ設定することができます。

備考 イベントの設定状態によりイベント・マークは異なります（「設定状態（有効／無効）を変更する」参照）。
また，すでにイベントが設定されている箇所で，新たにイベントを設定した場合は，複数のイベントが設定されていることを示すイベント・マーク（ ）が表示されます。

2.9.4.2 ブレーク・イベント（実行系）を削除する

設定したブレーク・イベント（実行系）を削除するには，エディタ パネル／逆アセンブル パネルにおいて，表示されているイベント・マークをクリックします。
また，イベント パネルにおいて，対象となるソフトウエア・ブレーク・イベント／ハードウエア・ブレーク・イベントを選択したのち，ツールバーの ボタンをクリックする操作でも削除することができます（「2.16.4 イベントを削除する」参照）。

2.9.5 変数 /I/O レジスタへのアクセスで停止する

ブレーク・イベント（アクセス系）を設定することにより，任意の変数，または I/O レジスタに対し，指定したアクセスがあった場合にプログラムの実行を停止させることができます。
また，この際に，アクセスした値を限定することもできます。
アクセス系のブレーク・イベントで指定できるアクセス種別は次のとおりです。

表 2.5 変数へのアクセス種別

アクセス種別	説明
リード	指定した変数 /I/O レジスタに，リード・アクセスした（読み込みを行った）際に実行中のプログラムを停止します。
ライト	指定した変数 /I/O レジスタに，ライト・アクセスした（書き込みを行った）際に実行中のプログラムを停止します。

アクセス種別	説明
リード／ライト	指定した変数 /I/O レジスタに，リード・アクセス／ライト・アクセスした（読み書きを行った）際に実行中のプログラムを停止します。

注意　　DMA（Direct Memory Access）によるアクセスは対象となりません。

ここでは，次の操作方法について説明します。

2.9.5.1 ブレーク・イベント（アクセス系）を設定する
2.9.5.2 ブレーク・イベント（アクセス系）を削除する

2.9.5.1 ブレーク・イベント（アクセス系）を設定する

変数，または I/O レジスタへのアクセスで，プログラムの実行を停止させるブレーク・イベントの設定は，次のいずれかの操作により行います。

注意　　ブレーク・イベントの設定に関しては（有効イベント数の制限など），「2.16.6 イベント設定に関する留意事項」も参照してください。

(1)　エディタ パネル／逆アセンブル パネル上の変数 /I/O レジスタにブレーク・イベント（アクセス系）を設定する場合
操作は，ソース・テキスト／逆アセンブル・テキストを表示しているエディタ パネル／逆アセンブル パネル上で行います。
ソース・テキスト／逆アセンブル・テキスト上の任意の変数，または I/O レジスタを選択したのち，目的のアクセス種別に従って，コンテキスト・メニューより次の操作を行います。
ただし，対象となる変数は，グローバル変数／関数内スタティック変数／ファイル内スタティック変数のみとなります。

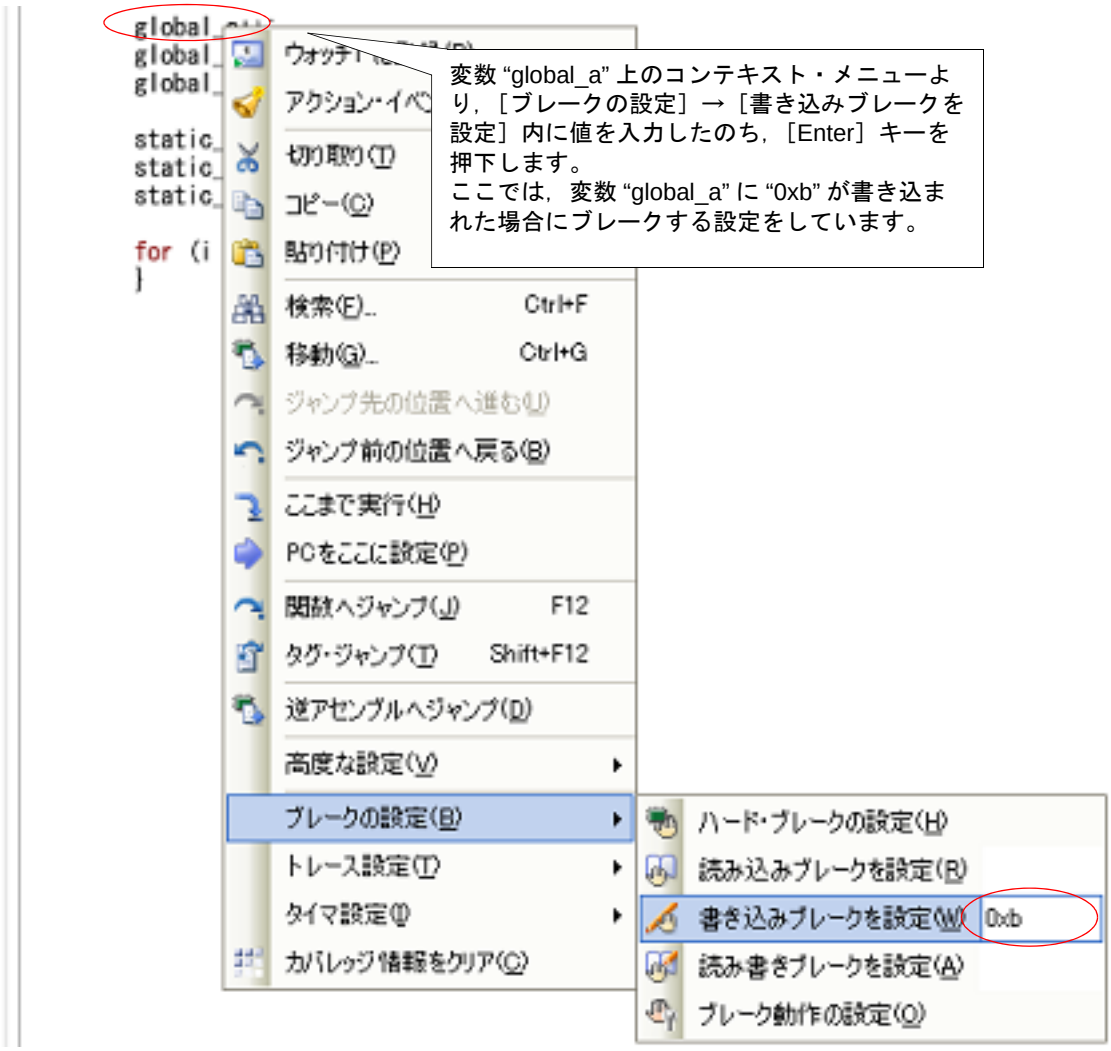
アクセス種別	操作方法
リード	［ブレークの設定］→［読み込みブレークを設定］を選択したのち，［Enter］キーを押下
ライト	［ブレークの設定］→［書き込みブレークを設定］を選択したのち，［Enter］キーを押下
リード／ライト	［ブレークの設定］→［読み書きブレークを設定］を選択したのち，［Enter］キーを押下

なお，この際に，コンテキスト・メニュー内のテキスト・ボックスに値を指定した場合，指定した値で読み込み／書き込みを行った場合のみブレークします。値を指定しない場合は，値にかかわらず，選択している変数に読み込み／書き込みを行った場合にブレークします。

注意 1.　　カレント・スコープ内の変数が対象となります。

注意 2.　　ブレーク・イベントは，アドレス表示がない行上の変数 /I/O レジスタを選択しても設定することはできません。

図 2.69 エディタ パネル上の変数に対するハードウエア・ブレーク・イベント（アクセス系）の設定例



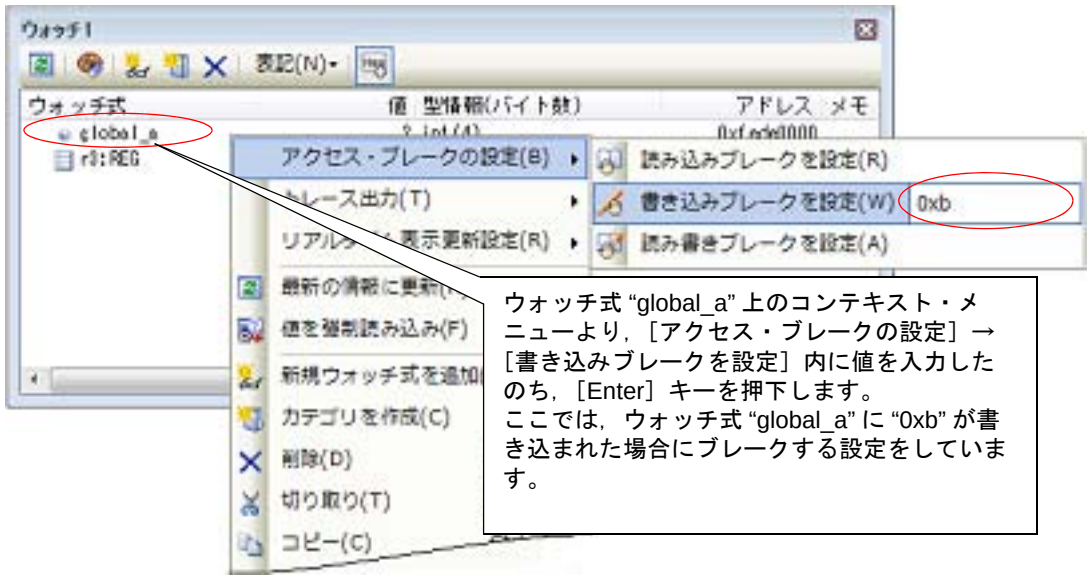
(2) 登録したウォッチ式にブレーク・イベント（アクセス系）を設定する場合
操作は，ウォッチ パネル上で行います。
対象となるウォッチ式を選択したのち（複数選択不可），目的のアクセス種別に従って，コンテキスト・メニューより次の操作を行います。
ただし，対象となるウォッチ式は，グローバル変数／関数内スタティック変数／ファイル内スタティック変数 / I/O レジスタのみとなります。

アクセス種別	操作方法
リード	［アクセス・ブレークの設定］→［読み込みブレークを設定］を選択したのち，［Enter］キーを押下
ライト	［アクセス・ブレークの設定］→［書き込みブレークを設定］を選択したのち，［Enter］キーを押下
リード／ライト	［アクセス・ブレークの設定］→［読み書きブレークを設定］を選択したのち，［Enter］キーを押下

なお，この際に，コンテキスト・メニュー内のテキスト・ボックスに値を指定した場合，指定した値で読み込み／書き込みを行った場合のみブレークします。値を指定しない場合は，値にかかわらず，選択しているウォッチ式に読み込み／書き込みを行った場合にブレークします。

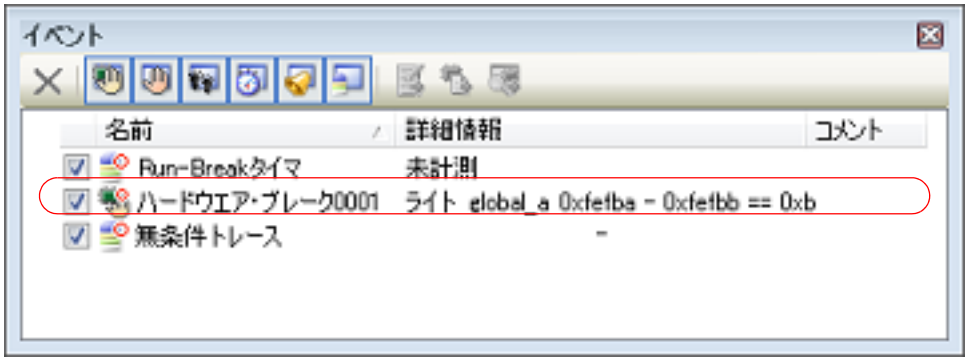
注意 カレント・スコープ内のウォッチ式が対象となります。
カレント・スコープ外のウォッチ式を対象とする場合は，スコープ指定したウォッチ式を選択してください。

図 2.70 ウォッチ式に対するブレーク・イベントの設定例



以上の操作を行うことにより，ブレーク・イベント（アクセス系）が設定されると，イベント パネルにおいて，ハードウエア・ブレーク・イベント（アクセス系）として管理されます（「2.16 イベントの管理」参照）。

図 2.71 イベント パネルのハードウエア・ブレーク・イベント（アクセス系）の設定例



2.9.5.2 ブレーク・イベント（アクセス系）を削除する

設定したブレーク・イベント（アクセス系）を削除する場合は，イベント パネルにおいて，対象となるハードウエア・ブレーク・イベントを選択したのち，ツールバーの ボタンをクリックします（「2.16 イベントの管理」参照）。

2.9.6 その他のブレーク要因

上記のほか，プログラムの実行が停止する原因（ブレーク要因）には次のものがあります。
なお，ブレーク要因は，プログラム停止時に，メイン・ウインドウのステータスバーのステータス・メッセージ，出力 パネル，またはトレース パネル【シミュレータ】で確認することができます。

表 2.6 その他のブレーク要因

要因	デバッグ・ツール		
	Full-spec emulator	E1/E20	シミュレータ
トレース・メモリを使い切った[注1]	○	○	○
ノン・マップ領域へのアクセス	−	−	○

要因	デバッグ・ツール		
	Full-spec emulator	E1/E20	シミュレータ
書き込み禁止領域への書き込み	−	−	○
テンポラリ・ブレーク[注2]の発生	○	○	○
ステップ実行回数オーバ	○	○	○
リレーブレーク[注3]の発生	○	○	○

注 1. プロパティ パネルの［デバッグ・ツール設定］タブ上の［トレース］カテゴリ内［トレース・メモリを使い切った後の動作］プロパティの設定に依存

注 2. CS+ 内部でのみ使用するブレーク（ユーザは使用不可）

注 3. マルチコア対応版における同期実行／同期ブレークのためのブレーク

2.10 メモリ，レジスタ，変数の表示／変更

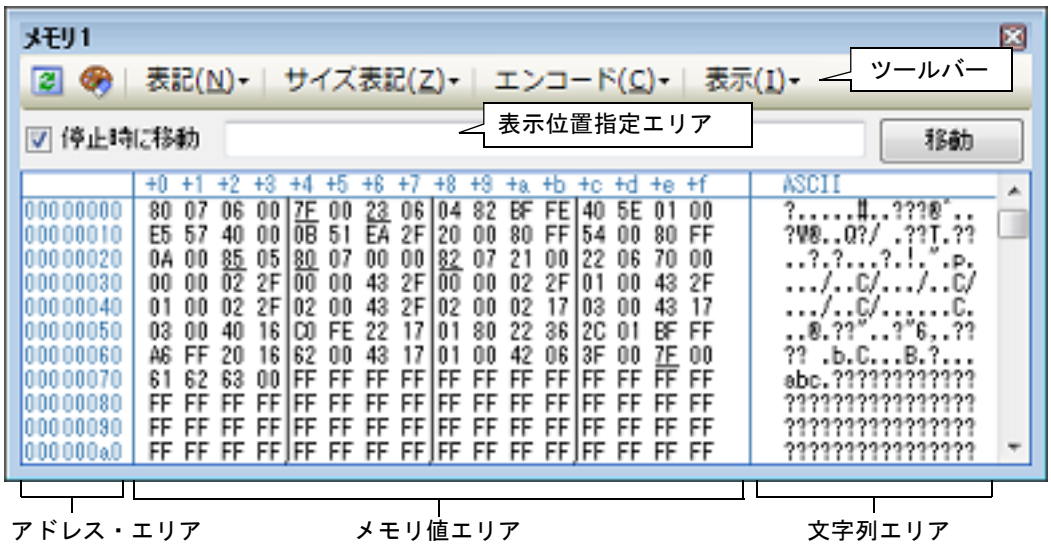
この節では，メモリ，レジスタ，および変数の内容を表示／変更する方法について説明します。

備考 マルチコア対応版を対象とした“メモリ／レジスタ／変数などの情報”については，「2.7 コア（PE）の選択」も参照してください。

2.10.1 メモリを表示／変更する

メモリの内容の表示，および値の変更は，次のメモリ パネルで行います。
［表示］メニュー→［メモリ］→［メモリ *1* ～ *4*］を選択してください。
メモリ パネルは，最大 4 個までオープンすることができ，各パネルはタイトルバーの“メモリ 1”，“メモリ 2”，“メモリ 3”，“メモリ 4”の名称で識別されます。
なお，各エリアの見方，および機能についての詳細は，メモリ パネルの項を参照してください。

図 2.72 メモリの内容の表示



備考 ツールバーの［表示］→ ボタンをクリックすることによりオープンするスクロール範囲設定 ダイアログにより，このパネルの垂直スクロール・バーのスクロール範囲（開始アドレス／終了アドレス）を設定することができます。

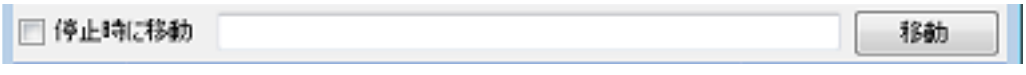
ここでは，次の操作方法について説明します。

2.10.1.1 表示位置を指定する

表示位置指定エリアにアドレス式を指定することにより，メモリ値の表示開始位置を指定することができます（デフォルトでは，0x0 番地より表示を開始します）。

備考 コンテキスト・メニューの［表示アドレス・オフセット値を設定 ...］を選択することでオープンするアドレス・オフセット設定 ダイアログにより，メモリ値の表示開始アドレスにオフセット値を設定することができます。

図 2.73 表示位置指定エリア（メモリ パネル）



(1) アドレス式の指定
表示したいメモリ値のアドレスとなるアドレス式をテキスト・ボックスに直接入力します。最大 1024 文字までの入力式を指定することができ，その計算結果を表示開始位置アドレスとして扱います。
ただし，マイクロコントローラのアドレス空間よりも大きいアドレス式を指定することはできません。

備考 1. このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

備考 2. 指定したアドレス式がシンボルを表現し，サイズが判明する場合では，そのシンボルの先頭アドレスから終了アドレスまでを選択状態で表示します。

(2) アドレス式の自動／手動評価の指定
表示開始位置を変更するタイミングは，［停止時に移動］チェック・ボックスの指定，および［移動］ボタンにより決定します。

［停止時に移動］		プログラム停止後，自動的にアドレス式の評価を行い，その計算結果のアドレスにキャレットが移動します。
		プログラム停止後，アドレス式の評価を自動的に行いません。 この場合，［移動］ボタンをクリックすることにより，アドレス式の評価を行います。
［移動］		［停止時に移動］チェック・ボックスのチェックをしなかった場合，このボタンをクリックすることによりアドレス式の評価を行い，その計算結果のアドレスにキャレットが移動します。

2.10.1.2 値の表示形式を変更する

ツールバーの次のボタンにより，このパネルのアドレス・エリア／メモリ値エリア／文字列エリアの表示形式を変更することができます。

表記	メモリ値の表示形式を変更する次のボタンを表示します。
 16 進数	メモリ値を 16 進数で表示します（デフォルト）。
 符号付き 10 進数	メモリ値を符号付き 10 進数で表示します。
 符号無し 10 進数	メモリ値を符号なし 10 進数で表示します。
 8 進数	メモリ値を 8 進数で表示します。
 2 進数	メモリ値を 2 進数で表示します。
サイズ表記	メモリ値のサイズの表示形式を変更する次のボタンを表示します。
 4 ビット	メモリ値を 4 ビット幅で表示します。
 1 バイト	メモリ値を 8 ビット幅で表示します（デフォルト）。
 2 バイト	メモリ値を 16 ビット幅で表示します。 対象メモリ領域のエンディアンに従って値を変換します。
 4 バイト	メモリ値を 32 ビット幅で表示します。 対象メモリ領域のエンディアンに従って値を変換します。
 8 バイト	メモリ値を 64 ビット幅で表示します。 対象メモリ領域のエンディアンに従って値を変換します。
エンコード	文字列のエンコードを変更する次のボタンを表示します。

	ASCII	文字列を ASCII コードで表示します（デフォルト）。
	Shift_JIS	文字列を Shift_JIS コードで表示します。
	EUC-JP	文字列を EUC-JP コードで表示します。
	UTF-8	文字列を UTF-8 コードで表示します。
	UTF-16	文字列を UTF-16 コードで表示します。
	Float	文字列を単精度浮動小数点数値[注]で表示します。
	Double	文字列を倍精度浮動小数点数値[注]で表示します。
	Float Complex	文字列を単精度浮動小数点数の複素数[注]で表示します。
	Double Complex	文字列を倍精度浮動小数点数の複素数[注]で表示します。
	Float Imaginary	文字列を単精度浮動小数点数の虚数[注]で表示します。
	Double Imaginary	文字列を倍精度浮動小数点数の虚数[注]で表示します。
表示		表示形式を変更する次のボタンを表示します。
	スクロール範囲を設定 ...	スクロール範囲を設定するためのスクロール範囲設定 ダイアログがオープンします。
	表示桁数を設定 ...	メモリ値エリアの表示桁数を設定するため，表示桁数設定 ダイアログをオープンします。
	表示アドレス・オフセット値を設定 ...	アドレス・エリアに表示するアドレスのオフセット値を設定するため，アドレス・オフセット設定 ダイアログをオープンします。

注　浮動小数点数値表示についての詳細は，メモリ パネルの項を参照してください。

2.10.1.3 メモリの内容を変更する

メモリの値は編集することができます。

メモリ値エリア／文字列エリアにおいて，対象メモリ値にキャレットを移動したのち，直接キーボードより編集します。メモリ値を編集すると変更箇所の表示色が変化し，この状態で［Enter］キーを押下することにより，変更した値がターゲット・メモリに書き込まれます（［Enter］キーの押下前に［Esc］キーを押下すると編集をキャンセルします）。

ただし，変更の際に入力可能な文字列は，現在指定されている表示進数で扱うことができる文字列に限ります。また，文字列エリアでの変更は，文字コードとして “ASCII” が指定されている場合のみ可能です。

なお，メモリの値の編集は，プログラム実行中の状態でも行うことができます。設定方法についての詳細は，「2.10.1.4 プログラム実行中にメモリの内容を表示／変更する」を参照してください。

値を変更する際において，留意する必要がある例を次に示します。

例 1.　表示ビット幅の最大値を越えた場合
10 進数 8 ビット表示において，表示値 “105” の “1” を編集して “3” を入力した場合，変更値は最大値である “127” となります。

例 2.　数値の途中に “-” を入力した場合
符号あり 10 進数 16 ビット表示において，表示値 “32768” を “32-68” と編集した場合，“3” と “2” が空白に変わり，変更値は “-68” となります。

例 3.　数値の途中に空白記号（スペース）を入力した場合
10 進数 16 ビット表示において，表示値 “32767” を “32 67” と編集した場合，“3” と “2” が空白に変わり，変更値は “67” となります。

例 4.　同一の値を入力した場合
現在のメモリ値と同一の値を指定した場合でも，指定した値をメモリに書き込みます。

2.10.1.4 プログラム実行中にメモリの内容を表示／変更する

メモリ パネル／ウォッチ パネルでは，プログラムの実行中に，リアルタイムにメモリ／ウォッチ式の内容を表示更新，および書き換えることができるリアルタイム表示更新機能を備えています。

このリアルタイム表示更新機能を有効化することにより，プログラムが停止している状態だけでなく，実行中の状態であっても，メモリ／ウォッチ式の値の表示／変更を行うことができます。
なお，リアルタイム表示更新機能は，CPU/ デバッグ・ツールが持つ RRM 機能（読み込み）【シミュレータ】，RAM モニタ機能（読み込み）【Full-spec emulator】【E1】【E20】，および DMM 機能（書き込み）により実現され，それぞれの機能による読み込み／書き込みが可能な対象領域は異なります。
リアルタイム表示更新機能を有効にするために，プロパティ パネルの［デバッグ・ツール設定］タブ上において，次の基本設定を行ってください。

表 2.7　　リアルタイム表示更新機能の基本設定

カテゴリ	プロパティ	設定値
［実行中のメモリ・アクセス］	［実行中に表示更新を行う］	［はい］（デフォルト）
	［表示更新間隔 [ms]］	［*100 ～ 65500 の整数*］

注意 1.　ローカル変数は，リアルタイム表示更新機能の対象外です。

注意 2.　RRM 機能や RAM モニタ機能で値を読み出す変数のサイズが複数バイト（2 バイト /4 バイト /8 バイト）の場合，変数へ値を代入する処理が 2 回に分けて行われる場合があります。
この 2 回の代入処理の間で変数の読み出しが行われると，変数へ値が代入される途中の値が読み出され，実際には代入していない値が表示されることがあるため注意が必要です。

注意 3.　選択しているマイクロコントローラがマルチコア対応版の場合では，全 PE のアクセスを対象に読み込み可能です。
ただし，Local RAM self 領域は，現在選択している PE のみが対象となります。

備考　メモリ パネル／ウォッチ パネルにおける値の書き換え方法についての詳細は，「2.10.1.3 メモリの内容を変更する」／「2.10.6.6 ウォッチ式の内容を変更する」を参照してください。

(1)　RRM 機能（読み込み）【シミュレータ】
プログラム実行中に，リアルタイムにメモリ／ウォッチ式の内容を読み込む機能です。
この領域に割り当てられているメモリ／ウォッチ式は，常にリアルタイムな表示が可能です。
RRM 機能による読み込みが可能な領域は次のとおりです。

表 2.8　　RRM 機能の対象領域

対象領域	シミュレータ
内蔵 ROM	○
内蔵 RAM	○
周辺 I/O 領域	－
データフラッシュ	－
エミュレーション・メモリ	－
ターゲット・メモリ	－
CPU レジスタ	○注
I/O レジスタ （読み込み保護対象 IOR を除く）	○

注　　トレーサ／タイマ動作中は不可

(2)　RAM モニタ機能（読み込み）【Full-spec emulator】【E1】【E20】
CPU の RAM モニタ機能を使用してメモリ／ウォッチ式の内容を読み込む機能です。
RAM モニタ機能による読み込みが可能な領域は次のとおりです。

注意　CPU ステータスがスタンバイ・モード（HALT/STOP/IDLE）に移行すると，タイムアウト・エラーを発生します。

表 2.9　　RAM モニタ機能の対象領域

対象領域	Full-spec emulator	E1/E20
内部 ROM	－	－
内部 RAM	○	○
周辺 I/O 領域	－	－
データフラッシュ	－	－
ターゲット・メモリ	－	－
CPU レジスタ	－	－
I/O レジスタ	－	－

ただし，RAM モニタ機能を有効にするためには，リアルタイム表示更新機能の基本設定に加え，次の設定が必要となります。

デバッグ・ツール	カテゴリ	プロパティ	設定値
Full-spec emulator	［実行中のメモリ・アクセス］	［実行中にアクセスする］	［はい］
E1/E20			

(3)　DMM 機能（書き込み）

プログラム実行中に，リアルタイムにメモリ／ウォッチ式に値を書き込む機能です。
DMM 機能による書き込みが可能な領域は次のとおりです。

注意 1.　　DMM 機能を使用して書き込みを行った場合のアトミック性の保証はありません。

注意 2.　　CPU ステータスがスタンバイ・モード（HALT/STOP/IDLE）に移行すると，タイムアウト・エラーを発生します。

表 2.10　　DMM 機能の対象領域

対象領域	Full-spec emulator	E1/E20	シミュレータ
内部 ROM	－	－	○
内部 RAM	○	○	○
周辺 I/O 領域	－	－	－
エミュレーション・メモリ	－	－	－
ターゲット・メモリ	－	－	－
CPU レジスタ	－	－	○注
I/O レジスタ （読み込み保護対象 IOR を除く）	－	－	○

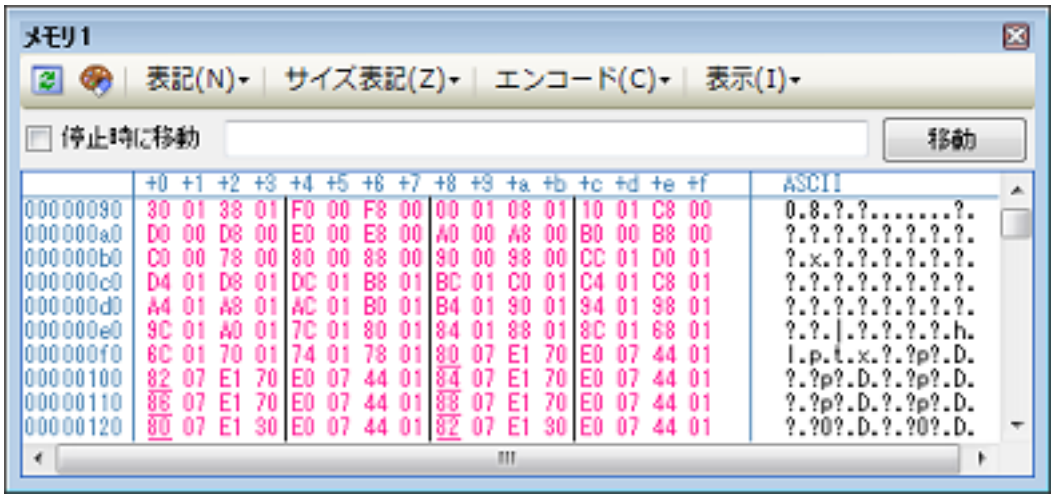
注　　トレーサ／タイマ動作中は不可

ただし，DMM 機能を有効にするためには，リアルタイム表示更新機能の基本設定に加え，次の設定が必要となります。

デバッグ・ツール	カテゴリ	プロパティ	設定値
Full-spec emulator E1/E20	［実行中のメモリ・アクセス］	［実行中にアクセスする］	［はい］
シミュレータ	設定不要		

なお，リアルタイム表示更新機能を行っているメモリ値／ウォッチ式は，メモリ パネル／ウォッチ パネルにおいてピンク色に強調表示されます。

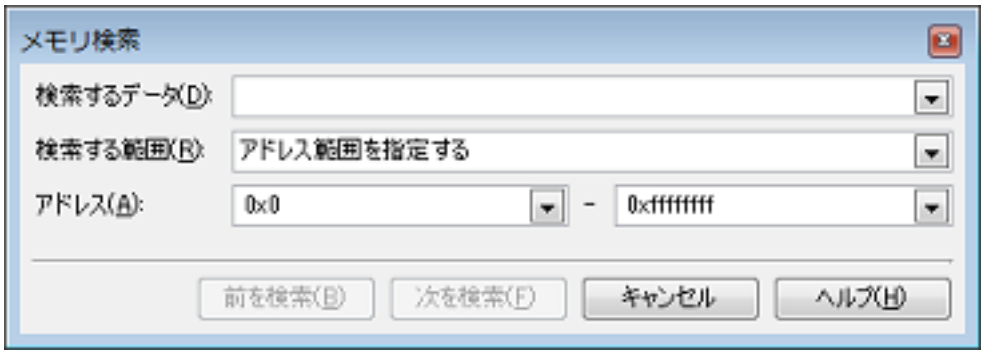
図 2.74 リアルタイム表示更新を行っているメモリ表示の例（メモリ パネル）



2.10.1.5 メモリの内容を検索する

メモリの値の検索は，コンテキスト・メニューの［検索 ...］を選択することによりオープンするメモリ検索 ダイアログで行います。検索の際は，メモリ値エリアと文字列エリアのうち，キャレットのあるエリアが対象となります。
このダイアログにおいて，次の手順で操作を行ってください。

図 2.75 メモリ内容の検索（メモリ検索 ダイアログ）



注意 1. プログラム実行中に，メモリの内容を検索することはできません。

注意 2. 浮動小数点数値表示している文字列を検索することはできません。

(1) ［検索するデータ］の指定
検索するデータを指定します。
テキスト・ボックスに直接入力するか（最大指定バイト数：256 バイト），またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。検索の対象がメモリ値エリアの場合，そのエリアと同じ表示形式（表示進数／サイズ）でデータを入力する必要があります。
また，検索の対象が文字列エリアの場合では，検索するデータとして文字列を指定する必要があります。指定した文字列は，そのエリアで表示しているエンコード形式でデータに変換され検索されます。
なお，このダイアログをオープンする直前にメモリ値を選択していた場合は，デフォルトでその値が表示されます。

(2) ［検索する範囲］の指定
検索する範囲を次のドロップダウン・リストより選択します。

アドレス範囲を指定する	［アドレス］で指定するアドレス範囲内で検索を行います。
メモリ・マッピング	選択したメモリ・マッピング範囲内で検索を行います。 このリスト項目は，メモリ・マッピング ダイアログで表示しているメモリ・マッピングを個々に表示します。 表示形式：＜*メモリ種別*＞ ＜*アドレス範囲*＞ ＜*サイズ*＞

(3) ［アドレス］の指定
この項目は，「(2) ［検索する範囲］の指定」で［アドレス範囲を指定する］を選択した場合のみ有効となります。
メモリ値検索の対象となるアドレス範囲を“開始アドレス - 終了アドレス”で指定します。それぞれのテキスト・ボックスにアドレス式を直接入力するか（最大指定文字数：1024 文字），またはドロップダウン・リストにより入力履歴項目（最大履歴個数：10 個）を選択することにより行います。入力したアドレス式の計算結果を，それぞれ開始アドレス／終了アドレスとして扱います。
ただし，検索可能なアドレスの上限値は，プログラム空間の上限アドレス（0x03FFFFFF）です（ミラー領域は検索対象となりません）。
また，32 ビットで表現できる値より大きいアドレス値を指定することはできません。

備考 1. このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

備考 2. “開始アドレス”が空欄の場合は，“0x0”の指定として扱われます。

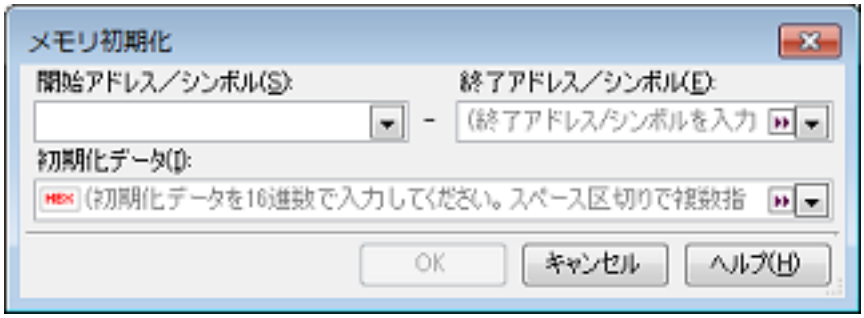
備考 3. “終了アドレス”が空欄の場合は，マイクロコントローラのアドレス空間の上限値の指定として扱われます。

(4) ［前を検索］／［次を検索］ボタンのクリック
［前を検索］ボタンをクリックすると，指定した範囲内でアドレスの小さい方向に検索を行い，検索結果箇所をメモリ パネル上で選択状態にします。
［次を検索］ボタンをクリックすると，指定した範囲内でアドレスの大きい方向に検索を行い，検索結果箇所をメモリ パネル上で選択状態にします。

2.10.1.6 メモリの内容を一括して変更（初期化）する

メモリの値を一括して変更（初期化）することができます。
コンテキスト・メニューの［初期化 ...］を選択することにより，指定したアドレス範囲のメモリ値を一括して変更するためのメモリ初期化 ダイアログがオープンします。
このダイアログにおいて，次の手順で操作を行ってください。

図 2.76 メモリ内容の一括変更（メモリ初期化 ダイアログ）



(1) ［開始アドレス／シンボル］と［終了アドレス／シンボル］の指定
メモリの内容を初期化するアドレス範囲を［開始アドレス／シンボル］と［終了アドレス／シンボル］に指定します。それぞれのテキスト・ボックスにアドレス式を直接入力するか（最大指定文字数：1024 文字），またはドロップダウン・リストにより入力履歴項目（最大履歴個数：10 個）を選択します。
入力したアドレス式の計算結果を，それぞれ開始アドレス／終了アドレスとして扱います。
なお，マイクロコントローラのアドレス空間よりも大きいアドレス値を指定することはできません。

注意 エンディアンの異なる領域をまたいだアドレス範囲を指定することはできません。

備考 このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

(2) ［初期化データ］の指定
メモリに書き込む初期化データを指定します。

16 進数の数値をテキスト・ボックスに直接入力するか，またはドロップダウン・リストにより入力履歴項目（最大履歴個数：10 個）を選択します。初期化データを複数指定する場合は，1 個 4 バイト（8 文字）までのデータを最大 16 個まで，半角スペースで区切り指定します。
個々の初期化データは，文字列終端より 2 文字単位で 1 バイトと解釈され，奇数文字数の場合は先頭 1 文字で 1 バイトと解釈されます。
なお，バイト数が 2 バイト以上の場合は，初期化対象のアドレス範囲のエンディアンのバイト列に変換してターゲット・メモリへの書き込み処理を行います。

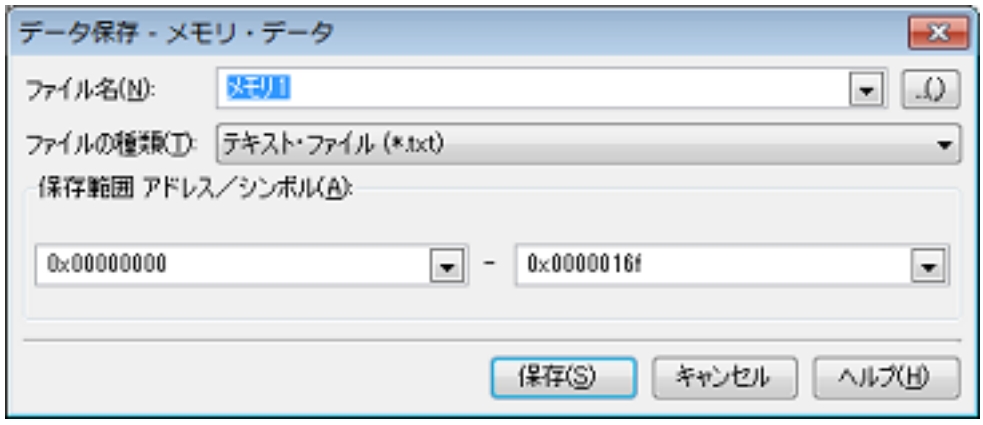
入力文字列（初期化データ）	書き込みイメージ（バイト単位）	
	リトル・エンディアン	ビッグ・エンディアン
1	01	01
0 12	00 12	00 12
00 012 345	00 12 00 45 03	00 00 12 03 45
000 12 000345	00 00 12 45 03 00	00 00 12 00 03 45

(3) ［OK］ボタンのクリック
［OK］ボタンをクリックします。
指定したアドレス範囲のメモリ領域に，指定した初期化データのパターンを繰り返し書き込みます（パターンの途中で終了アドレスに達した場合は書き込みを終了します）。
ただし，不正な値やアドレス式を指定している場合，メッセージを表示し，メモリ値の初期化は行いません。

2.10.1.7 メモリの表示内容を保存する

メモリの内容を範囲指定して，テキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
ファイルに保存する際は，デバッグ・ツールから最新の情報を取得し，このパネル上での表示形式に従ったデータで保存します。
［ファイル］メニュー→［名前を付けてメモリ・データを保存 ...］を選択すると，次のデータ保存 ダイアログがオープンします（この際，パネル上で範囲選択した状態でこの操作を行うと選択範囲のみのメモリ・データを保存することができます）。
このダイアログにおいて，次の手順で操作を行ってください。

図 2.77 メモリ・データの保存（データ保存 ダイアログ）



(1) ［ファイル名］の指定
保存するファイル名を指定します。
テキスト・ボックスに直接入力するか（最大指定文字数：259 文字），またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
また，［...］ボタンをクリックすることでオープンするデータ保存ファイルを選択 ダイアログにより，ファイルを選択することもできます。

(2) ［ファイルの種類］の指定
保存するファイルの形式を次のドロップダウン・リストにより選択します。
選択できるファイルの形式は次のとおりです。

リスト表示	形式
テキスト・ファイル (*.txt)	テキスト形式（デフォルト）
CSV(カンマ区切り) (*.csv)	CSV 形式[注]

注　　各データを “,” で区切り保存します。
なお，データ内に “,” が含まれている際の不正形式を避けるため，各データを ""（ダブルクォーテーション）で括り出力します。

(3)　［保存範囲 アドレス／シンボル］の指定
ファイルに保存する範囲を “ 開始アドレス ” と “ 終了アドレス ” で指定します。
それぞれのテキスト・ボックスに 16 進数の数値／アドレス式を直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
なお，パネル上で範囲選択している場合は，デフォルトでその選択範囲がテキスト・ボックスに指定されます。
範囲選択していない場合は，現在のパネルの表示範囲が指定されます。

備考　　このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

(4)　［保存］ボタンのクリック
指定したファイルに，指定した形式でメモリ・データを保存します。

図 2.78　　メモリ・データ保存の際の出力イメージ

【テキスト・ファイル（*.txt）で保存（16 進表記 /8 ビット幅 /ASCII コードの場合の例）】

```
           +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +a +b +c +d +e +f
00000000 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |
00000010 | 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 |
```

【CSV ファイル（*csv）で保存（16 進表記 /8 ビット幅 /ASCII コードの場合の例）】

```
00000000,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,
00000010,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
```

備考　　［ファイル］メニュー→［メモリ・データを保存］の選択によりパネルの内容を上書き保存する場合，メモリ パネル（メモリ 1 ～ 4）はそれぞれ個別に扱われます。
また，保存範囲についても，前回指定したアドレス範囲で保存されます。

2.10.2 CPU レジスタを表示／変更する

CPU レジスタ（プログラム・レジスタ／システム・レジスタ）の内容の表示，および値の変更は，次の CPU レジスタ パネルで行います。
［表示］メニュー→［CPU レジスタ］を選択してください。
なお，各エリアの見方，および機能についての詳細は，CPU レジスタ パネルの項を参照してください。

図 2.79 CPU レジスタの内容の表示（CPU レジスタ パネル）

ここでは，次の操作方法について説明します。

2.10.2.1 値の表示形式を変更する
2.10.2.2 CPU レジスタの内容を変更する
2.10.2.3 プログラム実行中に CPU レジスタの内容を表示／変更する
2.10.2.4 CPU レジスタの表示内容を保存する

2.10.2.1 値の表示形式を変更する

ツールバーの次のボタンにより，このパネルの［値］エリアの表示形式を変更することができます。

表記	値の表示形式を変更する次のボタンを表示します。
	選択している項目（下位項目を含む）の値を規定値で表示します（デフォルト）。
	選択している項目（下位項目を含む）の値を 16 進数で表示します。
	選択している項目（下位項目を含む）の値を符号付き 10 進数で表示します。
	選択している項目（下位項目を含む）の値を符号なし 10 進数で表示します。
	選択している項目（下位項目を含む）の値を 8 進数で表示します。
	選択している項目（下位項目を含む）の値を 2 進数で表示します。
	選択している項目（下位項目を含む）の文字列を ASCII コードで表示します。 対象が 2 バイト以上ある場合は，1 バイトずつの文字を並べて表示します。
	選択している項目を Float で表示します。 ただし，4 バイト・データ以外の場合は，規定値で表示します。
	選択している項目を Double で表示します。 ただし，8 バイト・データ以外の場合は，規定値で表示します。
	値表示の末尾に，その値の 16 進数表記を “（）” で囲んで併記します。

2.10.2.2 CPU レジスタの内容を変更する

CPU レジスタの値は，編集することができます。

［値］エリアにおいて，対象 CPU レジスタ値を選択したのち再度クリックすると，値が編集モードになります（［Esc］キーの押下で編集モードをキャンセルします）。
値をキーボードより直接編集したのち，［Enter］キーを押下することにより，変更した値がデバッグ・ツールのターゲット・メモリに書き込まれます。

注意　　この操作は，プログラム実行中に行うことはできません。

2.10.2.3 プログラム実行中に CPU レジスタの内容を表示／変更する

対象となる CPU レジスタをウォッチ式としてウォッチ パネルに登録することにより，プログラムが停止状態だけでなく，実行状態であっても CPU レジスタの値をリアルタイムに表示／変更することができます。
ウォッチ式についての詳細は，「2.10.6 ウォッチ式を表示／変更する」を参照してください。

2.10.2.4 CPU レジスタの表示内容を保存する

［ファイル］メニュー→［名前を付けて CPU レジスタ・データを保存 ...］を選択することにより，名前を付けて保存ダイアログをオープンし，CPU レジスタのすべての内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
ファイルに保存する際は，デバッグ・ツールから最新の情報を取得します。

図 2.80　　CPU レジスタ保存の際の出力イメージ

```
レジスタ名                値
-------------------------------------------------------------------------
カテゴリ名
 - レジスタ名             値
      :                   :
```

2.10.3 I/O レジスタを表示／変更する

I/O レジスタの内容の表示，および値の変更は，次の IOR パネルで行います。
［表示］メニュー→［IOR］を選択してください。
なお，各エリアの見方，および機能についての詳細は，IOR パネルの項を参照してください。

図 2.81　　I/O レジスタの内容の表示（IOR パネル）

ここでは，次の操作方法について説明します。

- 2.10.3.1 I/O レジスタを検索する
- 2.10.3.2 I/O レジスタを整理する
- 2.10.3.3 値の表示形式を変更する

2.10.3.1 I/O レジスタを検索する

I/O レジスタ名を検索することができます。
検索エリアにおいて，テキスト・ボックスに検索する I/O レジスタ名を指定します（大文字／小文字不問）。キーボードより文字列を直接入力するか（最大指定文字数：512 文字），ドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
次のいずれかのボタンをクリックします。

	テキスト・ボックスで指定している文字列を含む I/O レジス名を上方向に検索し，検索結果を選択状態にします。
	テキスト・ボックスで指定している文字列を含む I/O レジス名を下方向に検索し，検索結果を選択状態にします。

備考 1.　カテゴリ（フォルダ）により分類されて非表示の状態の I/O レジスタ名も検索します（展開して選択状態となります）。

備考 2.　検索対象の文字列入力後，［Enter］キーを押下することにより，  ボタンのクリックと同等の動作を行い，［Shift］＋［Enter］キーを押下することにより，  ボタンのクリックと同等の動作を行います。

2.10.3.2 I/O レジスタを整理する

各 I/O レジスタを任意のカテゴリ（フォルダ）で分類し，ツリー形式を編集することができます。

注意 1.　カテゴリ内にカテゴリを作成することはできません。

注意 2.　I/O レジスタの追加／削除はできません。

(1)　カテゴリを新規作成する場合
作成したい I/O レジスタ名にキャレットを移動したのち，ツールバーの  ボタンのクリックし，キーボードより新規カテゴリ名を直接入力します。

(2)　カテゴリ名を編集する場合
編集したいカテゴリ名を選択したのち，再度クリックし，キーボードよりカテゴリ名を直接編集します。

(3)　カテゴリを削除する場合
削除したいカテゴリを選択したのち，ツールバーの  ボタンをクリックします。
ただし，削除できるカテゴリは，空のカテゴリのみです。

(4)　表示順を変更する場合
I/O レジスタ名をカテゴリ内に直接ドラッグ・アンド・ドロップすることにより，I/O レジスタはカテゴリで分類されます。
また，カテゴリと I/O レジスタ名の表示の順番（上下位置）も，ドラッグ・アンド・ドロップ操作により自由に変更することができます。

2.10.3.3 値の表示形式を変更する

ツールバーの次のボタンにより，このパネルの［値］エリアの表示形式を変更することができます。

表記	値の表示形式を変更する次のボタンを表示します。
	選択している項目の値を 16 進数で表示します（デフォルト）。
	選択している項目の値を符号付き 10 進数で表示します。
	選択している項目の値を符号なし 10 進数で表示します。
	選択している項目の値を 8 進数で表示します。
	選択している項目の値を 2 進数で表示します。
	選択している項目の値を ASCII コードで表示します。

	選択している項目の値表示の末尾に，その値の 16 進数表記を “（）” で囲んで併記します。

2.10.3.4 I/O レジスタの内容を変更する

I/O レジスタの値は，編集することができます。
［値］エリアにおいて，対象 I/O レジスタ値を選択したのち再度クリックすると，値が編集モードになります（［Esc］キーの押下で編集モードをキャンセルします）。
値をキーボードより直接編集したのち，［Enter］キーを押下することにより，変更した値がデバッグ・ツールのターゲット・メモリに書き込まれます。

注意 1. この操作は，プログラム実行中に行うことはできません。

注意 2. 読み込み専用の I/O レジスタの値を変更することはできません。

備考 1. I/O レジスタのサイズより小さい桁の数値が入力された場合，上位の桁を 0 でパディングします。

備考 2. I/O レジスタのサイズより大きい桁の数値が入力された場合，上位の桁をマスクします。

備考 3. I/O レジスタの値には ASCII 文字による入力も可能です。

- I/O レジスタ名 “OSTM*nXX*” の値に “0x41” を書き込んだ場合
  → OSTM*nXX* に，“0x41” が書き込まれます。
- I/O レジスタ名 “OSTM*nXX*” の値に ASCII 文字 “'A'” を書き込んだ場合
  → OSTM*nXX* に，“0x41” が書き込まれます。

2.10.3.5 プログラム実行中に I/O レジスタの内容を表示／変更する

対象となる I/O レジスタをウォッチ式としてウォッチ パネルに登録することにより，プログラムが停止状態だけでなく，実行状態であっても I/O レジスタの値をリアルタイムに表示／変更することができます。
ウォッチ式についての詳細は，「2.10.6 ウォッチ式を表示／変更する」を参照してください。

2.10.3.6 I/O レジスタの表示内容を保存する

［ファイル］メニュー→［名前を付けて IOR データを保存 ...］を選択することにより，名前を付けて保存 ダイアログをオープンし，I/O レジスタのすべての内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます（このパネル上での表示／非表示の設定に関わらず，すべての I/O レジスタの値が対象となります）。
ファイルに保存する際は，I/O レジスタの値を再読み込みし，取得した最新の値を保存します。
ただし，読み込み保護対象の I/O レジスタの再読み込みは行いません。最新の内容を保存したい場合は，コンテキスト・メニューの［値を強制読み込み］を選択したのち，ファイルの保存を行ってください。

図 2.82 I/O レジスタ保存の際の出力イメージ

```
IOR 名                  値          型情報（バイト数）      アドレス
-------------------------------------------------------------------
カテゴリ名
 -IOR 名                値          型情報（バイト数）      アドレス
    :                    :                :                  :
```

2.10.4 グローバル変数／スタティック変数を表示／変更する

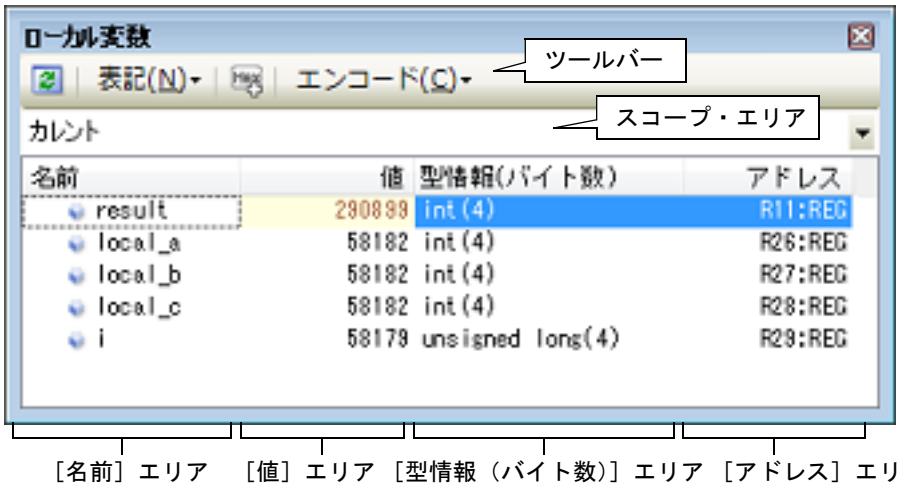
グローバル変数，またはスタティック変数の値の表示／変更は，ウォッチ パネルで行います。
値の表示／変更を行いたい変数をウォッチ式としてウォッチ パネルに登録してください。
ウォッチ式についての詳細は，「2.10.6 ウォッチ式を表示／変更する」を参照してください。

2.10.5 ローカル変数を表示／変更する

ローカル変数の内容の表示，および値の変更は，次のローカル変数 パネルで行います。
［表示］メニュー→［ローカル変数］を選択してください。
目的のローカル変数の内容を表示するためには，スコープ・エリアでスコープの選択をします。

ローカル変数 パネルでは，ローカル変数名や関数名を表示します。また，関数の引数もローカル変数として表示します。
なお，各エリアの見方，および機能についての詳細は，ローカル変数 パネルの項を参照してください。

図 2.83 ローカル変数の内容の表示（ローカル変数 パネル）



注意 プログラム実行中は，このパネルには何も表示されません。
プログラムの実行が停止したタイミングで，各エリアの表示を行います。

ここでは，次の操作方法について説明します。

2.10.5.1 値の表示形式を変更する
2.10.5.2 ローカル変数の内容を変更する
2.10.5.3 ローカル変数の表示内容を保存する

2.10.5.1 値の表示形式を変更する

ツールバーの次のボタンにより，このパネルの［値］エリアの表示形式を変更することができます。

表記		値の表示形式を変更する次のボタンを表示します。
	自動	このパネル上の値の表記を変数ごとの規定値で表示します（デフォルト）。
	16 進数	このパネル上の値を 16 進数で表示します。
	10 進数	このパネル上の値を 10 進数で表示します。
	8 進数	このパネル上の値を 8 進数で表示します。
	2 進数	このパネル上の値を 2 進数で表示します。
	配列のインデックスを 10 進数表記	このパネル上の配列のインデックスを 10 進数で表示します（デフォルト）。
	配列のインデックスを 16 進数表記	このパネル上の配列のインデックスを 16 進数で表示します。
	Float	このパネル上の値を Float で表示します。 ただし，4 バイト・データ以外，または型情報を持つ場合は，規定値で表示します。
	Double	このパネル上の値を Double で表示します。 ただし，8 バイト・データ以外，または型情報を持つ場合は，規定値で表示します。
		値表示の末尾に，その値の 16 進数表記を “（）” で囲んで併記します。

エンコード	文字列変数のエンコードを変更する次のボタンを表示します。
ASCII	文字列変数を ASCII コードで表示します（デフォルト）。
Shift_JIS	文字列変数を Shift_JIS コードで表示します。
EUC-JP	文字列変数を EUC-JP コードで表示します。
UTF-8	文字列変数を UTF-8 コードで表示します。
UTF-16	文字列変数を UTF-16 コードで表示します。

2.10.5.2 ローカル変数の内容を変更する

ローカル変数の値，および引数の値は，編集することができます。
［値］エリアにおいて，対象ローカル変数値／引数値を選択したのち再度クリックすると，値が編集モードになります（［Esc］キーの押下で編集モードをキャンセルします）。
値をキーボードより直接編集したのち，［Enter］キーを押下することにより，変更した値がデバッグ・ツールのターゲット・メモリに書き込まれます。この際に，値のチェックを行い，型に不適合な場合は編集を無効とします。

注意　この操作は，プログラム実行中に行うことはできません。

備考 1.　変数のサイズより小さい桁の数値が入力された場合，上位の桁を 0 でパディングします。

備考 2.　変数のサイズより大きい桁の数値が入力された場合，上位の桁をマスクします。

備考 3.　文字配列（char 型，unsigned char 型）に対しては，表示形式に ASCII が選択されている場合，文字列（ASCII/Shift_JIS/EUC-JP/Unicode（UTF-8/UTF-16））による値の入力も可能です。

備考 4.　ローカル変数の値には，次のように ASCII 文字による入力も可能です。

- ASCII 文字による入力の場合
  変数 “ch” の［値］エリアに “'A'” を入力
  → “ch” が割り当てられているメモリ領域に “0x41” を書き込む
- 数値による入力の場合
  変数 “ch” の［値］エリアに “0x41” を入力
  → “ch” が割り当てられているメモリ領域に “0x41” を書き込む
- 文字列（ASCII）による入力の場合
  文字配列 “str” の表示形式を ASCII に設定し，［値］エリアに “"ABC"” を入力
  → “str” が割り当てられているメモリ領域に “0x41，0x42，0x43，0x00” を書き込む

2.10.5.3 ローカル変数の表示内容を保存する

［ファイル］メニュー→［名前を付けてローカル変数データを保存 ...］を選択することにより，名前を付けて保存 ダイアログをオープンし，ローカル変数のすべての内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。

ファイルに保存する際は，デバッグ・ツールから最新の情報を取得します。

なお，配列，ポインタ型変数，構造体／共用体，CPU レジスタ（部分を表す名前が付与されているもののみ）を展開表示している場合では，各展開要素の値も保存されます。展開表示していない場合は，先頭に “+” マークが付与され，値は空欄となります。

図 2.84　　ローカル変数保存の際の出力イメージ

```
スコープ：現在のスコープ
[V] 変数　[P] 引数　[F] 関数
名前                    値          型情報 ( バイト数 )          アドレス
-------------------------------------------------------------------------
[V] 変数名 [1]          値          型情報 ( バイト数 )          アドレス
 -[V] 変数名 [0]        値          型情報 ( バイト数 )          アドレス
       :                :                 :                       :
```

2.10.6 ウォッチ式を表示／変更する

C 言語変数，CPU レジスタ，I/O レジスタ，およびアセンブラ・シンボルなどをウォッチ式として，次のウォッチ パネルに登録することにより，それらの値を常にデバッグ・ツールから取得し，一括して値を監視することができます。

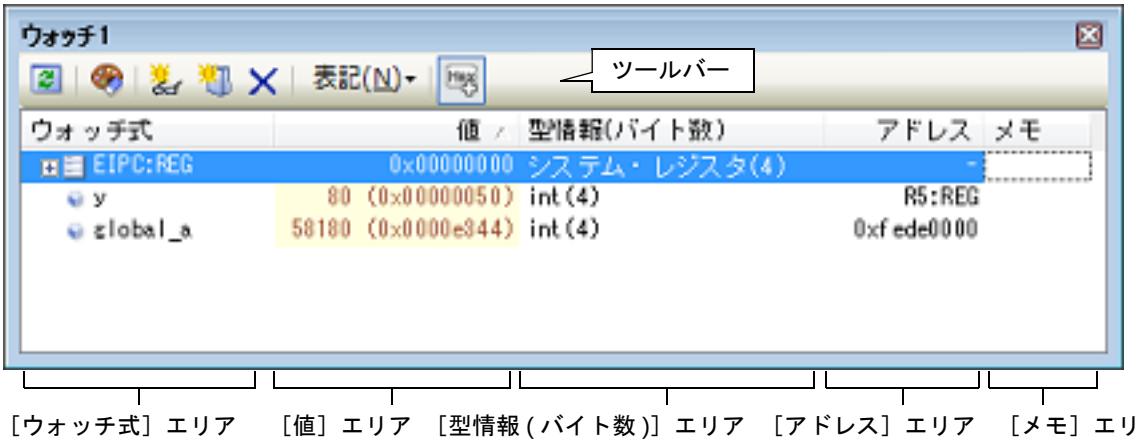
また，ウォッチ式は，プログラムが実行中の状態にあっても値の表示を逐次更新することができます（「2.10.6.7 プログラム実行中にウォッチ式の内容を表示／変更する」参照）。

ウォッチ パネルは，［表示］メニュー→［ウォッチ］→［ウォッチ *1* ～ *4*］の選択でオープンします。

ウォッチ パネルは，最大 4 個までオープンすることができます。各パネルは，タイトルバーの “ ウォッチ 1”，“ ウォッチ 2”，“ ウォッチ 3”，“ ウォッチ 4” の名称で識別され，それぞれのウォッチ パネルが個別にウォッチ式を登録／管理し，プロジェクトのユーザ情報として保存されます。

なお，各エリアの見方，および機能についての詳細は，ウォッチ パネルの項を参照してください。

図 2.85　　ウォッチ式の内容の表示（ウォッチ パネル）



ここでは，次の操作方法について説明します。

2.10.6.1 ウォッチ式を登録する

ウォッチ式の登録方法には，次の 3 通りがあります（デフォルトでは，ウォッチ式は登録されていません）。

注意 1. 1 つのウォッチ パネルにおいて，ウォッチ式は 3000 個まで登録することができます（上限値を越えて登録しようとした場合，メッセージを表示します）。

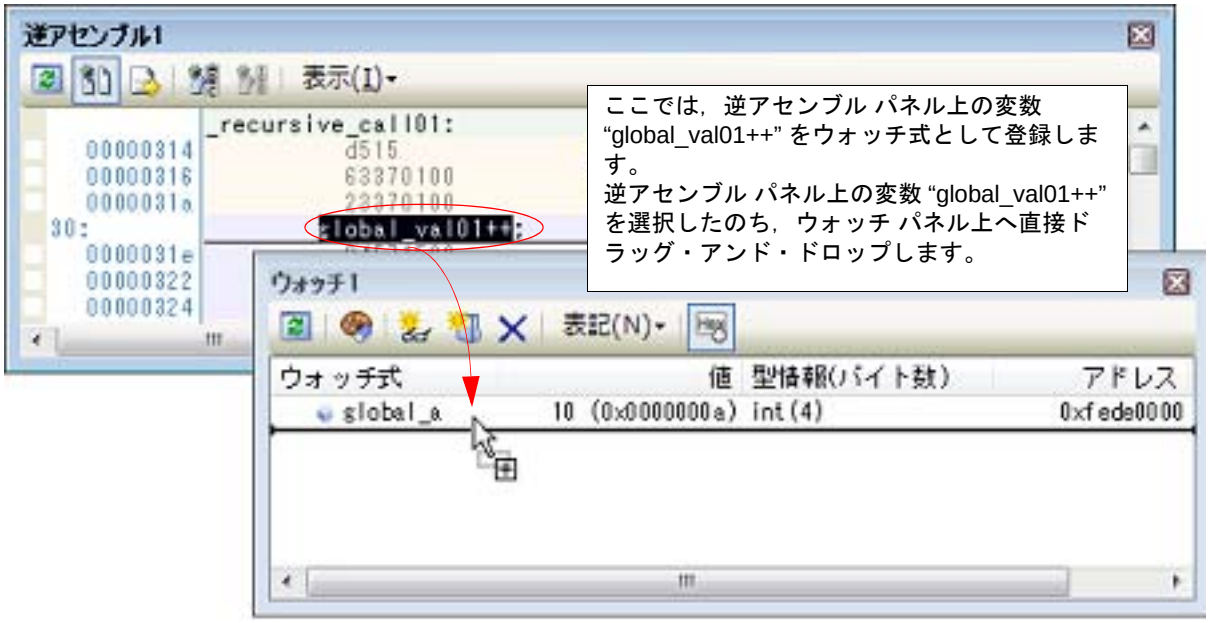
注意 2. コンパイラによる最適化のため，対象となる変数を使用していないブロックでは変数データがスタック／レジスタに存在しない場合があります。この場合，対象となる変数をウオッチ式として登録しても値の表示は “?” のままとなります。

備考 1. 各ウォッチ パネル（ウォッチ 1 ～ウォッチ 4）上で登録したウォッチ式は，それぞれ個別に管理され，プロジェクトのユーザ情報として保存されます。

備考 2. ウォッチ式は，同名を複数登録することができます。

(1) 他のパネルから登録する場合
CS+ の他のパネルから，ウォッチ式を登録することができます。
他のパネルにおいて，ウォッチ式として登録したい対象を任意のウォッチ パネル（ウォッチ 1 ～ウォッチ 4）上に直接ドラッグ・アンド・ドロップします。
なお，この操作が可能なパネルと，ウォッチ式として登録可能な対象との関係についての詳細は，「表 A.2 各パネルとウォッチ式として登録可能な対象の関係」を参照してください。

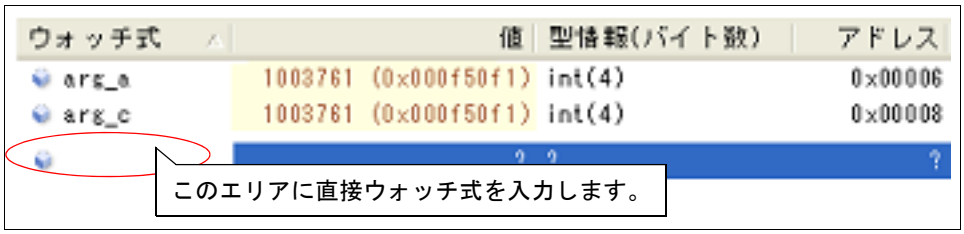
図 2.86 他のパネルからウォッチ式登録する場合の例



備考 ウォッチ式として登録したい対象を選択したのち，または対象文字列のいずれかにキャレットを移動したのち（対象は自動的に決定されます），コンテキスト・メニューの［ウォッチ 1 に登録］を選択することによっても同様にウォッチ式を登録することができます（ただし，ウォッチ パネル（ウォッチ 1）に限定）。

(2) **ウォッチ パネル上で**直接登録する場合
任意のウォッチ パネル（ウォッチ 1 ～ウォッチ 4）において，ツールバーの  ボタンをクリックすると，［ウォッチ式］エリアに次のエントリ・ボックスが表示されます。

図 2.87 ウォッチ式のエントリ・ボックス



エントリ・ボックス内に，キーボートより直接ウォッチ式を入力したのち，［Enter］キーを押下します。
なお，この際のウォッチ式の入力形式についての詳細は，次を参照してください。

- 「表 2.26 ウォッチ式の基本入力形式」
- 「表 A.3 C 言語関数をスコープ指定してウォッチ登録した場合の扱い」
- 「表 A.5 CPU レジスタをスコープ指定してウォッチ登録した場合の扱い」
- 「表 A.6 I/O レジスタをスコープ指定してウォッチ登録した場合の扱い」

備考 このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

(3) 他のアプリケーションから登録する場合
外部エディタなどから，C 言語変数 /CPU レジスタ /I/O レジスタ／アセンブラ・シンボルの文字列を選択し，ウォッチ パネル（ウォッチ 1 ～ウォッチ 4）に直接ドラッグ・アンド・ドロップします。
この場合，ドロップした文字列がそのままウォッチ式として登録されます。

2.10.6.2 登録したウォッチ式を整理する

登録したウォッチ式をカテゴリ（フォルダ）で分類し，ツリー形式で表示することができます（デフォルトでは，カテゴリは存在しません）。

注意 1. カテゴリ内にカテゴリを作成することはできません。

注意 2. 1 つのウォッチ パネルにおいて，カテゴリは 1500 個まで作成することができます（上限値を越えて作成しようとした場合，メッセージを表示します）。

(1) カテゴリを新規作成する場合
作成したい位置にキャレットを移動したのち，ツールバーの  ボタンのクリックし，キーボードより新規カテゴリ名を直接入力します。

(2) カテゴリ名を編集する場合
編集したいカテゴリ名を選択したのち，再度クリックし，キーボードよりカテゴリ名を直接編集します。

(3) カテゴリを削除する場合
削除したいカテゴリを選択したのち，ツールバーの  ボタンをクリックします。

(4) 表示順を変更する場合
登録済みのウォッチ式を作成したカテゴリ内に直接ドラッグ・アンド・ドロップすることにより，ウォッチ式はカテゴリで分類されます。
また，カテゴリとウォッチ式の表示の順番（上下位置）も，ドラッグ・アンド・ドロップ操作により自由に変更することができます。

備考 ウォッチ式／カテゴリを他のウォッチ パネル（ウォッチ 1 ～ウォッチ 4）にドラッグ・アンド・ドロップすると，ドロップ先のウォッチ パネルにウォッチ式／カテゴリがコピーされます。

