User's Manual

CubeSuite+ V1.00.00

統合開発環境

ユーザーズマニュアル　RL78 デバッグ編

対象デバイス

RL78 ファミリ

本資料に記載の全ての情報は本資料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　2011.04

ご注意書き

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

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

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

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

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

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

このマニュアルの使い方

このマニュアルは，RL78 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 CubeSuite+ について説明します。

CubeSuite+は，RL78 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，CubeSuite+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，CubeSuite+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用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。

第 1 章　概　　説
第 2 章　機　　能
付録 A　ウインドウ・リファレンス
付録 B　ユーザ・オープン・インタフェース
付録 C　索　　引

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的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	：$\overline{\times\times\times}$（端子，信号名称に上線）
	注	：本文中につけた注の説明
	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　…　××××
		16 進数　…　0x××××

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。

資　料　名		資料番号	
		和　文	英　文
CubeSuite+ 統合開発環境 ユーザーズ・マニュアル	起動編	R20UT0545J	R20UT0545E
	78K0 設計編	R20UT0546J	R20UT0546E
	78K0R 設計編	R20UT0547J	R20UT0547E
	RL78 設計編	R20UT0548J	R20UT0548E
	V850 設計編	R20UT0549J	R20UT0549E
	R8C 設計編	R20UT0550J	R20UT0550E
	78K0 コーディング編	R20UT0551J	R20UT0551E
	RL78,78K0R コーディング編	R20UT0552J	R20UT0552E
	V850 コーディング編	R20UT0553J	R20UT0553E
	コーディング編（CX コンパイラ）	R20UT0554J	R20UT0554E
	R8C コーディング編	R20UT0576J	R20UT0576E
	78K0 ビルド編	R20UT0555J	R20UT0555E
	RL78,78K0R ビルド編	R20UT0556J	R20UT0556E
	V850 ビルド編	R20UT0557J	R20UT0557E
	ビルド編（CX コンパイラ）	R20UT0558J	R20UT0558E
	R8C ビルド編	R20UT0575J	R20UT0575E
	78K0 デバッグ編	R20UT0559J	R20UT0559E
	78K0R デバッグ編	R20UT0560J	R20UT0560E
	RL78 デバッグ編	このマニュアル	R20UT0561E
	V850 デバッグ編	R20UT0562J	R20UT0562E
	R8C デバッグ編	R20UT0574J	R20UT0574E
	解析編	R20UT0563J	R20UT0563E
	メッセージ編	R20UT0407J	R20UT0407E

注意　上記関連資料は，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。

この資料に記載されている会社名，製品名などは，各社の商標または登録商標です。

〔メ　　モ〕

〔メ　　モ〕

〔メ　　モ〕

目　　次

第１章　概　　説　…　11

1. 1　概　　要　…　11
1. 2　特　　長　…　11

第２章　機　　能　…　13

2. 1　概　　要　…　13
2. 2　デバッグを始める前の準備　…　16
　2. 2. 1　ホスト・マシンとの接続を確認する　…　16
2. 3　デバッグ・ツールの動作環境設定　…　19
　2. 3. 1　使用するデバッグ・ツールを選択する　…　19
　2. 3. 2　【IECUBE】の場合　…　20
　2. 3. 3　【E1】の場合　…　30
　2. 3. 4　【E20】の場合　…　39
　2. 3. 5　【EZ Emulator】の場合　…　47
　2. 3. 6　【シミュレータ】の場合　…　55
2. 4　デバッグ・ツールとの接続／切断　…　62
　2. 4. 1　CubeSuite+ にデバッグ・ツールを接続する　…　62
　2. 4. 2　CubeSuite+ からデバッグ・ツールを切断する　…　62
2. 5　ダウンロード／アップロード　…　63
　2. 5. 1　ダウンロードを実行する　…　63
　2. 5. 2　応用的なダウンロード方法　…　66
　2. 5. 3　アップロードを実行する　…　71
2. 6　プログラムの表示と変更　…　73
　2. 6. 1　ソース・ファイルを表示する　…　74
　2. 6. 2　逆アセンブル結果を表示する　…　79
　2. 6. 3　他の処理と平行してビルドを実行する　…　83
　2. 6. 4　ライン・アセンブルを行う　…　84
2. 7　プログラムの実行　…　86
　2. 7. 1　マイクロコントローラ（CPU）をリセットする　…　86
　2. 7. 2　プログラムを実行する　…　87
　2. 7. 3　プログラムをステップ実行する　…　89
2. 8　プログラムの停止（ブレーク）　…　91
　2. 8. 1　プログラムの実行を手動で停止する　…　91
　2. 8. 2　任意の場所で停止する（ブレークポイント）　…　91
　2. 8. 3　変数 /SFR へのアクセスで停止する　…　94
　2. 8. 4　不正な実行を検出して停止する【IECUBE】　…　98
　2. 8. 5　その他のブレーク要因　…　99
2. 9　メモリ，レジスタ，変数の表示／変更　…　100
　2. 9. 1　メモリを表示／変更する　…　100
　2. 9. 2　CPU レジスタを表示／変更する　…　109
　2. 9. 3　SFR を表示／変更する　…　112

2. 9. 4　グローバル変数／スタティック変数を表示／変更する　…　114
2. 9. 5　ローカル変数を表示／変更する　…　114
2. 9. 6　ウォッチ式を表示／変更する　…　117
2. 10　スタックからの関数呼び出し情報の表示　…　123
2. 10. 1　コール・スタック情報を表示する　…　123
2. 11　実行履歴の収集【IECUBE】【シミュレータ】　…　125
2. 11. 1　トレース動作の設定をする　…　125
2. 11. 2　実行停止までの実行履歴を収集する　…　128
2. 11. 3　任意区間の実行履歴を収集する　…　128
2. 11. 4　条件を満たした場合の実行履歴を収集する　…　130
2. 11. 5　実行履歴を表示する　…　132
2. 11. 6　トレース・メモリをクリアする　…　134
2. 11. 7　トレース・データを検索する　…　134
2. 11. 8　実行履歴の表示内容を保存する　…　141
2. 12　実行時間の計測　…　143
2. 12. 1　実行開始から停止までの実行時間を計測する　…　143
2. 12. 2　任意区間の実行時間を計測する【IECUBE】【シミュレータ】　…　144
2. 12. 3　測定可能時間の範囲　…　146
2. 13　カバレッジの測定【IECUBE】【シミュレータ】　…　147
2. 13. 1　カバレッジ測定の設定をする　…　147
2. 13. 2　カバレッジ測定結果を表示する　…　148
2. 14　プログラム内へのアクションの設定　…　151
2. 14. 1　printf を挿入する　…　151
2. 15　イベントの管理　…　154
2. 15. 1　設定状態（有効／無効）を変更する　…　154
2. 15. 2　特定のイベント種別のみ表示する　…　155
2. 15. 3　イベントのアドレスにジャンプする　…　156
2. 15. 4　イベントを削除する　…　156
2. 15. 5　イベントにコメントを入力する　…　156
2. 15. 6　イベント設定に関する留意事項　…　156
2. 16　フック処理を設定する　…　159
2. 17　シミュレータ GUI の使用【シミュレータ】　…　161
2. 17. 1　マイコンの入出力波形を確認する　…　162
2. 17. 2　端子へ信号を入力する　…　163
2. 17. 3　シリアル通信を行う　…　164
2. 17. 4　ボタン /LED/ レベル・ゲージ等の部品を使用する　…　165
2. 18　入力値について　…　166
2. 18. 1　入力規約　…　166
2. 18. 2　入力不備箇所に対するアイコン表示　…　167

付録 A　ウインドウ・リファレンス　…　168

A. 1　説　　明　…　168

付録 B　ユーザ・オープン・インタフェース　…　437

B. 1　概　　要　…　437
B. 1. 1　インタフェース関数の種類　…　438
B. 1. 2　インタフェース方式　…　438

B. 1. 3　開発環境　…　439
B. 2　ユーザ・モデルの作成　…　440
B. 2. 1　プログラム構成　…　440
B. 2. 2　ユーザ・モデルのプログラミング　…　441
B. 2. 3　プログラム・ファイル（UserModel.c）の記述例　…　443
B. 2. 4　コンパイルとリンク　…　444
B. 3　ユーザ・モデルの組み込み　…　445
B. 3. 1　シミュレータ・コンフィギュレーション・ファイルへの記述　…　445
B. 3. 2　シミュレータ・コンフィギュレーション・ファイルの記述例　…　448
B. 4　提供インタフェース関数　…　449
B. 4. 1　概　　要　…　449
B. 4. 2　基本インタフェース関数　…　451
B. 4. 3　時間インタフェース関数　…　455
B. 4. 4　端子インタフェース関数　…　464
B. 4. 5　外部バス・インタフェース関数　…　478
B. 4. 6　シリアル・インタフェース関数　…　486
B. 4. 7　信号出力器インタフェース関数　…　511
B. 4. 8　エラー番号一覧　…　519
B. 5　ユーザ定義関数　…　521
B. 6　サンプル・プログラム（Timer モデル）　…　535
B. 6. 1　概　　要　…　535
B. 6. 2　構　　成　…　535
B. 6. 3　動　　作　…　535
B. 6. 4　プロジェクト・ファイル　…　536
B. 6. 5　プログラム詳細　…　537

付録 C　索　　引　…　541

第 1 章　概　　説

CubeSuite+ は，V850 マイクロコントローラ，R8C 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。

CubeSuite+ 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，CubeSuite+ 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

CubeSuite+ が提供するデバッグ機能を使用することにより，RL78 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，CubeSuite+ が提供するデバッグ機能の特長を示します。

- 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（IECUBE），オンチップ・デバッギング・エミュレータ（E1/E20/EZ Emulator），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（コード・フラッシュ）の対応
  IECUBE と接続することにより，フラッシュ・セルフ・プログラミング・エミュレーションを行うことができます。

- データフラッシュ・メモリへの書き込み
  選択しているマイクロコントローラがデータフラッシュ・メモリ内蔵品の場合では，通常のメモリ操作と同様のアクセス方法により，データフラッシュ・メモリの内容を表示／変更することができます（シミュレータを除く）。

- 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。

第 2 章 機 能

この章では，CubeSuite+ 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

CubeSuite+ 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) CubeSuite+ を起動する

Windows® の［スタート］メニューから CubeSuite+ を起動します。

備考 “CubeSuite+ を起動する”についての詳細は，「CubeSuite+ 起動編」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，「CubeSuite+ 起動編」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 CA78K0R 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，「CubeSuite+ 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（IECUBE/E1/E20/EZ Emulator/ 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) において選択したデバッグ・ツールのための動作環境を設定します。

- 【IECUBE】の場合
- 【E1】の場合
- 【E20】の場合
- 【EZ Emulator】の場合
- 【シミュレータ】の場合

(7) CubeSuite+ にデバッグ・ツールを接続する

CubeSuite+ 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) において作成したロード・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. 8. 2 任意の場所で停止する（ブレークポイント）」／「2. 8. 3 変数 /SFR 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. 15. 6 イベント設定に関する留意事項」を参照してください。

(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) においてブレークポイント／ブレーク・イベントを設定している場合では，その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【IECUBE】【シミュレータ】注
- 実行時間の計測注
- カバレッジの測定【IECUBE】【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. 15. 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

- シミュレータ GUI の使用【シミュレータ】

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
- SFR 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（ヘキサ・フォーマット／バイナリ・データ・フォーマットなど）で保存します。

(14) CubeSuite+ からデバッグ・ツールを切断する

CubeSuite+ 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，「CubeSuite+ 起動編」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

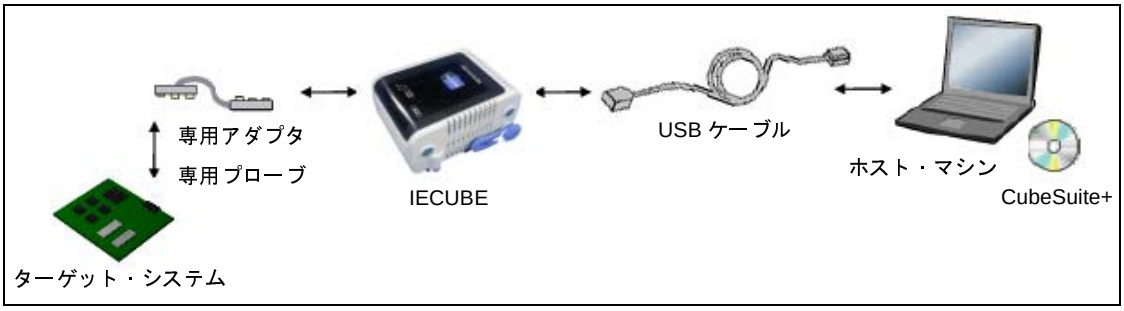
使用するデバッグ・ツールごとに，ホスト・マシンとの接続例を示します。

（1）【IECUBE】の場合
（2）【E1】の場合
（3）【E20】の場合
（4）【EZ Emulator】の場合
（5）【シミュレータ】の場合

（1）【IECUBE】の場合

ホスト・マシン，IECUBE，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，IECUBE のユーザーズ・マニュアルを参照してください。

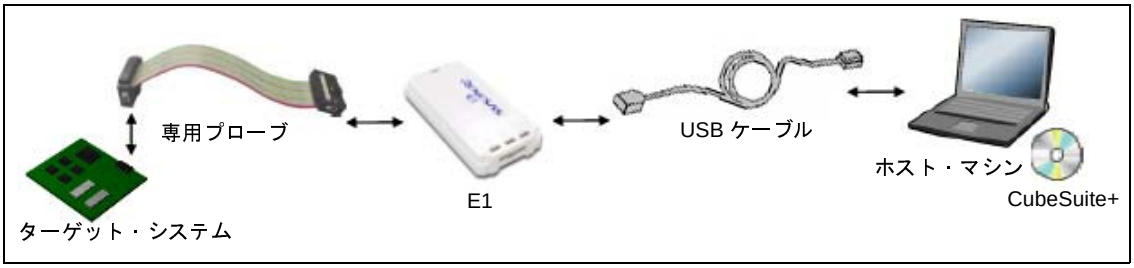
図 2—1　ホスト・マシンとデバッグ・ツールとの接続例【IECUBE】



（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】



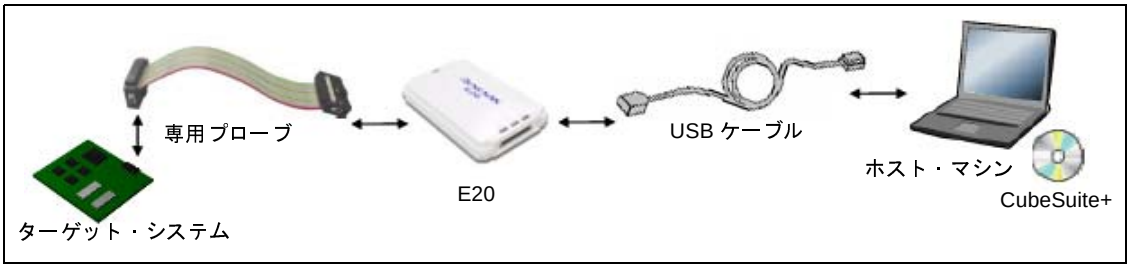
注意 ターゲット・システムとの通信方式として，シリアル通信のみサポートしています（JTAG 通信は使用不可）。

(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】



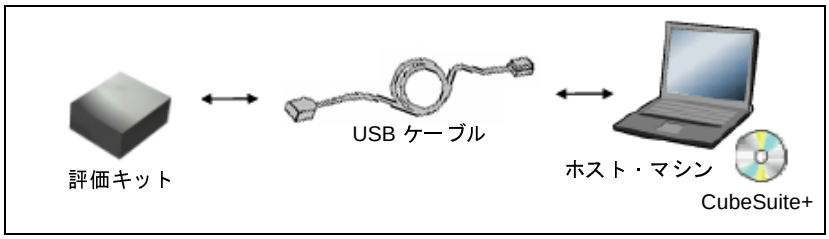
注意 ターゲット・システムとの通信方式として，シリアル通信のみサポートしています（JTAG 通信は使用不可）。

(4)【EZ Emulator】の場合

ホスト・マシン，評価キットなどを接続します。

接続方法についての詳細は，EZ Emulator のユーザーズ・マニュアルを参照してください。

図 2—4 ホスト・マシンとデバッグ・ツールとの接続例【EZ Emulator】



(5)【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—5 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。

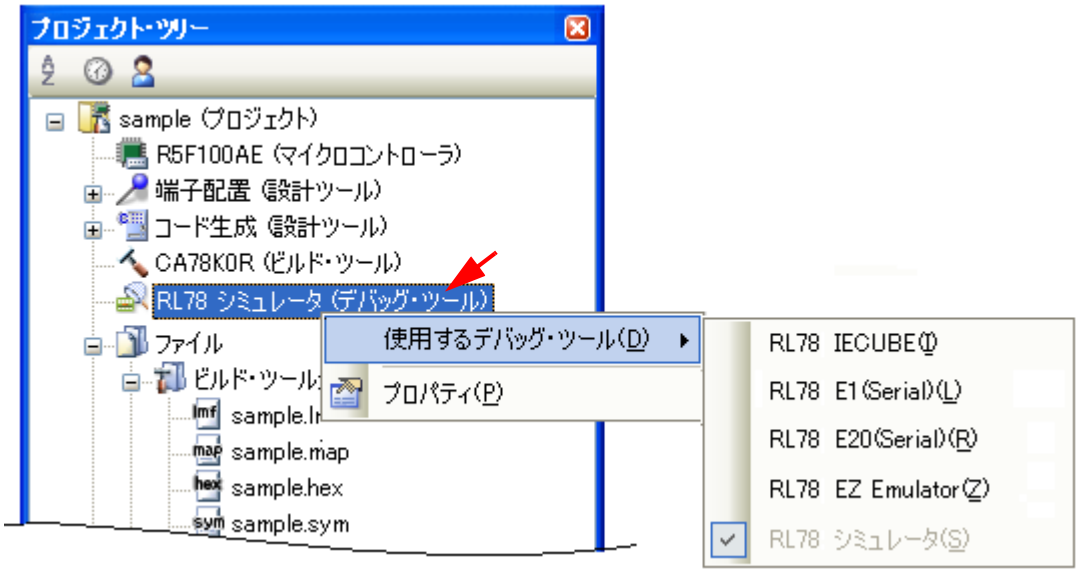
2. 3. 1 使用するデバッグ・ツールを選択する

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［RL78 *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・メニューにより行ってください。

図 2—6 使用するデバッグ・ツールの選択／切り替え



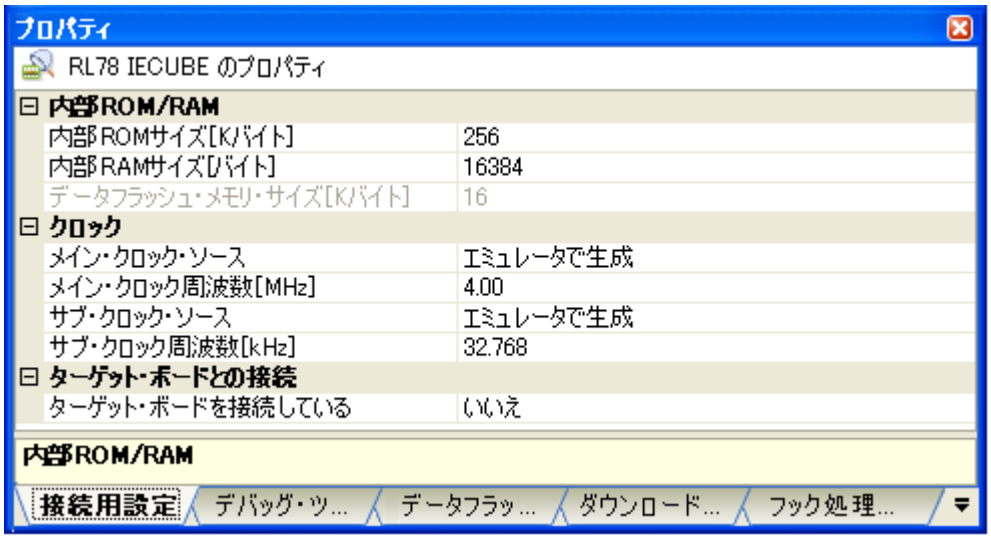
すでにプロパティ パネルがオープンしている場合，再び［RL78 *デバッグ・ツール名*(デバッグ・ツール)］ノードをクリックすると，選択したデバッグ・ツールのプロパティ パネルに表示が切り替わります。

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 【IECUBE】の場合

IECUBE 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—7 動作環境設定【IECUBE】（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。

（1）［接続用設定］タブ
（2）［デバッグ・ツール設定］タブ
（3）［データフラッシュ・エミュレーション設定］タブ
（4）［ダウンロード・ファイル設定］タブ
（5）［フック処理設定］タブ

（1）［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，デバッグ・ツールとの接続に関する設定を行います。

（a）［内部 ROM/RAM］
（b）［クロック］
（c）［ターゲット・ボードとの接続］

（a）［内部 ROM/RAM］

このカテゴリでは，内部 ROM/RAM に関する設定を行います。
デフォルトで，選択しているマイクロコントローラの内部 ROM/RAM サイズが設定されます。

備考 選択しているマイクロコントローラと同様のメモリ・マッピングでデバッグを行う場合は，このカテゴリ内の設定を変更する必要はありません。

図 2—8 ［内部 ROM/RAM］カテゴリ【IECUBE】

⊟ 内部ROM/RAM	
内部ROMサイズ[Kバイト]	256
内部RAMサイズ[バイト]	16384
データフラッシュ・メモリ・サイズ[Kバイト]	16

- ［内部 ROM サイズ [K バイト]］

  エミュレーションする内部 ROM サイズを指定します（単位：K バイト）。

  IECUBE のメモリ資源を利用して，マッピングを変更後にデバッグを行う場合は，ドロップダウン・リストにより指定してください。

- ［内部 RAM サイズ [バイト]］

  エミュレーションする内部 RAM サイズを指定します（単位：バイト）。

  IECUBE のメモリ資源を利用して，マッピングを変更後にデバッグを行う場合は，ドロップダウン・リストにより指定してください。

- ［データフラッシュ・メモリ・サイズ [K バイト]］

  選択しているマイクロコントローラのデータフラッシュ・メモリ領域のサイズを表示します（単位：K バイト）。

  このプロパティ値を変更することはできません。

注意　設定を変更する際は，他のメモリ・マッピング領域と重複しないよう注意が必要です。

(b)［クロック］

このカテゴリでは，クロックに関する設定を行います。

図 2—9 ［クロック］カテゴリ【IECUBE】

⊟ クロック	
メイン・クロック・ソース	エミュレータで生成
メイン・クロック周波数[MHz]	4.00
サブ・クロック・ソース	エミュレータで生成
サブ・クロック周波数[kHz]	32.768

- ［メイン・クロック・ソース］

  CPU に入力するメイン・クロック・ソースを次のドロップダウン・リストにより指定します。

クロック・ソケット	IECUBE のクロック・ソケットに実装されている発振器のクロックを使用します。 ただし，クロック・ソケットに発振器がない場合，この項目は表示されません。
外部	ターゲット・システムのメイン・クロック（矩形波）を使用します。 ただし，ターゲット・システムの電源が検出不能だった場合，この項目は表示されません。
エミュレータで生成	IECUBE 内部で生成したクロックを使用します（デフォルト）。

注意 IECUBE と接続中にこのプロパティを変更することはできません。

- ［メイン・クロック周波数 [MHz]］

このプロパティは，［メイン・クロック・ソース］プロパティにおいて，［エミュレータで生成］を指定した場合にのみ表示されます。

メイン・クロックの周波数をドロップダウン・リストにより指定します。

ドロップダウン・リストには，次の周波数が表示されます（単位：MHz）。

1.00，2.00，3.00，3.57，4.00（デフォルト），4.19，4.91，5.00，6.00，8.00，8.38，10.00，12.00，16.00，20.00

- ［サブ・クロック・ソース］

CPU と周辺機器に入力するサブ・クロック・ソースを次のドロップダウン・リストにより指定します。

外部	ターゲット・システムのメイン・クロック（矩形波）を使用します。
エミュレータで生成	IECUBE 内部で生成したクロックを使用します（デフォルト）。

注意 IECUBE と接続中にこのプロパティを変更することはできません。

- ［サブ・クロック周波数 [kHz]］

このプロパティは，［サブ・クロック・ソース］プロパティにおいて，［エミュレータで生成］を指定した場合にのみ表示されます。

サブ・クロック周波数をドロップダウン・リストにより指定します。

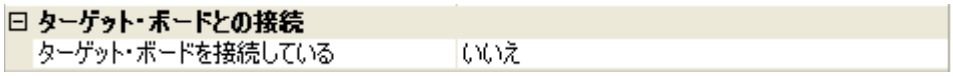
ドロップダウン・リストには，次の周波数が表示されます（単位：kHz）。

32.768（デフォルト），38.40

(c)［ターゲット・ボードとの接続］

このカテゴリでは，ターゲット・ボードとの接続に関する設定を行います。

図 2—10 ［ターゲット・ボードとの接続］カテゴリ【IECUBE】



- ［ターゲット・ボードを接続している］

IECUBE にターゲット・ボードを接続しているか否かをドロップダウン・リストにより指定します。

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（デフォルトでは［いいえ］が指定されます）。

注意 IECUBE と接続中にこのプロパティを変更することはできません。

(2) ［デバッグ・ツール設定］タブ

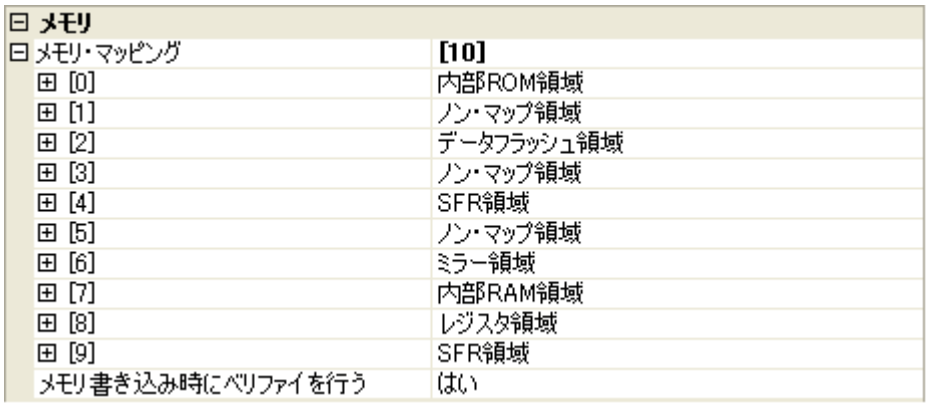
［デバッグ・ツール設定］タブでは，次に示すカテゴリごとに，デバッグ・ツールの基本設定を行います。

(a) ［メモリ］
(b) ［実行中のメモリ・アクセス］
(c) ［実行中のイベント設定］
(d) ［ブレーク］
(e) ［フェイルセーフ・ブレーク］
(f) ［トレース］
(g) ［カバレッジ］
(h) ［入力信号のマスク］

(a) ［メモリ］

このカテゴリでは，メモリに関する設定を行います。

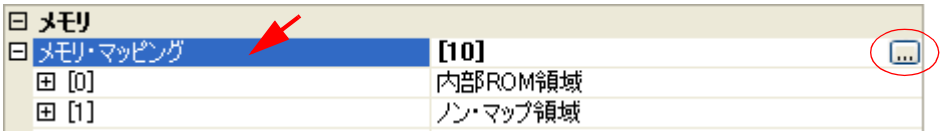
図 2—11 ［メモリ］カテゴリ【IECUBE】



- ［メモリ・マッピング］

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。
このパネル上でマッピング値を変更することはできません。メモリ・マッピングを追加する必要がある場合は，［メモリ・マッピング］プロパティを選択することで設定欄右端に表示される［...］ボタンのクリックによりオープンするメモリ・マッピング ダイアログで行います。
設定方法についての詳細は，メモリ・マッピング ダイアログの項を参照してください。

図 2—12 メモリ・マッピング ダイアログのオープン



注意 デバッグ・ツールと未接続の場合，ユーザにより追加されたメモリ・マッピング領域のみが表示対象となります。
デバッグ・ツールと接続することにより（「2. 4. 1 CubeSuite+ にデバッグ・ツールを接続する」参照），各メモリ種別ごとの詳細表示を行います。

- ［メモリ書き込み時にベリファイを行う］
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ドロップダウン・リストにより指定します。
ベリファイを行う場合は［はい］を選択してください（デフォルト）。

(b)［実行中のメモリ・アクセス］

このカテゴリでは，プログラム実行中におけるメモリ・アクセスに関する設定を行います。
このカテゴリ内の設定は，リアルタイム表示更新機能を使用する場合に必要となります。リアルタイム表示更新機能についての詳細は，「(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—13 ［実行中のメモリ・アクセス］カテゴリ【IECUBE】

⊟ 実行中のメモリ・アクセス	
実行を一瞬停止してアクセスする	いいえ
実行中に表示更新を行う	はい
表示更新間隔[ms]	500

- ［実行を一瞬停止してアクセスする］
プログラム実行中にはアクセスできないメモリ領域（ターゲット・メモリ領域 /SFR 領域 /CPU レジスタなど）に対して，アクセスを許可するか否かをドロップダウン・リストにより指定します。
アクセスを許可する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

- ［実行中に表示更新を行う］
プログラム実行中に，ウォッチ パネル／メモリ パネルの表示内容を更新するか否かをドロップダウン・リストにより指定します。
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］を指定した場合のみ表示されます。
プログラム実行中に，ウォッチ パネル／メモリ 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。
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（デフォルトでは［500］が指定されます）。

(c)［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—14 ［実行中のイベント設定］カテゴリ

⊟ 実行中のイベント設定	
実行を一瞬停止してイベントを設定する	いいえ

- ［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，プログラムの実行を強制的に一瞬停止させることで設定を行うか否かをドロップダウン・リストにより指定します。

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) 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（デフォルトでは［いいえ］が指定されます）。

(d)［ブレーク］

このカテゴリでは，ブレーク機能に関する設定を行います。

図 2—15 ［ブレーク］カテゴリ【IECUBE】

⊟ ブレーク	
優先的に使用するブレークポイントの種類	ソフトウエア・ブレーク
停止時にタイマ系周辺エミュレーションを停止する	いいえ
停止時にシリアル系周辺エミュレーションを停止する	いいえ
オープン・ブレーク機能を使用する	いいえ(Output signal)

- ［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，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使用するブレークポイントの種別を次のドロップダウン・リストにより指定します。

なお，ブレークポイントについての詳細は，「2. 8. 2 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。

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（デフォルト）。
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。

- ［停止時にタイマ系周辺エミュレーションを停止する］

プログラム実行停止時に，エミュレータのタイマ系周辺エミュレーション機能を停止するか否かをドロップダウン・リストにより指定します。

停止する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

- ［停止時にシリアル系周辺エミュレーションを停止する］

プログラム実行停止時に，エミュレータのシリアル系周辺エミュレーション機能を停止するか否かをドロップダウン・リストにより指定します。

停止する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

- ［オープン・ブレーク機能を使用する］

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オープン・ブレーク機能をサポートしている場合のみ表示されます。

オープン・ブレーク機能を使用するか否かを次のドロップダウン・リストにより指定します。

はい (Hi-Z)	CPU 停止時，オープン・ブレークの対象端子は Hi-Z になります。
いいえ (Output signal)	CPU 停止時，オープン・ブレークの対象端子は通常の状態のままで動作します（デフォルト）。

(e)［フェイルセーフ・ブレーク］

このカテゴリでは，フェイルセーフ・ブレーク機能に関する設定を行います。

フェイルセーフ・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. 8. 4　不正な実行を検出して停止する【IECUBE】」を参照してください。

(f)［トレース］

このカテゴリでは，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. 11　実行履歴の収集【IECUBE】【シミュレータ】」を参照してください。

(g)［カバレッジ］

このカテゴリでは，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. 13　カバレッジの測定【IECUBE】【シミュレータ】」を参照してください。

(h)［入力信号のマスク］

このカテゴリでは，入力信号のマスクに関する設定を行います。

図 2—16　［入力信号のマスク］カテゴリ【IECUBE】

⊟ 入力信号のマスク	
WAIT 信号をマスクする	いいえ
TARGET RESET 信号をマスクする	いいえ
INTERNAL RESET 信号をマスクする	いいえ
NMI 信号をマスクする	いいえ