2.10.6.3 登録したウォッチ式を編集する

登録したウォッチ式は，編集することができます。
対象ウォッチ式をダブルクリックすると，対象ウォッチ式が編集モードになります（［Esc］キーの押下で編集モードをキャンセルします）。
キーボードより直接内容を編集し，［Enter］キーを押下してください。

2.10.6.4 ウォッチ式を削除する

登録したウォッチ式を削除する場合は，ウォッチ パネルにおいて，削除したいウォッチ式を選択したのち，ツールバーの  ボタンをクリックします。

2.10.6.5 値の表示形式を変更する

ツールバーの次のボタンにより，このパネルの［値］エリアの表示形式を変更することができます。

表記	値の表示形式を変更する次のボタンを表示します。
 自動	選択しているウォッチ式の値の表記を変数ごとの規定値（「表 A.7 ウォッチ式の表示形式（デフォルト）」参照）で表示します（デフォルト）。
 16 進数	選択している項目の値を 16 進数で表示します。
 符号付き 10 進数	選択している項目の値を符号付き 10 進数で表示します。
 符号無し 10 進数	選択している項目の値を符号なし 10 進数で表示します。
 8 進数	選択している項目の値を 8 進数で表示します。
 2 進数	選択している項目の値を 2 進数で表示します。
 ASCII	選択している項目の値を ASCII コードで表示します。
 Float	選択している項目の値を Float で表示します。 ただし，選択しているウォッチ式が 4 バイト・データの場合のみ有効となります。
 Double	選択している項目の値を Double で表示します。 ただし，選択しているウォッチ式が 8 バイト・データの場合のみ有効となります。
	選択している項目の値表示の末尾に，その値の 16 進数表記を “（）” で囲んで併記します。 ただし，16 進数表記をしている場合は併記しません。

2.10.6.6 ウォッチ式の内容を変更する

ウォッチ式の値は，編集することができます。
［値］エリアにおいて，対象ウォッチ式の値をダブルクリックすると，値が編集モードになります（［Esc］キーの押下で編集モードをキャンセルします）。
値をキーボードより直接編集したのち，［Enter］キーを押下することにより，変更した値がデバッグ・ツールのターゲット・メモリに書き込まれます。
ただし，値を変更できるのは，C 言語変数 /CPU レジスタ /I/O レジスタ／アセンブラ・シンボルと 1 対 1 に対応するウォッチ式のみです。また，読み込み専用の I/O レジスタの値を変更することもできません。
なお，ウォッチ式の値の編集は，プログラム実行中の状態でも行うことができます。設定方法についての詳細は，「2.10.1.4 プログラム実行中にメモリの内容を表示／変更する」を参照してください。

備考 1. 変数のサイズより小さい桁の数値が入力された場合，上位の桁を 0 でパディングします。

備考 2. 変数のサイズより大きい桁の数値が入力された場合，上位の桁をマスクします。

備考 3. 文字配列（char 型，unsigned char 型）に対しては，表示形式に ASCII が選択されている場合，文字列（ASCII/Shift_JIS/EUC-JP/Unicode（UTF-8/UTF-16））による値の入力も可能です。

備考 4. ウォッチ式の値には，次のように ASCII 文字による入力も可能です。

- ASCII 文字による入力の場合
  変数 “ch” の［値］エリアに “’A’” を入力
  → “ch” が割り当てられているメモリ領域に “0x41” を書き込む
- 数値による入力の場合
  変数 “ch” の［値］エリアに “0x41” を入力
  → “ch” が割り当てられているメモリ領域に “0x41” を書き込む
- 文字列（ASCII）による入力の場合
  文字配列 “str” の表示形式を ASCII に設定し，［値］エリアに “”ABC”” を入力
  → “str” が割り当てられているメモリ領域に “0x41，0x42，0x43，0x00” を書き込む

2.10.6.7 プログラム実行中にウォッチ式の内容を表示／変更する

メモリ パネル／ウォッチ パネルでは，プログラムの実行中に，リアルタイムにメモリ／ウォッチ式の内容を表示更新，および書き換えることができるリアルタイム表示更新機能を備えています。
このリアルタイム表示更新機能を有効にすることにより，プログラムが停止している状態の時だけでなく，実行中の状態であっても，メモリ／ウォッチ式の値の表示／変更を行うことができます。
設定方法についての詳細は，「2.10.1.4 プログラム実行中にメモリの内容を表示／変更する」を参照してください。

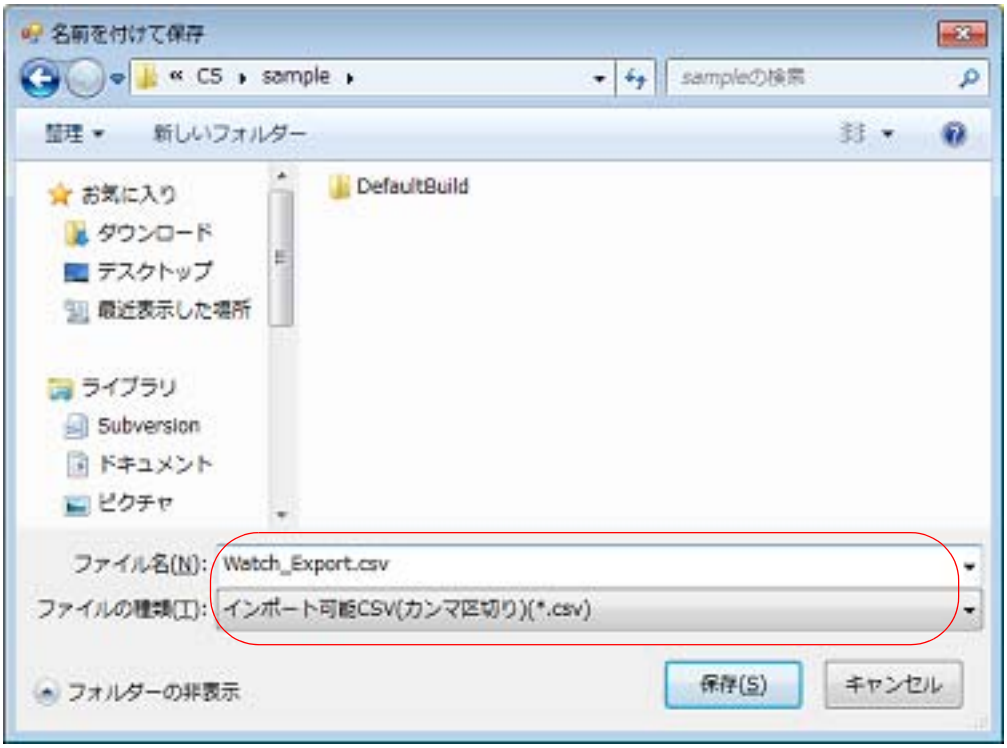
2.10.6.8 ウォッチ式をエクスポート／インポートする

現在登録しているウォッチ式をファイルにエクスポートし，そのファイルをインポートすることにより，ウォッチ式を再登録することができます。
この場合，次の操作を行ってください。

(1) ウォッチ式をエクスポートする
現在登録しているウォッチ式（カテゴリを含む）を，インポート可能なファイル形式で保存します。
ウォッチ パネルにフォーカスがある状態で，［ファイル］メニュー→［名前を付けてウォッチ・データを保存 ...］を選択します。
オープンする名前を付けて保存 ダイアログにおいて，次の指定を行ったのち，［保存］ボタンをクリックします。
［ファイル名］: 保存するファイル名（*.csv）を指定します。
［ファイルの種類］: “インポート可能 CSV(カンマ区切り) (*.csv)” を選択します。

注意 値，および型情報は保存されません。
また，配列や構造体などのウォッチ式を解析後に展開される項目は保存されません。

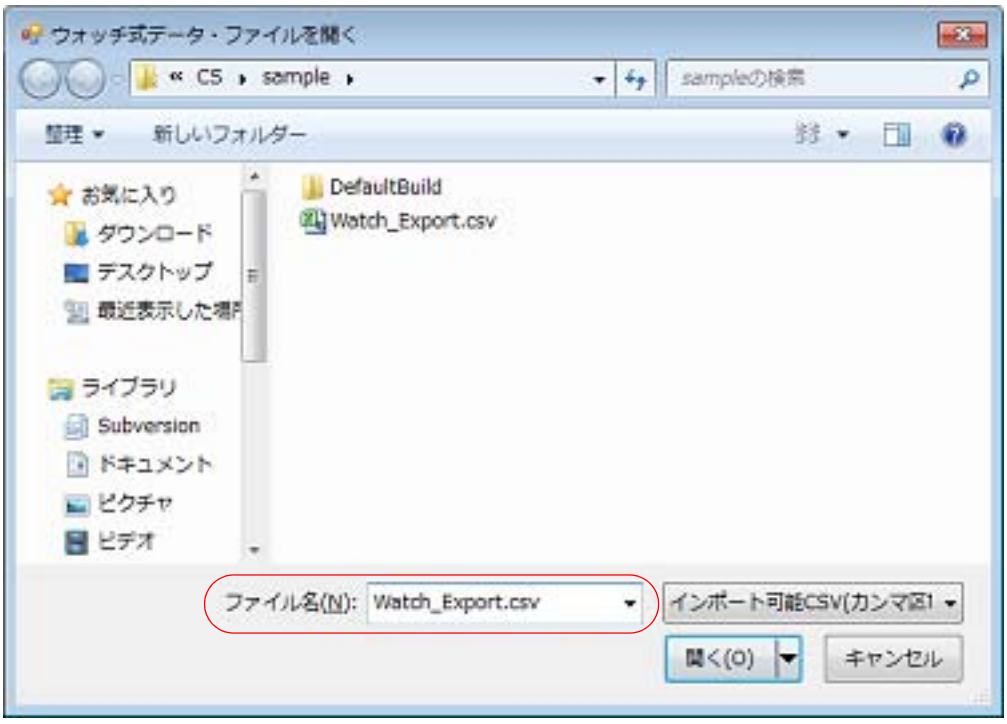
図 2.88 ウォッチ式のエクスポート



(2) ウォッチ式をインポートする
(1) でエクスポートしたファイルを，ウォッチ パネルにインポートします。
ウォッチ式をインポートしたいウォッチ パネルにおいて，コンテキスト・メニューの［ウォッチ式をインポート ...］を選択します。
オープンするウォッチ式データ・ファイルを開く ダイアログにおいて，先にエクスポートしたファイルを指定したのち，［開く］ボタンをクリックします。

備考 すでにウォッチ式が登録されている場合，最下部のウォッチ式の直後にインポートしたウォッチ式が登録されます。

図 2.89 ウォッチ式のインポート



2.10.6.9 ウォッチ式の表示内容を保存する

［ファイル］メニュー→［名前を付けてウォッチ・データを保存 ...］を選択することにより，名前を付けて保存 ダイアログをオープンし，ウォッチ式と値のすべての内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。

ファイルに保存する際は，すべてのウォッチ式の値を再読み込みし，取得した最新の値を保存します。

なお，配列，ポインタ型変数，構造体／共用体，レジスタ（部分名がついているもののみ）が展開表示している場合は，各展開要素の値も保存します。展開表示していない場合は，先頭に “+” マークを付与して値は空欄になります。

ただし，読み込み保護対象の I/O レジスタの再読み込みは行いません。最新の内容を保存したい場合は，コンテキスト・メニューの［値を強制読み込み］を選択したのち，ファイルの保存を行ってください。

図 2.90 ウォッチ・データ保存の際の出力イメージ

```
ウォッチ式      値       型情報 ( バイト数 )     アドレス      メモ
-----------------------------------------------------------------------
変数式          値       型情報 ( バイト数 )     アドレス      メモ
- カテゴリ名
  変数式        値       型情報 ( バイト数 )     アドレス      メモ
      :         :              :                   :          :
```

備考 ［ファイル］メニュー→［ウォッチ・データを保存］の選択によりパネルの内容を上書き保存した場合，ウォッチ パネル（ウォッチ 1 ～ 4）はそれぞれ個別に扱われます。

2.11 スタックからの関数呼び出し情報の表示

この節では，スタックからの関数呼び出し情報の表示方法について説明します。
CS+ が提供するコンパイラ（CC-RH）は，ANSI 規格に沿って関数呼び出し情報をスタックに積んでいます。この関数呼び出し情報（以降，コール・スタック情報と呼びます）を解析することで，関数の呼び出しの深さ，呼び出し元位置，および引数などを知ることができます。

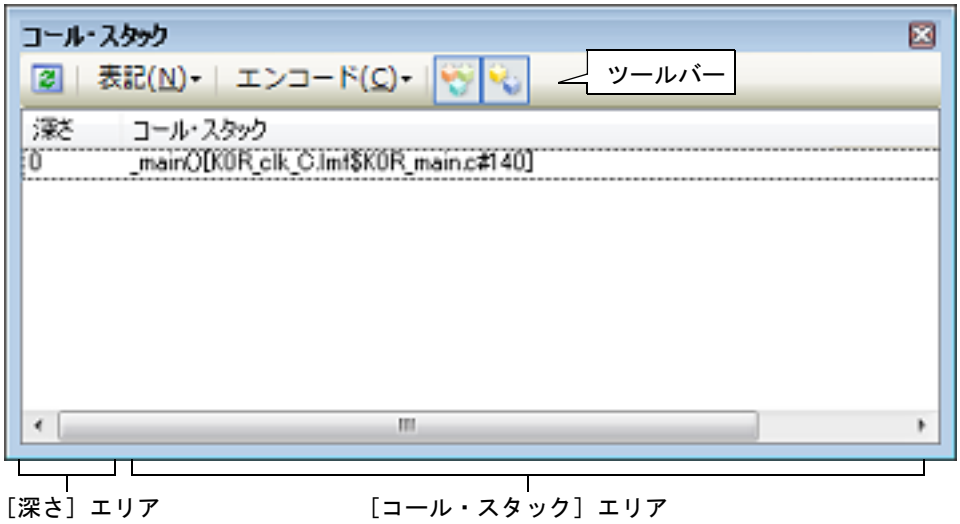
備考　　マルチコア対応版を対象とした“コール・スタック情報”については，「2.7 コア（PE）の選択」も参照してください。

2.11.1 コール・スタック情報を表示する

コール・スタック情報の表示は，次のコール・スタック パネルで行います。
［表示］メニュー→［コール・スタック］を選択してください。
なお，各エリアの見方，および機能についての詳細は，コール・スタック パネルの項を参照してください。

注意　　プログラム実行中は，このパネルには何も表示されません。
プログラムの実行が停止したタイミングで，各エリアの表示を行います。

図 2.91　　コール・スタック情報の表示（コール・スタック パネル）



ここでは，次の操作方法について説明します。

2.11.1.1 値の表示形式を変更する

ツールバーの次のボタンにより，このパネルの表示形式を変更することができます。

表記	値の表示形式を変更する次のボタンを表示します。
自動	このパネル上の値の表記を変数ごとの規定値で表示します（デフォルト）。
16 進数	このパネル上の値を 16 進数で表示します。
10 進数	このパネル上の値を 10 進数で表示します。
8 進数	このパネル上の値を 8 進数で表示します。
2 進数	このパネル上の値を 2 進数で表示します。

エンコード	文字列変数のエンコードを変更する次のボタンを表示します。
ASCII	このパネル上の文字列変数を ASCII コードで表示します（デフォルト）。
Shift_JIS	このパネル上の文字列変数を Shift_JIS コードで表示します。
EUC-JP	このパネル上の文字列変数を EUC-JP コードで表示します。
UTF-8	このパネル上の文字列変数を UTF-8 コードで表示します。
UTF-16	このパネル上の文字列変数を UTF-16 コードで表示します。

2.11.1.2 ソース行へジャンプする

行をダブルクリックすることにより，その行が示す関数呼び出し元のソース行にキャレットを移動した状態でエディタ パネルがオープンします（すでにオープンしている場合は，エディタ パネルにジャンプ）。

備考　コンテキスト・メニューの［逆アセンブルへジャンプ］を選択することにより，現在選択している行が示す関数呼び出し元のアドレスにキャレットを移動した状態で逆アセンブル パネル（逆アセンブル 1）がオープンします（すでにオープンしている場合は，逆アセンブル パネル（逆アセンブル 1）にジャンプ）。

2.11.1.3 ローカル変数を表示する

コンテキスト・メニューの［このときのローカル変数を表示］を選択することにより，現在選択している行が示す関数のローカル変数を表示するローカル変数 パネルをオープンします。

2.11.1.4 コール・スタック情報の表示内容を保存する

［ファイル］メニュー→［名前を付けてコール・スタック・データを保存 ...］を選択することにより，名前を付けて保存 ダイアログをオープンし，コール・スタック情報のすべての内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
ファイルに保存する際は，デバッグ・ツールから最新の情報を取得します。

図 2.92　コール・スタック情報保存の際の出力イメージ

```
深さ      コール・スタック
------------------------------------------------------------------------
 0        コール・スタック情報
 1        コール・スタック情報
 :              :
```

2.12 実行履歴の収集

この節では，プログラムの実行履歴の収集方法について説明します。
一般的に，プログラムの実行履歴をトレースと呼び，以降の記述で使用します。プログラムが暴走した場合，暴走後のメモリ内容やスタック情報などから原因を探ることは非常に困難ですが，収集したトレース・データの内容を解析することにより，暴走するまでの過程を直接探ることができ，プログラムの潜在的バグを発見するために有効です。

備考　　マルチコア対応版を対象とした“実行履歴の収集”については，「2.7 コア（PE）の選択」も参照してください。

2.12.1 トレース動作の設定をする

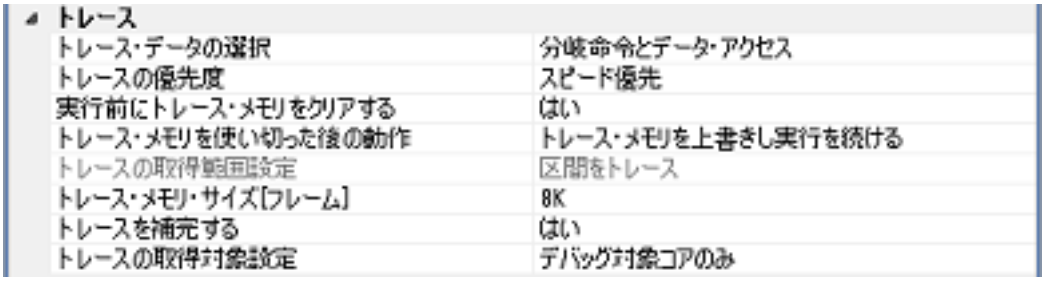
トレース機能が開始すると，現在実行中のプログラムの実行過程を記録したトレース・データがトレース・メモリに収集されます（プログラムの実行が停止すると，自動的にトレース機能も停止します）。
トレース機能を使用するためには，あらかじめトレースの動作に関する設定を行う必要があります。
なお，設定方法は，使用するデバッグ・ツールにより異なります。

2.12.1.1 【Full-spec emulator】の場合

設定は，プロパティ パネルの［デバッグ・ツール設定］タブ上の［トレース］カテゴリ内で行います。

注意　　プログラム実行中は，このカテゴリ内のプロパティを変更することはできません。

図 2.93　［トレース］カテゴリ【Full-spec emulator】



(1)　［トレース・データの選択］
収集するトレース・データの種類を次のドロップダウン・リストにより選択します。

分岐命令	プログラム実行中に発生した分岐処理の分岐元／分岐先の命令の PC 値をトレース・データとして収集します。
データ・アクセス	プログラム実行中に成立したアクセス系イベントのデータ情報をトレース・データとして収集します。
分岐命令とデータ・アクセス	プログラム実行中に発生した分岐処理の分岐元／分岐先の命令の PC 値，および成立したアクセス系イベントのデータ情報をトレース・データとして収集します（デフォルト）。

注意　　このプロパティを変更すると，トレース・メモリがクリアされます。

(2)　［トレースの優先度］
トレース機能を使用する際の優先度を次のドロップダウン・リストにより選択します。

スピード優先	リアルタイム性を優先してトレースを行います（デフォルト）。
データ優先	データの取りこぼしが発生しないように，CPU の実行パイプラインを一時的に停止してトレースを行います。

注意　　このプロパティを変更すると，トレース・メモリがクリアされます。

(3)　［実行前にトレース・メモリをクリアする］
トレース機能を開始する前に，トレース・メモリを一度クリア（初期化）するか否かを選択します。
クリアする場合は［はい］を選択してください（デフォルト）。

備考　　トレース パネルのツールバーの  ボタンをクリックすることにより，トレース・メモリを強制的にクリアすることができます。

(4)　［トレース・メモリを使い切った後の動作］
収集したトレース・データでトレース・メモリがいっぱいになった際の動作を，次のドロップダウン・リストにより選択します。

トレース・メモリを上書きし実行を続ける	トレース・メモリがいっぱいになると，古いトレース・データに上書きを続けます（デフォルト）。 ［実行前にトレース・メモリをクリアする］プロパティで［はい］を選択している場合は，再実行時，トレース・データをクリアしたのち，トレース・データの書き込みを行います。
トレースを停止する	トレース・メモリがいっぱいになると，トレース・データの書き込みを停止します（プログラムの実行は停止しません）。 ただし，［トレースの優先度］プロパティで［データ優先］を選択している場合は，この項目は表示されません。
停止する	トレース・メモリがいっぱいになると，トレース・データの書き込みを停止すると同時にプログラムの実行を停止します。

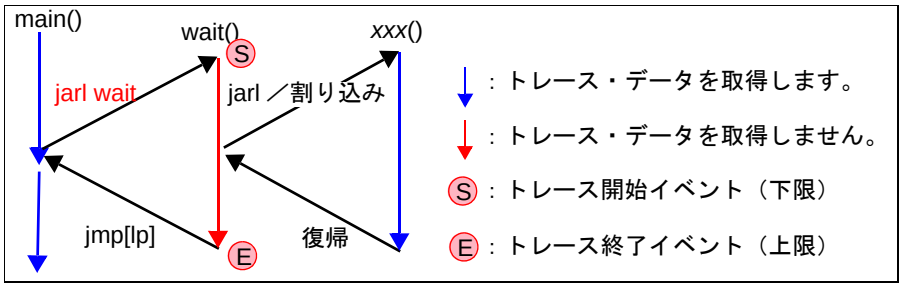
注意　　このプロパティを変更すると，トレース・メモリがクリアされます。

(5)　［トレースの取得範囲設定］
トレース・データの取得範囲を次のドロップダウン・リストにより選択します。
ただし，このプロパティは，デバッグ・ツールと接続時のみ変更することができます。

区間をトレース	トレース開始イベントとトレース終了イベントで設定した区間の実行履歴をトレース・データとして収集します（デフォルト）。
範囲外をトレース	トレース開始イベントとトレース終了イベントで設定した範囲外の実行履歴をトレース・データとして収集します。

注意　　このプロパティを変更した場合，現在設定しているトレース開始イベント／トレース終了イベントはすべて無効となります。

備考　　［範囲外をトレース］を選択した場合では，次のように，トレース開始イベント／トレース終了イベントで設定するアドレスを下限／上限とし，トレース・データの取得範囲が決定されます。



(6)　［トレース・メモリ・サイズ [フレーム]］
トレース・メモリのサイズ（トレース・フレーム数）を選択します。
なお，トレース・フレームはトレース・データの一単位を表し，フェッチ／ライト／リードなどで，それぞれ 1 つのトレース・フレームを使用します（デフォルト：[8K]）。

注意　　このプロパティを変更すると，トレース・メモリがクリアされます。

(7)　［トレースを補完する］
収集したトレース・データをトレース パネルで表示する際に，補完表示を行うか否かを選択します。
補完表示を行うことにより，ハードウエアではトレースできない分岐命令間の命令の表示が可能となります。
補完表示を行う場合は［はい］を選択してください（デフォルト）。
なお，この設定は，次回取得するトレース・データより反映されます。

(8)　［トレースの取得対象設定］
トレースの対象となるコアを次のドロップダウン・リストにより選択します。

デバッグ対象コアのみ	現在デバッグ対象に選択している PE のみを対象にトレース・データを収集します（デフォルト）。 トレース・データ収集後，PE を切り替えてもトレース パネルの表示内容は変わりません。
全てのコア	全 PE を対象にトレース・データを収集します。 トレース・データ収集後，PE を切り替えることにより，対応するトレース・データの内容をトレース パネルに表示します。

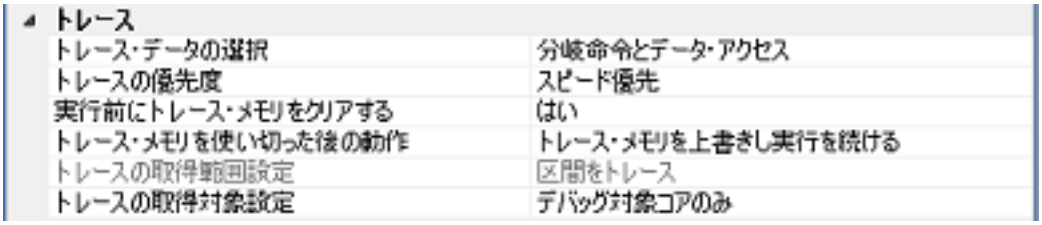
2.12.1.2 【E1】／【E20】の場合

設定は，プロパティ パネルの［デバッグ・ツール設定］タブ上の［トレース］カテゴリ内で行います。

注意 1.　接続したマイクロコントローラがトレース機能を搭載していない場合，デバッグ・ツールと接続後，このカテゴリ内のプロパティは変更不可状態となります（トレース機能を使用することはできません）。

注意 2.　プログラム実行中は，このカテゴリ内のプロパティを変更することはできません。

図 2.94　［トレース］カテゴリ【E1】【E20】



(1)　［トレース・データの選択］
収集するトレース・データの種類を次のドロップダウン・リストにより選択します。

分岐命令	プログラム実行中に発生した分岐処理の分岐元／分岐先の命令の PC 値をトレース・データとして収集します。
データ・アクセス	プログラム実行中に成立したアクセス系イベントのデータ情報をトレース・データとして収集します。
分岐命令とデータ・アクセス	プログラム実行中に発生した分岐処理の分岐元／分岐先の命令の PC 値，および成立したアクセス系イベントのデータ情報をトレース・データとして収集します（デフォルト）。

注意　このプロパティを変更すると，トレース・メモリがクリアされます。

(2)　［トレースの優先度］
トレース機能を使用する際の優先度を次のドロップダウン・リストにより選択します。

スピード優先	リアルタイム性を優先してトレースを行います（デフォルト）。
データ優先	データの取りこぼしが発生しないように，CPU の実行パイプラインを一時的に停止してトレースを行います。

注意　このプロパティを変更すると，トレース・メモリがクリアされます。

(3)　［実行前にトレース・メモリをクリアする］
トレース機能を開始する前に，トレース・メモリを一度クリア（初期化）するか否かを選択します。
クリアする場合は［はい］を選択してください（デフォルト）。

備考　トレース パネルのツールバーの  ボタンをクリックすることにより，トレース・メモリを強制的にクリアすることができます。

(4)　［トレース・メモリを使い切った後の動作］
収集したトレース・データでトレース・メモリがいっぱいになった際の動作を，次のドロップダウン・リストにより選択します。

トレース・メモリを上書きし実行を続ける	トレース・メモリがいっぱいになると，古いトレース・データに上書きを続けます（デフォルト）。 ［実行前にトレース・メモリをクリアする］プロパティで［はい］を選択している場合は，再実行時，トレース・データをクリアしたのち，トレース・データの書き込みを行います。
トレースを停止する	トレース・メモリがいっぱいになると，トレース・データの書き込みを停止します（プログラムの実行は停止しません）。
停止する	トレース・メモリがいっぱいになると，トレース・データの書き込みを停止すると同時にプログラムの実行を停止します。

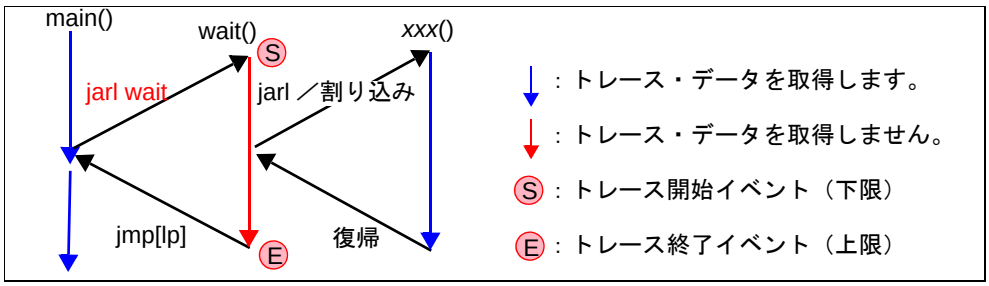
注意　　このプロパティを変更すると，トレース・メモリがクリアされます。

(5)　［トレースの取得範囲設定］
トレース・データの取得範囲を次のドロップダウン・リストにより選択します。
ただし，このプロパティは，デバッグ・ツールと接続時のみ変更することができます。

区間をトレース	トレース開始イベントとトレース終了イベントで設定した区間の実行履歴をトレース・データとして収集します（デフォルト）。
範囲外をトレース	トレース開始イベントとトレース終了イベントで設定した範囲外の実行履歴をトレース・データとして収集します。

注意　　このプロパティの設定を変更した場合，現在設定しているトレース開始イベント／トレース終了イベントはすべて無効となります。

備考　　［範囲外をトレース］を選択した場合では，次のように，トレース開始イベント／トレース終了イベントで設定するアドレスを下限／上限とし，トレース・データの取得範囲が決定されます。



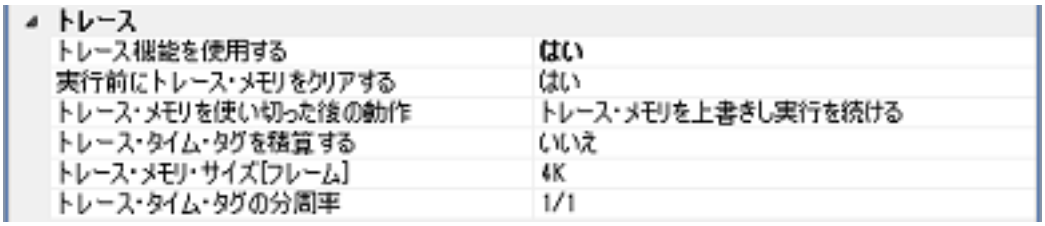
(6)　［トレースの取得対象設定］
トレースの対象となるコアを次のドロップダウン・リストにより選択します。

デバッグ対象コアのみ	現在デバッグ対象に選択している PE のみを対象にトレース・データを収集します（デフォルト）。 トレース・データ収集後，PE を切り替えてもトレース パネルの表示内容は変わりません。
全てのコア	全 PE を対象にトレース・データを収集します。 トレース・データ収集後，PE を切り替えることにより，対応するトレース・データの内容をトレース パネルに表示します。

2.12.1.3 【シミュレータ】の場合

設定は，プロパティ パネルの［デバッグ・ツール設定］タブ上の［トレース］カテゴリ内で行います。

図 2.95　［トレース］カテゴリ【シミュレータ】



(1)　［トレース機能を使用する］
トレース機能を使用するか否かを選択します。
トレース機能を使用する場合は［はい］を選択してください（デフォルト：［いいえ］）。

(2)　［実行前にトレース・メモリをクリアする］
トレース機能を開始する前に，トレース・メモリを一度クリア（初期化）するか否かを選択します。
クリアする場合は［はい］を選択してください（デフォルト）。

備考　トレース パネルのツールバーの  ボタンをクリックすることにより，トレース・メモリを強制的にクリアすることができます。

(3)　［トレース・メモリを使い切った後の動作］
トレース・メモリが収集したトレース・データでいっぱいになった際の動作を，次のドロップダウン・リストにより選択します。

トレース・メモリを上書きし実行を続ける	トレース・メモリがいっぱいになると，古いトレース・データに上書きを続けます（デフォルト）。 ［実行前にトレース・メモリをクリアする］プロパティで［はい］を選択している場合は，再実行時，トレース・データをクリアしたのち，トレース・データの書き込みを行います。
トレースを停止する	トレース・メモリがいっぱいになると，トレース・データの書き込みを停止します（プログラムの実行は停止しません）。
停止する	トレース・メモリがいっぱいになると，トレース・データの書き込みを停止すると同時にプログラムの実行を停止します。

(4)　［トレース・タイム・タグを積算する］
トレースの時間表示を積算表示にするか否かを選択します。
トレースの時間表示を積算表示にする場合は［はい］を，差分表示にする場合は［いいえ］を選択してください（デフォルト）。

(5)　［トレース・メモリ・サイズ [フレーム]］
トレース・メモリのサイズ（トレース・フレーム数）をドロップダウン・リストにより選択します。
なお，トレース・フレームはトレース・データの一単位を表し，フェッチ／ライト／リードなどで，それぞれ 1 つのトレース・フレームを使用します（デフォルト：［4K］）。

(6)　［トレース・タイム・タグの分周率］
トレースのタイム・タグ（トレース パネルの［時間］表示）で使用するカウンタの分周率を，ドロップダウン・リストにより選択します（デフォルト：［1/1］）。

2.12.2 実行停止までの実行履歴を収集する

デバッグ・ツールには，プログラムの実行開始から実行停止までの実行履歴を収集する機能があらかじめ用意されています。
これにより，プログラムの実行を開始することにより自動的にトレース・データの収集が開始し，実行停止とともにトレース・データの収集も終了します。
なお，収集したトレース・データの確認方法についての詳細は，「2.12.6 実行履歴を表示する」を参照してください。

備考　この機能は，デバッグ・ツールにデフォルトで設定されているビルトイン・イベントの 1 つである無条件トレース・イベントにより動作します。
したがって，イベント パネル上の無条件トレース・イベントのチェックを外し，無効状態にした場合，プログラムの実行開始に連動したトレース・データの収集は行いません（無条件トレース・イベントはデフォルトで有効状態に設定されています）。
なお，この無条件トレース・イベントと後述のトレース・イベント（「2.12.3 任意区間の実行履歴を収集する」参照）は排他使用のイベントとなります。そのため，トレース・イベントが有効状態で設定さ

れると，無条件トレース・イベントは自動的に無効状態に変更されます。

2.12.3 任意区間の実行履歴を収集する

トレース・イベントを設定することにより，プログラムの実行過程において，任意の区間の実行履歴のみをトレース・データとして収集することができます。
なお，トレース・イベントは，トレース開始イベントとトレース終了イベントで構成されます。
この機能を使用するためには，次の手順で操作を行います。

注意 1. トレース・イベントの設定に関しては（有効イベント数の制限など），「2.16.6 イベント設定に関する留意事項」も参照してください。

注意 2. 【シミュレータ】
トレーサ動作中は，トレース開始イベント／トレース終了イベントの設定／削除はできません。

2.12.3.1 トレース・イベントを設定する

トレース・イベントを設定するため，トレース・データの収集を開始／終了するトレース開始イベント／トレース終了イベントを設定します。
トレース開始イベント／トレース終了イベントの設定は，次いずれかの操作により行います。

(1) 実行系イベントの場合
実行系イベントをトレース開始イベント／トレース終了イベントに設定することにより，任意の箇所でトレース・データの収集を開始／終了させることができます。
操作は，ソース・テキスト／逆アセンブル・テキストを表示しているエディタ パネル／逆アセンブル パネルで行います。
各パネルのアドレス表示のある行にキャレットを移動したのち，目的のイベント種別に従って，コンテキスト・メニューより次の操作を行います。

イベント種別	操作方法
トレース開始	［トレース設定］→［トレース開始の設定］
トレース終了	［トレース設定］→［トレース終了の設定］

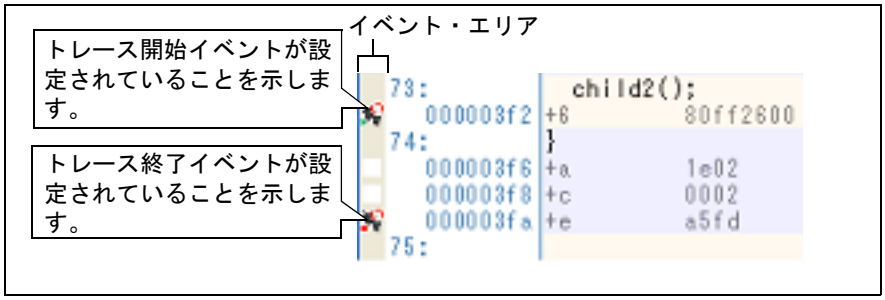
注意 【シミュレータ】
トレース終了イベントはトレース・データとして表示されません。
トレース・データとして表示する場合は，1 行下にトレース終了イベントを設定してください。

トレース開始イベント／トレース終了イベントは，キャレット位置の行に対応する先頭アドレスの命令に設定されます。
トレース開始イベント／トレース終了イベントが設定されると，設定した行のイベント・エリアに次のイベント・マークが表示されます。

表 2.11 トレース開始イベント／トレース終了イベント・マーク

イベント種別	イベント・マーク
トレース開始	
トレース終了	

図 2.96 トレース開始イベント／トレース終了イベントの設定例（逆アセンブル パネルの場合）

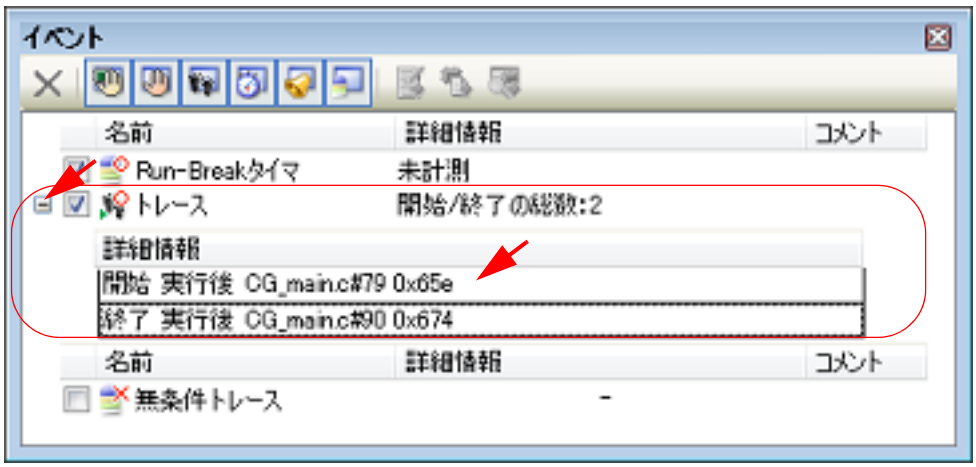


備考 【Full-spec emulator】【E1】【E20】
プロパティ パネルの［デバッグ・ツール設定］タブ上の［トレース］カテゴリ内［トレースの取得範囲設定］プロパティを［範囲外をトレース］に指定することにより，設定した範囲外の実行履歴をトレース・データとして取得することができます。

(2) アクセス系イベントの場合

トレース開始イベント／トレース終了イベントが設定されると，イベント パネル上において，トレース・イベントとして 1 つにまとめて管理され（「2.16 イベントの管理」参照），トレース・イベント項目の “+” マークをクリックすることにより，設定したトレース開始イベント／トレース終了イベントの詳細情報が表示されます。

図 2.97 イベント パネルのトレース開始イベント／トレース終了イベント（実行系）の設定例



備考 1. トレース開始イベント／トレース終了イベントのいずれかが有効状態で設定されると，イベント パネル上の無条件トレース・イベントのチェックが自動的に外れ，プログラムの実行開始に連動したトレース・データの収集は行いません（設定したトレース開始イベントの条件が成立するまでトレーサは動作しません）。

備考 2. トレース終了イベントが不要な場合は，未設定でもかまいません。

備考 3. イベントの設定状態によりイベント・マークは異なります（「2.16.1 設定状態（有効／無効）を変更する」参照）。
また，すでにイベントが設定されている箇所で，新たにイベントを設定した場合は，複数のイベントが設定されていることを示すイベント・マーク（ ）が表示されます。

備考 4. 【シミュレータ】
トレース開始イベント／トレース終了イベントのいずれかが有効状態で設定されると，プロパティ パネルの［デバッグ・ツール設定］タブ上の［トレース］カテゴリ内［トレース機能を使用する］プロパティの指定を自動的に［はい］に変更し，トレース機能が有効化されます。

2.12.3.2 プログラムを実行する

プログラムを実行します（「2.8 プログラムの実行」参照）。
トレース開始イベント／トレース終了イベントが設定されている命令が実行された際に，トレース・データの収集を開始／終了します。
なお，収集したトレース・データの確認方法についての詳細は，「2.12.6 実行履歴を表示する」を参照してください。

2.12.3.3 トレース・イベントを削除する

設定したトレース・イベントを削除するには，エディタ パネル／逆アセンブル パネルにおいて，イベント・エリア上のイベント・マークを右クリックすることで表示されるコンテキスト・メニューの［イベント削除］を選択します。
また，イベント パネルにおいて，対象となるトレース・イベントを選択したのち，ツールバーの  ボタンをクリックする操作でも削除することができます（「2.16.4 イベントを削除する」参照）。

注意　トレース・イベント内のトレース開始イベント，またはトレース終了イベントのみを削除することはできません（トレース開始イベント／トレース終了イベントのいずれかのイベント・マークを削除した場合，対応したすべてのイベント・マークが削除されます）。

2.12.4 条件を満たしたときのみの実行履歴を収集する【シミュレータ】

ある条件を満たした場合にのみプログラムの実行履歴を収集することができます。
ポイント・トレース・イベントを設定することにより，任意の変数，または I/O レジスタに対し，指定したアクセスがあった場合のみ，その情報をトレース・データとして収集します。
この機能を使用するためには，次の手順で操作を行います。

2.12.4.1 ポイント・トレース・イベントを設定する

ポイント・トレース・イベントの設定は，次のいずれかの操作により行います。

注意 1.　ポイント・トレース・イベントの設定に関しては（有効イベント数の制限など），「2.16.6 イベント設定に関する留意事項」も参照してください。

注意 2.　トレース動作中は，ポイント・トレース・イベントの設定／削除はできません。

注意 3.　DMA によるアクセスはトレース対象外です。

備考　ポイント・トレース・イベントのいずれかが有効状態で設定されると，プロパティ パネルの［デバッグ・ツール設定］タブ上の［トレース］カテゴリ内［トレース機能を使用する］プロパティの指定を自動的に［はい］に変更し，トレース機能が有効化されます。

(1)　エディタ パネル／逆アセンブル パネル上の変数 /I/O レジスタへのアクセスの場合
操作は，ソース・テキスト／逆アセンブル・テキストを表示しているエディタ パネル／逆アセンブル パネル上で行います。
各パネルにおいて，対象となる変数，または I/O レジスタを選択したのち，目的のアクセス種別に従って，コンテキスト・メニューより次の操作を行います。
ただし，対象となる変数は，グローバル変数／関数内スタティック変数／ファイル内スタティック変数のみとなります。

アクセス種別	操作方法
リード	［トレース設定］→［値をトレースに記録（読み込み時）］を選択します。
ライト	［トレース設定］→［値をトレースに記録（書き込み時）］を選択します。
リード／ライト	［トレース設定］→［値をトレースに記録（読み書き時）］を選択します。

注意　カレント・スコープ内の変数が対象となります。

(2)　登録したウォッチ式へのアクセスの場合
操作は，ウォッチ パネル上で行います。
対象となるウォッチ式を選択したのち，目的のアクセス種別に従って，コンテキスト・メニューより次の操作を行います（「2.10.6 ウォッチ式を表示／変更する」参照）。
ただし，対象となるウォッチ式は，グローバル変数／関数内スタティック変数／ファイル内スタティック変数 / I/O レジスタのみとなります。

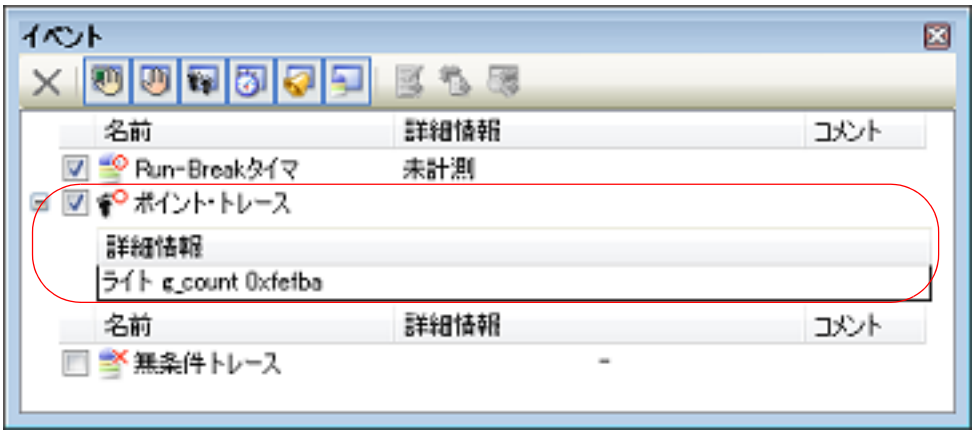
アクセス種別	操作方法
リード	［トレース出力］→［値をトレースに記録（読み込み時）］を選択します。

アクセス種別	操作方法
ライト	［トレース出力］→［値をトレースに記録（書き込み時）］を選択します。
リード／ライト	［トレース出力］→［値をトレースに記録（読み書き時）］を選択します。

注意 カレント・スコープ内のウォッチ式が対象となります。
カレント・スコープ以外のウォッチ式を対象とする場合は，スコープ指定したウォッチ式を選択してください。

以上の操作を行うことにより，対象変数，I/O レジスタ，またはウォッチ式にポイント・トレース・イベントが設定されたとみなされ，イベント パネルで管理されます（「2.16 イベントの管理」参照）。

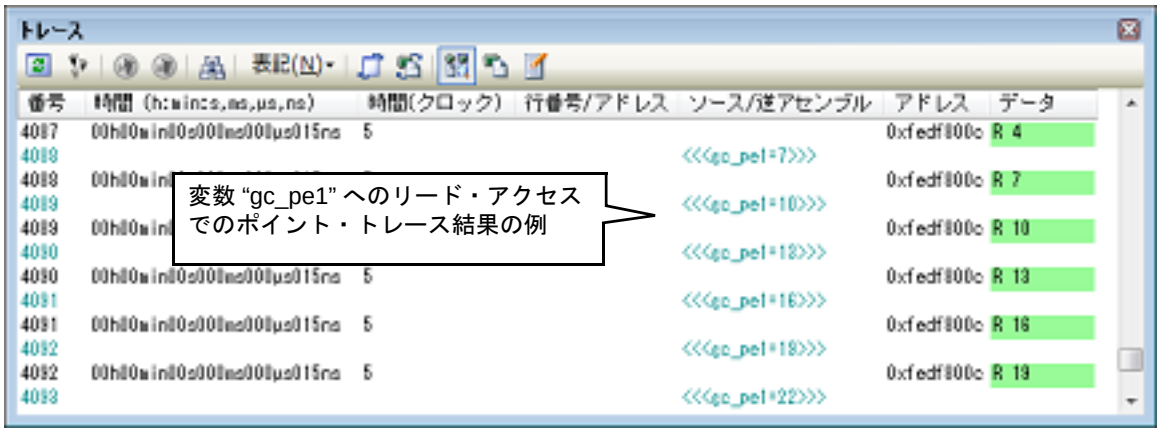
図 2.98 イベント パネルのポイント・トレース・イベントの設定例



2.12.4.2 プログラムを実行する

プログラムを実行します（「2.8 プログラムの実行」参照）。
プログラム実行中，設定したポイント・トレース・イベントの条件が満たされた場合，その情報がトレース・データとして収集されます。トレース・データの確認方法についての詳細は，「2.12.6 実行履歴を表示する」を参照してください。

図 2.99 ポイント・トレース・イベントの結果表示例



2.12.4.3 ポイント・トレース・イベントを削除する

設定したポイント・トレース・イベントを削除するには，イベント パネルにおいて，対象となるポイント・トレース・イベントを選択したのち，ツールバーの ボタンをクリックします（「2.16 イベントの管理」参照）。

2.12.5 実行履歴の収集を停止／再開する

プログラム実行中に実行履歴の収集を一時的に停止，および再開することができます。

2.12.5.1 実行履歴の収集を一時的に停止する

プログラムの実行を停止することなく，トレーサの動作のみを停止するには，トレース パネルのツールバーの  ボタンをクリックします。
プログラムを停止せずにトレース機能のみを停止させ，その時点までのトレース・データを確認する場合などに使用します。

2.12.5.2 実行履歴の収集を再開する

プログラム実行中にトレース機能を停止したのち，再度トレース・データの収集を開始するには，トレース パネルのツールバーの  ボタンをクリックします。
なお，再開前に収集したトレース・データは一度クリアされます。

2.12.6 実行履歴を表示する

収集したトレース・データの表示は，次のトレース パネルで行います。
［表示］メニュー→［トレース］を選択してください。
トレース・データは，デフォルトでソース・テキストと逆アセンブル・テキストを混合して表示しますが，表示モードを選択することにより，そのどちらか一方のみを表示させることもできます。
なお，各エリアの見方，および機能についての詳細は，トレース パネルの項を参照してください。

図 2.100 トレース・データの表示（トレース パネル【Full-spec emulator】【E1】【E20】）

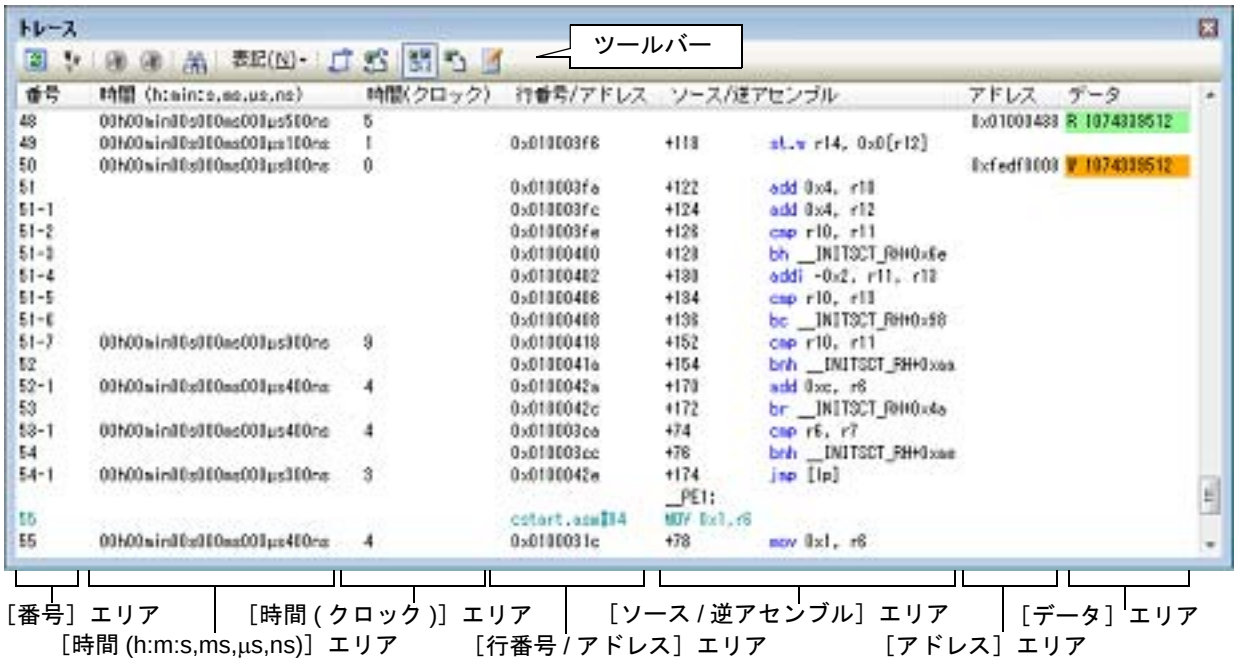


図 2.101 トレース・データの表示（トレース パネル【シミュレータ】）

ここでは，次の操作方法について説明します。

2.12.6.1 表示モードを変更する
2.12.6.2 値の表示形式を変更する
2.12.6.3 他のパネルと連動させる

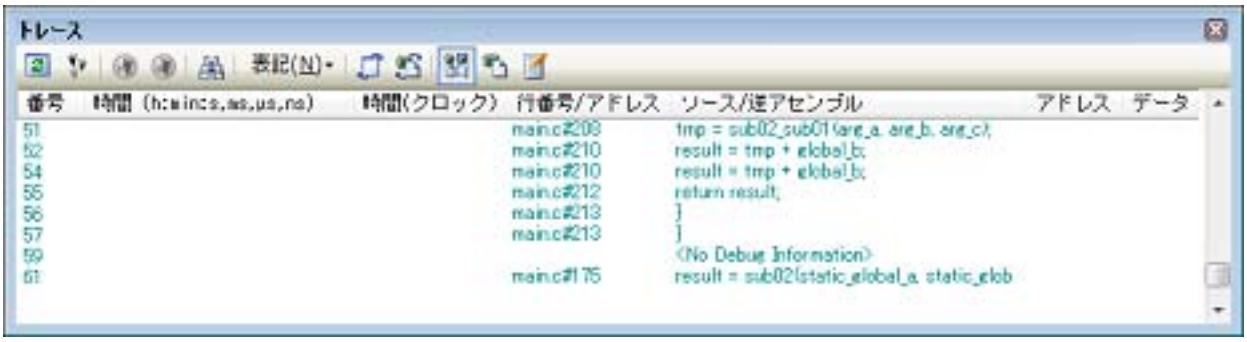
2.12.6.1 表示モードを変更する

次のツールバーのボタンをクリックすることで，用途に応じて表示モードを変更することができます。
ただし，トレーサが動作中の場合は無効となります。

表 2.12 トレース パネルの表示モード

ボタン	表示モード	表示内容
	混合表示モード	命令（逆アセンブル）／ラベル名／ソース・テキスト（対応するソース行）／ポイント・トレース結果／ブレーク要因を表示します（デフォルト）。
	逆アセンブル表示モード	命令（逆アセンブル）／ラベル名／ポイント・トレース結果／ブレーク要因を表示します。
	ソース表示モード	ソース・テキスト（対応するソース行）／ブレーク要因を表示します。 ただし，デバッグ情報が存在しない箇所を実行した場合は，“ デバッグ情報のない区間の実行 ” と表示します。

図 2.102 ソース表示モードの例（トレース パネル）



2.12.6.2 値の表示形式を変更する

ツールバーの次のボタンにより，このパネルの［行番号 / アドレス］エリア／［アドレス］エリア／［データ］エリアの表示形式を変更することができます。
ただし，トレーサが動作中の場合は無効となります。

表記	値の表示形式を変更する次のボタンを表示します。
	このパネル上の値を 16 進数で表示します（デフォルト）。
	このパネル上の値を 10 進数で表示します。
	このパネル上の値を 8 進数で表示します。
	このパネル上の値を 2 進数で表示します。

2.12.6.3 他のパネルと連動させる

現在選択している行のアドレスをポインタとして，他のパネルで対応箇所を連動して表示させることができます（フォーカスの移動は行いません）。
ツールバーの  ボタンをクリックすると，エディタ パネルと連動開始します。
ツールバーの  ボタンをクリックすると，逆アセンブル パネルと連動開始します。
なお，再度クリックすることにより，連動を中止します。

備考 コンテキスト・メニューの［ソースへジャンプ］／［逆アセンブルへジャンプ］を選択することにより，現在選択している行のアドレスに対応するソース行／アドレスにキャレットを移動した状態で，エディタ パネル／逆アセンブル パネルがオープンします（フォーカスの移動を行います）。

2.12.7 トレース・メモリをクリアする

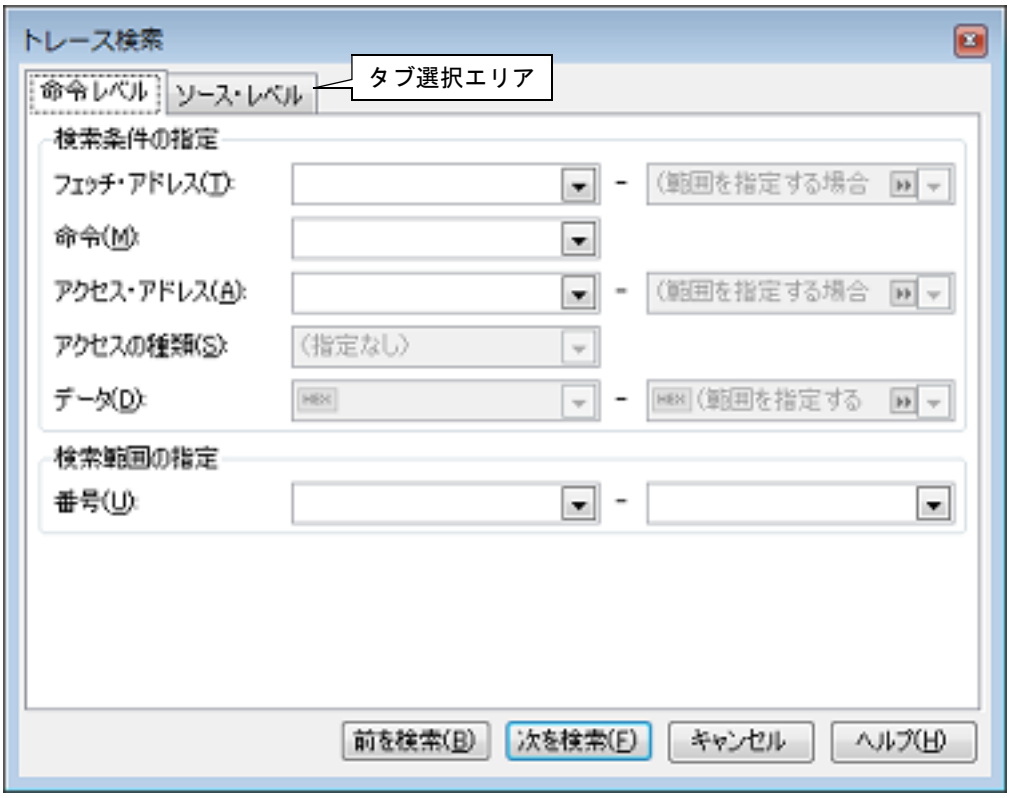
収集したトレース・データの内容をクリアするには，ツールバーの  ボタンをクリックします。
ただし，トレーサが動作中は無効となります。

備考 プロパティ パネルの［デバッグ・ツール設定］タブ上の［トレース］カテゴリ内の［実行前にトレース・メモリをクリアする］プロパティにおいて，［はい］を選択している場合は，プログラムの実行ごとにトレース・メモリがクリアされます。

2.12.8 トレース・データを検索する

収集したトレース・データの検索は，ツールバーの  ボタンをクリックすることによりオープンするトレース検索ダイアログにより行います（プログラム実行中は無効）。
このダイアログにおいて，次の操作を行ってください。
なお，タブ選択エリアのタブを選択することにより，命令レベル，またはソース・レベルでトレース・データを検索することができます。

図 2.103　　トレース・データの検索（トレース検索 ダイアログ）



ここでは，次の操作方法について説明します。

2.12.8.1 命令レベルで検索する
2.12.8.2 ソース・レベルで検索する

2.12.8.1 命令レベルで検索する

命令レベルでトレース・データを検索します。
［命令レベル］タブを選択したのち，次の手順で操作を行ってください。

注意　　命令レベルでトレース・データの検索を行う場合は，トレース パネルを混合表示モード，または逆アセンブル表示モードで表示している必要があります。

図 2.104 命令レベルでのトレース・データの検索

(1) ［フェッチ・アドレス］の指定
検索条件として必要な場合，フェッチ・アドレスを指定します。
アドレス式をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
フェッチ・アドレスの指定は範囲で指定することができます。この場合は，左右両方のテキスト・ボックスにアドレス値を指定することにより範囲を指定します。
右側のテキスト・ボックスが空欄，または“（範囲を指定する場合に入力）”の場合は，左側のテキスト・ボックスに指定された固定アドレスで検索を行います。
なお，マイクロコントローラのアドレス空間よりも大きいアドレス値が指定された場合は，上位のアドレス値をマスクして扱います。
また，32 ビットで表現できる値より大きいアドレス値を指定することはできません。

(2) ［命令］の指定
検索条件として必要な場合，命令の文字列を指定します。
ここで指定した文字列をトレース パネルの［ソース / 逆アセンブル］エリア内より検索します。
命令をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
なお，検索の際は，大文字／小文字は区別せず，部分一致も検索の対象とします。

(3) ［アクセス・アドレス］の指定
検索条件として必要な場合，アクセス・アドレスを指定します。
16 進数でアドレス値をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
アクセス・アドレスの指定は範囲で指定することができます。この場合は，左右両方のテキスト・ボックスにアドレス値を指定することにより範囲を指定します。
右側のテキスト・ボックスが空欄，または“（範囲を指定する場合に入力）”の場合は，左側のテキスト・ボックスに指定された固定アドレスで検索を行います。
なお，マイクロコントローラのアドレス空間よりも大きいアドレス値が指定された場合は，上位のアドレス値をマスクして扱います。
また，32 ビットで表現できる値より大きいアドレス値を指定することはできません。

(4) ［アクセスの種類］の指定
この項目は［アクセス・アドレス］の指定が指定された場合のみ有効となります。
アクセスの種類（リード／ライト，リード，ライト，ベクタ・リード，DMA）をドロップダウン・リストより選択します。
アクセスの種類を限定しない場合は，［(指定なし)］を選択してください。

(5) ［データ］の指定
この項目は［アクセス・アドレス］の指定が指定された場合のみ有効となります。
アクセスした数値を指定します。
16 進数値をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
数値の指定は範囲で指定することができます。この場合は，左右両方のテキスト・ボックスにデータを指定することにより範囲を指定します。
右側のテキスト・ボックスが空欄，または“（範囲を指定する場合に入力）”の場合は，左側のテキスト・ボックスに指定された固定数値で検索を行います。

(6) ［番号］の指定
検索するトレース・データの範囲を，トレース パネルの［番号］エリアに表示されている番号で指定します。
左右のテキスト・ボックスに，それぞれ開始番号と終了番号を指定します（デフォルトでは，“0”～“*最終番号*”が指定されます）。
10 進数で番号をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
左側のテキスト・ボックスが空欄の場合は，“0”の指定として扱われます。
右側のテキスト・ボックスが空欄の場合は，最終番号の指定として扱われます。

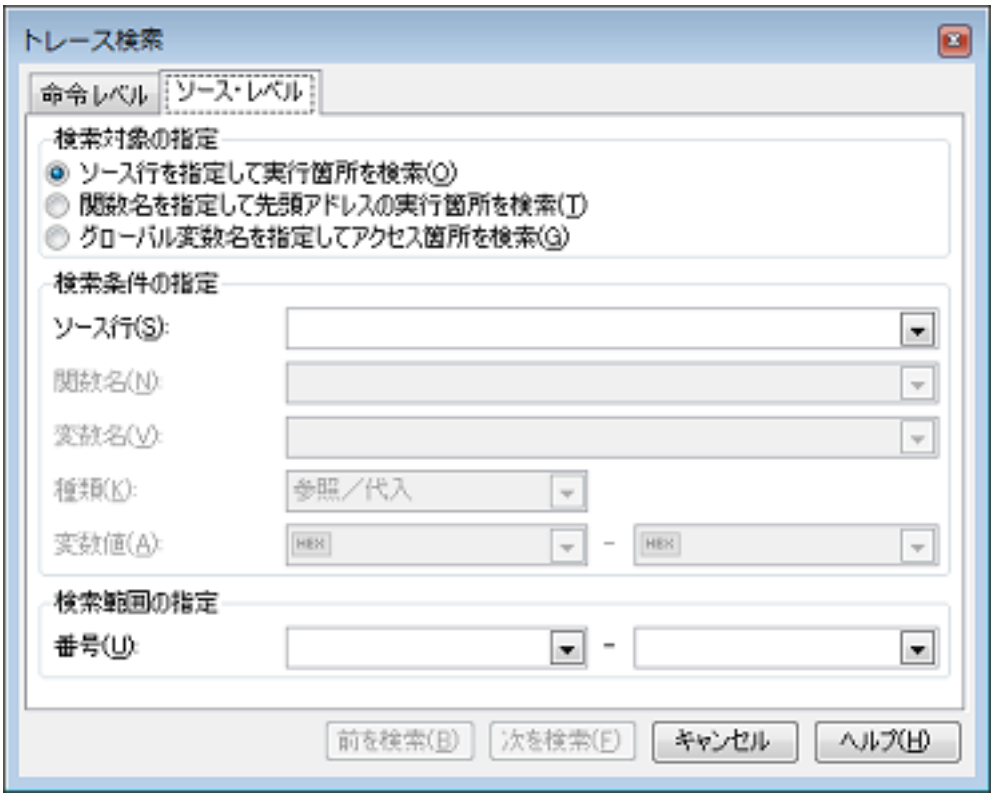
(7) ［前を検索］／［次を検索］ボタンのクリック
［前を検索］ボタンをクリックすると，番号の小さい方向に検索を行い，検索結果箇所をトレース パネル上で選択状態にします。
［次を検索］ボタンをクリックすると，番号の大きい方向に検索を行い，検索結果箇所をトレース パネル上で選択状態にします。

2.12.8.2 ソース・レベルで検索する

ソース・レベルでトレース・データを検索します。
［ソース・レベル］タブを選択してください。

注意 ソース・レベルで検索を行う場合は，トレース パネルを混合表示モード，またはソース表示モードで表示している必要があります。

図 2.105 ソース・レベルでのトレース・データの検索



(1) ソース行を指定して検索する場合（デフォルト）

［検索対象の指定］エリアにおいて，“ ソース行を指定して実行箇所を検索 ” を選択したのち，次の操作を行います。

- ［ソース行］の指定
  ここで指定した文字列をトレース パネルの［行番号 / アドレス］エリア内より検索します。
  検索するソース行に含まれる文字列を，テキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
  なお，検索の際は，大文字／小文字は区別せず，部分一致も検索の対象とします。

  例 1.　main.c#40

  例 2.　main.c

  例 3.　main

- ［番号］の指定
  検索するトレース・データの範囲を，トレース パネルの［番号］エリアに表示されている番号で指定します。
  左右のテキスト・ボックスに，それぞれ開始番号と終了番号を指定します（デフォルトでは，“0” ～ “ *最終番号* ” が指定されます）。
  10 進数で番号をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
  左側のテキスト・ボックスが空欄の場合は，“0” の指定として扱われます。
  右側のテキスト・ボックスが空欄の場合は，最終番号の指定として扱われます。

- ［前を検索］／［次を検索］ボタンのクリック
  ［前を検索］ボタンをクリックすると，番号の小さい方向に検索を行い，検索結果箇所をトレース パネル上で選択状態にします。
  ［次を検索］ボタンをクリックすると，番号の大きい方向に検索を行い，検索結果箇所をトレース パネル上で選択状態にします。

(2) 関数名を指定して検索する場合
［検索対象の指定］エリアにおいて，“ 関数名を指定して先頭アドレスの実行箇所を検索 ” を選択したのち，次の操作を行います。

- ［関数名］の指定
  検索する関数名を，テキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
  なお，検索の際は，大文字／小文字を区別し，完全一致のみを検索の対象とします。

- ［番号］の指定
  検索するトレース・データの範囲を，トレース パネルの［番号］エリアに表示されている番号で指定します。
  左右のテキスト・ボックスに，それぞれ開始番号と終了番号を指定します（デフォルトでは，“0” ～ “ *最終番号* ” が指定されます）。
  10 進数で番号をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
  左側のテキスト・ボックスが空欄の場合は，“0” の指定として扱われます。
  右側のテキスト・ボックスが空欄の場合は，最終番号の指定として扱われます。

- ［前を検索］／［次を検索］ボタンのクリック
  ［前を検索］ボタンをクリックすると，番号の小さい方向に検索を行い，検索結果箇所をトレース パネル上で選択状態にします。
  ［次を検索］ボタンをクリックすると，番号の大きい方向に検索を行い，検索結果箇所をトレース パネル上で選択状態にします。

(3) グローバル変数名を指定して検索する場合
［検索対象の指定］エリアにおいて，“ グローバル変数名を指定してアクセス箇所を検索 ” を選択したのち，次の操作を行います。

- ［変数名］の指定
  検索する変数名を，テキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
  なお，検索の際は，大文字／小文字を区別し，完全一致のみを検索の対象とします。

- ［種類］の指定
  アクセスの種類（参照／代入（デフォルト），参照，代入）をドロップダウン・リストより選択します。

- ［変数値］の指定
  アクセスした変数値をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。

変数値の指定は範囲で指定することができます。この場合は，左右両方のテキスト・ボックスに変数値を指定することにより範囲を指定します。
右側のテキスト・ボックスが空欄の場合は，左側のテキスト・ボックスに指定された固定変数値でアクセス箇所を検索を行います。

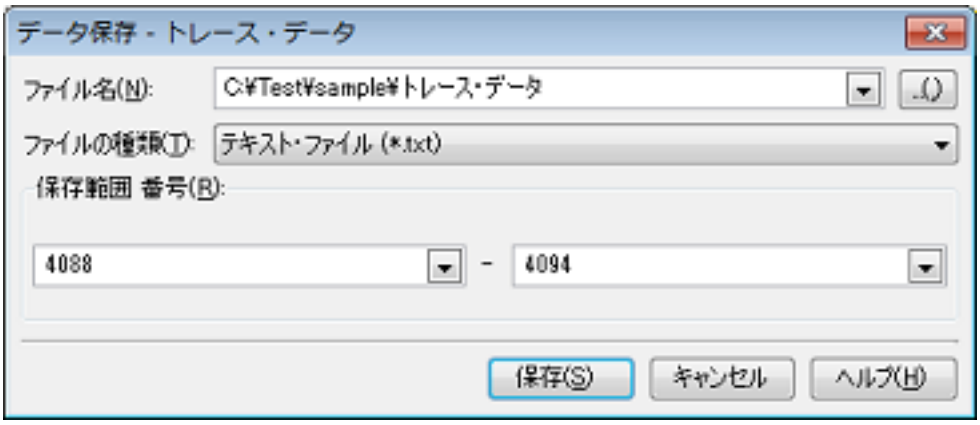
- ［番号］の指定
検索するトレース・データの範囲を，トレース パネルの［番号］エリアに表示されている番号で指定します。
左右のテキスト・ボックスに，それぞれ開始番号と終了番号を指定します（デフォルトでは，“0” ～ “ *最終番号* ” が指定されます）。
10 進数で番号をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
左側のテキスト・ボックスが空欄の場合は，“0” の指定として扱われます。
右側のテキスト・ボックスが空欄の場合は，最終番号の指定として扱われます。

- ［前を検索］／［次を検索］ボタンのクリック
［前を検索］ボタンをクリックすると，番号の小さい方向に検索を行い，検索結果箇所をトレース パネル上で選択状態にします。
［次を検索］ボタンをクリックすると，番号の大きい方向に検索を行い，検索結果箇所をトレース パネル上で選択状態にします。

2.12.9 実行履歴の表示内容を保存する

収集したトレース・データの内容を範囲指定して，テキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。ファイルに保存する際は，デバッグ・ツールから最新の情報を取得し，このパネル上での表示形式に従ったデータで保存します。
［ファイル］メニュー→［名前を付けてトレース・データを保存 ...］を選択すると，次のデータ保存 ダイアログがオープンします。
このダイアログにおいて，次の手順で操作を行ってください。

図 2.106 実行履歴の保存（データ保存 ダイアログ）



(1) ［ファイル名］の指定
保存するファイル名を指定します。
テキスト・ボックスに直接入力するか（最大指定文字数：259 文字），またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
また，［...］ボタンをクリックすることでオープンするデータ保存ファイルを選択 ダイアログにより，ファイルを選択することもできます。

(2) ［ファイルの種類］の指定
保存するファイルの形式を次のプドロップダウン・リストにより選択します。
選択できるファイルの形式は次のとおりです。

リスト表示	形式
テキスト・ファイル (*.txt)	テキスト形式（デフォルト）
CSV(カンマ区切り) (*.csv)	CSV 形式[注]

注　　各データを “,” で区切り保存します。
なお，データ内に “,” が含まれている際の不正形式を避けるため，各データを ""（ダブルクォーテーション）で括り出力します。

(3)　［保存範囲 番号］の指定
ファイルに保存する範囲を “ 開始トレース番号 ” と “ 終了トレース番号 ” で指定します。
それぞれのテキスト・ボックスに 10 進数の数値を直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
なお，すべてのトレース・データを保存する場合は，左側のドロップダウン・リストにおいて，［すべてのトレース・データ］を選択してください（右側のテキスト・ボックスは無効）。
パネル上で範囲選択している場合は，デフォルトでその選択範囲がテキスト・ボックスに指定されます。範囲選択していない場合は，現在のパネルの表示範囲が指定されます。

(4)　［保存］ボタンのクリック
指定したファイルに，指定した形式でトレース・データを保存します。

図 2.107　　トレース・データ保存の際の出力イメージ

```
番号　ターゲット　時間　クロック　行番号 / アドレス　パイプライン　ソース / 逆アセンブル　アドレス　データ
-------------------------------------------------------------------------------
番号　ターゲット　時間　クロック　行番号 / アドレス　パイプライン　ソース / 逆アセンブル　アドレス　データ
 :　　:　　:　　:　　:　　:　　:　　:　　:
```

備考　　出力されるトレース・データの項目は，使用しているデバッグ・ツールにより異なります。

2.13 実行時間の計測

この節では，プログラムの実行時間の計測方法について説明します。

備考　　マルチコア対応版を対象とした“実行時間の計測”については，「2.7 コア（PE）の選択」も参照してください。

2.13.1 実行停止までの実行時間を計測する

デバッグ・ツールには，プログラムの実行開始から実行停止までの実行時間（Run-Break 時間）を計測する機能があらかじめ用意されています。
したがって，プログラムの実行を開始することにより，自動的に実行時間の計測を行います。
計測結果は，次のいずれかの方法で確認することができます。

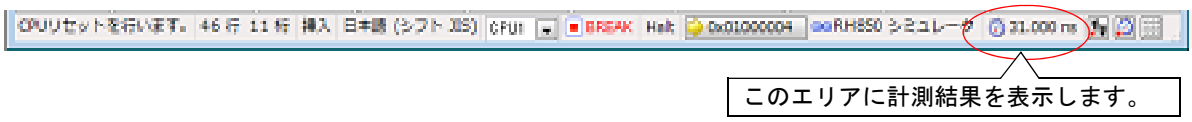
注意 1.　　ステップ実効中は正しい実行時間が表示されません。

注意 2.　　【シミュレータ】
Run-Break 時間を計測するためには，プロパティ パネルの［デバッグ・ツール設定］タブ上の［タイマ］カテゴリ内［タイマ機能を使用する］プロパティにおいて，［はい］が指定されている必要があります。

備考　　この機能は，デバッグ・ツールにデフォルトで設定されているビルトイン・イベントの 1 つである Run-Break タイマ・イベントにより動作します。

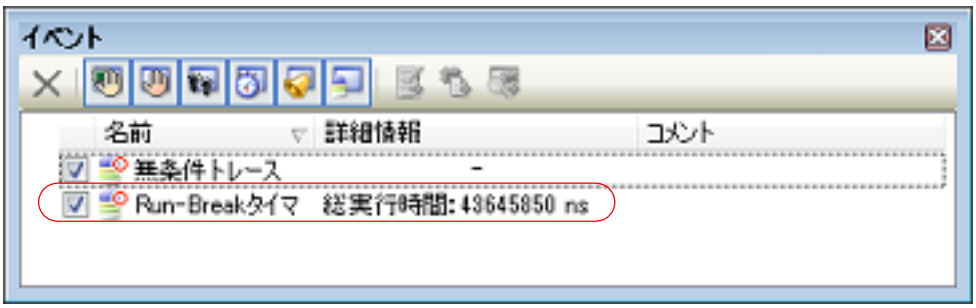
(1)　ステータスバーでの確認
プログラムの実行停止後，メイン・ウインドウ上のステータスバーにおいて計測結果を表示します（計測をしていない場合は“未計測”と表示）。

図 2.108　　Run-Break タイマ・イベントの測定結果例（ステータスバー）



(2)　イベント パネルでの確認
プログラムの実行停止後，［表示］メニュー→［イベント］の選択によりオープンするイベント パネル上において，Run-Break タイマ・イベントとして計測結果を表示します。

図 2.109　　Run-Break タイマ・イベントの測定結果例（イベント パネル）



2.13.2 任意区間の実行時間を計測する【シミュレータ】

タイマ計測イベントを設定することにより，プログラムの実行過程において，任意の区間の実行時間を計測することができます。
なお，タイマ計測イベントは，タイマ開始イベント／タイマ終了イベントで構成されます。
この機能を使用するためには，次の手順で操作を行います。

2.13.2.1 タイマ計測イベントを設定する
2.13.2.2 プログラムを実行する
2.13.2.3 タイマ計測イベントを削除する

注意 1.　　タイマ計測イベントの設定に関しては（有効イベント数の制限など），「2.16.6 イベント設定に関する留意事項」も参照してください。

注意 2.　タイマ機能を使用するためには，プロパティ パネルの［デバッグ・ツール設定］タブ上の［タイマ］カテゴリ内［タイマ機能を使用する］プロパティにおいて，［はい］が指定されている必要があります。

2.13.2.1 タイマ計測イベントを設定する

タイマ計測イベントを設定するため，タイマ計測を開始／終了するタイマ開始イベント／タイマ終了イベントを設定します。
タイマ開始イベント／タイマ終了イベントの設定は，次のいずれかの操作により行います。

(1)　実行系イベントの場合
実行系イベントをタイマ開始イベント／タイマ終了イベントに設定することにより，任意の区間の実行時間を計測することができます。
操作は，ソース・テキスト／逆アセンブル・テキストを表示しているエディタ パネル／逆アセンブル パネルで行います。
各パネルのアドレス表示のある行にキャレットを移動したのち，目的のイベント種別に従って，コンテキスト・メニューより次の操作を行います。

イベント種別	操作方法
タイマ開始	［タイマ設定］→［実行時にタイマ開始］→［タイマ *n* に設定[注]］
タイマ終了	［タイマ設定］→［実行時にタイマ終了］→［タイマ *n* に設定[注]］

注　　タイマ・イベントを設定するチャネル番号（n：1 ～ 8）を選択します。

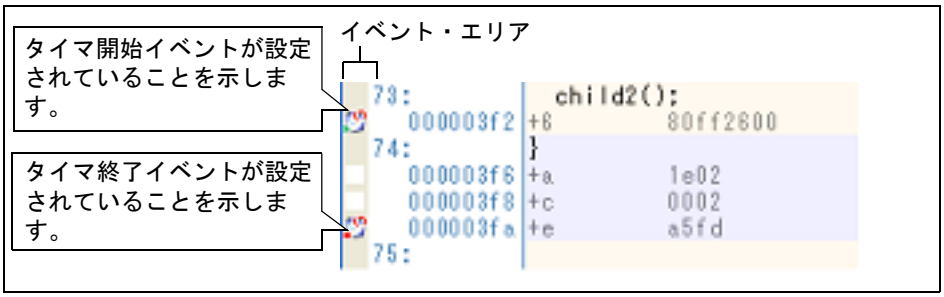
注意　　タイマ終了イベントは時間測定結果に含まれません。
時間測定結果に含める場合は，1 行下にタイマ終了イベントを設定してください。

タイマ開始イベント／タイマ終了イベントは，キャレット位置の行に対応する先頭アドレスの命令に設定されます。
タイマ開始イベント／タイマ終了イベントが設定されると，設定した行のイベント・エリアに次のイベント・マークが表示されます。

表 2.13　　タイマ開始イベント／タイマ終了のイベント・マーク

イベント種別	イベント・マーク
タイマ開始	
タイマ終了	

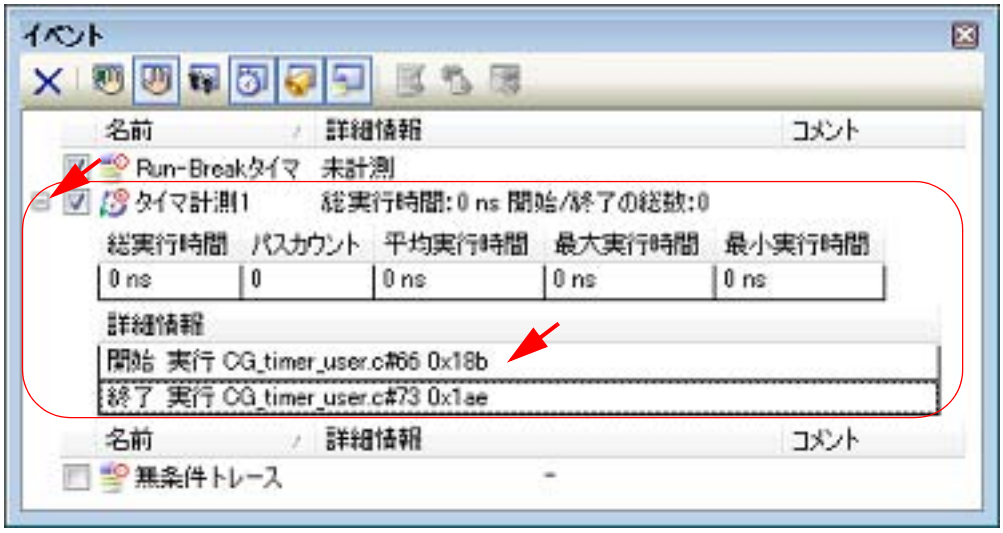
図 2.110　　タイマ開始イベント／タイマ終了イベントの設定例（逆アセンブル パネルの場合）



(2)　アクセス系イベントの場合

タイマ開始イベント／タイマ終了イベントが設定されると，イベント パネル上において，タイマ計測イベントとして 1 つにまとめて管理され（「2.16 イベントの管理」参照），タイマ計測イベント項目の “+” マークをクリックすることにより，設定したタイマ開始イベント／タイマ終了イベントの詳細情報が表示されます。

図 2.111 イベント パネルのタイマ開始イベント／タイマ終了イベント（実行系）の設定例

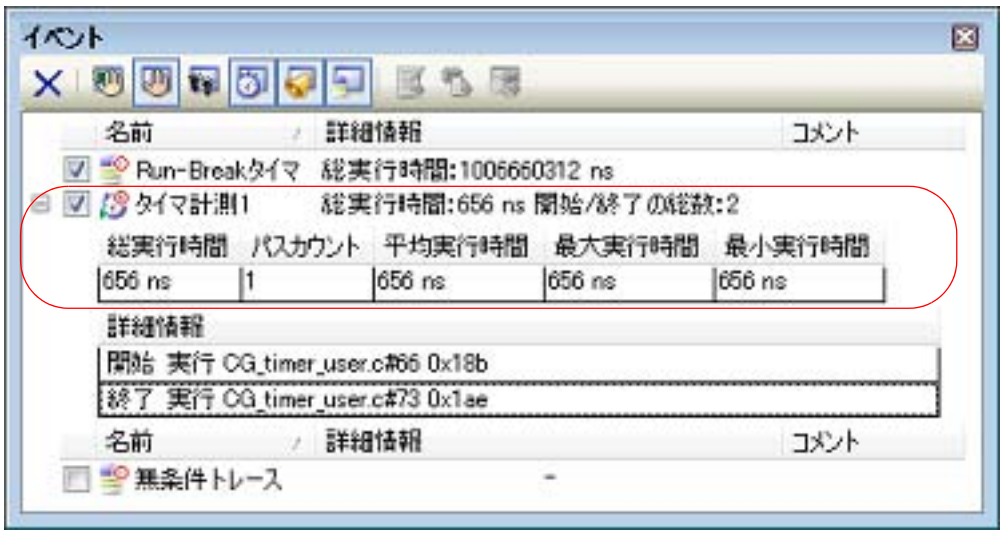


備考 イベントの設定状態によりイベント・マークは異なります（「2.16.1 設定状態（有効／無効）を変更する」参照）。
また，すでにイベントが設定されている箇所で，新たにイベントを設定した場合は，複数のイベントが設定されていることを示すイベント・マーク（ ）が表示されます。

2.13.2.2 プログラムを実行する

プログラムを実行します（「2.8 プログラムの実行」参照）。
タイマ開始イベント／タイマ終了イベントが設定されている命令が実行された際に，タイマ計測を開始／終了します。
計測結果は，プログラムの実行停止後，［表示］メニュー→［イベント］の選択によりオープンするイベント パネルにおいて，タイマ計測イベントして次のように確認することができます。
なお，このタイマ計測イベントは，タイマ開始イベント，またはタイマ終了イベントのいずれかが設定された場合に，イベント パネルでのみ表示されるイベント種別です。

図 2.112 タイマ計測イベント（タイマ開始イベント／タイマ終了イベント）の測定結果例



2.13.2.3 タイマ計測イベントを削除する

設定したタイマ計測イベントを削除するには，エディタ パネル／逆アセンブル パネルにおいて，イベント・エリア上のイベント・マークを右クリックすることで表示されるコンテキスト・メニューの［イベント削除］を選択します。
また，イベント パネルにおいて，対象となるタイマ計測イベント選択したのち，ツールバーの ボタンをクリックする操作でも削除することができます（「2.16.4 イベントを削除する」参照）。

注意　　タイマ計測イベント内のタイマ開始イベント，またはタイマ終了イベントのみを削除することはできません（タイマ開始イベント／タイマ終了イベントのいずれかのイベント・マークを削除した場合，対応したすべてのイベント・マークが削除されます）。

2.13.3 測定可能時間の範囲

Run-Break タイマ・イベント（「2.13.1 実行停止までの実行時間を計測する」参照），またはタイマ計測イベント（「2.13.2 任意区間の実行時間を計測する【シミュレータ】」参照）によるタイマ計測の測定可能時間の範囲は次のとおりです。

表 2.14　　測定可能時間の範囲

デバッグ・ツール	Run-Break タイマ・イベント		タイマ計測イベント
Full-spec emulator	最小	約 60 ナノ秒	－
	最大	約 4 分 20 秒（LPD 4Pin 使用時） オーバフロー検出あり	
E1/E20	最小	約 60 ナノ秒	－
	最大	約 4 分 20 秒（LPD 4Pin 使用時） オーバフロー検出あり	
シミュレータ	CPU クロック周波数に依存		CPU クロック周波数に依存

2.14 カバレッジの測定【シミュレータ】

この節では，カバレッジ機能を使用した，カバレッジ測定について説明します。
カバレッジ測定の方法にはいくつかの種類がありますが，CS+ では次の領域を対象に，ソース行／関数に対するフェッチ系のコード・カバレッジ測定（C0 カバレッジ），および変数に対するアクセス系のデータ・カバレッジ測定を行います。
CS+ では，次の両領域がカバレッジ測定の対象となります。

- 内蔵 ROM 領域（アドレス 0x000000 ～ 0x0FFFFF）の 1M バイト空間（固定の測定領域）
- 上記の固定測定領域以外の任意の 1M バイト空間（「2.14.1 カバレッジ測定の設定をする」参照）

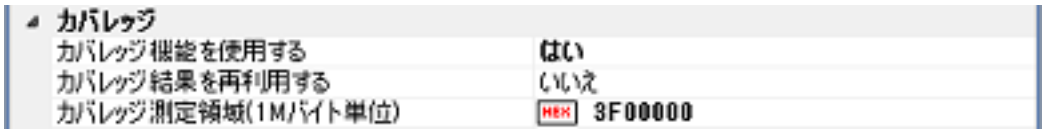
備考 1.　C0 カバレッジ：命令網羅率（ステートメント・カバレッジ）
たとえば，コード内のすべての命令（ステートメント）を 少なくとも 1 回は実行した場合，C0 = 100 % となります。

備考 2.　マルチコア対応版を対象とした“カバレッジの測定”については，「2.7 コア（PE）の選択」も参照してください。

2.14.1 カバレッジ測定の設定をする

カバレッジ機能を使用するためには，あらかじめカバレッジ測定に関する設定を行う必要があります。
設定は，プロパティ パネルの［デバッグ・ツール設定］タブ上の［カバレッジ］カテゴリ内で行います。

図 2.113　［カバレッジ］カテゴリ



(1)　［カバレッジ機能を使用する］
カバレッジ機能を使用するか否かを選択します。
カバレッジ機能を使用する場合は［はい］を選択してください（デフォルト：［いいえ］）。

(2)　［カバレッジ結果を再利用する］
このプロパティは，［カバレッジ機能を使用する］プロパティにおいて［はい］を選択した場合のみ表示されます。
デバッグ・ツールと切断時に，現在取得しているコード・カバレッジ測定結果を自動保存し，次回デバッグ・ツールと接続した際に，保存した測定結果の内容を再現するか否かを選択します。
前回取得したコード・カバレッジ測定結果の内容を再現する場合は，［はい］を選択してください（デフォルト：［いいえ］）。

注意　内蔵 ROM 領域のみがこの機能の対象となります。

(3)　［カバレッジ測定領域 (1M バイト単位)］
このプロパティは，［カバレッジ機能を使用する］プロパティにおいて［はい］を選択した場合のみ表示されます。
コード・カバレッジ測定の対象領域を指定します。
カバレッジ測定を行う内蔵 ROM 領域（0x000000 - 0x0FFFFF）以外の任意の 1M バイト空間の開始アドレスを，直接入力により 16 進数で指定してください（デフォルト：［100000］）。

2.14.2 カバレッジ測定結果を表示する

プログラムの実行が開始すると自動的にカバレッジ測定が開始し，実行停止とともにカバレッジ測定も終了します。

(1)　コード・カバレッジ率

(a)　ソース行／逆アセンブル行に対するコード・カバレッジ率の表示
カバレッジ測定結果の表示は，対象となるプログラムを表示しているエディタ パネル／逆アセンブル パネルで行われます。
各パネルでは，表 2.15 に示す計算方法で算出されたコード・カバレッジ率を基に，対象ソース・テキスト行／逆アセンブル結果行の背景色が表 2.16 のように色分け表示されます。
ただし，デバッグ・ツールと切断時，またはプログラム実行中は，結果の表示を行いません。

なお，取得したコード・カバレッジ測定結果は，エディタ パネル／逆アセンブル パネル上のコンテキスト・メニューの［カバレッジ情報のクリア］を選択することにより，すべてリセットすることができます（各パネル上の色分け表示もクリアされます）。

注意 マルチコア対応版を対象としている場合，Local RAM self 領域は，現在選択している PE（「2.7 コア（PE）の選択」参照）のアクセスのみを対象に測定結果を表示します。

表 2.15 ソース行／逆アセンブル行に対するコード・カバレッジ率の計算方法

パネル	計算方法
エディタ パネル	“ ソース行と対応するアドレス範囲内で実行されたバイト数 ”÷“ ソース行と対応するアドレス範囲内の総バイト数 ”
逆アセンブル パネル	“ 逆アセンブル結果行と対応するアドレス範囲内で実行されたバイト数 ” ÷“ 逆アセンブル結果行と対応するアドレス範囲内の総バイト数 ”

表 2.16 コード・カバレッジ測定結果の表示色（デフォルト）

コード・カバレッジ率	背景色
100 %	*ソース・テキスト／逆アセンブル結果*
1 ～ 99 %	*ソース・テキスト／逆アセンブル結果*
0 %（未実行）	*ソース・テキスト／逆アセンブル結果*

備考 1. 各パネルにおけるコード・カバレッジ測定結果の表示更新は，プログラム停止ごとに自動的に行われます。

備考 2. 上記の背景色は，オプション ダイアログにおける［全般 - フォントと色］カテゴリの設定に依存します。

備考 3. 上記の背景表示は，対象領域外の行に対しては行われません。

備考 4. ダウンロードしているロード・モジュールの更新日時より，現在オープンしているソース・ファイルの更新日時が新しい場合，エディタ パネルではコード・カバレッジ測定結果の表示は行われません。

図 2.114 コード・カバレッジ測定結果の表示例（エディタ パネル）

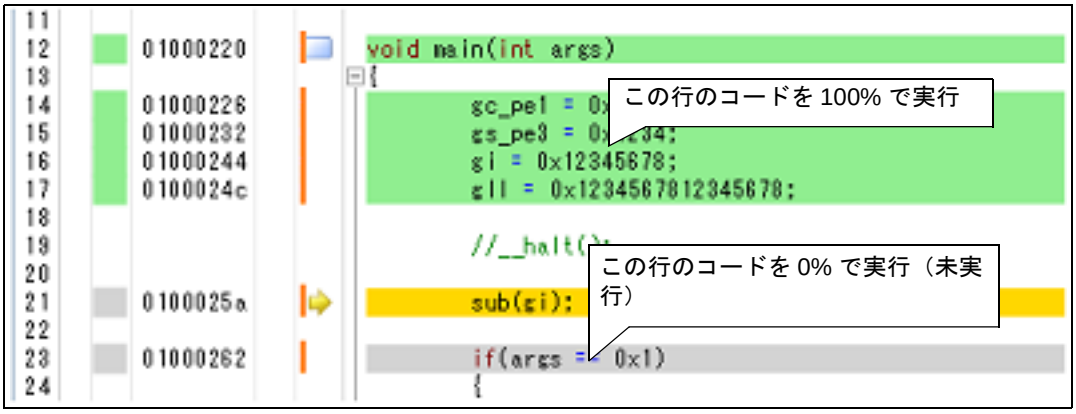
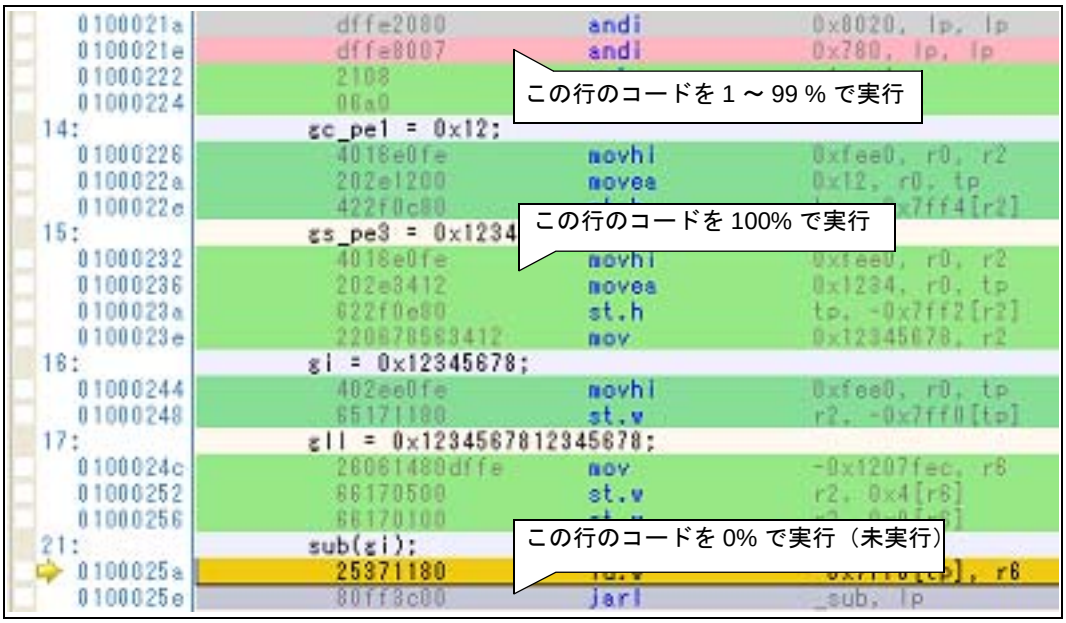


図 2.115 コード・カバレッジ測定結果の表示例（逆アセンブル パネル）



(b) 各関数に対するコード・カバレッジ率の表示
各関数に対するコード・カバレッジ率（関数の網羅率）は，解析ツールの関数 パネル内［コード・カバレッジ］項目で確認することができます。
“ 関数のコード・カバレッジ率 ” についての詳細は，「CS+ 統合開発環境 ユーザーズマニュアル 解析編」を参照してください。

(2) データ・カバレッジ率
各変数に対するデータ・カバレッジ率は，解析ツールの変数 パネル内［データ・カバレッジ］項目で確認することができます。
“ 変数のデータ・カバレッジ率 ” についての詳細は，「CS+ 統合開発環境 ユーザーズマニュアル 解析編」を参照してください。

2.15 プログラム内へのアクションの設定

この節では，プログラム内に，指定したアクションを設定する操作方法について説明します。

2.15.1 printf を挿入する

アクション・イベントの 1 つである Printf イベントを設定することにより，プログラムの実行を任意の箇所で一瞬停止させたのち，ソフトウエア処理により printf コマンドを実行させ，指定した変数式の値を出力 パネルに出力することができます。
この機能を使用するためには，次の手順で操作を行ってください。

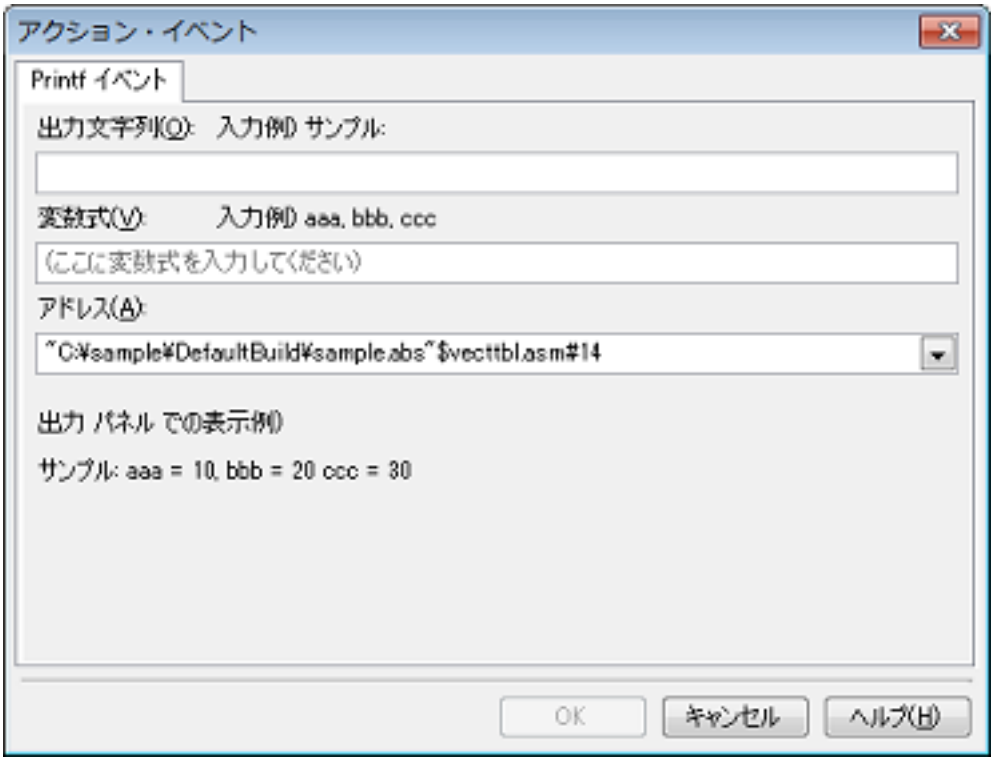
注意 1. 【Full-spec emulator】【E1】【E20】
Printf イベントはソフトウエア・ブレーク機能【Full-spec emulator】【E1】【E20】により実現されます。そのため，Printf イベントを設定するためには，あらかじめ，プロパティ パネルの［デバッグ・ツール設定］タブの［ブレーク］カテゴリ内［ソフトウエア・ブレークを使用する］プロパティで［はい］を選択してください。

注意 2. Printf イベントの設定に関しては（有効イベント数の制限など），「2.16.6 イベント設定に関する留意事項」も参照してください。

注意 3. ステップ実行中（  ），またはブレーク関連のイベントを無視したプログラム実行中（  ）にアクション・イベントは発生しません。

(1) Printf イベントを設定する
エディタ パネル／逆アセンブル パネル上で，printf コマンドを実行させたい箇所に Printf イベントを設定します。
各パネルのアドレス表示のある行にキャレットを移動したのち，コンテキスト・メニューの［アクション・イベントの登録 ...］を選択すると，次のアクション・イベント ダイアログがオープンします。
このダイアログにおいて，次の操作を行ってください。

図 2.116 Printf イベントを設定する（アクション・イベント ダイアログ：［Printf イベント］タブ）



(a) ［出力文字列］の指定
出力 パネルに出力する際に付与する文字列をキーボードより直接入力で指定します。
なお，出力する文字列は，1 行分のみ入力可能です（空白可）。

(b) ［変数式］の指定
Printf イベントの対象となる変数式を指定します。
変数式は，テキスト・ボックスに直接入力で指定します（最大指定文字数：1024 文字）。

“,” で区切ることにより，1 つの Printf イベントとして 10 個までの変数式を指定することができます。
エディタ パネル／逆アセンブル パネルにおいて，変数式を選択した状態でこのダイアログをオープンした場合では，選択している変数式がデフォルトで表示されます。
なお，変数式として指定できる基本入力形式と，その際に Printf イベントとして出力される値についての詳細は，「表 A.11 変数式と出力される値の関係（Printf イベント）」を参照してください。

備考 このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

(c) ［アドレス］の指定
Printf イベントを設定するアドレスを指定します。
デフォルトで，現在の指定位置のアドレスを表示します。
編集する場合は，テキスト・ボックスにアドレス式を直接入力するか（最大指定文字数：1024 文字），またはドロップダウン・リストにより入力履歴項目（最大履歴個数：10 個）を選択します。

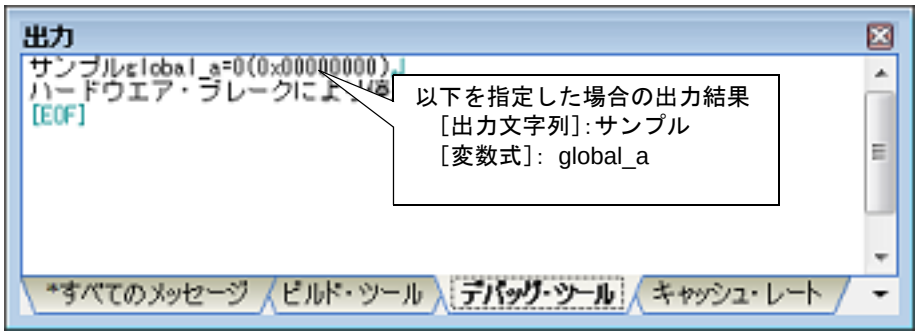
備考 このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

(d) ［OK］ボタンのクリック
ここで指定した Printf イベントをエディタ パネル／逆アセンブル パネル上のキャレット位置の行に設定します。
Printf イベントが設定されると，エディタ パネル／逆アセンブル パネルのイベント・エリアに マークが表示され，イベント パネルで管理されます（「2.16 イベントの管理」参照）。

(2) プログラムを実行する
プログラムを実行します（「2.8 プログラムの実行」参照）。
プログラムを実行することにより，Printf イベントを設定した箇所の命令実行直前でプログラムを一瞬停止し，指定した変数式の値を出力 パネルに出力します。

(3) 出力結果を確認する
出力 パネルの［デバッグ・ツール］タブでは，指定した変数式の値が次のように出力されます（「図 A.36 Printf イベントの出力結果フォーマット」参照）。

図 2.117 Printf イベントの出力結果例



(4) Printf イベントを編集する
一度設定した Printf イベントを編集することができます。
編集を行う場合は，イベント パネルにおいて，編集対象の Printf イベントを選択したのち，コンテキスト・メニューの［条件の編集 ...］を選択します。オープンするアクション・イベント ダイアログにおいて，編集が必要な項目を編集したのち，［OK］ボタンをクリックします。

2.16 イベントの管理

イベントとは，“ アドレス 0x1000 番地をフェッチした ”，“ アドレス 0x2000 番地にデータを書き込んだ ” などのデバッグにおけるマイコンの特定の状態を指します。
CS+ では，このイベントを任意の箇所でのブレーク，トレース動作の開始／終了，タイマ計測の開始／終了などのデバッグ機能のアクション・トリガとして利用します。
この節では，これらのイベントを管理する方法について説明します。
イベントは，一括して次のイベント パネルで管理されます。
［表示］メニュー→［イベント］を選択してください。
イベント パネルでは，現在設定されているイベントの詳細情報を一覧で確認することができ，各イベントの削除，設定状態（有効／無効）の切り替えを行うことができます。
なお，各エリアの見方，および機能についての詳細は，イベント パネルの項を参照してください。

図 2.118 設定したイベントの表示（イベント パネル）

備考 1. マルチコア対応版を対象とした “ イベントの発生 ” については，「2.7 コア（PE）の選択」も参照してください。

備考 2. 各種イベントの設定方法についての詳細は，次を参照してください。

- 「2.9.3 任意の場所で停止する（ブレークポイント）」
- 「2.9.4 任意の場所で停止する（ブレーク・イベント）」
- 「2.9.5 変数 /I/O レジスタへのアクセスで停止する」
- 「2.12.3 任意区間の実行履歴を収集する」
- 「2.12.4 条件を満たしたときのみの実行履歴を収集する【シミュレータ】」
- 「2.13.2 任意区間の実行時間を計測する【シミュレータ】」
- 「2.15.1 printf を挿入する」

2.16.1 設定状態（有効／無効）を変更する

対象となるイベント名のチェック・ボックスのチェックを変更することで，イベントの設定状態を変更することができます（イベントの設定状態を変更すると，対応してイベント・マークも変化します）。
イベントの設定状態には，次の種類があります。

図 2.119 イベント名のチェック・ボックス

表 2.17 イベントの設定状態

	有効状態	指定されている条件の成立で，対象となるイベントが発生します。 チェックを外すことにより，イベントを無効状態にすることができます。
	無効状態	指定されている条件が成立しても，対象となるイベントは発生しません。 チェックすることにより，イベントを有効状態にすることができます。
	保留状態	指定されている条件が，デバッグ対象のプログラムでは設定することができません。チェック・ボックスを操作することはできません。

備考 1. タイマ計測イベントを有効状態にするためには，タイマ開始イベントとタイマ終了イベントの両方の設定が必要となります。

備考 2. Run-Break タイマ・イベントを無効状態／保留状態にすることはできません。

備考 3. イベントの状態は，エディタ パネル／逆アセンブル パネル上のイベント・マークを右クリックすることで表示される，メニューからの選択でも変更することができます。

備考 4. 無条件トレース・イベントとトレース・イベントにおける有効／無効状態の設定は，排他制御となります。このため，ビルトイン・イベントである無条件トレース・イベントは，デフォルトで有効状態で設定されていますが，トレース開始イベント／トレース終了イベントのいずれかが設定されると同時に自動的に無効状態に変更され，トレース開始イベント／トレース終了イベントを 1 つにまとめたトレース・イベントが有効状態になります。
また逆に，設定されているトレース・イベントを無効状態にすると，自動的に無条件トレース・イベントが有効状態となります。

2.16.2 特定のイベント種別のみ表示する

ツールバーの次のボタンをクリックすることで，特定のイベント種別のみを表示することができます。

	ハードウエア・ブレーク関連のイベントを表示します。
【Full-spec emulator】【E1】【E20】	ソフトウエア・ブレーク関連のイベントを表示します。
	トレース関連のイベントを表示します。
	タイマ関連のイベントを表示します。
	アクション・イベント（Printf イベント）を表示します。
	ビルトイン・イベント（無条件トレース・イベント /Run-Break タイマ・イベント）を表示します。

2.16.3 イベントのアドレスにジャンプする

次のボタンをクリックすることにより，現在選択しているイベントのアドレスに対応して，各パネルにジャンプします。
ただし，トレース・イベント／タイマ計測イベント／ビルトイン・イベント（無条件トレース・イベント /Run-Break タイマ・イベント）を選択している場合は，このボタンは無効となります。

	選択しているイベントが設定されているアドレスに対応するソース行にキャレットを移動した状態で，エディタ パネルがオープンします。
	選択しているイベントが設定されているアドレスに対応する逆アセンブル結果にキャレットを移動した状態で，逆アセンブル パネルがオープンします。

	選択しているイベントが設定されているアドレスに対応するメモリ値にキャレットを移動した状態で，メモリ パネルがオープンします。

2.16.4 イベントを削除する

設定したイベント，およびイベント条件を削除するには，対象イベントを選択したのち，ツールバーの  ボタンをクリックします。
ただし，ビルトイン・イベントである無条件トレース・イベント /Run-Break タイマ・イベントを削除することはできません。

備考 1. 実行系のブレーク・イベントについては，エディタ パネル／逆アセンブル パネル上で表示されているイベント・マークをクリックすることで，イベントを削除することができます。

備考 2. 設定したイベントを一度にすべて削除する場合は，コンテキスト・メニューの［すべて選択］を選択したのち，  ボタンをクリックします（ビルトイン・イベントを除く）。

2.16.5 イベントにコメントを入力する

設定した各イベントに対して，ユーザが自由にコメントを入力することができます。
コメントの入力は，コメントを入力したいイベントを選択したのち，［コメント］エリアをクリックし，任意のテキストをキーボードから直接入力します（［Esc］キーの押下で編集モードをキャンセルします）。
コメントを編集したのち，［Enter］キーの押下，または編集領域以外へのフォーカスの移動により，編集を完了します。
なお，コメントは最大 256 文字まで入力することができ，使用中のユーザの設定として保存されます。

2.16.6 イベント設定に関する留意事項

ここでは，各種イベントの設定を行う際の留意事項を示します。

2.16.6.1 有効イベント数の制限

有効状態で同時に設定可能なイベントの個数には，次の制限があります。
したがって，新たに有効状態のイベントを設定する際にこの制限数を越えてしまう場合は，いったん設定しているイベントのいずれかを無効状態にする必要があります。

表 2.18　有効イベント数の制限

イベント種別	デバッグ・ツール		
	Full-spec emulator	E1/E20	シミュレータ
ハードウエア・ブレーク（実行系：実行後）	ー 12 注 1 4 注 1		ー
ハードウエア・ブレーク（実行系：実行前）			64
ハードウエア・ブレーク（アクセス系）			
ソフトウエア・ブレーク	2000（コード・フラッシュのみ）		ー
トレース（トレース開始／トレース終了）	8 + 7		64
ポイント・トレース	ー		64

イベント種別	デバッグ・ツール		
	Full-spec emulator	E1/E20	シミュレータ
タイマ計測 （タイマ開始／タイマ終了）	－		1 注 2
アクション （Printf イベント）	100		64 注 3

“*x* + *y*”： “ 実行系イベント：*x* 個 ”+“ アクセス系イベント：*y* 個 ”

注 1. ハードウエア・ブレーク（実行系：実行前）の 4 個はハードウエア・ブレーク（アクセス系）と兼用（ハードウエア・ブレーク（アクセス系）を 4 個設定すると，ハードウエア・ブレーク（実行系：実行前）は残り 8 個までしか設定できません）

注 2. タイマ開始イベント／タイマ終了イベントもそれぞれ 1 つのみ設定可

注 3. ハードウエア・ブレークと兼用

2.16.6.2 実行中に設定／削除可能なイベント種別

プログラム実行中，またはトレーサ／タイマ実行中に設定／削除可能なイベント種別は次のとおりです。

表 2.19 実行中に設定／削除可能なイベント種別

イベント種別	デバッグ・ツール		
	Full-spec emulator	E1/E20	シミュレータ
ハードウエア・ブレーク （実行系：実行後）	－		－
ハードウエア・ブレーク （実行系：実行前）	△		▲
ハードウエア・ブレーク （アクセス系）	△		▲
ソフトウエア・ブレーク	×		－
トレース （トレース開始／トレース終了）	▲		▲
ポイント・トレース	－		▲
タイマ計測 （タイマ開始／タイマ終了）	－		×
アクション （Printf イベント）	×		▲

△ ：プログラムの実行を一瞬停止することで可能注
▲ ：トレーサ／タイマ動作中は不可
×：不可
－：非サポート

注 プロパティ パネルの［デバッグ・ツール設定］タブ上の［実行中のイベント設定］カテゴリ内［実行を一瞬停止してイベントを設定する］プロパティにおいて，［はい］を選択することにより可能となります。

2.16.6.3 その他の注意事項

- ローカル変数にイベントを設定することはできません。

- ステップ実行中（リターン実行を含む），およびコンテキスト・メニューの［ここまで実行］によるプログラム実行中，イベントは発生しません。

- デバッグ対象のプログラムを再ダウンロードすることにより，既存のイベント設定位置が命令の途中になる場合における該当イベントの再設定方法は次のとおりです。

  - デバッグ情報がある場合
    イベント設定位置は常にソース・テキスト行の先頭に移動します。

  - デバッグ情報がない場合
    プロパティ パネルの［ダウンロード・ファイル設定］タブ上の［ダウンロード］カテゴリ内［イベント設定位置の自動変更方法］プロパティの設定に依存します。

- 内蔵 ROM/ 内蔵 RAM のサイズを変更することにより，イベント設定箇所がノン・マップ領域になった場合，設定しているイベントは発生しません（イベント パネル上でも無効状態／保留状態に変更されません）。

- 関数名や変数名を先頭のアンダー・バーの有無などで区別している場合，シンボル変換やブレーク・イベントの設定が不正になる場合があります。
  例：“_reset” と “__reset” などの 2 つの関数が存在する場合

2.17 フック処理を設定する

この節では，フック処理機能を使用し，デバッグ・ツールにフックを設定するための操作方法について説明します。
フック処理を設定することで，ロード・モジュールのダウンロード前後や CPU リセット後に，I/O レジスタ /CPU レジスタの値を自動的に変更することができます。
フック処理の設定は，プロパティ パネルの［フック処理設定］タブ上の［フック処理］カテゴリ内で行います。

備考　　たとえば，［ダウンロード前］プロパティで I/O レジスタを設定することにより，ダウンロードを高速に行うことができます。
また，外部 RAM へのダウンロードも，同様の設定で容易に行うことができます。

図 2.120　［フック処理］カテゴリ

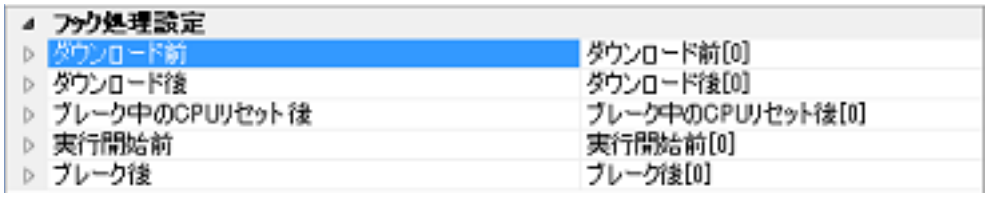


表 2.20　［フック処理］カテゴリのプロパティ

プロパティ	タイミング
ダウンロード前	ロード・モジュール・ファイルをダウンロードする直前に，指定した処理を行います。
ダウンロード後	ロード・モジュール・ファイルをダウンロードした直後に，指定した処理を行います。
ブレーク中の CPU リセット後	ブレーク中の CPU リセット直後に，指定した処理を行います。
実行開始前	プログラムの実行開始直前に，指定した処理を行います。
ブレーク後	プログラムの実行がブレークした直後に，指定した処理を行います。

［フック処理］カテゴリ内の各プロパティは，フック処理を行うタイミングを示し，プロパティ値の “[ ]” 内は，現在指定されている処理の数を示します（デフォルトで設定されているフック処理はありません）。
フック処理を行いたいプロパティに，目的の処理を次の手順で指定します。
処理の指定は，該当するプロパティを選択すると欄内右端に表示される［...］ボタンをクリックすることでオープンする，次のテキスト編集 ダイアログ上で行います。

図 2.121　テキスト編集 ダイアログのオープン

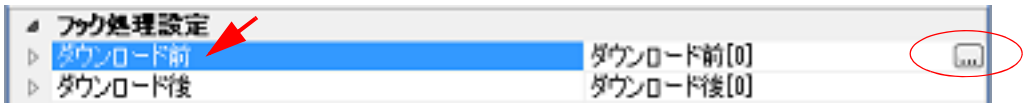
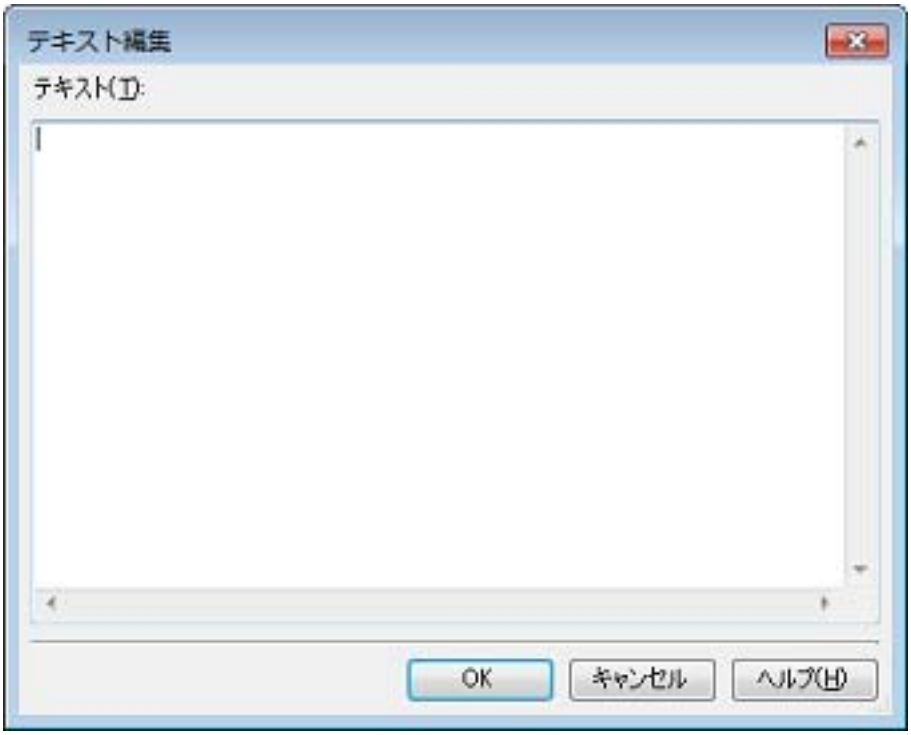


図 2.122 フック処理を設定する（テキスト編集 ダイアログ）



このダイアログにおいて，目的の処理を直接入力により指定します。
各処理の指定形式は次のとおりです。

【処理 1】
*I/O レジスタ*の内容を，*数値*に自動的に書き換えます。
指定形式：

```
I/O レジスタ名 数値
```

【処理 2】
*CPU レジスタ*の内容を，*数値*に自動的に書き換えます。
指定形式：

```
CPU レジスタ名 数値
```

【処理 3】
Python スクリプト・パス（絶対パス／プロジェクト・フォルダを基点とした相対パス）で指定したスクリプト・ファイルを実行します。
指定形式：

```
Source Python スクリプト・パス
```

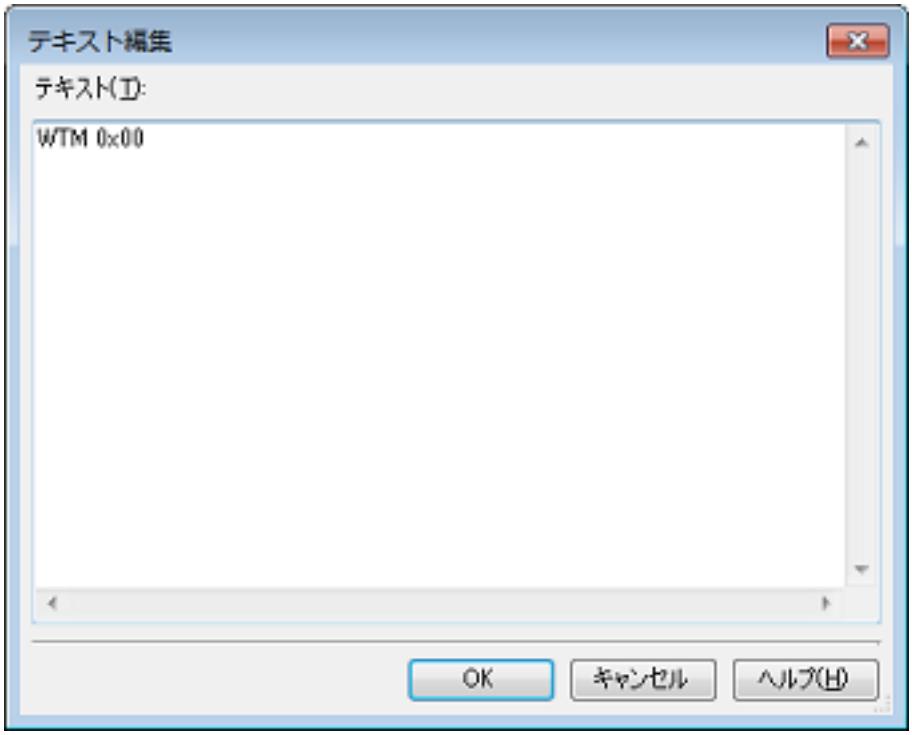
備考 1. 処理の指定の際，行頭に “#” を付与することにより，その行はコメント扱いとなります。

備考 2. 半角スペースは，タブ文字でも代用可能です。

注意 デバッガのフック処理から Python スクリプトを実行する場合，以下のコマンドが記載可能です。
debugger.Register.GetValue
debugger.Register.SetValue
debugger.Memory.GetValue
debugger.Memory.SetValue
それ以外の Python コマンドを使用したい場合 Python コンソールの Hook コマンドを使用してください。

1 処理につき 64 文字まで入力可能で，各プロパティごとに 128 個までの処理を指定することができます（テキスト編集 ダイアログ上の［テキスト］エリア内の 1 行が 1 処理に相当）。
処理の指定が完了したのち，［OK］ボタンをクリックすると，指定した処理がプロパティ パネル上に反映されます。

図 2.123　　フック処理設定の例



2.18 入力値について

この節では，各パネル／ダイアログに値を入力する際の留意事項について説明します。

2.18.1 入力規約

各パネル／ダイアログへの入力規約を次に示します。

(1) 文字セット
入力を許可している文字セットは次のとおりです。

表 2.21　　文字セットの一覧

文字セット	概要
ASCII	半角のアルファベット（英字），半角の数字，および半角の記号
Shift-JIS	全角のアルファベット（英字），全角の数字，全角の記号，ひらがな，全角のカタカナ，漢字，および半角のカタカナ
EUC-JP	全角のアルファベット（英字），全角の数字，全角の記号，ひらがな，全角のカタカナ，漢字，および半角のカタカナ
UTF-8	全角のアルファベット（英字），全角の数字，全角の記号，ひらがな，全角のカタカナ，漢字（中国語を含む），および半角のカタカナ
UTF-16 （Unicode）	全角のアルファベット（英字），全角の数字，全角の記号，ひらがな，全角のカタカナ，漢字（中国語を含む），および半角のカタカナ

(2) エスケープ・シーケンス
入力を許可しているエスケープ・シーケンスは次のとおりです。

表 2.22　　エスケープ・シーケンスの一覧

エスケープ・シーケンス	値	意味
¥0	0x00	null 文字
¥a	0x07	アラート
¥b	0x08	バックスペース
¥t	0x09	水平タブ
¥n	0x0A	改行
¥v	0x0B	垂直タブ
¥f	0x0C	フォーム・フィード
¥r	0x0D	キャリッジ・リターン
¥"	0x22	ダブルクォーテーション
¥'	0x27	シングルクォーテーション
¥?	0x3F	疑問符（? と入力された場合も疑問符として扱います）
¥\	0x5C	バックスラッシュ

(3) 数値
数値を入力する際に許可している進数は次のとおりです。

表 2.23　　進数の一覧

進数表記	概要
2 進数	0b で始まり，0 ～ 1 の数値が続く数値

進数表記	概要
8 進数	0 で始まり，0 ～ 7 の数字が続く数値
10 進数	0 以外で始まり，0 ～ 9 の数字が続く数値
16 進数	0x で始まり，0 ～ 9 の数字，および a ～ f の英字が続く数値 （英字の大文字／小文字については，不問） ただし，HEX マークが表示されている入力エリアでは，0x の接頭辞は必要ありません。

(4)　式と演算子

式とは，定数，レジスタ名，I/O レジスタ名，シンボル，およびこれらを演算子で結合したものを示します。
式には，アドレス式とウォッチ式があります。シンボルのアドレスを必要とする式をアドレス式，シンボルの値を必要とする式をウォッチ式と呼びます。

(a)　アドレス式と演算子

アドレス式では，シンボルのアドレスを使用して演算します。CPU レジスタ名が記述された場合のみ，値を使用して演算します。
アドレス式の基本入力形式は次のとおりです。

表 2.24　　アドレス式の基本入力形式

式	説明
C 言語変数名[注 1]	C 言語の変数のアドレス
式[*式*][注 2]	配列のアドレス
式. メンバ名	構造体／共用体のメンバのアドレス
式 -> メンバ名	ポインタの指し示す構造体／共用体のメンバのアドレス
CPU レジスタ名	CPU レジスタの値
I/O レジスタ名	I/O レジスタのアドレス
ラベル名[注 3]，EQU シンボル名[注 3]，[即値]	ラベルのアドレス，EQU シンボルの値，即値アドレス
整定数	アドレス

注 1.　C 言語変数の値がレジスタに割り付いている場合は，エラーになります。

注 2.　インデックスとして入力された式は，ウォッチ式として解析します。

注 3.　ラベル名または EQU シンボル名に “$” が含まれている場合，名前を “{}” で囲んでください（例：{$Label}）。
“I” は虚数のキーワードとなるため，CPU レジスタの “I” を指定する場合は，“:REG” を付加してください（例：I:REG）。

また，「表 2.24 アドレス式の基本入力形式」から，次の演算子を用いたアドレス式を構成することができます。

表 2.25　　演算子を用いたアドレス式の構成

式	説明
(*式*)	演算順序の指定
- *式*	符号反転
! *式*	論理否定
~ *式*	ビット反転
式 * *式*[注]	乗算

式	説明
式 / *式*[注]	除算
式 % *式*[注]	剰余算
式 + *式*[注]	加算
式 - *式*[注]	減算
式 & *式*[注]	ビットごとの論理積
式 ^ *式*[注]	ビットごとの排他的論理和
式 | *式*[注]	ビットごとの論理和

注　　変数，および関数は，変数／関数／整定数以外と演算子で結合することはできません（例：C 言語変数名 + I/O レジスタ名）。

(b)　ウォッチ式と演算子

ウォッチ式ではシンボルの値を使用して演算します。値が存在しない場合のみ，シンボルのアドレスを使用して演算します（例：main( ) + 1）。
ウォッチ式の基本入力形式は次のとおりです。

表 2.26　　ウォッチ式の基本入力形式

式	説明
C 言語変数名	C 言語の変数の値
式 [*式*]	配列の要素値
式. メンバ名	構造体／共用体のメンバ値
式 -> メンバ名	ポインタの指し示す構造体／共用体のメンバ値
* *式*	ポインタの変数の値
& *式*	配置アドレス
CPU レジスタ名	CPU レジスタの値
I/O レジスタ名	I/O レジスタの値
ラベル名[注]，EQU シンボル名[注]，[即値]	ラベルの値，EQU シンボルの値，即値アドレスの値
整定数	整数の定数値
浮動定数	浮動小数点の定数値
文字定数	文字定数値

注　　ラベル名または EQU シンボル名に “$” が含まれている場合，名前を “{}” で囲んでください（例：{$Label}）。
虚数の値には，大文字の “I” を掛けてください（例：1.0 + 2.0*I）。なお “I” は虚数のキーワードとなるため，CPU レジスタの “I” を指定する場合は，“:REG” を付加してください（例：I:REG）。

また，「表 2.26 ウォッチ式の基本入力形式」から，次の演算子を用いたウォッチ式を構成することができます。次表の演算子は，C 言語仕様に従って式を解析します。

表 2.27　　演算子を用いたウォッチ式の構成

式	説明
(*式*)	演算順序の指定
! *式*	論理否定

式	説明
~ *式*	ビット反転
式 * *式*[注]	乗算
式 / *式*[注]	除算
式 % *式*[注]	剰余算
式 + *式*[注]	加算
式 - *式*[注]	減算
式 & *式*[注]	ビットごとの論理積
式 ^ *式*[注]	ビットごとの排他的論理和
式 | *式*[注]	ビットごとの論理和

注　変数，および関数は，変数／関数／整定数以外と演算子で結合することはできません（例：C 言語変数名 + I/O レジスタ名）。

2.18.2 シンボル名の入力補完機能

シンボル名の入力補完とは，アドレス式などを入力する際に，プログラム中に存在するシンボル名のリストから 1 つを選択することにより，ユーザのシンボル名の入力作業を補佐する機能です。

シンボル名のリストは，この機能に対応するテキスト・ボックスにおいて，目的のシンボル名の一部が入力されている状態で［Ctrl］+［Space］キーを押下することにより表示されます。リスト内において，目的のシンボル名をダブルクリックすることで（または，［↑］／［↓］キーによりシンボル名を選択したのち，［Space］／［Enter］キーを押下），入力中のシンボル名が補完されます。

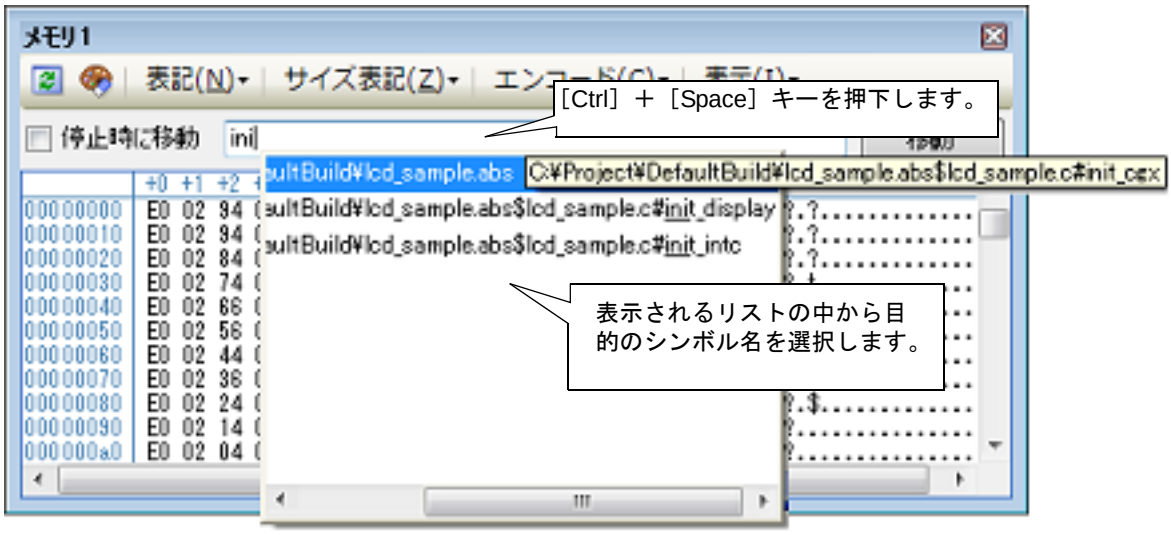
なお，この際に，［Space］／［Enter］キー以外のキーが押下された場合，または現在操作対象としているパネル／ダイアログからフォーカスが移動した場合は，シンボル名のリストは消失します（シンボル名の入力補完は行われません）。

注意 1.　テキスト・ボックスにおいて，1 文字も入力されていない場合，または候補が 1 つも存在しない場合は，シンボル名のリストは表示されません。

注意 2.　シンボル名の入力補完機能に必要な情報は，ロード・モジュール・ファイルのダウンロード時に生成されるため，この情報を生成するとダウンロード時間，およびホスト・マシンのメモリ消費量が増加します。シンボル名の入力補完機能を使用しない場合は，ダウンロード・ファイル ダイアログの［入力補完機能用の情報を生成する］項目で［いいえ］を選択し，この機能を無効化することを推奨します（デフォルト：［はい］）。
ただし，GHS コンパイラを使用している場合，この機能を無効化することはできません（［入力補完機能用の情報を生成する］項目の選択は無視されます）。

備考　シンボル名の入力時に，この機能を使用できるか否かは，該当するパネル／ダイアログの入力エリアの説明を参照してください。

図 2.124 シンボル名の入力補完機能



2.18.3 入力不備箇所に対するアイコン表示

CS+ が提供する一部のダイアログでは，不正な文字列が入力された際，設定すべき値として誤っていることを示すアイコンを該当箇所に表示することにより，入力の不備を警告します。