次に示す各プロパティの設定において，該当する信号をマスクする場合は［はい］を，マスクしない場合は［いいえ］をドロップダウン・リストにより指定してください（デフォルトでは，すべてのプロパティに［いいえ］が指定されます）。

- ［WAIT 信号をマスクする］注
- ［TARGET RESET 信号をマスクする］注
- ［INTERNAL RESET 信号をマスクする］
- ［NMI 信号をマスクする］

注 ［接続設定用］タブ上の［ターゲット・ボードとの接続］プロパティにおいて［いいえ］を指定している場合，デバッグ・ツールと接続時に，自動的に［はい］に固定となります（変更不可）。

(3)［データフラッシュ・エミュレーション設定］タブ

［データフラッシュ・エミュレーション設定］タブ【IECUBE】では，次に示すカテゴリごとに，データフラッシュ・エミュレーションの設定を行います。

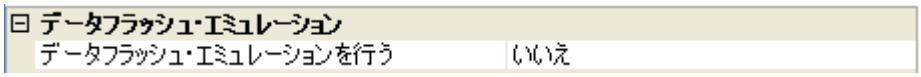
ただし，このタブは，選択しているマイクロコントローラがデータフラッシュ・メモリ内蔵品の場合のみ表示されます。

(a)［データフラッシュ・エミュレーション］
(b)［EEPROM ライブラリ・エラー］
(c)［書き込み／消去時間設定］

(a)［データフラッシュ・エミュレーション］

このカテゴリでは，データフラッシュ・エミュレーション機能に関する設定を行います。

図 2—17 ［データフラッシュ・エミュレーション］カテゴリ



- ［データフラッシュ・エミュレーションを行う］

データフラッシュ・エミュレーション機能を使用するか否かをドロップダウン・リストにより指定します。

データフラッシュ・エミュレーション機能を使用する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

(b)［EEPROM ライブラリ・エラー］

このカテゴリでは，EEPROM エミュレーション・ライブラリに関する設定として，EEPROM エミュレーション・ライブラリ関数の動作を設定します。

図 2—18 ［EEPROM ライブラリ・エラー］カテゴリ

⊟ EEPROMライブラリ・エラー	
WordWrite関数でエラーを返す	はい
WordWrite関数エラーの発生アドレス	HEX 0
BlockErase関数でエラーを返す	はい
BlockErase関数エラーの発生アドレス	HEX 0
BlockIVerify関数でエラーを返す	はい
BlockIVerify関数エラーの発生アドレス	HEX 0
BlockBlankCheck関数でエラーを返す	はい
BlockBlankCheck関数エラーの発生アドレス	HEX 0

次に示す各プロパティにおいて，該当する関数で発生するエラー値を強制的に返すか否かをドロップダウン・リストにより指定します（関数で発生するエラー値は，通常のエミュレーションでは返ることがありません）。デフォルトでは，すべてに［いいえ］が設定されています。

エラー値を強制的に返す場合は，該当するプロパティを［はい］に指定したのち，下段に表示される［*xxx* 関数でエラーを返す］プロパティにおいて，エラーを発生させるデータフラッシュ・メモリ領域内のアドレスを，0x0 ～ 0xFFFFFFFF の範囲の 16 進数で直接入力により指定してください（デフォルトでは［0］が指定されます）。

- ［WordWrite 関数でエラーを返す］
- ［BlockErase 関数でエラーを返す］
- ［BlockIVerify 関数でエラーを返す］
- ［BlockBlankCheck 関数でエラーを返す］

(c)［書き込み／消去時間設定］

このカテゴリでは，データフラッシュ・メモリへの書き込み，または消去時間に関する設定を行います。

図 2—19 ［書き込み／消去時間設定］カテゴリ

⊟ 書き込み／消去時間設定	
データフラッシュ・メモリへの書き込み時間	フラッシュ・マクロ・スペック上の想定回数
データフラッシュ・メモリの消去時間	フラッシュ・マクロ・スペック上の想定回数

- ［データフラッシュ・メモリへの書き込み時間］

データフラッシュ・メモリへ書き込む際の遅延時間を次のドロップダウン・リストにより指定します。

リトライなし	リトライ回数を 0 とし，遅延時間 =0（書き込み時間最速）とします。
フラッシュ・マクロ・スペック上の想定回数	使用するフラッシュ・マクロのスペック上の想定回数を遅延時間とします（デフォルト）。
フラッシュ・マクロ・スペック上の想定最大回数	使用するフラッシュ・マクロのスペック上の想定最大回数を遅延時間とします。
リトライ回数最大	リトライ回数を最大とし，遅延時間を最大（書き込み時間最長）とします。

- ［データフラッシュ・メモリの消去時間］

データフラッシュ・メモリの消去を行う際の遅延時間を次のドロップダウン・リストにより指定します。

リトライなし	リトライ回数を 0 とし，遅延時間 =0（消去時間最速）とします。
フラッシュ・マクロ・スペック上の想定回数	使用するフラッシュ・マクロのスペック上の想定回数を遅延時間とします（デフォルト）。
フラッシュ・マクロ・スペック上の想定最大回数	使用するフラッシュ・マクロのスペック上の想定最大回数を遅延時間とします。
リトライ回数最大	リトライ回数を最大とし，遅延時間を最大（消去時間最長）とします。

(4)［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，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。

(5)［フック処理設定］タブ

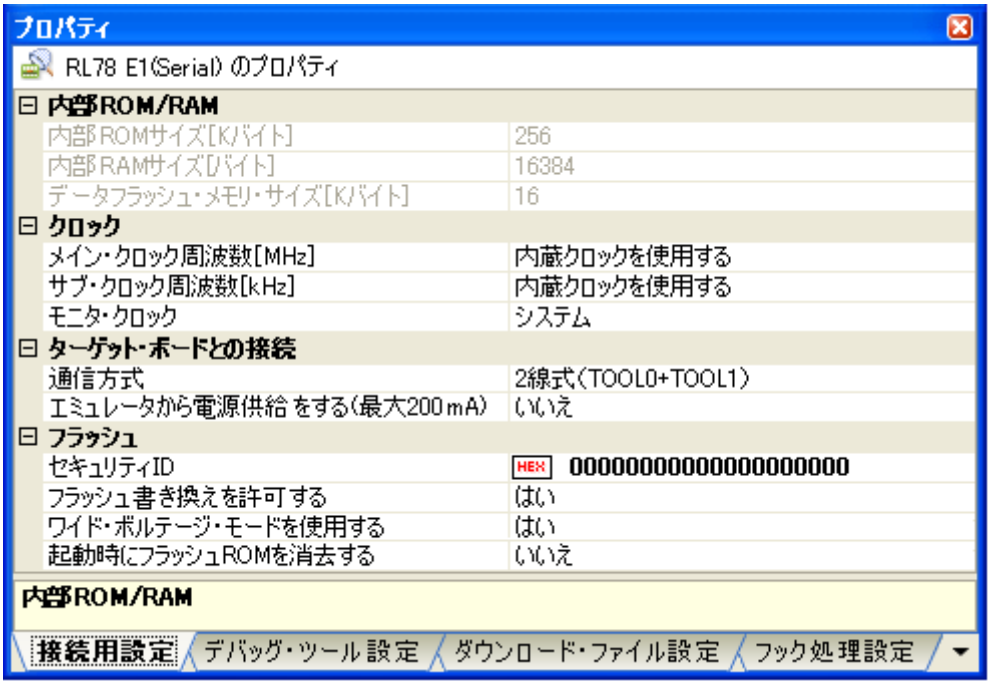
［フック処理設定］タブでは，デバッグ・ツールにフック処理の設定を行います。

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. 16　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。

図 2—20 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。

（1）［接続用設定］タブ
（2）［デバッグ・ツール設定］タブ
（3）［ダウンロード・ファイル設定］タブ
（4）［フック処理設定］タブ

（1）［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，デバッグ・ツールとの接続に関する設定を行います。

（a）［内部 ROM/RAM］
（b）［クロック］
（c）［ターゲット・ボードとの接続］
（d）［フラッシュ］

(a)［内部 ROM/RAM］

このカテゴリでは，内部 ROM/RAM に関する設定を表示します。

図 2—21 ［内部 ROM/RAM］カテゴリ【E1】

内部ROM/RAM	
内部ROMサイズ[Kバイト]	256
内部RAMサイズ[バイト]	16384
データフラッシュ・メモリ・サイズ[Kバイト]	16

- ［内部 ROM サイズ [K バイト]］

エミュレーションする内部 ROM サイズを表示します（単位：K バイト）。

このプロパティ値を変更することはできません。

- ［内部 RAM サイズ [バイト]］

エミュレーションする内部 RAM サイズを表示します（単位：K バイト）。

このプロパティ値を変更することはできません。

- ［データフラッシュ・メモリ・サイズ [K バイト]］

選択しているマイクロコントローラのデータフラッシュ・メモリ領域のサイズを表示します（単位：K バイト）。

選択しているマイクロコントローラがデータフラッシュ・メモリ非搭載品の場合は，［0］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。

(b)［クロック］

このカテゴリでは，クロックに関する設定を行います。

図 2—22 ［クロック］カテゴリ【E1】

クロック	
メイン・クロック周波数[MHz]	内蔵クロックを使用する
サブ・クロック周波数[kHz]	内蔵クロックを使用する
モニタ・クロック	システム

- ［メイン・クロック周波数 [MHz]］

メイン・クロックの周波数を指定します。

ドロップダウン・リストによる選択か，または 0.001 ～ 99.999（単位：MHz）の範囲の数値を直接入力で指定します。

なお，X1/X2 発振の場合は，クロック周波数を指定してください。

また，PLL 搭載品で外部クロック発振の場合は，発振器／発振子の周波数（PLL 設定前）を指定してください。

ドロップダウン・リストには，次の周波数が表示されます（単位：MHz）。

内蔵クロックを使用する（デフォルト），2.00，3.00，3.57，4.00，4.19，4.91，5.00，6.00，8.00，8.38，10.00，12.00，16.00，20.00

備考 メイン・クロック周波数は，E1 とホスト・マシンの通信の同期に使用します。
CPU の動作周波数を設定するものではありません。

- ［サブ・クロック周波数 [kHz]］

サブ・クロック周波数を指定します。
ドロップダウン・リストによる選択か，または 0.001 ～ 99.999（単位：kHz）の範囲の数値を直接入力で指定します。
なお，外部サブ・クロックの場合は，LIOTRM でトリミング後の周波数を指定してください。
ドロップダウン・リストには，次の周波数が表示されます（単位：kHz）。
内蔵クロックを使用する（デフォルト），32.768，38.40

備考 サブ・クロック周波数は，E1 とホスト・マシンの通信の同期に使用します。
CPU の動作周波数を設定するものではありません。

- ［モニタ・クロック］

プログラム停止中に使用するクロックを指定します。
次のドロップダウン・リストにより指定します。

システム	メイン・クロックで動作します（デフォルト）。
ユーザ	プログラムで設定されているクロックで動作します。

(c)［ターゲット・ボードとの接続］

このカテゴリでは，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 と接続中に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—23 ［ターゲット・ボードとの接続］カテゴリ【E1】

ターゲット・ボードとの接続	
通信方式	2線式(TOOL0+TOOL1)
エミュレータから電源供給をする(最大200mA)	はい
供給電圧	3.3V

- ［通信方式］

E1 がターゲット・システム上のマイクロコントローラとシリアル通信を行う際の通信方式を，次のドロップダウン・リストにより指定します。
なお，デフォルトの指定，および選択可能なポートの種類はマイクロコントローラに依存します。

1 線式 (TOOL0)	通信方式を 1 線式 (TOOL0) とします。

2 線式 (TOOL0+TOOL1)	通信方式を 2 線式 (TOOL0+TOOL1) とします。

注意 ［実行中のメモリ・アクセス］カテゴリ内［実行を一瞬停止してアクセスする］プロパティにおいて［はい］を指定している場合，このプロパティで［1 線式 (TOOL0)］を選択すると，デバッグ・ツールの応答速度が非常に遅くなります。

- ［エミュレータから電源共有をする (最大 200mA)］

E1 からターゲット・システムへ電源を供給するか否かをドロップダウン・リストにより指定します。
電源を供給する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

- ［供給電源］

このプロパティは，［エミュレータから電源共有をする (最大 200mA)］プロパティにおいて，［はい］を指定した場合のみ表示されます。
ターゲット・システムへ供給する電圧を次のドロップダウン・リストにより指定します。

3.3V（デフォルト），5.0V

(d)［フラッシュ］

このカテゴリでは，フラッシュ書き換えに関する設定を行います。

図 2—24 ［フラッシュ］カテゴリ

⊟ フラッシュ	
セキュリティID	HEX 00000000000000000000
フラッシュ書き換えを許可する	はい
ワイド・ボルテージ・モードを使用する	はい
起動時にフラッシュROMを消去する	いいえ

- ［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（オンチップ・デバッグ・セキュリティ ID）をサポートしている場合のみ表示されます。
内部 ROM，または内部フラッシュ・メモリ上のコードを読み出す際のセキュリティ ID を指定します。
直接入力により，20 桁の 16 進数（10 バイト）で指定します（デフォルトでは［00000000000000000000］が指定されます）。
なお，オンチップ・デバッグ・セキュリティ ID 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。

注意 E1 と接続中にこのプロパティを変更することはできません。

- ［フラッシュ書き換えを許可する］

フラッシュ・メモリの書き換えを許可するか否かをドロップダウン・リストにより指定します。
許可する場合は［はい］を選択してください（デフォルト）。

なお，［いいえ］を指定した場合，デバッグ・ツールからフラッシュ・メモリ領域に対する書き換え操作が一切できなくなります。

- ［ワイド・ボルテージ・モードを使用する］

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，フラッシュ書き換えのワイド・ボルテージ・モードをサポートしている場合のみ表示されます。
フラッシュ書き換えを行う際に，ワイド・ボルテージ・モードで行うかをどうかをドロップダウン・リストにより指定します。
ワイド・ボルテージ・モードで行う場合は［はい］を選択してください（デフォルト）。

注意 E1 と接続中にこのプロパティを変更することはできません。

- ［起動時にフラッシュ ROM を消去する］

デバッグ・ツールと接続する際に，フラッシュ ROM の消去を行うか否かをドロップダウン・リストにより指定します。
消去する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。
なお，このプロパティは，デバッグ・ツールと接続後，自動的に［いいえ］に設定されます。

注意 E1 と接続中にこのプロパティを変更することはできません。

(2)［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，デバッグ・ツールの基本設定を行います。

(a)［メモリ］
(b)［実行中のメモリ・アクセス］
(c)［ブレーク］
(d)［入力信号のマスク］

(a) ［メモリ］

このカテゴリでは，メモリに関する設定を行います。

図 2—25 ［メモリ］カテゴリ【E1】

⊟ メモリ	
⊟ メモリ・マッピング	[10]
⊞ [0]	内部ROM領域
⊞ [1]	ノン・マップ領域
⊞ [2]	データフラッシュ領域
⊞ [3]	ノン・マップ領域
⊞ [4]	SFR領域
⊞ [5]	ノン・マップ領域
⊞ [6]	ミラー領域
⊞ [7]	内部RAM領域
⊞ [8]	レジスタ領域
⊞ [9]	SFR領域
メモリ書き込み時にベリファイを行う	はい

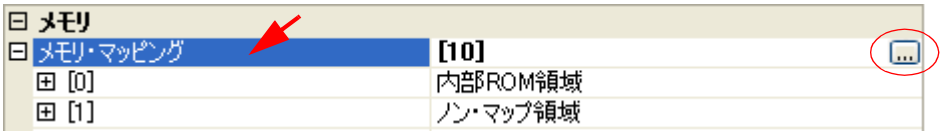
- ［メモリ・マッピング］

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。

このパネル上でマッピング値を変更することはできません。メモリ・マッピングを追加する必要がある場合は，［メモリ・マッピング］プロパティを選択することで設定欄右端に表示される［...］ボタンのクリックによりオープンするメモリ・マッピング ダイアログで行います。

設定方法についての詳細は，メモリ・マッピング ダイアログの項を参照してください。

図 2—26 メモリ・マッピング ダイアログのオープン



注意 デバッグ・ツールと未接続の場合，ユーザにより追加されたメモリ・マッピング領域のみが表示対象となります。

デバッグ・ツールと接続することにより（「2. 4. 1 CubeSuite+ にデバッグ・ツールを接続する」参照），各メモリ種別ごとの詳細表示を行います。

- ［メモリ書き込み時にベリファイを行う］

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ドロップダウン・リストにより指定します。

ベリファイを行う場合は［はい］を選択してください（デフォルト）。

(b)［実行中のメモリ・アクセス］

このカテゴリでは，プログラム実行中におけるメモリ・アクセスに関する設定を行います。

このカテゴリ内の設定は，リアルタイム表示更新機能を使用する場合に必要となります。リアルタイム表示更新機能についての詳細は，「(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—27 ［実行中のメモリ・アクセス］カテゴリ【E1】

⊟ 実行中のメモリ・アクセス	
実行を一瞬停止してアクセスする	いいえ
実行中に表示更新を行う	はい
表示更新間隔[ms]	500

- ［実行を一瞬停止してアクセスする］

プログラム実行中に，メモリに対してアクセスを許可するか否かをドロップダウン・リストにより指定します。

アクセスを許可する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

ただし，［ターゲット・ボードとの接続］カテゴリ内［通信方式］プロパティにおいて［1 線式 (TOOL0)］を指定している場合，［はい］を選択するとデバッグ・ツールの応答速度が非常に遅くなるためメッセージを表示します。

- ［実行中に表示更新を行う］

プログラム実行中に，ウォッチ パネル／メモリ パネルの表示内容を更新するか否かをドロップダウン・リストにより指定します。

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］を指定した場合にのみ有効となります。

プログラム実行中に，ウォッチ パネル／メモリ 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。

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（デフォルトでは［500］が指定されます）。

(c)［ブレーク］

このカテゴリでは，ブレーク機能に関する設定を行います。

図 2—28 ［ブレーク］カテゴリ【E1】

⊟ ブレーク	
優先的に使用するブレーク・ポイントの種類	ソフトウエア・ブレーク
停止時にタイマ系周辺エミュレーションを停止する	いいえ
停止時にシリアル系周辺エミュレーションを停止する	いいえ

- ［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，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使用するブレークポイントの種別を次のドロップダウン・リストにより指定します。

なお，ブレークポイントについての詳細は，「2. 8. 2 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。

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（デフォルト）。
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。

- ［停止時にタイマ系周辺エミュレーションを停止する］

プログラム実行停止時に，エミュレータのタイマ系周辺エミュレーション機能を停止するか否かをドロップダウン・リストにより指定します。

停止する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

なお，選択しているマイクロコントローラがオープン・ブレーク機能搭載品の場合，［はい］を指定すると，CPU 停止時にオープン・ブレークの対象端子が Hi-Z になります（［いいえ］を指定している場合では，対象端子は通常状態のままで動作します）。

- ［停止時にシリアル系周辺エミュレーションを停止する］

プログラム実行停止時に，エミュレータのシリアル系周辺エミュレーション機能を停止するか否かをドロップダウン・リストにより指定します。

停止する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

(d)［入力信号のマスク］

このカテゴリでは，入力信号のマスクに関する設定を行います。

図 2—29 ［入力信号のマスク］カテゴリ

⊟ 入力信号のマスク	
TARGET RESET 信号をマスクする	いいえ
INTERNAL RESET 信号をマスクする	いいえ

次に示す各プロパティの設定において，該当する信号をマスクする場合は［はい］を，マスクしない場合は［いいえ］をドロップダウン・リストにより指定してください（デフォルトでは，すべてのプロパティに［いいえ］が指定されます）。

- ［TARGET RESET 信号をマスクする］
- ［INTERNAL RESET 信号をマスクする］

(3)［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，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。

(4)［フック処理設定］タブ

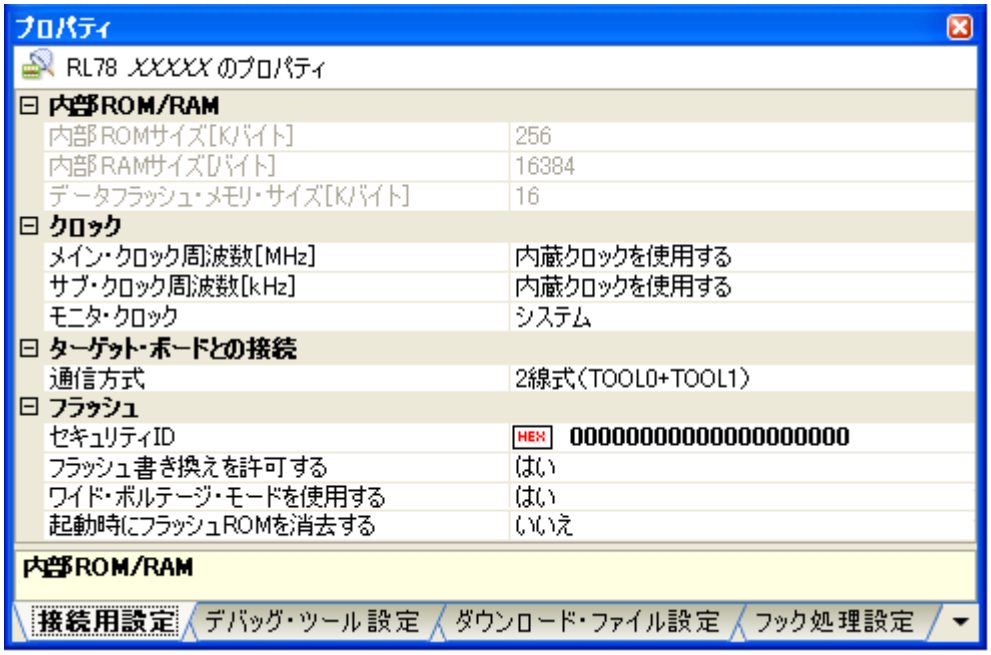
［フック処理設定］タブでは，デバッグ・ツールにフック処理の設定を行います。

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. 16 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。

図 2—30 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。

（1）［接続用設定］タブ
（2）［デバッグ・ツール設定］タブ
（3）［ダウンロード・ファイル設定］タブ
（4）［フック処理設定］タブ

（1）［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，デバッグ・ツールとの接続に関する設定を行います。

（a）［内部 ROM/RAM］
（b）［クロック］
（c）［ターゲット・ボードとの接続］
（d）［フラッシュ］

(a)［内部 ROM/RAM］

このカテゴリでは，内部 ROM/RAM に関する設定を表示します。

図 2—31 ［内部 ROM/RAM］カテゴリ【E20】

内部ROM/RAM	
内部ROMサイズ[Kバイト]	256
内部RAMサイズ[バイト]	16384
データフラッシュ・メモリ・サイズ[Kバイト]	16

- ［内部 ROM サイズ [K バイト]］

エミュレーションする内部 ROM サイズを表示します（単位：K バイト）。

このプロパティ値を変更することはできません。

- ［内部 RAM サイズ [バイト]］

エミュレーションする内部 RAM サイズを表示します（単位：K バイト）。

このプロパティ値を変更することはできません。

- ［データフラッシュ・メモリ・サイズ [K バイト]］

選択しているマイクロコントローラのデータフラッシュ・メモリ領域のサイズを表示します（単位：K バイト）。

選択しているマイクロコントローラがデータフラッシュ・メモリ非搭載品の場合は，［0］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。

(b)［クロック］

このカテゴリでは，クロックに関する設定を行います。

図 2—32 ［クロック］カテゴリ【E20】

クロック	
メイン・クロック周波数[MHz]	内蔵クロックを使用する
サブ・クロック周波数[kHz]	内蔵クロックを使用する
モニタ・クロック	システム

- ［メイン・クロック周波数 [MHz]］

メイン・クロックの周波数を指定します。

ドロップダウン・リストによる選択か，または 0.001 ～ 99.999（単位：MHz）の範囲の数値を直接入力で指定します。

なお，X1/X2 発振の場合は，クロック周波数を指定してください。

また，PLL 搭載品で外部クロック発振の場合は，発振器／発振子の周波数（PLL 設定前）を指定してください。

ドロップダウン・リストには，次の周波数が表示されます（単位：MHz）。

内蔵クロックを使用する（デフォルト），2.00，3.00，3.57，4.00，4.19，4.91，5.00，6.00，8.00，8.38，10.00，12.00，16.00，20.00

備考 メイン・クロック周波数は，E20 とホスト・マシンの通信の同期に使用します。
CPU の動作周波数を設定するものではありません。

- ［サブ・クロック周波数 [kHz]］

サブ・クロック周波数を指定します。
ドロップダウン・リストによる選択か，または 0.001 ～ 99.999（単位：kHz）の範囲の数値を直接入力で指定します。
なお，外部サブ・クロックの場合は，LIOTRM でトリミング後の周波数を指定してください。
ドロップダウン・リストには，次の周波数が表示されます（単位：kHz）。
内蔵クロックを使用する（デフォルト），32.768，38.40

備考 サブ・クロック周波数は，E20 とホスト・マシンの通信の同期に使用します。
CPU の動作周波数を設定するものではありません。

- ［モニタ・クロック］

プログラム停止中に使用するクロックを指定します。
次のドロップダウン・リストにより指定します。

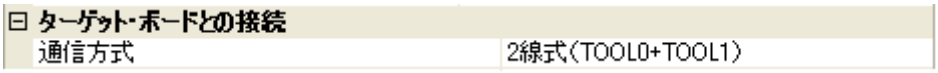
システム	メイン・クロックで動作します（デフォルト）。
ユーザ	プログラムで設定されているクロックで動作します。

(c)［ターゲット・ボードとの接続］

このカテゴリでは，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 と接続中に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—33 ［ターゲット・ボードとの接続］カテゴリ【E20】



- ［通信方式］

E20 がターゲット・システム上のマイクロコントローラとシリアル通信を行う際の通信方式を，次のドロップダウン・リストにより指定します。
なお，デフォルトの指定，および選択可能なポートの種類はマイクロコントローラに依存します。

1 線式 (TOOL0)	通信方式を 1 線式 (TOOL0) とします。
2 線式 (TOOL0+TOOL1)	通信方式を 2 線式 (TOOL0+TOOL1) とします。

注意 ［実行中のメモリ・アクセス］カテゴリ内［実行を一瞬停止してアクセスする］プロパティにおいて［はい］を指定している場合，このプロパティで［1 線式 (TOOL0)］を選択すると，デバッグ・ツールの応答速度が非常に遅くなります。

(d)［フラッシュ］

このカテゴリでは，フラッシュ書き換えに関する設定を行います。

図 2—34 ［フラッシュ］カテゴリ

⊟ フラッシュ	
セキュリティID	HEX 00000000000000000000
フラッシュ書き換えを許可する	はい
ワイド・ボルテージ・モードを使用する	はい
起動時にフラッシュROMを消去する	いいえ

- ［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（オンチップ・デバッグ・セキュリティ ID）をサポートしている場合のみ表示されます。
内部 ROM，または内部フラッシュ・メモリ上のコードを読み出す際のセキュリティ ID を指定します。
直接入力により，20 桁の 16 進数（10 バイト）で指定します（デフォルトでは［00000000000000000000］が指定されます）。
なお，オンチップ・デバッグ・セキュリティ ID 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。

注意 E20 と接続中にこのプロパティを変更することはできません。

- ［フラッシュ書き換えを許可する］

フラッシュ・メモリの書き換えを許可するか否かをドロップダウン・リストにより指定します。
許可する場合は［はい］を選択してください（デフォルト）。
なお，［いいえ］を指定した場合，デバッグ・ツールからフラッシュ・メモリ領域に対する書き換え操作が一切できなくなります。

- ［ワイド・ボルテージ・モードを使用する］

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，フラッシュ書き換えのワイド・ボルテージ・モードをサポートしている場合のみ表示されます。
フラッシュ書き換えを行う際に，ワイド・ボルテージ・モードで行うかをどうかをドロップダウン・リストにより指定します。
ワイド・ボルテージ・モードで行う場合は［はい］を選択してください（デフォルト）。

注意 E20 と接続中にこのプロパティを変更することはできません。

- ［起動時にフラッシュ ROM を消去する］

デバッグ・ツールと接続する際に，フラッシュ ROM の消去を行うか否かをドロップダウン・リストにより指定します。

消去する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

なお，このプロパティは，デバッグ・ツールと接続後，自動的に［いいえ］に設定されます。

注意　E20 と接続中にこのプロパティを変更することはできません。

(2)［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，デバッグ・ツールの基本設定を行います。

(a)［メモリ］

(b)［実行中のメモリ・アクセス］

(c)［ブレーク］

(d)［入力信号のマスク］

(a)［メモリ］

このカテゴリでは，メモリに関する設定を行います。

図 2—35 ［メモリ］カテゴリ【E20】

⊟ メモリ	
⊟ メモリ・マッピング	[10]
⊞ [0]	内部ROM領域
⊞ [1]	ノン・マップ領域
⊞ [2]	データフラッシュ領域
⊞ [3]	ノン・マップ領域
⊞ [4]	SFR領域
⊞ [5]	ノン・マップ領域
⊞ [6]	ミラー領域
⊞ [7]	内部RAM領域
⊞ [8]	レジスタ領域
⊞ [9]	SFR領域
メモリ書き込み時にベリファイを行う	はい

- ［メモリ・マッピング］

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。

このパネル上でマッピング値を変更することはできません。メモリ・マッピングを追加する必要がある場合は，［メモリ・マッピング］プロパティを選択することで設定欄右端に表示される［...］ボタンのクリックによりオープンするメモリ・マッピング ダイアログで行います。

設定方法についての詳細は，メモリ・マッピング ダイアログの項を参照してください。

図 2—36　メモリ・マッピング ダイアログのオープン

注意　デバッグ・ツールと未接続の場合，ユーザにより追加されたメモリ・マッピング領域のみが表示対象となります。
デバッグ・ツールと接続することにより（「2. 4. 1　CubeSuite+ にデバッグ・ツールを接続する」参照），各メモリ種別ごとの詳細表示を行います。

- ［メモリ書き込み時にベリファイを行う］
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ドロップダウン・リストにより指定します。
ベリファイを行う場合は［はい］を選択してください（デフォルト）。

(b)［実行中のメモリ・アクセス］

このカテゴリでは，プログラム実行中におけるメモリ・アクセスに関する設定を行います。
このカテゴリ内の設定は，リアルタイム表示更新機能を使用する場合に必要となります。リアルタイム表示更新機能についての詳細は，「(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—37　［実行中のメモリ・アクセス］カテゴリ【E20】

- ［実行を一瞬停止してアクセスする］
プログラム実行中に，メモリに対してアクセスを許可するか否かをドロップダウン・リストにより指定します。
アクセスを許可する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。
ただし，［ターゲット・ボードとの接続］カテゴリ内［通信方式］プロパティにおいて［1 線式(TOOL0)］を指定している場合，［はい］を選択するとデバッグ・ツールの応答速度が非常に遅くなるためメッセージを表示します。

- ［実行中に表示更新を行う］
プログラム実行中に，ウォッチ パネル／メモリ パネルの表示内容を更新するか否かをドロップダウン・リストにより指定します。
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］を指定した場合にのみ有効となります。

プログラム実行中に，ウォッチ パネル／メモリ 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。

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（デフォルトでは［500］が指定されます）。

(c)［ブレーク］

このカテゴリでは，ブレーク機能に関する設定を行います。

図 2—38 ［ブレーク］カテゴリ【E20】

⊟ ブレーク	
優先的に使用するブレーク・ポイントの種類	ソフトウエア・ブレーク
停止時にタイマ系周辺エミュレーションを停止する	いいえ
停止時にシリアル系周辺エミュレーションを停止する	いいえ

- ［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，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使用するブレークポイントの種別を次のドロップダウン・リストにより指定します。

なお，ブレークポイントについての詳細は，「2. 8. 2 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。

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（デフォルト）。
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。

- ［停止時にタイマ系周辺エミュレーションを停止する］

プログラム実行停止時に，エミュレータのタイマ系周辺エミュレーション機能を停止するか否かをドロップダウン・リストにより指定します。

停止する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

なお，選択しているマイクロコントローラがオープン・ブレーク機能搭載品の場合，［はい］を指定すると，CPU 停止時にオープン・ブレークの対象端子が Hi-Z になります（［いいえ］を指定している場合では，対象端子は通常状態のままで動作します）。