備考 アイコン上にマウス・カーソルを移動した際には，入力すべき文字列に関する情報がポップアップ表示されます。

A. ウインドウ・リファレンス

この付録では，CS+ でデバッグを行う際に使用するウインドウ／パネル／ダイアログについての詳細を説明します。

A.1 説　　明

次に，デバッグに関するウインドウ／パネル／ダイアログの一覧を示します。

表 A.1　　ウインドウ／パネル／ダイアログ一覧

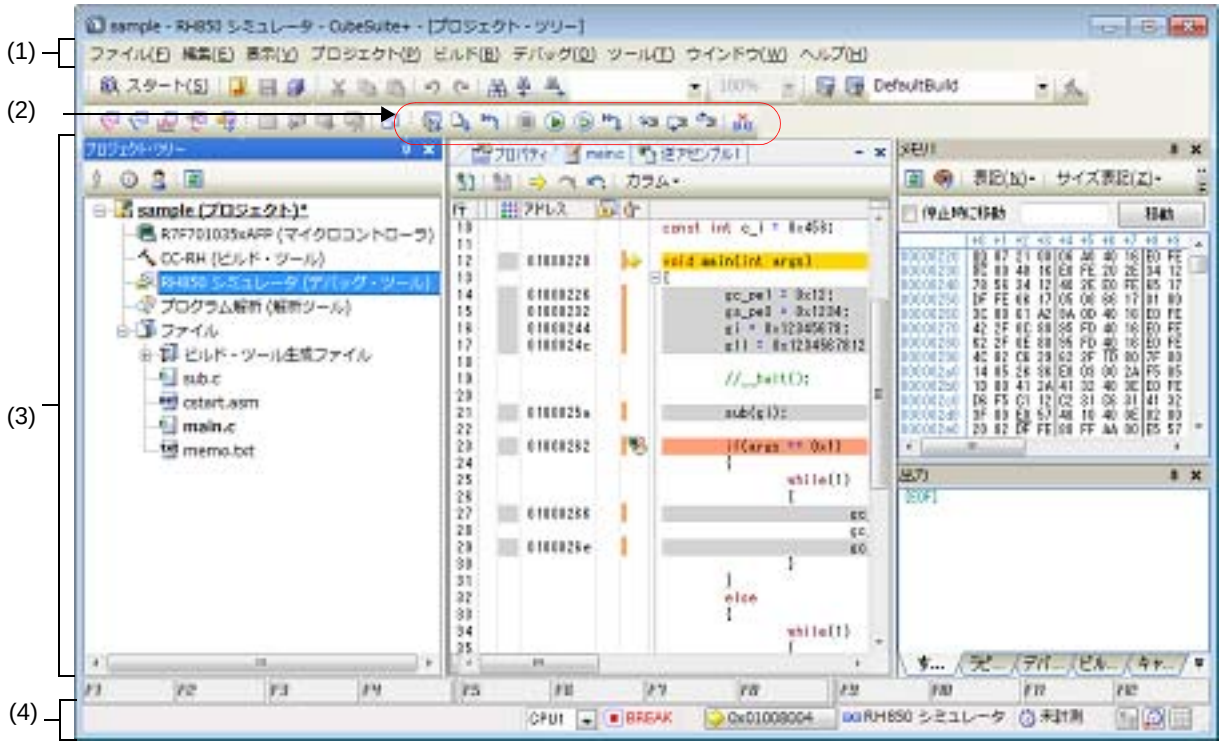
ウインドウ／パネル／ダイアログ名	機能概要
メイン・ウインドウ	プログラムの実行制御，および各パネルのオープン
デバッグ・マネージャ パネル	デバッグ対象とするコア（PE）の選択，およびその状態の表示
プロジェクト・ツリー パネル	使用するデバッグ・ツールの選択
プロパティ パネル	プロジェクト・ツリー パネルで選択しているデバッグ・ツールについて，詳細情報の表示，および設定の変更
メモリ パネル	メモリの値の表示，および値の変更
逆アセンブル パネル	メモリ値を逆アセンブルした結果の表示，ライン・アセンブル，および命令レベル・デバッグ
CPU レジスタ パネル	CPU レジスタ（プログラム・レジスタ／システム・レジスタ）の内容の表示，および値の変更
IOR パネル	I/O レジスタの内容の表示，および値の変更
ローカル変数 パネル	ローカル変数の内容の表示，および値の変更
ウォッチ パネル	登録したウォッチ式の内容の表示，および値の変更
コール・スタック パネル	関数呼び出しのコール・スタック情報の表示
トレース パネル	デバッグ・ツールから取得したトレース・データの表示
イベント パネル	設定イベントの詳細情報の表示，有効／無効の切り替え，および削除
出力 パネル	ビルド・ツール／デバッグ・ツール／各プラグインから出力されるメッセージ，または検索・置換 ダイアログ による一括検索を行った際の結果の表示
メモリ・マッピング ダイアログ	メモリ・マッピングの表示
ダウンロード・ファイル ダイアログ	ダウンロードする際のファイルの選択，およびダウンロード条件の設定
フラッシュ・オプションの設定 ダイアログ【Full-spec emulator】【E1】【E20】	フラッシュ・メモリのオプション設定
優先するブート・ローダ・プロジェクトの選択 ダイアログ	優先してデバッグ対象とするブート・ローダ・プロジェクトの選択
アクション・イベント ダイアログ	アクション・イベントの設定
表示桁数設定 ダイアログ	メモリ パネルにおけるメモリ値の表示桁数の設定
アドレス・オフセット設定 ダイアログ	メモリ パネルにおけるアドレス表示のオフセット値の設定
メモリ初期化 ダイアログ	メモリの初期化
メモリ検索 ダイアログ	メモリの検索

ウインドウ／パネル／ダイアログ名	機能概要
印刷アドレス範囲設定 ダイアログ	逆アセンブル パネルにおける印刷範囲の設定
トレース検索 ダイアログ	トレース・データの検索
スクロール範囲設定 ダイアログ	メモリ パネル／逆アセンブル パネルのスクロール範囲の設定
指定位置へ移動 ダイアログ	指定した位置にキャレットを移動
データ保存 ダイアログ	各パネルの表示内容，およびアップロード・データの保存

メイン・ウインドウ

CS+ を起動した際，最初にオープンするウインドウです。
デバッグを行う際は，このウインドウからプログラムの実行制御，および各パネルのオープン操作を行います。

図 A.1 メイン・ウインドウ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］

［オープン方法］

- Windows の［スタート］→［プログラム］→［Renesas Electronics CS+］→［CS+ for CC］を選択

［各エリアの説明］

(1) メニューバー
デバッグ関連のメニュー項目は次のとおりです。

備考 各メニューから引き出される項目は，ユーザ設定 ダイアログでカスタマイズすることができます。

(a) ［表示］
［表示］メニューの各項目，および機能は次のとおりです（デフォルト）。

デバッグ・マネージャ	デバッグ・マネージャ パネルをオープンします。 ただし，選択しているマイクロコントローラがシングルコア版の場合，またはデバッグ・ツールと切断時は無効となります。
ウォッチ	ウォッチ パネルをオープンするために，次のカスケード・メニューを表示します。 ただし，デバッグ・ツールと切断時は無効となります。

ウォッチ 1	ウォッチ パネル（ウォッチ 1）をオープンします。
ウォッチ 2	ウォッチ パネル（ウォッチ 2）をオープンします。
ウォッチ 3	ウォッチ パネル（ウォッチ 3）をオープンします。
ウォッチ 4	ウォッチ パネル（ウォッチ 4）をオープンします。
ローカル変数	ローカル変数 パネルをオープンします。 ただし，デバッグ・ツールと切断時は無効となります。
コール・スタック	コール・スタック パネルをオープンします。 ただし，デバッグ・ツールと切断時は無効となります。
メモリ	メモリ パネルをオープンするために，次のカスケード・メニューを表示します。 ただし，デバッグ・ツールと切断時は無効となります。
メモリ 1	メモリ パネル（メモリ 1）をオープンします。
メモリ 2	メモリ パネル（メモリ 2）をオープンします。
メモリ 3	メモリ パネル（メモリ 3）をオープンします。
メモリ 4	メモリ パネル（メモリ 4）をオープンします。
IOR	IOR パネルをオープンします。 ただし，デバッグ・ツールと切断時は無効となります。
CPU レジスタ	CPU レジスタ パネルをオープンします。 ただし，デバッグ・ツールと切断時は無効となります。
トレース	トレース パネルをオープンします。 ただし，デバッグ・ツールと切断時は無効となります。
逆アセンブル	逆アセンブル パネルをオープンするために，次のカスケード・メニューを表示します。 ただし，デバッグ・ツールと切断時は無効となります。
逆アセンブル 1	逆アセンブル パネル（逆アセンブル 1）をオープンします。
逆アセンブル 2	逆アセンブル パネル（逆アセンブル 2）をオープンします。
逆アセンブル 3	逆アセンブル パネル（逆アセンブル 3）をオープンします。
逆アセンブル 4	逆アセンブル パネル（逆アセンブル 4）をオープンします。
イベント	イベント パネルをオープンします。 ただし，デバッグ・ツールと切断時は無効となります。
現在の PC 位置を開く	カレント PC 位置（PC レジスタ値）をエディタ パネルで表示します。 ただし，デバッグ・ツールと切断時は無効となります。
ジャンプ前の位置へ戻る	定義箇所へジャンプ（「2.6.2.4 シンボル定義箇所へ移動する」参照）する前の位置へ戻ります。
ジャンプ先の位置へ進む	[ジャンプ前の位置へ戻る] を実行する前の位置へ進みます。
タグ・ジャンプ	エディタ パネル／出力 パネルにおいて，キャレットのある行にファイル名／行／桁の情報がある場合，該当するファイルの該当行／該当桁へジャンプします。

(b) [デバッグ]

[デバッグ] メニューの各項目，および機能は次のとおりです（デフォルト）。

デバッグ・ツールへダウンロード	アクティブ・プロジェクトで現在選択しているデバッグ・ツールに，指定されたファイルをダウンロードします。 デバッグ・ツールと切断時の場合は，自動的にデバッグ・ツールに接続し，ダウンロードを実行します。 ただし，プログラム実行中，またはビルド（ラピット・ビルドを除く）実行中は無効となります。
ビルド & デバッグ・ツールへダウンロード	プロジェクトのビルドを行い，ビルド後にアクティブ・プロジェクトで現在選択しているデバッグ・ツールにダウンロードを実行します。 デバッグ・ツールと切断時の場合は，自動的にデバッグ・ツールに接続し，ダウンロードを実行します。 ただし，プログラム実行中，またはビルド（ラピット・ビルドを除く）実行中は無効となります。 なお，ビルドに失敗した場合，ダウンロードは実行しません。
リビルド & デバッグ・ツールへダウンロード	プロジェクトのリビルドを行い，リビルド後にアクティブ・プロジェクトで現在選択しているデバッグ・ツールにダウンロードを実行します。 デバッグ・ツールと切断時の場合は，自動的にデバッグ・ツールに接続し，ダウンロードを実行します。 ただし，プログラム実行中，またはビルド（ラピット・ビルドを除く）実行中は無効となります。 なお，リビルドに失敗した場合，ダウンロードは実行しません。
デバッグ・ツールへ接続	アクティブ・プロジェクトで現在選択しているデバッグ・ツールに接続します。 ただし，デバッグ・ツールと接続時、ビルド（ラピット・ビルドを除く）実行中，またはサポート範囲外のバージョンのコンパイラを使用している場合は無効となります。
デバッグ・ツールからアップロード ...	メモリ内容をファイルに保存するためのデータ保存 ダイアログをオープンします。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。
デバッグ・ツールから切断	現在接続中のデバッグ・ツールとの通信を切断します。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。
使用するデバッグ・ツール	使用するデバッグ・ツールを選択するためのカスケード・メニューを表示します。 なお，プロジェクトで選択しているマイクロコントローラの種類により，表示されるデバッグ・ツールは異なります。
RH850 Full-spec emulator	Full-spec emulator を使用します。
RH850 E1(LPD)	E1 を LPD 通信方式で使用します。
RH850 E20(LPD)	E20 を LPD 通信方式で使用します。
RH850 シミュレータ	シミュレータを使用します。
停止	現在実行中のプログラムを強制的に停止します。 ただし，プログラム停止時，またはデバッグ・ツールと切断時は無効となります。
実行	プログラムをカレント PC 位置から実行し，設定されているブレーク・イベントの条件が成立した場合，実行中のプログラムを停止します。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。
ブレークせずに実行	プログラムをカレント PC 位置から実行し，設定されているブレーク・イベント／アクション・イベントを無視してプログラムの実行を続けます。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。

ステップ・イン	カレント PC 位置からステップ実行し[注]，各パネルの内容を更新します。 関数呼び出しの場合は，呼び出された関数の先頭で停止します。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。
ステップ・オーバー	カレント PC 位置からステップ実行し[注]，各パネルの内容を更新します。 jarl 命令による関数呼び出しの場合は，その関数内のソース行／命令すべてを 1 ステップとみなして実行し，関数から戻る箇所まで実行します（jarl 命令を実行したときと同じネストになるまで，ステップ実行します）。 なお，jarl 命令以外の場合，［ステップ・イン］の選択と同じ動作となります。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。
リターン・アウト	現在の関数からリターンするまで（呼び出し関数に戻るまで）実行します[注]。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。
CPU リセット	CPU をリセットします（プログラムは実行しません）。 ただし，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。
リスタート	CPU をリセットしたのち，リセット番地からプログラムを実行します。 ただし，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。

注 ステップ実行には，ソース・レベル単位と命令レベル単位の実行方法があります。
詳細は，「2.8.3 プログラムをステップ実行する」を参照してください。

(2) デバッグ・ツールバー
デバッグ・ツールバーは，プログラムの実行を制御するためのコマンドをまとめたボタン群です。
各ボタン，および機能は次のとおりです（デフォルト）。

備考 1. 各ツールバーのボタンは，ユーザ設定 ダイアログでカスタマイズすることができます。また，同ダイアログにより，新規にツールバーを作成することもできます。

備考 2. ツールバー上を右クリックすることで表示されるコンテキスト・メニューにより，ツールバー上に表示／非表示するグループを選択することができます。

	プロジェクトのビルドを行い，ビルド後にアクティブ・プロジェクトのデバッグ・ツールにダウンロードを実行します。 デバッグ・ツールと切断時の場合は，自動的にデバッグ・ツールに接続し，ダウンロードを実行します。 ただし，ビルドに失敗した場合，ダウンロードは実行されません。 ［デバッグ］メニュー→［ビルド & デバッグ・ツールへダウンロード］の選択と同等です。
	アクティブ・プロジェクトのデバッグ・ツールに，指定されたファイルをダウンロードします。 デバッグ・ツールと切断時の場合は，自動的にデバッグ・ツールに接続し，ダウンロードを実行します。 ただし，プログラム実行中，またはビルド（ラピット・ビルドを除く）実行中は無効となります。 ［デバッグ］メニュー→［デバッグ・ツールへダウンロード］の選択と同等です。
	CPU をリセットします（プログラムは実行しません）。 ただし，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。 ［デバッグ］メニュー→［CPU リセット］の選択と同等です。
	現在実行中のプログラムを強制的に停止します。 ただし，プログラム停止時，またはデバッグ・ツールと切断時は無効となります。 ［デバッグ］メニュー→［停止］の選択と同等です。

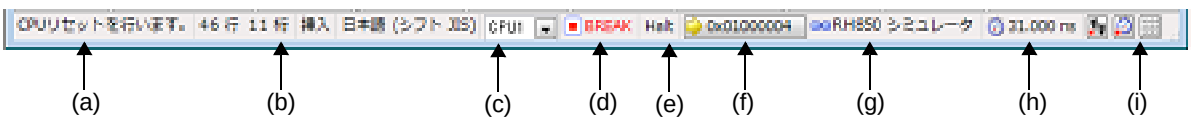
	プログラムをカレント PC 位置から実行し，設定されているブレーク・イベントの条件が成立した場合，実行中のプログラムを停止します。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。 ［デバッグ］メニュー→［実行］の選択と同等です。
	プログラムをカレント PC 位置から実行し，設定されているブレーク・イベント／アクション・イベントを無視してプログラムの実行を続けます。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。 ［デバッグ］メニュー→［ブレークせずに実行］の選択と同等です。
	CPU をリセットしたのち，リセット番地からプログラムを実行します。 ただし，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。 ［デバッグ］メニュー→［リスタート］の選択と同等です。
	カレント PC 位置からでステップ実行し[注]，各パネルの内容を更新します（ステップ・イン実行）。 関数呼び出しの場合は，呼び出された関数の先頭で停止します。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。 ［デバッグ］メニュー→［ステップ・イン］の選択と同等です。
	カレント PC 位置からステップ実行し[注]，各パネルの内容を更新します（ステップ・オーバ実行）。 jarl 命令による関数呼び出しの場合は，その関数内のソース行／命令すべてを 1 ステップとみなして実行し，関数から戻る箇所まで実行します（jarl 命令を実行したときと同じネストになるまで，ステップ実行します）。 なお，jarl 命令以外の場合，  ボタンのクリックと同じ動作となります。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。 ［デバッグ］メニュー→［ステップ・オーバー］の選択と同等です。
	現在の関数からリターンするまで（呼び出し関数に戻るまで）実行します[注]（リターン・アウト実行）。 ただし，プログラム実行中，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。 ［デバッグ］メニュー→［リターン・アウト］の選択と同等です。
	現在接続中のデバッグ・ツールとの通信を切断します。 ただし，ビルド（ラピット・ビルドを除く）実行中，またはデバッグ・ツールと切断時は無効となります。 ［デバッグ］メニュー→［デバッグ・ツールから切断］の選択と同等です。

注 ステップ実行には，ソース・レベル単位と命令レベル単位の実行方法があります。
詳細は，「2.8.3 プログラムをステップ実行する」を参照してください。

(3) パネル表示エリア
各種パネルを表示するエリアです。
表示内容についての詳細は，各パネルの項を参照してください。

(4) ステータスバー
ステータスバーは，次の情報を表示します。

図 A.2 ステータスバー



(a) ステータス・メッセージ
次のメッセージを表示します。

- 選択しているメニュー項目の簡易説明
- パネル／ダイアログにおいて入力値が不正な場合のメッセージ

- 検索・置換 ダイアログにより検索した際に，指定文字列が見つからなかった場合のメッセージ
- ブレークした際のブレーク要因（「2.9 プログラムの停止（ブレーク）」参照）

(b) フォーカス・パネルのステータス情報
現在フォーカスのあるパネルのステータス情報（キャレット位置や上書き／挿入モードなどの情報）を表示します。
ただし，ステータス情報を持たないパネルの場合は非表示となります。

(c) デバッグ対象コアの指定
デバッグ対象とするコア（PE）を指定します（「2.7 コア（PE）の選択」参照）。
ただし，選択しているマイクロコントローラがシングルコア版の場合，またはデバッグ・ツールと切断時の場合は非表示となります。

(d) 実行状態
プログラムの現在の実行状態を次のアイコンと文字列で示します。
ただし，デバッグ・ツールと切断時の場合は非表示となります。

プログラムの状態	表示内容
実行中	RUN
停止中	BREAK
ステップ実行中	STEP

(e) CPU 状態
デバッグ・ツールの現在の CPU の状態を表示します。
なお，同時に複数の状態になっている場合は“&”で区切って状態を列挙して表示します。
ただし，デバッグ・ツールと切断時の場合は非表示となります。

デバッグ・ツール	表示内容	CPU 状態
Full-spec emulator E1/E20	Halt	HALT モード中
	Stop	STOP モード中
	Reset	リセット状態
	Pow Off	ターゲットに電源が供給されていない状態
シミュレータ	Halt	HALT モード中
	StopIdle	STOP/IDLE モード中
	Reset	リセット状態

(f) カレント PC 位置
現在のカレント PC 位置の値を 16 進数で表示します。
このエリアをクリックすると，エディタ パネル上のカレント PC 位置へキャレットを移動します。
また，このエリアにマウスを重ねることにより，“カレント PC: 0x *カレント PC 値*（*ソース名* # *行数*[注]）”をポップアップ表示します。
ただし，デバッグ・ツールと切断時の場合は非表示となります。

注 情報の取得が不可能な場合は，“*シンボル名* + *オフセット値*”となります。

備考 プログラム実行中は，“実行中”と表示します。
ただし，リアルタイム表示更新を行っている場合，設定している表示更新間隔で PC 位置を更新して表示します。

(g) デバッグ・ツールとの接続状態
現在のデバッグ・ツールとの接続状態を次のアイコンと文字列で示します。

接続状態	表示内容
接続中	*デバッグ・ツール名*
切断中	非接続

(h) Run-Break タイマ結果

Run-Break タイマの計測結果（「2.13.1 実行停止までの実行時間を計測する」参照）を表示します。表示単位は計測結果に依存します。
ただし，デバッグ・ツールと切断時の場合は非表示となります。

状態	表示内容
計測していない状態	未計測
計測中	計測中
オーバフローした場合	OVERFLOW

(i) デバッグ・ツールの状態
現在のデバッグ・ツールの各機能の状態を次のアイコンで示します。
ただし，デバッグ・ツールと切断時の場合は非表示となります。

機能	使用する		使用しない
	動作中	停止中	
トレース			
タイマ			
カバレッジ			

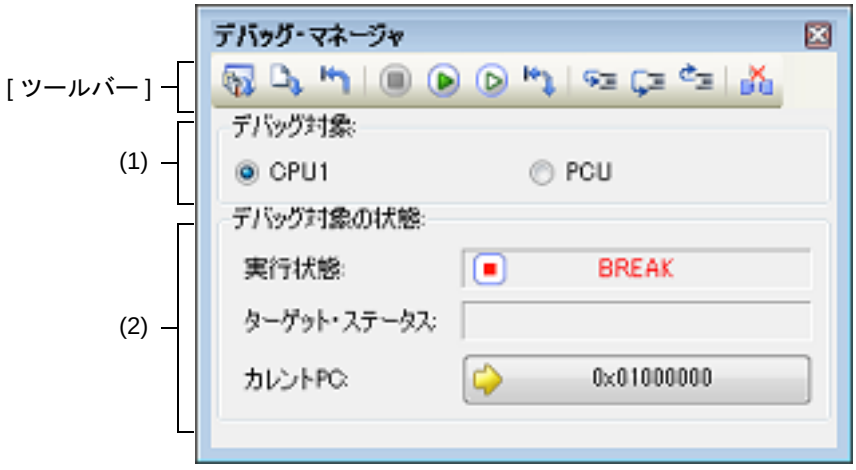
備考 【シミュレータ】
プログラム実行が停止している場合，対象アイコンをクリックすることにより，“使用する”／“使用しない”の状態を変更することができます（プロパティ パネルの［デバッグ・ツール設定］タブ上の［トレース］／［タイマ］／［カバレッジ］カテゴリ内［トレース機能を使用する］／［タイマ機能を使用する］／［カバレッジ機能を使用する］プロパティの指定に反映されます）。

デバッグ・マネージャ パネル

選択しているマイクロコントローラがマルチコア対応版の場合において，デバッグ対象とするコア（PE：プロセッサ・エレメント）の選択，およびその状態の表示を行います（「2.7 コア（PE）の選択」参照）。
なお，このパネルは，デバッグ・ツールと接続時のみオープンすることができます。

注意 選択しているマイクロコントローラがシングルコア版の場合，このパネルをオープンすることはできません。

図 A.3 デバッグ・マネージャ パネル



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［ツールバー］

［オープン方法］

- ［表示］メニュー→［デバッグ・マネージャ］を選択

［各エリアの説明］

(1) ［デバッグ対象コアの指定］エリア
デバッグ対象とする PE をオプション・ボタンにより指定します。
ただし，プログラム実行中はこのエリアは無効となります。

備考 デバッグ対象コアの指定は，メイン・ウインドウのステータスバー上においても行うことができます。

(2) ［デバッグ対象コアの状態］エリア
現在選択しているコアの状態を表示します。

備考 このエリアに表示される内容は，メイン・ウインドウのステータスバー上においても確認することができます。

(a) ［実行状態］
現在のプログラムの実行状態を次のアイコンと文字列で示します。

プログラムの状態	表示内容
実行中	RUN
停止中	BREAK
ステップ実行中	STEP

(b) ［コア・ステータス］
現在のデバッグ・ツールのコアの状態を示します。
なお，同時に複数の状態になっている場合は “&” で区切って状態を列挙して表示します。

デバッグ・ツール	表示内容	状態
Full-spec emulator E1/E20	Halt	HALT モード中
	StopIdle	ハードウエア STOP/ ソフトウエア STOP/IDLE モード中
	Hold	バス・ホールド中
	Wait	ウエイト状態
	Reset	リセット状態
	Pow Off	ターゲットに電源が供給されていない状態
シミュレータ	Halt	HALT モード中
	StopIdle	STOP/IDLE モード中
	Reset	リセット状態

(c) ［カレント PC］
現在のカレント PC 位置の値を 16 進数で示します。
このボタンをクリックすると，エディタ パネル上のカレント PC 位置へキャレットを移動します。

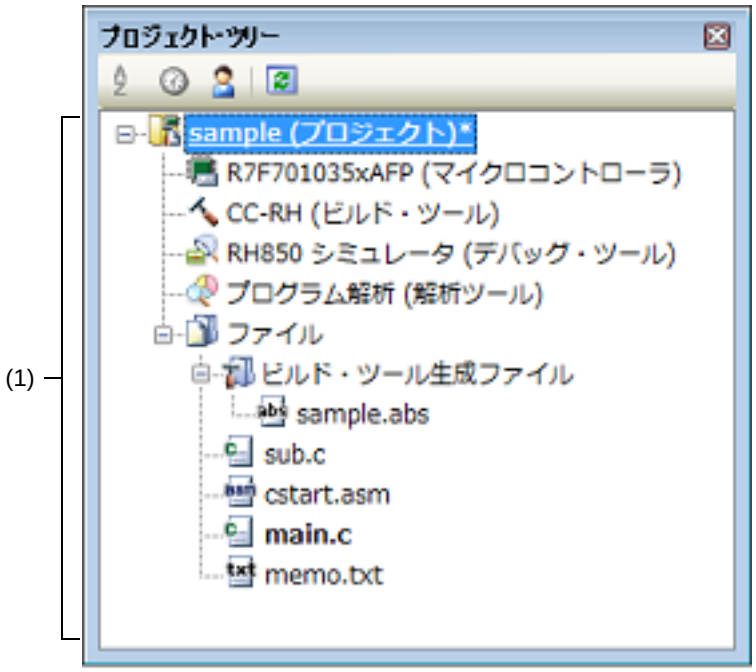
［ツールバー］

ツールバーの機能は，メイン・ウインドウ上のデバッグ・ツールバーの機能と同等です。
各ボタンの機能についての詳細は，「(2) デバッグ・ツールバー」を参照してください。

プロジェクト・ツリー パネル

プロジェクトの構成要素（マイクロコントローラ，ビルド・ツール，デバッグ・ツールなど）をツリー形式で表示します。
なお，使用するデバッグ・ツールの選択／切り替えは，このパネル上で行います。

図 A.4 プロジェクト・ツリー パネル



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］メニュー→［プロジェクト・ツリー］を選択

［各エリアの説明］

(1) プロジェクト・ツリー エリア
プロジェクトの構成要素を次のノードでツリー表示します。

ノード	説明
品種名 デバッグ・ツール名 (デバッグ・ツール)	- *品種名* プロジェクトで選択しているマイクロコントローラの品種名（RH850）を表示します。 - *デバッグ・ツール名* プロジェクトで使用するデバッグ・ツール名（Full-spec emulator/E1(LPD)/ E20(LPD)/ シミュレータ）を表示します[注]。 なお，新規プロジェクト作成時は，シミュレータが設定されます。

注 選択しているマイクロコントローラの種類により，使用可能なデバッグ・ツールは異なります。

ノードを選択すると，その詳細情報（プロパティ）がプロパティ パネルに表示され，設定の変更を行うことができます（プロパティ パネルがオープンしていない場合は，ノードをダブルクリックすることでオープンします）。

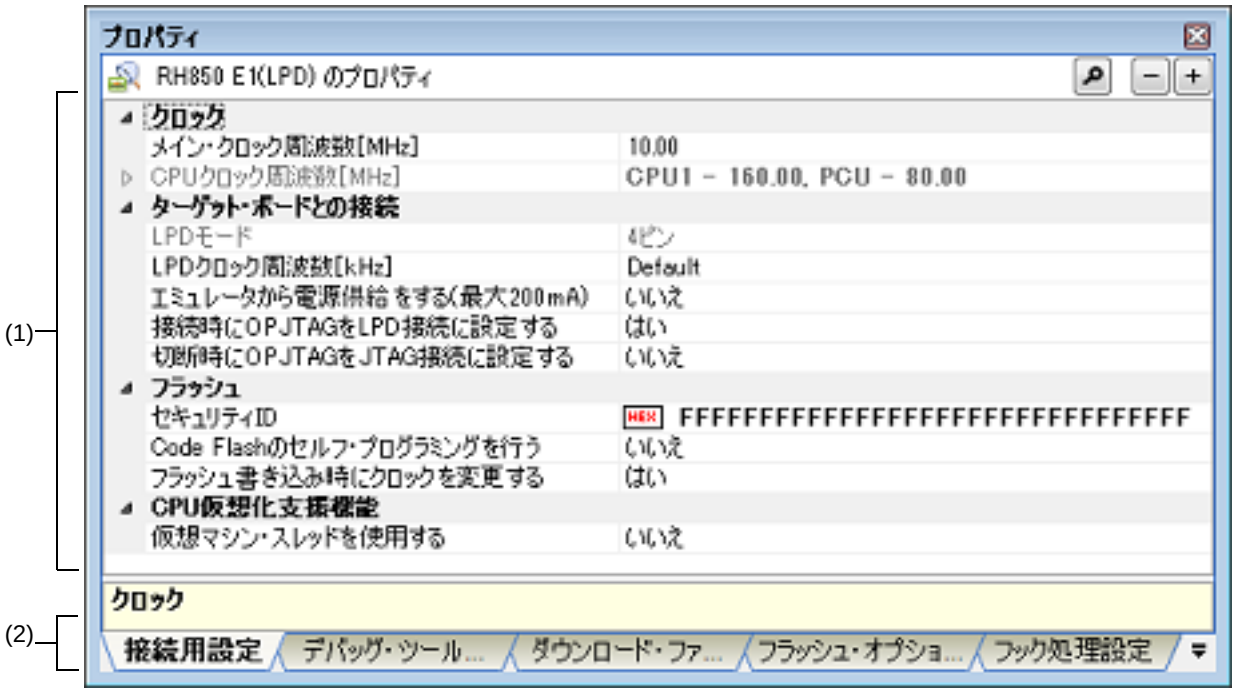
［コンテキスト・メニュー］

使用するデバッグ・ツール	使用するデバッグ・ツールを選択するためのカスケード・メニューを表示します。 なお，プロジェクトで選択しているマイクロコントローラの種類により，表示されるデバッグ・ツールは異なります。
RH850 Full-spec emulator	Full-spec emulator を使用します。
RH850 E1(LPD)	E1 を LPD 通信方式で使用します。
RH850 E20(LPD)	E20 を LPD 通信方式で使用します。
RH850 シミュレータ	シミュレータを使用します。
プロパティ	選択しているデバッグ・ツールのプロパティをプロパティ パネルに表示します。

プロパティ パネル

プロジェクト・ツリー パネルで選択しているデバッグ・ツールについて，カテゴリ別に詳細情報の表示，および設定の変更を行います。

図 A.5 プロパティ パネル（E1 を選択した場合の例）



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［［編集］メニュー（プロパティ パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- プロジェクト・ツリー パネルにおいて，使用する［*品種名 デバッグ・ツール名* (デバッグ・ツール)］ノードを選択したのち，［表示］メニュー，またはコンテキスト・メニューより［プロパティ］を選択
- プロジェクト・ツリー パネルにおいて，使用する［*品種名 デバッグ・ツール名* (デバッグ・ツール)］ノードをダブルクリック

備考 このパネルがすでにオープンしている場合，プロジェクト・ツリー パネル上において，使用する［*品種名 デバッグ・ツール名* (デバッグ・ツール)］ノードを選択することにより，選択したデバッグ・ツールの詳細情報を表示します。

［各エリアの説明］

(1) 詳細情報表示／変更エリア
プロジェクト・ツリー パネルで選択しているデバッグ・ツールの詳細情報を，カテゴリ別のリスト形式で表示し，設定の変更を直接行うことができるエリアです。
⊟ マークは，そのカテゴリ内に含まれているすべてのプロパティ項目が展開表示されていることを示し，また，⊞ マークは，カテゴリ内のプロパティ項目が折りたたみ表示されていることを示します。展開／折りたたみ表示の切り替えは，このマークのクリック，またはカテゴリ名のダブルクリックにより行うことができます。
なお，各プロパティ項目設定欄内に表示される HEX マークは，16 進数入力専用のテキスト・ボックスであることを示します。

カテゴリ，およびそれに含まれるプロパティ項目の表示内容／設定方法についての詳細は，該当するタブの項を参照してください。

(2) タブ選択エリア
タブを選択することにより，詳細情報を表示するカテゴリが切り替わります。
このパネルには，次のタブが存在します（各タブ上における表示内容／設定方法についての詳細は，該当するタブの項を参照してください）。

- ［接続用設定］タブ
- ［デバッグ・ツール設定］タブ
- ［ダウンロード・ファイル設定］タブ
- ［フラッシュ・オプション設定］タブ【Full-spec emulator】【E1】【E20】
- ［フック処理設定］タブ

［［編集］メニュー（プロパティ パネル専用部分）］

元に戻す	直前に行ったプロパティの値の編集作業を取り消します。
切り取り	プロパティの値を編集中の場合，選択している文字列を切り取ってクリップ・ボードに移動します。
コピー	選択しているプロパティの値の文字列をクリップ・ボードにコピーします。
貼り付け	プロパティの値を編集中の場合，クリップ・ボードの内容を挿入します。
削除	プロパティの値を編集中の場合，選択している文字列を削除します。
すべて選択	プロパティの値を編集中の場合，選択しているプロパティの値文字列をすべて選択します。
検索 ...	検索・置換 ダイアログを［クイック検索］タブが選択状態でオープンします。

［コンテキスト・メニュー］

【文字列編集中以外の場合】

デフォルトに戻す	選択しているプロパティ項目の設定値をデフォルトに戻します。
すべてデフォルトに戻す	現在選択しているタブ上の設定値をすべてデフォルトに戻します。

【文字列編集中の場合】

元に戻す	直前に行ったプロパティの値の編集作業を取り消します。
切り取り	プロパティの値を編集中の場合，選択している文字列を切り取ってクリップ・ボードに移動します。
コピー	選択しているプロパティの値文字列をクリップ・ボードにコピーします。
貼り付け	プロパティの値を編集中の場合，クリップ・ボードの内容を挿入します。
削除	プロパティの値を編集中の場合，選択している文字列を削除します。
すべて選択	プロパティの値を編集中の場合，選択しているプロパティの値文字列をすべて選択します。

［接続用設定］タブ

［接続用設定］タブでは，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

(1) ［クロック］
(2) ［ターゲット・ボードとの接続］【Full-spec emulator】【E1】【E20】
(3) ［フラッシュ］【Full-spec emulator】【E1】【E20】
(4) ［コンフィギュレーション］【シミュレータ】
(5) ［CPU 仮想化支援機能］

図 A.6 プロパティ パネル：［接続用設定］タブ【Full-spec emulator】

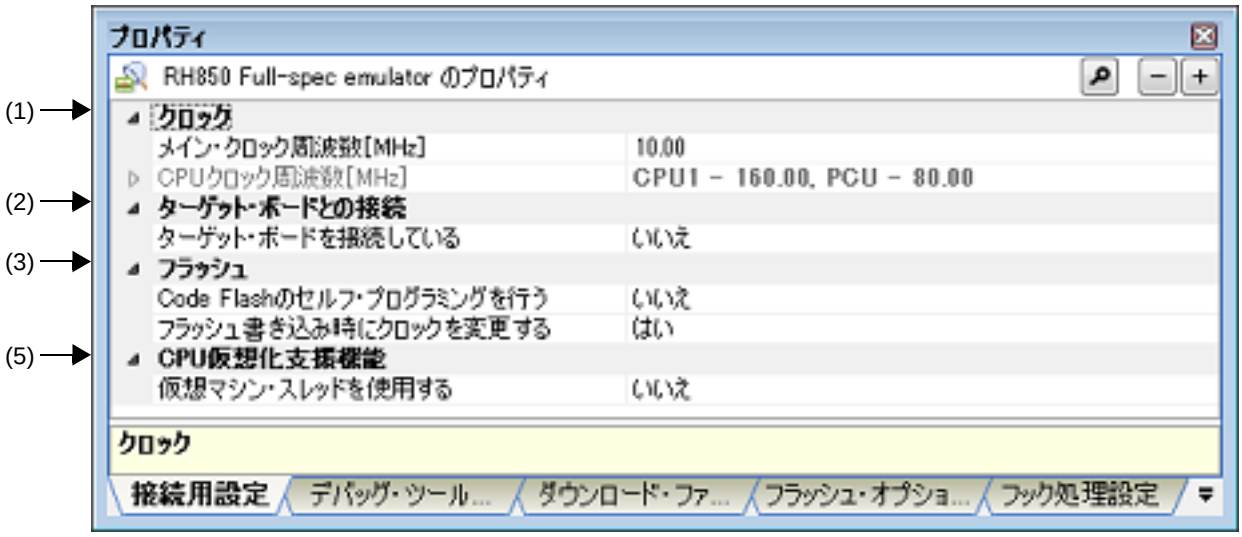


図 A.7 プロパティ パネル：［接続用設定］タブ【E1】【E20】

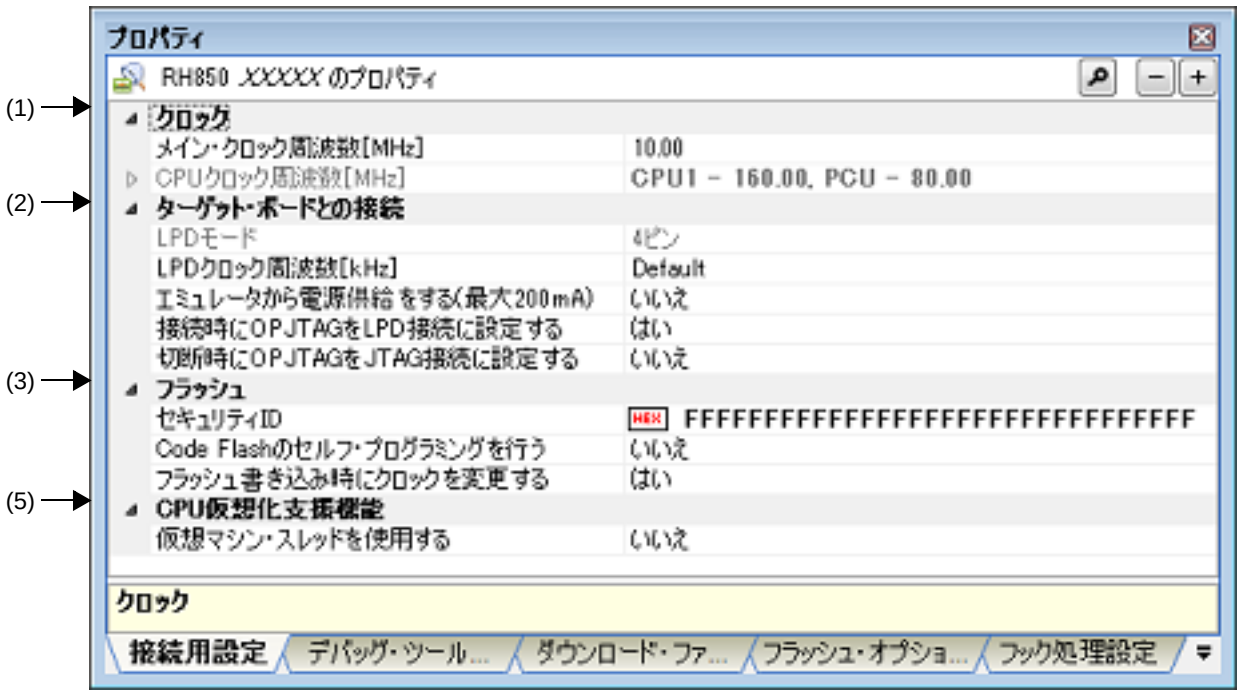
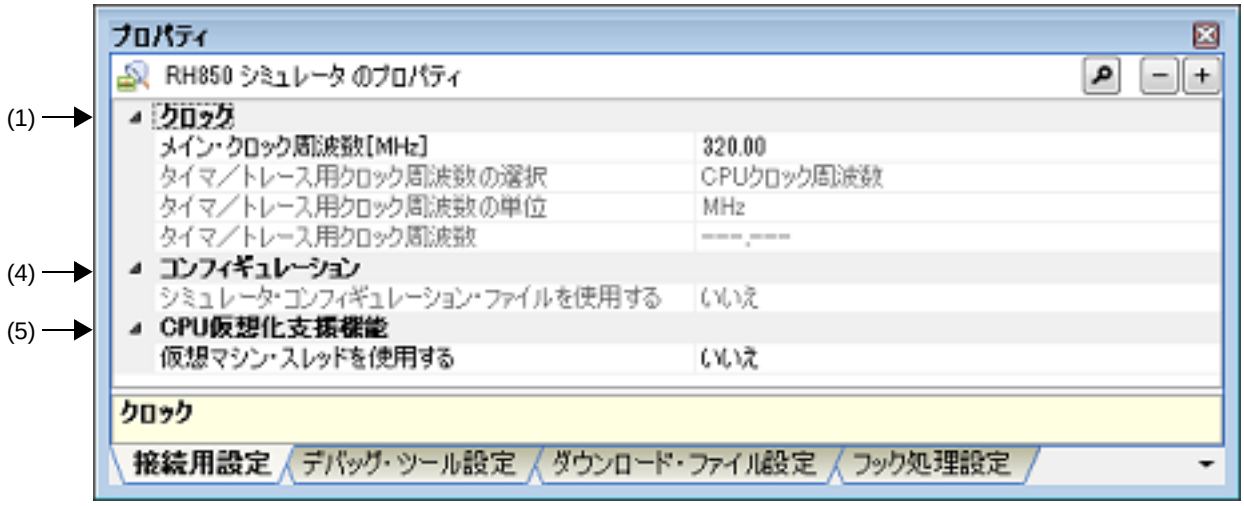


図 A.8 プロパティ パネル：［接続用設定］タブ【シミュレータ】



［各カテゴリの説明］

(1) ［クロック］

クロックに関する詳細情報の表示，および設定の変更を行います。

メイン・クロック周波数 [MHz]	メイン・クロック周波数（逓倍前）を MHz 単位で指定します。	
	デフォルト	【Full-spec emulator】【E1】【E20】 10.00 【シミュレータ】 320.00
	変更方法	ドロップダウン・リストによる選択，またはキーボードからの直接入力
	指定可能値	- ドロップダウン・リストによる次のいずれか 【Full-spec emulator】【E1】【E20】 10.00，20.00（単位：MHz） 【シミュレータ】 1.00，2.00，3.00，3.57，4.00，4.19，4.91，5.00，6.00，7.20，8.00，8.38，9.60，10.00，12.00，16.00，20.00，25.00，30.00，32.00，33.33，34.00，40.00，48.00，50.00，64.00，80.00，160.00，240.00，320.00（単位：MHz） - テキスト入力による次の範囲 0.001 ～ 999.999（単位：MHz）
CPU クロック周波数 [MHz] 【Full-spec emulator】 【E1】【E20】	CPU クロック周波数（逓倍後）をコアごとに指定します。 各コアの CPU クロック周波数は，下段のサブプロパティで指定します。CPU クロック周波数は，トレースのタイム・スタンプ情報を実時間に換算する際に使用されます。 なお，表示するサブプロパティの数は選択しているマイクロコントローラの種類に依存します。	
コア名称（サブプロパティ） 【Full-spec emulator】 【E1】【E20】	選択しているマイクロコントローラが持つコアの名称を表示します。	
	デフォルト	選択しているマイクロコントローラに依存
	変更方法	変更不可

CPU クロック周波数（サブプロパティ） 【Full-spec emulator】 【E1】【E20】	*コア名称*の CPU クロック周波数を指定します。	
	デフォルト	選択しているマイクロコントローラに依存
	変更方法	ドロップダウン・リストによる選択，またはキーボードからの直接入力
	指定可能値	0.001 ～ 999.999（単位：MHz）
タイマ／トレース用クロック周波数の選択 【シミュレータ】	タイマ／トレース機能に使用するクロック周波数を表示します。	
	デフォルト	CPU クロック周波数
	変更方法	変更不可
タイマ／トレース用クロック周波数の単位 【シミュレータ】	タイマ／トレース機能に使用するクロック周波数の単位を表示します。	
	デフォルト	MHz
	変更方法	変更不可
タイマ／トレース用クロック周波数 【シミュレータ】	タイマ／トレース機能に使用するクロック周波数の値を表示します。 ただし，デバッグ・ツールと切断時は “---_---” を表示します。	
	デフォルト	320.00
	変更方法	変更不可

(2) ［ターゲット・ボードとの接続］【Full-spec emulator】【E1】【E20】
ターゲット・ボードとの接続状態に関する詳細情報の表示，および設定の変更を行います。

注意 デバッグ・ツールが CS+ に接続している場合，このカテゴリ内のプロパティを変更することはできません。

ターゲット・ボードを接続している 【Full-spec emulator】	Full-spec emulator にターゲット・ボードを接続しているか否かを選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	ターゲット・ボードを接続しています。
		いいえ	ターゲット・ボードを接続していません。
LPD モード 【E1】【E20】	LPD 通信方式のモードを選択します。		
	デフォルト	選択しているマイクロコントローラに依存	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	選択しているマイクロコントローラに依存	
ボーレート [kbps] 【E1】【E20】	LPD 通信方式の通信速度を選択します。 なお，このプロパティは，［LPD モード］プロパティにおいて［1 ピン］を選択した場合のみ表示されます。		
	デフォルト	500	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	500，1000，2000（単位：kbps）	

LPD クロック周波数 [kHz] 【E1】【E20】	LPD 通信方式のクロック周波数を選択します。 Default を選択した場合，マイクロコントローラ固有のデフォルト値で接続処理を行います。 なお，このプロパティは，［LPD モード］プロパティにおいて［4 ピン］を選択した場合のみ表示されます。		
	デフォルト	Default	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	Default，5500，11000（単位：kHz）	
エミュレータから電源供給をする（最大 200mA） 【E1】	E1 からターゲット・ボードに電源を供給するか否かを選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	電源を供給します。
		いいえ	電源を供給しません。
供給電圧 【E1】	E1 からターゲット・ボードに供給する電圧を選択します。 なお，このプロパティは，［エミュレータから電源供給をする（最大 200mA）］プロパティにおいて［はい］を選択した場合のみ表示されます。		
	デフォルト	3.3V	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	3.3V，5.0V	
接続時に OPJTAG を LPD 接続に設定する 【E1】【E20】	デバッグ・ツールとの接続時にシリアル・プログラミング・モードで起動し，オプション・バイトの設定を LPD 接続に変更するか否かを選択します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	シリアル・プログラミング・モードで起動し，OPJTAG をチェックします。この際に LPD 設定でない場合は LPD 設定に変更し，その後デバッグ・モードに移行します。
		いいえ	デバッグ・モードで起動し，OPJTAG をチェックします。この際に LPD 設定でない場合はメッセージ・ダイアログを表示します。
切断時に OPJTAG を JTAG 接続に設定する 【E1】【E20】	デバッグ・ツールとの接続を切断する際に，オプション・バイトの設定を JTAG 接続に変更するか否かを選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択 ただし，［接続時に OPJTAG を LPD 接続に設定する］プロパティで［いいえ］を選択している場合は変更不可	
	指定可能値	はい	切断時にオプション・バイトの設定を JTAG に変更します。
		いいえ	切断時にオプション・バイトの設定を変更しません。

(3) ［フラッシュ］【Full-spec emulator】【E1】【E20】
フラッシュ書き換えに関する詳細情報の表示，および設定の変更を行います。

注意 デバッグ・ツールが CS+ に接続している場合，このカテゴリ内のプロパティを変更することはできません。

セキュリティ ID 【E1】【E20】	内蔵 ROM，または内蔵フラッシュ・メモリ上のコードを読み出す際の ID コードを指定します注 1。		
	デフォルト	FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	
	変更方法	キーボードからの直接入力	
	指定可能値	32 桁の 16 進数（16 バイト）	
Code Flash のセルフ・プログラミングを行う	フラッシュ・セルフ・プログラミング機能のフラッシュ・セルフ・ライブラリを使用して，Code Flash を書き換えるか否かを選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	Code Flash の書き換えを行います。この場合，Code Flash のキャッシュは行われません。
		いいえ	Code Flash の書き換えを行いません。
フラッシュ書き込み時にクロックを変更する	フラッシュ・メモリに書き込みを行う際に，一時的にクロック・スピードを変更するか否かを選択します注 2。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	フラッシュ・メモリに書き込みを行う際に，クロック・アップを行います。
		いいえ	フラッシュ・メモリに書き込みを行う際に，クロック・アップを行いません。

注 1. ID コードについての詳細は，使用するデバッグ・ツールのユーザーズ・マニュアルを参照してください。

注 2. ［はい］を選択した場合，CPU クロックだけではなく周辺クロックも変化するため，ブレーク中も動作している周辺システムに影響がある可能性があります。
［いいえ］を選択した場合，設定したクロック・スピードが低いと，デバッガ操作によるフラッシュ書き換え時間が長くなります。

(4) ［コンフィギュレーション］【シミュレータ】
このカテゴリ内のプロパティは，常に無効です。

(5) ［CPU 仮想化支援機能］
このカテゴリ内のプロパティは，常に無効です。

［デバッグ・ツール設定］タブ

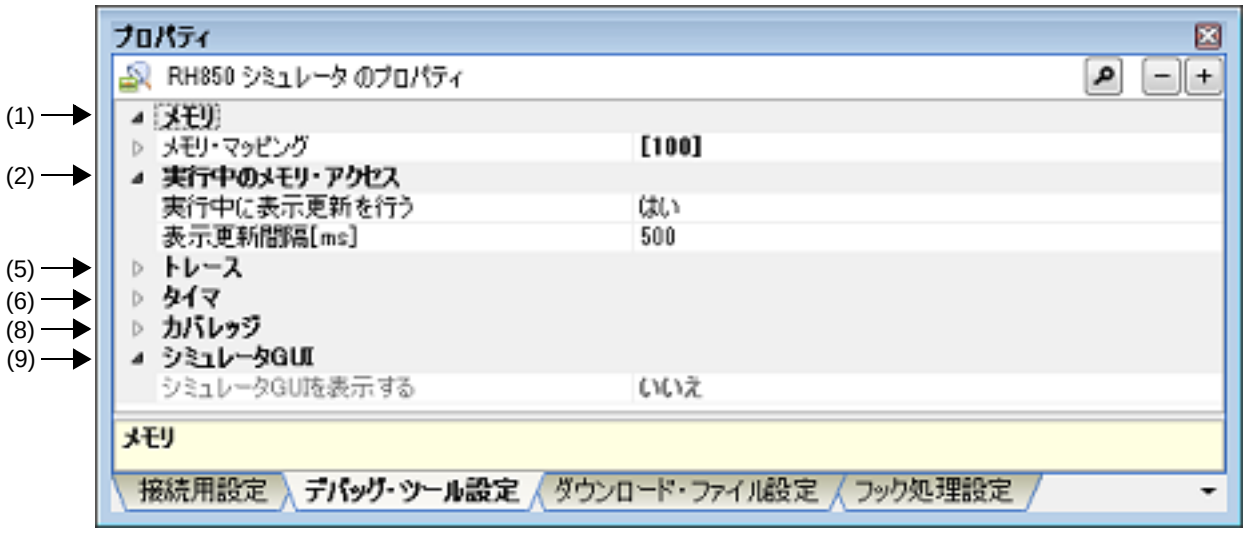
［デバッグ・ツール設定］タブでは，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。

(1) ［メモリ］
(2) ［実行中のメモリ・アクセス］
(3) ［実行中のイベント設定］【Full-spec emulator】【E1】【E20】
(4) ［ブレーク］【Full-spec emulator】【E1】【E20】
(5) ［トレース］
(6) ［タイマ］【シミュレータ】
(7) ［入力信号のマスク］【Full-spec emulator】【E1】【E20】
(8) ［カバレッジ］【シミュレータ】
(9) ［シミュレータ GUI］【シミュレータ】

図 A.9 プロパティ パネル：［デバッグ・ツール設定］タブ【Full-spec emulator】

図 A.10 プロパティ パネル：［デバッグ・ツール設定］タブ【E1】【E20】

図 A.11 プロパティ パネル：［デバッグ・ツール設定］タブ【シミュレータ】



［各カテゴリの説明］

(1) ［メモリ］

メモリに関する詳細情報の表示，および設定の変更を行います。

メモリ・マッピング	メモリ・マッピングの状況をメモリ領域の種別[注]ごとに表示します。	
	デフォルト	[*マイクロコントローラ固有のメモリ・マッピング領域種別の合計*]
	変更方法	編集不可
	表示内容	メモリ・マッピングの状況をメモリ領域の種別ごとに表示します。 なお，各メモリ種別の “+” マークをクリックすると，次の詳細情報を表示します。 - メモリ種別 - 開始アドレス - 終了アドレス

メモリ書き込み時にベリファイを行う【Full-spec emulator】【E1】【E20】	メモリ値の初期化を行う際に，ベリファイを行うか否かを選択します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	ベリファイを行います。
		いいえ	ベリファイを行いません。

注 デバイス・ファイルに登録されているメモリ・マッピング領域の種別です。

(2) ［実行中のメモリ・アクセス］
プログラム実行中のメモリ・アクセス（リアルタイム表示更新機能（「2.10.1.4 プログラム実行中にメモリの内容を表示／変更する」参照））に関する詳細情報の表示，および設定の変更を行います。

実行中にアクセスする【Full-spec emulator】【E1】【E20】	プログラム実行中に内蔵 RAM 領域にアクセスするか否かを選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	実行中に内蔵 RAM 領域にアクセスします。
		いいえ	実行中に内蔵 RAM 領域にアクセスしません。
実行中に表示更新を行う	プログラム実行中に，メモリ パネル／ウォッチ パネルの表示内容を更新するか否かを選択します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	実行中に表示を更新します。
		いいえ	実行中に表示を更新しません。
表示更新間隔 [ms]	プログラム実行中に，メモリ パネル／ウォッチ パネルの表示内容を更新する間隔を 100 ms 単位で指定します。 なお，このプロパティは，［実行中に表示更新を行う］プロパティにおいて［はい］を選択した場合のみ表示されます。		
	デフォルト	500	
	変更方法	キーボードからの直接入力	
	指定可能値	100 ～ 65500 の整数（単位：100 ms 未満の端数切り上げ）	

(3) ［実行中のイベント設定］【Full-spec emulator】【E1】【E20】
実行中のイベント設定機能に関する詳細情報の表示，および設定の変更を行います。

実行を一瞬停止してイベントを設定する	プログラム実行中，またはトレーサ／タイマ動作中には設定することができないイベント（「2.16.6.2 実行中に設定／削除可能なイベント種別」参照）を，プログラムの実行，またはトレーサ／タイマの動作を強制的に一瞬停止させることで設定を行うか否かを選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	プログラムの実行，またはトレーサ／タイマの動作を一瞬停止してイベントを設定します。
		いいえ	プログラム実行中，またはトレーサ／タイマ動作中に対象イベントを設定することはできません。

(4) ［ブレーク］【Full-spec emulator】【E1】【E20】
ブレーク機能に関する詳細情報の表示，および設定の変更を行います。

ソフトウエア・ブレークを使用する	ソフトウエア・ブレーク機能【Full-spec emulator】【E1】【E20】注を使用するか否かを選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択 ただし，プログラム実行中は変更不可	
	指定可能値	はい	ソフトウエア・ブレーク機能を使用します。
		いいえ	ソフトウエア・ブレーク機能を使用しません。
優先的に使用するブレークポイントの種類	エディタ パネル／逆アセンブル パネルにおいて，ソース行，または実行アドレスに対してマウスのワンクリック操作でブレークポイントを設定する際に，優先的に使用するブレークポイントの種別を選択します。		
	デフォルト	ソフトウエア・ブレーク	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	ソフトウエア・ブレーク	ソフトウエア・ブレークポイントを優先的に設定します。
		ハードウエア・ブレーク	ハードウエア・ブレークポイントを優先的に設定します。
停止時に周辺エミュレーションを停止する	実行停止時に，エミュレータの周辺エミュレーション機能を停止（Peripheral Break）するか否かを選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	周辺エミュレーション機能を停止します。
		いいえ	周辺エミュレーション機能を停止しません。

注 1 度ソフトウエア・ブレーク機能を使用したのち［いいえ］を選択した場合，それまで設定していたすべてのソフトウエア・ブレークポイント，および Printf イベントは無効状態となります。この場合，このプロパティを［はい］に再設定しても自動的に有効状態には戻りません（手動で設定を行う必要があります）。

(5) ［トレース］

トレース機能に関する詳細情報の表示，および設定の変更を行います（「2.12.1 トレース動作の設定をする」参照）。

注意 1. 【Full-spec emulator】【E1】【E20】
プログラム実行中は，このカテゴリ内のプロパティを変更することはできません。

注意 2. 【E1】【E20】
接続したマイクロコントローラがトレース機能を搭載していない場合，デバッグ・ツールと接続後，このカテゴリ内のプロパティは変更不可状態となります（トレース機能を使用することはできません）。

トレース・データの選択 【Full-spec emulator】 【E1】【E20】	収集するトレース・データの種類を選択します注1。		
	デフォルト	分岐命令とデータ・アクセス	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	分岐命令	プログラム実行中に発生した分岐処理の分岐元／分岐先の命令の PC 値をトレース・データとして収集します。
		データ・アクセス	プログラム実行中に成立したアクセス系イベントのデータ情報をトレース・データとして収集します。
		分岐命令とデータ・アクセス	プログラム実行中に発生した分岐処理の分岐元／分岐先の命令の PC 値，および成立したアクセス系イベントのデータ情報をトレース・データとして収集します。
トレースの優先度 【Full-spec emulator】 【E1】【E20】	トレース・データを収集する際の優先度を選択します注1。		
	デフォルト	スピード優先	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	スピード優先	リアルタイム性を優先してトレースを行います。
		データ優先	データの取りこぼしが発生しないように，CPU の実行パイプラインを一時的に停止してトレースを行います。
トレース機能を使用する 【シミュレータ】	トレース機能を使用するか否かを選択します注2。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択 ただし，プログラム実行中は変更不可	
	指定可能値	はい	トレース機能を使用します。
		いいえ	トレース機能を使用しません。
実行前にトレース・メモリをクリアする	実行前にトレース・メモリをクリアするか否かを選択します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	トレース・メモリをクリアします。
		いいえ	トレース・メモリをクリアしません。
トレース・メモリを使い切った後の動作	トレース・メモリが，収集したトレース・データで満たされた際の動作を選択します注1。		
	デフォルト	トレース・メモリを上書きし実行を続ける	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	トレース・メモリを上書きし実行を続ける	トレース・メモリを使い切ると，古いトレース・データに上書きを続けます。
		トレースを停止する注3	トレース・メモリを使い切ると，トレース・データの書き込みを停止します（実行は停止しません）。
		停止する	トレース・メモリを使い切ると，トレース・データの書き込みを中止すると同時に実行を停止します。

トレース・タイム・タグを積算する【シミュレータ】	トレース パネルに表示するトレース時間の表示方法を選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	トレースの時間表示を積算値で表示します。
		いいえ	トレースの時間表示を差分値で表示します。
トレース・タイム・タグの分周率【シミュレータ】	トレースのタイム・タグ表示（トレース パネルの［時間］表示）で使用するカウンタの分周率を選択します。 なお，分周率の設定を変更すると，タイム・タグで表示されるカウンタのカウント・アップに必要なクロック数も変更されます。		
	デフォルト	1/1	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	1/1，1/2，1/4，1/8，1/16，1/32，1/64，1/128，1/256，1/512，1/1K，1/4K，1/8K，1/16K，1/64K，1/256K，1/1M	
トレースの取得範囲設定【Full-spec emulator】【E1】【E20】	トレース・データの取得範囲を選択します。		
	デフォルト	区間をトレース	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	区間をトレース	トレース開始イベントとトレース終了イベントで指定した区間の実行履歴をトレース・データとして収集します。
		範囲外をトレース	トレース開始イベントとトレース終了イベントで指定した範囲外の実行履歴をトレース・データとして収集します。
トレース・メモリ・サイズ [フレーム]【Full-spec emulator】【シミュレータ】	トレース・データを格納するメモリ・サイズをトレース・フレーム[注4]数で選択します[注1]。		
	デフォルト	【Full-spec emulator】 8K 【シミュレータ】 4K	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	【Full-spec emulator】 8K，32K，64K，128K，256K，512K 【シミュレータ】 4K，8K，12K，16K，20K，24K，28K，32K，36K，40K，44K，48K，52K，56K，60K，64K，128K 192K，256K，320K，384K，448K，512K，576K，640K，704K，768K，832K，896K，960K，1M，2M，3M	
トレースを補完する【Full-spec emulator】	収集したトレース・データを表示する際に，補完表示を行うか否かを選択します。 補完表示を行うことにより，ハードウエアではトレースできない分岐命令間の命令を表示します。 なお，この設定は，次回取得するトレース・データより反映されます。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	トレース・データの補完表示を行います。
		いいえ	トレース・データの補完表示を行いません。

トレースの取得対象設定 【Full-spec emulator】 【E1】【E20】	トレースの対象となるコアを選択します。		
	デフォルト	デバッグ対象コアのみ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	デバッグ対象コアのみ	現在デバッグ対象に選択している PE のみを対象にトレース・データを収集します（デフォルト）。
		全てのコア	全 PE を対象にトレース・データを収集します。

注 1. 【Full-spec emulator】【E1】【E20】
このプロパティを変更すると，トレース・メモリがクリアされます。

注 2. エディタ パネル／逆アセンブル パネルにおいて，コンテキスト・メニュー→［トレース開始の設定］／［トレース終了の設定］を選択した場合，このプロパティは自動的に［はい］に変更されます。

注 3. ［トレースの優先度］プロパティで［データ優先］を選択している場合，この項目は表示されません。

注 4. トレース・フレームはトレース・データの一単位を表します。
フェッチ／ライト／リードなどで，それぞれ 1 つのトレース・フレームを使用します。

(6) ［タイマ］【シミュレータ】
タイマ機能に関する詳細情報の表示，および設定の変更を行います。

タイマ機能を使用する	タイマ機能を使用するか否かを選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	タイマ機能を使用します。
		いいえ	タイマ機能を使用しません。

(7) ［入力信号のマスク］【Full-spec emulator】【E1】【E20】
入力信号のマスクに関する詳細情報の表示，および設定の変更を行います。

WAIT 信号をマスクする	WAIT 信号をエミュレータに入力しないようにマスクするか否かを選択します。		
	デフォルト	【Full-spec emulator】 はい 【E1】【E20】 いいえ	
	変更方法	ドロップダウン・リストによる選択[注]	
	指定可能値	はい	WAIT 信号をマスクします。
		いいえ	WAIT 信号をマスクしません。
RESET 信号をマスクする	RESET 信号をエミュレータに入力しないようにマスクするか否かを選択します。		
	デフォルト	【Full-spec emulator】 はい 【E1】【E20】 いいえ	
	変更方法	ドロップダウン・リストによる選択[注]	
	指定可能値	はい	RESET 信号をマスクします。
		いいえ	RESET 信号をマスクしません。

マスクする RESET 信号の選択	マスクする RESET 信号を選択します。 なお，このプロパティは，［RESET 信号をマスクする］プロパティにおいて［はい］を選択した場合のみ表示されます。		
	デフォルト	【Full-spec emulator】 TARGET RESET 信号 【E1】【E20】 TARGET RESET 信号と INTERNAL RESET 信号	
	変更方法	【Full-spec emulator】 ドロップダウン・リストによる選択 【E1】【E20】 変更不可	
	指定可能値	TARGET RESET 信号	TARGET RESET 信号をマスクします。
		TARGET RESET 信号と INTERNAL RESET 信号	TARGET RESET 信号と INTERNAL RESET 信号をマスクします。

注 【Full-spec emulator】
［接続用設定］タブ上の［ターゲット・ボードとの接続］【Full-spec emulator】【E1】【E20】カテゴリ内［ターゲット・ボードを接続している］プロパティを［いいえ］に指定している場合，このプロパティは，デバッグ・ツールとの接続時に自動的に［はい］に固定されます（変更不可）。

(8) ［カバレッジ］【シミュレータ】
カバレッジ機能に関する詳細情報の表示，および設定の変更を行います。

カバレッジ機能を使用する	カバレッジ機能を使用するか否かを選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	カバレッジ機能を使用します。
		いいえ	カバレッジ機能を使用しません。
カバレッジ結果を再利用する	デバッグ・ツールと接続時／切断時に，カバレッジ測定結果のロード／セーブを行うか否かを選択します。 なお，このプロパティは，［カバレッジ機能を使用する］プロパティにおいて［はい］を選択した場合のみ表示されます。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	カバレッジ測定結果のロード／セーブを行います。
		いいえ	カバレッジ測定結果のロード／セーブを行いません。
カバレッジ測定領域 (1M バイト)	内蔵 ROM 領域以外のカバレッジ測定の対象領域を指定します。 カバレッジ測定を行う任意の 1M バイト空間の開始アドレスを指定します。 なお，このプロパティは，［カバレッジ機能を使用する］プロパティにおいて［はい］を選択した場合のみ表示されます。		
	デフォルト	100000	
	変更方法	キーボードからの直接入力	
	指定可能値	内蔵 ROM 領域以外のアドレス（シンボル指定不可）	

(9) ［シミュレータ GUI］【シミュレータ】
シミュレータ GUI 機能に関する詳細情報の表示，および設定の変更を行います。

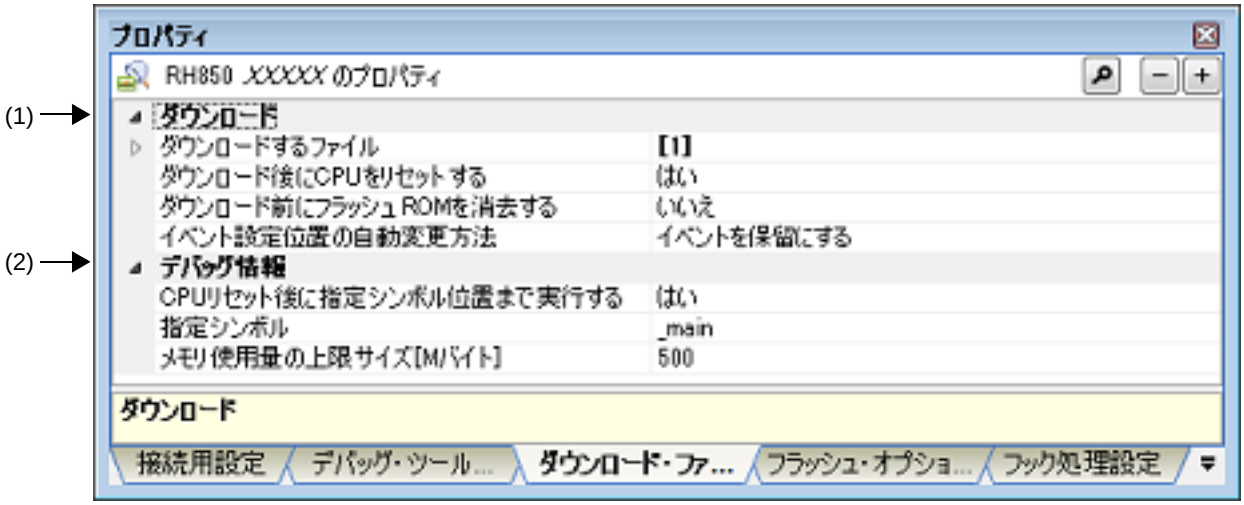
注意 選択しているマイクロコントローラのシミュレータが周辺機能シミュレーションをサポートしていない（命令シミュレーション版）場合，このカテゴリ内のプロパティはすべて無効となります。

シミュレータ GUI を表示する	シミュレータ GUI の機能を使用するため，シミュレータ GUI ウインドウを表示するか否かを選択します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択 ただし，プログラム実行中は変更不可	
	指定可能値	はい	シミュレータ GUI ウインドウを表示します。
		いいえ	シミュレータ GUI ウインドウを表示しません。
実行開始時に最前面表示する	プログラムの実行開始時に，シミュレータ GUI ウインドウを最前面に表示するか否かを選択します。 なお，このプロパティは，［シミュレータ GUI を表示する］プロパティにおいて［はい］を選択した場合のみ表示されます。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	最前面に表示します。
		いいえ	最前面に表示しません。

［ダウンロード・ファイル設定］タブ

［ダウンロード・ファイル設定］タブでは，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。
なお，ダウンロード方法については，「2.5 ダウンロード／アップロード」を参照してください。

図 A.12 プロパティ パネル：［ダウンロード・ファイル設定］タブ



［各カテゴリの説明］

(1) ［ダウンロード］
ダウンロードに関する詳細情報の表示，および設定の変更を行います。

ダウンロードするファイル	ダウンロードするファイルを指定します[注1]。 サブプロパティとして，ダウンロードするファイル名，およびダウンロード条件を下段に展開表示します。		
	デフォルト	[*ダウンロードするファイルの数*]	
	変更方法	ダウンロード・ファイル ダイアログによる選択 ダウンロード・ファイル ダイアログは，このプロパティを選択すると欄内右端に表示される［...］ボタンをクリックすることでオープンします（このパネル上でダウンロード・ファイルを指定することはできません）。	
ダウンロード後に CPU をリセットする	ダウンロード後に CPU をリセットするか否かを選択します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	ダウンロード後に CPU をリセットします。
		いいえ	ダウンロード後に CPU をリセットしません。
ダウンロード前にフラッシュ ROM を消去する 【Full-spec emulator】【E1】【E20】	ダウンロード前にフラッシュ ROM を消去するか否かを選択します。		
	デフォルト	いいえ	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	ダウンロード前にフラッシュ ROM を消去します。
		いいえ	ダウンロード前にフラッシュ ROM を消去しません。