- ［停止時にシリアル系周辺エミュレーションを停止する］

プログラム実行停止時に，エミュレータのシリアル系周辺エミュレーション機能を停止するか否かをドロップダウン・リストにより指定します。

停止する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

(d)［入力信号のマスク］

このカテゴリでは，入力信号のマスクに関する設定を行います。

図 2—39 ［入力信号のマスク］カテゴリ

入力信号のマスク	
TARGET RESET 信号をマスクする	いいえ
INTERNAL RESET 信号をマスクする	いいえ

次に示す各プロパティの設定において，該当する信号をマスクする場合は［はい］を，マスクしない場合は［いいえ］をドロップダウン・リストにより指定してください（デフォルトでは，すべてのプロパティに［いいえ］が指定されます）。

- ［TARGET RESET 信号をマスクする］
- ［INTERNAL RESET 信号をマスクする］

(3)［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，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。

(4)［フック処理設定］タブ

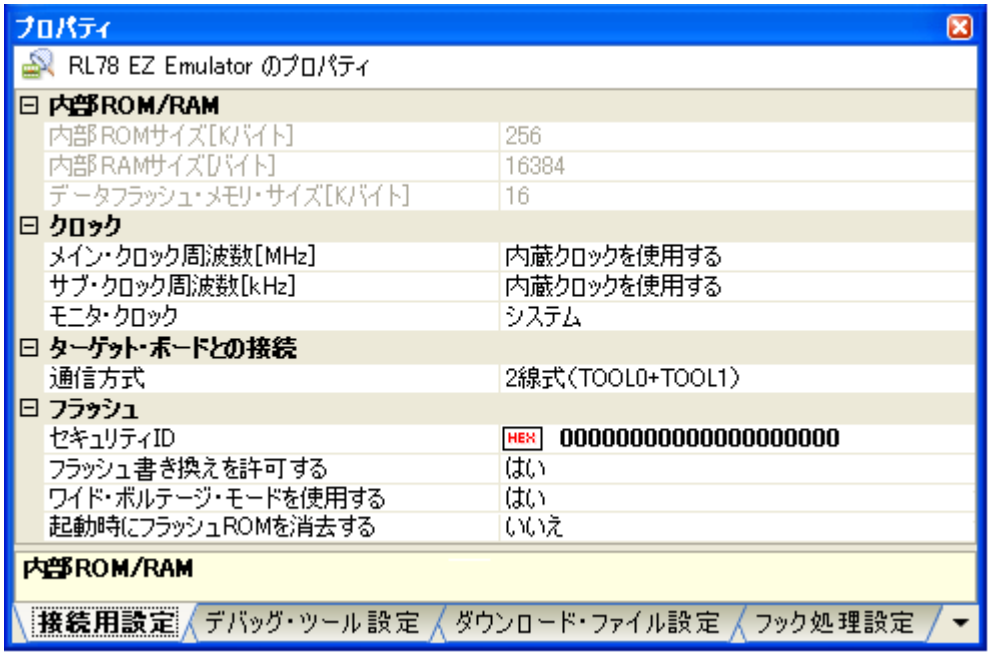
［フック処理設定］タブでは，デバッグ・ツールにフック処理の設定を行います。

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. 16 フック処理を設定する」を参照してください。

2. 3. 5 【EZ Emulator】の場合

EZ 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—40 動作環境設定【EZ 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。

(1) [接続用設定] タブ
(2) [デバッグ・ツール設定] タブ
(3) [ダウンロード・ファイル設定] タブ
(4) [フック処理設定] タブ

(1) [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，デバッグ・ツールとの接続に関する設定を行います。

(a) [内部 ROM/RAM]
(b) [クロック]
(c) [ターゲット・ボードとの接続]
(d) [フラッシュ]

(a)［内部 ROM/RAM］

このカテゴリでは，内部 ROM/RAM に関する設定を表示します。

図 2—41 ［内部 ROM/RAM］カテゴリ【EZ Emulator】

内部ROM/RAM	
内部ROMサイズ[Kバイト]	256
内部RAMサイズ[バイト]	16384
データフラッシュ・メモリ・サイズ[Kバイト]	16

- ［内部 ROM サイズ [K バイト]］

エミュレーションする内部 ROM サイズを表示します（単位：K バイト）。

このプロパティ値を変更することはできません。

- ［内部 RAM サイズ [バイト]］

エミュレーションする内部 RAM サイズを表示します（単位：K バイト）。

このプロパティ値を変更することはできません。

- ［データフラッシュ・メモリ・サイズ [K バイト]］

選択しているマイクロコントローラのデータフラッシュ・メモリ領域のサイズを表示します（単位：K バイト）。

選択しているマイクロコントローラがデータフラッシュ・メモリ非搭載品の場合は，［0］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。

(b)［クロック］

このカテゴリでは，クロックに関する設定を行います。

図 2—42 ［クロック］カテゴリ【EZ Emulator】

クロック	
メイン・クロック周波数[MHz]	内蔵クロックを使用する
サブ・クロック周波数[kHz]	内蔵クロックを使用する
モニタ・クロック	システム

- ［メイン・クロック周波数 [MHz]］

メイン・クロックの周波数を指定します。

ドロップダウン・リストによる選択か，または 0.001 ～ 99.999（単位：MHz）の範囲の数値を直接入力で指定します。

なお，X1/X2 発振の場合は，クロック周波数を指定してください。

また，PLL 搭載品で外部クロック発振の場合は，発振器／発振子の周波数（PLL 設定前）を指定してください。

ドロップダウン・リストには，次の周波数が表示されます（単位：MHz）。

内蔵クロックを使用する（デフォルト），2.00，3.00，3.57，4.00，4.19，4.91，5.00，6.00，8.00，8.38，10.00，12.00，16.00，20.00

備考 メイン・クロック周波数は，EZ Emulator とホスト・マシンの通信の同期に使用します。
CPU の動作周波数を設定するものではありません。

- ［サブ・クロック周波数 [kHz]］

サブ・クロック周波数を指定します。
ドロップダウン・リストによる選択か，または 0.001 ～ 99.999（単位：kHz）の範囲の数値を直接入力で指定します。
なお，外部サブ・クロックの場合は，LIOTRM でトリミング後の周波数を指定してください。
ドロップダウン・リストには，次の周波数が表示されます（単位：kHz）。

内蔵クロックを使用する（デフォルト），32.768，38.40

備考 サブ・クロック周波数は，EZ Emulator とホスト・マシンの通信の同期に使用します。
CPU の動作周波数を設定するものではありません。

- ［モニタ・クロック］

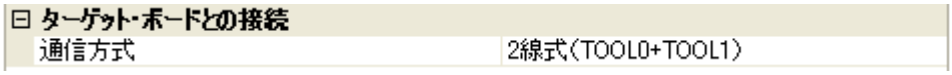
プログラム停止中に使用するクロックを指定します。
次のドロップダウン・リストにより指定します。

システム	メイン・クロックで動作します（デフォルト）。
ユーザ	プログラムで設定されているクロックで動作します。

(c)［ターゲット・ボードとの接続］

このカテゴリでは，EZ 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。

図 2—43 ［ターゲット・ボードとの接続］カテゴリ【EZ Emulator】



- ［通信方式］

EZ Emulator がターゲット・システム上のマイクロコントローラとシリアル通信を行う際の通信方式を，次のドロップダウン・リストにより指定します。
なお，デフォルトの指定，および選択可能なポートの種類はマイクロコントローラに依存します。

1 線式 (TOOL0)	通信方式を 1 線式 (TOOL0) とします。
2 線式 (TOOL0+TOOL1)	通信方式を 2 線式 (TOOL0+TOOL1) とします。

注意 1. ［実行中のメモリ・アクセス］カテゴリ内［実行を一瞬停止してアクセスする］プロパティにおいて［はい］を指定している場合，このプロパティで［1 線式 (TOOL0)］を選択すると，デバッグ・ツールの応答速度が非常に遅くなります。

2. EZ Emulator と接続中にこのプロパティを変更することはできません。

(d)［フラッシュ］

このカテゴリでは，フラッシュ書き換えに関する設定を行います。

図 2—44 ［フラッシュ］カテゴリ【EZ Emulator】

⊟ フラッシュ	
セキュリティID	HEX 00000000000000000000
フラッシュ書き換えを許可する	はい
ワイド・ボルテージ・モードを使用する	はい
起動時にフラッシュROMを消去する	いいえ

- ［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（オンチップ・デバッグ・セキュリティ ID）をサポートしている場合のみ表示されます。
内部 ROM，または内部フラッシュ・メモリ上のコードを読み出す際のセキュリティ ID を指定します。
直接入力により，20 桁の 16 進数（10 バイト）で指定します（デフォルトでは［00000000000000000000］が指定されます）。
なお，オンチップ・デバッグ・セキュリティ ID についての詳細は，EZ Emulator のユーザーズ・マニュアルを参照してください。

注意 EZ Emulator と接続中にこのプロパティを変更することはできません。

- ［フラッシュ書き換えを許可する］

フラッシュ・メモリの書き換えを許可するか否かをドロップダウン・リストにより指定します。
許可する場合は［はい］を選択してください（デフォルト）。
なお，［いいえ］を指定した場合，デバッグ・ツールからフラッシュ・メモリ領域に対する書き換え操作が一切できなくなります。

- ［ワイド・ボルテージ・モードを使用する］

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，フラッシュ書き換えのワイド・ボルテージ・モードをサポートしている場合のみ表示されます。
フラッシュ書き換えを行う際に，ワイド・ボルテージ・モードで行うかをどうかをドロップダウン・リストにより指定します。
ワイド・ボルテージ・モードで行う場合は［はい］を選択してください（デフォルト）。

注意 EZ Emulator と接続中にこのプロパティを変更することはできません。

- ［起動時にフラッシュ ROM を消去する］

デバッグ・ツールと接続する際に，フラッシュ ROM の消去を行うかするか否かをドロップダウン・リストにより指定します。

消去する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

なお，このプロパティは，デバッグ・ツールと接続後，自動的に［いいえ］に設定されます。

注意　EZ Emulator と接続中にこのプロパティを変更することはできません。

(2)［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，デバッグ・ツールの基本設定を行います。

(a)［メモリ］
(b)［実行中のメモリ・アクセス］
(c)［ブレーク］
(d)［入力信号のマスク］

(a)［メモリ］

このカテゴリでは，メモリに関する設定を行います。

図 2—45　［メモリ］カテゴリ【EZ Emulator】

⊟ メモリ	
⊟ メモリ・マッピング	[10]
⊞ [0]	内部ROM領域
⊞ [1]	ノン・マップ領域
⊞ [2]	データフラッシュ領域
⊞ [3]	ノン・マップ領域
⊞ [4]	SFR領域
⊞ [5]	ノン・マップ領域
⊞ [6]	ミラー領域
⊞ [7]	内部RAM領域
⊞ [8]	レジスタ領域
⊞ [9]	SFR領域
メモリ書き込み時にベリファイを行う	はい

- ［メモリ・マッピング］

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。

このパネル上でマッピング値を変更することはできません。メモリ・マッピングを追加する必要がある場合は，［メモリ・マッピング］プロパティを選択することで設定欄右端に表示される［...］ボタンのクリックによりオープンするメモリ・マッピング ダイアログで行います。

設定方法についての詳細は，メモリ・マッピング ダイアログの項を参照してください。

図 2—46　メモリ・マッピング ダイアログのオープン

注意　デバッグ・ツールと未接続の場合，ユーザにより追加されたメモリ・マッピング領域のみが表示対象となります。
デバッグ・ツールと接続することにより（「2. 4. 1　CubeSuite+ にデバッグ・ツールを接続する」参照），各メモリ種別ごとの詳細表示を行います。

- ［メモリ書き込み時にベリファイを行う］
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ドロップダウン・リストにより指定します。
ベリファイを行う場合は［はい］を選択してください（デフォルト）。

(b)［実行中のメモリ・アクセス］

このカテゴリでは，プログラム実行中におけるメモリ・アクセスに関する設定を行います。

このカテゴリ内の設定は，リアルタイム表示更新機能を使用する場合に必要となります。リアルタイム表示更新機能についての詳細は，「(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—47　［実行中のメモリ・アクセス］カテゴリ【EZ Emulator】

- ［実行を一瞬停止してアクセスする］
プログラム実行中に，メモリに対してアクセスを許可するか否かをドロップダウン・リストにより指定します。
アクセスを許可する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。
ただし，［ターゲット・ボードとの接続］カテゴリ内［通信方式］プロパティにおいて［1 線式(TOOL0)］を指定している場合，［はい］を選択するとデバッグ・ツールの応答速度が非常に遅くなるためメッセージを表示します。

- ［実行中に表示更新を行う］
プログラム実行中に，ウォッチ パネル／メモリ パネルの表示内容を更新するか否かをドロップダウン・リストにより指定します。
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］を指定した場合にのみ有効となります。

プログラム実行中に，ウォッチ パネル／メモリ 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。

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（デフォルトでは［500］が指定されます）。

(c)［ブレーク］

このカテゴリでは，ブレーク機能に関する設定を行います。

図 2—48 ［ブレーク］カテゴリ【EZ Emulator】

⊟ ブレーク	
優先的に使用するブレーク・ポイントの種類	ソフトウエア・ブレーク
停止時にタイマ系周辺エミュレーションを停止する	いいえ
停止時にシリアル系周辺エミュレーションを停止する	いいえ

- ［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，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使用するブレークポイントの種別を次のドロップダウン・リストにより指定します。

なお，ブレークポイントについての詳細は，「2. 8. 2 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。

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（デフォルト）。
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。

- ［停止時にタイマ系周辺エミュレーションを停止する］

プログラム実行停止時に，エミュレータのタイマ系周辺エミュレーション機能を停止するか否かをドロップダウン・リストにより指定します。

停止する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

なお，選択しているマイクロコントローラがオープン・ブレーク機能搭載品の場合，［はい］を指定すると，CPU 停止時にオープン・ブレークの対象端子が Hi-Z になります（［いいえ］を指定している場合では，対象端子は通常状態のままで動作します）。

- ［停止時にシリアル系周辺エミュレーションを停止する］

プログラム実行停止時に，エミュレータのシリアル系周辺エミュレーション機能を停止するか否かをドロップダウン・リストにより指定します。

停止する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

(d)［入力信号のマスク］

このカテゴリでは，入力信号のマスクに関する設定を行います。

図 2—49 ［入力信号のマスク］カテゴリ【EZ Emulator】

⊟ 入力信号のマスク	
TARGET RESET 信号をマスクする	いいえ
INTERNAL RESET 信号をマスクする	いいえ

次に示す各プロパティの設定において，該当する信号をマスクする場合は［はい］を，マスクしない場合は［いいえ］をドロップダウン・リストにより指定してください（デフォルトでは，すべてのプロパティに［いいえ］が指定されます）。

- ［TARGET RESET 信号をマスクする］
- ［INTERNAL RESET 信号をマスクする］

(3)［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，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。

(4)［フック処理設定］タブ

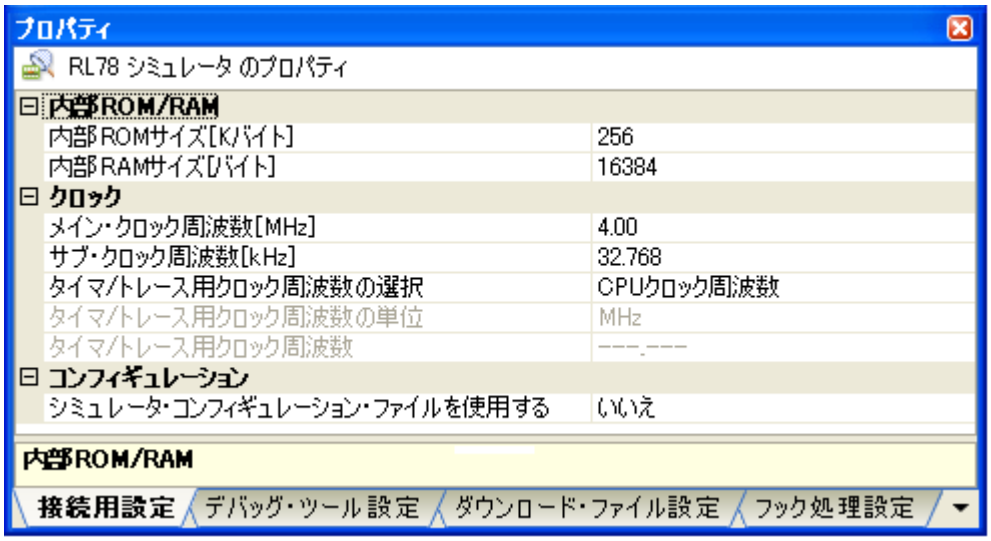
［フック処理設定］タブでは，デバッグ・ツールにフック処理の設定を行います。

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. 16 フック処理を設定する」を参照してください。

2. 3. 6 【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—50 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。

(1) [接続用設定] タブ
(2) [デバッグ・ツール設定] タブ
(3) [ダウンロード・ファイル設定] タブ
(4) [フック処理設定] タブ

備考 使用する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る場合，シミュレータ GUI を使用することができます。シミュレータ GUI についての詳細は，「2. 17 シミュレータ GUI の使用【シミュレータ】」を参照してください。

(1) [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，デバッグ・ツールとの接続に関する設定を行います。

(a) [内部 ROM/RAM]
(b) [クロック]
(c) [コンフィギュレーション]

(a) [内部 ROM/RAM]

このカテゴリでは，内部 ROM/RAM に関する設定を行います。

デフォルトで，選択しているマイクロコントローラの内部 ROM/RAM サイズが設定されます。

選択しているマイクロコントローラと同様のメモリ・マッピングでデバッグを行う場合は，このカテゴリ内の設定を変更する必要はありません。

図 2—51 ［内部 ROM/RAM］カテゴリ【シミュレータ】

⊟ 内部ROM/RAM	
内部ROMサイズ[Kバイト]	256
内部RAMサイズ[バイト]	16384

- ［内部 ROM サイズ [K バイト]］

シミュレーションする内部 ROM サイズを指定します（単位：K バイト）。

マッピングを変更後にデバッグを行う場合は，ドロップダウン・リストにより指定します。

- ［内部 RAM サイズ [バイト]］

シミュレーションする内部 RAM サイズを指定します（単位：バイト）。

マッピングを変更後にデバッグを行う場合は，ドロップダウン・リストにより指定します。

(b)［クロック］

このカテゴリでは，クロックに関する設定を行います。

図 2—52 ［クロック］カテゴリ【シミュレータ】

⊟ クロック	
メイン・クロック周波数[MHz]	4.00
サブ・クロック周波数[kHz]	32.768
タイマ/トレース用クロック周波数の選択	CPUクロック周波数
タイマ/トレース用クロック周波数の単位	MHz
タイマ/トレース用クロック周波数	---.---

- ［メイン・クロック周波数 [MHz]］

メイン・クロックの周波数を指定します。

ドロップダウン・リストによる選択か，0.001 ～ 99.999（単位：MHz）の範囲の周波数を直接入力で指定します。

ドロップダウン・リストには，次の周波数が表示されます（単位：MHz）。

2.00，3.00，3.57，4.00（デフォルト），4.19，4.91，5.00，6.00，8.00，8.38，10.00，12.00，16.00，20.00

- ［サブ・クロック周波数 [kHz]］

サブ・クロックの周波数を指定します。ドロップダウン・リストによる選択か，0.001 ～ 99.999（単位：kHz）の範囲の数値を直接入力で指定します。

ドロップダウン・リストには，次の周波数が表示されます（単位：kHz）。

32.768（デフォルト），38.40

- ［タイマ / トレース用クロック周波数の選択］

タイマ／トレース機能を使用する際のクロック周波数を指定します。

次のドロップダウン・リストより選択します。

CPU クロック周波数	CPU クロック周波数を使用します（デフォルト）。
周波数の指定	周波数を任意に指定します（下段のプロパティ項目が有効となります）。

- ［タイマ / トレース用クロック周波数の単位］

このプロパティは，［タイマ / トレース用クロック周波数の選択］プロパティにおいて［周波数の指定］を指定した場合のみ指定可能です。

タイマ／トレース用クロックの周波数の単位を指定します。

次のドロップダウン・リストより選択します。

MHz	周波数の単位を MHz とします（デフォルト）。
KHz	周波数の単位を kHz とします。

- ［タイマ / トレース用クロック周波数］

このプロパティは，［タイマ / トレース用クロック周波数の選択］プロパティでの指定により，次のように動作が異なります。

- ［周波数の指定］を指定した場合

  タイマ／トレース用クロックの周波数を指定します。

  直接入力により，1 kHz ～ 999.999 MHz の範囲となるよう数値を指定してください（デフォルトでは［4.00］が指定されます）。

  なお，周波数の単位は，［タイマ / トレース用クロック周波数の単位］プロパティでの指定に依存します。

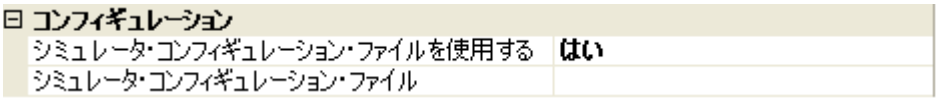
- ［CPU クロック周波数］を指定した場合

  デバッグ・ツールと切断時は［---_---］を，デバッグ・ツールと接続時は［*CPU クロック周波数*］を表示します（変更不可）。

(c)［コンフィギュレーション］

このカテゴリでは，シミュレータのカスタマイズに関する設定を行います。

図 2—53 ［コンフィギュレーション］カテゴリ



- ［シミュレータ・コンフィギュレーション・ファイルを使用する］

シミュレータに対して，ユーザ・カスタマイズ（ユーザ・モデルの追加）を行うためのシミュレータ・コンフィギュレーション・ファイルを使用するか否かをドロップダウン・リストにより指定します。

使用する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

- ［シミュレータ・コンフィギュレーション・ファイル］

このプロパティは，［シミュレータ・コンフィギュレーション・ファイルを使用する］プロパティにおいて，［はい］を指定した場合のみ表示されます。

使用するシミュレータ・コンフィギュレーション・ファイルを指定します。

ファイル名を直接入力するか，このプロパティを選択することで設定欄右端に表示される［...］ボタンのクリックによりオープンするシミュレータ・コンフィギュレーション・ファイルを選択 ダイアログ【シミュレータ】によりファイルを選択することで指定します。

注意　シミュレータと接続中にこのプロパティを変更することはできません。

(2)［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，デバッグ・ツールの基本設定を行います。

(a)［メモリ］
(b)［実行中のメモリ・アクセス］
(c)［ブレーク］
(d)［トレース］
(e)［タイマ］
(f)［カバレッジ］
(g)［シミュレータ GUI］

(a)［メモリ］

このカテゴリでは，メモリに関する設定を行います。

図 2—54　［メモリ］カテゴリ【シミュレータ】

⊟ メモリ	
⊟ メモリ・マッピング	[10]
⊞ [0]	内部ROM領域
⊞ [1]	ノン・マップ領域
⊞ [2]	データフラッシュ領域
⊞ [3]	ノン・マップ領域
⊞ [4]	SFR領域
⊞ [5]	ノン・マップ領域
⊞ [6]	ミラー領域
⊞ [7]	内部RAM領域
⊞ [8]	レジスタ領域
⊞ [9]	SFR領域

- ［メモリ・マッピング］

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。

このパネル上でマッピング値を変更することはできません。メモリ・マッピングを追加する必要がある場合は，［メモリ・マッピング］プロパティを選択することで設定欄右端に表示される［...］ボタンのクリックによりオープンするメモリ・マッピング ダイアログで行います。

設定方法についての詳細は，メモリ・マッピング ダイアログの項を参照してください。

図 2—55　メモリ・マッピング ダイアログのオープン

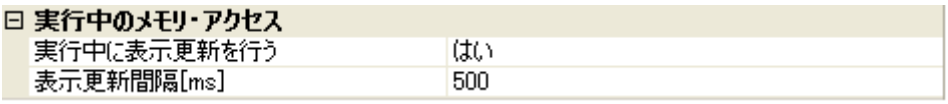
注意　デバッグ・ツールと未接続の場合，ユーザにより追加されたメモリ・マッピング領域のみが表示対象となります。
デバッグ・ツールと接続することにより（「2. 4. 1　CubeSuite+ にデバッグ・ツールを接続する」参照），各メモリ種別ごとの詳細表示を行います。

(b)［実行中のメモリ・アクセス］

このカテゴリでは，プログラム実行中におけるメモリ・アクセスに関する設定を行います。

このカテゴリ内の設定は，リアルタイム表示更新機能を使用する場合に必要となります。リアルタイム表示更新機能についての詳細は，「(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—56　［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，ウォッチ パネル／メモリ パネルの表示内容を更新するか否かをドロップダウン・リストにより指定します。

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］を指定した場合にのみ有効となります。

プログラム実行中に，ウォッチ パネル／メモリ 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。

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（デフォルトでは［500］が指定されます）。

(c)［ブレーク］

このカテゴリでは，ブレーク機能に関する設定を行います。

図 2—57 ［ブレーク］カテゴリ【シミュレータ】

⊟ ブレーク	
停止時にブレーク位置の命令を実行	いいえ

- ［停止時にブレーク位置の命令を実行］

ブレークポイントによるプログラム実行停止のタイミングを，ブレークポイントが設定されている位置の命令実行後とするか，または命令実行前とするかを指定します。

ドロップダウン・リストにより，命令実行後に停止する場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

なお，ブレークポイントについての詳細は，「2. 8. 2 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。

(d)［トレース］

このカテゴリでは，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. 11 実行履歴の収集【IECUBE】【シミュレータ】」を参照してください。

(e)［タイマ］

このカテゴリでは，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. 12 実行時間の計測」を参照してください。

図 2—58 ［タイマ］カテゴリ【シミュレータ】

⊟ タイマ	
タイマ機能を使用する	いいえ

- ［タイマ機能を使用する］

タイマ機能を使用するか否かをドロップダウン・リストにより指定します。

タイマ機能を使用する場合は［はい］を選択してくだい（デフォルトでは［いいえ］が指定されます）。

(f)［カバレッジ］

このカテゴリでは，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. 13 カバレッジの測定【IECUBE】【シミュレータ】」を参照してください。

(g)［シミュレータ GUI］

このカテゴリでは，シミュレータ GUI に関する設定を行います。

シミュレータ GUI の機能，およびこのカテゴリ内の設定についての詳細は，「2. 17 シミュレータ GUI の使用【シミュレータ】」を参照してください。

(3)［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，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。

(4)［フック処理設定］タブ

［フック処理設定］タブでは，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. 16 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，CubeSuite+ とデバッグ・ツールとの接続方法，および切断方法について説明します。

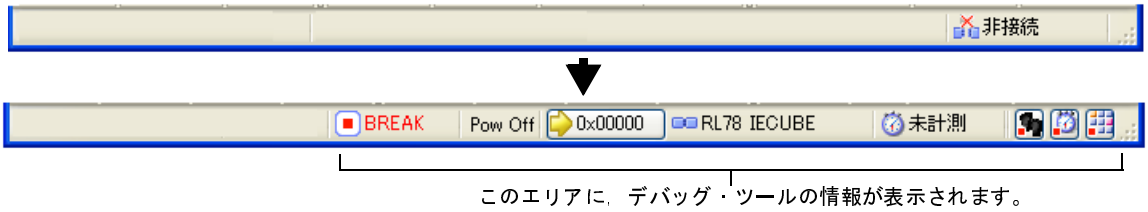
2. 4. 1　CubeSuite+ にデバッグ・ツールを接続する

［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，現在のアクティブ・プロジェクトで選択しているデバッグ・ツールとの通信を開始します。

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—59　デバッグ・ツールとの接続に成功したステータスバー



備考 1.　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。

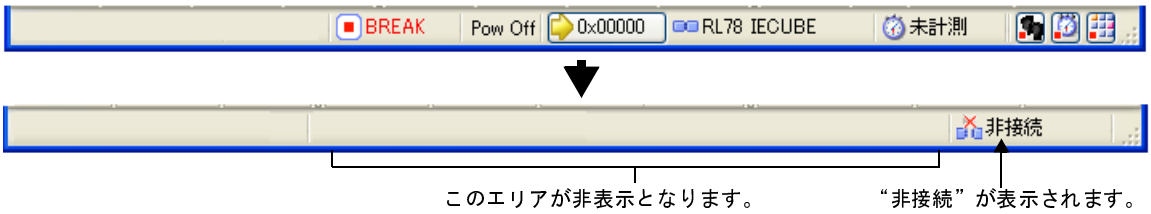
2.　【シミュレータ】
選択しているマイクロコントローラのシミュレータが周辺機能シミュレーションをサポートしている場合，デバッグ・ツールと接続すると，シミュレータ GUI ウインドウがオープンします（デフォルト）。

2. 4. 2　CubeSuite+ からデバッグ・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，現在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—60　デバッグ・ツールから切断したステータスバー



備考　デバッグ・ツールと切断すると，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（*.lmf）など）を CubeSuite+ へダウンロードする方法と，デバッグ中のメモリ内容を CubeSuite+ 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。

(1)［ダウンロード］カテゴリの設定

図 2—61 ［ダウンロード］カテゴリ

⊟ ダウンロード	
⊟ ダウンロードするファイル	[1]
⊟ [0]	Debug Build¥a.lmf
ファイル	Debug Build¥a.lmf
ファイルの種類	ロード・モジュール・ファイル
オブジェクトをダウンロードする	はい
シンボルをダウンロードする	はい
ダウンロード後にCPUをリセットする	はい
ダウンロード前にフラッシュ ROMを消去する	いいえ
イベント設定位置の自動変更方法	イベントを保留にする

(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（CubeSuite+ 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，「CubeSuite+ 起動編」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ドロップダウン・リストにより指定します。

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］

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ドロップダウン・リストにより指定します。

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（デフォルトでは［いいえ］が指定されます）。

(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度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—62 ［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］を指定している場合のみ）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ドロップダウン・リストにより指定します。

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］を指定している場合では，このプロパティで［はい］を指定すると，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］を指定した場合では，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］を指定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。

0 ～“*アドレス空間の終了アドレス*”の範囲のアドレス式を直接入力で指定してください（デフォルトでは［_main］が指定されます）。

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。

アセンブラ・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)［スタートアップ開始シンボル］

スタートアップ・ルーチンのテキスト領域（コード領域）の開始シンボルを指定します。

0 ～“*アドレス空間の終了アドレス*”の範囲のアドレス式を直接入力で指定してください（デフォルトでは［_@cstart］が指定されます）。

なお，アセンブラ・ソースの場合は，ここでの設定は不要となります。

(d)［スタートアップ終了シンボル］

スタートアップ・ルーチンのテキスト領域（コード領域）の終了シンボルを指定します。

0 ～“*アドレス空間の終了アドレス*”の範囲のアドレス式を直接入力で指定してください（デフォルトでは［_@cend］が指定されます）。

なお，アセンブラ・ソースの場合は，ここでの設定は不要となります。

注意 1. ダウンロード直後に自動的にソース・テキストを表示するためには，スタートアップ・シンボルが正しく指定されている必要があります。

2. 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］を指定してください。

(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度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。

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，SFR/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. 16　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，およびその際のダウンロード条件は変更することができます。
CubeSuite+ では，次のファイル形式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形式

ファイル形式	拡張子	備考
ロード・モジュール・フォーマット	lmf	
インテル・ヘキサ・フォーマット（標準）	hex, hxb, hxf	
インテル・ヘキサ・フォーマット（拡張）	hex, hxb, hxf	1M バイトまで可能
モトローラ S タイプ・ヘキサ・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）	hex, hxb, hxf	
拡張テクトロニクス・ヘキサ・フォーマット	hex, hxb, hxf	
バイナリ・データ・フォーマット	bin	