イベント設定位置の自動変更方法	再ダウンロードすることにより，現在設定されているイベントの設定位置（アドレス）が命令の途中になる場合の再設定方法を選択します[注 2]。		
	デフォルト	イベントを保留にする	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	命令の先頭に移動する	命令の先頭アドレスにイベントを再設定します。
		イベントを保留にする	対象イベントを保留状態にします。

注 1. メイン・プロジェクト／サブプロジェクトでビルド対象に指定しているファイルは，ダウンロードの対象ファイルから削除することはできません（デフォルトで自動的にダウンロード・ファイルとして登録されます）。なお，ダウンロード可能なファイル形式については，「表 2.1 ダウンロード可能なファイル」を参照してください。

注 2. デバッグ情報がないイベント設定位置のみが対象となります。デバッグ情報がある場合のイベント設定位置は，常にソース・テキスト行の先頭に移動します。

(2) ［デバッグ情報］

デバッグ情報に関する詳細情報の表示，および設定の変更を行います。

CPU リセット後に指定シンボル位置まで実行する	CPU リセット後に，プログラムを指定シンボル位置まで実行するか否かを選択します。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	プログラムを指定シンボル位置まで実行します。
		いいえ	CPU リセット後にプログラムを実行しません。
指定シンボル	CPU リセット後に，プログラムを実行して停止する位置を指定します。 なお，このプロパティは，［CPU リセット後に指定シンボル位置まで実行する］プロパティにおいて［はい］を選択した場合のみ表示されます。		
	デフォルト	_main	
	変更方法	キーボードからの直接入力	
	指定可能値	0 ～ “ *アドレス空間の終了アドレス* ” のアドレス式	
メモリ使用量の上限サイズ [M バイト]	デバッグ情報の読み込みで使用するメモリ・サイズの上限値を指定します[注]。		
	デフォルト	500	
	変更方法	キーボードからの直接入力	
	指定可能値	100 ～ 1000 の 10 進数値	

注 上限値を小さくした場合，デバッグ情報の破棄と再読み込みが頻繁に行われるため，デバッグ・ツールの応答性が低下する場合があります。

［フラッシュ・オプション設定］タブ【Full-spec emulator】【E1】【E20】

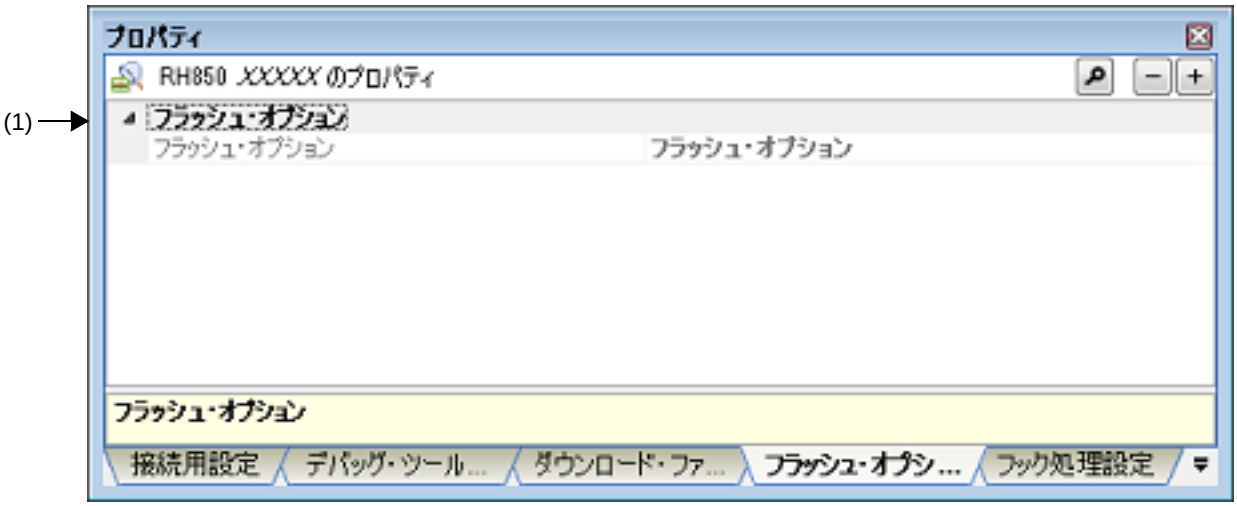
［フラッシュ・オプション設定］タブでは，マイクロコントローラに搭載されているフラッシュ・メモリのためのオプション設定を行います。
ただし，このタブは，選択しているマイクロコントローラがフラッシュ・オプションに対応している場合のみ表示されます。

注意 1. デバッグ・ツールと接続している場合のみ，このタブの設定を行うことができます。

注意 2. このタブの設定を変更すると，選択しているマイクロコントローラの種類により CPU リセットが発生する場合があります。

(1) ［フラッシュ・オプション］

図 A.13 プロパティ パネル：［フラッシュ・オプション設定］タブ



［各カテゴリの説明］

(1) ［フラッシュ・オプション］
フラッシュ・オプションに関する詳細情報の表示，および設定の変更を行います。

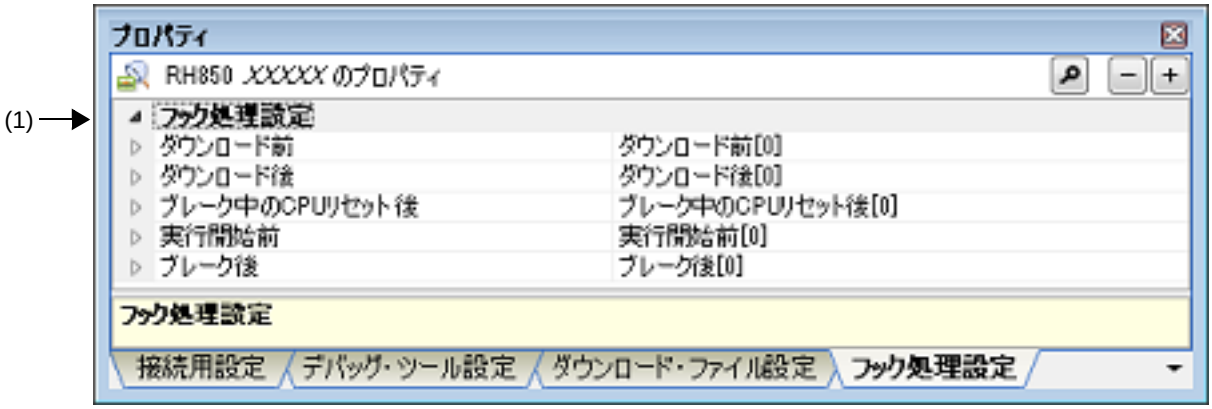
フラッシュ・オプション	フラッシュ・メモリのオプションを指定します。	
	デフォルト	フラッシュ・オプション
	変更方法	フラッシュ・オプションの設定 ダイアログ【Full-spec emulator】【E1】【E20】による指定 フラッシュ・オプションの設定 ダイアログは，このプロパティを選択すると欄内右端に表示される［...］ボタンをクリックすることでオープンします（このパネル上でフラッシュ・メモリのオプションを指定することはできません）。 なお，設定したフラッシュ・メモリのオプションの内容は，このパネル上では表示されません。

［フック処理設定］タブ

［フック処理設定］タブでは，次に示すカテゴリごとに詳細情報の表示，および設定の変更を行います。
なお，フック処理の設定方法については，「2.17 フック処理を設定する」を参照してください。

(1) ［フック処理設定］

図 A.14 プロパティ パネル：［フック処理設定］タブ



［各カテゴリの説明］

(1) ［フック処理設定］
フック処理に関する詳細情報の表示，および設定の変更を行います。

ダウンロード前	ロード・モジュール・ファイルをダウンロードする直前に行う処理[注]を指定します。	
	デフォルト	ダウンロード前 [0]（“[ ]” 内は現在の指定処理数を示す）
	変更方法	テキスト編集 ダイアログによる指定 テキスト編集 ダイアログは，このプロパティを選択すると欄内右端に表示される［...］ボタンをクリックすることでオープンします（このパネル上で処理を指定することはできません）。
	指定可能値	128 個までの処理（テキスト編集 ダイアログ上の 1 行が 1 処理に相当） ただし，1 処理につき 64 文字まで入力可
ダウンロード後	ロード・モジュール・ファイルをダウンロードした直後に行う処理[注]を指定します。	
	デフォルト	ダウンロード前 [0]（“[ ]” 内は現在の指定処理数を示す）
	変更方法	テキスト編集 ダイアログによる指定 テキスト編集 ダイアログは，このプロパティを選択すると欄内右端に表示される［...］ボタンをクリックすることでオープンします（このパネル上で処理を指定することはできません）。
	指定可能値	128 個までの処理（テキスト編集 ダイアログ上の 1 行が 1 処理に相当） ただし，1 処理につき 64 文字まで入力可
ブレーク中の CPU リセット後	ブレーク中の CPU リセット直後に行う処理[注]を指定します。	
	デフォルト	ダウンロード前 [0]（“[ ]” 内は現在の指定処理数を示す）
	変更方法	テキスト編集 ダイアログによる指定 テキスト編集 ダイアログは，このプロパティを選択すると欄内右端に表示される［...］ボタンをクリックすることでオープンします（このパネル上で処理を指定することはできません）。
	指定可能値	128 個までの処理（テキスト編集 ダイアログ上の 1 行が 1 処理に相当） ただし，1 処理につき 64 文字まで入力可

実行開始前	プログラムの実行開始直前に行う処理[注]を指定します。	
	デフォルト	ダウンロード前 [0]（“[ ]” 内は現在の指定処理数を示す）
	変更方法	テキスト編集 ダイアログによる指定 テキスト編集 ダイアログは，このプロパティを選択すると欄内右端に表示される［...］ボタンをクリックすることでオープンします（このパネル上で処理を指定することはできません）。
	指定可能値	128 個までの処理（テキスト編集 ダイアログ上の 1 行が 1 処理に相当） ただし，1 処理につき 64 文字まで入力可
ブレーク後	プログラムの実行がブレークした直後に行う処理[注]を指定します。	
	デフォルト	ダウンロード前 [0]（“[ ]” 内は現在の指定処理数を示す）
	変更方法	テキスト編集 ダイアログによる指定 テキスト編集 ダイアログは，このプロパティを選択すると欄内右端に表示される［...］ボタンをクリックすることでオープンします（このパネル上で処理を指定することはできません）。
	指定可能値	128 個までの処理（テキスト編集 ダイアログ上の 1 行が 1 処理に相当） ただし，1 処理につき 64 文字まで入力可

注 次の 3 つの処理の中から目的に応じた処理の指定形式をテキスト編集 ダイアログに入力します。

【処理 1】

*I/O レジスタ*の内容を，*数値*に自動的に書き換えます。

指定形式：

```
I/O レジスタ名 数値
```

【処理 2】

*CPU レジスタ*の内容を，*数値*に自動的に書き換えます。

指定形式：

```
CPU レジスタ名 数値
```

【処理 3】

Python スクリプト・パス（絶対パス／プロジェクト・フォルダを基点とした相対パス）で指定したスクリプト・ファイルを実行します。

指定形式：

```
Source Python スクリプト・パス
```

メモリ パネル

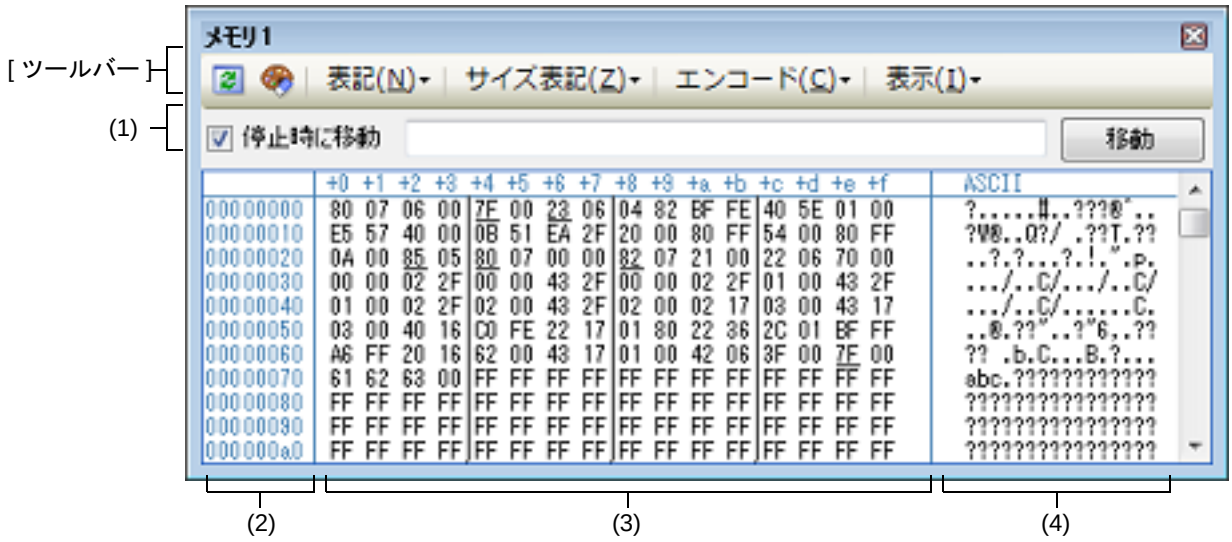
メモリの内容の表示，変更を行います（「2.10.1 メモリを表示／変更する」参照）。
このパネルは，最大 4 個までオープンすることができます。各パネルは，タイトルバーの“ メモリ 1”，“ メモリ 2”，“ メモリ 3”，“ メモリ 4” の名称で識別されます。
プログラムの実行後，メモリの値が変化すると表示を自動的に更新します（ステップ実行時には，ステップ実行ごとに表示を逐次更新）。
また，リアルタイム表示更新機能を有効にすることにより，プログラム実行中であっても，値の表示をリアルタイムに更新することも可能です。
なお，このパネルは，デバッグ・ツールと接続時のみオープンすることができます。

注意 データフラッシュ領域のメモリ値を変更すると，選択しているマイクロコントローラの種別により，CPU リセットを発生する場合があります。

備考 1. ツールバーの 100% ，または［Ctrl］キーを押下しながらマウス・ホイールを前後方に動かすことにより，本パネルの表示を拡大／縮小することができます。

備考 2. ツールバーの［表示］→ ボタンをクリックすることによりオープンするスクロール範囲設定 ダイアログにより，このパネルの垂直スクロール・バーのスクロール範囲を設定することができます。

図 A.15 メモリ パネル



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［ツールバー］
- ［［ファイル］メニュー（メモリ パネル専用部分）］
- ［［編集］メニュー（メモリ パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］メニュー→［メモリ］→［メモリ *1* ～ *4*］を選択

［各エリアの説明］

(1) 表示位置指定エリア
アドレス式を指定することにより，メモリ値の表示開始位置を指定することができます。
次の指定を順次行います。

(a) アドレス式の指定
表示したいメモリ値のアドレスとなるアドレス式をテキスト・ボックスに直接入力します。最大 1024 文字までの入力式を指定することができ，その計算結果を表示開始位置アドレスとして扱います。
ただし，マイクロコントローラのアドレス空間よりも大きいアドレス式を指定することはできません。

備考 1. このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

備考 2. 指定したアドレス式がシンボルを表現し，サイズが判明する場合では，そのシンボルの先頭アドレスから終了アドレスまでを選択状態で表示します。

(b) アドレス式の自動／手動評価の指定
表示開始位置を変更するタイミングは，［停止時に移動］チェック・ボックスの指定，および［移動］ボタンにより決定します。

［停止時に移動］	☑	プログラム停止後，自動的にアドレス式の評価を行い，その計算結果のアドレスにキャレットが移動します。
	☐	プログラム停止後，アドレス式の評価を自動的に行いません。 この場合，［移動］ボタンをクリックすることにより，アドレス式の評価を行います。
［移動］		［停止時に移動］チェック・ボックスのチェックをしなかった場合，このボタンをクリックすることによりアドレス式の評価を行い，その計算結果のアドレスにキャレットが移動します。

(2) アドレス・エリア
メモリのアドレスを表示します（16 進数表記固定）。
デフォルトで，0x0 番地より表示を開始します。
ただし，コンテキスト・メニューの［表示アドレス・オフセット値を設定 ...］を選択することでオープンするアドレス・オフセット設定 ダイアログにより，開始アドレスにオフセット値を設定することができます。
アドレス幅は，プロジェクトで指定しているマイクロコントローラのメモリ空間のアドレス幅となります。
このエリアを編集することはできません。

注意 設定されたオフセット値は，メモリ値エリアの表示桁数の指定に従って自動的に変更されます。

(3) メモリ値エリア
メモリ値を表示／変更します。
メモリ値の表示進数／表示幅／表示桁の指定は，ツールバーのボタン，またはコンテキスト・メニューの［表記］／［サイズ表記］／［表示］の選択により行います（「2.10.1.2 値の表示形式を変更する」参照）。
メモリ値として表示されるマークや色の意味は次のとおりです（表示の際の文字色／背景色はオプション ダイアログにおける［全般 - フォントと色］カテゴリの設定に依存）。

表示例（デフォルト）			説明
00	文字色	青	ユーザにより，値が変更されているメモリ値（［Enter］キーによりターゲット・メモリに書き込まれます）
	背景色	標準色	
00 （下線）	文字色	標準色	シンボルが定義されているアドレスのメモリ値（ウォッチ式の登録を行うことができます）
	背景色	標準色	
00	文字色	茶色	プログラムの実行により，値が変化したメモリ値[注] ツールバーの  ボタンをクリックすると，強調表示をリセットします。
	背景色	クリーム	
00	文字色	ピンク	リアルタイム表示更新機能を行っているメモリ値
	背景色	標準色	

表示例（デフォルト）			説明	
00	文字色	標準色	リード／フェッチ	リアルタイム表示更新機能を行っている場合，現在のメモリ値のアクセス状態
	背景色	薄緑		
00	文字色	標準色	ライト	
	背景色	オレンジ		
00	文字色	標準色	リードとライト	
	背景色	薄青		
00	文字色	グレー	リード不可の領域のメモリ値	
	背景色	標準色		
??	文字色	グレー	メモリ・マッピングされていない領域	
	背景色	標準色		
--	文字色	グレー	書き換え不可能領域（I/O レジスタ領域 /I/O 保護領域など），またはメモリ値の取得に失敗した場合	
	背景色	標準色		
**	文字色	標準色	プログラム実行中に，リアルタイム表示更新領域以外の領域を表示指定した場合，またはメモリ値の取得に失敗した場合	
	背景色	標準色		

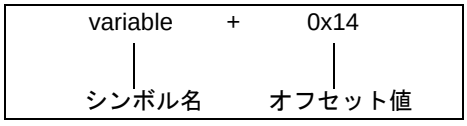
注　プログラム実行直前において，メモリ パネルで表示されていたアドレス範囲のメモリ値のみが対象となります。また，プログラムの実行前後での値の比較であるため，実行結果が同一値となった場合は強調表示を行いません。

注意　このエリアの表示桁数は，コンテキスト・メニューの［サイズ表記］の指定に従って自動的に変更します。

このエリアは，次の機能を備えています。

(a)　ポップアップ表示

メモリ値にマウス・カーソルを重ねることにより，マウス・カーソルが指しているアドレスに対して前方向に存在する一番近いシンボルを基準にして，次の内容をポップアップ表示します。
ただし，シンボル情報が存在しない場合（下線が非表示の場合）はポップアップ表示は行いません。



シンボル名	シンボル名を表示します。
オフセット値	アドレスにシンボルが定義されていない場合は，前方向に存在する一番近いシンボルからのオフセット値を表示します（16 進数表示固定）。

(b)　リアルタイム表示更新機能

リアルタイム表示更新機能を使用することにより，プログラムが停止している状態の時だけでなく，実行中の状態であっても，メモリ値の表示／変更を行うことができます。
リアルタイム表示更新機能についての詳細は，「2.10.1.4 プログラム実行中にメモリの内容を表示／変更する」を参照してください。

(c)　メモリ値の変更

メモリ値の変更は，対象メモリ値にキャレットを移動したのち，直接キーボードより編集することで行います。
メモリ値を編集すると変更箇所の表示色が変化し，この状態で［Enter］キーを押下することにより変更した値がターゲット・メモリに書き込まれます（［Enter］キーの押下前に［Esc］キーを押下すると編集をキャンセルします）。
メモリ値の変更方法についての詳細は，「2.10.1.3 メモリの内容を変更する」を参照してください。

(d) メモリ値の検索／初期化
コンテキスト・メニューの［検索 ...］を選択することにより，指定したアドレス範囲のメモリ内容を検索するためのメモリ検索 ダイアログをオープンします（「2.10.1.5 メモリの内容を検索する」参照）。
また，コンテキスト・メニューの［初期化 ...］を選択することにより，指定したアドレス範囲のメモリ内容を一括して変更するためのメモリ初期化 ダイアログをオープンします（「2.10.1.6 メモリの内容を一括して変更（初期化）する」参照）。

(e) コピー／貼り付け
メモリ値をマウスにより範囲選択することで，その箇所の内容を文字列としてクリップ・ボードにコピーすることができ，その内容をキャレット位置に貼り付けることができます。
これらの操作は，コンテキスト・メニューの選択，または［編集］メニューの選択により行います。
ただし，貼り付け操作は，貼り付け対象の文字列とそのエリアの表示形式（表示進数／ビット幅）が一致する場合のみ可能です（表示形式が一致しない場合は，メッセージを表示します）。
なお，このエリアで扱うことができる文字コードと文字列は次のとおりです（これ以外の文字列を貼り付けた場合は，メッセージを表示します）。

文字コード	ASCII
文字列	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, A, B, C, D, E, F

(f) ウォッチ式の登録
シンボルが定義されているアドレスでは，メモリ値に下線が表示され，ウォッチ式として登録可能であることを示します。
このメモリ値を選択，またはメモリ値のいずれかにキャレットを置いた状態で，コンテキスト・メニューの［ウォッチ 1 に登録］を選択することにより，指定したアドレスのシンボル名がウォッチ式としてウォッチ パネル（ウォッチ 1）に登録されます。

注意 下線表示のないメモリ値をウォッチ式に登録することはできません。

(g) メモリ値の保存
［ファイル］メニュー→［名前を付けてメモリ・データを保存 ...］を選択することにより，データ保存 ダイアログをオープンし，このパネルの内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
メモリ値の保存方法についての詳細は，「2.10.1.7 メモリの表示内容を保存する」を参照してください。

(4) 文字列エリア
メモリの値を文字コードに変換して表示します。
文字コードの指定は，ツールバー／コンテキスト・メニューの［エンコード］の選択により行います（デフォルトでは ASCII コードで表示します）。
また，このエリアでは，メモリ値を浮動小数点数値に変換して文字列として表示することができます。この場合は，コンテキスト・メニューの［エンコード］より次の項目の選択を行ってください。

項目	表記		サイズ
Float	単精度浮動小数点数		32 ビット
	数値	＜*符号*＞＜*仮数部*＞ e ＜*符号*＞＜*指数部*＞	
	無限大	Inf，および -Inf	
	非数	NaN	
	例	+ 1.234567e+123	
Double	倍精度浮動小数点数		64 ビット
	数値	＜*符号*＞＜*仮数部*＞ e ＜*符号*＞＜*指数部*＞	
	無限大	Inf，および -Inf	
	非数	NaN	
	例	+ 1.2345678901234e+123	
Float Complex	単精度浮動小数点数の複素数		64 ビット
	＜*単精度浮動小数点数*＞＜*単精度浮動小数点数*＞ * I		

項目	表記	サイズ
Double Complex	倍精度浮動小数点数の複素数	128 ビット
	＜倍精度浮動小数点数＞ ＜倍精度浮動小数点数＞ * I	
Float Imaginary	単精度浮動小数点数の虚数	32 ビット
	＜単精度浮動小数点数＞ * I	
Double Imaginary	倍精度浮動小数点数の虚数	64 ビット
	＜倍精度浮動小数点数＞ * I	

注意 指定されている文字コード，または浮動小数点数値の最小サイズが " 表示バイト数 × 表示桁数バイト数 " より大きい場合，このエリアには何も表示されません。

このエリアは，次の機能を備えています。

(a) 文字列の変更
文字列の変更は，対象文字列にキャレットを移動したのち，直接キーボードより編集することで行います。
文字列を編集すると変更箇所の表示色が変化し，この状態で［Enter］キーの押下することにより変更した値がターゲット・メモリに書き込まれます（［Enter］キーの押下前に［Esc］キーを押下すると編集をキャンセルします）。

注意 浮動小数点数値表示している文字列を変更することはできません。

(b) 文字列の検索
コンテキスト・メニューの［検索 ...］を選択することにより，文字列を検索するためのメモリ検索 ダイアログをオープンします（「2.10.1.5 メモリの内容を検索する」参照）。

(c) コピー／貼り付け
文字列をマウスにより範囲選択することで，その箇所の内容を文字列としてクリップ・ボードにコピーすることができ，その内容をキャレット位置に貼り付けることができます。
これらの操作は，コンテキスト・メニューの選択，または［編集］メニューの選択により行います。
ただし，貼り付け操作は，文字コードとして ASCII が指定されている場合のみ可能です（ASCII 以外が指定されている場合は，メッセージを表示します）。

［ツールバー］

	デバッグ・ツールから最新の情報を取得し，表示を更新します。
	プログラム実行により値が変化した箇所を示す強調表示をリセットします。 ただし，プログラム実行中は無効となります。
表記	メモリ値の表示形式を変更する次のボタンを表示します。
 16 進数	メモリ値を 16 進数で表示します（デフォルト）。
 符号付き 10 進数	メモリ値を符号付き 10 進数で表示します。
 符号無し 10 進数	メモリ値を符号なし 10 進数で表示します。
 8 進数	メモリ値を 8 進数で表示します。
 2 進数	メモリ値を 2 進数で表示します。
サイズ表記	メモリ値のサイズの表示形式を変更する次のボタンを表示します。

4 ビット	メモリ値を 4 ビット幅で表示します。
1 バイト	メモリ値を 8 ビット幅で表示します（デフォルト）。
2 バイト	メモリ値を 16 ビット幅で表示します。 対象メモリ領域のエンディアンに従って値を変換します。
4 バイト	メモリ値を 32 ビット幅で表示します。 対象メモリ領域のエンディアンに従って値を変換します。
8 バイト	メモリ値を 64 ビット幅で表示します。 対象メモリ領域のエンディアンに従って値を変換します。
エンコード	文字列のエンコードを変更する次のボタンを表示します。
ASCII	文字列を ASCII コードで表示します（デフォルト）。
Shift_JIS	文字列を Shift_JIS コードで表示します。
EUC-JP	文字列を EUC-JP コードで表示します。
UTF-8	文字列を UTF-8 コードで表示します。
UTF-16	文字列を UTF-16 コードで表示します。
Float	文字列を単精度浮動小数点数値で表示します。
Double	文字列を倍精度浮動小数点数値で表示します。
Float Complex	文字列を単精度浮動小数点数の複素数で表示します。
Double Complex	文字列を倍精度浮動小数点数の複素数で表示します。
Float Imaginary	文字列を単精度浮動小数点数の虚数で表示します。
Double Imagimary	文字列を倍精度浮動小数点数の虚数で表示します。
表示	表示形式を変更する次のボタンを表示します。
スクロール範囲を設定 ...	スクロール範囲を設定するためのスクロール範囲設定 ダイアログがオープンします。
表示桁数を設定	メモリ値エリアの表示桁数を設定するため，表示桁数設定 ダイアログをオープンします。
表示アドレス・オフセット値を設定	アドレス・エリアに表示するアドレスのオフセット値を設定するため，アドレス・オフセット設定 ダイアログをオープンします。

[[ファイル]メニュー（メモリ パネル専用部分）]

メモリ パネル専用の［ファイル］メニューは次のとおりです（その他の項目は共通）。
ただし，プログラム実行中はすべて無効となります。

メモリ・データを保存	メモリの内容を前回保存したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存します（「(g) メモリ値の保存」参照）。 なお，起動後に初めてこの項目を選択した場合は，［名前を付けてメモリ・データを保存 ...］の選択と同等の動作となります。
名前を付けてメモリ・データを保存 ...	メモリの内容を指定したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存するために，データ保存 ダイアログをオープンします（「(g) メモリ値の保存」参照）。

[[編集]メニュー（メモリ パネル専用部分）]

メモリ パネル専用の［編集］メニューは次のとおりです（その他の項目はすべて無効）。
ただし，プログラム実行中はすべて無効となります。

コピー	選択している範囲を文字列としてクリップ・ボードにコピーします。
貼り付け	クリップ・ボードにコピーされている文字列をキャレット位置に貼り付けます。 メモリ値エリアに貼り付ける場合:「(e) コピー／貼り付け」参照 文字列エリアに貼り付ける場合:「(c) コピー／貼り付け」参照
検索 ...	メモリ検索 ダイアログをオープンします。 検索対象となる箇所は，メモリ値エリアと文字列エリアのうち，キャレットのあるエリア内となります。

［コンテキスト・メニュー］

ウォッチ 1 に登録	選択しているメモリ値に定義されているシンボルをウォッチ パネル（ウォッチ 1）に登録します。 ウォッチ式として登録される際は変数名として登録されるため，スコープにより表示されるシンボル名は変化します。 ただし，キャレット位置のメモリ値に対応するアドレスにシンボルが定義されていない場合は無効となります（「(f) ウォッチ式の登録」参照）。
検索 ...	メモリ検索 ダイアログをオープンします。 検索対象となる箇所は，メモリ値エリアと文字列エリア（浮動小数点数値表示を選択している場合を除く）のうち，キャレットのあるエリア内となります。 ただし，プログラム実行中は無効となります。
初期化 ...	メモリ初期化 ダイアログをオープンします。
最新の情報に更新	デバッグ・ツールから最新の情報を取得し，表示を更新します。
コピー	選択している範囲を文字列としてクリップ・ボードにコピーします。 ただし，プログラム実行中は無効となります。
貼り付け	クリップ・ボードにコピーされている文字列をキャレット位置に貼り付けます。 ただし，プログラム実行中は無効となります。 メモリ値エリアに貼り付ける場合:「(e) コピー／貼り付け」参照 文字列エリアに貼り付ける場合:「(c) コピー／貼り付け」参照
表記	メモリ値エリアの表示進数を指定するため，次のカスケード・メニューを表示します。
16 進数	メモリ値を 16 進数で表示します（デフォルト）。
符号付き 10 進数	メモリ値を符号付き 10 進数で表示します。
符号無し 10 進数	メモリ値を符号なし 10 進数で表示します。
8 進数	メモリ値を 8 進数で表示します。
2 進数	メモリ値を 2 進数で表示します。
サイズ表記	メモリ値エリアの表示幅を指定するため，次のカスケード・メニューを表示します。
4 ビット	メモリ値を 4 ビット幅で表示します。
1 バイト	メモリ値を 8 ビット幅で表示します（デフォルト）。
2 バイト	メモリ値を 16 ビット幅で表示します。 対象メモリ領域のエンディアンに従って値を変換します。
4 バイト	メモリ値を 32 ビット幅で表示します。 対象メモリ領域のエンディアンに従って値を変換します。
8 バイト	メモリ値を 64 ビット幅で表示します。 対象メモリ領域のエンディアンに従って値を変換します。

エンコード	文字列エリアの表示形式を指定するため，次のカスケード・メニューを表示します。
ASCII	文字列を ASCII コードで表示します（デフォルト）。
Shift_JIS	文字列を Shift_JIS コードで表示します。
EUC-JP	文字列を EUC-JP コードで表示します。
UTF-8	文字列を UTF-8 コードで表示します。
UTF-16	文字列を UTF-16 コードで表示します。
Float	文字列を単精度浮動小数点数値で表示します。
Double	文字列を倍精度浮動小数点数値で表示します。
Float Complex	文字列を単精度浮動小数点数の複素数で表示します。
Double Complex	文字列を倍精度浮動小数点数の複素数で表示します。
Float Imaginary	文字列を単精度浮動小数点数の虚数で表示します。
Double Imaginary	文字列を倍精度浮動小数点数の虚数で表示します。
表示	表示形式を変更するため，次のカスケード・メニューを表示します。
スクロール範囲を設定 ...	スクロール範囲を設定するため，スクロール範囲設定 ダイアログをオープンします。
表示桁数を設定 ...	メモリ値エリアの表示桁数を設定するため，表示桁数設定 ダイアログをオープンします。
表示アドレス・オフセット値を設定 ...	アドレス・エリアに表示するアドレスのオフセット値を設定するため，アドレス・オフセット設定 ダイアログをオープンします。
強調表示	チェックすることにより，プログラムの実行により値が変更されたメモリ値を強調表示します（デフォルト）。 ただし，プログラム実行中は無効となります。
リアルタイム表示更新設定	リアルタイム表示更新設定のため，次のカスケード・メニューを表示します（「(b) リアルタイム表示更新機能」参照）。
リアルタイム表示更新全体設定	リアルタイム表示更新機能の全般設定を行うため，プロパティ パネルをオープンします。

逆アセンブル パネル

メモリ内容を逆アセンブルした結果（逆アセンブル・テキスト）の表示，ライン・アセンブル（「2.6.4 ライン・アセンブルを行う」参照），命令レベル・デバッグ（「2.8.3 プログラムをステップ実行する」参照），およびコード・カバレッジ測定結果の表示【シミュレータ】（「2.14 カバレッジの測定【シミュレータ】」参照）を行います。

このパネルは，最大 4 個までオープンすることができます。各パネルは，タイトルバーの“逆アセンブル 1”，“逆アセンブル 2”，“逆アセンブル 3”，“逆アセンブル 4”の名称で識別されます。

混合表示モードを選択することにより，コード・データに対応するソース・ファイル中のソース・テキストも表示することができます（デフォルト）。

なお，このパネルは，デバッグ・ツールと接続時のみオープンすることができます。

注意 このパネルにフォーカスがある状態でプログラムをステップ実行した場合，実行単位は命令レベル単位となります（「2.8.3 プログラムをステップ実行する」参照）。

備考 1. ツールバーの［表示］→ ボタンをクリックするでオープンするスクロール範囲設定 ダイアログにより，このパネルの垂直スクロール・バーのスクロール範囲を設定することができます。

備考 2. ［ファイル］メニュー→［印刷 ...］を選択することにより，現在このパネルで表示しているの画像イメージを印刷することができます。

備考 3. ツールバーの 100% ，または［Ctrl］キーを押下しながらマウス・ホイールを前後方に動かすことにより，本パネルの表示を拡大／縮小することができます。

図 A.16 逆アセンブル パネル（混合表示モードの場合）

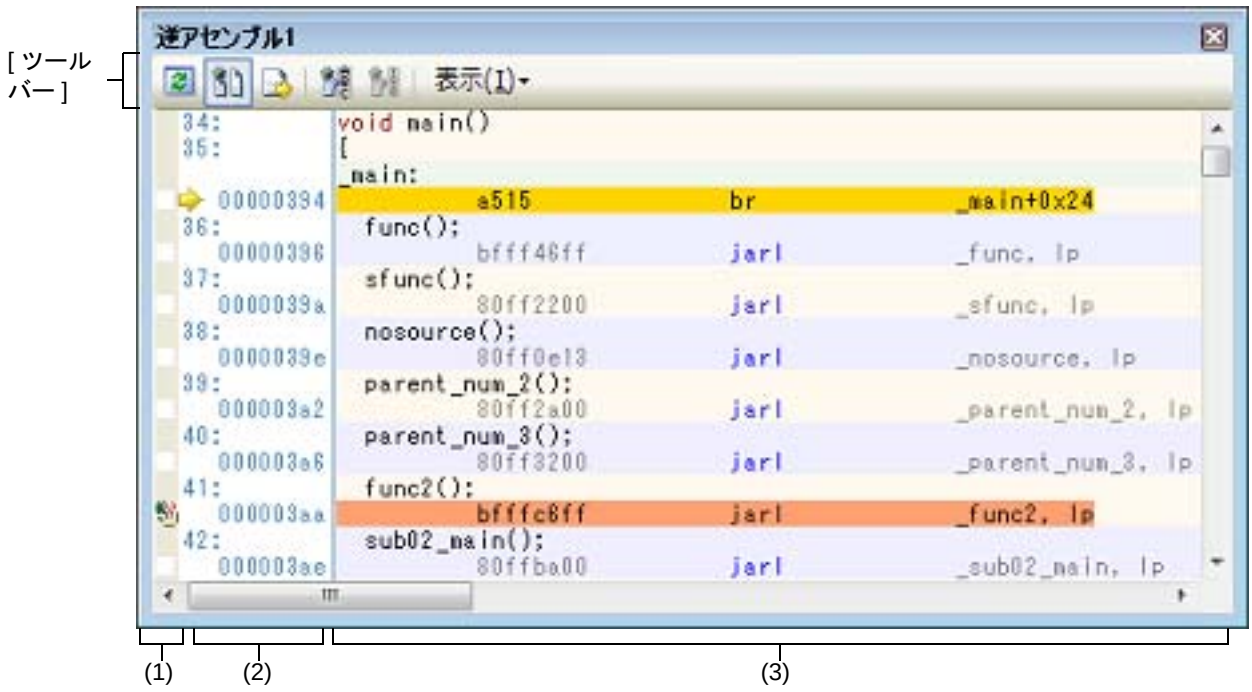


図 A.17 逆アセンブル パネル（逆アセンブル表示モードの場合）

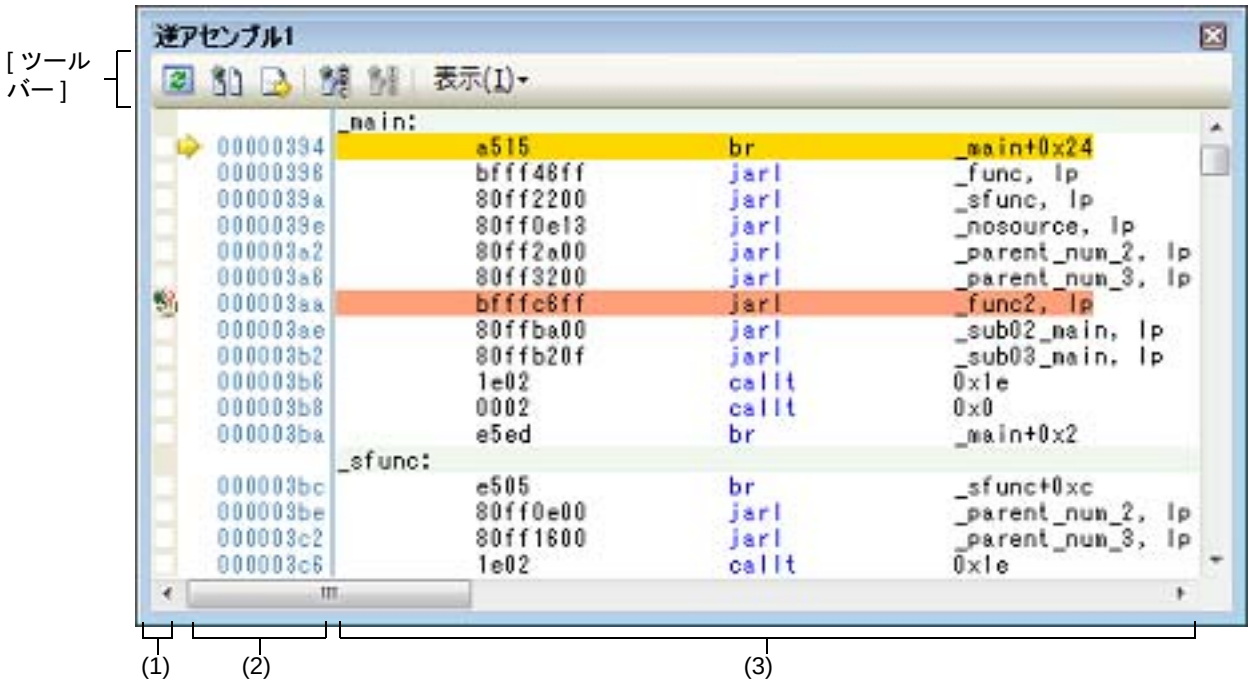
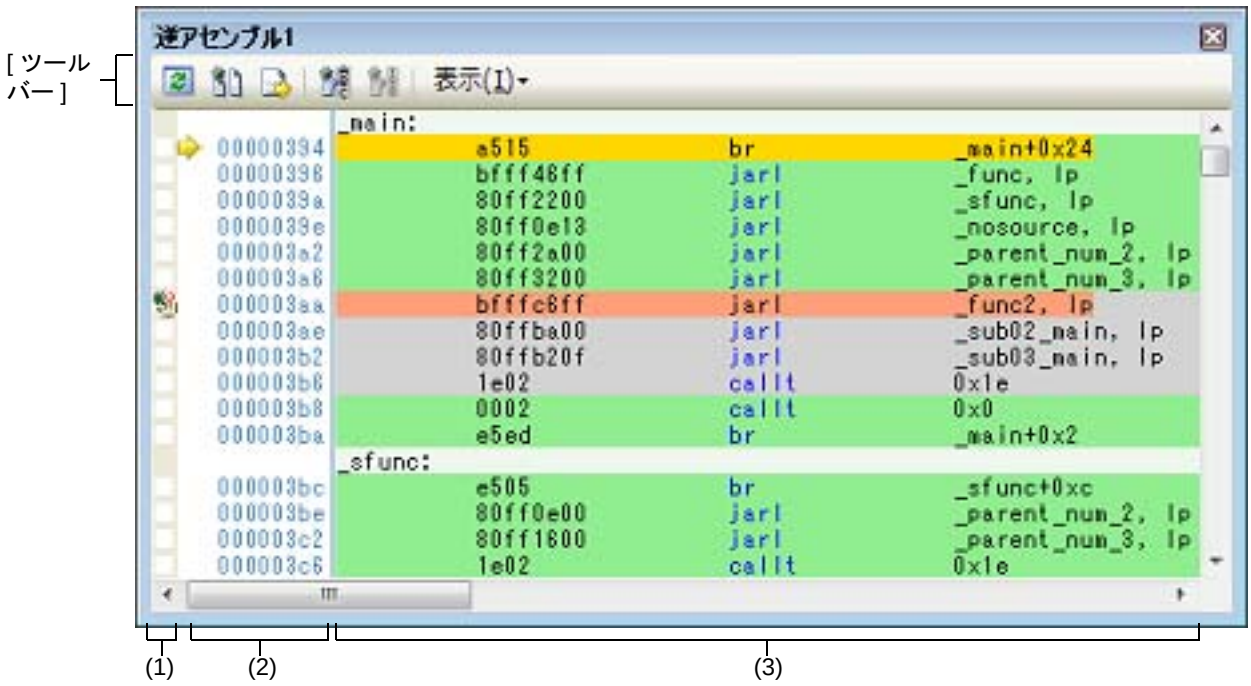


図 A.18 逆アセンブル パネル（コード・カバレッジ測定結果を表示した場合）【シミュレータ】



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［ツールバー］
- ［［ファイル］メニュー（逆アセンブル パネル専用部分）］
- ［［編集］メニュー（逆アセンブル パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］メニュー→［逆アセンブル］→［逆アセンブル *1* ～ *4*］を選択

［各エリアの説明］

(1) イベント・エリア
イベントの設定が可能な行は，背景色を白色で表示します（背景色がグレー表示の行は，イベントの設定が不可能であることを示すします）。
また，現在設定しているイベントがある場合，そのイベント設定行に，イベント種別を示すイベント・マークを表示します。
このエリアは，次の機能を備えています。

(a) ブレークポイントの設定／削除
ブレークポイントを設定したい箇所をマウスでクリックすることにより，容易にブレークポイントを設定することができます。
ブレークポイントは，クリックした行位置に対応する先頭アドレスの命令に対して設定されます。
ブレークポイントを設定すると，設定した行にイベント・マークを表示します。また，設定したブレークポイントの詳細情報をイベント パネルに反映します。
なお，すでにいずれかのイベント・マークを表示している箇所でこの操作を行った場合は，そのイベントを削除し，ブレークポイントの設定は行いません。
ただし，イベントの設定は，背景色が白色で表示されている行に対してのみ行うことができます。
なお，ブレークポイントの設定方法についての詳細は，「2.9.3 任意の場所で停止する（ブレークポイント）」を参照してください。

(b) 各種イベントの状態変更
各種イベント・マークを右クリックすることにより，次のメニューが表示され，選択したイベントの状態の変更を行うことができます。

有効化	選択しているイベントを有効状態にします。 指定されている条件の成立で，対象となるイベントが発生します。 なお，複数のイベントが設定されていることを示すイベント・マーク（  ）を選択している場合は，設定されているすべてのイベントを有効状態にします。
無効化	選択しているイベントを無効状態にします。 指定されている条件が成立しても，対象となるイベントは発生しません。 なお，複数のイベントが設定されていることを示すイベント・マーク（  ）を選択している場合は，設定されているすべてのイベントを無効状態にします。
イベント削除	選択しているイベントを削除します。 なお，複数のイベントが設定されていることを示すイベント・マーク（  ）を選択している場合は，設定されているすべてのイベントを削除します。
詳細設定情報表示	選択しているイベントの詳細情報を表示するイベント パネルをオープンします。

(c) ポップアップ表示
イベント・マークにマウス・カーソルを重ねることにより，そのイベントのイベント名／詳細情報／イベントに付加されたコメントをポップアップ表示します。
なお，該当箇所に複数のイベントが設定されている場合，最大 3 つまで，各イベントの情報を列挙して表示します。

(2) アドレス・エリア
行ごとの逆アセンブル開始アドレスを表示します（16 進数表記固定）。
また，カレント PC 位置（PC レジスタ値）を示すカレント PC マーク（  ）を表示します。
アドレス幅は，プロジェクトで指定しているマイクロコントローラのメモリ空間のアドレス幅となります。
なお，混合表示モード時におけるソース・テキスト行に対しては，開始アドレスに対応するソース・ファイル中の行番号（*xxx* :）を表示します。
このエリアは，次の機能を備えています。

備考 実行停止時のデバッグ対象コアと現在のデバッグ対象コアが異なる場合など，カレント PC 位置が無効状態となった場合は，カレント PC マークは  から  へ変化します。

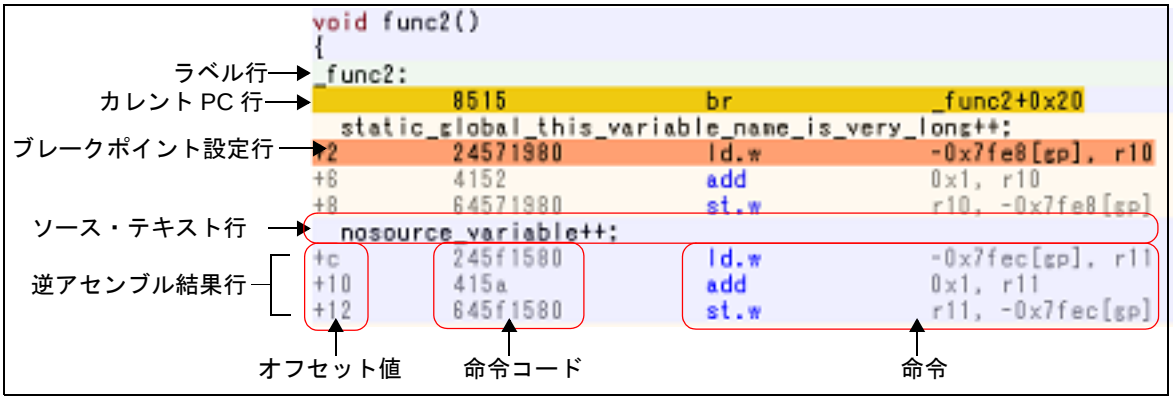
(a) ポップアップ表示
アドレス／ソース行番号にマウス・カーソルを重ねることにより，次の情報をポップアップ表示します。

アドレス	形式：＜*ロード・モジュール名*＞[注]＄＜*ラベル 名*＞＋＜*オフセット値*＞ 例 1： test1.abs$main + 0x10 例 2： subfunction + 0x20
ソース行番号	形式：＜*ロード・モジュール名*＞[注]＄＜*ファイル名*＞＃＜*行番号*＞ 例 1： test1.abs$main.c#40 例 2： main.c#100

注　ロード・モジュール名は，ロード・モジュール・ファイルが複数ダウンロードされている場合のみ表示します。

(3) 逆アセンブル・エリア
対象となるソース・テキスト行に続き，逆アセンブル結果行を次のように表示します。

図 A.19　逆アセンブル・エリアの表示内容（混合表示モードの場合）



ラベル行		アドレスにラベルが定義されている場合は，ラベル名を表示し，行全体を薄緑色で強調表示します。
カレント PC 行		カレント PC 位置（PC レジスタ値）のアドレスと対応する行を強調表示[注 1]します。
ブレークポイント設定行		ブレークポイントが設定されている行を強調表示[注 1]します。
ソース・テキスト行		コード・データに対応するソース・テキストを表示します[注 2]。
逆アセンブル結果行	オフセット値	アドレスにラベルが定義されていない場合は，一番近いラベルからのオフセット値を表示します[注 3]。
	命令コード	逆アセンブルの対象となったコードを 16 進数で表示します。
	命令	逆アセンブル結果として命令を表示します。ニモニックは青色で強調表示します。

注 1.　強調色は，オプション ダイアログにおける［全般 - フォントと色］カテゴリの設定に依存します。

注 2.　ツールバーの  ボタン（トグル）のクリック，またはコンテキスト・メニューの［混合表示モード］のチェックを外すことにより，ソース・テキストを非表示にすることができます（デフォルトでチェックされています）。

注 3.　オフセット値はデフォルトでは表示されません。表示する場合は，ツールバーの  ボタンのクリック，またはコンテキスト・メニューの［ラベルのオフセットを表示］を選択してください。

このエリアは，次の機能を備えています。

(a) ライン・アセンブル
表示している命令／コードは，編集（ライン・アセンブル）することができます。
操作方法についての詳細は，「2.6.4 ライン・アセンブルを行う」を参照してください。

(b) 命令レベルでのプログラム実行
このパネルにフォーカスがある状態でプログラムをステップ実行することにより，命令レベル単位で実行を制御することができます。

操作方法についての詳細は，「2.8.3 プログラムをステップ実行する」を参照してください。

(c) 各種イベントの設定
コンテキスト・メニューの［ブレークの設定］／［トレース設定］／［タイマ設定］などを選択することにより，現在キャレットのあるアドレスに各種イベントを設定することができます。
イベントを設定すると，対応するイベント・マークをイベント・エリアに表示します。また，設定したイベントの詳細情報をイベント パネルに反映します。
ただし，イベントの設定は，イベント・エリアにおいて，背景色が白色で表示されている行に対してのみ行うことができます。
なお，各種イベントの設定方法についての詳細は，次を参照してください。

- 「2.9.4 任意の場所で停止する（ブレーク・イベント）」
- 「2.9.5 変数 /I/O レジスタへのアクセスで停止する」
- 「2.12.3 任意区間の実行履歴を収集する」
- 「2.12.4 条件を満たしたときのみの実行履歴を収集する【シミュレータ】」
- 「2.13.2 任意区間の実行時間を計測する【シミュレータ】」
- 「2.15.1 printf を挿入する」

備考 ブレークポイントの設定／削除は，イベント・エリアにおいても簡単に行うことができます（「(a) ブレークポイントの設定／削除」参照）。

(d) ウォッチ式の登録
表示している C 言語変数 /CPU レジスタ /I/O レジスタ／アセンブラ・シンボルをウォッチ式としてウォッチ パネルに登録することができます。
操作方法についての詳細は，「2.10.6.1 ウォッチ式を登録する」を参照してください。

(e) シンボル定義箇所への移動
シンボルを参照している命令にキャレットを移動した状態で，ツールバーの  ボタンをクリック，またはコンテキスト・メニューの［シンボルへ移動］を選択することにより，キャレット位置のシンボルが定義されているアドレスにキャレット位置を移動します。
また，この操作に続き，ツールバーの  ボタンをクリック，またはコンテキスト・メニューの［*アドレス*に戻る］を選択すると，キャレット移動前のシンボルを参照している命令にキャレット位置を戻します（*アドレス*はシンボルを参照している命令のアドレス値を表示）。

(f) ソース行／メモリ値へのジャンプ
コンテキスト・メニューの［ソースへジャンプ］を選択することにより，現在のキャレット位置のアドレスに対応するソース行にキャレットを移動した状態でエディタ パネルがオープンします（すでにオープンしている場合は，エディタ パネルにジャンプ）。
また，同様に［メモリへジャンプ］を選択することにより，現在のキャレット位置のアドレスに対応するメモリ値にキャレットを移動した状態でメモリ パネル（メモリ 1）がオープンします（すでにオープンしている場合は，メモリ パネル（メモリ 1）にジャンプ）。

(g) コード・カバレッジ測定結果の表示【シミュレータ】
カバレッジ機能を有効としている場合，プログラムの実行により取得したコード・カバレッジ測定結果を基に，カバレッジ測定対象領域に相当する行を強調表示します。
カバレッジ測定についての詳細は，「2.14 カバレッジの測定【シミュレータ】」を参照してください。

(h) 逆アセンブル・データの保存
［ファイル］メニュー→［名前を付けて逆アセンブル・データを保存 ...］を選択することにより，データ保存 ダイアログをオープンし，このパネルの内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
逆アセンブル・データの保存方法についての詳細は，「2.6.2.5 逆アセンブル結果の表示内容を保存する」を参照してください。

［ツールバー］

	デバッグ・ツールから最新の情報を取得し，表示を更新します。
	このパネルの表示モードとして，混合表示モード（デフォルト）と逆アセンブル表示モードを切り替えます（「2.6.2.1 表示モードを変更する」参照）。
	キャレット位置をカレント PC 値に追従するように指定します。
	選択しているシンボルの定義位置へキャレットを移動します。

	 ボタンで移動する直前の位置（*アドレス*）へ移動します。
表示	逆アセンブル・エリアの表示形式を変更する次のボタンを表示します。
	ラベルのオフセット値を表示します。アドレスにラベルが定義されていない場合，一番近いラベルからのオフセット値を表示します。
	アドレス値を“シンボル＋オフセット値”で表示します（デフォルト）。 ただし，アドレス値にシンボルが定義されている場合は，シンボルのみを表示します。
	レジスタ名を機能名称で表示します（デフォルト）。
	レジスタ名を絶対名称で表示します。
	スクロール範囲を設定するためのスクロール範囲設定 ダイアログがオープンします。

[[ファイル]メニュー（逆アセンブル パネル専用部分）]

逆アセンブル パネル専用の［ファイル］メニューは次のとおりです（その他の項目は共通）。
ただし，プログラム実行中はすべて無効となります。

逆アセンブル・データを保存	逆アセンブルの内容を前回保存したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存します（「(h) 逆アセンブル・データの保存」参照）。 なお，起動後に初めてこの項目を選択した場合は，［名前を付けて逆アセンブル・データを保存 ...］の選択と同等の動作となります。
名前を付けて逆アセンブル・データを保存 ...	逆アセンブルの内容を指定したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存するために，データ保存 ダイアログをオープンします（「(h) 逆アセンブル・データの保存」参照）。
印刷 ...	このパネルの内容を印刷するために，印刷アドレス範囲設定 ダイアログをオープンします。

[[編集]メニュー（逆アセンブル パネル専用部分）]

逆アセンブル パネル専用の［編集］メニューは次のとおりです（その他の項目はすべて無効）。

コピー	行を選択している場合，選択している行の内容を文字列としてクリップ・ボードにコピーします。 編集モードの場合，選択している文字列をクリップ・ボードにコピーします。
名前の変更	キャレット位置の命令／命令コードを編集するために，編集モードに移行します（「2.6.4 ライン・アセンブルを行う」参照）。 ただし，プログラム実行中は無効となります。
検索 ...	検索・置換 ダイアログを［一括検索］タブが選択状態でオープンします。
置換 ...	検索・置換 ダイアログを［一括置換］タブが選択状態でオープンします。
移動 ...	指定したアドレスへキャレットを移動するため，指定位置へ移動 ダイアログをオープンします。

[コンテキスト・メニュー]

(1) 逆アセンブル・エリア／アドレス・エリア
(2) イベント・エリア【Full-spec emulator】【E1】【E20】

(1) 逆アセンブル・エリア／アドレス・エリア

ウォッチ 1 に登録	選択している文字列，またはキャレット位置の単語をウォッチ式としてウォッチ パネル（ウォッチ 1）に登録します（単語の判断は現在のビルド・ツールに依存）。 ウォッチ式として登録する際は変数名として登録されるため，スコープにより表示されるシンボル名は変化します。

アクション・イベントの登録 ...	キャレット位置のアドレスにアクション・イベントを設定するため，アクション・イベント ダイアログをオープンします。
ここまで実行	カレント PC 値で示されるアドレスから，キャレット位置の行に対応するアドレスまでプログラムを実行します。 ただし，プログラム実行中，またはビルド（ラピット・ビルドを除く）実行中は無効となります。
PC をここに設定	カレント PC 値を現在キャレットのある行のアドレスに変更します。 ただし，プログラム実行中，またはビルド（ラピット・ビルドを除く）実行中は無効となります。
移動 ...	指定したアドレスへキャレットを移動するため，指定位置へ移動 ダイアログをオープンします。
シンボルへ移動	選択しているシンボルの定義位置へキャレットを移動します。
*アドレス*へ戻る	[シンボルへ移動] で移動する直前の位置（*アドレス*）へ移動します。 ただし，*アドレス*にシンボル名が表示されていない場合は無効となります。
ブレークの設定	ブレーク関連のイベントを設定するために，次のカスケード・メニューを表示します。
ハード・ブレークの設定	キャレット位置のアドレスにブレークポイント（ハードウエア・ブレーク・イベント）を設定します（「2.9.3 任意の場所で停止する（ブレークポイント）」参照）。
ソフト・ブレークの設定【Full-spec emulator】【E1】【E20】	キャレット位置のアドレスにブレークポイント（ソフトウエア・ブレーク・イベント）を設定します（「2.9.3 任意の場所で停止する（ブレークポイント）」参照）。
読み込みブレークを設定	キャレット位置，または選択している変数（グローバル変数／関数内スタティック変数／ファイル内スタティック変数）/I/O レジスタに，リード・アクセスのブレーク・イベントを設定します（「2.9.5.1 ブレーク・イベント（アクセス系）を設定する」参照）。
書き込みブレークを設定	キャレット位置，または選択している変数（グローバル変数／関数内スタティック変数／ファイル内スタティック変数）/I/O レジスタに，ライト・アクセスのブレーク・イベントを設定します（「2.9.5.1 ブレーク・イベント（アクセス系）を設定する」参照）。
読み書きブレークを設定	キャレット位置，または選択している変数（グローバル変数／関数内スタティック変数／ファイル内スタティック変数）/I/O レジスタに，リード／ライト・アクセスのブレーク・イベントを設定します（「2.9.5.1 ブレーク・イベント（アクセス系）を設定する」参照）。
ブレーク動作の設定	プロパティ パネルをオープンし，ブレーク機能の設定を行います。

トレース設定	トレース関連のイベントを設定するために，次のカスケード・メニューを表示します。
トレース開始の設定	キャレット位置のアドレスの命令が実行された際に，プログラムの実行履歴を示すトレース・データの収集を開始するトレース開始イベントを設定します（「2.12.3 任意区間の実行履歴を収集する」参照）注1。
トレース終了の設定	キャレット位置のアドレスの命令が実行された際に，プログラムの実行履歴を示すトレース・データの収集を終了するトレース終了イベントを設定します（「2.12.3 任意区間の実行履歴を収集する」参照）注1。
値をトレースに記録（読み込み時）	キャレット位置，または選択している変数（グローバル変数／関数内スタティック変数／ファイル内スタティック変数）/I/O レジスタにリード・アクセスした際に，その値をトレース・メモリに記録するポイント・トレース・イベントを設定します（「2.12.4 条件を満たしたときのみの実行履歴を収集する【シミュレータ】」参照）。
値をトレースに記録（書き込み時）	キャレット位置，または選択している変数（グローバル変数／関数内スタティック変数／ファイル内スタティック変数）/I/O レジスタにライト・アクセスした際に，その値をトレース・メモリに記録するポイント・トレース・イベントを設定します（「2.12.4 条件を満たしたときのみの実行履歴を収集する【シミュレータ】」参照）。
値をトレースに記録（読み書き時）	キャレット位置，または選択している変数（グローバル変数／関数内スタティック変数／ファイル内スタティック変数）/I/O レジスタにリード／ライト・アクセスした際に，その値をトレース・メモリに記録するポイント・トレース・イベントを設定します（「2.12.4 条件を満たしたときのみの実行履歴を収集する【シミュレータ】」参照）。
トレース結果の表示	トレース パネルをオープンし，取得したトレース・データを表示します。
トレース動作の設定	プロパティ パネルをオープンし，トレース機能の設定を行います。 ただし，トレーサ動作中は無効となります。
タイマ設定	タイマ関連のイベントを設定するために，次のカスケード・メニューを表示します（「2.13.2 任意区間の実行時間を計測する【シミュレータ】」参照）。
実行時にタイマ開始	キャレット位置のアドレスの命令が実行された際に，プログラムの実行時間の計測を開始するタイマ開始イベントを設定します注2。
タイマ *n* に設定	タイマ開始イベントを設定するチャネル（*n*：1 ～ 8）を指定します。
実行時にタイマ終了	キャレット位置のアドレスの命令が実行された際に，プログラムの実行時間の計測を終了するタイマ終了イベントを設定します注2。
タイマ *n* に設定	タイマ開始イベントを設定するチャネル（*n*：1 ～ 8）を指定します。
タイマ結果の表示	イベント パネルをオープンし，タイマ関連のイベントのみ表示します。
カバレッジ情報をクリア【シミュレータ】	デバッグ・ツールが保持しているコード・カバレッジ測定結果をすべてクリアします。
命令の編集	キャレット位置の行の命令を編集するために，編集モードに移行します（「2.6.4 ライン・アセンブルを行う」参照）。 ただし，プログラム実行中は無効となります。
コードの編集	キャレット位置の行の命令コードを編集するために，編集モードに移行します（「2.6.4 ライン・アセンブルを行う」参照）。 ただし，プログラム実行中は無効となります。

表示	逆アセンブル・エリアの表示内容を設定するために，次のカスケード・メニューを表示します。
ラベルのオフセットを表示	ラベルのオフセット値を表示します。アドレスにラベルが定義されていない場合，一番近いラベルからのオフセット値を表示します。
アドレス値をシンボルで表示	アドレス値を“シンボル＋オフセット値”で表示します（デフォルト）。 ただし，アドレス値にシンボルが定義されている場合は，シンボルのみを表示します。
レジスタを機能名称で表示	レジスタ名を機能名称で表示します（デフォルト）。
レジスタを絶対名称で表示	レジスタ名を絶対名称で表示します。
スクロール範囲を設定 ...	スクロール範囲を設定するためのスクロール範囲設定 ダイアログがオープンします。
混合表示	このパネルの表示モードとして，混合表示モード（デフォルト）と逆アセンブル表示モードを切り替えます（「2.6.2.1 表示モードを変更する」参照）。
ソースへジャンプ	キャレット位置のアドレスに対応するソース行にキャレットを移動した状態で，エディタ パネルがオープンします。
メモリへジャンプ	キャレット位置のアドレスに対応するメモリ値にキャレットを移動した状態で，メモリ パネル（メモリ 1）がオープンします。

注 1. 【シミュレータ】
プロパティ パネル上の［トレース］カテゴリ内［トレース機能を使用する］プロパティの設定を自動的に［はい］にします。

注 2. 【シミュレータ】
プロパティ パネル上の［タイマ］【シミュレータ】カテゴリ内［タイマ機能を使用する］プロパティの設定を自動的に［はい］にします。

(2) イベント・エリア【Full-spec emulator】【E1】【E20】

ハードウエア・ブレークを優先する	マウスのワンクリック操作で設定できるブレークの種類をハードウエア・ブレークポイントとします（プロパティ パネル上の［ブレーク］【Full-spec emulator】【E1】【E20】カテゴリ内［優先的に使用するブレークポイントの種類］プロパティの設定に反映されます）。
ソフトウエア・ブレークを優先する	マウスのワンクリック操作で設定できるブレークの種類をソフトウエア・ブレークポイントとします（プロパティ パネル上の［ブレーク］【Full-spec emulator】【E1】【E20】カテゴリ内［優先的に使用するブレークポイントの種類］プロパティの設定に反映されます）。

CPU レジスタ パネル

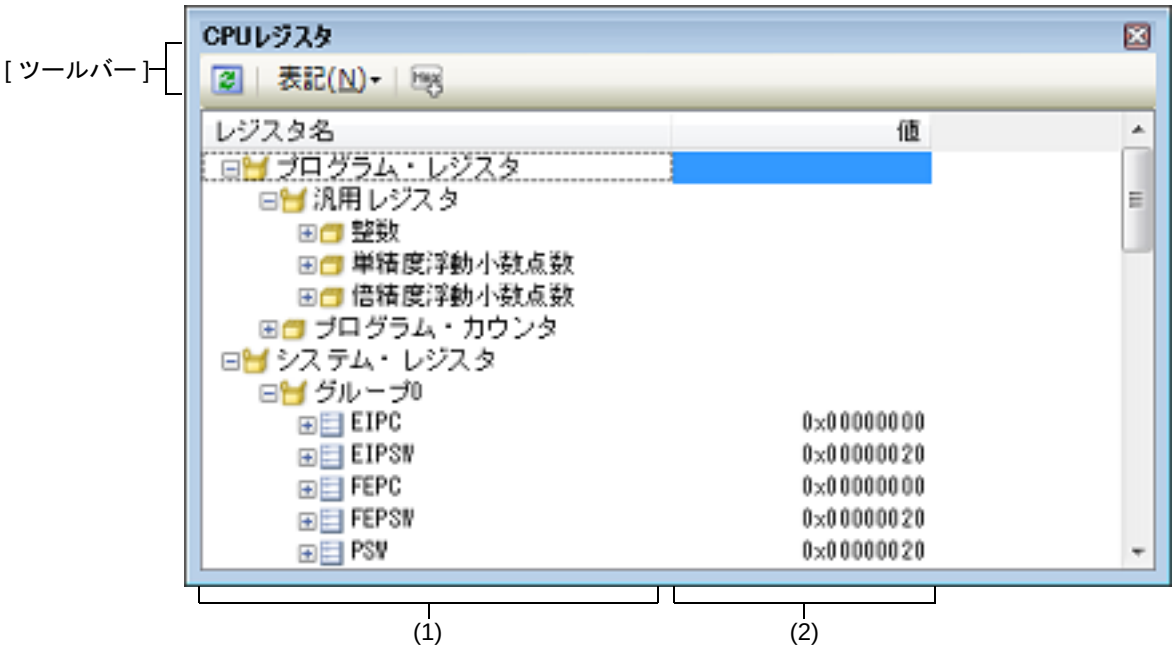
CPU レジスタ（プログラム・レジスタ／システム・レジスタ）の内容の表示，および値の変更を行います（「2.10.2 CPU レジスタを表示／変更する」参照）。
なお，このパネルは，デバッグ・ツールと接続時のみオープンすることができます。

注意 選択しているマイクロコントローラがマルチコア対応版の場合，コア（PE）の選択を切り替えることにより，選択した PE に対応した内容の表示／値の変更を行います（「2.7 コア（PE）の選択」参照）。

備考 1. ツールバーの 100% ，または［Ctrl］キーを押下しながらマウス・ホイールを前後方に動かすことにより，本パネルの表示を拡大／縮小することができます。

備考 2. パネル上の各エリアの区切り線をダブルクリックすることにより，該当エリアの内容を省略することなく表示可能な最小幅に変更することができます。

図 A.20 CPU レジスタ パネル



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［ツールバー］
- ［［ファイル］メニュー（CPU レジスタ パネル専用部分）］
- ［［編集］メニュー（CPU レジスタ パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］メニュー→［CPU レジスタ］を選択

［各エリアの説明］

(1) ［レジスタ名］エリア
レジスタの種別をカテゴリ（フォルダ）として分類し，各レジスタ名を一覧表示します。
表示される各アイコンの意味は次のとおりです。
なお，カテゴリ名／レジスタ名を編集／削除することはできません。

	このカテゴリに属するレジスタ名を表示している状態を示します。アイコンをダブルクリック，または “-” マークをクリックすると，カテゴリを閉じてレジスタ名を非表示にします。
	このカテゴリに属するレジスタ名が非表示の状態を示します。アイコンをダブルクリック，または “+” マークをクリックすると，カテゴリを開いてレジスタ名を表示します。
	レジスタ名を示します。アイコンをダブルクリック，または “+” ／ “-” マークをクリックすると，下階層のレジスタ名（レジスタの部分を表す名称）を表示／非表示します。
	レジスタ名（レジスタの部分を表す名称）を示します。

このエリアは，次の機能を備えています。

(a) ウォッチ式の登録
CPU レジスタ／カテゴリをウォッチ式としてウォッチ パネルに登録することができます。
操作方法についての詳細は，「2.10.6.1 ウォッチ式を登録する」を参照してください。

備考 1. カテゴリを対象としてウォッチ式の登録を行った場合，そのカテゴリに属するすべての CPU レジスタがウォッチ式として登録されます。

備考 2. 登録したウォッチ式には，自動的にスコープ指定が付与されます。

(2) ［値］エリア
各 CPU レジスタの値を表示／変更します。
表示進数は，ツールバーのボタン，またはコンテキスト・メニューより選択することができます。また，常に 16 進数表示を併記する表示形式を選択することもできます。
CPU レジスタの値として表示されるマークや色の意味は次のとおりです（文字色／背景色はオプション ダイアログにおける［全般 - フォントと色］カテゴリの設定に依存）。

表示例（デフォルト）			説明
0x0	文字色	青色	ユーザにより，値が変更されている CPU レジスタの値（［Enter］キーによりターゲット・メモリに書き込まれます）
	背景色	標準色	
0x0	文字色	茶色	プログラムの実行により，値が変化した CPU レジスタの値 プログラムを再実行させることにより，強調色をリセットします。
	背景色	クリーム	

このエリアは，次の機能を備えています。

(a) CPU レジスタ値の変更
CPU レジスタ値の変更は，対象 CPU レジスタ値を選択したのち，再度クリックし，キーボードからの直接入力により行います（［Esc］キーの押下で編集モードをキャンセルします）。
CPU レジスタ値を編集したのち，［Enter］キーの押下，または編集領域以外へのフォーカスの移動により，デバッグ・ツールのレジスタに書き込まれます。

(b) CPU レジスタ値の保存
［ファイル］メニュー→［名前を付けて CPU レジスタ・データを保存 ...］を選択することにより，名前を付けて保存 ダイアログをオープンし，このパネルのすべての内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
CPU レジスタ値の保存方法についての詳細は，「2.10.2.4 CPU レジスタの表示内容を保存する」を参照してください。

［ツールバー］

	デバッグ・ツールから最新の情報を取得し，表示を更新します。 ただし，プログラム実行中は無効となります。
表記	値の表示形式を変更する次のボタンを表示します。
	選択している項目（下位項目を含む）の値を規定値で表示します（デフォルト）。
	選択している項目（下位項目を含む）の値を 16 進数で表示します。
	選択している項目（下位項目を含む）の値を符号付き 10 進数で表示します。
	選択している項目（下位項目を含む）の値を符号なし 10 進数で表示します。
	選択している項目（下位項目を含む）の値を 8 進数で表示します。
	選択している項目（下位項目を含む）の値を 2 進数で表示します。
	選択している項目（下位項目を含む）の文字列を ASCII コードで表示します。対象が 2 バイト以上ある場合は，1 バイトずつの文字を並べて表示します。
	選択している項目を Float で表示します。 ただし，4 バイト・データ以外の場合は，規定値で表示します。
	選択している項目を Double で表示します。 ただし，8 バイト・データ以外の場合は，規定値で表示します。
	値表示の末尾に，その値の 16 進数表記を “（）” で囲んで併記します。

［［ファイル］メニュー（CPU レジスタ パネル専用部分）］

CPU レジスタ パネル専用の［ファイル］メニューは次のとおりです（その他の項目は共通）。
ただし，プログラム実行中はすべて無効となります。

CPU レジスタ・データを保存	このパネルの内容を前回保存したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存します（「(b) CPU レジスタ値の保存」参照）。 なお，起動後に初めてこの項目を選択した場合は，［名前を付けて CPU レジスタ・データを保存 ...］の選択と同等の動作となります。
名前を付けて CPU レジスタ・データを保存 ...	このパネルの内容を指定したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存するために，名前を付けて保存 ダイアログをオープンします（「(b) CPU レジスタ値の保存」参照）。

［［編集］メニュー（CPU レジスタ パネル専用部分）］

CPU レジスタ パネル専用の［編集］メニューは次のとおりです（その他の項目はすべて無効）。

切り取り	選択範囲の文字列を切り取り，クリップ・ボードにコピーします。 ただし，文字列を編集中の場合のみ有効となります。
コピー	編集中の場合，選択している文字列をクリップ・ボードにコピーします。 行を選択している場合，レジスタ／カテゴリをクリップ・ボードにコピーします。 なお，コピーした項目は，ウォッチ パネルに貼り付け可能です。
貼り付け	クリップ・ボードにコピーされている文字列を キャレット位置に貼り付けます。 ただし，文字列を編集中の場合のみ有効となります。
すべて選択	すべての項目を選択状態にします。
検索 ...	検索・置換 ダイアログを ［一括検索］タブが選択状態でオープンします。
置換 ...	検索・置換 ダイアログを ［一括置換］タブが選択状態でオープンします。

［コンテキスト・メニュー］

ウォッチ 1 に登録	選択しているレジスタ名／カテゴリをウォッチ パネル（ウォッチ 1）に登録します。
コピー	編集中の場合，選択している文字列をクリップ・ボードにコピーします。 行選択している場合，レジスタ項目／カテゴリをクリップ・ボードにコピーします。 なお，コピーした項目は，ウォッチ パネルに貼り付け可能です。
表記	表示形式を指定するため，次のカスケード・メニューを表示します。
自動	選択している項目（下位項目を含む）の値を規定値で表示します（デフォルト）。
16 進数	選択している項目（下位項目を含む）の値を 16 進数で表示します。
符号付き 10 進数	選択している項目（下位項目を含む）の値を符号付き 10 進数で表示します。
符号無し 10 進数	選択している項目（下位項目を含む）の値を符号なし 10 進数で表示します。
8 進数	選択している項目（下位項目を含む）の値を 8 進数で表示します。
2 進数	選択している項目（下位項目を含む）の値を 2 進数で表示します。
ASCII	選択している項目（下位項目を含む）の文字列を ASCII コードで表示します。 対象が 2 バイト以上ある場合は，1 バイトずつの文字を並べて表示します。
Float	選択している項目を Float で表示します。 ただし，4 バイト・データ以外の場合は，規定値で表示します。
Double	選択している項目を Double で表示します。 ただし，8 バイト・データ以外の場合は，規定値で表示します。
16 進数値を併記	値表示の末尾に，その値の 16 進数表記を “（）” で囲んで併記します。

IOR パネル

I/O レジスタの内容の表示，および値の変更を行います（「2.10.3 I/O レジスタを表示／変更する」参照）。
なお，このパネルは，デバッグ・ツールと接続時のみオープンすることができます。

注意 1. 読み込み動作によってマイクロコントローラが動作してしまう I/O レジスタは，読み込み保護対象となるため，値の読み込みは行いません（［値］に “?” を表示）。読み込み保護対象の I/O レジスタ の内容を取得したい場合は，コンテキスト・メニューの［値を強制読み込み］を選択することで，1 度だけ値の読み込みが可能です。
なお，いったん［値を強制読み込み］を選択すると “？ ” 表示ではなくなるため，読み込み保護対象の I/O レジスタであることがわからなくなるため注意が必要です。

注意 2. 選択しているマイクロコントローラがマルチコア対応版の場合，コア（PE）の選択を切り替えることにより，選択した PE に対応した内容の表示／値の変更を行います（「2.7 コア（PE）の選択」参照）。

備考 1. ツールバーの  ，または［Ctrl］キーを押下しながらマウス・ホイールを前後方に動かすことにより，本パネルの表示を拡大／縮小することができます。

備考 2. パネル上の各エリアの区切り線をダブルクリックすることにより，該当エリアの内容を省略することなく表示可能な最小幅に変更することができます。

図 A.21 IOR パネル

ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［ツールバー］
- ［［ファイル］メニュー（IOR パネル専用部分）］
- ［［編集］メニュー（IOR パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］メニュー→［IOR］を選択

［各エリアの説明］

(1) 検索エリア
I/O レジスタ名の検索を行います。

	検索対象の文字列を指定します（大文字／小文字不問）。 キーボードより文字列を直接入力するか（最大指定文字数：512 文字），ドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
	テキスト・ボックスで指定している文字列を含む I/O レジス名を上方向に検索し，検索結果を選択状態にします。
	テキスト・ボックスで指定している文字列を含む I/O レジス名を下方向に検索し，検索結果を選択状態にします。

備考 1. カテゴリ（フォルダ）により分類されて非表示の状態の I/O レジスタ名も検索します（展開して選択状態となります）。

備考 2. 検索対象の文字列入力後，［Enter］キーを押下することにより，  ボタンのクリックと同等の動作を行い，［Shift］＋［Enter］キーを押下することにより，  ボタンのクリックと同等の動作を行います。

(2) ［IOR］エリア

I/O レジスタの種別をカテゴリ（フォルダ）として分類し，各 I/O レジスタ名を一覧表示します。
表示される各アイコンの意味は次のとおりです。

	このカテゴリに属する I/O レジスタ名を表示している状態を示します。アイコンをダブルクリック，または “-” マークをクリックすると，カテゴリを閉じ I/O レジスタ名を非表示にします。 なお，カテゴリはデフォルトでは存在しません。必要な場合は，カテゴリを新規作成したのち，ツリーの編集を行ってください。
	I/O レジスタ名が非表示の状態を示します。アイコンをダブルクリック，または “+” マークをクリックすると，カテゴリを開き I/O レジスタ名を表示します。 なお，カテゴリはデフォルトでは存在しません。必要な場合は，カテゴリを新規作成したのち，ツリーの編集を行ってください。
	I/O レジスタ名を示します。

備考 このエリアのヘッダ部をクリックすることにより，カテゴリ名を文字コード順でソートします（カテゴリ内 I/O レジスタ名も同様にソートします）。

このエリアは，次の機能を備えています。

(a) ツリーの編集

各 I/O レジスタを任意のカテゴリ（フォルダ）で分類し，ツリー形式を編集することができます。
カテゴリを新規に作成する場合は，作成したい I/O レジスタ名にキャレットを移動したのち，ツールバーの  ボタンのクリック，またはコンテキスト・メニューの［カテゴリを作成］を選択し，任意にカテゴリ名称を入力することにより行います（最大指定文字数：1024 文字）。
なお，カテゴリを削除する場合は，削除したいカテゴリを選択したのち，ツールバーの  ボタンのクリック，またはコンテキスト・メニューの［削除］を選択します。ただし，削除できるカテゴリは，空のカテゴリのみです。
また，カテゴリ名を編集する場合は，編集したいカテゴリ名を選択したのち，次のいずれかの操作により行います。

- 再度クリック後，キーボードよりカテゴリ名を直接編集
- ［編集］メニュー→［名前の変更］を選択後，キーボードよりカテゴリ名を直接編集
- ［F2］キーを押下後，キーボードよりカテゴリ名を直接編集

カテゴリを作成したのち，I/O レジスタ名をカテゴリ内に直接ドラッグ・アンド・ドロップすることにより，各 I/O レジスタをカテゴリで分類したツリー形式で表示します。
同様に，カテゴリと I/O レジスタ名の表示の順番（上下位置）も，ドラッグ・アンド・ドロップ操作により自由に変更することができます。

注意 1. カテゴリ内にカテゴリを作成することはできません。

注意 2. I/O レジスタの追加／削除はできません。

(b) ウォッチ式の登録

I/O レジスタ／カテゴリをウォッチ式としてウォッチ パネルに登録することができます。
操作方法についての詳細は，「2.10.6.1 ウォッチ式を登録する」を参照してください。

備考 1. カテゴリを対象としてウォッチ式の登録を行った場合，そのカテゴリに属するすべての I/O レジスタがウォッチ式として登録されます。

備考 2. 登録したウォッチ式には，自動的にスコープ指定が付与されます。

(3) ［値］エリア

I/O レジスタの値を表示／変更します。
表示進数は，ツールバーのボタン，またはコンテキスト・メニューより選択することができます。また，常に 16 進数表示を併記する表示形式を選択することもできます。
I/O レジスタの値として表示されるマークや色の意味は次のとおりです（文字色／背景色はオプション ダイアログにおける［全般 - フォントと色］カテゴリの設定に依存）。

表示例（デフォルト）			説明
0x0	文字色	青色	ユーザにより，値が変更されている I/O レジスタの値（［Enter］キーによりターゲット・メモリに書き込まれます）
	背景色	標準色	
0x0	文字色	茶色	プログラムの実行により，値が変化した I/O レジスタの値 ツールバーの  ボタン，またはコンテキスト・メニューの［表示色をリセット］を選択することにより，強調表示をリセットします。
	背景色	クリーム	
?	文字色	グレー	読み込み保護対象の I/O レジスタ注の値
	背景色	標準色	

注 読み込み動作によってマイクロコントローラが動作してしまう I/O レジスタを示します。
読み込み保護対象の I/O レジスタの内容を取得する場合は，コンテキスト・メニューの［値を強制読み込み］を選択することにより行ってください。

注意 1 バイト /2 バイト I/O レジスタと，1 バイト /2 バイト I/O レジスタに割り付けられている 1 ビット I/O レジスタでは，値を取得するタイミングが異なります。このため，同一の I/O レジスタの値を表示していても値が異なる場合があります。

備考 このエリアのヘッダ部をクリックすることにより，値を数値の昇順でソートします。

このエリアは，次の機能を備えています。

(a) I/O レジスタ値の変更
I/O レジスタの値の変更は，対象 I/O レジスタ値を選択したのち，再度クリックし，キーボードからの直接入力により行います（［Esc］キーの押下で編集モードをキャンセルします）。
I/O レジスタ値を編集したのち，［Enter］キーの押下，または編集領域以外へのフォーカスの移動により，デバッグ・ツールのターゲット・メモリに書き込まれます。
I/O レジスタ値の変更方法についての詳細は，「2.10.3.4 I/O レジスタの内容を変更する」を参照してください。

(b) I/O レジスタ値の保存
［ファイル］メニュー→［名前を付けて *IOR* データを保存 ...］を選択することにより，名前を付けて保存 ダイアログをオープンし，I/O レジスタのすべての内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
I/O レジスタ値の保存方法についての詳細は，「2.10.3.6 I/O レジスタの表示内容を保存する」を参照してください。

(4) ［型情報（バイト数）］エリア

各 I/O レジスタの型情報を次の形式で表示します。

- ＜*I/O レジスタの種類*＞ [＜*アクセス属性*＞ ＜*アクセス可能サイズ*＞] (＜*サイズ*＞)

アクセス属性	アクセス属性として，次のいずれかを表示します。	
	R	リードのみ可能
	W	ライトのみ可能
	R/W	リード／ライト可能
アクセス可能サイズ	アクセス可能なすべてのサイズを，ビット単位で小さい順に “,” で区切り列挙します（1 ～ 32 ビット）。	

サイズ	I/O レジスタのサイズを表示します。 バイト単位で表示可能な場合はバイト単位で，ビット単位でのみ表示可能な場合はビット単位で単位を付与して表示します。

例 1. 「IOR [R/W 1.8] (1 バイト)」の場合
リード／ライト可能，1 ビット・アクセス /8 ビット・アクセス可能，サイズが 1 バイトの I/O レジスタ

例 2. 「IOR [R/W 1] (1 ビット)」の場合
リード／ライト可能，1 ビット・アクセス可能，サイズが 1 ビットの I/O レジスタ

備考 このエリアのヘッダ部をクリックすることにより，型情報を文字コード順でソートします。

(5) ［アドレス］エリア
各 I/O レジスタがマッピングされているアドレスを表示します（16 進数表記固定）。
ただし，ビット・レジスタの場合は，次の例のようにビット・オフセット値を付与して表示します。

例 1. 「0xFF40」の場合
アドレス “0xFF40” に割り当てられている

例 2. 「0xFF40.4」の場合
アドレス “0xFF40” のビット 4 に割り当てられている（ビット・レジスタ）

備考 このエリアのヘッダ部をクリックすることにより，アドレスを数値の昇順でソートします。

［ツールバー］

	デバッグ・ツールから最新の情報を取得し，表示を更新します。 読み込み保護対象の I/O レジスタの再読み込みは行いません。 ただし，プログラム実行中は無効となります。
	選択している I/O レジスタに対して，プログラム実行により値が変化したことを示す強調表示をリセットします。 ただし，プログラム実行中は無効となります。
	新規カテゴリ（フォルダ）を追加します。テキスト・ボックスに直接カテゴリ名を入力します。 なお，新規に作成できるカテゴリの数に制限はありませが，カテゴリ内にカテゴリを作成することはできません。 ただし，プログラム実行中は無効となります。
	選択している範囲の文字列を削除します。 空のカテゴリが選択状態の場合は，そのカテゴリを削除します（I/O レジスタの削除不可）。
表記	値の表示形式を変更する次のボタンを表示します。
	選択している項目の値を 16 進数で表示します（デフォルト）。
	選択している項目の値を符号付き 10 進数で表示します。
	選択している項目の値を符号なし 10 進数で表示します。
	選択している項目の値を 8 進数で表示します。
	選択している項目の値を 2 進数で表示します。
	選択している項目の値を ASCII コードで表示します。
	選択している項目の値表示の末尾に，その値の 16 進数表記を “（）” で囲んで併記します。

［[ファイル］メニュー（IOR パネル専用部分）］

IOR パネル専用の［ファイル］メニューは次のとおりです（その他の項目は共通）。
ただし，プログラム実行中はすべて無効となります。

IOR データを保存	このパネルの内容を前回保存したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存します（「(b) I/O レジスタ値の保存」参照）。 なお，起動後に初めてこの項目を選択した場合は，［名前を付けて IOR データを保存 ...］の選択と同等の動作となります。
名前を付けて IOR データを保存 ...	このパネルの内容を指定したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存するために，名前を付けて保存 ダイアログをオープンします（「(b) I/O レジスタ値の保存」参照）。

［［編集］メニュー（IOR パネル専用部分）］

IOR パネル専用の［編集］メニューは次のとおりです（その他の項目はすべて無効）。

切り取り	選択している範囲の文字列を切り取ってクリップ・ボードに移動します（I/O レジスタ／カテゴリの切り取り不可）。
コピー	選択している範囲の文字列をクリップ・ボードにコピーします。 I/O レジスタ／カテゴリが選択状態の場合は，その項目をコピーします。 なお，コピーした項目は，ウォッチ パネルに貼り付け可能です。
貼り付け	テキストが編集状態の場合，クリップ・ボードの内容をキャレット位置に挿入します（I/O レジスタ／カテゴリの貼り付け不可）。
削除	選択している範囲の文字列を削除します。 空のカテゴリが選択状態の場合は，その項目を削除します（I/O レジスタの削除不可）。
すべて選択	テキストが編集状態の場合，すべての文字列を選択します。 テキストが編集状態以外の場合，すべての I/O レジスタ／カテゴリを選択状態にします。
名前の変更	選択しているカテゴリの名称を編集します。
検索 ...	検索エリアのテキスト・ボックスにフォーカスを移動します。
移動 ...	指定した I/O レジスタへキャレットを移動するため，指定位置へ移動 ダイアログをオープンします。

［コンテキスト・メニュー］

ウォッチ 1 に登録	選択している I/O レジスタ／カテゴリをウォッチ パネル（ウォッチ 1）に登録します。
最新の情報に更新	デバッグ・ツールから最新の情報を取得し，表示を更新します。 読み込み保護対象の I/O レジスタの再読み込みは行いません。 ただし，プログラム実行中は無効となります。
値を強制読み込み	読み込み保護対象の I/O レジスタの値を 1 回強制的に読み込みます。
移動 ...	指定位置へ移動 ダイアログをオープンします。
カテゴリを作成	新規カテゴリ（フォルダ）を追加します。テキスト・ボックスに直接カテゴリ名を入力します。 なお，新規に作成できるカテゴリの数に制限はありませが，カテゴリ内にカテゴリを作成することはできません。 ただし，プログラム実行中は無効となります。
コピー	選択している範囲の文字列をクリップ・ボードにコピーします。 I/O レジスタ／カテゴリが選択状態の場合は，その項目をコピーします。 なお，コピーした項目は，に貼り付け可能です。

削除	選択している範囲の文字列を削除します。 空のカテゴリが選択状態の場合は，その項目を削除します（I/O レジスタの削除不可）。
表記	表示形式を指定するため，次のカスケード・メニューを表示します。
16 進数	選択している項目の値を 16 進数で表示します（デフォルト）。
符号付き 10 進数	選択している項目の値を符号付き 10 進数で表示します。
符号無し 10 進数	選択している項目の値を符号なし 10 進数で表示します。
8 進数	選択している項目の値を 8 進数で表示します。
2 進数	選択している項目の値を 2 進数で表示します。
ASCII	選択している項目の値を ASCII コードで表示します。
16 進数値を併記	選択している項目の値表示の末尾に，その値の 16 進数表記を “（）” で囲んで併記します。
表示色をリセット	選択している I/O レジスタに対して，プログラム実行により値が変化したことを示す強調表示をリセットします。

ローカル変数 パネル

ローカル変数の内容の表示，および値の変更を行います（「2.10.5 ローカル変数を表示／変更する」参照）。
なお，このパネルは，デバッグ・ツールと接続時のみオープンすることができます。

注意 1. プログラム実行中は，このパネルには何も表示されません。プログラムの実行が停止したタイミングで，各エリアの表示を行います。

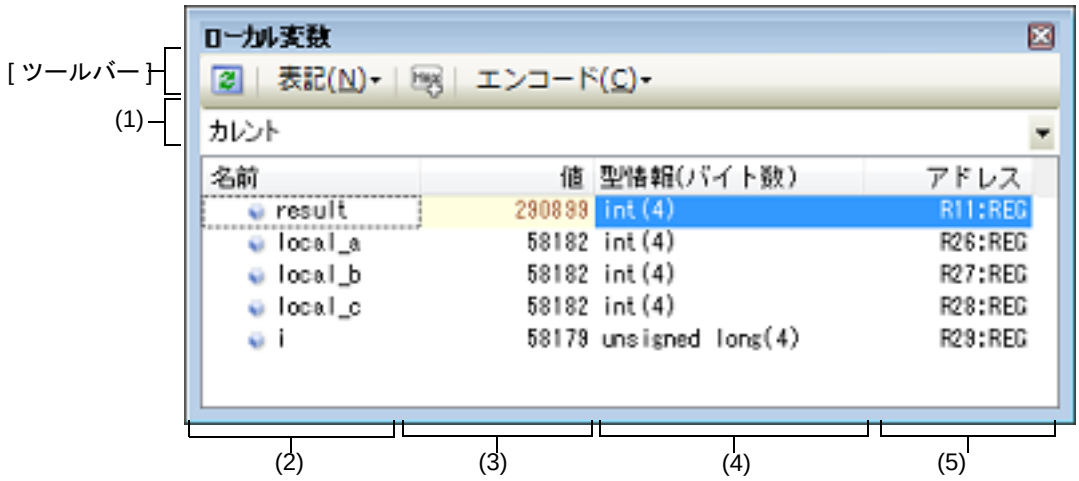
注意 2. コンパイラによる最適化のため，対象となる変数を使用していないブロックでは変数データがスタック／レジスタに存在しない場合があります。この場合は対象となる変数は表示されません。

注意 3. 選択しているマイクロコントローラがマルチコア対応版の場合，コア（PE）の選択を切り替えることにより，選択した PE に対応した内容の表示／値の変更を行います（「2.7 コア（PE）の選択」参照）。

備考 1. ツールバーの 100% ，または［Ctrl］キーを押下しながらマウス・ホイールを前後方に動かすことにより，本パネルの表示を拡大／縮小することができます。

備考 2. パネル上の各エリアの区切り線をダブルクリックすることにより，該当エリアの内容を省略することなく表示可能な最小幅に変更することができます。

図 A.22 ローカル変数 パネル



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［ツールバー］
- ［［ファイル］メニュー（ローカル変数 パネル専用部分）］
- ［［編集］メニュー（ローカル変数 パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］メニュー→［ローカル変数］を選択

［各エリアの説明］

(1) スコープ・エリア
表示するローカル変数のスコープをドロップダウン・リストにより選択します。
選択できる項目は次のとおりです。

項目	動作
カレント	カレント PC 値のスコープでのローカル変数を表示します。
＜*深さ*＞ ＜ *関数名* ()[*ファイル名* # *行番号*] ＞[注]	呼び出し元の関数のスコープでのローカル変数を表示します。 プログラム実行後，選択したスコープが存在するかぎり，ここで選択したスコープを保ちます。

注 コール・スタック パネルで表示している関数呼び出し元を表示します。

(2) ［名前］エリア

ローカル変数名，および関数名を表示します。関数の引数もローカル変数として表示します。
また，配列，ポインタ型変数，構造体／共用体は，階層構造をツリー形式で表示します。
このエリアを編集することはできません。
表示される各アイコンの意味は次のとおりです。

	変数を示します。 Auto 変数，内部スタティック変数，Register 変数の表示も行います[注]。 配列，ポインタ型変数，構造体／共用体は，階層構造をツリー形式で表示します。 先頭に “+” マークがある場合は，これをクリックすることにより次を展開表示します（展開後 “-” マークに変化）。	
	配列	配列中の全要素
	ポインタ型変数	ポインタが指し示す先の変数 なお，ポインタが指し示す先がポインタの場合は，さらに “+” マークを付与し，これをクリックすることにより参照先を表示します。 ただし，ポインタの指す値が不明な場合は，“?” を表示します。
	構造体／共用体	構造体／共用体の全メンバ
	引数を示します。	
	関数を示します。	

注 Auto 変数を表示する場合，関数のプロローグ（関数の “{”）やエピローグ（関数の “}”）ではローカル変数の正確な値を表示することができません（Auto 変数のアドレスは，スタック・ポインタ（SP）からの相対アドレスとなり，関数内で SP の値が確定するまで確定しません。プロローグやエピローグでは SP の操作が行われており，SP の値が確定していません。このため，プロローグやエピローグでは正確な値の表示ができません）。

このエリアは，次の機能を備えています。

(a) ウォッチ式の登録

C 言語変数をウォッチ式としてウォッチ パネルに登録することができます。
操作方法についての詳細は，「2.10.6.1 ウォッチ式を登録する」を参照してください。

備考 登録したウォッチ式には，自動的にスコープ指定が付与されます。

(b) メモリへのジャンプ

コンテキスト・メニューの［メモリへジャンプ］を選択することにより，選択しているローカル変数が配置されているアドレスにキャレットを移動した状態でメモリ パネル（メモリ 1）がオープンします（すでにオープンしている場合はメモリ パネル（メモリ 1）にジャンプ）。

(3) ［値］エリア

ローカル変数の値を表示／変更します。
表示進数や文字列のエンコードは，ツールバーのボタン，またはコンテキスト・メニューより選択することができます。また，常に 16 進数表示を併記する表示形式を選択することもできます。
ローカル変数の値として表示されるマークや色の意味は次のとおりです（文字色／背景色はオプション ダイアログにおける［全般 - フォントと色］カテゴリの設定に依存）。

表示例（デフォルト）			説明
0x0	文字色	青色	ユーザにより，値が変更されているローカル変数値（[Enter] キーによりターゲット・メモリに書き込まれます）
	背景色	標準色	
0x0	文字色	茶色	プログラムの実行により，値が変化したローカル変数値[注] プログラムを再び実行することにより，強調色がリセットされます。
	背景色	クリーム	
?	文字色	グレー	ローカル変数の値を取得できない場合
	背景色	標準色	

注 プログラムの実行開始位置からブレークした位置で同じ変数名を表示していて，かつ，その変数値が変化している場合が対象となります。

このエリアは，次の機能を備えています。

(a) ローカル変数値／引数値の変更
ローカル変数値，および引数値の変更は，対象ローカル変数値を選択したのち，再度クリックし，キーボードからの直接入力により行います（[Esc] キーの押下で編集モードをキャンセルします）。
ローカル変数値，および引数値を編集したのち，[Enter] キーの押下，または編集領域以外へのフォーカスの移動により，デバッグ・ツールのターゲット・メモリに書き込まれます。
ローカル変数値／引数値の変更方法についての詳細は，「2.10.5.2 ローカル変数の内容を変更する」を参照してください。

(b) ローカル変数値の保存
[ファイル] メニュー→ [名前を付けてローカル変数データを保存 ...] を選択することにより，名前を付けて保存 ダイアログをオープンし，このパネルのすべての内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
ローカル変数値の保存方法についての詳細は，「2.10.5.3 ローカル変数の表示内容を保存する」を参照してください。

(4) [型情報（バイト数）] エリア
ローカル変数の型名を表示します。表記は C 言語の記述に従います。
配列の場合は “[ ]” 内に要素数を，関数の場合は “()” 内にサイズ（バイト数）を付与して表示します。
なお，このエリアを編集することはできません。

(5) [アドレス] エリア
ローカル変数のアドレスを表示します。変数がレジスタに割り当てれられている場合は，レジスタ名を表示します。
このエリアを編集することはできません。

[ツールバー]

	デバッグ・ツールから最新の情報を取得し，表示を更新します。
表記	値の表示形式を変更する次のボタンを表示します。

自動	このパネル上の値の表記を変数ごとの規定値で表示します。
16 進数	このパネル上の値を 16 進数で表示します。
10 進数	このパネル上の値を 10 進数で表示します。
8 進数	このパネル上の値を 8 進数で表示します。
2 進数	このパネル上の値を 2 進数で表示します。
配列のインデックスを 10 進数表記	このパネル上の配列のインデックスを 10 進数で表示します（デフォルト）。
配列のインデックスを 16 進数表記	このパネル上の配列のインデックスを 16 進数で表示します。
Float	このパネル上の値を Float で表示します。 ただし，4 バイト・データ以外，または型情報を持つ場合は，規定値で表示します。
Double	このパネル上の値を Double で表示します。 ただし，8 バイト・データ以外，または型情報を持つ場合は，規定値で表示します。
	値表示の末尾に，その値の 16 進数表記を “（）” で囲んで併記します。
エンコード	文字列変数のエンコードを変更する次のボタンを表示します。
ASCII	文字列変数を ASCII コードで表示します（デフォルト）。
Shift_JIS	文字列変数を Shift_JIS コードで表示します。
EUC-JP	文字列変数を EUC-JP コードで表示します。
UTF-8	文字列変数を UTF-8 コードで表示します。
UTF-16	文字列変数を UTF-16 コードで表示します。

[[ファイル] メニュー（ローカル変数 パネル専用部分）]

ローカル変数 パネル専用の［ファイル］メニューは次のとおりです（その他の項目は共通）。
ただし，プログラム実行中はすべて無効となります。

ローカル変数データを保存	このパネルの内容を前回保存したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存します（「(b) ローカル変数値の保存」参照）。 なお，起動後に初めてこの項目を選択した場合は，［名前を付けてローカル変数データを保存 ...］の選択と同等の動作となります。
名前を付けてローカル変数データを保存 ...	このパネルの内容を指定したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存するために，名前を付けて保存 ダイアログをオープンします（「(b) ローカル変数値の保存」参照）。

[[編集] メニュー（ローカル変数 パネル専用部分）]

ローカル変数 パネル専用の［編集］メニューは次のとおりです（その他の項目はすべて無効）。

コピー	選択している行の内容，または文字列をクリップ・ボードにコピーします。
すべて選択	項目をすべて選択状態にします。
名前の変更	選択しているローカル変数の値を変更するために，編集モードに移行します（「2.10.5.2 ローカル変数の内容を変更する」参照）。 ただし，プログラム実行中は無効となります。
検索 ...	検索・置換 ダイアログを ［一括検索］ タブが選択状態でオープンします。

置換 ...	検索・置換 ダイアログを［一括置換］タブが選択状態でオープンします。

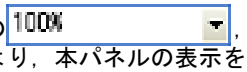
［コンテキスト・メニュー］

ウォッチ 1 に登録	選択しているローカル変数をウォッチ パネル（ウォッチ 1）に登録します。
コピー	選択している行の内容，または文字列をクリップ・ボードにコピーします。
表記	表示形式を指定するために，次のカスケード・メニューを表示します。
自動	このパネル上の値の表記を変数ごとの規定値で表示します（デフォルト）。
16 進数	このパネル上の値を 16 進数で表示します。
10 進数	このパネル上の値を 10 進数で表示します。
8 進数	このパネル上の値を 8 進数で表示します。
2 進数	このパネル上の値を 2 進数で表示します。
配列のインデックスを 10 進表記	このパネル上の配列のインデックスを 10 進数で表示します（デフォルト）。
配列のインデックスを 16 進表記	このパネル上の配列のインデックスを 16 進数で表示します。
Float	このパネル上の値を Float で表示します。 ただし，4 バイト・データ以外，または型情報を持つ場合は，規定値で表示します。
Double	このパネル上の値を Double で表示します。 ただし，8 バイト・データ以外，または型情報を持つ場合は，規定値で表示します。
16 進数値を併記	値表示の末尾に，その値の 16 進数表記を “（）” で囲んで併記します。
エンコード	文字コードを指定するため，次のカスケード・メニューを表示します。
ASCII	文字列変数を ASCII コードで表示します。
Shift_JIS	文字列変数を Shift_JIS コードで表示します（デフォルト）。
EUC-JP	文字列変数を EUC-JP コードで表示します。
UTF-8	文字列変数を UTF-8 コードで表示します。
UTF-16	文字列変数を UTF-16 コードで表示します。
メモリへジャンプ	選択している行が示すアドレスにキャレットを移動した状態で，メモリ パネル（メモリ 1）がオープンします。

ウォッチ パネル

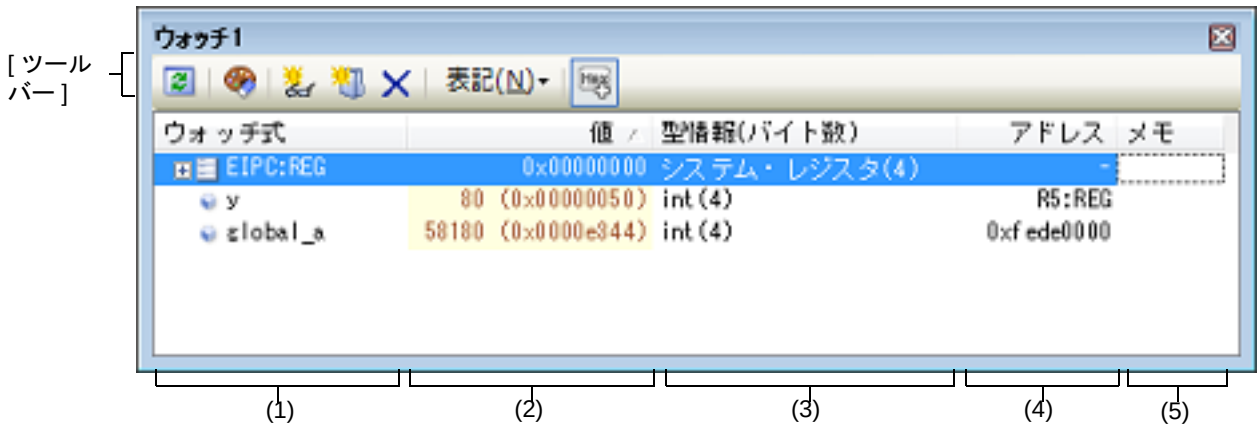
登録したウォッチ式の内容の表示，および値の変更を行います（「2.10.6 ウォッチ式を表示／変更する」参照）。
このパネルは，最大 4 個までオープンすることができます。各パネルは，タイトルバーの“ウォッチ 1”，“ウォッチ 2”，“ウォッチ 3”，“ウォッチ 4”の名称で識別され，それぞれ個別にウォッチ式の登録／削除／移動を行うことができます。
ウォッチ式の登録はこのパネル上から行えますが，エディタ パネル／逆アセンブル パネル／メモリ パネル／CPU レジスタ パネル／ローカル変数 パネル／IOR パネルより行うことも可能です。
ウォッチ式が登録されている状態のパネルをクローズした場合，そのパネルは非表示となりますが，登録されていたウォッチ式の情報は保持されます（再度そのパネルをオープンした際に，ウォッチ式が登録されている状態でオープンします）。
プログラムの実行後，ウォッチ式の値が変化すると表示を自動的に更新します（ステップ実行時には，ステップ実行ごとに表示を逐次更新）。
また，リアルタイム表示更新機能を有効にすることにより，プログラム実行中であっても，値の表示をリアルタイムに更新することも可能です。
なお，このパネルは，デバッグ・ツールと接続時のみオープンすることができます。

注意 選択しているマイクロコントローラがマルチコア対応版の場合，コア（PE）の選択を切り替えることにより，選択した PE に対応した内容の表示／値の変更を行います（「2.7 コア（PE）の選択」参照）。

備考 1. ツールバーの  ，または［Ctrl］キーを押下しながらマウス・ホイールを前後方に動かすことにより，本パネルの表示を拡大／縮小することができます。

備考 2. パネル上の各エリアの区切り線をダブルクリックすることにより，該当エリアの内容を省略することなく表示可能な最小幅に変更することができます。

図 A.23 ウォッチ パネル



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［ツールバー］
- ［［ファイル］メニュー（ウォッチ パネル専用部分）］
- ［［編集］メニュー（ウォッチ パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］メニュー→［ウォッチ］→［ウォッチ *1* ～ *4*］を選択

［各エリアの説明］

(1) ［ウォッチ式］エリア
登録しているウォッチ式を一覧で表示します。

このエリアの表タイトル部をクリックすることにより，一覧内のウォッチ式をアルファベット順でソートすることができます。
また，カテゴリ（フォルダ）を自由に作成してウォッチ式を分類し，ツリー形式で表示することができます（「(a) ツリーの編集」参照）。
表示される各アイコンの意味は次のとおりです。

	このカテゴリに属するウォッチ式を表示している状態を示します。アイコンをダブルクリック，または “-” マークをクリックすると，カテゴリを閉じウォッチ式を非表示にします。
	このカテゴリに属するウォッチ式が非表示の状態を示します。アイコンをダブルクリック，または “+” マークをクリックすると，カテゴリを開きウォッチ式を表示します。
	ウォッチ式が変数であることを示します。 配列，ポインタ型変数，構造体／共用体を示すウォッチ式の先頭には，“+” ／ “-” マークを表示し，これをクリックすることにより展開／折りたたみ表示します。
	ウォッチ式が関数であることを示します。
	ウォッチ式が即値であることを示します。
	ウォッチ式が式であることを示します。
	ウォッチ式が I/O レジスタであることを示します。
	ウォッチ式が CPU レジスタであることを示します。 下階層のレジスタ（レジスタの部分）を持つウォッチ式の先頭には，“+” ／ “-” マークを表示し，これをクリックすることにより展開／折りたたみ表示します。

このエリアは，次の機能を備えています。

(a) ツリーの編集

ウォッチ式をカテゴリ（フォルダ）で分類し，ツリー形式で表示することができます。
カテゴリを新規に作成する場合は，作成したい位置にキャレットを移動したのち，ツールバーの  ボタンのクリック，またはコンテキスト・メニューの［カテゴリを作成］を選択し，任意にカテゴリ名称を入力することにより行います。
なお，カテゴリを削除する場合は，削除したいカテゴリを選択したのち，ツールバーの  ボタンのクリック，またはコンテキスト・メニューの［削除］を選択します。
また，作成したカテゴリ名を編集する場合は，編集したいカテゴリ名を選択したのち，次のいずれかの操作により行います。

- 再度クリック後，キーボードよりカテゴリ名を直接編集
- ［編集］メニュー→［名前の変更］を選択後，キーボードよりカテゴリ名を直接編集
- ［F2］キーを押下後，キーボードよりカテゴリ名を直接編集

カテゴリを作成したのち，登録済みのウォッチ式をカテゴリ内に直接ドラッグ・アンド・ドロップすることにより，各ウォッチ式をカテゴリで分類したツリー形式で表示します。
同様に，カテゴリとウォッチ式の表示の順番（上下位置）も，ドラッグ・アンド・ドロップ操作により自由に変更することができます。

注意 1. カテゴリ内にカテゴリを作成することはできません。

注意 2. 1 つのウォッチ パネルにおいて，カテゴリは 1500 個まで作成することができます（上限値を越えて作成しようとした場合，メッセージを表示します）。

備考 ウォッチ式／カテゴリを他のウォッチ パネル（ウォッチ 1 ～ウォッチ 4）にドラッグ・アンド・ドロップすると，ドロップ先のウォッチ パネルにウォッチ式／カテゴリがコピーされます。

(b) 展開／折りたたみ表示

配列，ポインタ型変数，構造体／共用体，レジスタ（部分を表す名前がついているもののみ）を示すウォッチ式の先頭には，“+” マークを表示し，これをクリックすることにより次を展開表示します（展開後 “-” マークに変化）。

ウォッチ式	展開表示の際の内容
配列	配列中の全要素 コンテキスト・メニューの［表記］→［ASCII］を選択することにより，文字列として表示可能です（最大表示文字数：256 文字）。 ただし，エンコードの種類により表示不可能な場合は，“.” または “?” を表示します。
ポインタ型変数	ポインタが指し示す先の変数
構造体／共用体	構造体／共用体の全メンバ
レジスタ	レジスタを構成するビット／ビット列の名称 例）ECR レジスタの場合 FECC レジスタ EICC レジスタ

(c) 新規ウォッチ式の登録
新規にウォッチ式を登録する方法には，次の 3 通りがあります。

<1> 他のパネルからのウォッチ式の登録
他のパネル上において，ウォッチ式として登録したい対象に対して，次のいずれかの操作を行います。

- 対象文字列を選択したのち，任意のウォッチ パネル（ウォッチ 1 ～ウォッチ 4）上のこのエリアに直接ドラッグ・アンド・ドロップ
- 対象文字列を選択したのち，または対象文字列のいずれかにキャレットを移動したのち（対象は自動的に決定されます），コンテキスト・メニューの［ウォッチ 1 に登録］を選択
- 対象文字列を［編集］メニュー→［コピー］したのち，任意のウォッチ パネル（ウォッチ 1 ～ウォッチ 4）上のこのエリアで［編集］メニュー→［貼り付け］を選択

なお，この操作が可能なパネルとウォッチ式として登録可能な対象との関係は次のとおりです。

表 A.2 各パネルとウォッチ式として登録可能な対象の関係

パネル名	ウォッチ式として登録可能な対象
エディタ パネル	C 言語変数 /CPU レジスタ /I/O レジスタ／アセンブラ・シンボル
逆アセンブル パネル	C 言語変数 /CPU レジスタ /I/O レジスタ／アセンブラ・シンボル
CPU レジスタ パネル	CPU レジスタ[注]
ローカル変数 パネル	C 言語変数（ローカル変数）
IOR パネル	I/O レジスタ[注]

注 自動的にスコープ指定がウォッチ式に付与されます。

<2> ウォッチ パネル上での直接登録
任意のウォッチ パネル（ウォッチ 1 ～ウォッチ 4）において，ツールバーの  ボタンをクリック，またはコンテキスト・メニューの［新規ウォッチ式を追加］を選択することにより，このエリアの最下段に新規ウォッチ式用のエントリ・ボックスが表示されます。
エントリ・ボックスの［ウォッチ式］エリアにおいて，キーボートより直接ウォッチ式を入力したのち，［Enter］キーを押下します。
ウォッチ式の入力形式については，「(b) ウォッチ式と演算子」を参照してください。

また，ウォッチ式は，スコープを指定して登録することができます。スコープ指定してウォッチ式を登録した場合の扱いは次のとおりです。

注意 1. ロード・モジュール名，またはファイル名に空白や次の記号が含まれている場合，名前をダブル・クォーテーション “ ” で囲んでください。
$，#，(，)，[，]，&，^，~，%，+，−，*，/，:，?，’，|，¥，<，>，！
例：“c:¥folder¥prog.abs”$file.c#func#var

注意 2. 同名の関数が存在する場合はパラメータの型名を明記してください（例：func(int, int)）。

表 A.3 C 言語関数をスコープ指定してウォッチ登録した場合の扱い

スコープ指定	ロード・モジュール名	ソース・ファイル名	関数名	検索対象
prog$file#func	prog	file	func	スタティック関数
prog$func	prog	グローバル	func	グローバル関数
file#func	カレント	file	func	スタティック関数
func	カレント	カレント	func	すべて注

注 カレント PC 値のスコープからスタティック関数，グローバル関数の順で検索します。スコープ範囲外のスタティック関数は検索対象外になります。

表 A.4 C 言語変数をスコープ指定してウォッチ登録した場合の扱い

スコープ指定	ロード・モジュール名	ソース・ファイル名	関数名	変数名	検索対象
prog$file#func#var	prog	file	func	var	スタティック関数内スタティック変数注 1
prog$file#var	prog	file	グローバル	var	ファイル内スタティック変数
prog$var	prog	グローバル	グローバル	var	グローバル変数
file#func#var	カレント	file	func	var	スタティック関数内スタティック変数注 1
file#var	カレント	file	グローバル	var	ファイル内スタティック変数
var	カレント	カレント	カレント	var	すべて注 2

注 1. カレント PC 値が指定関数内にある場合は，スタティック宣言されていないローカル変数も検索対象となります。

注 2. カレント PC 値のスコープからローカル変数，ファイル内スタティック変数，グローバル変数の順で検索します。スコープ範囲外のローカル変数およびファイル内スタティック変数は，検索対象外となります。

表 A.5 CPU レジスタをスコープ指定してウォッチ登録した場合の扱い

スコープ指定	レジスタ・バンク	CPU レジスタ名
r10:REG	（なし）	r10

表 A.6 I/O レジスタをスコープ指定してウォッチ登録した場合の扱い

スコープ指定	I/O レジスタ名
P0:IOR	P0
P0	P0

備考 1. このエリアで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

備考 2. 即値はアドレスとして扱われます。また，即値に演算子を使用することはできません。

備考 3. ウォッチ式として，シンボルを使用した演算式を指定することはできません。

備考 4. 同名の C 言語変数 /CPU レジスタ /I/O レジスタが存在する際に，スコープ指定せずにそれらを登録した場合，次の順にシンボルを解決し，値を表示します。
C 言語変数＞ CPU レジスタ＞ I/O レジスタ

備考 5. 同名のローカル変数とグローバル変数が存在する際に，スコープを指定せずにシンボル名のみ登録した場合，カレント PC 値のスコープを基にシンボルを解決し，値を表示します。

備考 6. ウォッチ式として単に “I” と指定した場合，虚数のキーワードとして解釈します。レジスタ “I” の値を取得する場合は，レジスタの後ろに “:REG” を付加してください。

備考 7. IOR パネル /CPU レジスタ パネルよりウォッチ式を登録した場合，ウォッチ式には自動的にスコープ指定が付与されます。

<3> 他のアプリケーションからの登録
外部エディタなどから，C 言語変数 /CPU レジスタ /I/O レジスタ／アセンブラ・シンボルの文字列を選択し，次のいずれかの操作を行います。

- 対象文字列を，任意のウォッチ パネル（ウォッチ 1 ～ウォッチ 4）上のこのエリアに直接ドラッグ・アンド・ドロップ
- 対象文字列をクリップ・ボードにコピーしたのち，任意のウォッチ パネル（ウォッチ 1 ～ウォッチ 4）上のこのエリアで［編集］メニュー→［貼り付け］を選択

注意 1. 1 つのウォッチ パネルにおいて，ウォッチ式は 3000 個まで登録することができます（上限値を越えて登録しようとした場合，メッセージを表示します）。

注意 2. コンパイラによる最適化のため，対象となる変数を使用していないブロックでは変数データがスタック／レジスタに存在しない場合があります。この場合，対象となる変数をウオッチ式として登録しても値の表示は “?” のままとなります。

備考 1. 各ウォッチ パネル（ウォッチ 1 ～ウォッチ 4）上で登録したウォッチ式は，それぞれ個別に管理され，プロジェクトのユーザ情報として保存されます。

備考 2. ウォッチ式は，同名を複数登録することができます。

備考 3. 登録したウォッチ式をファイルにエクスポートし，そのファイルをインポートすることにより，ウォッチ式を再登録することができます（「2.10.6.8 ウォッチ式をエクスポート／インポートする」参照）。

(d) ウォッチ式の編集
登録済みのウォッチ式の編集は，対象ウォッチ式をダブルクリックし，対象ウォッチ式を編集モードにしたのち，キーボードから編集内容を直接入力して行います（［Esc］キーの押下で編集モードをキャンセルします）。
ウォッチ式を編集したのち，［Enter］キーを押下すると編集を完了します。

(e) ウォッチ式の削除
ツールバーの  ボタンのクリック，またはコンテキスト・メニューの［削除］を選択することにより，選択しているウォッチ式を削除します。

(f) 各種イベントの設定
コンテキスト・メニューの［アクセス・ブレークの設定］／［トレース出力］を選択することにより，選択しているウォッチ式に各種イベントを設定することができます。
アクセス系のブレーク・イベントが設定された場合，ウォッチ式のアイコンが変化します（ウォッチ式のアイコンの下にブレーク・イベントのイベント・マークを重ねて表示）。トレース・イベントの場合は，ウォッチ式のマークに変化はありません。
イベントを設定することにより，設定したイベントの詳細情報がイベント パネルに反映されます。
ただし，イベントの設定は，対象となるウォッチ式がグローバル変数／関数内スタティック変数／ファイル内スタティック変数 /I/O レジスタの場合のみ行うことができます。
なお，各種イベントの設定方法についての詳細は，次を参照してください。

- 「2.9.5 変数 /I/O レジスタへのアクセスで停止する」
- 「2.12.4 条件を満たしたときのみの実行履歴を収集する【シミュレータ】」

(g) メモリ定義アドレスへのジャンプ
コンテキスト・メニューの［メモリへジャンプ］を選択することにより，選択しているウォッチ式が定義されているアドレスにキャレットを移動した状態でメモリ パネル（メモリ 1）がオープンします（すでにオープンしている場合は，メモリ パネル（メモリ 1）にジャンプ）。
ただし，同時に複数のウォッチ式を選択している場合，または I/O レジスタ /CPU レジスタを選択している場合は，無効となります。

(2) ［値］エリア
登録しているウォッチ式の値を表示／変更します。
なお，ウォッチ式が関数ポインタの場合は，関数名を表示します。
表示進数やエンコードは，ツールバーのボタン，またはコンテキスト・メニューより選択することができます。
また，常に 16 進数値を併記する表示形式を選択することもできます。

なお，デフォルトの表示形式は，ウォッチ式の型に依存して，次のように自動的に決定されます。

表 A.7 ウォッチ式の表示形式（デフォルト）

ウォッチ式の型	表示形式
char，signed char，unsigned char	ASCII 文字に続き“（）”内に 16 進数値を併記
short，signed short，short int，signed short int，int，signed，signed int，long，signed long，long int，signed long int	符号付き 10 進数値に続き“（）”内に 16 進数値を併記
unsigned short，unsigned short int，unsigned，unsigned int，unsigned long，unsigned long int	符号なし 10 進数値に続き“（）”内に 16 進数値を併記
float	Float（サイズが 4 バイトの場合）値に続き“（）”内に 16 進数値を併記
double，long double	Double（サイズが 8 バイトの場合）値に続き“（）”内に 16 進数値を併記
char，signed char，unsigned char へのポインタ	文字列 エンコード：Shift_JIS
char，signed char，unsigned char 以外へのポインタ	16 進数
char，signed char，unsigned char 型の配列	文字列 エンコード：Shift_JIS
bit，boolean，_boolean	符号なし 10 進数値に続き“（）”内に 16 進数値を併記
列挙型	列挙定数値に続き“（）”内に 16 進数値を併記
ラベル，即値アドレス，EQU シンボル	符号付き 10 進数値に続き“（）”内に 16 進数値を併記
ビット・シンボル	符号なし 10 進数値に続き“（）”内に 16 進数値を併記
その他	16 進数

また，ウォッチ式の値として表示されるマークや色の意味は次のとおりです（文字色／背景色はオプション ダイアログにおける［全般 - フォントと色］カテゴリの設定に依存）。

表示例（デフォルト）			説明
0x0	文字色	青色	ユーザにより，値が変更されているウォッチ式の値（[Enter] キーによりターゲット・メモリに書き込まれます）
	背景色	標準色	
0x0	文字色	ピンク	リアルタイム表示更新機能を行っているウォッチ式の値
	背景色	標準色	
0x0	文字色	茶色	プログラムの実行により，値が変化したウォッチ式の値 ツールバーの  ボタン，またはコンテキスト・メニューの［表示色をリセット］を選択することにより，強調表示をリセットします。
	背景色	クリーム	
?	文字色	グレー	存在しない変数をウォッチ式として登録した場合，またはウォッチ式の値を取得できなかった場合（読み込み保護対象の I/O レジスタ[注]や，変数がスコープを外れた場合など）
	背景色	標準色	

注 読み込み動作によってマイクロコントローラが動作してしまう I/O レジスタは，読み込み保護対象となるため，値の読み込みは行いません（[値] に “?” を表示）。
読み込み保護対象の I/O レジスタ の内容を取得したい場合は，コンテキスト・メニューの［値を強制読み込み］を選択することで，1 度だけ値の読み込みが可能です。

備考 1. 各ウォッチ式は，登録された順序で値の取得を行います。
このため，同一の I/O レジスタを複数登録した場合，値を取得するタイミングに差が生じるため，表示される値が異なる場合があります。

備考 2. 16 進数値を併記している場合では，指定表記の値と 16 進数の値を個別に読み出します。
このため，値を取得するタイミングに差を生じるため，指定表記値と 16 進数値が異なる場合があります。

このエリアは，次の機能を備えています。

(a) リアルタイム表示更新機能
リアルタイム表示更新機能を使用することにより，プログラムが停止している状態の時だけでなく，実行中の状態であっても，登録したウォッチ式の値の表示／変更を行うことができます。
リアルタイム表示更新機能についての詳細は，「2.10.1.4 プログラム実行中にメモリの内容を表示／変更する」を参照してください。

(b) ウォッチ式の値の変更
ウォッチ式の値の変更は，対象ウォッチ式の値を選択したのち，再度クリックし，キーボードからの直接入力により行います（［Esc］キーの押下で編集モードをキャンセルします）。
ウォッチ式の値を編集したのち，［Enter］キーの押下，または編集領域以外へのフォーカスの移動により，ターゲット・メモリに書き込まれます。
ウォッチ式の値の変更方法についての詳細は，「2.10.6.6 ウォッチ式の内容を変更する」を参照してください。

(c) ウォッチ式の値の保存
［ファイル］メニュー→［名前を付けてウォッチ・データを保存 ...］を選択することにより，名前を付けて保存 ダイアログをオープンし，このパネルのすべての内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
ウォッチ式の値の保存方法については，「2.10.6.9 ウォッチ式の表示内容を保存する」を参照してください。

(3) ［型情報（バイト数）］エリア
ウォッチ式に対して，次の形式で型情報を表示します。

ウォッチ式	表示形式	
単独の CPU レジスタ	＜*CPU レジスタの種類*＞（＜*サイズ*[注 1]＞）	
単独の I/O レジスタ	＜*I/O レジスタの種類*＞（＜*アクセス属性*＞ ＜*アクセス可能サイズ*＞＜*サイズ*[注 1]＞）	
	アクセス属性	R ： 読み出しのみ可能 W ： 書き込みのみ可能 R/W ： 読み出し／書き込み可能
	アクセス可能サイズ	アクセス可能なすべてのサイズを，ビット単位で小さい順に“,” で区切り列挙します（1 ～ 32 ビット）。
判別不能	?	
上記以外	＜*C コンパイラの判定に従ったウォッチ式の型*[注 2]＞（＜*サイズ*[注 1]＞）	

注 1. ウォッチ式のサイズをバイト単位で示します。
ただし，ビット IOR/C 言語ビット・フィールドについては，ビット単位で表示し，数値の末尾に“ ビット ” 表記を付与します。

注 2. ウォッチ式をコンパイルする際に，どの型として扱われるかを示します。

(4) ［アドレス］エリア
各ウォッチ式がマッピングされているアドレスを表示します（16 進数表記固定）。
ただし，ウォッチ式が，単独の CPU レジスタの場合は “-” を，また判別不能の場合では，“?” を表示します。

備考 ウォッチ式が I/O レジスタで，ビット・レジスタの場合は，次のようにビット・オフセット値を付与して表示します。

例 アドレス “0xFF40” のビット 4 に割り当てられている（ビット・レジスタ）の場合
表示内容：0xFF40.4

(5) ［メモ］エリア
ウォッチ式／カテゴリに対して，ユーザが自由にコメントを入力することができます。

このエリアに入力したコメントの内容は，各ウォッチ式／カテゴリに対して個別に保持され，プロジェクトのユーザ情報として保存されます。したがって，ウォッチ式／カテゴリを削除すると，対応するメモの内容も破棄されます。
ただし，配列，レジスタなどを展開表示している場合，各展開要素に対してコメントを入力することはできません。
コメントを編集する場合は，編集したい項目をダブルクリックすることにより，選択した項目が編集モードとなります（[Esc] キーの押下で編集モードをキャンセルします）。最大 256 文字までの文字列をキーボードより直接入力することができます（改行コードは無効）。
文字列編集後，[Enter] キーの押下，または編集領域以外へのフォーカスの移動により，文字列編集を完了します。

[ツールバー]

	登録しているウォッチ式のすべての値を再取得し，表示を更新します。 ただし，読み込み保護対象の I/O レジスタの再読み込みは行いません。
	選択しているウォッチ式に対して，プログラムの実行により値が変化したことを示す強調表示をリセットします。 ただし，プログラム実行中は無効となります。
	新規ウォッチ式を登録します。テキスト・ボックスに直接ウォッチ式を入力します（「(c) 新規ウォッチ式の登録」参照）。 なお，1 つのウォッチ パネルに登録可能なウォッチ式数は，最大 3000 個までです。
	新規カテゴリ（フォルダ）を追加します。テキスト・ボックスに直接カテゴリ名を入力します。 なお，1 つのウォッチ パネルに作成可能なカテゴリ数は，最大 1500 個までです（カテゴリ内のカテゴリ作成は不可）。
	選択している範囲の文字列を削除します。 ウォッチ式／カテゴリが選択状態の場合は，その項目を削除します。 ただし，ウォッチ式の展開項目を選択している場合は無効となります。
表記	値の表示形式を変更する次のボタンを表示します。
 自動	選択しているウォッチ式の値の表記を変数ごとの規定値（「表 A.7 ウォッチ式の表示形式（デフォルト）」参照）で表示します（デフォルト）。
 16 進数	選択している項目の値を 16 進数で表示します。
 符号付き 10 進数	選択している項目の値を符号付き 10 進数で表示します。
 符号無し 10 進数	選択している項目の値を符号なし 10 進数で表示します。
 8 進数	選択している項目の値を 8 進数で表示します。
 2 進数	選択している項目の値を 2 進数で表示します。
 ASCII	選択している項目の値を ASCII コードで表示します。
 Float	選択している項目の値を Float で表示します。 ただし，選択しているウォッチ式が 4 バイト・データの場合のみ有効となります。
 Double	選択している項目の値を Double で表示します。 ただし，選択しているウォッチ式が 8 バイト・データの場合のみ有効となります。
	選択している項目の値表示の末尾に，その値の 16 進数表記を “（）” で囲んで併記します。 ただし，16 進数表記をしている場合は併記しません。

[[ファイル] メニュー（ウォッチ パネル専用部分）]

ウォッチ パネル専用の [ファイル] メニューは次のとおりです（その他の項目は共通）。
ただし，プログラム実行中はすべて無効となります。

ウォッチ・データを保存	このパネルの内容を前回保存したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存します（「(c) ウォッチ式の値の保存」参照）。 なお，起動後に初めてこの項目を選択した場合は，［名前を付けてウォッチ・データを保存 ...］の選択と同等の動作となります。
名前を付けてウォッチ・データを保存 ...	このパネルの内容を指定したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存するために，名前を付けて保存 ダイアログをオープンします（「(c) ウォッチ式の値の保存」参照）。

［［編集］メニュー（ウォッチ パネル専用部分）］

ウォッチ パネル専用の［編集］メニューは次のとおりです（その他の項目はすべて無効）。

切り取り	選択範囲の文字列を切り取り，クリップ・ボードにコピーします。 ウォッチ式／カテゴリが選択状態の場合は，その項目を切り取ります。 ただし，ウォッチ式の展開項目を選択している場合は無効となります。
コピー	選択している範囲を文字列としてクリップ・ボードにコピー します。 ウォッチ式／カテゴリが選択状態の場合は，その項目をコピーします。 ただし，ウォッチ式の展開項目を選択している場合は無効となります。
貼り付け	テキストが編集状態の場合，クリップ・ボードの内容をキャレット位置に挿入します。 テキストが編集状態以外の場合で，ウォッチ式がクリップ・ボードにコピーされている場合は，コピーされているウォッチ式をキャレット位置に登録します。
削除	選択している範囲の文字列を削除します。 ウォッチ式／カテゴリが選択状態の場合は，その項目を削除します。 ただし，ウォッチ式の展開項目を選択している場合は無効となります。
すべて選択	テキストが編集状態の場合，すべての文字列を選択します。 テキストが編集状態以外の場合，すべてのウォッチ式／カテゴリを選択状態にします。
名前の変更	選択しているウォッチ式，またはカテゴリの名称を編集します。
検索 ...	検索・置換 ダイアログを ［一括検索］タブが選択状態でオープンします。
置換 ...	検索・置換 ダイアログを ［一括置換］タブが選択状態でオープンします。

［コンテキスト・メニュー］

アクセス・ブレークの設定	この項目は，選択しているウォッチ式がグローバル変数／関数内スタティック変数／ファイル内スタティック変数，および I/O レジスタの場合のみ有効です（複数選択不可）。 アクセス系のブレーク・イベントを設定するために，次のカスケード・メニューを表示します（「2.9.5.1 ブレーク・イベント（アクセス系）を設定する」参照）。
読み込みブレークを設定	選択しているウォッチ式に，リード・アクセスのブレーク・イベントを設定します。
書き込みブレークを設定	選択しているウォッチ式に，ライト・アクセスのブレーク・イベントを設定します。
読み書きブレークを設定	選択しているウォッチ式に，リード／ライト・アクセスのブレーク・イベントを設定します。
トレース出力	この項目は，選択しているウォッチ式がグローバル変数／関数内スタティック変数／ファイル内スタティック変数，および I/O レジスタの場合のみ有効です（複数選択不可）。 トレース関連のイベントを設定するために，次のカスケード・メニューを表示します（「2.12.4.1 ポイント・トレース・イベントを設定する 」参照）。

値をトレースに記録（読み込み時）	選択しているウォッチ式にリード・アクセスした際に，その値をトレース・メモリに記録するポイント・トレース・イベントを設定します。
値をトレースに記録（書き込み時）	選択しているウォッチ式にライト・アクセスした際に，その値をトレース・メモリに記録するポイント・トレース・イベントを設定します。
値をトレースに記録（読み書き時）	選択しているウォッチ式にリード／ライト・アクセスした際に，その値をトレース・メモリに記録するポイント・トレース・イベントを設定します。
トレース	トレース パネルをオープンし，取得したトレース・データを表示します。
リアルタイム表示更新設定	リアルタイム表示更新設定のため，次のカスケード・メニューを表示します（「(a) リアルタイム表示更新機能」参照）。
リアルタイム表示更新全体設定	リアルタイム表示更新機能の全般設定を行うため，プロパティ パネルをオープンします。
最新の情報に更新	登録しているウォッチ式のすべての値を再取得し，表示を更新します。 ただし，読み込み保護対象の I/O レジスタの再読み込みは行いません。
値を強制読み込み	読み込み保護対象の I/O レジスタの値を強制的に一度読み込みます。 ただし，プログラム実行中は無効となります。
新規ウォッチ式を追加	新規ウォッチ式を登録します。テキスト・ボックスに直接ウォッチ式を入力します（「(c) 新規ウォッチ式の登録」参照）。 なお，1 つのウォッチ パネルに登録可能なウォッチ式数は，最大 3000 個までです。
カテゴリを作成	新規カテゴリ（フォルダ）を追加します。テキスト・ボックスに直接カテゴリ名を入力します。 なお，1 つのウォッチ パネルに作成可能なカテゴリ数は，最大 1500 個までです（カテゴリ内のカテゴリ作成は不可）。
削除	選択している範囲の文字列を削除します。 ウォッチ式／カテゴリが選択状態の場合は，その項目を削除します。 ただし，ウォッチ式の展開項目を選択している場合は無効となります。
切り取り	選択している範囲の文字列を切り取ってクリップ・ボードに移動します。 ウォッチ式／カテゴリが選択状態の場合は，その項目を切り取ります。 ただし，ウォッチ式の展開項目を選択している場合は無効となります。
コピー	選択している範囲の文字列をクリップ・ボードにコピーします。 ウォッチ式／カテゴリが選択状態の場合は，その項目をコピーします。
貼り付け	テキストが編集状態の場合，クリップ・ボードの内容をキャレット位置に挿入します。 テキストが編集状態以外の場合で，ウォッチ式がクリップ・ボードにコピーされている場合は，コピーされているウォッチ式をキャレット位置に登録します。 ただし，ウォッチ式の展開項目を選択している場合は無効となります。
名前の変更	選択しているウォッチ式，またはカテゴリの名称を編集します。
ウォッチ式をインポート ...	ウォッチ式をインポートするために，ウォッチ式データ・ファイルを開く ダイアログをオープンします（「2.10.6.8 ウォッチ式をエクスポート／インポートする」参照）。
表記	表示形式を指定するため，次のカスケード・メニューを表示します。