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—63　ダウンロード・ファイル ダイアログのオープン

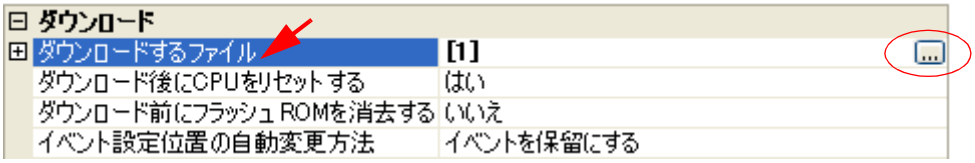
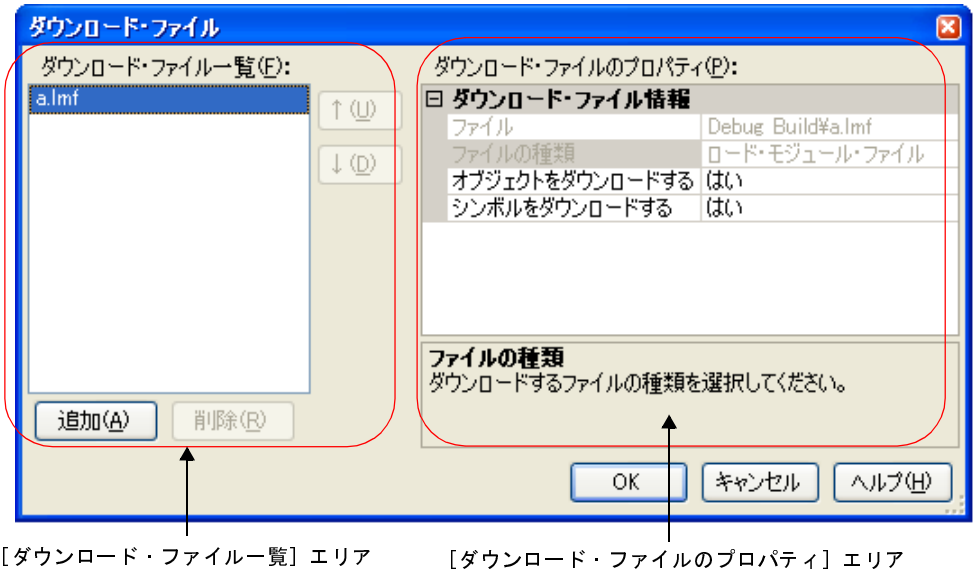


図 2—64　応用的なダウンロード方法（ダウンロード・ファイル ダイアログ）



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。

（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
（2）ダウンロード・ファイル（*.hex/*.hxb/*.hxf/*.bin）を追加する
（3）ヘキサ／バイナリ・データ・フォーマットのファイルでソース・レベル・デバッグを行う

注意　複数のロード・モジュール・ファイル（*.lmf）をダウンロードすることはできません。

（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

ダウンロードするロード・モジュール・ファイル（*.lmf）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。

（a）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。

（b）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・ファイルの現在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指定します。		
	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指定します注。		
	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。

注 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。

(c)［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) ダウンロード・ファイル（*.hex/*.hxb/*.hxf/*.bin）を追加する

ダウンロードするファイル（ヘキサ・フォーマット（*.hex/*.hxb/*.hxf）／バイナリ・データ・フォーマット（*.bin））を追加する場合は，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。

注意 複数のロード・モジュール・ファイル（*.lmf）をダウンロードすることはできません。

(a)［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。

(b) 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（ヘキサ・フォーマット（*.hex, *.hxb, *.hxf）／バイナリ・データ・フォーマット（*.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形式」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，［ヘキサ・ファイル］，または［バイナリ・データ・ファイル］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	次のいずれか - ロード・モジュール・ファイル - ヘキサ・ファイル - バイナリ・データ・ファイル
オフセット	この項目は，ダウンロードするファイルがヘキサ・フォーマット（*.hex, *.hxb, *.hxf）の場合のみ表示されます。 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。	
	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 ～ 0xFFFFFF の 16 進数
開始アドレス	この項目は，ダウンロードするファイルがバイナリ・データ・フォーマット（*.bin）の場合のみ表示されます。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。	
	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 ～ 0xFFFFFF 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。

(c) 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。

(d)［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）。

(3) ヘキサ／バイナリ・データ・フォーマットのファイルでソース・レベル・デバッグを行う

ヘキサ・フォーマット（*.hex/*.hxb/*.hxf），またはバイナリ・データ・フォーマット（*.bin）のファイルをダウンロードの対象ファイルと指定している場合でも，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。

(a)［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。

(b) 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	ダウンロードするヘキサ・フォーマット（*.hex/*.hxb/*.hxf），またはバイナリ・データ・フォーマット（*.bin）のファイルを作成する元となったロード・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］を指定します（デフォルト）。
オブジェクトをダウンロードする	［いいえ］を指定します。
シンボルをダウンロードする	［はい］を指定します（デフォルト）。

(c)［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）。

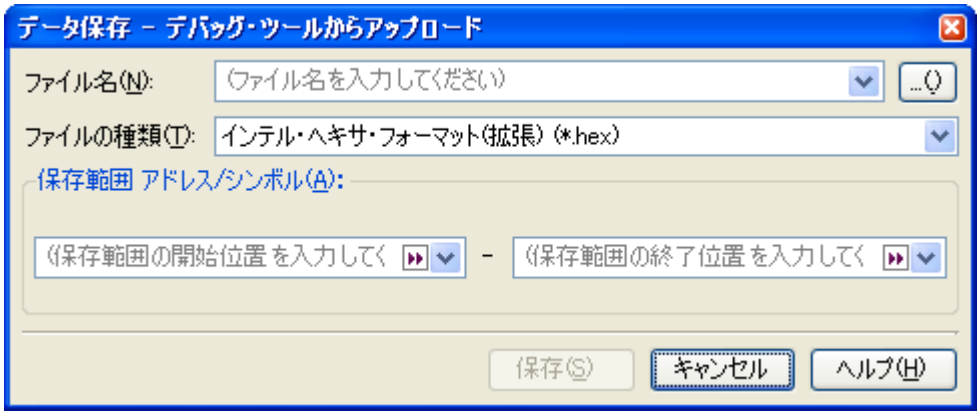
2. 5. 3 アップロードを実行する

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—65 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。

表 2—2 アップロード可能なファイル形式

リスト表示	形式
インテル・ヘキサ・フォーマット（拡張）(*.hex)	ヘキサ・フォーマット
モトローラ・ヘキサ・フォーマット（S0, S2, S8 - 24 ビット）(*.hex)	ヘキサ・フォーマット
拡張テクトロニクス・ヘキサ・フォーマット (*.hex)	ヘキサ・フォーマット
バイナリ・データ (*.bin)	バイナリ・データ・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）。

(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（「2. 7. 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（「2. 13. 2 カバレッジ測定結果を表示する」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. 7. 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（「2. 13. 2 カバレッジ測定結果を表示する」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（*.lmf）をダウンロードする必要がありますが，ヘキサ・フォーマット（*.hex, *.hxb, *.hxf），またはバイナリ・データ・フォーマット（*.bin）のファイル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（「(3) ヘキサ／バイナリ・データ・フォーマット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

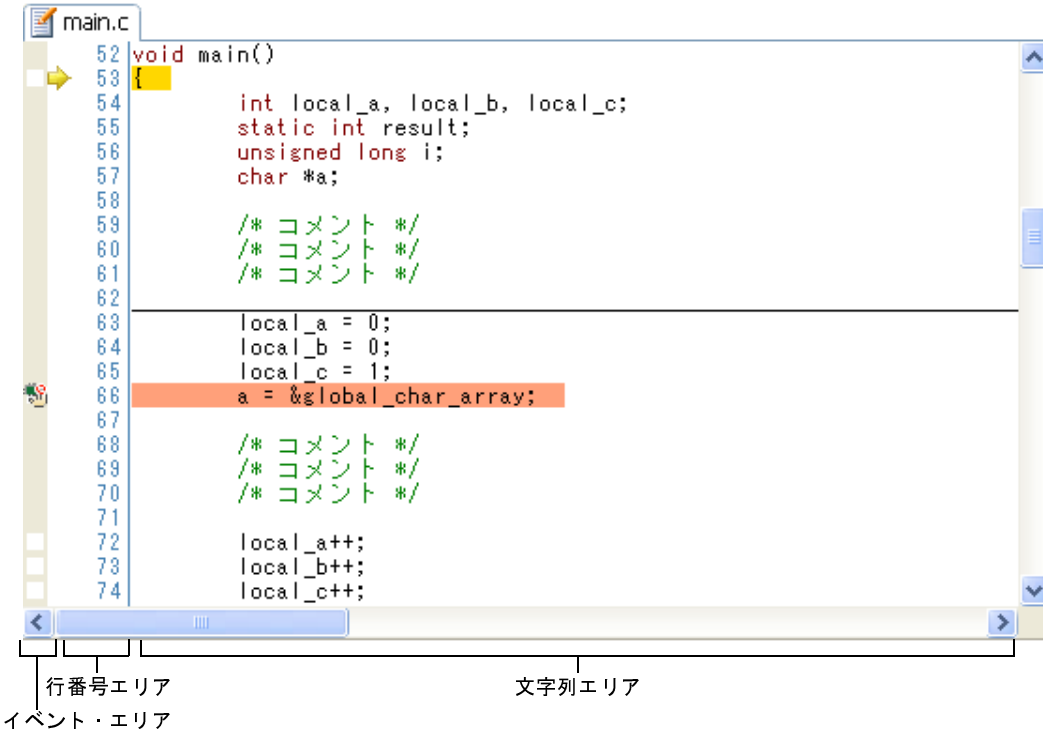
ソース・ファイルの表示は，次のエディタ パネルで行います。

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，エディタ パネルの項を参照してください。

図 2—66　ソース・ファイルの表示（エディタ パネル）



ここでは，次の操作方法について説明します。

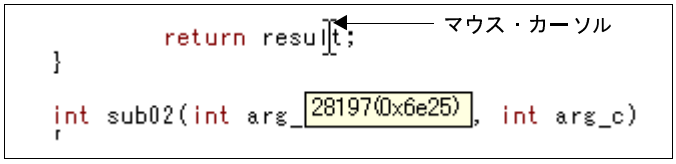
（1）変数値を表示する
（2）文字列を検索する
（3）指定行へ移動する
（4）関数へジャンプする
（5）タグ・ジャンプする

（1）変数値を表示する

ソース・テキスト上の変数にマウス・カーソルを重ねることにより，変数値をポップアップ表示します。

この際の変数値の表示形式は，変数値の型に依存します（「表 A—9　ウォッチ式の表示形式（デフォルト）」と同等）。

図 2—67 マウス・カーソルによる変数のポップアップ表示（エディタ パネル）



（2）文字列を検索する

ソース・テキスト内の文字列の検索は，ツールバーの ボタンを選択することによりオープンする検索・置換 ダイアログで行います。この際に，エディタ パネル上で範囲選択した状態で上記操作を行った場合は，選択している範囲内で検索を行います。

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—68 ソース・テキスト内の文字列検索（検索・置換 ダイアログ）

（a）［検索する文字列］の指定

検索する文字列を入力します。

デフォルトでは，エディタ パネル上のキャレット位置の単語（変数／関数）が表示されます。

変更する場合は，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）。

（b）［検索場所］の指定

ドロップダウン・リストより，［現在のパネル（*ファイル名*）］を選択します。

ただし，エディタ パネル上で範囲選択している箇所から検索を行いたい場合は，［選択範囲］を選択します。

（c）［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，指定した検索場所でアドレスの小さい方向に検索を行い，検索結果箇所をエディタ 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，指定した検索場所でアドレスの大きい方向に検索を行い，検索結果箇所をエディタ パネル上で選択状態にします。

備考 1. ［オプション］ボタンをクリックすることにより，ワイルド・カードの使用／大文字と小文字の区別／単語単位の検索などを指定することができます。

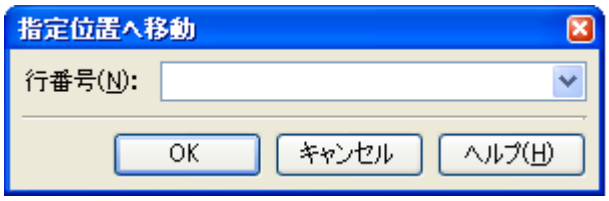
2. 検索・置換 ダイアログでは，［一括検索］／［クイック置換］／［一括置換］タブを選択することにより，様々な検索／置換操作を行うことができます。

(3) 指定行へ移動する

ソース・テキスト上の指定行への移動は，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—69 ソース・テキストの指定行へ移動（指定位置へ移動 ダイアログ）



(a)［行番号］の指定

キャレットを移動したい行番号を指定します。

テキスト・ボックスに直接入力するか（10 進数固定），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）。

(b)［OK］ボタンのクリック

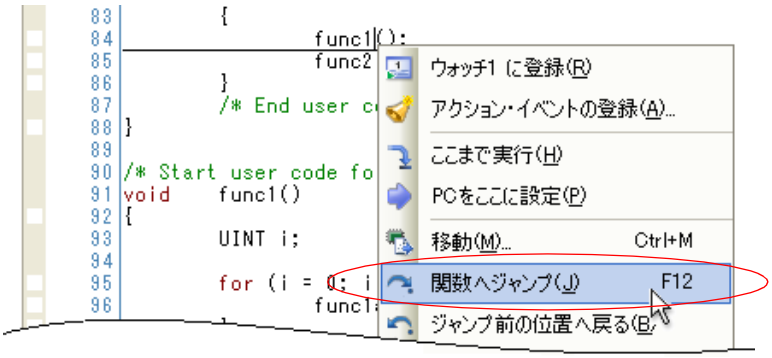
指定した行番号へキャレットを移動します。

(4) 関数へジャンプする

現在選択している文字列，またはキャレット位置の単語を関数名と判断し，該当する関数へジャンプすることができます（対象関数内で最初の実行可能行へジャンプ）。

ソース・テキスト上で，対象関数にキャレットを移動したのち，コンテキスト・メニューの［関数へジャンプ］を選択してください。

図 2—70 関数へジャンプ



なお，この機能は，使用するビルド・ツールに依存して次の条件を満たしている場合のみ有効となります。

(a) CA78K0R の場合

- 対象がアクティブ・プロジェクト内の関数[注1]である
- アクティブ・プロジェクトに指定されているプロジェクトの種類が“アプリケーション”である
- シンボル情報を持つファイル[注2]が［ダウンロードするファイル］に指定されている
  ただし，デバッグ・ツールと切断している場合は，［ダウンロードするファイル］の 1 番目に指定されている

(b) 外部ビルド・ツールの場合

- 対象がアクティブ・プロジェクト内の関数[注1]である
- エディタ パネルにフォーカスがある
- シンボル情報を持つファイル[注2]が［ダウンロードするファイル］に指定されている
  ただし，デバッグ・ツールと切断している場合は，［ダウンロードするファイル］の 1 番目にに指定されている

注 1. デバッグ・ツールと切断している場合は，スタティック関数へのジャンプは不可

2. ヘキサ・ファイルの場合，シンボル情報をダウンロードする設定が必要（「(3) ヘキサ／バイナリ・データ・フォーマットのファイルでソース・レベル・デバッグを行う」参照）。

注意 1 行に複数のステートメントを記述している場合，不正な箇所へジャンプすることがあります。

備考 単語の判断は，現在のビルド・ツールに依存します。

(5) タグ・ジャンプする

現在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，該当ファイルを新たなエディタ パネルにオープンし，該当行／桁へジャンプすることができます（すでにオープンしている場合は，該当エディタ パネルにジャンプ）。

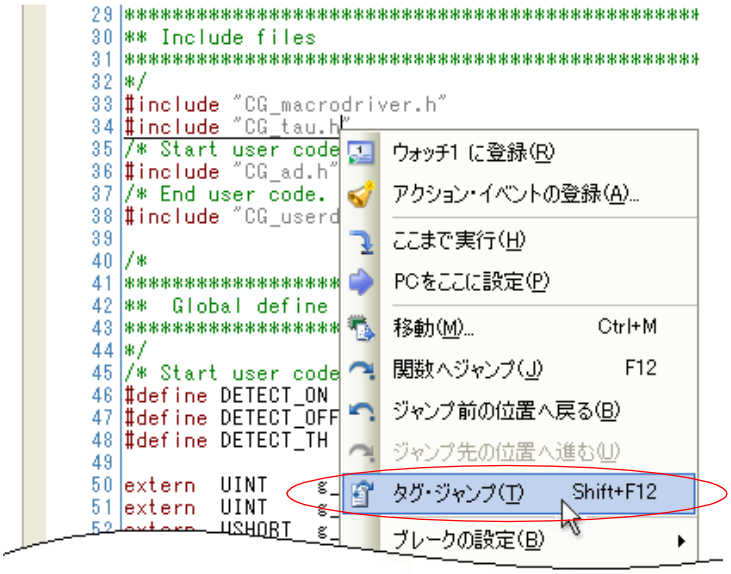
ソース・テキスト上で，対象行にキャレットを移動したのち，コンテキスト・メニューの［タグ・ジャンプ］を選択してください。

タグ・ジャンプの動作は次のとおりです。

表 2—3 タグ・ジャンプの動作

文字列の例	動作
C:¥work¥src.c	ファイル“C:¥work¥src.c”の先頭行にジャンプ
Tmp¥src.c	ファイル“.Tmp¥src.c”の先頭行にジャンプ
C:¥work¥src.c(10)	ファイル“C:¥work¥src.c”の 10 行目にジャンプ
C:¥"work sub¥src.c"(10)	ファイル“C:¥work sub¥src.c”の 10 行目にジャンプ
C:¥work¥src.c(10,5)	ファイル“C:¥work¥src.c”の 10 行 5 桁目にジャンプ

図 2—71 タグ・ジャンプ



備考 1. 大文字／小文字の区別は行いません。

2. 相対パスによる指定の場合は，ファイルが登録されているプロジェクト・フォルダを基点とします。ただし，プロジェクトに属さない場合は，アクティブ・プロジェクトを基点としてます。

3. パス指定（パス／ファイル名）にスペースを含む場合は，"" で括られている必要があります。

2. 6. 2　逆アセンブル結果を表示する

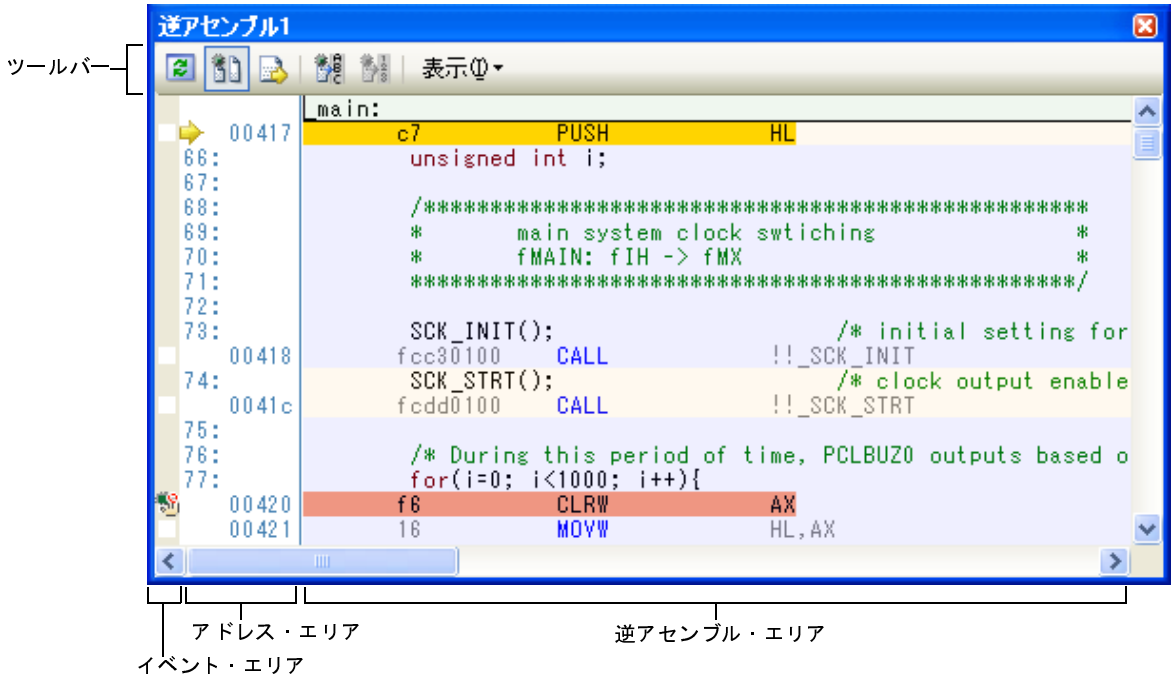
ダウンロードしたプログラム（メモリ内容）の逆アセンブル結果（逆アセンブル・テキスト）の表示は，次の逆アセンブル パネルで行います。

［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—72　逆アセンブル結果の表示（逆アセンブル パネル）



ここでは，次の操作方法について説明します。

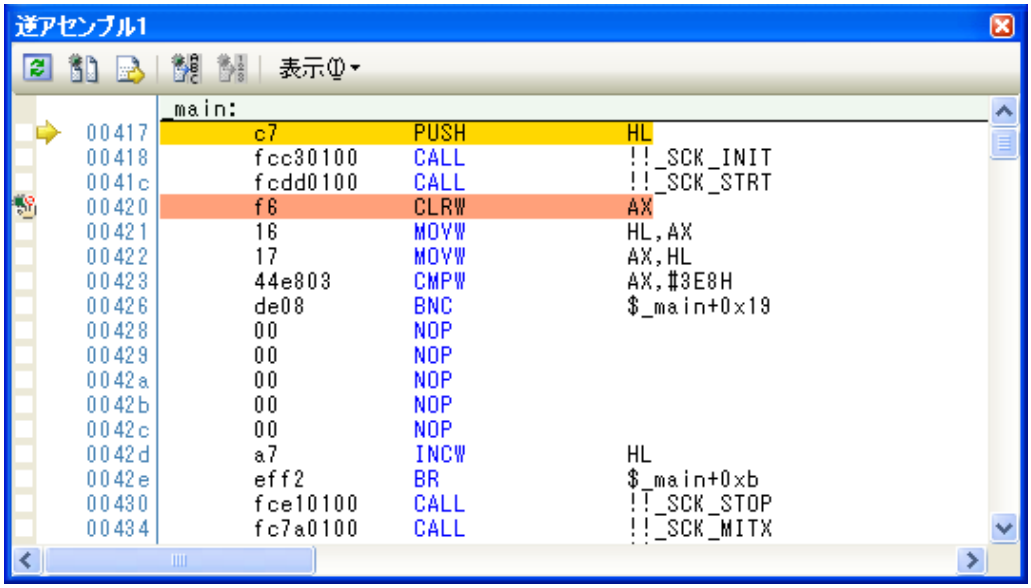
（1）表示モードを変更する
（2）表示形式を変更する
（3）指定アドレスへ移動する
（4）シンボル定義箇所へ移動する
（5）逆アセンブル結果の表示内容を保存する

（1）表示モードを変更する

逆アセンブル結果は，デフォルトで混合表示モード（逆アセンブル・テキストとソース・テキストを混合表示）で表示されます。

ツールバーの ボタン（トグル）のクリックにより，ソース・テキストを表示／非表示にすることができます。

図 2—73　ソース・テキストを非表示にした場合の表示例



(2) 表示形式を変更する

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

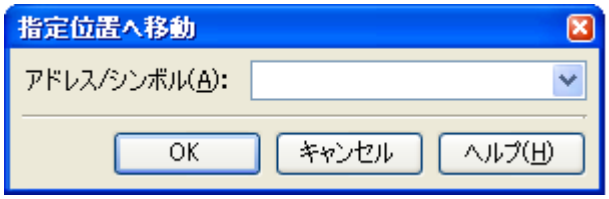
表示	表示形式を変更する次のボタンを表示します。
	ラベルのオフセット値を表示します。アドレスにラベルが定義されていない場合，一番近いラベルからのオフセット値を表示します。
	アドレス値を“シンボル＋オフセット値”で表示します（デフォルト）。 ただし，アドレス値にシンボルが定義されている場合は，シンボルのみを表示します。
	レジスタ名を機能名称で表示します（デフォルト）。
	レジスタ名を絶対名称で表示します。

(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—74　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）



(a)［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，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）。

(b)［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。

(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。

(5) 逆アセンブル結果の表示内容を保存する

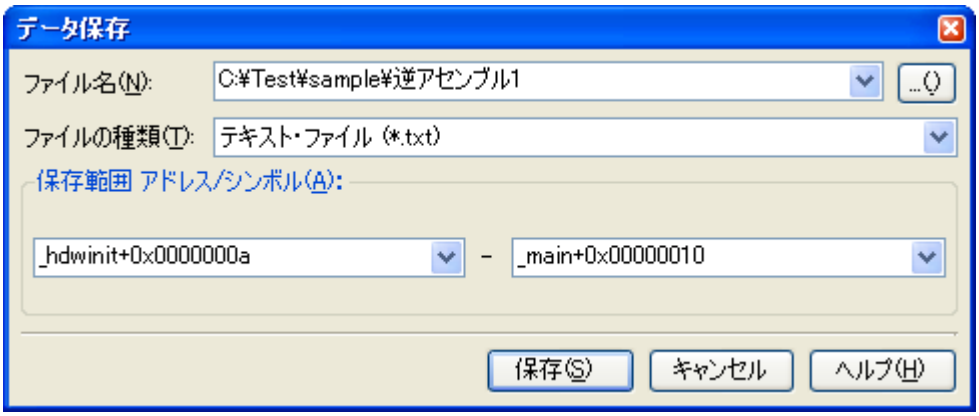
逆アセンブル結果の内容をテキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。

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—75 逆アセンブル・データの保存（データ保存 ダイアログ）



(a)［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。

(b) [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。

(c) [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。

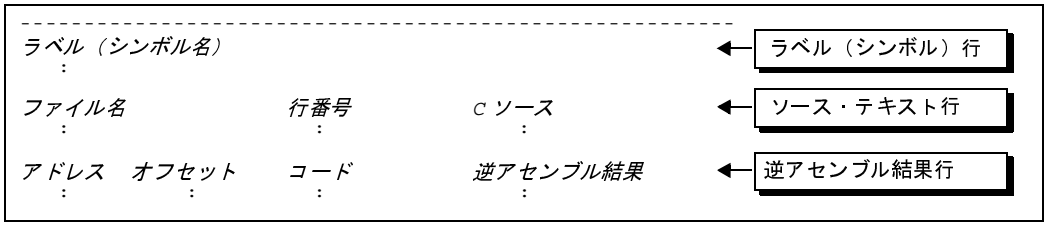
それぞれのテキスト・ボックスに 16 進数の数値／アドレス式を直接入力するか，またはドロップダウン・リストより入力履歴項目を選択します（最大履歴数：10 個）。

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。

(d) [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—76 逆アセンブル・データ保存の際の出力イメージ



備考 [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。

2. 6. 3 他の処理と平行してビルドを実行する

CubeSuite+ 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）。

(1) 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／アセンブラ・ソース・ファイル／ヘッダ・ファイル／リンク・ディレクティブ・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

(2) 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／アセンブラ・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／アセンブラ・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。

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。

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。

（1）命令を編集する
（2）命令コードを編集する

（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。

（a）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。

（b）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。

（c）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. 2 行目の MOVW 命令（4 バイト命令）を DEC 命令（1 バイト命令）で上書きした場合

編集前	0461CF	ADDW AX,#0CF61H
	CBF820FE	MOVW SP,#0FE20H
	FC8E1800	CALL !!_funcA
	53C0	MOV B, #0C0H
編集後	0461CF	ADDW AX,#0CF61H
	93	DEC B
	00	NOP
	00	NOP
	00	NOP
	FC8E1800	CALL !!_funcA
	53C0	MOV B, #0C0H

2. 1行目のADDW命令（3バイト命令）をMOVW命令（4バイト命令）で上書きした場合

編集前	0461CF ADDW AX,#0CF61H CBF820FE MOVW SP,#0FE20H FC8E1800 CALL !!_funcA 53C0 MOV B, #0C0H
編集後	CBF820FE MOVW SP, #0FE20H 00 NOP 00 NOP 00 NOP FC8E1800 CALL !!_funcA 53C0 MOV B, #0C0H

（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。

（a）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。

（b）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。

（c）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 プログラムの実行

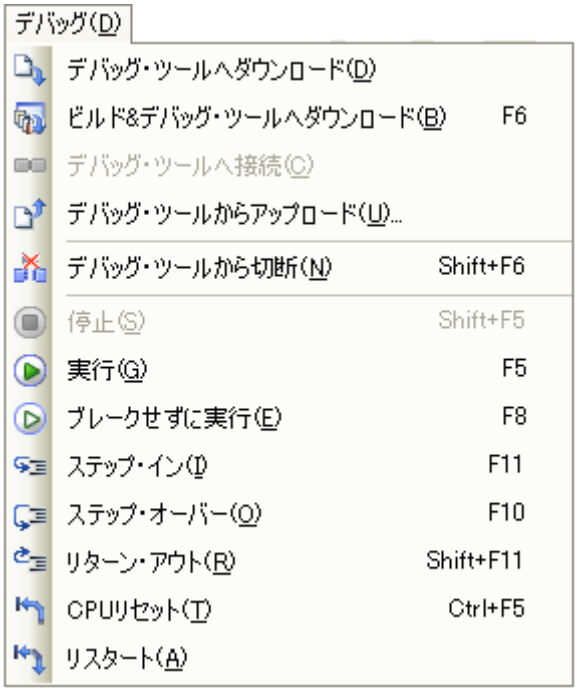
この節では，プログラムの実行方法について説明します。

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。

図 2—77 デバッグ・ツールバー

図 2—78 ［デバッグ］メニュー



2. 7. 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 CPU リセット後に，SFR/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. 16 フック処理を設定する」参照）。

2. 7. 2　プログラムを実行する

プログラムを実行する操作方法には次の種類があります。
デバッグの目的に応じて，操作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. 8　プログラムの停止（ブレーク）」を参照してください。

（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
（2）現在のアドレスから実行する
（3）PC 値を変更してから実行する

備考　プログラムの実行直前に，SFR/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. 16　フック処理を設定する」参照）。

（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. 8. 1　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. 8. 2　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. 8. 3　変数 /SFR へのアクセスで停止する」参照）
- フェイルセーフ・ブレークの発生（「2. 8. 4　不正な実行を検出して停止する【IECUBE】」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）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。

（a）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. 8. 1　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. 8. 2　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. 8. 3　変数 /SFR へのアクセスで停止する」参照）
- フェイルセーフ・ブレークの発生（「2. 8. 4　不正な実行を検出して停止する【IECUBE】」参照）
- その他のブレーク要因の発生

(b) 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. 8. 1 プログラムの実行を手動で停止する」参照）
- フェイルセーフ・ブレークの発生（「2. 8. 4 不正な実行を検出して停止する【IECUBE】」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。

(c) 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. 8. 1 プログラムの実行を手動で停止する」参照）
- フェイルセーフ・ブレークの発生（「2. 8. 4 不正な実行を検出して停止する【IECUBE】」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。

(3) PC 値を変更してから実行する

カレント PC 値を任意の箇所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) 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. 7. 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，次に示すように現在どのパネルにフォーカスがあるかで自動的に決定されます。

- 逆アセンブル パネル以外にフォーカスがある場合： ソース・レベル単位によるステップ実行[注]
- 逆アセンブル パネルにフォーカスがある場合： 命令レベル単位によるステップ実行

注 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，命令レベル単位のステップ実行となります。

ステップ実行には，次の種類があります。

（1）関数内にステップ・インする（ステップ・イン実行）
（2）関数をステップ・オーバする（ステップ・オーバ実行）
（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。

2. 【シミュレータ】以外
ステップ実行中は，割り込みが禁止されます。また，フェイルセーフ・ブレーク【IECUBE】を発生しません。
【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。

3. 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。

（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。

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。

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) 関数をステップ・オーバする（ステップ・オーバ実行）

CALL/CALLT/CALLF 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（CALL/CALLT/CALLF 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，CALL/CALLT/CALLF 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。

(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。

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。

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。

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。

5. 関数リターン直後の位置からリターン・アウト実行はできません。

6. 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. 8　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。

備考　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. 8. 1　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。

2. 8. 2　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。

ブレークポイントは，マウスのワン・クリックで設定することができます。

ブレークポイントを設定するためには，あらかじめ使用するブレークポイントの種別，および動作の設定を行う必要があります。

ここでは，次の操作方法について説明します。

（1）使用するブレークポイントの種別／動作を設定する
（2）ブレークポイントを設定する
（3）ブレークポイントを削除する

（1）使用するブレークポイントの種別／動作を設定する

使用するブレークポイントの種別／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。

なお，設定内容は，使用するデバッグ・ツールにより異なります。

（a）【シミュレータ】以外の場合
（b）【シミュレータ】の場合

（a）【シミュレータ】以外の場合

図 2—79　［ブレーク］カテゴリ【IECUBE】

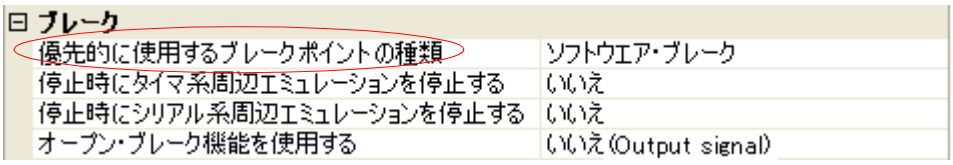


図 2—80 ［ブレーク］カテゴリ【E1】【E20】【EZ Emulator】

⊟ ブレーク	
優先的に使用するブレーク・ポイントの種類	ソフトウエア・ブレーク
停止時にタイマ系周辺エミュレーションを停止する	いいえ
停止時にシリアル系周辺エミュレーションを停止する	いいえ

［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，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指定します。

ただし，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（「(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。

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。

ソフトウエア・ブレーク	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ます（デフォルト）。 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。
ハードウエア・ブレーク	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停止させます注。 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。

注 ハードウエア・ブレークには，指定したアドレスの命令実行前にブレークする“実行前ブレーク”と，命令実行後にブレークする“実行後ブレーク”があり，これらはそれぞれデバッグ・ツールの資源を用いて実現している機能です。CubeSuite+ では，ハードウエア・ブレークのブレークポイントを設定する際，まず“実行前ブレーク”の資源を使用し，資源がなくなり次第，“実行後ブレーク”の資源を使用します（「(1) 有効イベント数の制限」参照）。そのため，ユーザが実行前／実行後のどちらかを選択することはできません。

(b)【シミュレータ】の場合

図 2—81 ［ブレーク］カテゴリ【シミュレータ】

⊟ ブレーク	
停止時にブレーク位置の命令を実行	いいえ

［停止時にブレーク位置の命令を実行］プロパティで，ブレークポイントによるプログラム実行停止のタイミングを，指定したアドレスの命令実行前（実行前ブレーク）とするか，または命令実行後（実行後ブレーク）とするかを指定します。

命令実行前にブレークする場合は［いいえ］を，命令実行後にブレークする場合は［はい］を選択してください（デフォルトでは［いいえ］が指定されます）。

なお，設定したブレークポイントは，すべてハードウエア・ブレーク・イベントとして扱われます。

注意 ［はい］を指定した場合，現在設定されているアクション・イベントは，すべてハードウエア・ブレーク・イベントとして動作します（「2. 14 プログラム内へのアクションの設定」参照）。

(2) 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。

パネルのイベント・エリアにおいて，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。

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。

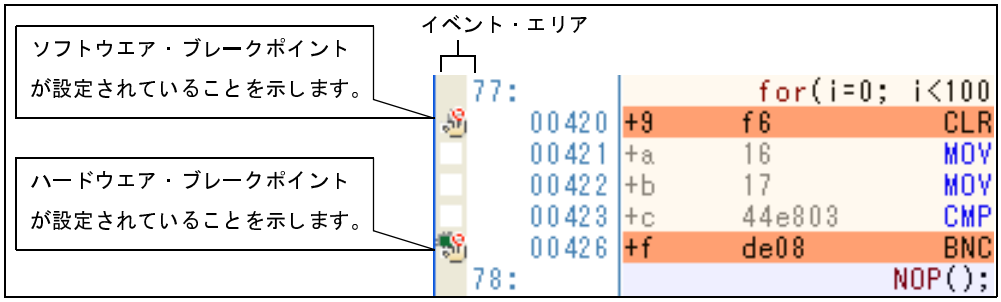
ブレークポイントが設定されると，設定した箇所に次のイベント・マークが表示され，ソース・テキスト行／逆アセンブル・テキスト行が強調表示されます。

また，対象アドレスにブレーク・イベント（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）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. 15　イベントの管理」参照）。

表 2—4　ブレークポイントのイベント・マーク

種別	イベント・マーク
ソフトウエア・ブレーク （【シミュレータ】以外）	
ハードウエア・ブレーク	

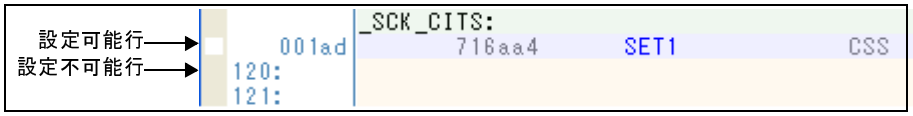
図 2—82　ブレークポイントの設定例



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. 15. 6 イベント設定に関する留意事項」も参照してください。

2.　データフラッシュ・メモリ領域にソフトウエア・ブレークを設定することはできません。

3.　ブレークポイントは，次図に示すように，イベント・エリアにおいて背景色がグレー表示の行（その行が対応するアドレスに変換不可であることを示す）では設定することはできません。



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. 15. 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. 【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。

3. 【シミュレータ】以外

次に示す操作により，「(1) 使用するブレークポイントの種別／動作を設定する」の指定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。

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

(3) 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 パネル上において，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. 8. 3 変数 /SFR 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 SFR 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 /SFR 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 /SFR 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。
リード／ライト	指定した変数 /SFR 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 【IECUBE】【E1】【E20】【EZ Emulator】

DMA によるアクセスも対象となります。

ここでは，次の操作方法について説明します。

(1) 変数 /SFR へのブレーク・イベントを設定する

(2) 変数 /SFR へのブレーク・イベントを削除する

(1) 変数 /SFR へのブレーク・イベントを設定する

変数，または SFR 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 1. 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. 15. 6 イベント設定に関する留意事項」も参照してください。

2. 32 ビット（4 バイト）の変数に対しては，ここで説明するアクセス系のブレーク・イベントの設定はできません。

また，16 ビット（2 バイト）の変数に対する 1 バイトでのアクセスの場合，そのアクセスを検出することはできません。

(a) ソース・テキスト／逆アセンブル・テキスト上の変数 /SFR にブレーク・イベントを設定する場合

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 SFR を選択したのち，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。

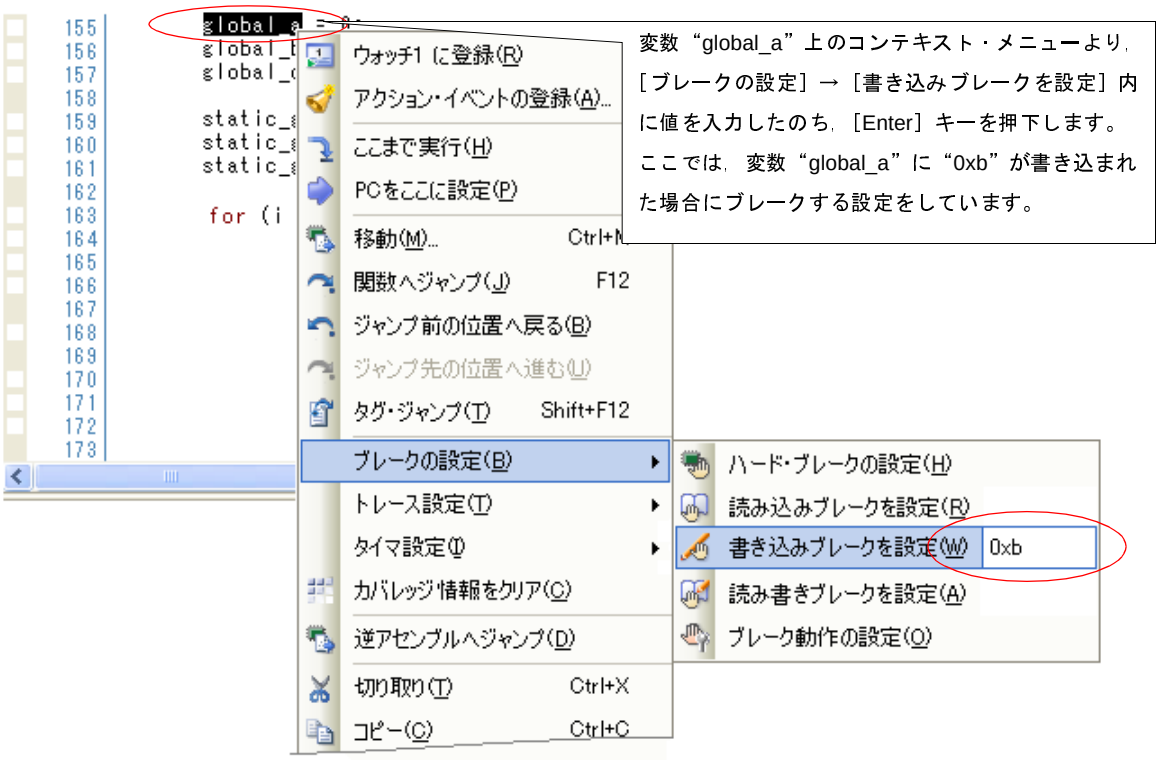
なお，この操作を行うことにより，対象変数，または SFR にブレーク・イベント（アクセス系）が設定されたとみなされ，イベント パネルで管理されます（「2. 15 イベントの管理」参照）。

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］キーを押下します。 この際に，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を行った場合にブレークします。
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］キーを押下します。 この際に，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書き込みを行った場合にブレークします。
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］キーを押下します。 この際に，メニュー内のテキスト・ボックスに値を指定した場合，指定した値で読み書き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書き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。

2. イベント・エリアにおいて背景色がグレー表示の行（その行が対応するアドレスに変換不可であることを示す）上の変数 /SFR を選択してもブレーク・イベントを設定することはできません。このブレーク・イベントを設定する際は，イベント・エリアの背景色が白色の行にある変数 /SFR を選択してください。

図 2—83　ソース・テキスト上の変数に対するブレーク・イベントの設定例



(b) 登録したウォッチ式にブレーク・イベントを設定する場合

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），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 /SFR のみとなります。

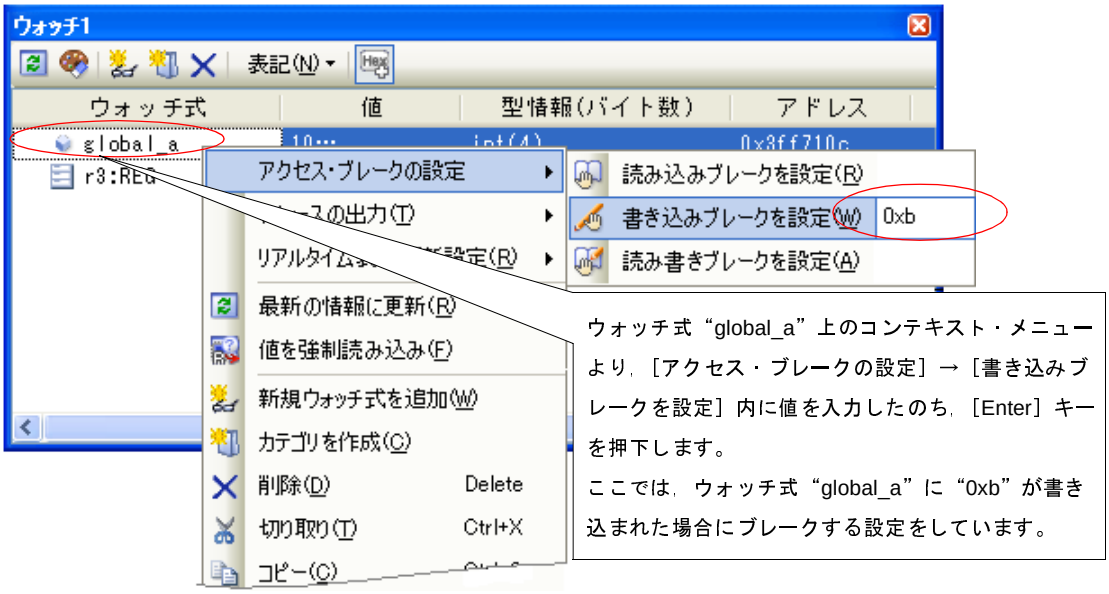
なお，この操作を行うことにより，対象ウォッチ式にブレーク・イベント（アクセス系）が設定されたとみなされ，イベント パネルで管理されます（「2. 15　イベントの管理」参照）。

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］キーを押下します。 この際に，メニュー内のテキスト・ボックスに値を指定した場合，指定された値で読み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，値に関わらず，選択しているウォッチ式に読み込みを行った場合にブレークします。
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］キーを押下します。 この際に，メニュー内のテキスト・ボックスに値を指定した場合，指定された値で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，値に関わらず，選択しているウォッチ式に書き込みを行った場合にブレークします。

アクセス種別	操作方法
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］キーを押下します。 この際に，メニュー内のテキスト・ボックスに値を指定した場合，指定された値で読み書き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，値に関わらず，選択しているウォッチ式に読み書きを行った場合にブレークします。

備考　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—84　ウォッチ式に対するブレーク・イベントの設定例



(2) 変数 /SFR へのブレーク・イベントを削除する

設定したブレーク・イベントを削除する場合は，［表示］メニュー→［イベント］の選択でオープンするイベント パネルにおいて，削除したいイベント名を選択したのち，同パネルのツールバーの ボタンをクリックします（「2. 15　イベントの管理」参照）。

2. 8. 4　不正な実行を検出して停止する【IECUBE】

内部 ROM/ 内部 RAM/SFR/ 外部メモリなどへの不正なアクセスによる暴走の可能性を検出し，プログラムの実行を強制的にブレークさせることができます（フェイルセーフ・ブレーク機能）。

この機能には，多彩なブレーク条件が用意されており，それぞれのブレーク条件の有効／無効の設定は，プロパティ パネルの［デバッグ・ツール設定］タブ上の［フェイルセーフ・ブレーク］カテゴリ内の各プロパティにより個別に行います。

注意　ステップ実行中は，フェイルセーフ・ブレーク機能は無効となります。

図 2—85　［フェイルセーフ・ブレーク］カテゴリ

⊟ フェイルセーフ・ブレーク	
フェッチ禁止領域からのフェッチ直後に停止する	はい
書き込み禁止領域への書き込み直後に停止する	はい
読み込み禁止SFR からの読み込み直後に停止する	はい
書込み禁止SFR への書き込み直後に停止する	はい
ユーザ・スタック・オーバーフロー発生直後に停止する	いいえ
ユーザ・スタック・トップ・アドレス	@STEND
ユーザ・スタック・アンダーフロー発生直後に停止する	いいえ
ユーザ・スタック・ボトム・アドレス	@STBEG
未初期化スタック・ポインタ操作直後に停止する	はい
未初期化RAMからの読み込み直後に停止する	はい
非メモリ・マッピング領域へのアクセス直後に停止する	はい
奇数アドレスへのワード・アクセス直後に停止する	はい
周辺からのフェイル・セーフ発生直後に停止する	はい
フラッシュ・イリーガル発生直後に停止する	はい

次に示す各プロパティの設定において，有効とする場合は［はい］を，無効とする場合は［いいえ］をドロップダウン・リストにより指定してください。

デフォルトでは，すべてのプロパティに［はい］が指定されます（ただし，一部を除く）。

- ［フェッチ禁止領域からのフェッチ直後に停止する］
- ［書き込み禁止領域への書き込み直後に停止する］
- ［読み込み禁止 SFR からの読み込み直後に停止する］
- ［書き込み禁止 SFR への書き込み直後に停止する］
- ［ユーザ・スタック・オーバーフロー発生直後に停止する］ **注 1**
- ［ユーザ・スタック・アンダーフロー発生直後に停止する］ **注 2**
- ［未初期化スタック・ポインタ操作直後に停止する］
- ［未初期化 RAM からの読み込み直後に停止する］
- ［非メモリ・マッピング領域へのアクセス直後に停止する］
- ［奇数アドレスへのワード・アクセス直後に停止する］
- ［周辺からのフェイル・セーフ発生直後に停止する］
- ［フラッシュ・イリーガル発生直後に停止する］

注 1. デフォルトで，［いいえ］が指定されます。
［はい］を指定した場合，下段の［ユーザ・スタック・トップ・アドレス］プロパティで，ユーザ・スタックのトップ・アドレスを設定する必要があります（デフォルトで［@STEND］が指定されます）。

2. デフォルトで，［いいえ］が指定されます。
［はい］を指定した場合，下段の［ユーザ・スタック・ボトム・アドレス］プロパティで，ユーザ・スタックのエンド・アドレス設定する必要があります（デフォルトで［@STBEG］が指定されます）。

2. 8. 5 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	使用するデバッグ・ツール		
	IECUBE	E1/E20 EZ Emulator	シミュレータ
トレース・メモリを使い切った注 1	○	－	○
トレース・ディレイ・ブレークの発生	○	－	－
タイマ・オーバ・ブレークの発生注 2	○	－	－
ノン・マップ領域へのアクセス	○	－	○
書き込み禁止領域への書き込み	○	－	○
奇数番地へのワード・アクセス	○	－	○
テンポラリ・フレークの発生	○	○	○
フラッシュ・イリーガル・ブレークの発生	○	－	－
周辺チップ機能に関するプログラムの不正動作の発生注 3	○	－	－
実行の失敗，または不明な原因	○	○	－

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

2. 「2. 12. 3 測定可能時間の範囲」参照

3. 詳細は，周辺エミュレーション・ボードに関する資料を参照

2.9　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。

2.9.1　メモリを表示／変更する

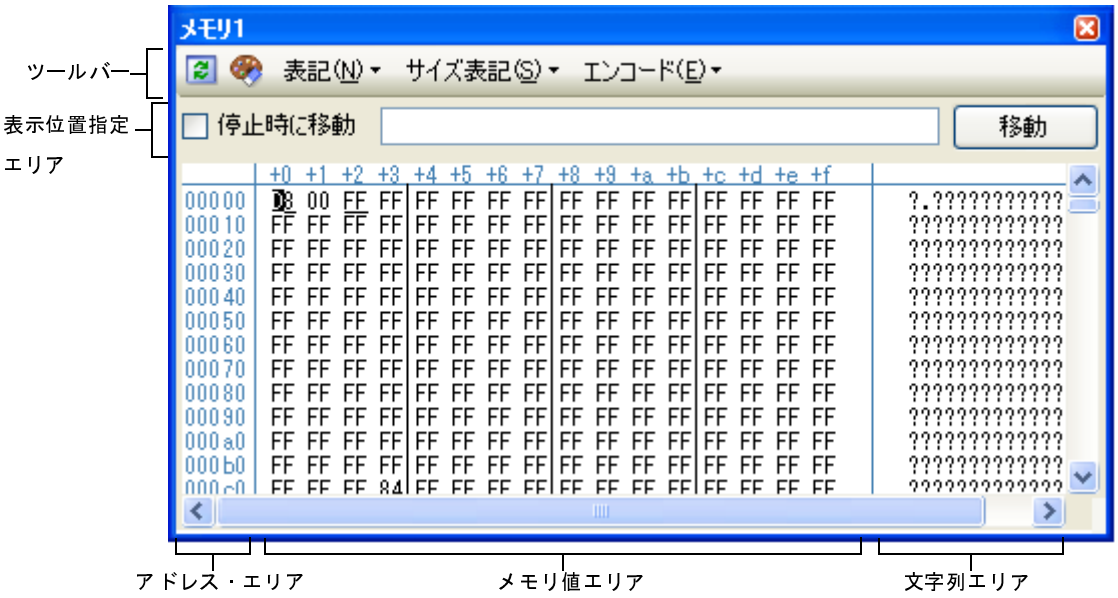
メモリの内容の表示，および値の変更は，次のメモリ パネルで行います。

［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—86　メモリの内容の表示（メモリ パネル）



ここでは，次の操作方法について説明します。

（1）表示位置を指定する

（2）値の表示形式を変更する

（3）メモリの内容を変更する

（4）プログラム実行中にメモリの内容を表示／変更する

（5）メモリの内容を検索する

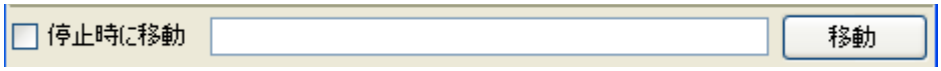
（6）メモリの内容を一括して変更（初期化）する

（7）メモリの表示内容を保存する

(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，“0”番地より表示を開始します）。

図 2—87 表示位置指定エリア（メモリ パネル）



(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。

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式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。

ただし，32 ビットで表現できる値より大きいアドレス式を指定することはできません。

(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。

備考 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) 値の表示形式を変更する

メモリ値エリア／文字列エリアの表示形式は，ツールバーの次のボタンにより，自由に変更することができます。

ただし，プログラム実行中は，ボタンは無効となります。

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。
Hex	メモリ値を 16 進数で表示します（デフォルト）。
S Dec	メモリ値を符号付き 10 進数で表示します。
U Dec	メモリ値を符号なし 10 進数で表示します。
Oct	メモリ値を 8 進数で表示します。
Bin	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	メモリ値を 4 ビット幅で表示します。
8	メモリ値を 8 ビット幅で表示します（デフォルト）。
16	メモリ値を 16 ビット幅で表示します。
32	メモリ値を 32 ビット幅で表示します。
64	メモリ値を 64 ビット幅で表示します。
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。
SJIS	文字列を ASCII コードで表示します（デフォルト）。
SJIS	文字列を Shift_JIS コードで表示します。
EUC	文字列を EUC-JP コードで表示します。
UTF	文字列を UTF-8 コードで表示します。

(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。

備考 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，「(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。

なお，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。

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。

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。

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。

(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。

備考 リアルタイム表示更新機能は，デバッグ・ツールの RRM（Real-time RAM Monitor）機能，擬似 RRM 機能，DMM（Dynamic Memory Modification）機能，および擬似 DMM 機能により実現されます。
擬似 RRM 機能／擬似 DMM 機能では，プログラムの実行を一瞬ブレークさせ，ソフトウエア・エミュレーションによる読み込み／書き込みを行います。

リアルタイム表示更新機能を使用した読み込み／書き込みが可能となる対象領域は，使用するデバッグ・ツールとプロパティ パネルの設定内容により異なります。

次に示すプロパティ パネルの［デバッグ・ツール設定］タブ上の［実行中のメモリ・アクセス］カテゴリ内の各プロパティの設定内容に基づき，リアルタイム表示更新機能の使用目的に応じ，表 2—7/ 表 2—8 の設定を行ってください。

設定記号	［実行中のメモリ・アクセス］カテゴリ内プロパティ	設定値	備考
A	［実行中に表示更新を行う］	［はい］（デフォルト）	RRM/DMM 機能の有効化
	［表示更新間隔 [ms]］	［*100 ～ 65500 の整数*］	
B	［実行を一瞬停止してアクセスする］	［はい］	擬似 RRM/ 擬似 DMM 機能の有効化（シミュレータを除く）

(a) 読み込みの場合

表 2—7 リアルタイム表示更新機能（読み込み）の対象領域

領域名	IECUBE		E1/E20 EZ Emulator		シミュレータ	
	設定	対象領域	設定	対象領域	設定	対象領域
内部 ROM	A	全領域	A+B	全領域	A	全領域
内部 RAM	A	全領域（レジスタ領域を除く）	A+B	全領域（レジスタ領域を除く）	A	全領域（レジスタ領域を除く）

<table>
<tr><th rowspan="2">領域名</th><th colspan="2">IECUBE</th><th colspan="2">E1/E20
EZ Emulator</th><th colspan="2">シミュレータ</th></tr>
<tr><th>設定</th><th>対象領域</th><th>設定</th><th>対象領域</th><th>設定</th><th>対象領域</th></tr>
<tr><td>エミュレーション・メモリ</td><td colspan="2">不可</td><td colspan="2" rowspan="2">不可</td><td rowspan="2">A</td><td rowspan="2">全領域</td></tr>
<tr><td>ターゲット・メモリ</td><td>A+B</td><td>全領域</td></tr>
<tr><td rowspan="2">CPU レジスタ</td><td>A</td><td>汎用レジスタ /PC のみ</td><td rowspan="2">A+B</td><td rowspan="2">バンク指定の汎用レジスタのみ</td><td rowspan="3">A</td><td rowspan="3">全領域
（トレーサ／タイマ動作中は不可）</td></tr>
<tr><td>A+B</td><td>全領域</td></tr>
<tr><td>SFR</td><td>A+B</td><td>全領域</td><td>A+B</td><td>全領域
（BCDADJ を除く）</td></tr>
<tr><td>データフラッシュ</td><td>A</td><td>全領域</td><td>A+B</td><td>全領域</td><td colspan="2">不可</td></tr>
</table>

（b）書き込みの場合

表 2—8　リアルタイム表示更新機能（書き込み）の対象領域

<table>
<tr><th rowspan="2">領域名</th><th colspan="2">IECUBE</th><th colspan="2">E1/E20
EZ Emulator</th><th colspan="2">シミュレータ</th></tr>
<tr><th>設定</th><th>対象領域</th><th>設定</th><th>対象領域</th><th>設定</th><th>対象領域</th></tr>
<tr><td>内部 ROM</td><td colspan="2">不可</td><td colspan="2">不可</td><td>A</td><td>全領域</td></tr>
<tr><td>内部 RAM</td><td>A</td><td>全領域</td><td>A+B</td><td>全領域
（レジスタ領域を除く）</td><td>A</td><td>全領域
（レジスタ領域を除く）</td></tr>
<tr><td>エミュレーション・メモリ</td><td colspan="2">不可</td><td colspan="2" rowspan="2">不可</td><td rowspan="2">A</td><td rowspan="2">全領域</td></tr>
<tr><td>ターゲット・メモリ</td><td rowspan="3">A+B</td><td rowspan="3">全領域</td></tr>
<tr><td>CPU レジスタ</td><td>A+B</td><td>バンク指定の汎用レジスタのみ</td><td rowspan="2">A</td><td rowspan="2">全領域
（トレーサ／タイマ動作中は不可）</td></tr>
<tr><td>SFR</td><td>A+B</td><td>全領域</td></tr>
<tr><td>データフラッシュ</td><td colspan="2">不可</td><td colspan="2">不可</td><td colspan="2">不可</td></tr>
</table>

注意 1.　リアルタイム表示更新機能を実行中に，CPU ステータスが HALT/STOP/IDLE モードに移行すると，モニタ・タイムアウト・エラーとなります。

2.　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。

3.　【IECUBE】

リアルタイム表示更新機能とソフトウエア・ブレークは，排他使用の機能です。

このため，上記の設定を行った場合，ソフトウエア・ブレークを使用することはできません。

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，「(3) メモリの内容を変更する」／「(6) ウォッチ式の内容を変更する」を参照してください。

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，次のように強調表示されます（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）。

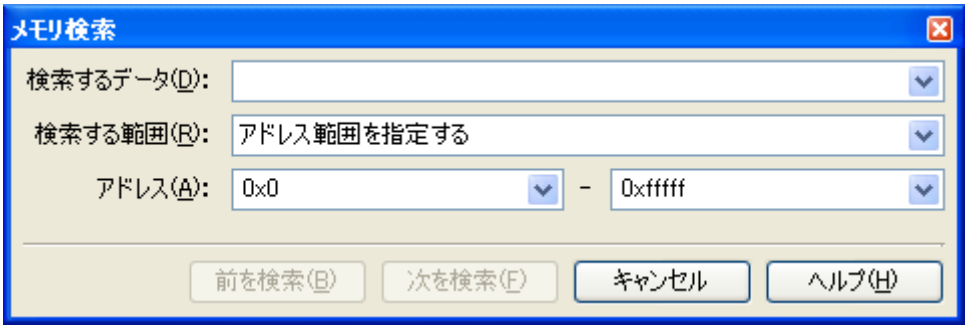
図 2—88　リアルタイム表示更新を行っているメモリ表示の例（メモリ パネル）

(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—89　メモリ内容の検索（メモリ検索 ダイアログ）



注意　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。

(a)［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。

(b)［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（ノン・マップ領域を除く）。 表示形式：*＜メモリ種別＞ ＜アドレス範囲＞ ＜サイズ＞*

(c)［アドレス］の指定

この項目は，「(b)［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，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。

備考 1. “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。

2. “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。

(d)［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。

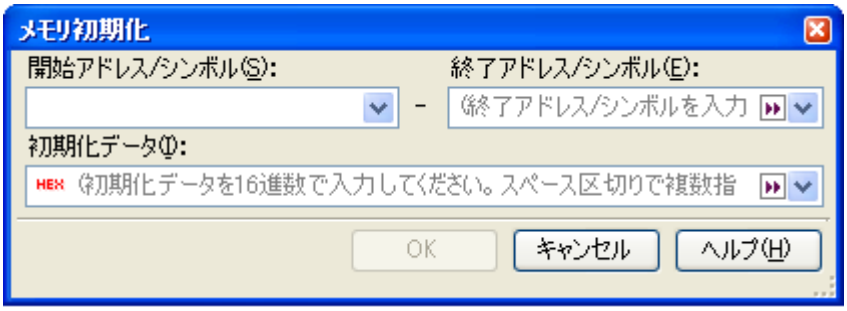
(6) メモリの内容を一括して変更（初期化）する

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—90 メモリ内容の一括変更（メモリ初期化 ダイアログ）



(a)［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。

(b)［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，次のようにリトル・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）
1	01
0 12	00 12
00 012 345	00 12 00 45 03
000 12 000345	00 00 12 45 03 00

(c)［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。

(7) メモリの表示内容を保存する

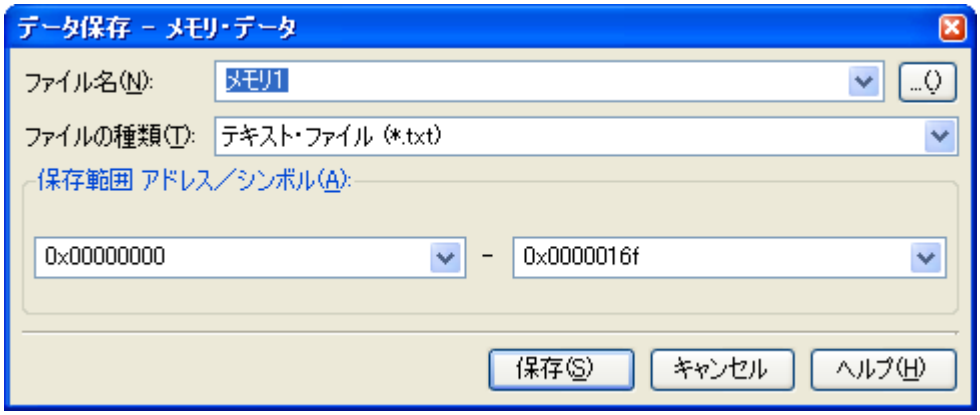
メモリの内容を範囲指定して，テキスト・ファイル（*.txt）/CSV ファイル（*.csv）に保存することができます。

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—91 メモリ・データの保存（データ保存 ダイアログ）



(a)［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。

(b)［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。

(c)［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。

(d)［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—92 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
0000 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |
0010 | 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）

```
0000,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,
0010,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,11,
```

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。

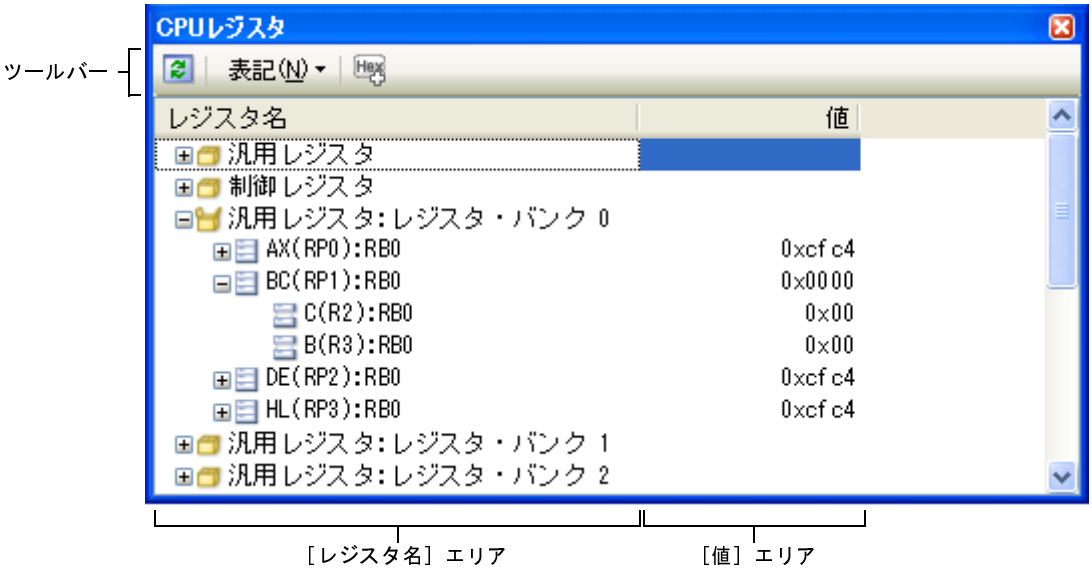
2. 9. 2 CPU レジスタを表示／変更する

CPU レジスタ（汎用レジスタ／制御レジスタ）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—93　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。