自動	選択している項目の表記を変数ごとの規定値（「表 A.7 ウォッチ式の表示形式（デフォルト）」参照）で表示します（デフォルト）。
16 進数	選択している項目を 16 進数で表示します。
符号付き 10 進数	選択している項目を符号付き 10 進数で表示します。
符号無し 10 進数	選択している項目を符号なし 10 進数で表示します。
8 進数	選択している項目を 8 進数で表示します。
2 進数	選択している項目を 2 進数で表示します。
ASCII	選択している項目を ASCII コードで表示します。
16 進数値を併記	選択している項目の値表示の末尾に，その値の 16 進数表記を“（）”で囲んで併記します。 ただし，16 進数表記をしている場合は併記しません。
Float	選択している項目を Float で表示します。 ただし，選択しているウォッチ式が 4 バイト・データ以外，または型情報を持つ場合は，規定値（「表 A.7 ウォッチ式の表示形式（デフォルト）」参照）で表示します。
Double	選択している項目を Double で表示します。 ただし，選択しているウォッチ式が 8 バイト・データ以外，または型情報を持つ場合は，規定値（「表 A.7 ウォッチ式の表示形式（デフォルト）」参照）で表示します。
配列のインデックスを 10 進表記	すべての配列のインデックスを 10 進数で表示します。
配列のインデックスを 16 進表記	すべての配列のインデックスを 16 進数で表示します。
エンコード	文字コードを指定するため，次のカスケード・メニューを表示します。
ASCII	選択している項目を ASCII コードで表示します。
Shift_JIS	選択している項目を Shift_JIS コードで表示します（デフォルト）。
EUC-JP	選択している項目を EUC-JP コードで表示します。
UTF-8	選択している項目を UTF-8 コードで表示します。
UTF-16	選択している項目を UTF-16 コードで表示します。
サイズ表記	サイズを指定するため，次のカスケード・メニューを表示します。
1 バイト	選択している項目を 8 ビット・データとして表示します。
2 バイト	選択している項目を 16 ビット・データとして表示します。
4 バイト	選択している項目を 32 ビット・データとして表示します。
8 バイト	選択している項目を 64 ビット・データとして表示します。
メモリへジャンプ	選択しているウォッチ式が定義されているアドレスへキャレットを移動した状態でメモリ パネル（メモリ 1）をオープンします（「(g) メモリ定義アドレスへのジャンプ」参照）。
表示色をリセット	選択しているウォッチ式に対して，プログラムの実行により値が変化したことを示す強調表示をリセットします。 ただし，プログラム実行中は無効となります。

コール・スタック パネル

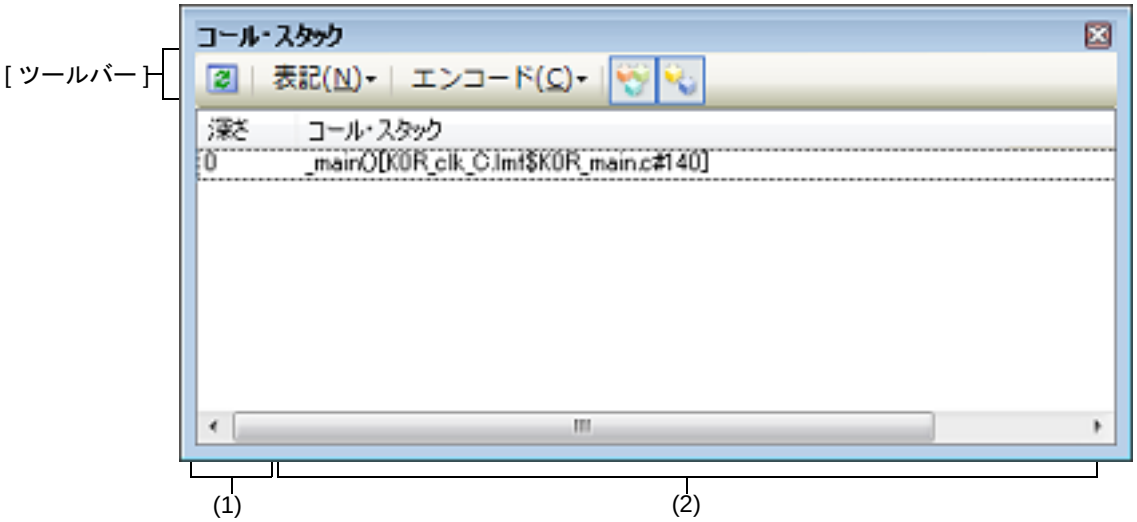
関数呼び出しのコール・スタック情報の表示を行います（「2.11.1 コール・スタック情報を表示する」参照）。
なお，このパネルは，デバッグ・ツールと接続時のみオープンすることができます。

注意 1. プログラム実行中は，このパネルには何も表示されません。
プログラムの実行が停止したタイミングで，各エリアの表示を行います。

注意 2. 選択しているマイクロコントローラがマルチコア対応版の場合，コア（PE）の選択を切り替えることにより，選択した PE に対応した内容の表示を行います（「2.7 コア（PE）の選択」参照）。

備考 ツールバーの 100% ，または［Ctrl］キーを押下しながらマウス・ホイールを前後方に動かすことにより，本パネルの表示を拡大／縮小することができます。

図 A.24 コール・スタック パネル



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［ツールバー］
- ［［ファイル］メニュー（コール・スタック パネル専用部分）］
- ［［編集］メニュー（コール・スタック パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］メニュー→［コール・スタック］を選択

［各エリアの説明］

(1) ［深さ］エリア
呼び出しの深さを表示します。
カレント PC 位置を表示している行を 0 とし，呼び出し元に 1 から順に番号を付与します。

(2) ［コール・スタック］エリア
現在のソース位置とスタックに積まれているコール・スタック情報（関数呼び出し元位置／各関数の引数など）を表示します。
ツールバーの／ボタン，またはコンテキスト・メニューの［引数表示］／［モジュール・ファイル名表示］の選択による状態により，このエリアに表示する表示形式は次のように異なります。

状態	表示形式
- 引数を表示する - モジュール・ファイル名を表示する	*＜関数＞* (*＜引数＞* = *＜引数値*注*＞* ,...)[*＜モジュール・ファイル名＞* $ *＜ファイル名＞* # *＜行番号＞*]（デフォルト）
- 引数を表示する - モジュール・ファイル名を表示しない	*＜関数＞* (*＜引数＞* = *＜引数値*注*＞* ,...)[*＜ファイル名＞* # *＜行番号＞*]
- 引数を表示しない - モジュール・ファイル名を表示する	*＜関数＞* ()[*＜モジュール・ファイル名＞* $ *＜ファイル名＞* # *＜行番号＞*]
- 引数を表示しない - モジュール・ファイル名を表示しない	*＜関数＞* ()[*＜ファイル名＞* # *＜行番号＞*]

注　引数値が文字列の場合，最大 20 文字まで表示します。

備考　配列の引数は，配列としてではなくポインタとして渡されます（C 言語仕様）。そのため，引数が配列の場合，ポインタ扱いとして表示します。

このエリアは，次の機能を備えています。

(a) ソース行／逆アセンブル行へのジャンプ
コンテキスト・メニューの［ソースへジャンプ］を選択することにより，現在選択している行が示す関数呼び出し元のソース行にキャレットを移動した状態でエディタ パネルがオープンします（すでにオープンしている場合は，エディタ パネルにジャンプ）。
また，同様に［逆アセンブルへジャンプ］を選択することにより，現在選択している行が示す関数呼び出し元のアドレスにキャレットを移動した状態で逆アセンブル パネル（逆アセンブル 1）がオープンします（すでにオープンしている場合は，逆アセンブル パネル（逆アセンブル 1）にジャンプ）。

備考　行をダブルクリックすることでも，対象ソース行へジャンプすることができます。

(b) コール・スタック情報の保存
［ファイル］メニュー→［名前を付けてコール・スタック・データを保存 ...］を選択することにより，名前を付けて保存 ダイアログをオープンし，このパネルのすべての内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
コール・スタック情報の保存方法についての詳細は，「2.11.1.4 コール・スタック情報の表示内容を保存する」を参照してください。

［ツールバー］

	デバッグ・ツールから最新の情報を取得し，表示を更新します。
表記	値の表示形式を変更する次のボタンを表示します。
 自動	このパネル上の値の表記を変数ごとの規定値で表示します（デフォルト）。
 16 進数	このパネル上の値を 16 進数で表示します。
 10 進数	このパネル上の値を 10 進数で表示します。
 8 進数	このパネル上の値を 8 進数で表示します。
 2 進数	このパネル上の値を 2 進数で表示します。
エンコード	文字列変数のエンコードを変更する次のボタンを表示します。

ASCII	このパネル上の文字列変数を ASCII コードで表示します（デフォルト）。
Shift_JIS	このパネル上の文字列変数を Shift_JIS コードで表示します。
EUC-JP	このパネル上の文字列変数を EUC-JP コードで表示します。
UTF-8	このパネル上の文字列変数を UTF-8 コードで表示します。
UTF-16	このパネル上の文字列変数を UTF-16 コードで表示します。
	モジュール・ファイル名を付加して表示します（デフォルト）。
	関数呼び出しのパラメータ（引数）を付加して表示します（デフォルト）。

[[ファイル]メニュー（コール・スタック パネル専用部分）]

コール・スタック パネル専用の［ファイル］メニューは次のとおりです（その他の項目は共通）。
ただし，プログラム実行中はすべて無効となります。

コール・スタック・データを保存	このパネルの内容を前回保存したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存します（「(b) コール・スタック情報の保存」参照）。 なお，起動後に初めてこの項目を選択した場合は，［名前を付けてコール・スタック・データを保存 ...］の選択と同等の動作となります。
名前を付けてコール・スタック・データを保存 ...	このパネルの内容を指定したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存するために，名前を付けて保存 ダイアログをオープンします（「(b) コール・スタック情報の保存」参照）。

[[編集]メニュー（コール・スタック パネル専用部分）]

コール・スタック パネル専用の［編集］メニューは次のとおりです（その他の項目はすべて無効）。

コピー	選択している行の内容を文字列としてクリップ・ボードにコピーします。
すべて選択	項目をすべて選択状態にします。
検索 ...	検索・置換 ダイアログを ［一括検索］タブが選択状態でオープンします。
置換 ...	検索・置換 ダイアログを ［一括置換］タブが選択状態でオープンします。

[コンテキスト・メニュー]

コピー	選択している行の内容を文字列としてクリップ・ボードにコピーします。
モジュール・ファイル名表示	モジュール・ファイル名を付加して表示します（デフォルト）。
引数表示	関数呼び出しのパラメータ（引数）を付加して表示します（デフォルト）。
表記	表示形式を指定するために，次のカスケード・メニューを表示します。
自動	このパネル上の値の表記を変数ごとの規定値で表示します（デフォルト）。
16 進数	このパネル上の値を 16 進数で表示します。
10 進数	このパネル上の値を 10 進数で表示します。
8 進数	このパネル上の値を 8 進数で表示します。
2 進数	このパネル上の値を 2 進数で表示します。

エンコード	文字コードを指定するため，次のカスケード・メニューを表示します。
ASCII	文字列変数を ASCII コードで表示します（デフォルト）。
Shift_JIS	文字列変数を Shift_JIS コードで表示します。
EUC-JP	文字列変数を EUC-JP コードで表示します。
UTF-8	文字列変数を UTF-8 コードで表示します。
UTF-16	文字列変数を UTF-16 コードで表示します。
逆アセンブルへジャンプ	選択している行が示す関数呼び出し元のアドレスにキャレットを移動した状態で，逆アセンブル パネル（逆アセンブル 1）がオープンします。
ソースへジャンプ	選択している行が示す関数呼び出し元のソース行にキャレットを移動した状態で，エディタ パネルがオープンします。
このときのローカル変数を表示	選択している行が示す関数のローカル変数を表示するローカル変数 パネルをオープンします。

トレース パネル

プログラムの実行履歴を記録したトレース・データの表示を行います（「2.12 実行履歴の収集」参照）。
トレース・データは，デフォルトでソース・テキストと逆アセンブル・テキストを混合して表示しますが，表示モードを選択することにより，そのどちらか一方のみを表示させることもできます。
プログラムの実行停止後，最新のトレース・データが表示されるよう表示位置を自動更新します。
なお，このパネルは，デバッグ・ツールと接続時のみオープンすることができます。

注意 【Full-spec emulator】【E1】【E20】
プロパティ パネルの［デバッグ・ツール設定］タブの［トレース］カテゴリ内［トレースの取得対象設定］プロパティにおいて［全てのコア］を選択してトレース・データを収集した場合では，コア（PE）の選択を切り替えることにより，選択した PE に対応したトレース・データの表示を行います（「2.7 コア（PE）の選択」参照）。

備考 1. ツールバーの 100% ，または［Ctrl］キーを押下しながらマウス・ホイールを前後方に動かすことにより，本パネルの表示を拡大／縮小することができます。

備考 2. パネル上の各エリアの区切り線をダブルクリックすることにより，該当エリアの内容を省略することなく表示可能な最小幅に変更することができます。

図 A.25 トレース パネル【Full-spec emulator】【E1】【E20】

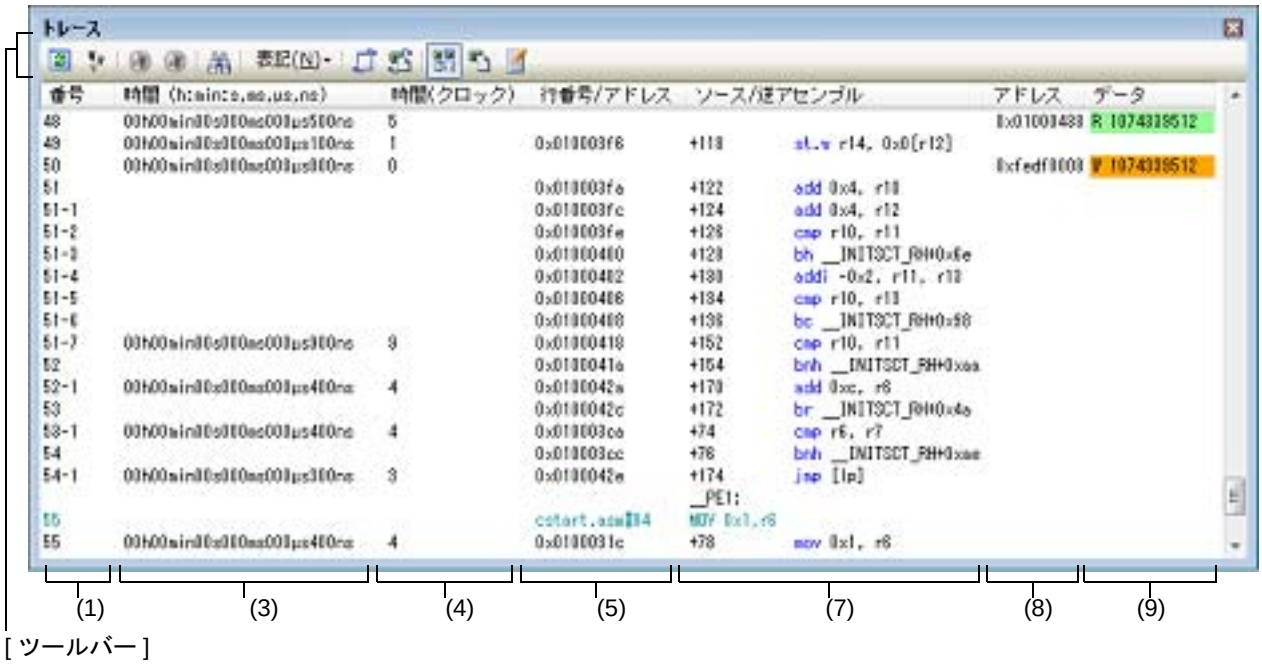
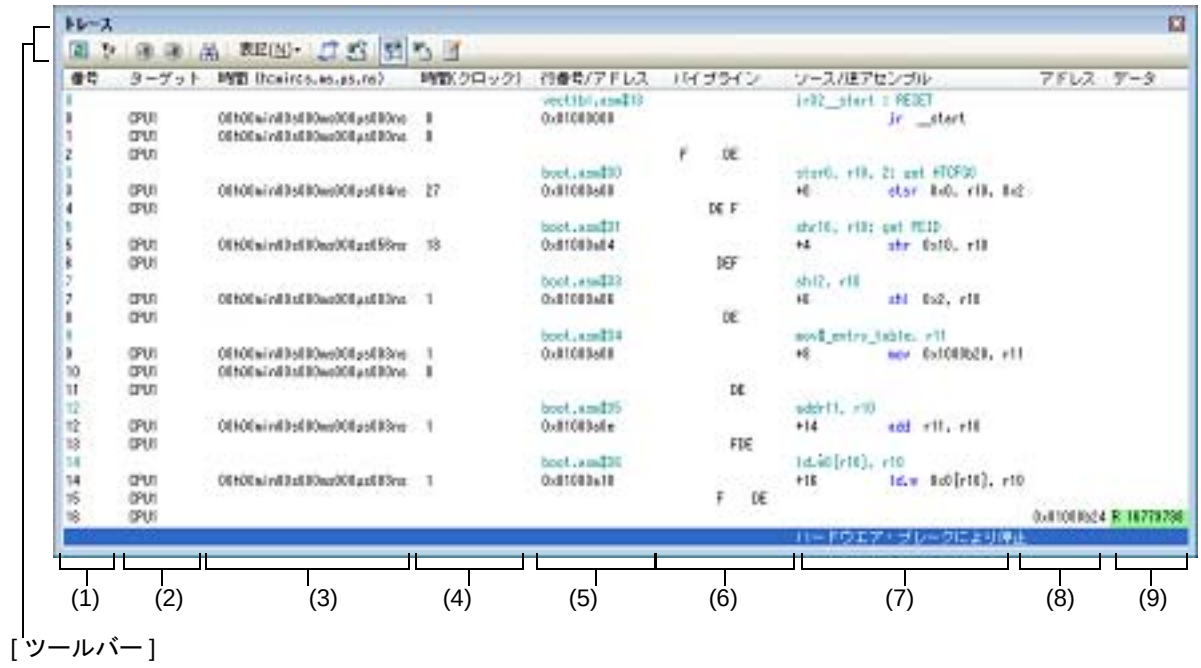


図 A.26 トレース パネル【シミュレータ】



ここでは，次の項目について説明します。

- [オープン方法]
- [各エリアの説明]
- [ツールバー]
- [[ファイル] メニュー（トレース パネル専用部分）]
- [[編集] メニュー（トレース パネル専用部分）]
- [コンテキスト・メニュー]

[オープン方法]

- [表示] メニュー→ [トレース] を選択
- エディタ パネル／逆アセンブル パネルにおいて，コンテキスト・メニューの [トレース設定] → [トレース結果の表示] を選択

[各エリアの説明]

(1) [番号] エリア
トレース・フレームに対応するトレース番号を表示します。

(2) [ターゲット] エリア【シミュレータ】
トレースの対象となったコア名を表示します。

(3) [時間 (h:min:s,ms,μs,ns)] エリア
プログラムの実行開始から各フレームの命令実行，またはメモリ・アクセスの要因が発生するまでに要した時間を“時間，分，秒，ミリ秒，マイクロ秒，ナノ秒”の単位で表示します。

備考 1. 【Full-spec emulator】【E1】【E20】
時間表示は相対時間となります。

備考 2. 【シミュレータ】
時間表示を積算時間とするか差分時間とするかは，プロパティ パネルの [デバッグ・ツール設定] タブ上の [トレース] カテゴリ内 [トレース・タイム・タグを積算] プロパティの設定に依存します。

(4) [時間 (クロック)] エリア

プログラムの実行開始から各フレームの命令実行，またはメモリ・アクセスの要因が発生するまでに要した時間を CPU クロック数で表示します。

備考 1. 【Full-spec emulator】【E1】【E20】
時間表示は差分 CPU クロック数となります。

備考 2. 【シミュレータ】
時間表示を積算 CPU クロック数とするか差分 CPU クロック数とするかは，プロパティ パネルの［デバッグ・ツール設定］タブ上の［トレース］カテゴリ内［トレース・タイム・タグを積算］プロパティの設定に依存します。

(5) ［行番号 / アドレス］エリア
ソース・ファイルの行番号，またはアセンブル命令のアドレスを表示します。
表示進数や文字列のエンコードは，ツールバーのボタン，またはコンテキスト・メニューより選択することができます。
表示形式は次のとおりです。

表示行の種類	表示形式
ソース・テキスト	*＜ファイル名＞ # ＜行番号＞*
命令（逆アセンブル）	*＜アドレス＞*
上記以外	―

備考 次の実行履歴を表示しないため，行番号は連番にはなりません。

- CPU レジスタ・アクセス
- オペランド・アクセス
- 無効フェッチ

(6) ［パイプライン］エリア【シミュレータ】
パイプラインの実行状況を表示します。
文字数は 20 文字で，1 文字が 1 クロック分のステージに相当します。各ステージは，そのステージが実行されたクロックに対する 20 の剰余の値を文字列のインデックスとして表示します。
各ステージを表す文字の意味は次のとおりです。

ステージ	文字
フェッチ	F
デコード	D
実行	E

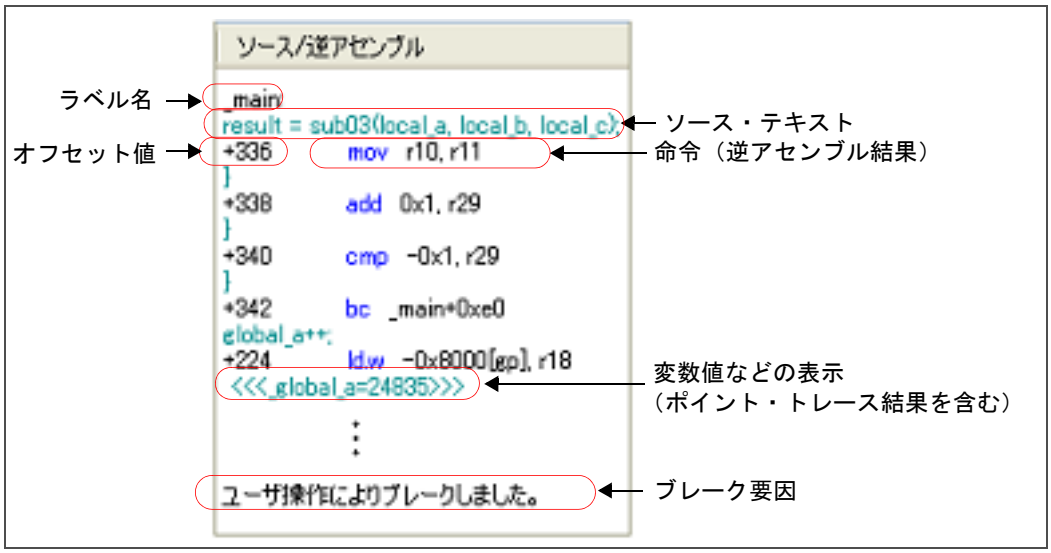
例 1. F：10 クロック目，D：11 クロック目，E：13 クロック目の場合
表示：FD_E_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

例 2. F：18 クロック目，D：19 クロック目，E：20 クロック目の場合
表示：E_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _FD

備考 “_” は半角スペースを示します。

(7) ［ソース / 逆アセンブル］エリア
収集したトレース・データを次のように表示します。
なお，表示モードの選択により，このエリアに表示される項目は異なります（「(a) 表示モード」参照）。

図 A.27 ［ソース / 逆アセンブル］エリアの表示内容（**デフォルト**）



ラベル名	アドレスにラベルが定義されている場合は，ラベル名を表示します。
オフセット値	アドレスにラベルが定義されていない場合は，一番近いラベルからのオフセット値を表示します。
ソース・テキスト	混合表示モード／ソース表示モードを選択している場合，対応するソース・テキストを表示します。 ただし，デバッグ情報が存在しない箇所を実行した場合は，“ デバッグ情報なし ” と表示します。 なお，ソース行の実行時にアクセスされた変数[注 1]/I/O レジスタの値が解析可能な場合は，その値をソース行に続き次の形式で表示します。 - <<< *変数名 = 変数値* >>> - <<<*I/O レジスタ名 =I/O レジスタ値* >>> 例：a=b; <<<a=5>>> また，ポイント・トレースの結果を表示する場合も同様の形式で表示します。
命令（逆アセンブル結果）	混合表示モード／逆アセンブル表示モードを選択している場合，対応する命令（逆アセンブル結果）を表示します[注 2]。ニモニックは強調表示されます。
ブレーク要因 【シミュレータ】	プログラムがブレークした要因を表示します。

注 1. メモリへのアクセスが発生した場合，対象アドレスにシンボルが割り当たっている場合にかぎり，該当シンボルを変数とみなして表示します。
ただし，4 バイトまでの変数が対象となります。
なお，乗算などの記述が，標準ライブラリで処理されている場合，標準ライブラリで使用している SADDR 領域のラベルが表示される場合があります。

注 2. トレース・データの取りこぼしがあった場合は，“(LOST)” を表示し，該当行全体をエラー色で表示します（エラー色はオプション ダイアログにおける［全般 - フォントと色］カテゴリの設定に依存）。

このエリアは，次の機能を備えています。

(a) 表示モード
ツールバーのボタン，またはコンテキスト・メニューの選択により，次の 3 つの表示モードを選択することができます。

表示モード	表示内容
混合表示モード	命令（逆アセンブル）／ラベル名／ソース・テキスト（対応するソース行）／ポイント・トレース結果／ブレーク要因を表示します（デフォルト）。
逆アセンブル表示モード	命令（逆アセンブル）／ラベル名／ポイント・トレース結果／ブレーク要因を表示します。

表示モード	表示内容
ソース表示モード	ソース・テキスト（対応するソース行）／ブレーク要因を表示します。 ただし，デバッグ情報が存在しない箇所を実行した場合は，“ デバッグ情報なし ” と表示します。

(b) ソース行／逆アセンブル行へのジャンプ
コンテキスト・メニューの［ソースへジャンプ］を選択することにより，現在のキャレット位置の行に対応するソース行にキャレットを移動した状態でエディタ パネルがオープンします（すでにオープンしている場合は，エディタ パネルにジャンプ）。
また，同様に［逆アセンブルへジャンプ］を選択することにより，現在のキャレット位置の行のフェッチ・アドレスにキャレットを移動した状態で逆アセンブル パネル（逆アセンブル 1）がオープンします（すでにオープンしている場合は，逆アセンブル パネル（逆アセンブル 1）にジャンプ）。

(c) 他のパネルとの連動
ツールバーの／ボタン，またはコンテキスト・メニューの［ウインドウ連動］→［ソースと連動］／［逆アセンブルと連動］を選択することにより，このパネル上のキャレット位置のアドレスをポインタとして，エディタ パネル／逆アセンブル パネルで対応箇所を連動して表示させることができます（フォーカスの移動は行いません）。

(d) ポップアップ表示
マウス・カーソルを行に重ねることにより，その行に対応するすべてのエリア（項目）のデータを縦並びにポップアップ表示します。

(e) トレース・データの保存
［ファイル］メニュー→［名前を付けてトレース・データを保存 ...］を選択することにより，データ保存 ダイアログをオープンし，このパネルの内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。
トレース・データの保存方法についての詳細は，「2.12.9 実行履歴の表示内容を保存する」を参照してください。

(8) ［アドレス］エリア
メモリ・アクセスの対象アドレスを表示します。
ただし，I/O レジスタへのアクセスの場合は，アドレスの代わりに I/O レジスタ名を表示します（アクセスが複数ある場合は次の行に表示）。
表示進数は，ツールバーのボタン，またはコンテキスト・メニューより選択することができます。

(9) ［データ］エリア
アクセスしたデータ値，およびその際のアクセス種別を表示します。
ただし，CPU レジスタ・アクセスは表示しません。
表示進数や文字列のエンコードは，ツールバーのボタン，またはコンテキスト・メニューより選択することができます。
データ値，およびアクセス種別の表示形式は次のとおりです（文字色／背景色はオプション ダイアログにおける［全般 - フォントと色］カテゴリの設定に依存）。

表示例（デフォルト）			メモリ・アクセス種別
R *データ値*	文字色	標準色	リード・アクセス
	背景色	薄緑	
W *データ値*	文字色	標準色	ライト・アクセス
	背景色	オレンジ	
RW *データ値*	文字色	標準色	リードとライト・アクセス
	背景色	薄青	
VECT *データ値*	文字色	標準色	ベクタ・リード・アクセス
	背景色	薄緑	

［ツールバー］

	デバッグ・ツールから最新の情報を取得し，表示を更新します。 ただし，トレーサ動作中は無効となります。
	トレース・メモリをクリア（初期化）し，このパネルの表示もクリアします。 ただし，トレーサ動作中は無効となります。
	トレーサの動作を開始します。 現在，このパネルで表示している内容をクリアします。 ただし，トレーサ動作中は無効となります。
	トレーサの動作を停止します。 新たに取得したトレース・データの内容に表示を更新します。 ただし，トレーサ停止中は無効となります。
	トレース検索 ダイアログをオープンします。
表記	値の表示形式を変更する次のボタンを表示します。 ただし，トレーサ動作中は無効となります。
	このパネル上の値を 16 進数で表示します（デフォルト）。
	このパネル上の値を 10 進数で表示します。
	このパネル上の値を 8 進数で表示します。
	このパネル上の値を 2 進数で表示します。
	選択している行に連動してエディタ パネルをスクロールします。
	選択している行に連動して逆アセンブル パネルをスクロールします。
	表示モードを混合表示モードにします（デフォルト）。 ただし，トレーサ動作中は無効となります。
	表示モードを逆アセンブル表示モードにします。 ただし，トレーサ動作中無効となります。
	表示モードをソース表示モードにします。 ただし，トレーサ動作中は無効となります。

［［ファイル］メニュー（トレース パネル専用部分）］

トレース パネル専用の［ファイル］メニューは次のとおりです（その他の項目は共通）。
ただし，プログラム実行中はすべて無効となります。

トレース・データを保存	トレース・データの内容を前回保存したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存します（「(e) トレース・データの保存」参照）。 なお，起動後に初めてこの項目を選択した場合は，［名前を付けてトレース・データを保存 ...］の選択と同等の動作となります。 ただし，トレーサ動作中は無効となります。
名前を付けてトレース・データを保存 ...	トレース・データの内容を指定したテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存するために，データ保存 ダイアログをオープンします（「(e) トレース・データの保存」参照）。 ただし，トレーサ動作中は無効となります。

［［編集］メニュー（トレース パネル専用部分）］

トレース パネル専用の［編集］メニューは次のとおりです（その他の項目はすべて無効）。

コピー	選択している行の内容を文字列としてクリップ・ボードにコピーします（複数行選択不可）。 ただし，トレーサ動作中は無効となります。
検索 ...	トレース検索 ダイアログをオープンします。

［コンテキスト・メニュー］

トレース・クリア	トレース・メモリをクリア（初期化）し，このパネルの表示もクリアします。 ただし，トレーサ動作中は無効となります。
トレース開始	トレーサの動作を開始します（「2.12.5.2 実行履歴の収集を再開する」参照）。 現在，このパネルで表示している内容をクリアします。 ただし，トレーサ動作中は無効となります。
トレース停止	トレーサの動作を停止します（「2.12.5.1 実行履歴の収集を一時的に停止する」参照）。 新たに取得したトレース・データの内容に表示を更新します。 ただし，トレーサ停止中は無効となります。
検索 ...	トレース検索 ダイアログをオープンします。
コピー	選択している行の内容を文字列としてクリップ・ボードにコピーします（複数行選択不可）。 ただし，トレーサ動作中は無効となります。
混合表示	表示モードを混合表示モードにします。 ただし，トレーサ動作中は無効となります。
逆アセンブル表示	表示モードを逆アセンブル表示モードにします。 ただし，トレーサ動作中は無効となります。
ソース表示	表示モードをソース表示モードにします。 ただし，トレーサ動作中は無効となります。
表記	表示進数を指定するために，次のカスケード・メニューを表示します。 ただし，トレーサ動作中は無効となります。
16 進数	このパネル上の値を 16 進数で表示します（デフォルト）。
10 進数	このパネル上の値を 10 進数で表示します。
8 進数	このパネル上の値を 8 進数で表示します。
2 進数	このパネル上の値を 2 進数で表示します。
ウインドウ連動	他のパネルとの連動を行うために，次のカスケード・メニューを表示します。
ソースと連動	キャレット位置の行に連動してエディタ パネルをスクロールします。
逆アセンブルと連動	キャレット位置の行に連動して逆アセンブル パネルをスクロールします。
逆アセンブルへジャンプ	キャレット位置の行のフェッチ・アドレスにキャレットを移動した状態で，逆アセンブル パネル（逆アセンブル 1）がオープンします。
ソースへジャンプ	キャレット位置の行に対応するソース行にキャレットを移動した状態で，エディタ パネルがオープンします。
メモリへジャンプ	キャレット位置の行に対応するメモリ値にキャレットを移動した状態で，メモリ パネルがオープンします。

イベント パネル

エディタ パネル／逆アセンブル パネル／ウォッチ パネル上で設定したイベントの詳細情報の表示，設定状態の有効／無効の切り替え，および削除などを行います（「2.16 イベントの管理」参照）。
なお，このパネルは，デバッグ・ツールと接続時のみオープンすることができます。

備考 1. イベントの設定に関しては，「2.16.6 イベント設定に関する留意事項」を参照してください。

備考 2. 解析ツールの関数 パネル／変数 パネルで設定したイベントもこのパネルで管理します。

備考 3. ツールバーの 100% ，または［Ctrl］キーを押下しながらマウス・ホイールを前後方向に動かすことにより，本パネルの表示を拡大／縮小することができます。

備考 4. パネル上の各エリアの区切り線をダブルクリックすることにより，該当エリアの内容を省略することなく表示可能な最小幅に変更することができます。

図 A.28 イベント パネル

ここでは，次の項目について説明します。

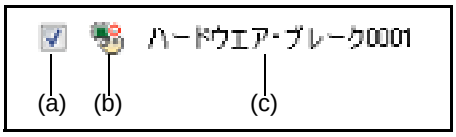
- ［オープン方法］
- ［各エリアの説明］
- ［ツールバー］
- ［［編集］メニュー（イベント パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］メニュー→［イベント］を選択
- 【シミュレータ】
  エディタ パネル／逆アセンブル パネルにおいて，コンテキスト・メニューの［タイマ設定］→［タイマ結果の表示］を選択

［各エリアの説明］

(1) ［名前］エリア
現在設定されているイベント名を次の形式で一覧表示します。



備考　ツールバーのボタンの選択により，表示するイベント種別を限定することができます（「[ツールバー]」参照）。

(a) チェック・ボックス
イベントの設定状態を表示／変更します。
なお，イベントの設定状態を変更すると，対応してイベント・マークも変化します。

☑	有効状態	指定されている条件の成立で，対象となるイベントが発生します。 チェックを外すことにより，イベントを無効状態にすることができます。
☐	無効状態	指定されている条件が成立しても，対象となるイベントは発生しません。 チェックすることにより，イベントを有効状態にすることができます。
■	保留状態	指定されている条件が，デバッグ対象のプログラムでは設定することができません。 チェック・ボックスを操作することはできません。

備考 1.　タイマ計測イベントを有効状態にするためには，タイマ開始イベントとタイマ終了イベントの両方の設定が必要となります。

備考 2.　Run-Break タイマ・イベントを無効状態／保留状態にすることはできません。

備考 3.　無条件トレース・イベントとトレース・イベントにおける有効／無効状態の設定は，排他制御となります。このため，ビルトイン・イベントである無条件トレース・イベントは，デフォルトで有効状態で設定されていますが，トレース開始イベント／トレース終了イベントのいずれかが設定されると同時に自動的に無効状態に変更され，トレース・イベント（トレース開始イベント／トレース終了イベントを 1 つにまとめたイベント）が有効状態になります。
また逆に，設定されているトレース・イベントを無効状態にすると，自動的に無条件トレース・イベントが有効状態となります。

(b) イベント・マーク
イベント・マークは，イベントの種別を示すとともに，現在の設定状態を示します。
表示されるイベント・マークとその意味は次のとおりです。

表 A.8　イベント・マーク

イベント種別	有効状態	無効状態	保留状態	備考
ハードウエア・ブレーク				ハードウエア・ブレーク・ポイントを含む
ソフトウエア・ブレーク				ソフトウエア・ブレーク・ポイントを含む
関数の先頭へのブレーク				解析ツールにより設定可能なブレーク・イベント
変数のアクセス・ブレーク				
無条件トレース			–	–
Run-Break タイマ		–	–	–
トレース				イベント パネルでのみ表示
トレース開始				エディタ パネル／逆アセンブル パネルでのみ表示
トレース終了				
タイマ計測				イベント パネルでのみ表示
タイマ開始				エディタ パネル／逆アセンブル パネルでのみ表示
タイマ終了				
ポイント・トレース				–

イベント種別	有効状態	無効状態	保留状態	備考
Printf イベント				−
上記イベントの複数設定	 注 1	 注 2	 注 3	エディタ パネル／逆アセンブル パネルでのみ表示

注 1. 複数のイベントの中で，1 つでも有効状態のイベントがある場合

注 2. 複数のイベントの中で，有効状態のイベントがなく，1 つでも無効状態のイベントがある場合

注 3. 複数のイベントのすべてが保留状態の場合

(c) イベント名

イベント名として，イベント種別と ID 番号を表示します。
ID 番号は，イベント種別ごとに 0001 からの番号が自動的に付与されます（一度設定したイベントを削除した場合でも ID 番号の振り直しは行いません）。
表示されるイベント種別は次のとおりです。

表 A.9 イベント種別

イベント種別	説明
ハードウエア・ブレーク（ブレーク[注 1]）	デバッグ・ツールが，プログラム実行中にブレーク条件を逐次確認し，条件を満たした際にプログラムをブレークさせるイベントです。 →「2.9.3 任意の場所で停止する（ブレークポイント）」参照 →「2.9.4 任意の場所で停止する（ブレーク・イベント）」参照 →「2.9.5 変数 /I/O レジスタへのアクセスで停止する」参照
ソフトウエア・ブレーク（ブレーク[注 1]）	ブレークさせるアドレスの命令コードをブレーク用の命令に書き換え，その命令を実行した際にプログラムをブレークさせるイベントです。 →「2.9.3 任意の場所で停止する（ブレークポイント）」参照
関数の先頭へのブレーク	解析ツールの関数一覧パネルより設定されるハードウエア・ブレーク（実行系）です。
変数のアクセス・ブレーク	解析ツールの変数一覧パネルより設定されるハードウエア・ブレーク（アクセス系）です。
無条件トレース	プログラムの実行開始と同時に自動的にトレース・データを収集し，実行停止とともにトレース・データの収集を停止します。 このイベントは，ビルトイン・イベント[注 2] であるため，削除することはできません（デフォルトで有効状態で設定されています）。 →「2.12.2 実行停止までの実行履歴を収集する」参照
Run-Break タイマ	プログラムの実行開始と同時に自動的にプログラムの実行時間の計測を開始し，実行停止とともに実行時間の計測を終了します。このイベントは，ビルトイン・イベント[注 2] であるため，削除することはできません（デフォルトで有効状態で設定されています）。 →「2.13.1 実行停止までの実行時間を計測する」参照
トレース	トレース開始イベント，およびトレース終了イベントにより設定された条件を満たした際に，トレース・データの収集を開始／終了するイベントです（トレース開始イベント／トレース終了イベントのいずれかが設定されると表示されます）。 →「2.12.3 任意区間の実行履歴を収集する」参照
タイマ計測 *n*	タイマ開始イベント，およびタイマ終了イベントにより設定された条件を満たした際に，プログラムの実行時間の計測を開始／終了するイベントです（タイマ開始イベント／タイマ終了イベントのいずれかが設定されると表示されます）。 なお，“*n*” は，設定したタイマのチャネル番号を示します。 →「2.13.2 任意区間の実行時間を計測する【シミュレータ】」参照

イベント種別	説明
ポイント・トレース	プログラムの実行により，指定した変数 /I/O レジスタにアクセスした際に，その情報をトレース・メモリに記録するイベントです。 →「2.12.4 条件を満たしたときのみの実行履歴を収集する【シミュレータ】」参照
Printf イベント	プログラムの実行を任意の箇所で一瞬停止させたのち，ソフトウエア処理により printf コマンドを実行させるイベントです（アクション・イベント）。 →「2.15.1 printf を挿入する」参照

注 1. マウスのワンクリック操作により設定されたブレークポイント（「2.9.3.1 ブレークポイントを設定する」参照）は，“ ブレーク ” と表示します。

注 2. デバッグ・ツールにデフォルトで設定されているイベントです。

(2) ［詳細情報］エリア

各イベントに関する詳細情報を表示します。
表示される情報の内容は，イベント種別によって異なります。
イベント種別ごとの詳細情報の見方は次のとおりです。

表 A.10 イベント種別ごとの詳細情報

イベント種別	表示内容[注 1]	
ハードウエア・ブレーク（発生条件：実行系）	表示形式 1	*＜発生条件＞ ＜ファイル名 # 行番号＞ ＜アドレス＞*
	表示例	実行前 main.c#39 0x100
		実行後 sub.c#100 0x200
		実行前 ― 0x300
		実行 main.c#39 0x300 【シミュレータ】
	表示形式 2	*＜発生条件＞ ＜シンボル + オフセット＞ ＜アドレス＞*
	表示例	実行前 funcA + 0x10 0x100
		実行後 funcB + 0x20 0x200
		実行前 ― 0x300
ハードウエア・ブレーク（発生条件：アクセス系）	表示形式 1	*＜発生条件＞ ＜ファイル名 # 変数名＞ ＜アドレス（範囲）＞ ＜比較条件＞ ＜比較値＞*
	表示例	リード main.c#variable10x100 - 0x101 == 0x5
		ライト sub.c#variable20x200 - 0x200 == 0x7
		リード／ライトsub2.c#variable30x300 - 0x303 == 0x8
	表示形式 2	*＜発生条件＞ ＜ファイル名 # 関数名 # 変数名＞ ＜アドレス（範囲）＞ ＜比較条件＞ ＜比較値＞*
	表示例	リード main.c#func1#variable10x100 - 0x101 == 0x10
	表示形式 3	*＜発生条件＞ ＜変数名＞ ＜アドレス（範囲）＞ ＜比較条件＞ ＜比較値＞*
	表示例	ライトvariable10x100 - 0x101 == 0x10

イベント種別	表示内容[注1]	
ソフトウエア・ブレーク	表示形式 1	*＜発生条件＞ ＜ファイル名 # 行番号＞ ＜アドレス＞*
	表示例	実行前　main.c#40　0x102
		実行前　sub.c#101　0x204
	表示形式 2	*＜発生条件＞ ＜シンボル + オフセット＞ ＜アドレス＞*
	表示例	実行前　funcA + 0x12　0x102
無条件トレース	表示形式	―
	表示例	―
Run-Break タイマ	表示形式	総実行時間 : *＜総実行時間＞*
	表示例	総実行時間 : 1000ms
		総実行時間 : OVERFLOW
トレース （発生条件：実行系）	表示形式	開始／終了の総数 : *＜トレース開始／トレース終了イベントの総数＞*[注2] *＜開始／終了＞ ＜トレース開始／トレース終了の詳細情報＞*
	表示例	開始／終了の総数 : 4 - 開始　実行後main.c#1000x300 - 開始　実行後funcA + 0x1000x300 - 終了　実行後main.c#2000x100 - 終了　実行後funcA + 0x100x100
タイマ計測 *n* （発生条件：実行系）	表示形式	総実行時間 : *＜総実行時間＞* 開始／終了の総数 : *＜タイマ開始／タイマ終了イベントの総数＞*[注2] - *＜総実行時間＞ ＜パスカウント＞ ＜平均実行時間＞ ＜最大行時間＞ ＜最小実行時間＞* - *＜開始／終了＞ ＜タイマ開始／タイマ終了の詳細情報＞*
	表示例	総実行時間 : 10ms 開始／終了の総数 : 4 - 総実行時間 : 10ms　パスカウント : 5　平均実行時間 : 2ms 最大実行時間 : 4ms　最小実行時間 : 1ms - 開始実行後　main.c#1000x300 - 開始実行後　funcA + 0x300x100 - 終了実行後　main.c#1000x300 - 終了実行後　funcA + 0x500x100
ポイント・トレース （発生条件：アクセス系）	表示形式 1	*＜発生条件＞ ＜変数名＞ ＜変数のアドレス＞*
	表示例	リード　variable1　0x100
	表示形式 2	*＜発生条件＞ ＜ファイル名 # 変数名＞ ＜変数のアドレス＞*
	表示例	ライト　sub.c#variable20x200
	表示形式 3	*＜発生条件＞ ＜ファイル名 #* 関数名 *# 変数名＞ ＜変数のアドレス＞*
	表示例	リード／ライトsub.c#func1#variabl3　0x300

イベント種別	表示内容[注1]	
Printf イベント （アクション・イベント）	表示形式	*＜発生条件＞ ＜ファイル名 # 行番号＞ ＜アドレス＞ ＜Print イベントの設定＞*
	表示例	実行前main.c#39 0x100aaa, bbb, ccc
		実行後sub.c#1000x200aaa の結果の表示 : aaa

注 1. 表示形式の詳細は次のとおりです。

＜発生条件＞	次の条件のいずれか 1 つを表示します。 【Full-spec emulator】【E1】【E20】 実行系 : 実行前，実行後 アクセス系: リード，ライト，リード／ライト 【シミュレータ】 実行系 : 実行 アクセス系: リード，ライト，リード／ライト
＜ファイル名 # 行番号＞	ソース・ファイル名とソース・ファイル中の行番号を表示します。表示形式はウォッチ式のスコープ指定式と同等です。 複数のロード・モジュール・ファイルをダウンロードしている場合では，*＜ロード・モジュール名＄ファイル名 # 行番号＞*を表示します。 なお，逆アセンブル パネルで設定されたイベントでは，次の場合，*行番号をシンボル + オフセット*形式で表示します。 - 行情報があり，指定されたイベント設定位置が行情報の先頭でない場合 - 行情報がなく，シンボル情報がある場合 また，次の場合は，*行番号*を “ー ” で表示します。 - 行情報がなく，シンボル情報がない場合
＜変数名＞	ソース・ファイル中の変数名を表示します。表示形式はウォッチ式のスコープ指定式と同等です。
＜比較条件＞	比較の条件（==）を表示します。比較値が指定されなかった場合は表示しません。
＜比較値＞	比較値を表示します。比較値が指定されなかった場合は表示しません。
＜アドレス＞	指定された変数の，メモリ領域中の開始アドレス - 終了アドレスを表示します（16 進数表記固定）。
＜開始／終了＞	詳細情報の内容が，開始イベントか終了イベントかを表示します。
＜パスカウント＞	タイマのパスカウントを表示します。 なお，タイマ・オーバフロー発生時（「2.13.3 測定可能時間の範囲」参照），または不正な値の場合は “OVERFLOW” を表示します。 また，未計測の場合は，“ 未計測 ” を表示します。
＜総実行時間＞	タイマの総実行時間の測定結果を表示します。 単位は，ns/μs/ms/s/min/clock のいずれか 1 つが表示されます（ただし，“min” の場合は "s" も同時に表示）。 なお，タイマ・オーバフロー発生時（「2.13.3 測定可能時間の範囲」参照），または不正な値の場合は “OVERFLOW” を表示します。 また，未計測の場合は，“ 未計測 ” を表示します。

＜*平均実行時間*＞	タイマの平均実行時間の測定結果を表示します。 単位は，ns/μs/ms/s/min/clock のいずれか 1 つが表示されます（ただし，“min” の場合は "s" も同時に表示）。 なお，タイマ・オーバフロー発生時（「2.13.3 測定可能時間の範囲」参照），または不正な値の場合は “OVERFLOW” を表示します。 また，未計測の場合は，“ 未計測 ” を表示します。
＜*最大実行時間*＞	タイマの最大実行時間の測定結果を表示します。 単位は，ns/μs/ms/s/min/clock のいずれか 1 つが表示されます（ただし，“min” の場合は "s" も同時に表示）。 なお，タイマ・オーバフロー発生時（「2.13.3 測定可能時間の範囲」参照），または不正な値の場合は “OVERFLOW” を表示します。 また，未計測の場合は，“ 未計測 ” を表示します。
＜*最小実行時間*＞	タイマの最小実行時間の測定結果を表示します。 単位は，ns/μs/ms/s/min/clock のいずれか 1 つが表示されます（ただし，“min” の場合は "s" も同時に表示）。 なお，タイマ・オーバフロー発生時（「2.13.3 測定可能時間の範囲」参照），または不正な値の場合は “OVERFLOW” を表示します。 また，未計測の場合は，“ 未計測 ” を表示します。
＜*Print イベントの設定*＞	アクション・イベント ダイアログ上で指定した，*出力文字列* : *変数式*を表示します。

注 2. この行をクリックすることにより，下行の詳細情報を表示します。

(3) ［コメント］エリア

設定されている各イベントに対して，ユーザが自由にコメントを入力できるエリアです。
コメントの入力は，コメントを入力したいイベントを選択後，このエリアをクリックするか，またはコンテキスト・メニューの［コメントの編集］を選択したのち，任意のテキストをキーボードから直接入力します（[Esc] キーの押下で編集モードをキャンセルします）。
コメントを編集したのち，［Enter］キーの押下，または編集領域以外へのフォーカスの移動により，編集を完了します。
なお，コメントは最大 256 文字まで入力することができ，使用中のユーザの設定として保存されます。

［ツールバー］

	選択しているイベント，およびイベント条件を削除します。 ただし，ビルトイン・イベント（無条件トレース・イベント /Run-Break タイマ・イベント）を削除することはできません。
	ハードウエア・ブレーク関連のイベントを表示します（デフォルト）。
 【Full-spec emulator】【E1】【E20】	ソフトウエア・ブレーク関連のイベントを表示します（デフォルト）。
	トレース関連のイベントを表示します（デフォルト）。
	タイマ関連のイベントを表示します（デフォルト）。
	アクション・イベント関連（Printf イベント）を表示します（デフォルト）。
	ビルトイン・イベント関連（無条件トレース・イベント /Run-Break タイマ・イベント）を表示します（デフォルト）。
	選択しているイベント[注]が設定されているアドレスに対応するソース行にキャレットを移動した状態で，エディタ パネルがオープンします。

	選択しているイベント注が設定されているアドレスに対応する逆アセンブル結果にキャレットを移動した状態で，逆アセンブル パネル（逆アセンブル 1）がオープンします。
	選択しているイベント注が設定されているアドレスに対応するメモリ値にキャレットを移動した状態で，メモリ パネル（メモリ 1）がオープンします。

注 トレース・イベント／タイマ計測イベント／ビルトイン・イベント（無条件トレース・イベント /Run-Break タイマ・イベント）以外のイベントが対象となります。

[[編集] メニュー（イベント パネル専用部分)]

イベント パネル専用の［編集］メニューは次のとおりです（その他の項目はすべて無効)。

削除	選択しているイベント，およびイベント条件を削除します。 ただし，ビルトイン・イベント（無条件トレース・イベント /Run-Break タイマ・イベント）を削除することはできません。
すべて選択	このパネルに表示されているすべてのイベントを選択状態にします。
検索 ...	検索・置換 ダイアログを［一括検索］タブが選択状態でオープンします。
置換 ...	検索・置換 ダイアログを［一括置換］タブが選択状態でオープンします。

[コンテキスト・メニュー]

有効化	選択しているイベントを有効状態にします。 ただし，選択しているイベントがすでに有効状態の場合は無効となります。
無効化	選択しているイベントを無効状態にします。 ただし，選択しているイベントがすでに無効状態の場合は無効となります。
削除	選択しているイベントを削除します。 ただし，ビルトイン・イベント（無条件トレース・イベント /Run-Break タイマ・イベント）を削除することはできません。
すべて選択	現在表示しているすべてのイベントを選択状態にします。
表示選択	表示するイベント種別を限定するために，次のカスケード・メニューを表示します。 デフォルトでは，すべての項目が選択されています。
ハードウエア・ブレーク	ハードウエア・ブレーク関連のイベントを表示します。
ソフトウエア・ブレーク	ソフトウエア・ブレーク関連のイベントを表示します。
タイマ	タイマ関連のイベントを表示します。
トレース	トレース関連のイベントを表示します。
アクション・イベント	アクション・イベント（Printf イベント）を表示します。
ビルトイン・イベント	ビルトイン・イベント（無条件トレース・イベント /Run-Break タイマ）を表示します。
タイマ設定	タイマ関連の設定をするために，次のカスケード・メニューを表示します。 ただし，タイマ関連のイベントを選択している場合のみ有効です。

タイマの初期化	選択しているイベント（Run-Break タイマ・イベントを除く）で使用するタイマを初期化します。
ナノ秒表示	選択しているイベントのタイマ結果をナノ秒（ns）単位で表示します。
マイクロ秒表示	選択しているイベントのタイマ結果をマイクロ秒（μs）単位で表示します。
ミリ秒表示	選択しているイベントのタイマ結果をミリ秒（ms）単位で表示します。
秒表示	選択しているイベントのタイマ結果を秒（s）単位で表示します。
分表示	選択しているイベントのタイマ結果を分（min）単位で表示します。
クロック表示	選択しているイベントのタイマ結果をクロック（clock）単位で表示します。
メモリへジャンプ	選択しているイベント[注]が設定されているアドレスに対応するメモリ値にキャレットを移動した状態で，メモリ パネル（メモリ 1）がオープンします。
逆アセンブルへジャンプ	選択しているイベント[注]が設定されているアドレスに対応する逆アセンブル結果にキャレットを移動した状態で，逆アセンブル パネル（逆アセンブル 1）がオープンします。
ソースへジャンプ	選択しているイベント[注]が設定されているアドレスに対応するソース行にキャレットを移動した状態で，エディタ パネルがオープンします。
条件の編集 ...	選択しているアクション・イベント（Printf イベント）を編集するために アクション・イベント ダイアログをオープンします。
コメントの編集	選択しているイベントのコメントを編集モードにします。 すでにコメントが存在する場合は，その文字列のすべてを選択状態にします。

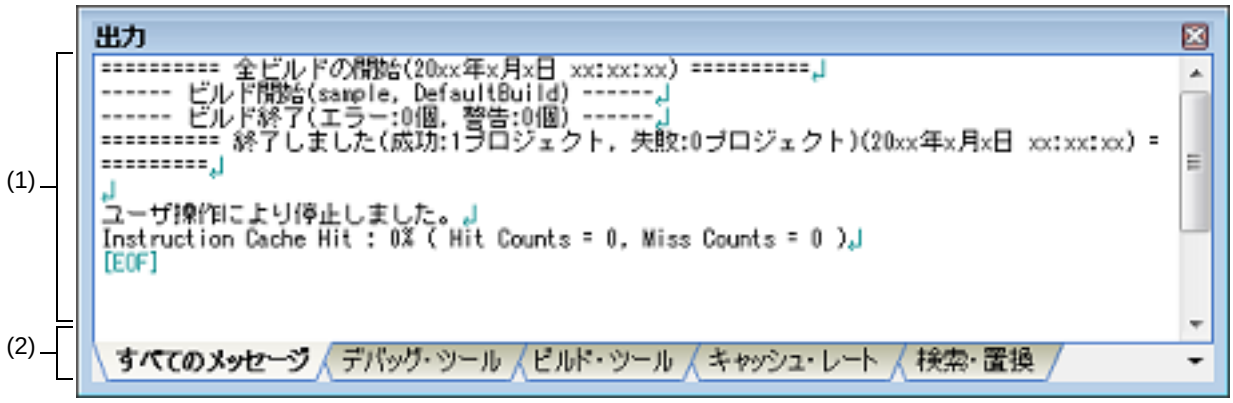
注 トレース・イベント／タイマ計測イベント／ビルトイン・イベント（無条件トレース・イベント /Run-Break タイマ・イベント）以外のイベントが対象となります。

出力 パネル

CS+ が提供している各種コンポーネント（デバッグ・ツールを含む，設計ツール／ビルド・ツールなど）から出力されるメッセージの表示，または検索・置換 ダイアログによる一括検索を行った際の結果，および Printf イベント（「2.15.1 printf を挿入する」参照）による出力結果の表示を行います。
メッセージは，出力元のツールごとに分類されたタブ上でそれぞれ個別に表示されます。

備考 ツールバーの 100% ，または［Ctrl］キーを押下しながらマウス・ホイールを前後方に動かすことにより，本パネルの表示を拡大／縮小することができます。

図 A.29 出力 パネル



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［［ファイル］メニュー（出力 パネル専用部分）］
- ［［編集］メニュー（出力 パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- ［表示］メニュー→［出力］を選択

［各エリアの説明］

(1) メッセージ・エリア
各ツールから出力されたメッセージ，検索結果，および Printf イベントによる出力結果を表示します。
ビルド結果／検索結果（一括検索）の表示では，ビルド／検索を行うごとに以前のメッセージをクリアしたのち，新しいメッセージを表示します（［すべてのメッセージ］タブを除く）。
なお，メッセージの表示色は，出力メッセージの種別により，次のように異なります（文字色／背景色はオプション ダイアログにおける［全般 - フォントと色］カテゴリの設定に依存）。

メッセージ種別	表示例（デフォルト）			説明
通常メッセージ	AaBbCc	文字色	黒	何らかの情報を通知する際に表示されます。
		背景色	白	
警告メッセージ	AaBbCc	文字色	青	操作に対して，何らかの警告を通知する際に表示されます。
		背景色	標準色	
エラー・メッセージ	AaBbCc	文字色	赤	致命的なエラー，または操作ミスにより実行が不可能な場合に表示されます。
		背景色	薄グレー	

このエリアは，次の機能を備えています。

(a) タグ・ジャンプ
出力されたメッセージをダブルクリック，またはメッセージにキャレットを移動したのち，［Enter］キーを押下することにより，エディタ パネルをオープンして該当ファイルの該当行番号を表示します。
これにより，ビルド時に出力されたエラー・メッセージなどから，ソース・ファイルの該当するエラー行へジャンプすることができます。

(b) ヘルプの表示
警告メッセージ，またはエラー・メッセージを表示している行にキャレットがある状態で，コンテキスト・メニューの［メッセージに関するヘルプ］を選択するか，または［F1］キーを押下することにより，その行のメッセージに関するヘルプを表示します。

(c) ログの保存
［ファイル］メニュー→［名前を付けて出力 - *タブ名*を保存 ...］を選択することにより，名前を付けて保存 ダイアログをオープンし，現在選択しているタブ上に表示されている全内容をテキスト・ファイル（*.txt）に保存することができます（非選択状態のタブ上のメッセージは保存の対象となりません）。

(2) タブ選択エリア
メッセージの出力元を示すタブを選択します。
デバッグ・ツールでは，次のタブを使用します。

タブ名	説明
すべてのメッセージ	CS+ が提供している全コンポーネント（デバッグ・ツールを含む，設計ツール／ビルド・ツールなど）から出力されるメッセージを表示します（ラピッド・ビルドの実行によるメッセージを除く）。
デバッグ・ツール	CS+ が提供している各種コンポーネント（デバッグ・ツールを含む，設計ツール／ビルド・ツールなど）から出力されるメッセージのうち，デバッグ・ツールが出力するメッセージを表示します。
キャッシュ・レート【シミュレータ】	ブレーク時に，キャッシュ・ヒット率（キャッシュへのアクセス回数に対するヒット回数の割合）を表示します。
ビルド・ツール	ビルド・ツールから出力されたメッセージを表示します。
検索・置換	検索・置換 ダイアログによる一括検索結果を表示します。

注意 新たなメッセージが非選択状態のタブ上に出力されても，自動的なタブの表示切り替えは行いません。この場合，タブ名の先頭に “*” マークが付加し，新たなメッセージが出力されていることを示します。

［［ファイル］メニュー（出力 パネル専用部分）］

出力 パネル専用の［ファイル］メニューは次のとおりです（その他の項目は共通）。
ただし，プログラム実行中はすべて無効となります。

出力 - *タブ名*を保存	現在選択しているタブ上に表示されている内容を，前回保存したテキスト・ファイル（*.txt）に保存します（「(c) ログの保存」参照）。 なお，起動後に初めてこの項目を選択した場合は，［名前を付けて*タブ名*を保存 ...］の選択と同等の動作となります。 ただし，ビルド実行中は無効となります。
名前を付けて出力 - *タブ名*を保存 ...	現在選択しているタブ上に表示されている内容を，指定したテキスト・ファイル（*.txt）に保存するために，名前を付けて保存 ダイアログをオープンします（「(c) ログの保存」参照）。

［［編集］メニュー（出力 パネル専用部分）］

出力 パネル専用の［編集］メニューは次のとおりです（その他の項目はすべて無効）。

コピー	選択している文字列をクリップ・ボードにコピーします。
すべて選択	現在選択しているタブ上に表示されているすべてのメッセージを選択状態にします。
検索 ...	検索・置換 ダイアログを［クイック検索］タブが選択状態でオープンします。
置換 ...	検索・置換 ダイアログを［一括置換］タブが選択状態でオープンします。

［コンテキスト・メニュー］

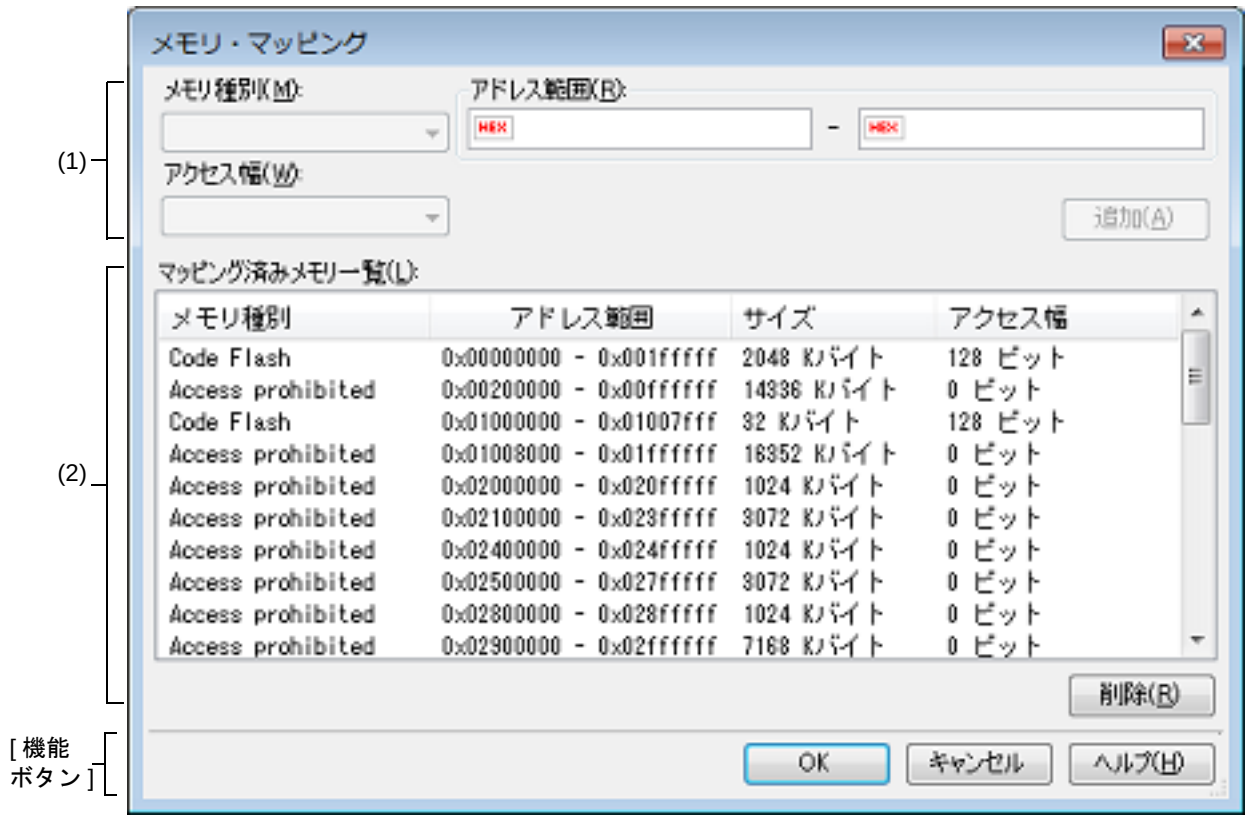
コピー	選択している文字列をクリップ・ボードにコピーします。
すべて選択	現在選択しているタブ上に表示されているすべてのメッセージを選択状態にします。
クリア	現在選択しているタブ上に表示されているすべてのメッセージを消去します。
タグ・ジャンプ	エディタ パネルをオープンし，キャレット位置のメッセージに該当するファイルの該当行番号にジャンプします。
検索の中止	現在実行中の検索を中止します。 ただし，検索を実行していない場合は無効となります。
メッセージに関するヘルプ	現在のキャレット位置のメッセージに関するヘルプを表示します。 ただし，警告メッセージ／エラー・メッセージのみが対象となります。

メモリ・マッピング ダイアログ

メモリ・マッピングの状況を表示します。

注意 選択したマイクロコントローラがマルチコア対応版の場合，コア（PE）の選択を切り替えることにより，PE ごとのメモリ・マッピングの状況を表示します（「2.7 コア（PE）の選択」参照）。

図 A.30 メモリ・マッピング ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロパティ パネルの［デバッグ・ツール設定］タブにおいて，［メモリ］カテゴリ内［メモリ・マッピング］プロパティを選択することにより表示される［...］ボタンをクリック

注意 プログラム実行中は，このダイアログをオープンすることはできません。

［各エリアの説明］

(1) 追加メモリ・マッピング指定エリア
このエリアは常に無効です。

(2) ［マッピング済みメモリー覧］エリア

(a) 一覧の表示
マイクロコントローラ内のメモリ・マッピング情報を表示します。
このエリアを編集することはできません。

メモリ種別	メモリ種別を表示します。
アドレス範囲	アドレス範囲を＜*開始アドレス*＞ - ＜*終了アドレス*＞で表示します。 “0x” を付与した 16 進数表示固定です。
サイズ	サイズを 10 進数で表示します（単位：バイト /K バイト[注]）。
アクセス幅	アクセス幅を表示します（単位：ビット）。