（1）値の表示形式を変更する
（2）CPU レジスタの内容を変更する
（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
（4）CPU レジスタの表示内容を保存する

（1）値の表示形式を変更する

［値］エリアの表示形式は，ツールバーの次のボタンにより，自由に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。
auto	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）。
Hex	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。
S Dec	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。
U Dec	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。
Oct	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。
Bin	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。
Asc	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。
Flo	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。
Dbl	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。
Hex	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) 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。

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。

(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. 9. 6　ウォッチ式を表示／変更する」を参照してください。

(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—94　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
      :                   :
```

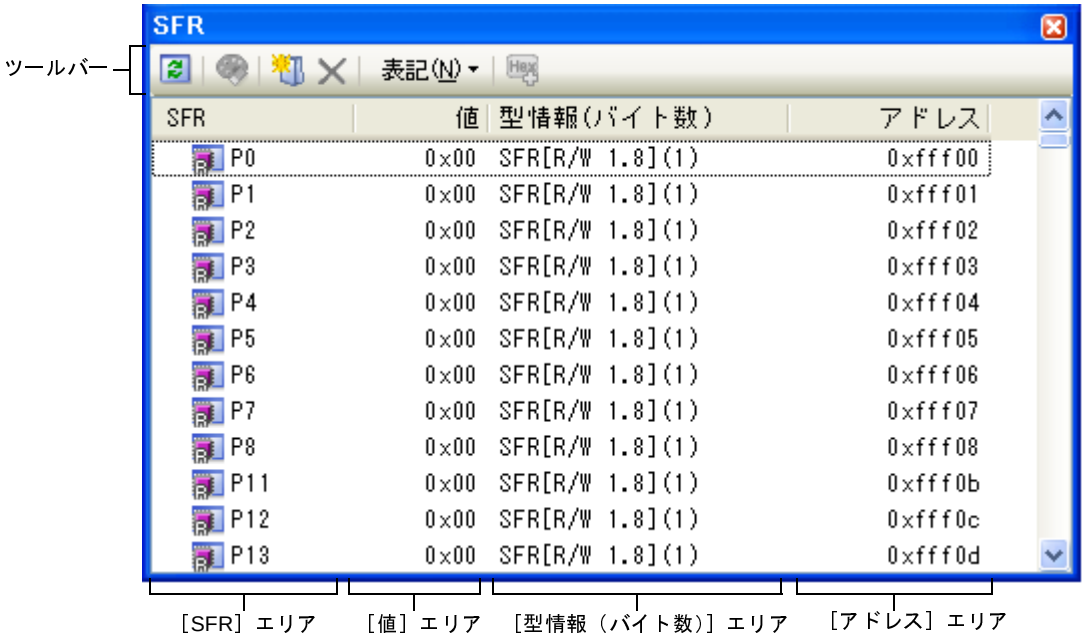
2. 9. 3 SFR を表示／変更する

SFR の内容の表示，および値の変更は，次の SFR パネルで行います。

［表示］メニュー→［SFR］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，SFR パネルの項を参照してください。

図 2—95 SFR の内容の表示（SFR パネル）



ここでは，次の操作方法について説明します。

（1）SFR を整理する

（2）値の表示形式を変更する

（3）SFR の内容を変更する

（4）プログラム実行中に SFR の内容を表示／変更する

（5）SFR の表示内容を保存する

（1）SFR を整理する

各 SFR 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。

2. SFR の追加／削除はできません。

（a）カテゴリを新規作成する場合

作成したい SFR 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。

(b) 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。

(c) 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。

(d) 表示順を変更する場合

SFR 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，SFR はカテゴリで分類されます。

また，カテゴリと SFR 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) 値の表示形式を変更する

［値］エリアの表示形式は，ツールバーの次のボタンにより，自由に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。
Hex	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）。
S Dec	選択している項目の値を符号付き 10 進数で表示します。
U Dec	選択している項目の値を符号なし 10 進数で表示します。
Oct	選択している項目の値を 8 進数で表示します。
Bin	選択している項目の値を 2 進数で表示します。
Asc	選択している項目の値を ASCII コードで表示します。
Hex	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。

(3) SFR の内容を変更する

SFR の値は，編集することができます。

［値］エリアにおいて，対象 SFR 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。

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。

2. 読み込み専用の SFR の値を変更することはできません。

なお，SFR の値には ASCII 文字による入力も可能です。

例 1. SFR 名“DMC0”の値に“0x41”を書き込んだ場合

→ DMC0 に，“0x41”が書き込まれます。

2. SFR 名“DMC0”の値に ASCII 文字“'A'”を書き込んだ場合

→ DMC0 に，“0x41”が書き込まれます。

(4) プログラム実行中に SFR の内容を表示／変更する

対象となる SFR 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 SFR 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. 9. 6　ウォッチ式を表示／変更する」を参照してください。

(5) SFR の表示内容を保存する

［ファイル］メニュー→［名前を付けて SFR 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，SFR 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 SFR の値が対象となります）。

ファイルに保存する際は，SFR 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 SFR 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—96　SFR 保存の際の出力イメージ

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

2. 9. 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. 9. 6　ウォッチ式を表示／変更する」を参照してください。

2. 9. 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。

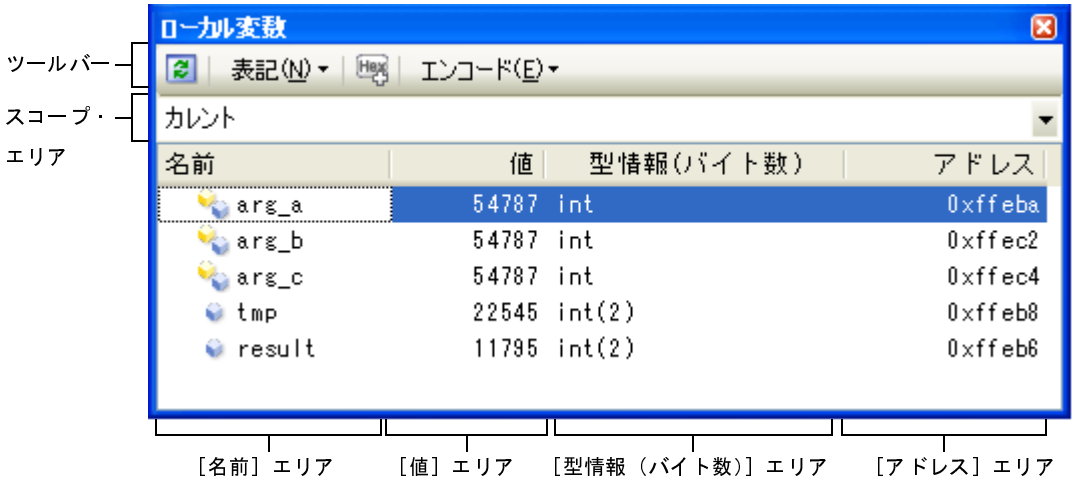
目的のローカル変数の内容を表示するためには，スコープ・エリアでスコープの選択をします。

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。

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—97　ローカル変数の内容の表示（ローカル変数 パネル）



ここでは，次の操作方法について説明します。

（1）値の表示形式を変更する
（2）ローカル変数の内容を変更する
（3）ローカル変数の表示内容を保存する

（1）値の表示形式を変更する

［値］エリアの表示形式は，ツールバーの次のボタンにより，自由に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。
auto	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）。
Hex	このパネル上の値を 16 進数で表示します。
Dec	このパネル上の値を 10 進数で表示します。
Oct	このパネル上の値を 8 進数で表示します。
Bin	このパネル上の値を 2 進数で表示します。
Dec	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）。
Hex	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。
Flo	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。
Dbl	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。
Hex	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。
Asc	文字列変数を ASCII コードで表示します（デフォルト）。
SJIS	文字列変数を Shift_JIS コードで表示します。
EUC	文字列変数を EUC-JP コードで表示します。
UTF	文字列変数を UTF-8 コードで表示します。

(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。

なお，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。

例 1. 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

2. 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

3. 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

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。

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。

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)）による値の入力も可能です。

(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—98　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. 9. 6　ウォッチ式を表示／変更する

C 言語変数，CPU レジスタ，SFR，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。

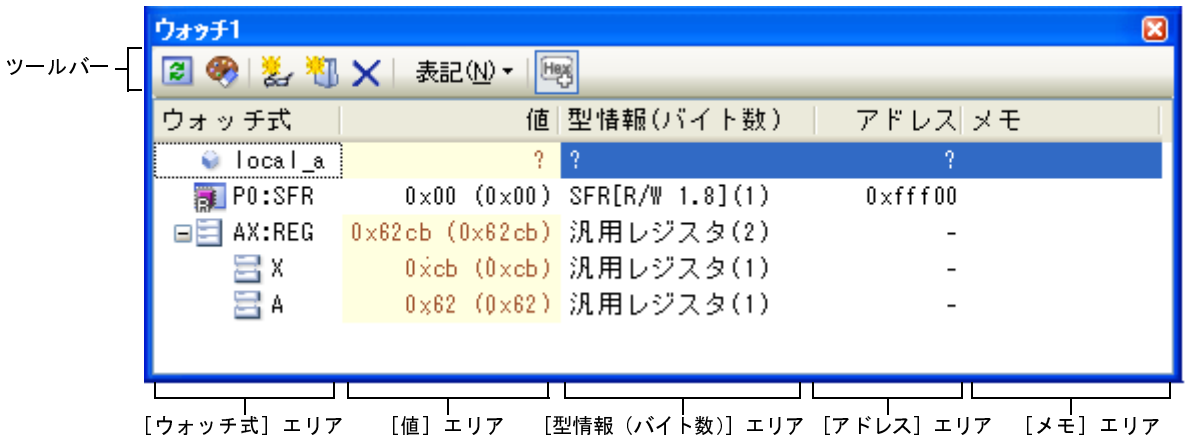
また，ウォッチ式は，プログラムが実行中の状態にあっても値の表示を逐次更新することができます（「(7) プログラム実行中にウォッチ式の内容を表示／変更する」参照）。

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—99　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。

(1) ウォッチ式を登録する
(2) 登録したウォッチ式を整理する
(3) 登録したウォッチ式を編集する
(4) ウォッチ式を削除する
(5) 値の表示形式を変更する
(6) ウォッチ式の内容を変更する
(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
(8) ウォッチ式の表示内容を保存する

(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 つのウォッチ パネルにおいて，ウォッチ式は 128 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）。

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。

2. ウォッチ式は，同名を複数登録することができます。

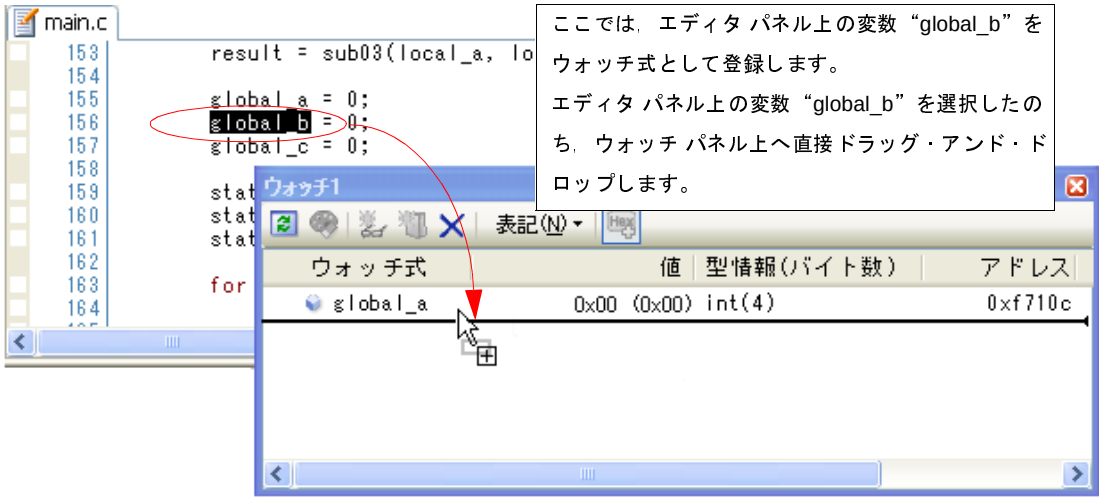
(a) 他のパネルから登録する場合

CubeSuite+ 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—4　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—100　他のパネルからウォッチ式登録する場合の例

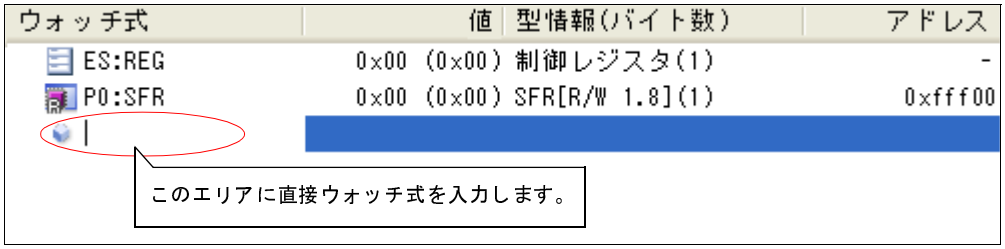


備考　ウォッチ式として登録したい対象を選択したのち，または対象文字列のいずれかにキャレットを移動したのち（対象は自動的に決定されます），コンテキスト・メニューの［ウォッチ 1 に登録］を選択することによっても同様にウォッチ式を登録することができます（ただし，ウォッチ パネル（ウォッチ 1）に限定）。

(b) 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—101 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は，次の表を参照してください。

- 「表 A—5 ウォッチ式の入力形式」
- 「表 A—6 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」
- 「表 A—7 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—8 SFR をスコープ指定してウォッチ登録した場合の扱い」

(c) 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 /SFR/ 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) 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。

2. 1 つのウォッチ パネルにおいて，カテゴリは 64 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）。

(a) 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。

(b) 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。

(c) 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。

(d) 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。

(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。

(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。

(5) 値の表示形式を変更する

[値] エリアの表示形式は，ツールバーの次のボタンにより，自由に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。
auto	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—9 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）。
Hex	選択している項目の値を 16 進数で表示します。
S Dec	選択している項目の値を符号付き 10 進数で表示します。
U Dec	選択している項目の値を符号なし 10 進数で表示します。
Oct	選択している項目の値を 8 進数で表示します。
Bin	選択している項目の値を 2 進数で表示します。
Asc	選択している項目の値を ASCII コードで表示します。
Flo	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。
Dbl	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。
Hex	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。

(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 /SFR/ 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 SFR の値を変更することもできません。

備考 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，「(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。

なお，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。

例 1. 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

2. 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

3. 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

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。

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。

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)）による値の入力も可能です。

(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，「(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。

(8) 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 SFR 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—102　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. 10 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。

CubeSuite+ が提供するコンパイラ（CA78K0R）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。

2. 10. 1 コール・スタック情報を表示する

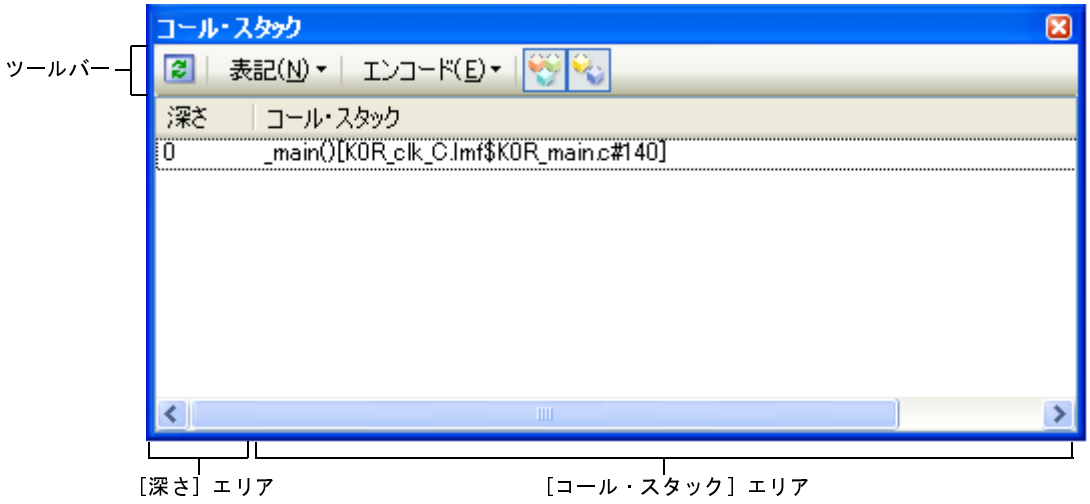
コール・スタック情報の表示は，次のコール・スタック パネルで行います。

［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—103 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。

（1）値の表示形式を変更する
（2）ソース行へジャンプする
（3）ローカル変数を表示する
（4）コール・スタック情報の表示内容を保存する

（1）値の表示形式を変更する

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

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。
auto	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）。
Hex	このパネル上の値を 16 進数で表示します。
Dec	このパネル上の値を 10 進数で表示します。
Oct	このパネル上の値を 8 進数で表示します。
Bin	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。
Asc	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）。
SJIS	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	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	このパネル上の文字列変数を UTF-8 コードで表示します。

（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，選択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）。

（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。

（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—104　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
 :                    :
```

注　信頼性の喪失したスタック・データより古い情報を取得した場合，その行の情報は“()”で括って出力されます。

2. 11　実行履歴の収集【IECUBE】【シミュレータ】

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。

注意　【E1】【E20】【EZ Emulator】

トレース機能はサポートしていません。

2. 11. 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。

（1）【IECUBE】の場合

（2）【シミュレータ】の場合

（1）【IECUBE】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—105　［トレース］カテゴリ【IECUBE】

⊟ トレース	
実行前にトレース・メモリをクリアする	はい
トレース・メモリを使い切った後の動作	トレース・メモリを上書きし実行を続ける
トレース・タイム・タグの分周率	1/1(4ns/0.3min)

（a）［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）するか否かをドロップダウン・リストにより指定します。

クリアする場合は［はい］を指定してください（デフォルト）。

備考　トレース パネル【IECUBE】【シミュレータ】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。

（b）［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・リストにより指定します。

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］を指定している場合は，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）。 [実行前にトレース・メモリをクリアする] プロパティで［いいえ］を指定している場合は，再び実行してもトレース・データの書き込みは行いません。
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。 [実行前にトレース・メモリをクリアする] プロパティで［いいえ］を指定している場合は，再び実行してもプログラムは実行せずに停止します。

(c)［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【IECUBE】【シミュレータ】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・リストにより指定します。

ドロップダウン・リストには，次の分周率が表示されます（“()”内は分解能，および外部クロックが240MHz の場合の最大測定時間を示す）。

1/1(4ns/0.3min)（デフォルト），1/2(8ns/0.6min)，1/4(17ns/1.2min)，
1/8(33ns/2.4min)，1/16(67ns/4.8min)，1/32(133ns/9.5min)，
1/64(267ns/19.1min)，1/128(533ns/38.2min)，1/256(1067ns/1.3h)，
1/512(2133ns/2.5h)，1/1024(4267ns/5.1h)，1/2048(8533ns/10.2h)，
1/4096(17067ns/20.4h)

(2)【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—106 ［トレース］カテゴリ【シミュレータ】

⊟ トレース	
トレース機能を使用する	はい
実行前にトレース・メモリをクリアする	はい
トレース・メモリを使い切った後の動作	トレース・メモリを上書きし実行を続ける
トレース・タイム・タグを積算する	いいえ
トレース・メモリ・サイズ[フレーム]	4K

(a)［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ドロップダウン・リストにより指定します。

トレース機能を使用する場合は［はい］を指定してください（デフォルトでは［いいえ］が指定されます）。

(b)［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）するか否かをドロップダウン・リストにより指定します。

クリアする場合は［はい］を指定してください（デフォルト）。

備考 トレース パネル【IECUBE】【シミュレータ】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。

(c)［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・リストにより指定します。

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］を指定している場合は，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）。 ［実行前にトレース・メモリをクリアする］プロパティで［いいえ］を指定している場合は，再び実行してもトレース・データの書き込みは行いません。
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。 ［実行前にトレース・メモリをクリアする］プロパティで［いいえ］を指定している場合は，再び実行してもプログラムは実行せずに停止します。

(d)［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ドロップダウン・リストにより指定します。

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］を指定してください（デフォルト）。

(e)［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・リストにより指定します。

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。

ドロップダウン・リストには，次のトレース・フレーム数が表示されます。

4K（デフォルト），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

2. 11. 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. 11. 5 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. 11. 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. 11. 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。

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. 15. 6 イベント設定に関する留意事項」も参照してください。

2. 【シミュレータ】
トレース動作中は，トレース開始イベント／トレース終了イベントを設定することはできません。

(1) トレース開始イベント／トレース終了イベントを設定する

エディタ パネル／逆アセンブル パネルにおいて，トレース・データの収集を開始／終了するイベントを設定します。

(a) トレース開始イベントの設定方法

トレース・データの収集を開始したい行／アドレス注にキャレットを移動したのち，コンテキスト・メニューの［トレース設定］→［トレース開始の設定］を選択します。

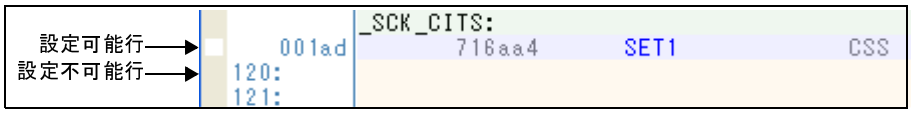
トレース開始イベントが，キャレット位置の行／アドレスに対応する先頭アドレスの命令に対して設定されます。

(b) トレース終了イベントの設定方法

トレース・データの収集を終了したい行／アドレス[注]にキャレットを移動したのち，コンテキスト・メニューの［トレース設定］→［トレース終了の設定］を選択します。

トレース終了イベントが，キャレット位置の行／アドレスに対応する先頭アドレスの命令に対して設定されます。

注　トレース開始イベント／トレース終了イベントは，次図に示すように，イベント・エリアにおいて背景色がグレー表示の行（その行が対応するアドレスに変換不可であることを示す）では設定することはできません。

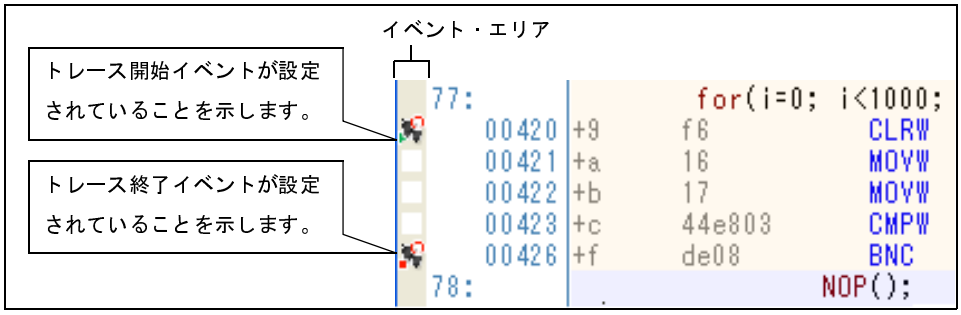


トレース開始イベント／トレース終了イベントが設定されると，設定した行／アドレスのイベント・エリアに次のイベント・マークが表示されます。また，イベント パネル上において，トレース・イベントとして１つにまとめて管理されます（トレース・イベント項目の“+”マークをクリックすることにより，設定したトレース開始イベント／トレース終了イベントの詳細情報が表示されます）。

表 2—9　トレース開始イベント／トレース終了イベント・マーク

種別	イベント・マーク
トレース開始	
トレース終了	

図 2—107　トレース開始イベント／トレース終了イベントの設定例



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）。

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。

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. 15. 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。

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) プログラムを実行する

プログラムを実行します（「2. 7 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. 11. 5 実行履歴を表示する」を参照してください。

(3) トレース開始イベント／トレース終了イベントを削除する

設定したトレース開始イベント／トレース終了イベントを削除する場合は，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。

2. 11. 4 条件を満たした場合の実行履歴を収集する

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。

(1) 変数 /SFR へのアクセスが発生したとき

ポイント・トレース・イベントを設定することにより，指定した変数，または SFR をアクセスした際に，その情報をトレース・データとして収集します。

次のいずれかの操作により，ポイント・トレース・イベントを設定してください。

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. 15. 6 イベント設定に関する留意事項」も参照してください。

2. 【IECUBE】

32 ビット（4 バイト）の変数に対しては，ポイント・トレース・イベントの設定はできません。また，16 ビット（2 バイト）の変数に対する 1 バイトでのアクセスの場合，そのアクセスを検出することはできません。

備考 【IECUBE】

DMA によるアクセスも対象となります。

(a) ソース・テキスト／逆アセンブル・テキスト上の変数 /SFR 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。

対象となる変数，または SFR を選択したのち，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。

なお，この操作を行うことにより，対象変数 /SFR 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. 15 イベントの管理」参照）。

アクセス種別	操作方法
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。

備考 カレント・スコープ内の変数が対象となります。

(b) 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，コンテキスト・メニューより次の操作を行います（「2. 9. 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 /SFR のみとなります。

なお，この操作を行うことにより，対象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. 15 イベントの管理」参照）。

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。
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。

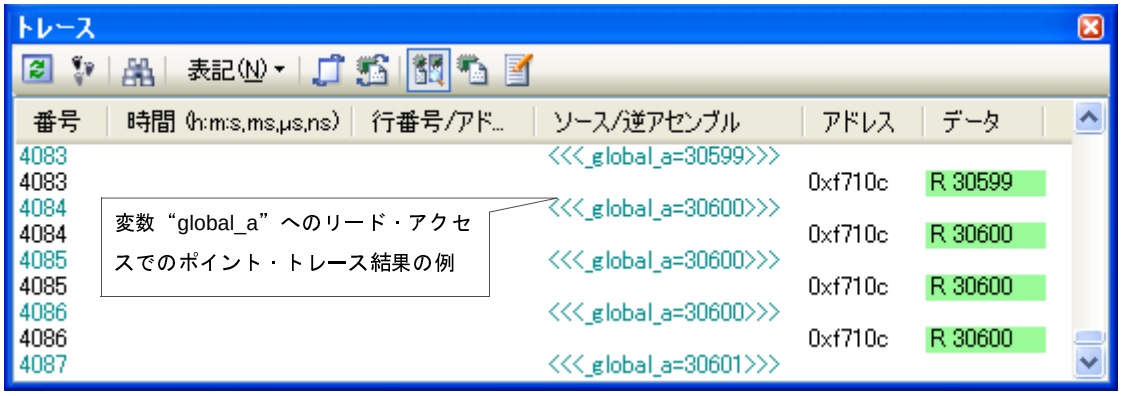
備考 カレント・スコープ内のウォッチ式が対象となります。
カレント・スコープ以外のウォッチ式を対象とする場合は，スコープ指定したウォッチ式を選択してください。

ポイント・トレース・イベントの設定が完了したのち，プログラムを実行します（「2. 7 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. 11. 5 実行履歴を表示する」を参照してください。

なお，設定したポイント・トレース・イベントを削除する場合は，［表示］メニュー→［イベント］の選択でオープンするイベント パネルにおいて，削除したいイベント名を選択したのち，同パネルのツールバーの ボタンをクリックします（「2. 15 イベントの管理」参照）。

図 2—108 ポイント・トレース・イベントの結果表示例（シミュレータを使用した場合）



2. 11. 5 実行履歴を表示する

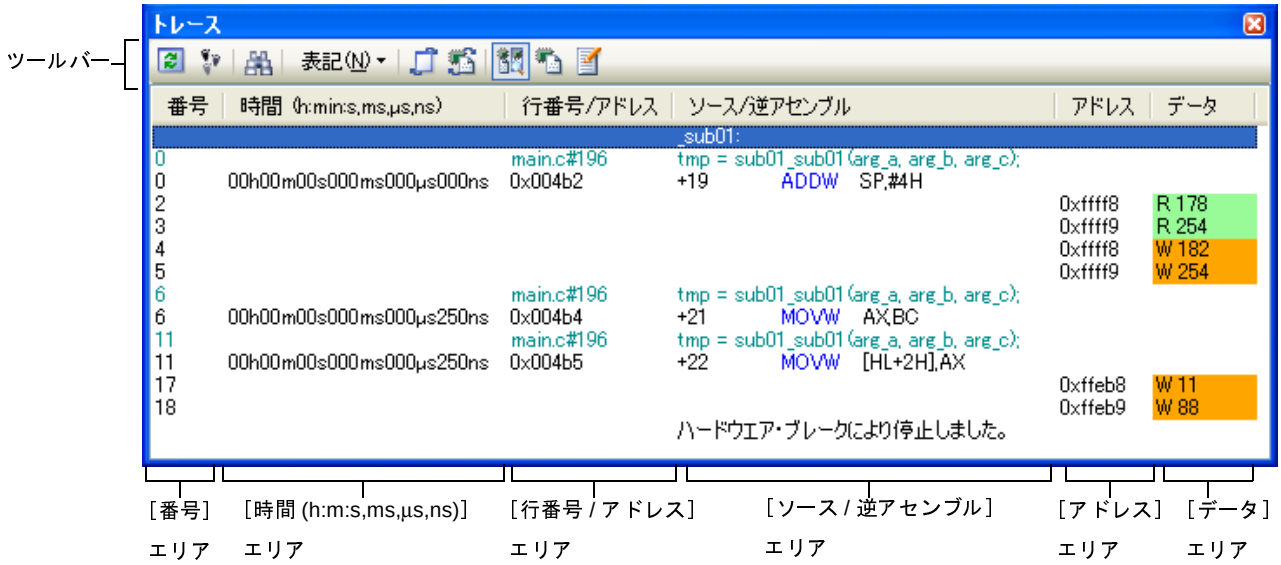
収集したトレース・データの表示は，次のトレース パネル【IECUBE】【シミュレータ】で行います。

［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，デフォルトで逆アセンブル・テキストとソース・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【IECUBE】【シミュレータ】の項を参照してください。

図 2—109 トレース・データの表示（トレース パネル）



ここでは，次の操作方法について説明します。

（1）表示モードを変更する

（2）値の表示形式を変更する

（3）他のパネルと連動させる

(1) 表示モードを変更する

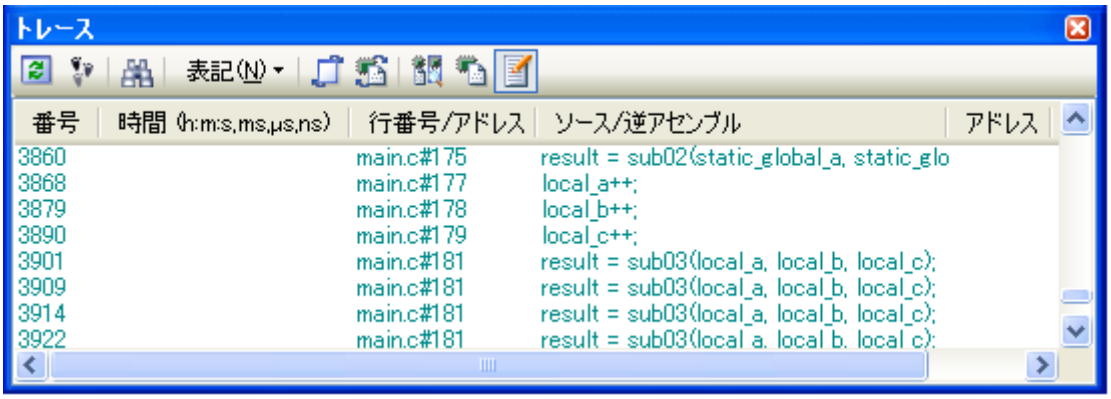
次のツールバーのボタンをクリックすることで，用途に応じて表示モードを変更することができます。

ただし，トレース機能が動作中の場合は無効となります。

表 2—10 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／リセット要因【IECUBE】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／リセット要因【IECUBE】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／リセット要因【IECUBE】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—110 ソース表示モードの例（トレース パネル）



(2) 値の表示形式を変更する

［行番号 / アドレス］エリア／［アドレス］エリア／［データ］エリアの表示形式は，ツールバーの次のボタンにより，自由に変更することができます。

ただし，プログラム実行中は，ボタンは無効となります。

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

(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. 11. 6 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，トレース機能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］を指定している場合は，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. 11. 7 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【IECUBE】【シミュレータ】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，タブ選択エリア上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。

ただし，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【IECUBE】【シミュレータ】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。

また，ソース・レベルで検索を行う場合は，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—111 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。

（1）命令レベルで検索する

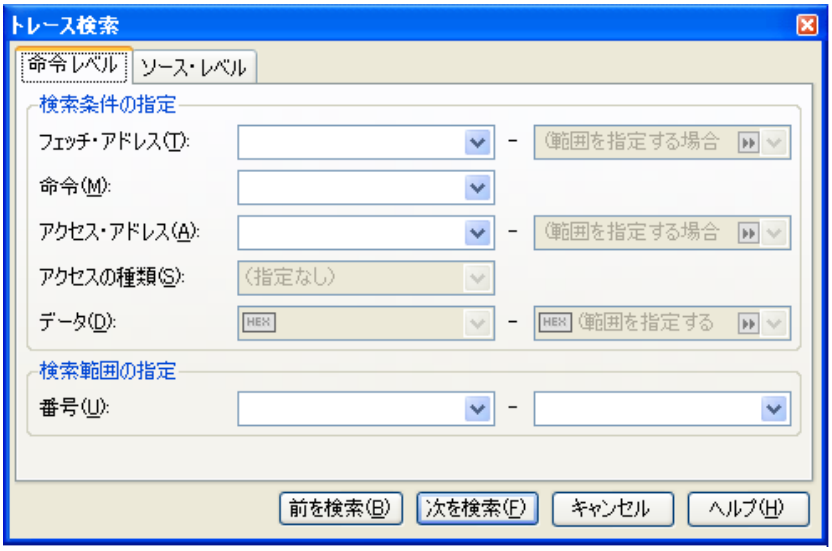
（2）ソース・レベルで検索する

（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。

図 2—112 命令レベルでのトレース・データの検索



(a)［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。

(b)［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【IECUBE】【シミュレータ】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］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。

(d)［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。

(e)［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。

(f)［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【IECUBE】【シミュレータ】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。

(g)［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【IECUBE】【シミュレータ】上で選択状態にします。

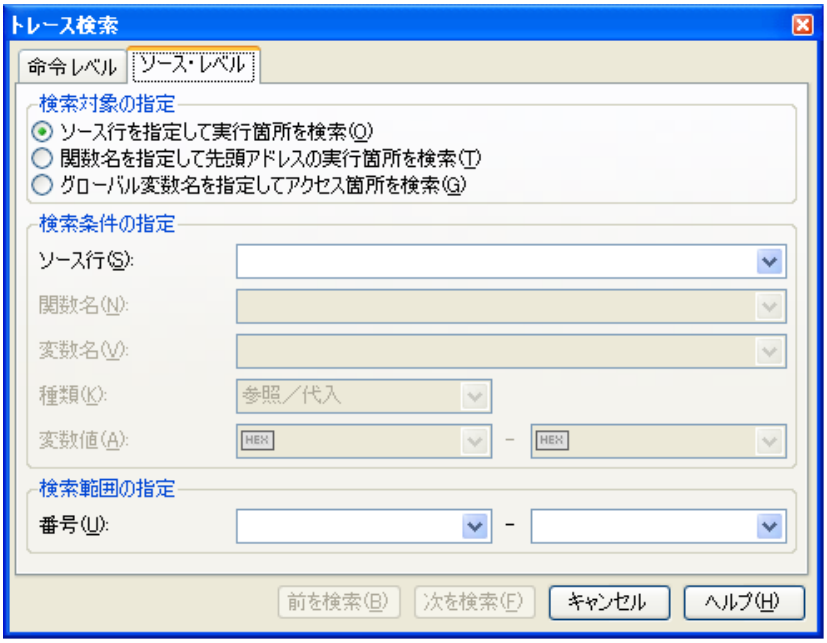
［次を検索］ボタンをクリックすると，番号の大きい方向に検索を行い，検索結果箇所をトレース パネル【IECUBE】【シミュレータ】上で選択状態にします。

(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。

図 2—113 ソース・レベルでのトレース・データの検索



(a) 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【IECUBE】【シミュレータ】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

2. main.c

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【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】上で選択状態にします。

(b) 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【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】上で選択状態にします。

(c) 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【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】上で選択状態にします。

2. 11. 8 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—114 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—115 トレース・データ保存の際の出力イメージ

```
番号      時間      行番号 / アドレス      ソース / 逆アセンブル      アドレス      データ
-----------------------------------------------------------------------------------------
番号      時間      行番号 / アドレス      ソース / 逆アセンブル      アドレス      データ
 :         :            :                      :                        :            :
```

2. 12　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。

2. 12. 1　実行開始から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-Beak 時間）を測定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　【シミュレータ】

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。

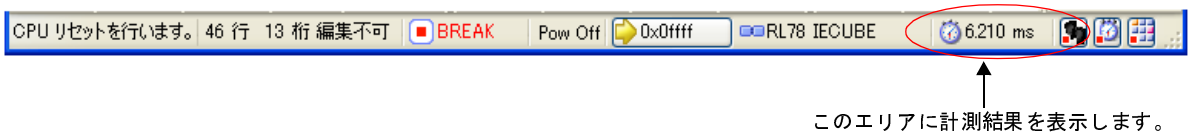
備考　この機能は，デバッグ・ツールにデフォルトで設定されているビルトイン・イベントの 1 つである Run-Break タイマ・イベントにより動作します。

Run-Break タイマ・イベントは，常に有効状態に設定されているイベントです（変更不可）。

(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）。

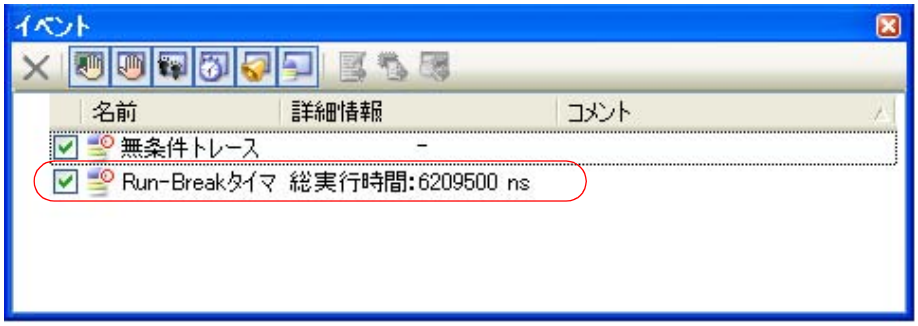
図 2—116　Run-Break タイマ・イベントの測定結果例（ステータスバー）



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



2. 12. 2　任意区間の実行時間を計測する【IECUBE】【シミュレータ】

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）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。

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.　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. 15. 6　イベント設定に関する留意事項」も参照してください。

2.　【E1】【E20】【EZ Emulator】
タイマ計測イベントはサポートしていません。

3.　【シミュレータ】
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。

（1）タイマ開始イベント／タイマ終了イベントを設定する

エディタ パネル／逆アセンブル パネルにおいて，タイマ計測を開始／終了するイベントを設定します。

（a）タイマ開始イベントの設定方法

タイマ計測を開始したい行／アドレス注にキャレットを移動したのち，コンテキスト・メニューの［タイマ設定］→［実行時にタイマ開始］を選択します。

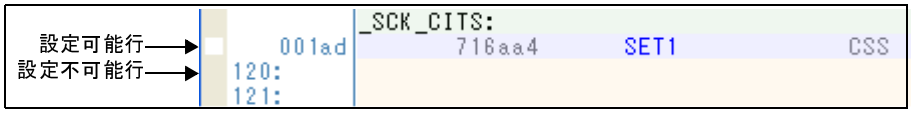
タイマ開始イベントが，キャレット位置の行／アドレスに対応する先頭アドレスの命令に対して設定されます。

（b）タイマ終了イベントの設定方法

タイマ計測をを終了したい行／アドレス注にキャレットを移動したのち，コンテキスト・メニューの［タイマ設定］→［実行時にタイマ終了］を選択します。

タイマ終了イベントが，キャレット位置の行／アドレスに対応する先頭アドレスの命令に対して設定されます。

注　タイマ開始イベント／タイマ終了イベントは，次図に示すように，イベント・エリアにおいて背景色がグレー表示の行（その行が対応するアドレスに変換不可であることを示す）では設定することはできません。



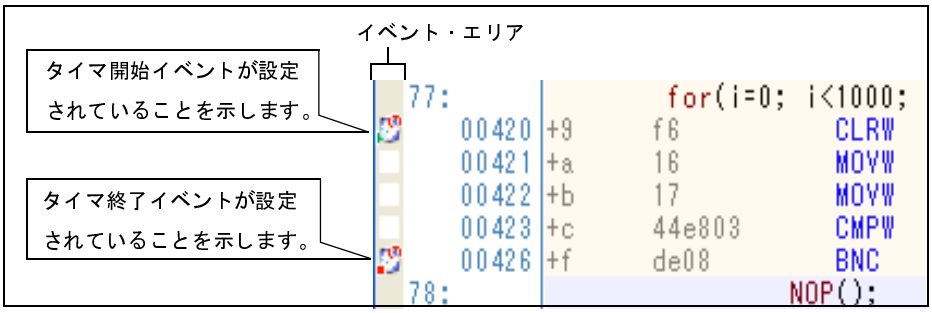
タイマ開始イベント／タイマ終了イベントが設定されると，該当行／アドレスのイベント・エリアに次のイベント・マークが表示されます。

また，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ます（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情報を確認することができます）。

表 2—11　タイマ開始イベント／タイマ終了のイベント・マーク

種別	イベント・マーク
タイマ開始	
タイマ終了	

図 2—118　タイマ開始イベント／タイマ終了イベントの設定例



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. 15. 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) プログラムを実行する

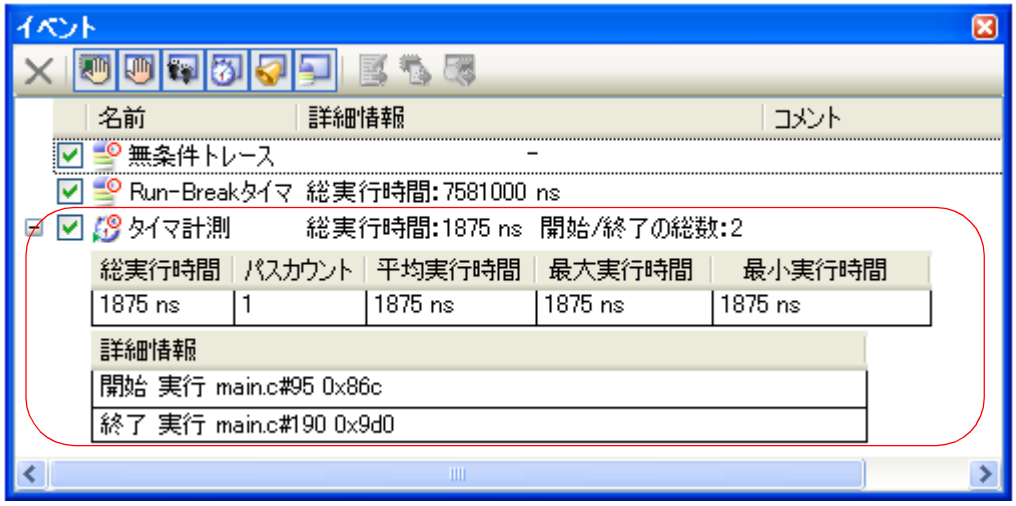
プログラムを実行します（「2. 7　プログラムの実行」参照）。

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と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—119 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



(3) タイマ開始イベント／タイマ終了イベントを削除する

設定したタイマ開始イベント／タイマ終了イベントを削除する場合は，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。

2. 12. 3 測定可能時間の範囲

Run-Break タイマ・イベント（「2. 12. 1 実行開始から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. 12. 2 任意区間の実行時間を計測する【IECUBE】【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。

なお，測定可能な最大時間を越えた場合は，タイマ・オーバ・ブレークを発生し，プログラムの実行を停止します。

表 2—12 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	
IECUBE	最小	17 ナノ秒	最小	17 ナノ秒
	最大	1 分 13 秒	最大	1 分 13 秒 最大通過回数：4294967295 回
E1/E20 EZ Emulator	最小	100 マイクロ秒	—	
	最大	約 119 時間 18 分		
シミュレータ	タイマ／トレース用クロック周波数に依存		タイマ／トレース用クロック周波数に依存	

2. 13 カバレッジの測定【IECUBE】【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，CubeSuite+ 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。

カバレッジ測定の対象となる領域は次のとおりです。

表 2—13 カバレッジ測定の対象領域

デバッグ・ツール	対象領域
IECUBE	内部 ROM/RAM，データフラッシュ，ターゲット・メモリ
シミュレータ	内部 ROM/RAM，エミュレーション ROM/RAM，ターゲット・メモリ

注意 【E1】【E20】【EZ Emulator】

カバレッジ機能はサポートしていません。

備考 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。

2. 13. 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。

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。

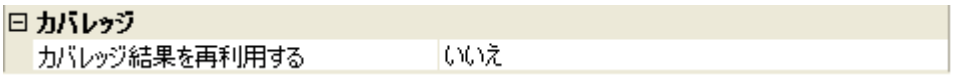
（1）【IECUBE】の場合

（2）【シミュレータ】の場合

（1）【IECUBE】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［カバレッジ］カテゴリ内で行います。

図 2—120 ［カバレッジ］カテゴリ【IECUBE】



（a）［カバレッジ結果を再利用する］

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ドロップダウン・リストにより指定します。

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（デフォルトでは［いいえ］が指定されます）。

なお，測定結果を保存するファイル（raw.csr.cv）は，現在ダウンロードしているロード・モジュール・ファイルが存在するフォルダに作成されます。

(2)【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［カバレッジ］カテゴリ内で行います。

図 2—121 ［カバレッジ］カテゴリ【シミュレータ】

⊟ カバレッジ	
カバレッジ機能を使用する	はい
カバレッジ結果を再利用する	いいえ

(a)［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ドロップダウン・リストにより指定します。

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（デフォルトでは［いいえ］が指定されます）。

(b)［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］を指定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ドロップダウン・リストにより指定します。

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（デフォルトでは［いいえ］が指定されます）。

2. 13. 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

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—14 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—15 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）。

表 2—14　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—15　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。

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。

3. 上記の背景表示は，対象領域外（「表 2—13　カバレッジ測定の対象領域」参照）の行に対しては行われません。

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—122　コード・カバレッジ測定結果の表示例（エディタ パネル）

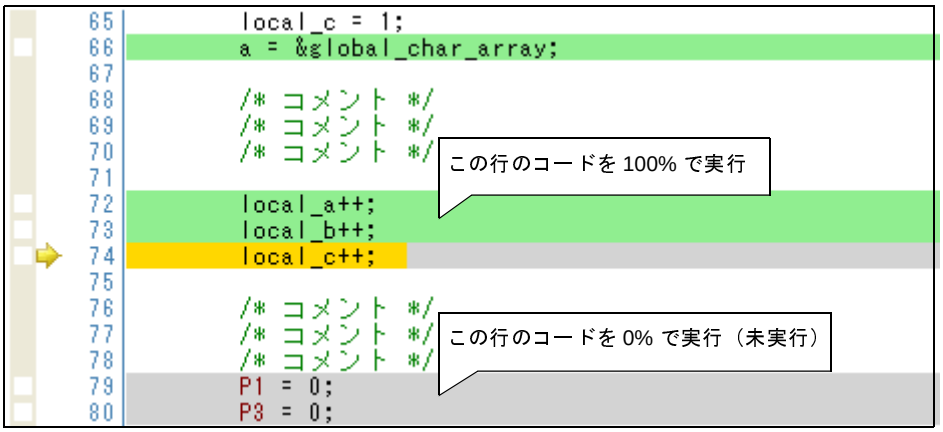
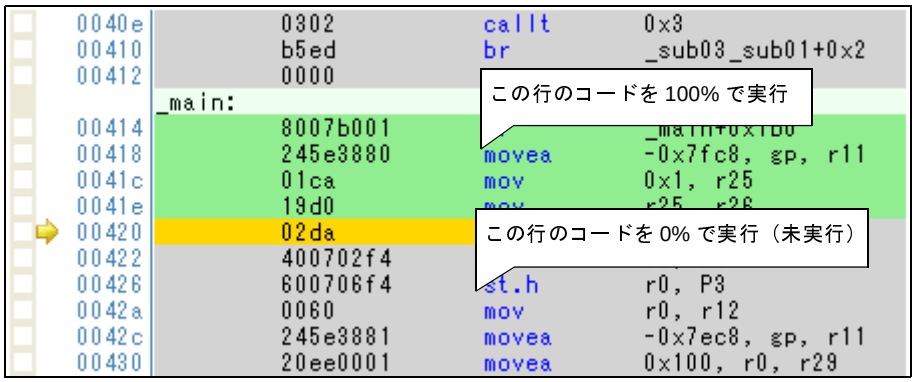


図 2—123 コード・カバレッジ測定結果の表示例（逆アセンブル パネル）



(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，「CubeSuite+ 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，「CubeSuite+ 解析編」を参照してください。

2. 14　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. 14. 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.　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. 15. 6　イベント設定に関する留意事項」も参照してください。

2.　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実行中（）にアクション・イベントは発生しません。

3.　【シミュレータ】
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［停止時にブレーク位置の命令を実行］プロパティを［はい］に指定している場合，設定したアクション・イベントはすべてブレーク・イベントとして動作します（Printf イベントは発生しません）。

4.　一度設定した Printf イベントの変数式を編集することはできません。編集が必要な場合は，該当する Printf イベントを削除したのち，新たに Printf イベントを設定してください。

（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。

エディタ パネル／逆アセンブル パネルにおいて，Printf イベントを設定したい行／アドレス注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。

注　アクション・イベントは，次図に示すように，イベント・エリアにおいて背景色がグレー表示の行（その行が対応するアドレスに変換不可であることを示す）では設定することはできません。

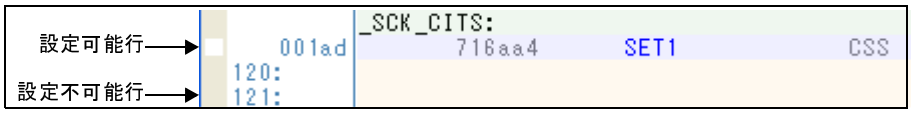
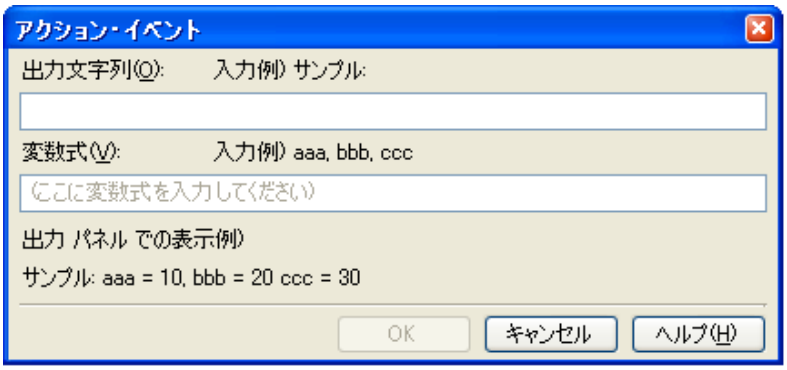


図 2—124　Printf イベントを設定する（アクション・イベント ダイアログ）



(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—14　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。

(c)［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. 15　イベントの管理」参照）。

(2) プログラムを実行する

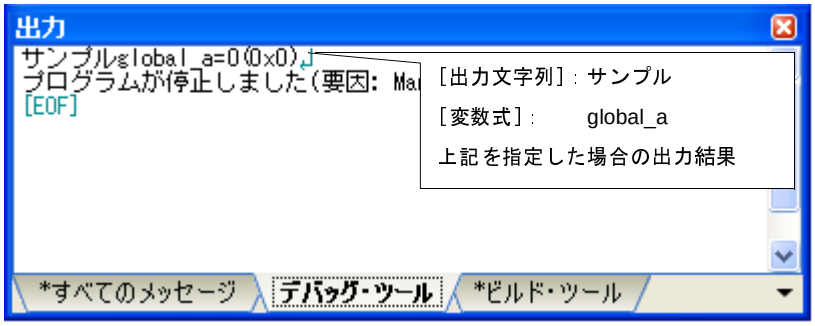
プログラムを実行します（「2. 7　プログラムの実行」参照）。

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 2—125 Printf イベントの出力結果例



2. 15 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。

CubeSuite+ 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。

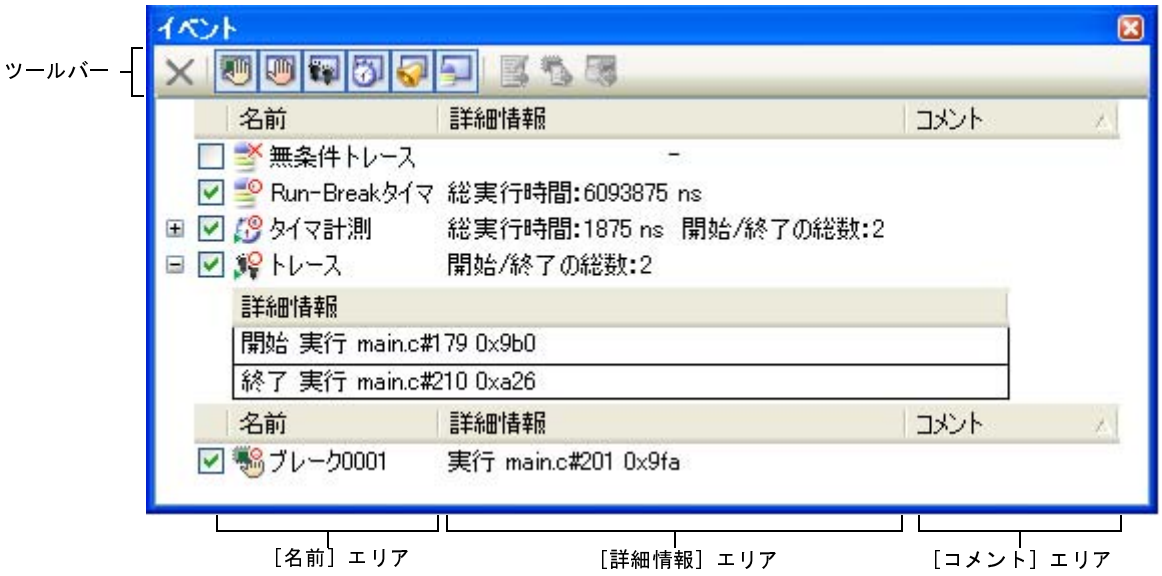
イベントは，一括して次のイベント パネルで管理されます。

［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—126 設定したイベントの表示（イベント パネル）



2. 15. 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—127 イベント名のチェック・ボックス

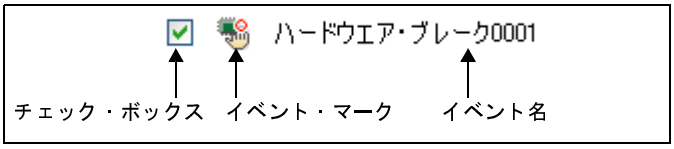


表 2—16 イベントの設定状態

	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。

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。

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。

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. 15. 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。
（【シミュレータ】以外）	ソフトウエア・ブレーク関連のイベントを表示します。
【IECUBE】【シミュレータ】	トレース関連のイベントを表示します。
【IECUBE】【シミュレータ】	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. 15. 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. 15. 4 イベントを削除する

設定したイベントを削除するには，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。

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. 15. 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. 15. 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。

（1）有効イベント数の制限

（2）実行中に設定／削除可能なイベント種別

（3）その他の注意事項

(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—17 有効イベント数の制限

イベント種別	デバッグ・ツール		
	IECUBE	E1/E20 EZ Emulator	シミュレータ
ハードウエア・ブレーク（実行系：実行前）	4 ～ 8 注1	0	64 注3
ハードウエア・ブレーク（実行系：実行後）	8	1 注2	
ハードウエア・ブレーク（アクセス系）	8		
ソフトウエア・ブレーク	2000		—
トレース（トレース開始／トレース終了）	4 + 4 注4	—	32 注4
ポイント・トレース	8 + 8	—	64
タイマ計測（タイマ開始／タイマ終了）	4 + 4 注4	—	1
アクション（Printf イベント）	100 注5		64 注6

注 1. イベントを設定するアドレスと命令の内容に依存して変動

2. ハードウエア・ブレーク（実行系：実行後）とハードウエア・ブレーク（アクセス系）で兼用

3. 実行前ブレーク／実行後ブレークはプロパティ パネルにおいて指定可

4. 今版では 1 組のみ設定可（ただし，開始イベント／終了イベントは複数設定可）

5. ソフトウエア・ブレークと兼用（ただし，有効／無効状態に関わらず 100 個まで）

6. ハードウエア・ブレークと兼用（ただし，有効／無効状態に関わらず 64 個まで）

(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—18 実行中に設定／削除可能なイベント種別

機能	デバッグ・ツール		
	IECUBE	E1/E20 EZ Emulator	シミュレータ
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（実行系：実行後）	○	—	▲
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）	○	—	▲

機能	デバッグ・ツール		
	IECUBE	E1/E20 EZ Emulator	シミュレータ
アクション（Printf イベント）	△	―	―

○ ：可能

△ ：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，または非サポート

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］を指定することにより可能となります。

(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。

- 内部 ROM/ 内部 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）。

- 【シミュレータ】以外
  32 ビット（4 バイト）の変数に対して，発生条件がアクセス系のハードウエア・ブレーク・イベント／ポイント・トレース・イベント【IECUBE】を設定することはできません。
  また，16 ビット（2 バイト）の変数に対して，1 バイトでアクセスしている場合，発生条件がアクセス系のハードウエア・ブレーク・イベント／ポイント・トレース・イベント【IECUBE】は，そのアクセスを検出することはできません。

2. 16 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，SFR/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 SFR 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—128 ［フック処理］カテゴリ

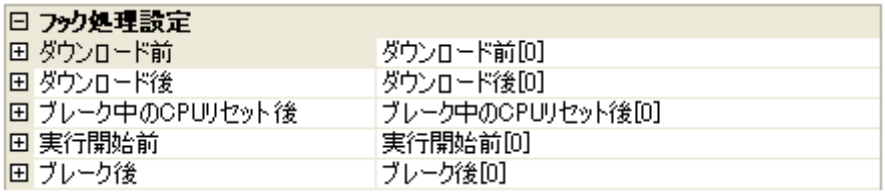


表 2—19 ［フック処理］カテゴリのプロパティ

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	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—129 テキスト編集 ダイアログのオープン

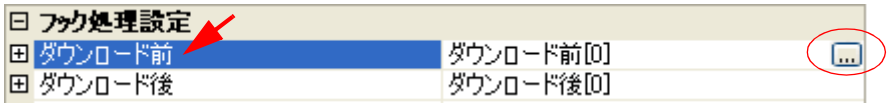
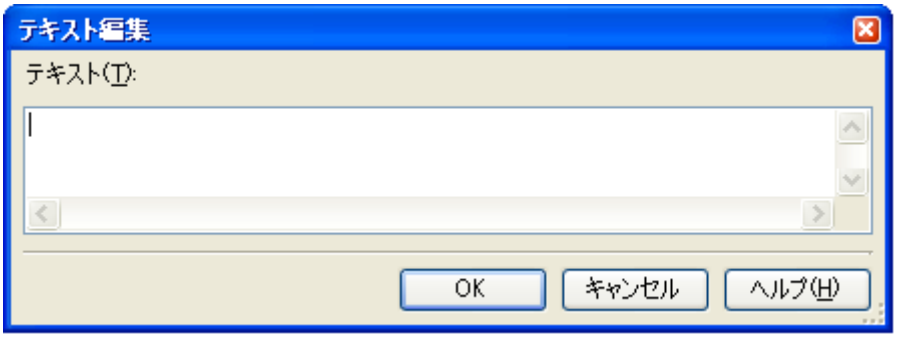


図 2—130 フック処理を設定する（テキスト編集 ダイアログ）



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。

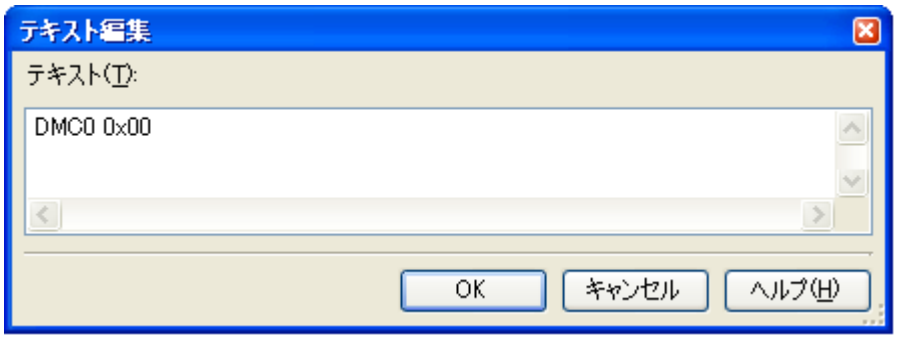
処理の指定形式は次のとおりです。ここで指定した *SFR/CPU レジスタ*が，該当するタイミングで，指定した*数値*に自動的に書き換えられます。

- *SFR 名/CPU レジスタ名* + 半角スペース + *数値*

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—131 フック処理設定の例



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。

2. 半角スペースは，タブ文字でも代用可能です。

2. 17 シミュレータ GUI の使用【シミュレータ】

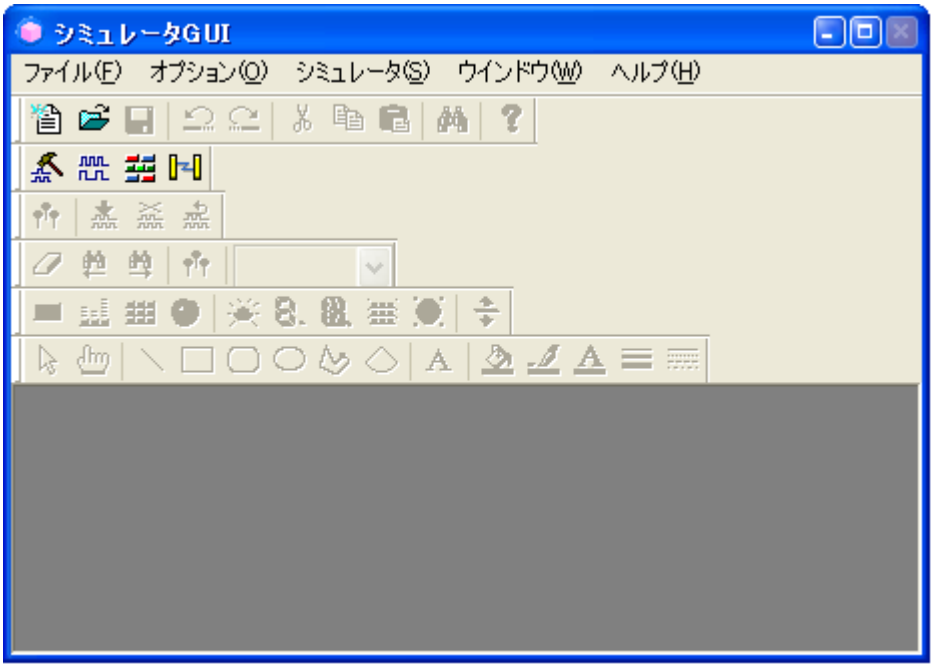
この節では，シミュレータ GUI の使用方法について説明します。

なお，この節で説明するシミュレータ GUI は，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る場合のみ提供する機能です。

シミュレータ GUI のすべての操作は，次のシミュレータ GUI ウインドウより行います。このウインドウは，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器シミュレーションをサポートしている場合で，かつ使用するデバッグ・ツールに“シミュレータ”を選択している場合，デバッグ・ツールと接続すると自動的にオープンします（デフォルト）。

備考 シミュレータ GUI ウインドウ，およびこのウインドウよりオープンする各種ウインドウは，CubeSuite+ のメイン・ウインドウとドッキング表示することはできません。