注 1024 の倍数の場合のみ，K バイト単位で表示します。

(b) ボタン

ボタン	機能
削除	このボタンは常に無効です。

［機能ボタン］

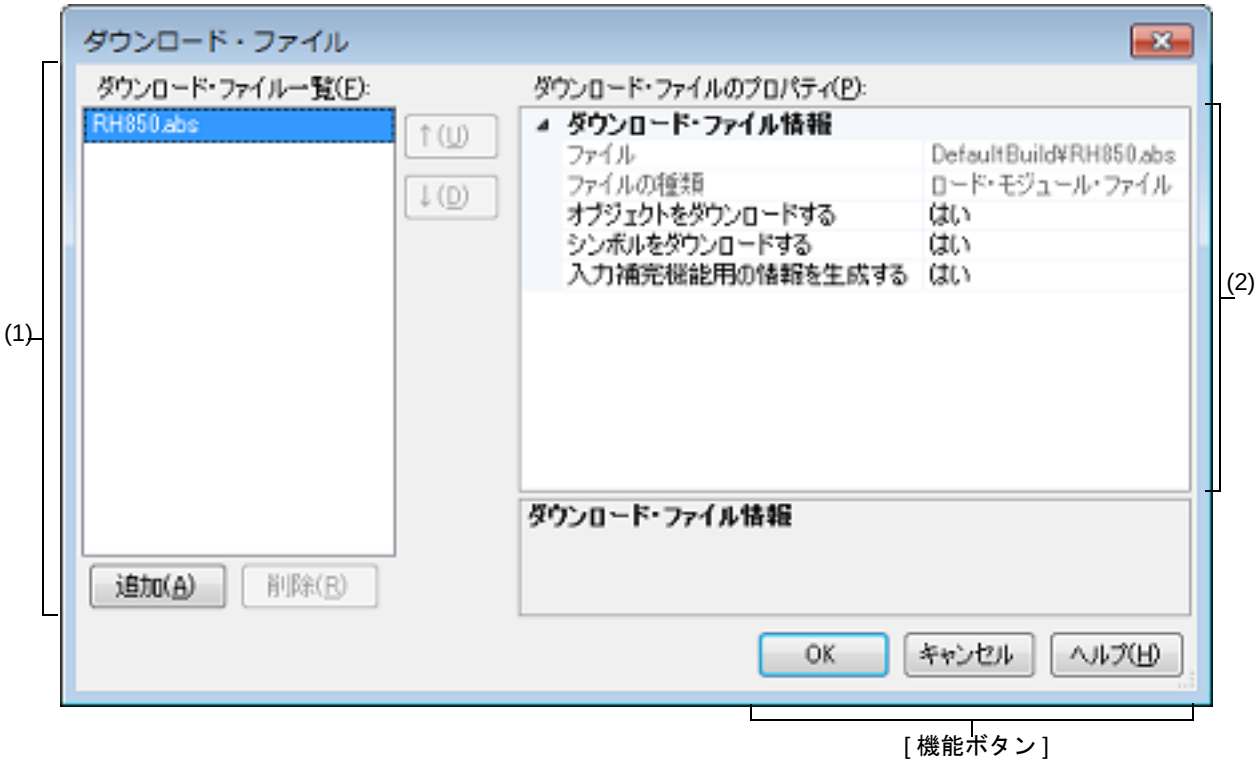
ボタン	機能
OK	このダイアログをクローズします。
キャンセル	このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

ダウンロード・ファイル ダイアログ

ダウンロードする際のファイルの選択，およびダウンロード条件の設定を行います（「2.5 ダウンロード／アップロード」参照）。
プロジェクト（メイン・プロジェクト／サブプロジェクト）でビルド対象に指定しているファイルは，自動的にダウンロードの対象ファイルとして登録されます（削除不可）。

注意 プログラム実行中は，このダイアログをオープンすることはできません。

図 A.31 ダウンロード・ファイル ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロパティ パネルの［ダウンロード・ファイル設定］タブにおいて，［ダウンロード］カテゴリ内［ダウンロードするファイル］プロパティを選択することにより表示される［...］ボタンをクリック

［各エリアの説明］

(1) ［ダウンロード・ファイル一覧］エリア

(a) 一覧の表示
ダウンロードするファイル名の一覧を表示します。デフォルトで，プロジェクト（メイン・プロジェクト／サブプロジェクト）においてビルド対象に指定しているファイル名を表示します（削除不可）。
ここでの表示順序が，ダウンロードの際の実行順序となります。
新規にダウンロード・ファイルを追加する場合は，このエリア内の［追加］ボタンをクリックし，［ダウンロード・ファイルのプロパティ］エリアにおいて，追加するファイルのダウンロード条件を指定します。

(b) ボタン

ボタン	機能
↑	選択しているファイルを 1 行上に移動します。 ただし，最上部のファイル，またはプロジェクトのビルド対象に指定しているファイルを選択している場合は無効となります。
↓	選択しているファイルを 1 行下に移動します。 ただし，最下部のファイル，またはプロジェクトのビルド対象に指定しているファイルを選択している場合は無効となります。
追加	一覧に空欄の項目（“-”）を 1 つ追加し，選択状態にします。 ［ダウンロード・ファイルのプロパティ］エリアにおいて，追加するファイルのダウンロード条件を指定してください。 ただし，すでに 20 個以上のファイルが登録されている場合は無効となります。
削除	選択しているファイルを一覧から削除します。 ただし，プロジェクトのビルド対象に指定しているファイルは削除することはできません。

備考 1. ファイル名にマウス・カーソルを合わせることにより，対象ファイルのパス情報をポップアップ表示します。

備考 2. ファイル名をマウスでドラッグすることにより，一覧内の表示順序を変更することができます。ただし，プロジェクトでビルド対象に指定しているファイルの表示順序を変更することはできません。

(2) ［ダウンロード・ファイルのプロパティ］エリア

(a) ［ダウンロード・ファイル情報］

［ダウンロード・ファイル一覧］エリアで選択しているファイルに対して，ダウンロード条件の表示／設定変更を行います。
また，［追加］ボタンにより，新規にダウンロード・ファイルを追加する場合は，ここで追加ファイルのダウンロード条件を指定します。

ファイル	ダウンロードするファイルを指定します。		
	デフォルト	*ファイル名*（ただし，新規追加の場合は空欄）	
	変更方法	キーボードからの直接入力，またはこの項目を選択すると欄内右端に表示される［...］ボタン[注 1]のクリックによりオープンするダウンロードするファイルを選択 ダイアログによる指定	
	指定可能値	「表 2.1 ダウンロード可能なファイル」参照 最大指定文字数：259 文字	
ファイルの種類	ダウンロードするファイルのファイル形式を選択します。		
	デフォルト	ロード・モジュール・ファイル	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	ロード・モジュール・ファイル	ロード・モジュール・フォーマット（*.abs）を指定します。
		ヘキサ・ファイル	インテル拡張ヘキサ・フォーマット（*.hex）を指定します。
		S レコード・ファイル	モトローラ・S タイプ・フォーマット（*.mot）を指定します。
		バイナリ・データ・ファイル	バイナリ・フォーマット（*.bin）を指定します。

オフセット	指定したファイルのダウンロードを開始するアドレスからのオフセット値を指定します。 なお，この項目は，［ファイルの種類］に［ヘキサ・ファイル］，または［S レコード・ファイル］を選択している場合のみ表示されます。		
	デフォルト	0	
	変更方法	キーボードからの直接入力	
	指定可能値	0x0 ～ 0xFFFFFFFF の 16 進数	
開始アドレス	指定したファイルをダウンロードする開始アドレスを指定します。 なお，この項目は，［ファイルの種類］に［バイナリ・データ・ファイル］を選択している場合のみ表示されます。		
	デフォルト	0	
	変更方法	キーボードからの直接入力	
	指定可能値	0x0 ～ 0xFFFFFFFF の 16 進数	
オブジェクトをダウンロードする	指定したファイルからオブジェクト情報をダウンロードするか否かを選択します。 なお，この項目は，［ファイルの種類］に［ロード・モジュール・ファイル］を選択している場合のみ表示されます。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	オブジェクト情報をダウンロードします。
		いいえ	オブジェクト情報をダウンロードしません。
シンボルをダウンロードする	指定したファイルからシンボル情報をダウンロードするか否かを選択します[注 2]。 なお，この項目は，［ファイルの種類］に［ロード・モジュール・ファイル］を選択している場合のみ表示されます。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	シンボル情報をダウンロードします。
		いいえ	シンボル情報をダウンロードしません。
入力補完機能用の情報を生成する	ダウンロード時に，シンボル名の入力補完機能のための情報を生成するか否かを選択します）[注 3]。 なお，この項目は，［ファイルの種類］に［ロード・モジュール・ファイル］を選択している場合のみ表示されます。		
	デフォルト	はい	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	はい	シンボル名の入力補完機能用の情報を生成します（入力補完機能を使用します）。
		いいえ	シンボル名の入力補完機能用の情報を生成しません（入力補完機能を使用しません）。

注 1. ［ダウンロード・ファイル一覧］エリアにおいて，プロジェクトのビルド対象のファイルを選択している場合，またはプログラム実行中は，［...］ボタンは表示されません。

注 2. シンボル情報をダウンロードしない場合，ソース・レベル・デバッグを行うことはできません。

注 3. ［はい］を選択した場合，ダウンロード時間，およびホスト・マシンのメモリ消費量が増加します。シンボル名の入力補完機能を使用しない場合は，［いいえ］を選択することを推奨します。

［機能ボタン］

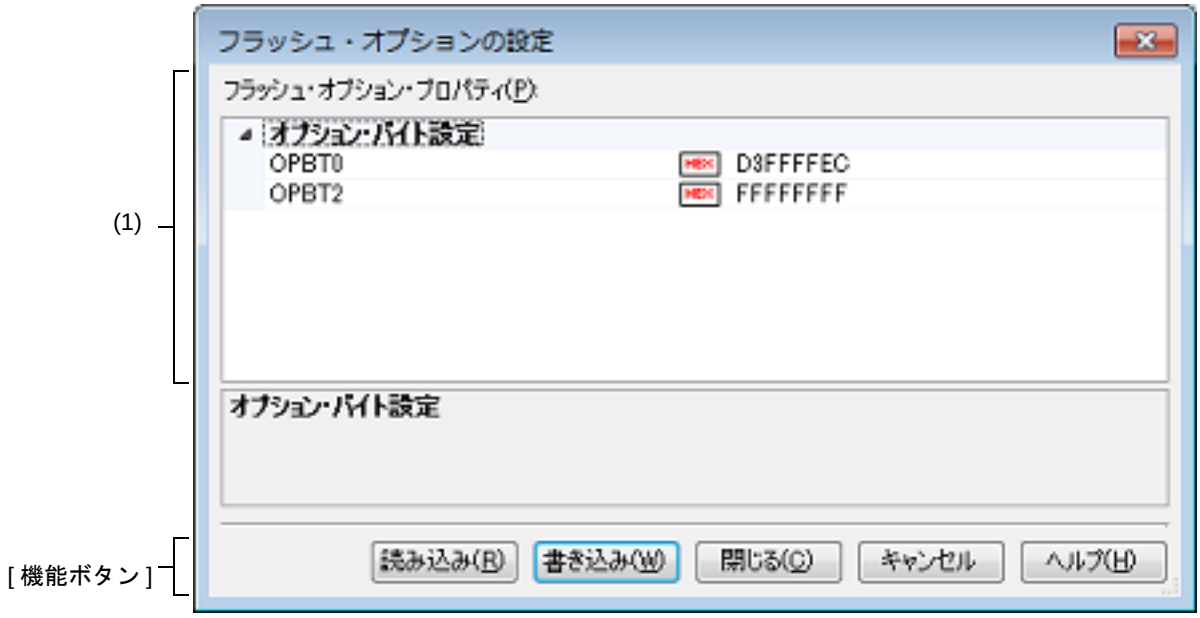
ボタン	機能
OK	ダウンロード・ファイルの設定を終了し，このダイアログをクローズします。
キャンセル	ダウンロード・ファイルの変更を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

フラッシュ・オプションの設定 ダイアログ【Full-spec emulator】【E1】【E20】

マイクロコントローラに搭載されているフラッシュ・メモリのためのオプション設定を行います。
なお，このダイアログは，デバッグ・ツールと接続時のみオープンすることができます。

注意 このダイアログで設定変更を行ったのち，［書き込み］ボタンをクリックすると，CPU リセットが発生します。

図 A.32 フラッシュ・オプションの設定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- プロパティ パネルの［フラッシュ・オプション設定］タブ【Full-spec emulator】【E1】【E20】において，［フラッシュ・オプション］カテゴリ内［フラッシュ・オプション］プロパティを選択することにより表示される［...］ボタンをクリック

［各エリアの説明］

(1) ［フラッシュ・オプション・プロパティ］エリア

(a) ［オプション・バイト設定］
フラッシュ・メモリのオプション・バイトの設定を行います。

OPBT*0* ～ *15*	オプション・バイトを指定します。	
	デフォルト	デバイス・ファイルに格納されている初期値
	変更方法	キーボードからの直接入力
	指定可能値	0x0 ～ 0xFFFFFFFF の範囲の 16 進数

注意 選択しているマイクロコントローラの種類により，表示されるオプション・バイト（OPB*0* ～ *15*）の個数は異なります（欠番を含む場合もあります）。

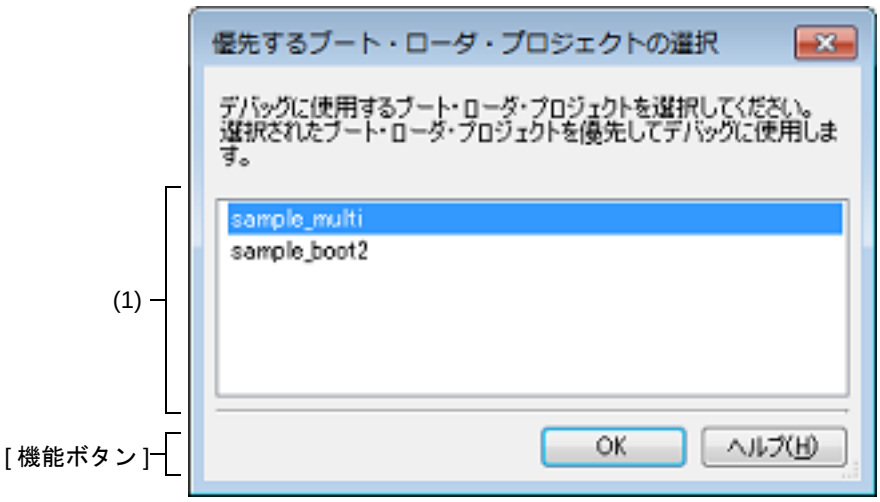
［機能ボタン］

ボタン	機能
読み込み	デバッグ・ツールに現在設定されている値を読み込み，このダイアログの各項目に反映します。
書き込み	現在の設定値をデバッグ・ツールに書き込み，プロジェクトに反映したのちこのダイアログをクローズします。
閉じる	現在の設定値をプロジェクトに反映し，このダイアログをクローズします。
キャンセル	設定の変更を反映せずに，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

優先するブート・ローダ・プロジェクトの選択 ダイアログ

コア単体のデバッグを行う際において，そのアプリケーション・プロジェクトが複数のブート・ローダ・プロジェクトの構成要素となっている場合，優先してデバッグ対象とするブート・ローダ・プロジェクトを選択します。

図 A.33 優先するブート・ローダ・プロジェクトの選択 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

各ブート・ローダ・プロジェクト（マルチコア設定ツール）のプロパティ パネルにおいて，［デバッグ］カテゴリ内［デバッグ時に優先する］プロパティがすべて［はい］，またはすべて［いいえ］に選択されている状態で次の操作を行った場合

- デバッグ・ツールとの通信開始（「2.4.1 デバッグ・ツールを接続する」参照）

［各エリアの説明］

(1) 優先するブート・ローダ・プロジェクト選択エリア
表示されるブート・ローダ・プロジェクトの一覧から，他のブート・ローダ・プロジェクトより優先してデバッグ対象とするブート・ローダ・プロジェクトを選択します。

［機能ボタン］

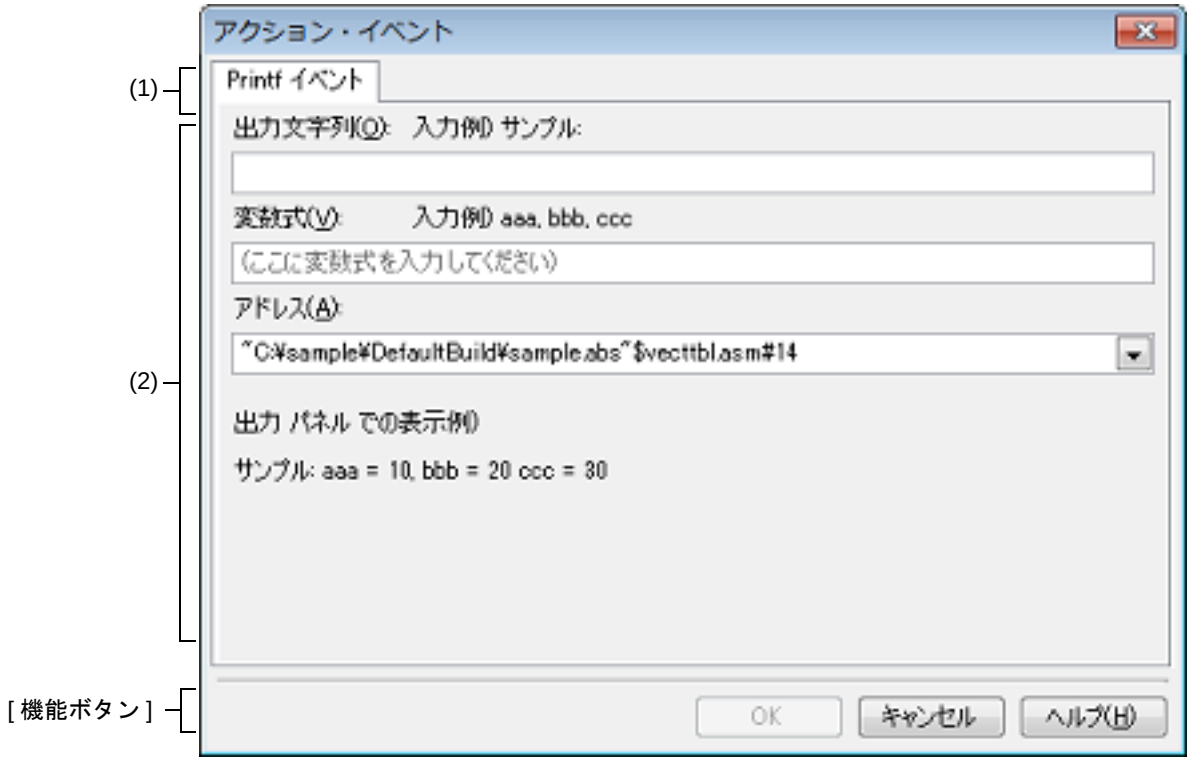
ボタン	機能
OK	選択したブート・ローダ・プロジェクトを優先してデバッグ対象とします。 また，選択したブート・ローダ・プロジェクトのみがデバッグ対象の優先プロジェクトとなるように，すべてのブート・ローダ・プロジェクトのプロパティ設定（［デバッグ］カテゴリ→［デバッグ時に優先する］プロパティ）を変更します。
ヘルプ	このダイアログのヘルプを表示します。

アクション・イベント ダイアログ

アクション・イベントの設定を行います（「2.15 プログラム内へのアクションの設定」参照）。
なお，このダイアログは，デバッグ・ツールと接続時のみオープンすることができます。

注意 アクション・イベントの設定に関しては（有効イベント数の制限など），（「2.16.6 イベント設定に関する留意事項」も参照してください。

図 A.34 アクション・イベント ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- エディタ パネルにおいて，アクション・イベントを設定したい行にキャレットを移動したのち，コンテキスト・メニュー→［アクション・イベントの登録 ...］を選択
- 逆アセンブル パネルにおいて，アクション・イベントを設定したいアドレスにキャレットを移動したのち，コンテキスト・メニュー→［アクション・イベントの登録 ...］を選択
- イベント パネルにおいて，アクション・イベントを選択したのち，コンテキスト・メニュー→［条件の編集 ...］を選択

［各エリアの説明］

(1) タブ選択エリア
タブを選択することにより，設定するアクション・イベントの種類が切り替わります。
このダイアログには，次のタブが存在します。

- ［Printf イベント］タブ

注意 コンテキスト・メニューの［条件の編集 ...］の選択によりこのダイアログをオープンした場合，このエリアは非表示となります。

(2) イベント条件設定エリア
アクション・イベントの詳細条件を設定します。
設定方法についての詳細は，該当するタブの項を参照してください。

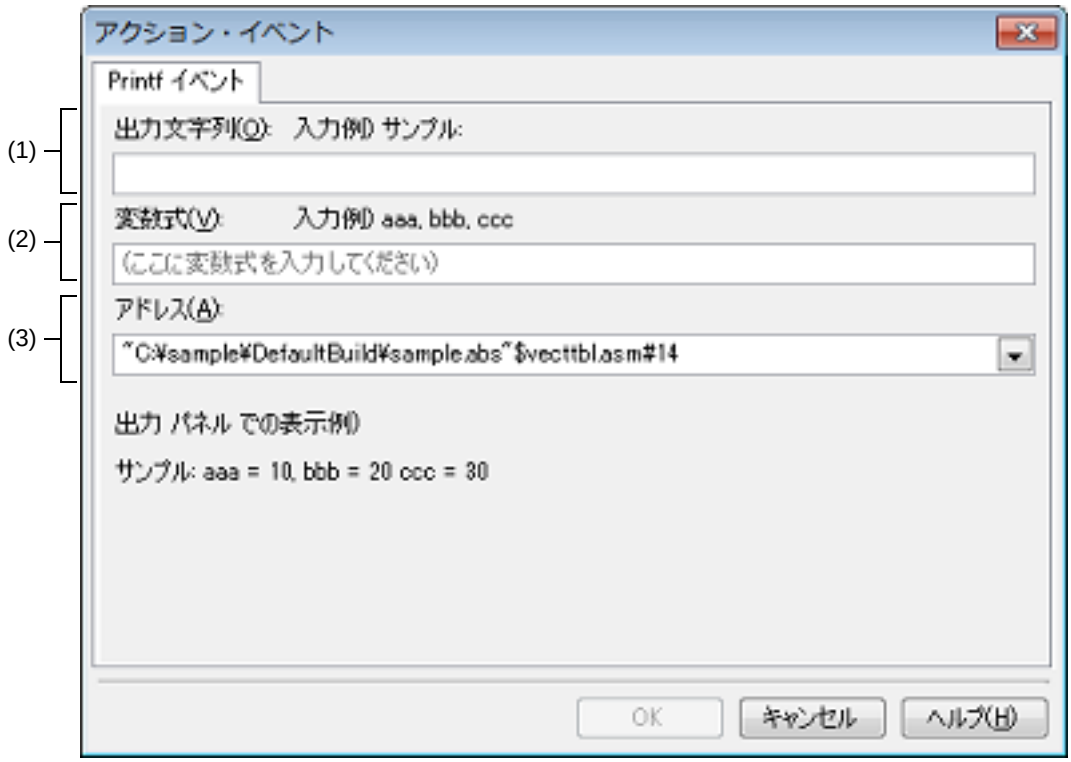
［機能ボタン］

ボタン	機能
OK	アクション・イベントの設定を終了し，指定したアクション・イベントを指定した位置に設定します。
キャンセル	アクション・イベントの設定を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

［Printf イベント］タブ

アクション・イベントとして，Printf イベントの設定を行います（「2.15 プログラム内へのアクションの設定」参照）。
Printf イベントとは，プログラムの実行を指定した箇所で一瞬停止させ，ソフトウエア処理によりコマンド（printf）を実行させる機能です。Printf イベントを設定すると，このイベントを設定した箇所の命令実行直前にプログラムが一瞬停止し，このダイアログで指定した変数式の値を出力 パネルに出力します。

図 A.35 アクション・イベント ダイアログ：［Printf イベント］タブ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］

［オープン方法］

- エディタ パネルにおいて，Printf イベントを設定したい行にキャレットを移動したのち，コンテキスト・メニュー→［アクション・イベントの登録 ...］を選択
- 逆アセンブル パネルにおいて，Printf イベントを設定したいアドレスにキャレットを移動したのち，コンテキスト・メニュー→［アクション・イベントの登録 ...］を選択
- イベント パネルにおいて，Printf イベントを選択したのち，コンテキスト・メニュー→［条件の編集 ...］を選択

［各エリアの説明］

(1) ［出力文字列］エリア
出力 パネルに出力する際に付与する文字列をキーボードより直接入力で指定します（最大指定文字数：1024 文字）。
なお，出力する文字列は，1 行分のみ入力可能です（空白可）。

(2) ［変数式］エリア
Printf イベントの対象となる変数式を指定します。
変数式は，テキスト・ボックスに直接入力で指定します（最大指定文字数：1024 文字）。
“,” で区切ることにより，1 つの Printf イベントとして 10 個までの変数式を指定することができます。

エディタ パネル／逆アセンブル パネルにおいて，変数式を選択した状態でこのダイアログをオープンした場合では，選択している変数式がデフォルトで表示されます。
なお，変数式として指定できる基本入力形式と，その際に Printf イベントとして出力される値は次のとおりです。

表 A.11 変数式と出力される値の関係（Printf イベント）

式	出力される値
C 言語変数名	C 言語の変数の値
変数式 [*変数式*]	配列の要素値
変数式 . メンバ名	構造体／共用体のメンバ値
変数式 -> メンバ名	ポインタの指し示す構造体／共用体のメンバ値
* *変数式*	ポインタの変数の値
& *変数式*	配置アドレス
CPU レジスタ名	CPU レジスタの値
I/O レジスタ名	I/O レジスタの値
ラベル名[注]，EQU シンボル名[注]，[即値]	ラベルの値，EQU シンボルの値，即値アドレスの値

注 ラベル名，または EQU シンボル名に “$” が含まれている場合，名前を “{}” で囲んでください（例：{$Label}）。
虚数の値には，大文字の “I” を掛けてください（例：1.0 + 2.0*I）。なお，“I” は虚数のキーワードとなるため，CPU レジスタの “I” を指定する場合は，“:REG” を付加してください（例：I:REG）。

備考 このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

(3) ［アドレス］エリア
Printf イベントを設定するアドレスを指定します。
テキスト・ボックスにアドレス式を直接入力するか（最大指定文字数：1024 文字），またはドロップダウン・リストにより入力履歴項目（最大履歴個数：10 個）を選択します。デフォルトで，現在の指定位置のアドレスを表示します。

備考 このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

なお，出力 パネル上における，Printf イベントによる出力結果のフォーマットは次のとおりです。

図 A.36 Printf イベントの出力結果フォーマット

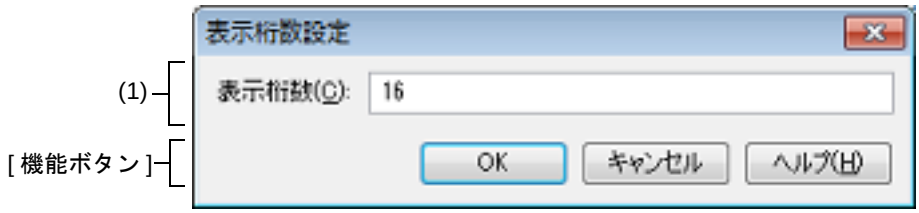
指定された文字列 変数式 1 = 値 1, 変数式 2 = 値 2, 変数式 3 = 値 3, ...

指定された文字列	［出力文字列］で指定した文字列
変数式 1 ～ 10	［変数式］で指定した文字列
値 1 ～ 10	“ *変数式 1 ～ 10*” に対する変数値 値は変数の型に応じた表示形式（「表 A.7 ウォッチ式の表示形式（デフォルト）」参照）で表示します（指定された変数式が取得不能の場合は “?” を表示）。 また，“()” 内に 16 進数値も併記します（表示不能の場合は “-” を表示）。

表示桁数設定 ダイアログ

メモリ パネルにおいて，メモリ値の表示桁数の設定を行います。

図 A.37 表示桁数設定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- メモリ パネルにおいて，コンテキスト・メニューの［表示桁数を設定 ...］を選択

［各エリアの説明］

(1) ［表示桁数］エリア
表示する桁数を 10 進数で直接入力により指定します。
指定可能な値の範囲は，現在のメモリ パネルにおける［サイズ表記］の設定により，次のように異なります。

サイズ表記	指定可能な範囲
4 ビット	2 ～ 512 注
1 バイト	1 ～ 256
2 バイト	1 ～ 128
4 バイト	1 ～ 64
8 バイト	1 ～ 32

注 偶数値でのみ指定できます（奇数値が入力された場合，1 つ大きな値に変更されます）。

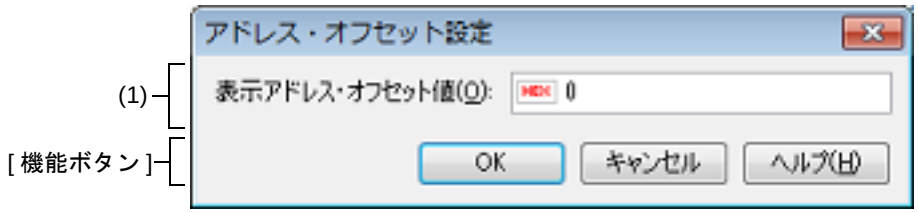
［機能ボタン］

ボタン	機能
OK	指定した桁数でメモリ値を表示します。
キャンセル	設定を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

アドレス・オフセット設定 ダイアログ

メモリ パネルのアドレス・エリアにおいて，開始アドレスのオフセット値を設定します。

図 A.38 アドレス・オフセット設定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- メモリ パネルにおいて，コンテキスト・メニューの［表示アドレス・オフセット値を設定 ...］を選択

［各エリアの説明］

(1) ［表示アドレス・オフセット値］エリア
アドレス表示のオフセット値を 16 進数で直接入力により指定します。
指定可能な値の範囲は，現在のメモリ パネルにおいて 1 行に表示されているメモリのバイト数により，次のように異なります。
指定可能な範囲：0x0 ～ (*［サイズ表記］の設定* × *表示桁数*) － 1

例 ［サイズ表記］：1 バイト／表示桁数：16 桁の場合

オフセット値	アドレス・エリアの表示内容
0x0（デフォルト）	0000 0010 0020
0x1	0001 0011 0021
0x2	0002 0012 0022

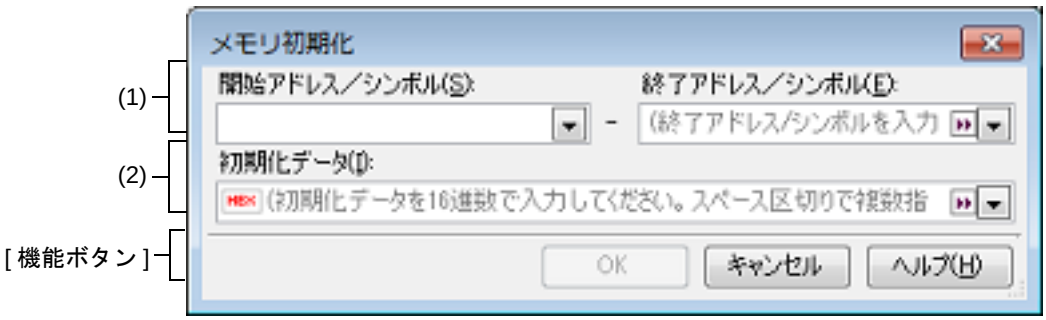
［機能ボタン］

ボタン	機能
OK	指定したオフセット値でメモリのアドレス表示を行います。
キャンセル	設定を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

メモリ初期化 ダイアログ

メモリ値の初期化を行います（「2.10.1.6 メモリの内容を一括して変更（初期化）する」参照）。
指定したアドレス範囲のメモリ領域に，指定した初期化データのパターンを繰り返し書き込みます。

図 A.39 メモリ初期化 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- メモリ パネルにおいて，コンテキスト・メニュー→［初期化 ...］を選択

［各エリアの説明］

(1) 範囲指定エリア
メモリ値を初期化するアドレス範囲を［開始アドレス／シンボル］と［終了アドレス／シンボル］に指定します。それぞれのテキスト・ボックスにアドレス式を直接入力するか（最大指定文字数：1024 文字），またはドロップダウン・リストにより入力履歴項目（最大履歴個数：10 個）を選択します。
入力したアドレス式の計算結果を，それぞれ開始アドレス／終了アドレスとして扱います。
なお，マイクロコントローラのアドレス空間よりも大きいアドレス値を指定することはできません。

注意 エンディアンの異なる領域をまたいだアドレス範囲を指定することはできません。

備考 このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

(2) ［初期化データ］エリア
メモリに書き込む初期化データを指定します。
初期化データの指定は，16 進数の数値をテキスト・ボックスに直接入力するか，またはドロップダウン・リストにより入力履歴項目（最大履歴個数：10 個）を選択することにより行います。
初期化データを複数指定する場合は，1 個 4 バイト（8 文字）までのデータを最大 16 個まで，半角スペースで区切り指定します。
個々の初期化データは，文字列終端より 2 文字単位で 1 バイトと解釈され，奇数文字数の場合は先頭 1 文字で 1 バイトと解釈されます。
なお，バイト数が 2 バイト以上の場合は，初期化対象のアドレス範囲のエンディアンのバイト列に変換してターゲット・メモリへの書き込み処理を行います。

入力文字列（初期化データ）	書き込みイメージ（バイト単位）	
	リトル・エンディアン	ビッグ・エンディアン
1	01	01
0 12	00 12	00 12
00 012 345	00 12 00 45 03	00 00 12 03 45

入力文字列 （初期化データ）	書き込みイメージ（バイト単位）	
	リトル・エンディアン	ビッグ・エンディアン
000 12 000345	00 00 12 45 03 00	00 00 12 00 03 45

［機能ボタン］

ボタン	機能
OK	指定したアドレス範囲のメモリ領域に，指定した初期化データのパターンを繰り返し書き込みます（パターンの途中で終了アドレスに達した場合は書き込みを終了します）。
キャンセル	メモリ値の初期化の設定を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

メモリ検索 ダイアログ

メモリ値の検索を行います（「2.10.1.5 メモリの内容を検索する」参照）。
このダイアログをオープンする直前にメモリ パネル上でキャレットが存在した，メモリ値エリア／文字列エリアのどちらかが検索の対象となります。

注意 メモリの検索は，プログラム空間のみが対象となります（ミラー領域は検索対象となりません）。

図 A.40 メモリ検索 ダイアログ

ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- メモリ パネルにおいて，コンテキスト・メニュー→［検索 ...］を選択

［各エリアの説明］

(1) ［検索するデータ］エリア
検索するデータを指定します。
テキスト・ボックスに直接入力するか（最大指定バイト数：256 バイト），またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
検索の対象がメモリ パネル上のメモリ値エリアの場合，そのエリアと同じ表示形式（表示進数／サイズ）でデータを入力する必要があります。
また，検索の対象が文字列エリアの場合では，検索するデータとして，文字列を指定する必要があります。指定した文字列は，そのエリアで表示しているエンコード形式でデータに変換され検索されます。
なお，このダイアログをオープンする直前にメモリ値を選択していた場合は，デフォルトでその値が表示されます。

(2) ［検索する範囲］エリア
検索する範囲を次のドロップダウン・リストより選択します。

アドレス範囲を指定する	［アドレス］エリアで指定したアドレス範囲内で検索を行います。
メモリ・マッピング	選択したメモリ・マッピング範囲内で検索を行います。 このリスト項目は，メモリ・マッピング ダイアログで表示しているメモリ・マッピングを個々に表示します。 表示形式：＜*メモリ種別*＞ ＜*アドレス範囲*＞ ＜*サイズ*＞

(3) ［アドレス］エリア
この項目は，［検索する範囲］エリアで［アドレス範囲を指定する］を選択した場合のみ有効となります。
メモリ値検索の対象となるアドレス範囲を“開始アドレス”と“終了アドレス”で指定します。それぞれのテキスト・ボックスにアドレス式を直接入力するか（最大指定文字数：1024 文字），またはドロップダウン・リストにより入力履歴項目（最大履歴個数：10 個）を選択します。
入力したアドレス式の計算結果を，それぞれ開始アドレス／終了アドレスとして扱います。

ただし，検索可能なアドレスの上限値は，プログラム空間の上限アドレス（0x03FFFFFF）です（ミラー領域は検索対象となりません）。
また，32 ビットで表現できる値より大きいアドレス値を指定することはできません。

備考 1. このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

備考 2. “ 開始アドレス ” が空欄の場合は，“0x0” の指定として扱われます。

備考 3. “ 終了アドレス ” が空欄の場合は，マイクロコントローラのアドレス空間の上限値の指定として扱われます。

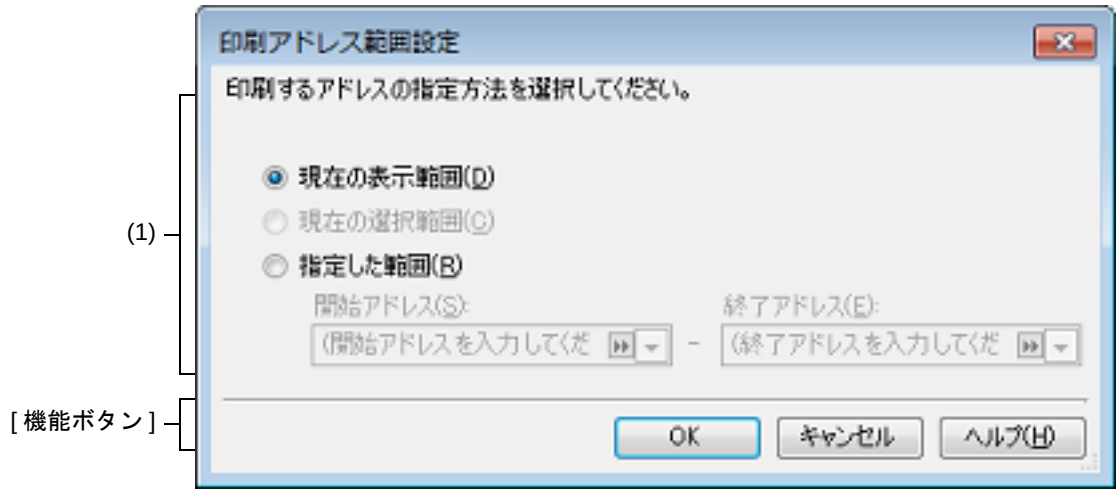
［機能ボタン］

ボタン	機能
前を検索	［検索する範囲］エリア／［アドレス］エリアで指定した範囲内で，アドレスの小さい方向に検索を行います。検索結果箇所をメモリ パネル上で選択状態にします。 ただし，不正な値を指定している場合，またはプログラム実行中は，メッセージを表示し，メモリ値の検索は行いません。 また，メモリ パネルが非表示の場合，または他のパネルにフォーカスがある状態からこのダイアログへフォーカスを移動した場合，このボタンは無効となります。
次を検索	［検索する範囲］エリア／［アドレス］エリアで指定した範囲内で，アドレスの大きい方向に検索を行います。検索結果箇所をメモリ パネル上で選択状態にします。 ただし，不正な値を指定している場合，またはプログラム実行中は，メッセージを表示し，メモリ値の検索は行いません。 また，メモリ パネルが非表示の場合，または他のパネルにフォーカスがある状態からこのダイアログへフォーカスを移動した場合，このボタンは無効となります。
キャンセル	メモリ値の検索の設定を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

印刷アドレス範囲設定 ダイアログ

逆アセンブル パネルの内容を印刷する際に，対象となるアドレス範囲の指定を行います。

図 A.41 印刷アドレス範囲設定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- 逆アセンブル パネルにおいて，［ファイル］メニュー→［印刷 ...］を選択

［各エリアの説明］

(1) 範囲指定エリア
印刷する範囲を指定するために，次のオプション・ボタンのいずれか 1 つを選択します。

(a) ［現在の表示範囲］（デフォルト）
逆アセンブル パネルで現在表示している範囲のみを印刷します。

(b) ［現在の選択範囲］
逆アセンブル パネルで現在選択している範囲のみを印刷します。
ただし，逆アセンブル パネルにおいて，何も選択していない場合は無効となります。

(c) ［指定した範囲］
印刷の対象となるアドレス範囲を［開始アドレス］と［終了アドレス］で指定します。
それぞれのテキスト・ボックスにアドレス式を直接入力するか（最大指定文字数：1024 文字），またはドロップダウン・リストにより入力履歴項目（最大履歴個数：10 個）を選択します。

備考 このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

［機能ボタン］

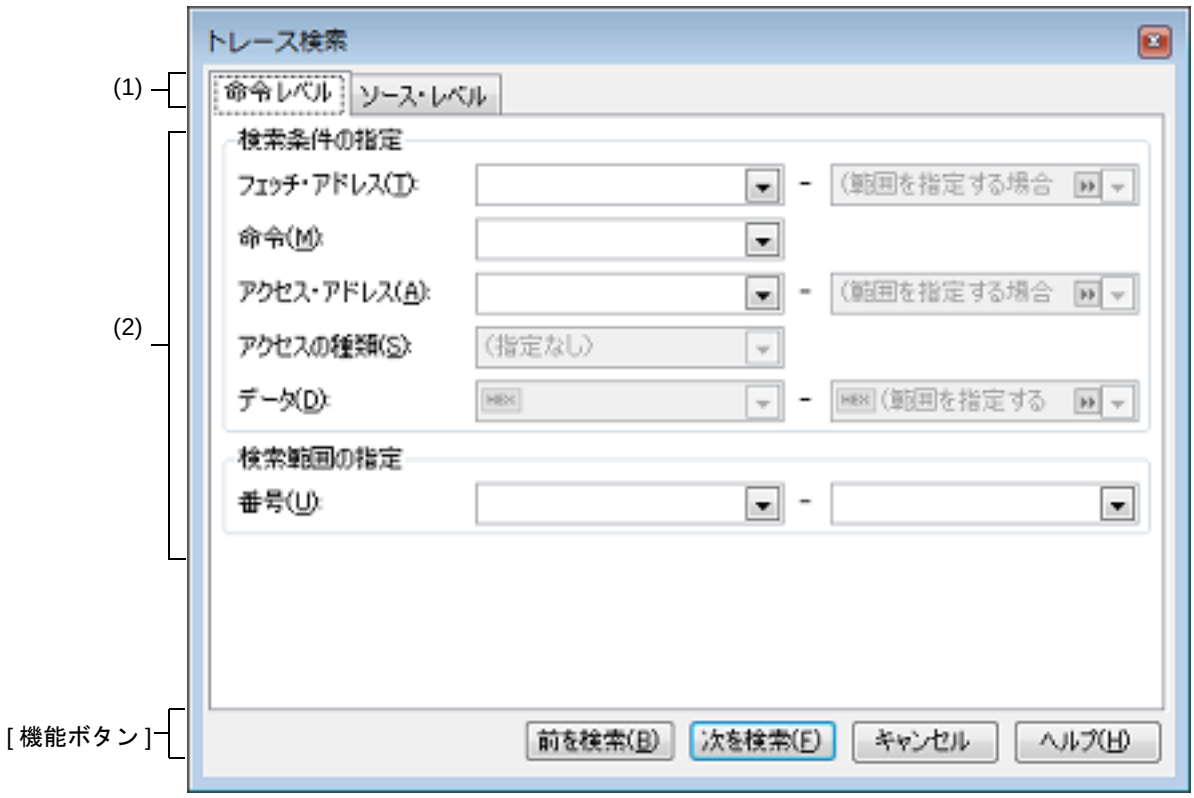
ボタン	機能
OK	指定した範囲で逆アセンブル パネルの内容を印刷するために，このダイアログをクローズして Windows の印刷用 ダイアログをオープンします。
キャンセル	範囲選択の設定を無視し，このダイアログをクローズします。

ボタン	機能
ヘルプ	このダイアログのヘルプを表示します。

トレース検索 ダイアログ

トレース・データの検索を行います（「2.12.8 トレース・データを検索する」参照）。
命令レベル／ソース・レベルを選択して検索することができます。

図 A.42 トレース検索 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- トレース パネルにおいて，ツールバーの ボタンを選択
- トレース パネルにおいて，コンテキスト・メニュー→［検索 ...］を選択

［各エリアの説明］

(1) タブ選択エリア
タブを選択することにより，検索するレベルが切り替わります。
このダイアログには，次のタブが存在します。

- ［命令レベル］タブ
- ［ソース・レベル］タブ

(2) 検索条件設定エリア
検索する際の詳細条件を設定します。
表示内容／設定方法についての詳細は，該当するタブの項を参照してください。

［機能ボタン］

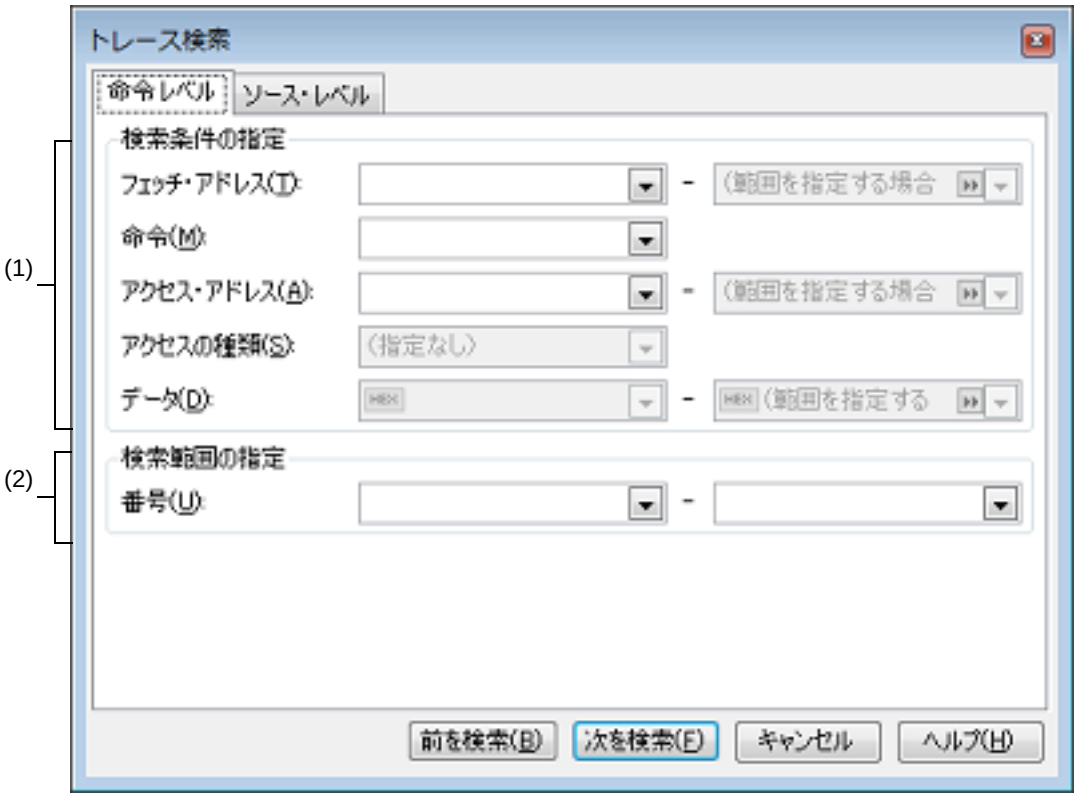
ボタン	機能
前を検索	指定した範囲内で，番号の小さい方向に検索を行います。 検索結果箇所をトレース パネル上で選択状態にします。 ただし，不正な値を指定している場合，またはプログラム実行中は，メッセージを表示し，トレース・データの検索は行いません。 また，トレース パネルが非表示の場合，または他のパネルにフォーカスがある状態からこのダイアログへフォーカスを移動した場合，このボタンは無効となります。
次を検索	指定した範囲内で，番号の大きい方向に検索を行います。 検索結果箇所をトレース パネル上で選択状態にします。 ただし，不正な値を指定している場合，またはプログラム実行中は，メッセージを表示し，トレース・データの検索は行いません。 また，トレース パネルが非表示の場合，または他のパネルにフォーカスがある状態からこのダイアログへフォーカスを移動した場合，このボタンは無効となります。
キャンセル	トレース・データの検索の設定を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

［命令レベル］タブ

取得したトレース・データを命令レベルで検索します。

注意 トレース パネルをソース表示モードで表示している場合，このタブで命令レベルの検索を行っても対象を正しく検索することはできません。
命令レベルの検索を行う際は，混合表示モード，または逆アセンブル表示モードで表示を行ってください。

図 A.43 トレース検索 ダイアログ：［命令レベル］タブ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］

［オープン方法］

- トレース パネルにおいて，ツールバーの  ボタンを選択
- トレース パネルにおいて，コンテキスト・メニュー→［検索 ...］を選択

［各エリアの説明］

(1) ［検索条件の指定］エリア

(a) ［フェッチ・アドレス］
検索条件として必要な場合，フェッチ・アドレスを指定します。
アドレス式をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
フェッチ・アドレスの指定は範囲で指定することができます。この場合は，左右両方のテキスト・ボックスにアドレス式を指定することにより範囲を指定します。
右側のテキスト・ボックスが空欄，または“（範囲を指定する場合に入力）”の場合は，左側のテキスト・ボックスに指定された固定アドレスで検索を行います。

なお，マイクロコントローラのアドレス空間よりも大きいアドレス値が指定された場合は，上位のアドレス値をマスクして扱います。
また，32 ビットで表現できる値より大きいアドレス値を指定することはできません。

(b) ［命令］
検索条件として必要な場合，命令の文字列を指定します。
ここで指定した文字列をトレース パネルの［ソース / 逆アセンブル］エリア内より検索します。
命令をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
なお，検索の際は，大文字／小文字は区別せず，部分一致も検索の対象とします。

(c) ［アクセス・アドレス］
検索条件として必要な場合，アクセス・アドレスを指定します。
アドレス式をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
アクセス・アドレスの指定は範囲で指定することができます。この場合は，左右両方のテキスト・ボックスにアドレス式を指定することにより範囲を指定します。
右側のテキスト・ボックスが空欄，または“（範囲を指定する場合に入力）”の場合は，左側のテキスト・ボックスに指定された固定アドレスで検索を行います。
なお，マイクロコントローラのアドレス空間よりも大きいアドレス値が指定された場合は，上位のアドレス値をマスクして扱います。
また，32 ビットで表現できる値より大きいアドレス値を指定することはできません。

(d) ［アクセスの種類］
この項目は［アクセス・アドレス］が指定された場合のみ有効となります。
アクセスの種類を次のドロップダウン・リストより選択します。

（指定なし）	アクセスの種類を限定しません。
リード／ライト	リード，またはライトした箇所を検索します。
リード	リードした箇所を検索します。
ライト	ライトした箇所を検索します。
ベクタ・リード	ベクタ・リードした箇所を検索します。
DMA	今版では無効です。

(e) ［データ］
この項目は［アクセス・アドレス］が指定された場合のみ有効となります。
アクセスした数値を指定します。
16 進数値をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
数値の指定は範囲で指定することができます。この場合は，左右両方のテキスト・ボックスにデータを指定することにより範囲を指定します。
右側のテキスト・ボックスが空欄，または“（範囲を指定する場合に入力）”の場合は，左側のテキスト・ボックスに指定された固定数値で検索を行います。

(2) ［検索範囲の指定］エリア

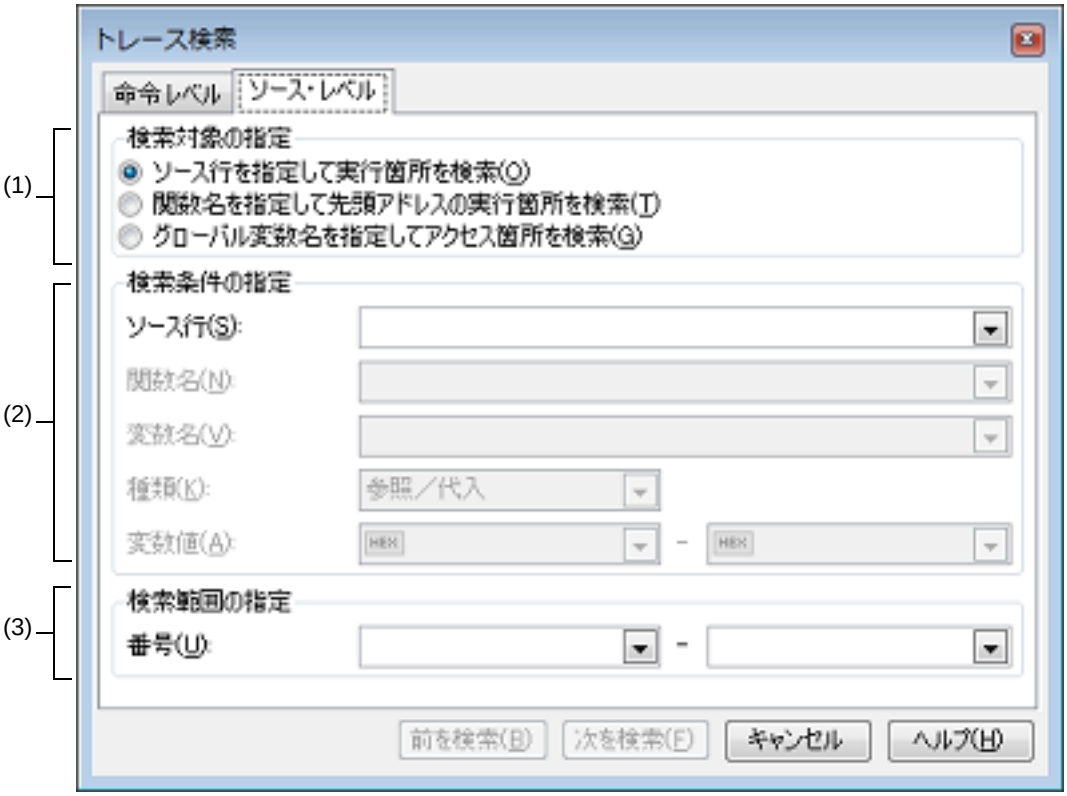
(a) ［番号］
検索するトレース・データの範囲を，トレース パネルの［番号］エリアに表示されている番号で指定します。
左右のテキスト・ボックスに，それぞれ開始番号と終了番号を指定します（デフォルトでは，“0”～“*最終番号*”が指定されます）。
10 進数で番号をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
左側のテキスト・ボックスが空欄の場合は，“0”の指定として扱われます。
右側のテキスト・ボックスが空欄の場合は，最終番号の指定として扱われます。

［ソース・レベル］タブ

取得したトレース・データをソース・レベルで検索します。

注意 トレース パネルを逆アセンブル表示モードで表示している場合，このタブでソース・レベルの検索を行っても対象を正しく検索することはできません。
ソース・レベルの検索を行う際は，混合表示モード，またはソース表示モードで表示を行ってください。

図 A.44 トレース検索 ダイアログ：［ソース・レベル］タブ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］

［オープン方法］

- トレース パネルにおいて，ツールバーの  ボタンを選択
- トレース パネルにおいて，コンテキスト・メニュー→［検索 ...］を選択

［各エリアの説明］

(1) ［検索対象の指定］エリア
検索する対象を次のオプション・ボタンの中から選択します。

ソース行を指定して実行箇所を検索	指定したソースの実行箇所を検索します（デフォルト）。 検索条件として［ソース行］の指定のみが有効となります。
関数名を指定して先頭アドレスの実行箇所を検索	指定した関数の実行箇所を検索します。 検索条件として［関数名］の指定のみが有効となります。

グローバル変数名を指定してアクセス箇所を検索	指定したグローバル変数をアクセスした箇所を検索します。 検索条件として［変数名］／［種類］／［変数値］の指定のみが有効となります。

(2) ［検索条件の指定］エリア

(a) ［ソース行］
この項目は “ ソース行を指定して実行箇所を検索 ” が選択された場合のみ有効となります。
ここで指定した文字列をトレース パネルの［行番号 / アドレス］エリア内より検索します。検索するソース行に含まれる文字列を，テキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
なお，検索の際は，大文字／小文字は区別せず，部分一致も検索の対象とします。

例 1. main.c#40

例 2. main.c

例 3. main

(b) ［関数名］
この項目は “ 関数名を指定して先頭アドレスの実行箇所を検索 ” が選択された場合のみ有効となります。
検索する関数名を，テキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
なお，検索の際は，大文字／小文字を区別し，完全一致のみを検索の対象とします。

(c) ［変数名］
この項目は “ グローバル変数名を指定してアクセス箇所を検索 ” が選択された場合のみ有効となります。
検索する変数名を，テキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
なお，検索の際は，大文字／小文字を区別し，完全一致のみを検索の対象とします。

(d) ［種類］
この項目は “ グローバル変数名を指定してアクセス箇所を検索 ” が選択された場合のみ有効となります。
アクセスの種類（参照／代入（デフォルト），参照，代入）をドロップダウン・リストより選択します。

(e) ［変数値］
この項目は “ グローバル変数名を指定してアクセス箇所を検索 ” が選択された場合のみ有効となります。
アクセスした変数値を 16 進数で指定します。
変数値をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
変数値の指定は範囲で指定することができます。この場合は，左右両方のテキスト・ボックスに変数値を指定することにより範囲を指定します。
右側のテキスト・ボックスが空欄の場合は，左側のテキスト・ボックスに指定された固定変数値でアクセス箇所を検索を行います。

(3) ［検索範囲の指定］エリア

(a) ［番号］
検索するトレース・データの範囲を，トレース パネルの［番号］エリアに表示されている番号で指定します。
左右のテキスト・ボックスに，それぞれ開始番号と終了番号を指定します（デフォルトでは，“0” ～ “ *最終番号* ” が指定されます）。
10 進数で番号をテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
左側のテキスト・ボックスが空欄の場合は，“0” の指定として扱われます。
右側のテキスト・ボックスが空欄の場合は，最終番号の指定として扱われます。

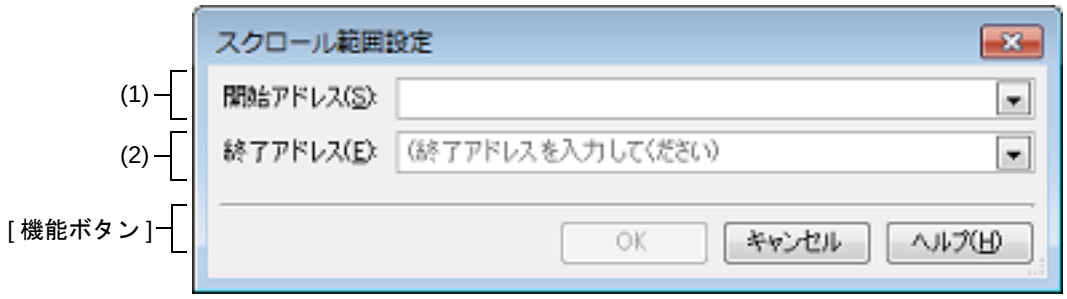
スクロール範囲設定 ダイアログ

メモリ パネル／逆アセンブル パネルの垂直スクロール・バーのスクロール範囲の設定を行います。
適正な範囲を設定することにより，パネルの垂直スクロール・バー上のスライダーの大きさが変化し，マウスによるドラッグなどの操作性が向上します。

注意 このダイアログによりスクロール範囲を設定したのち，ライン・アセンブルなどの実行により指定したアドレス式が表すアドレスに変更が生じても，スクロール範囲の修正は行いません。

備考 [Page Up] ／ [Page Down] ／ [↑] ／ [↓] キー，スクロール・バー端のボタン，またはジャンプ系のメニュー項目の選択による移動は，スクロール範囲外でも可能です。

図 A.45 スクロール範囲設定 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- メモリ パネルにおいて，ツールバーの［表示］→ボタンをクリック
- メモリ パネルにおいて，コンテキスト・メニューの［表示］→［スクロール範囲の設定 ...］を選択
- 逆アセンブル パネルにおいて，ツールバーの［表示］→ボタンをクリック
- 逆アセンブル パネルにおいて，コンテキスト・メニューの［表示］→［スクロール範囲の設定 ...］を選択

［各エリアの説明］

(1) ［開始アドレス］エリア
スクロールする範囲の開始アドレスを指定します。
アドレス式をテキスト・ボックスに直接入力するか（最大指定文字数：1024 文字），またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
なお，ドロップダウン・リスト内の“全範囲”を指定すると，スクロール範囲の設定は行いません（範囲は制限されません）。

備考 このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

(2) ［終了アドレス］エリア
スクロールする範囲の終了アドレスを指定します。
アドレス式をテキスト・ボックスに直接入力するか（最大指定文字数：1024 文字），またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
ただし，［開始アドレス］において，“全範囲”を指定している場合，このエリアは無効となります。
なお，ドロップダウン・リスト内の“全範囲”を指定すると，スクロール範囲の設定は行いません（範囲は制限されません）。

備考 このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

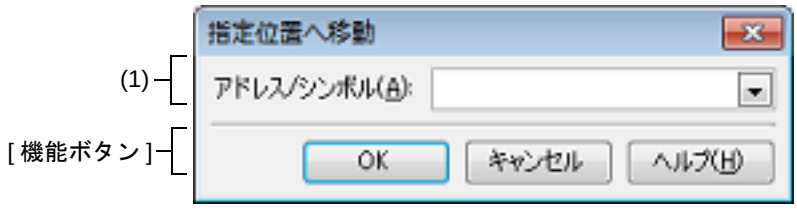
［機能ボタン］

ボタン	機能
OK	指定したスクロール範囲を対象パネルに設定し，開始アドレスを表示の先頭として対象パネルにキャレットを移動します。
キャンセル	設定を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

指定位置へ移動 ダイアログ

指定した位置にキャレットを移動します。

図 A.46 指定位置へ移動 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- 逆アセンブル パネルにフォーカスがある状態で，［編集］メニュー→［移動 ...］を選択
- IOR パネルにフォーカスがある状態で，［編集］メニュー→［移動 ...］を選択
- 逆アセンブル パネルにおいて，コンテキスト・メニューの［移動 ...］を選択
- IOR パネルにおいて，コンテキスト・メニューの［移動 ...］を選択

［各エリアの説明］

(1) ［アドレス / シンボル］／［IOR］エリア
キャレットを移動したい箇所を指定します。
テキスト・ボックスに直接入力するか（最大指定文字数：1024 文字），またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
対象となるパネルにより，指定内容は次のように異なります。

対象パネル	指定内容
逆アセンブル パネル	アドレス式
IOR パネル	I/O レジスタ名

備考 逆アセンブル パネルよりこのダイアログをオープンした場合，このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

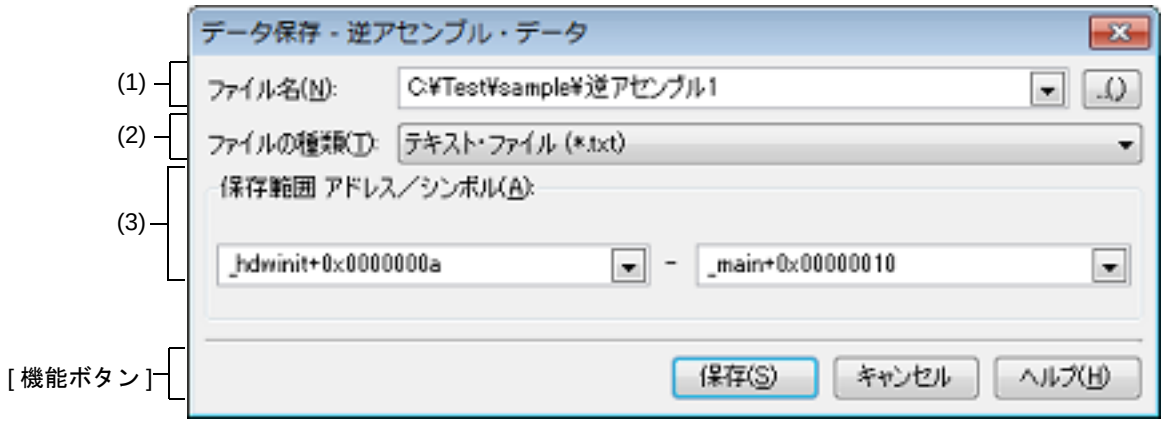
［機能ボタン］

ボタン	機能
OK	指定した位置を表示の先頭として対象パネルにキャレットを移動します。
キャンセル	設定を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

データ保存 ダイアログ

逆アセンブル パネル／メモリ パネル／トレース パネルの表示内容，およびアップロード・データの保存（「2.5.3 アップロードを実行する」参照）を行います。
なお，このダイアログは，デバッグ・ツールと接続時のみオープンすることができます。

図 A.47　データ保存 ダイアログ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［各エリアの説明］
- ［機能ボタン］

［オープン方法］

- 逆アセンブル パネルにフォーカスがある状態で，［ファイル］メニュー→［名前を付けて逆アセンブル・データを保存 ...］を選択
- メモリ パネルにフォーカスがある状態で，［ファイル］メニュー→［名前を付けてメモリ・データを保存 ...］を選択
- トレース パネルにフォーカスがある状態で，［ファイル］メニュー→［名前を付けてトレース・データを保存 ...］を選択
- ［デバッグ］メニュー→［デバッグ・ツールからアップロード ...］を選択

［各エリアの説明］

(1)　［ファイル名］エリア
保存するファイル名を指定します。
テキスト・ボックスに直接入力するか（最大指定文字数：259 文字），またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
また，［...］ボタンをクリックすることでオープンするデータ保存ファイルを選択 ダイアログにより，ファイルを選択することもできます。
なお，パス情報を含まずファイル名のみを指定した場合は，プロジェクト・フォルダが対象となります。

(2)　［ファイルの種類］エリア
保存するファイルの形式を次のドロップダウン・リストにより選択します。
保存する対象により，選択できるファイルの形式が次のように異なります。

(a)　パネルの表示内容を保存する場合

テキスト・ファイル (*.txt)	テキスト形式（デフォルト）
CSV(カンマ区切り) (*.csv)	CSV 形式[注]

注　各データを “,” で区切り保存します。
なお，データ内に “,” が含まれている際の不正形式を避けるため，各データを ""（ダブルクォーテーション）で括り出力します。

(b) アップロード・データを保存する場合

インテル・ヘキサ・フォーマット (*.hex)	インテル拡張ヘキサ・フォーマット
モトローラ S フォーマット (*.mot)	モトローラ・S タイプ・フォーマット
バイナリ・データ (*.bin)	バイナリ・フォーマット

備考　アップロードについての詳細は，「2.5.3 アップロードを実行する」を参照してください。

(3) ［保存範囲 *xxx*］エリア
ファイルに保存する際の保存範囲を指定します。
それぞれのテキスト・ボックスに直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。
保存する対象により，指定方法が次のように異なります。

保存対象	説明
逆アセンブル パネル	保存するアドレス範囲を，開始アドレスと終了アドレスで指定します。 16 進数の数値，またはアドレス式による入力が可能です。 パネル上で範囲選択している場合は，デフォルトでその選択範囲が指定されます。 範囲選択していない場合は，現在のパネルの表示範囲が指定されます。
メモリ パネル	保存するメモリ範囲を，開始アドレスと終了アドレスで指定します。 16 進数の数値，またはアドレス式による入力が可能です。 範囲選択していない場合は，現在のパネルの表示範囲が指定されます。
トレース パネル	- 保存範囲を指定する場合 保存するトレース範囲を開始トレース番号[注]と終了トレース番号で指定します。 10 進数の数値のみ入力が可能です。 - すべてのトレース・データを保存する場合 左側のドロップダウン・リストにより，［すべてのトレース・データ］を選択します。右側のテキスト・ボックスが無効となり，現在取得しているトレース・データのすべてが保存の対象となります。 デフォルトでは，現在のパネルの表示範囲が指定されます。
アップロード・データ	保存するメモリ範囲を開始アドレスと終了アドレスで指定します。 16 進数の数値，またはアドレス式による入力が可能です。

注　トレース パネル上の［番号］エリアに表示されている番号を示します。

備考　このテキスト・ボックスで［Ctrl］+［Space］キーを押下することにより，現在のキャレット位置のシンボル名を補完することができます（「2.18.2 シンボル名の入力補完機能」参照）。

［機能ボタン］

ボタン	機能
保存	指定したファイルに，指定した形式でデータを保存します。
キャンセル	データ保存の設定を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2014.08.01	ー	初版発行

CS+ V3.00.00 ユーザーズマニュアル
RH850 デバッグ・ツール編

発行年月日　2014年 8月 1日　Rev.1.00
発行　ルネサス エレクトロニクス株式会社
〒211-8668 神奈川県川崎市中原区下沼部 1753

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口

http://www.renesas.com

※営業お問合せ窓口の住所は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス株式会社 〒100-0004 千代田区大手町2-6-2（日本ビル）

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/contact/

Colophon 2.0

CS+ V3.00.00

RENESAS

ルネサス エレクトロニクス株式会社

R20UT3082JJ0100