図 2—132 シミュレータ GUI を使用する（シミュレータ GUI ウインドウ）



なお，シミュレータ GUI ウインドウの表示に関する設定は，次のプロパティ パネルの［デバッグ・ツール設定］タブ上の［シミュレータ GUI］カテゴリ内で行うことができます。

必要に応じて，次の設定を行ってください。

注意 デバッグ・ツールと接続後，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。

図 2—133 ［シミュレータ GUI］カテゴリ

⊟ シミュレータGUI	
シミュレータGUIを表示する	はい
実行開始時に最前面表示する	はい

(1)［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ドロップダウン・リストにより指定します。

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。

(2)［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］を指定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ドロップダウン・リストにより指定します。

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）。

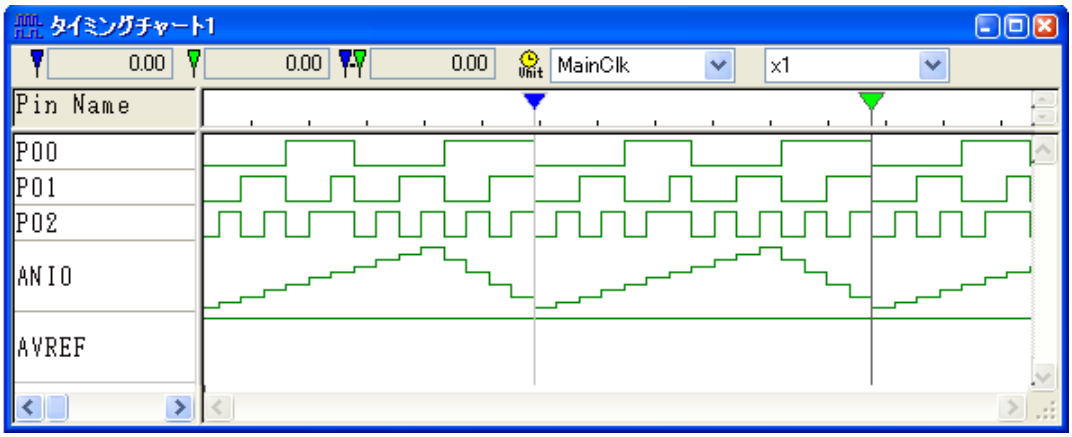
2. 17. 1 マイコンの入出力波形を確認する

マイコンの入出力波形の確認は，シミュレータ GUI ウインドウのツールバーの ボタンをクリックすることによりオープンする，次のタイミングチャート ウインドウで行うことができます。

このウインドウでは，マイコンの端子に対する入力信号と出力信号をタイミング・チャートで表示します。

操作方法についての詳細は，タイミングチャート ウインドウの項を参照してください。

図 2—134 マイコンの入出力波形の確認（タイミングチャート ウインドウ）



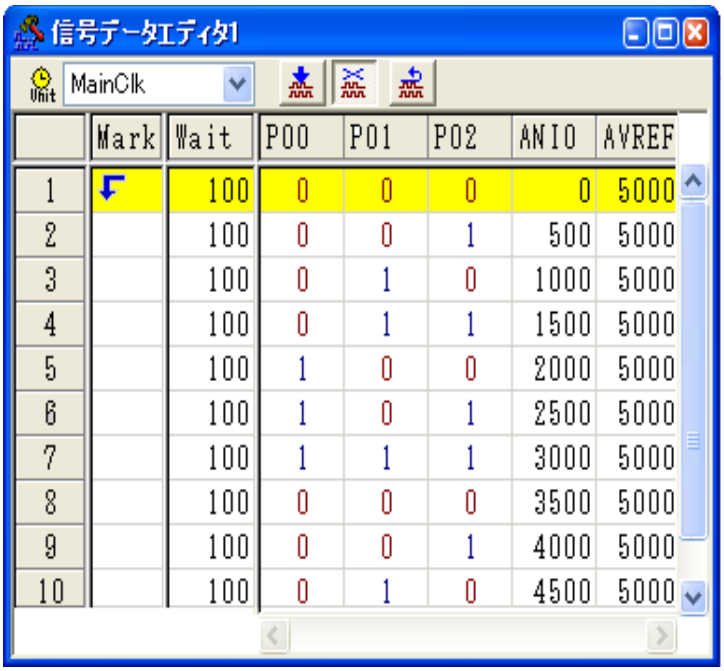
2. 17. 2　端子へ信号を入力する

端子への入力信号の設定は，シミュレータ GUI ウインドウのツールバーの ボタンをクリックすることによりオープンする，次の信号データエディタ ウインドウで行うことができます。

このウインドウでは，入力端子に対して任意のタイミングの入力信号データを数値で設定することができます。

操作方法についての詳細は，信号データエディタ ウインドウの項を参照してください。

図 2—135　端子への入力信号の設定（信号データエディタ ウインドウ）



	Mark	Wait	P00	P01	P02	ANI0	AVREF
1		100	0	0	0	0	5000
2		100	0	0	1	500	5000
3		100	0	1	0	1000	5000
4		100	0	1	1	1500	5000
5		100	1	0	0	2000	5000
6		100	1	0	1	2500	5000
7		100	1	1	1	3000	5000
8		100	0	0	0	3500	5000
9		100	0	0	1	4000	5000
10		100	0	1	0	4500	5000

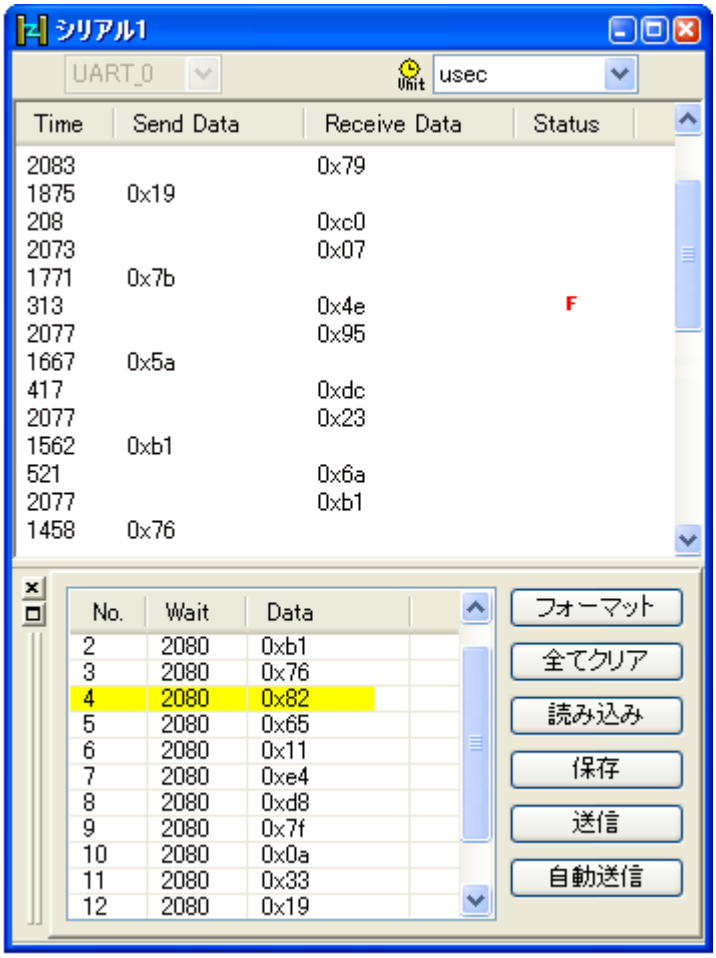
2. 17. 3 シリアル通信を行う

シリアル通信の設定は，シミュレータ GUI ウインドウのツールバーの ボタンをクリックすることによりオープンする，次のシリアル ウインドウで行うことができます。

このウインドウでは，CPU に搭載されているシリアル・インタフェースと通信を行うためのシリアル入出力機能として，マイクロコントローラのシリアル受信端子へのデータ入力と送信端子からの出力データの取得が可能です。

操作方法についての詳細は，シリアル ウインドウの項を参照してください。

図 2—136 シリアル通信を行う（シリアル ウインドウ）



2. 17. 4 ボタン /LED/ レベル・ゲージ等の部品を使用する

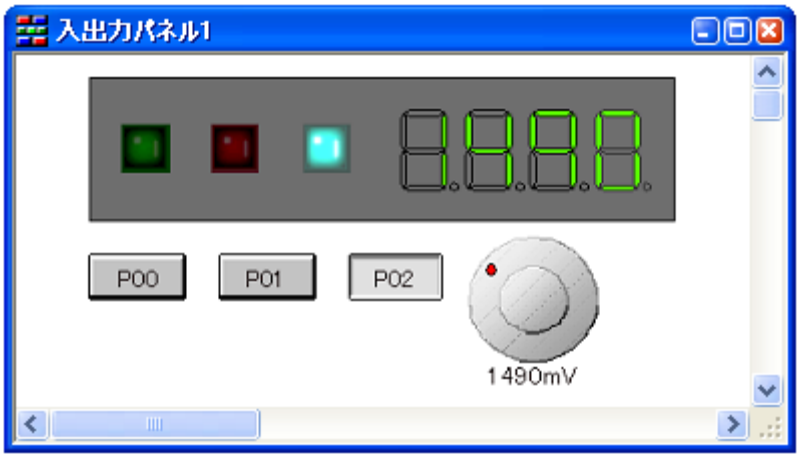
シミュレータ GUI では，周辺 I/O との入出力部を GUI 化した標準的な接続部品（ボタン /LED/ レベル・ゲージなど）を提供することで，入力操作，および出力表示のシミュレーションを可能にしています。

各種接続部品の設定は，シミュレータ GUI ウインドウのツールバーの ボタンをクリックすることによりオープンする，次の入出力パネル ウインドウで行うことができます。

このウインドウにおいて各種接続部品の設定を行うことにより，疑似的なターゲット・システムを構築することができます。

操作方法についての詳細は，入出力パネル ウインドウの項を参照してください。

図 2—137 各種接続部品の設定（入出力パネル ウインドウ）



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—20 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

(2) 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—21 進数の一覧

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。

(3) 式と演算子

式とは，定数，レジスタ名，SFR 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。

シンボルとして，SFR 名，ラベル名，関数名，変数名が記述された場合は，そのアドレスをシンボルの値として演算します。

式の基本入力形式は次のとおりです。

表 2—22 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 [*インデクス*]	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
CPU レジスタ名	CPU レジスタの値
SFR 名	SFR の値
ラベル名，EQU シンボル名，即値アドレス	ラベルの値，EQU シンボルの値，即値アドレスの値
ビット・シンボル	ビット・シンボル の値

2. 18. 2 入力不備箇所に対するアイコン表示

CubeSuite+ 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。

付録 A　ウインドウ・リファレンス

この付録では，CubeSuite+ 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
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
エディタ パネル	テキスト・ファイルの表示／編集，およびソース・レベル・デバッグ
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（汎用レジスタ／制御レジスタ）の内容の表示，および値の変更
SFR パネル	SFR 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【IECUBE】【シミュレータ】	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・マッピングの設定
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
テキスト編集 ダイアログ	複数行のテキストの入力，編集
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
ファイルの保存設定 ダイアログ	ファイルのエンコード，および改行コードの設定
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
トレース検索 ダイアログ【IECUBE】【シミュレータ】	トレース・データの検索
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
処理中表示 ダイアログ	処理の進捗状況の表示
オプション ダイアログ	各種環境の設定
ダウンロードするファイルを選択 ダイアログ	ダウンロード・ファイルの選択
名前を付けて保存 ダイアログ	ファイル，またはパネルの内容の新規保存
データ保存ファイルを選択 ダイアログ	データを保存する際のファイルの選択
シミュレータ・コンフィギュレーション・ファイルを選択 ダイアログ【シミュレータ】	シミュレータ・コンフィギュレーション・ファイルの選択

表 A—2　ウインドウ／ダイアログ一覧（シミュレータ GUI 部専用）

ウインドウ／パネル／ダイアログ名	機能概要
シミュレータ GUI ウインドウ	各種シミュレータ GUI ウインドウのオープン，および操作
書式設定 ダイアログ	ウインドウ色，フォント等の設定
信号データエディタ ウインドウ	信号データの入力設定
ループ設定 ダイアログ	信号データエディタ ウインドウのループ情報設定
端子選択 ダイアログ	信号データエディタ ウインドウ，およびタイミングチャート ウインドウの表示端子の選択
タイミングチャート ウインドウ	入力信号と出力信号のタイミング・チャート表示
データ検索 ダイアログ	タイミングチャート ウインドウの詳細検索
入出力パネル ウインドウ	疑似的なターゲット・システムの構築
Parts Button Properties ダイアログ	ボタンの端子接続情報を設定
Analog Button Properties ダイアログ	アナログ・ボタンの端子接続情報を設定
Parts Key Properties ダイアログ	キー・マトリクス LED の端子接続情報を設定
Parts Level Gauge Properties ダイアログ	レベル・ゲージの端子接続情報を設定
Parts Led Properties ダイアログ	LED の端子接続情報を設定
Parts Segment LED Properties ダイアログ	7/14 セグメント LED の端子接続情報を設定
Parts Matrix Led Properties ダイアログ	マトリクス LED の端子接続情報を設定
Parts Buzzer Properties ダイアログ	ブザーの端子接続情報を設定
プルアップ／プルダウン設定 ダイアログ	プルアップ／プルダウン抵抗の端子接続情報を設定
Object Properties ダイアログ	図形／文字／ビットマップの端子への接続情報の設定
部品一覧 ダイアログ	入出力パネル ウインドウ上のオブジェクトの端子接続状況の一覧表示
シリアル ウインドウ	シリアル・インタフェースの通信ウインドウ

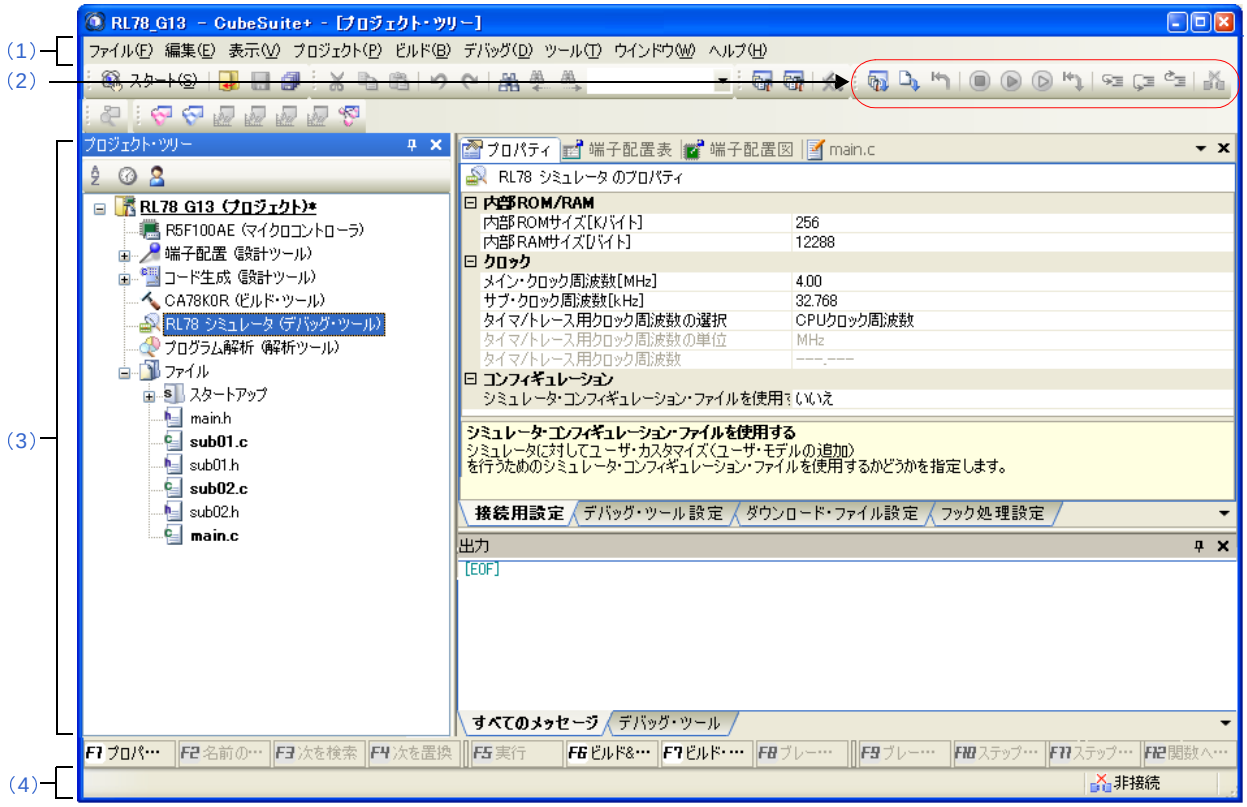
ウインドウ／パネル／ダイアログ名	機能概要
フォーマット（UART）ダイアログ	シリアル・フォーマットの設定（UART）
フォーマット（CSI）ダイアログ	シリアル・フォーマットの設定（CSI）

メイン・ウインドウ

CubeSuite+ 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 CubeSuite+］→［CubeSuite+］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）。

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。
SFR	SFR 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 【IECUBE】【シミュレータ】	トレース パネル【IECUBE】【シミュレータ】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（「(4) 関数へジャンプする」／「(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（「(5) タグ・ジャンプする」参照）。

(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＆デバッグ・ツールへダウンロード］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・ツールへ接続	アクティブ・プロジェクトのデバッグ・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＆デバッグ・ツールへダウンロード］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，［ビルド＆デバッグ・ツールへダウンロード］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。
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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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。 CALL/CALLT/CALLF 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（CALL/CALLT/CALLF 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，CALL /CALLT/CALLF 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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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. 7. 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。

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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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）。 CALL/CALLT/CALLF 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（CALL/CALLT/CALLF 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，CALL/CALLT/CALLF 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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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. 7. 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

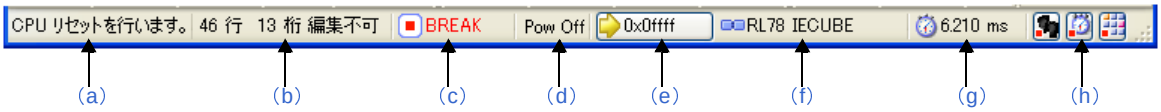
各種パネルを表示するエリアです。

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. 8　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) 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

(d) 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
IECUBE	Halt	HALT モード中
	Stop	STOP モード中
	Wait	ウエイト状態
	Reset	リセット状態
	PowOff	ターゲットに電源が供給されていない状態
E1/E20 EZ Emulator	Reset	リセット状態
	PowOff	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
	Stop	STOP モード中
	Reset	リセット状態

(e) 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，次の情報をポップアップ表示します。

- カレント PC：0x *カレント PC 値*（*ソース名* # *行数*[注]）

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。

(f) 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

（g）Run-Break タイマ結果

Run-Break タイマの計測結果（「2. 12. 1　実行開始から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

（h）デバッグ・ツールの状態【IECUBE】【シミュレータ】

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。

機能が停止中の場合，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[注]の状態を変更することができます。

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	動作中	停止中（使用する）	使用しない
トレース			
タイマ			
カバレッジ			

注　【IECUBE】

トレース機能／タイマ機能／カバレッジ機能は常に使用するため，変更することはできません（“使用しない”のアイコンは表示されません）。

【シミュレータ】

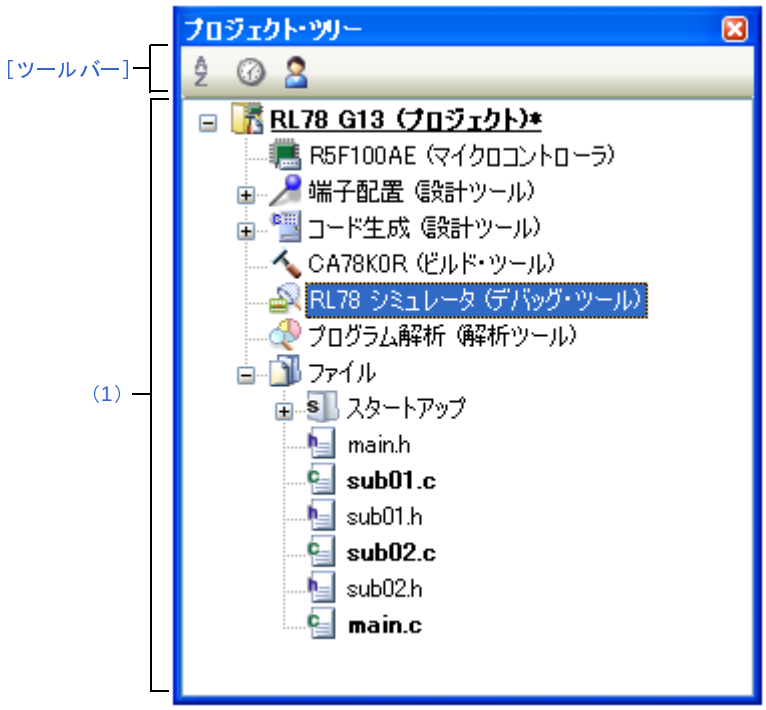
プロパティ パネルの［デバッグ・ツール設定］タブ上の［トレース］／［タイマ］／［カバレッジ］カテゴリ内［トレース機能を使用する］／［タイマ機能を使用する］／［カバレッジ機能を使用する］プロパティの指定に反映します。

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—3　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］
- ［ツールバー］
- ［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
RL78 *デバッグ・ツール名* (デバッグ・ツール)	- *デバッグ・ツール名* 使用するデバッグ・ツール（IECUBE，E1(Serial)，E20(Serial)，EZ Emulator，シミュレータ）を表示します。 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。

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）。

[ツールバー]

／／	カテゴリ・ノード，およびファイル名を名前（文字コード）順でソート表示します。 クリックの繰り返しにより，昇順表示／降順表示が切り替わります。	
		名前順のソート表示を行っていないことを示します（デフォルト）。
		降順表示を行っていることを示します。
		昇順表示を行っていることを示します。
／／	カテゴリ・ノード，およびファイル名をタイムスタンプ順でソート表示します。 クリックの繰り返しにより，昇順表示／降順表示が切り替わります。	
		タイムスタンプ順のソート表示を行っていないことを示します（デフォルト）。
		降順表示を行っていることを示します。
		昇順表示を行っていることを示します。
／	カテゴリ・ノード，およびファイル名をユーザが指定した順で表示します。	
		カスタマイズ表示を行っていないことを示します。
		カスタマイズ表示を行っていることを示します（デフォルト）。 カテゴリ・ノード，またはファイル名をドラッグ・アンド・ドロップすることにより，表示順をカスタマイズすることができます。

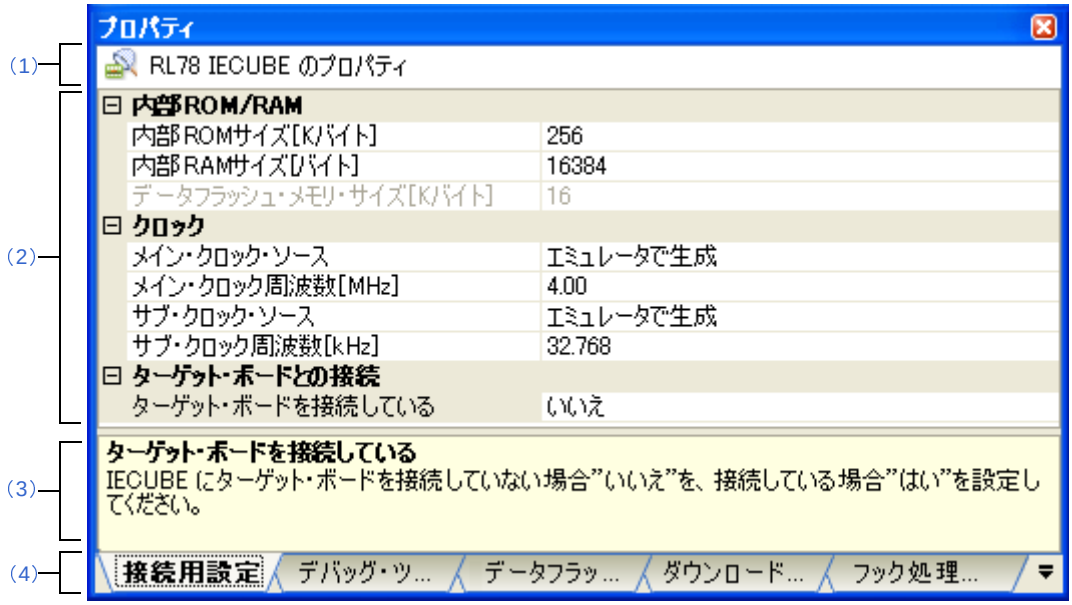
[コンテキスト・メニュー]

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。
RL78 IECUBE	IECUBE を使用します。
RL78 E1(Serial)	E1 をシリアル通信方式で使用します。
RL78 E20(Serial)	E20 をシリアル通信方式で使用します。
RL78 EZ Emulator	評価キットなどを接続して使用します。
RL78 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—4　プロパティ パネル（IECUBE 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［RL78 *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［RL78 *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［RL78 *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) 対象名エリア

プロジェクト・ツリー パネルで選択しているデバッグ・ツールの名称を表示します。

(2) 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。

(3) プロパティの説明エリア

詳細情報表示／変更エリアで選択したカテゴリやプロパティの簡単な説明を表示します。

(4) 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
- ［データフラッシュ・エミュレーション設定］タブ【IECUBE】
- ［ダウンロード・ファイル設定］タブ
- ［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。

[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）［内部 ROM/RAM］
（2）［クロック］
（3）［ターゲット・ボードとの接続］（【シミュレータ】以外）
（4）［フラッシュ］【E1】【E20】【EZ Emulator】
（5）［コンフィギュレーション］【シミュレータ】

図 A—5　プロパティ パネル：［接続用設定］タブ【IECUBE】

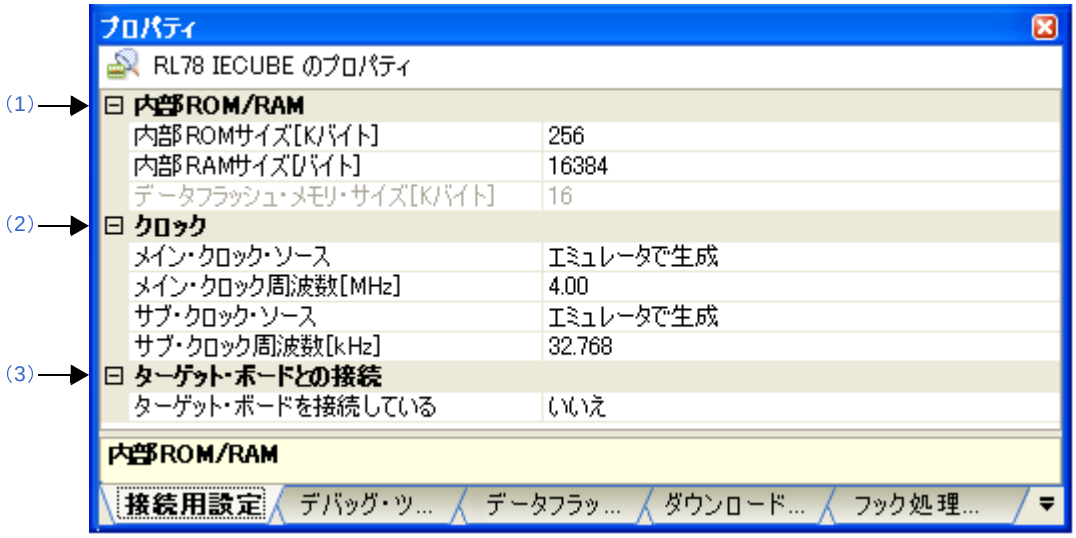


図 A—6　プロパティ パネル：［接続用設定］タブ【E1】

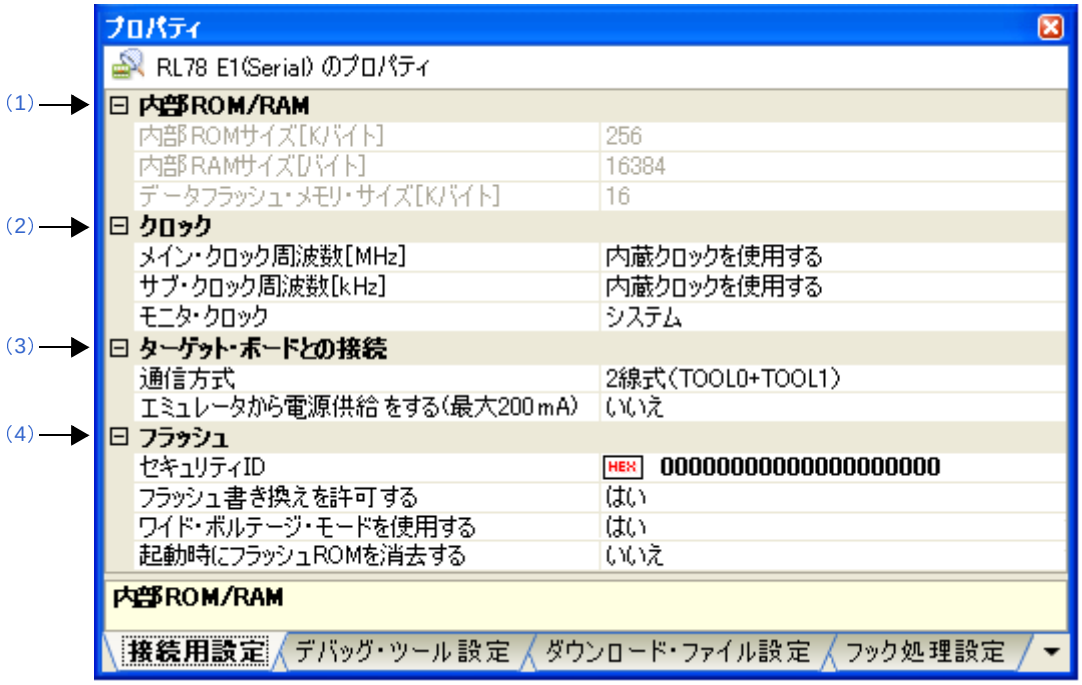


図 A—7　プロパティ パネル：［接続用設定］タブ【E20】【EZ Emulator】

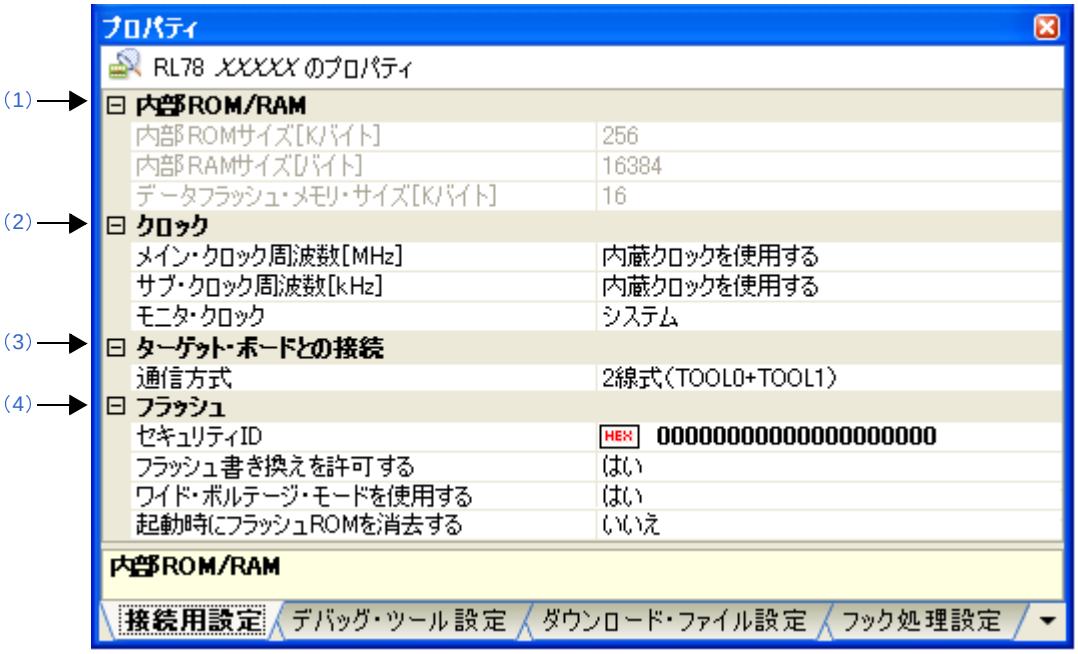
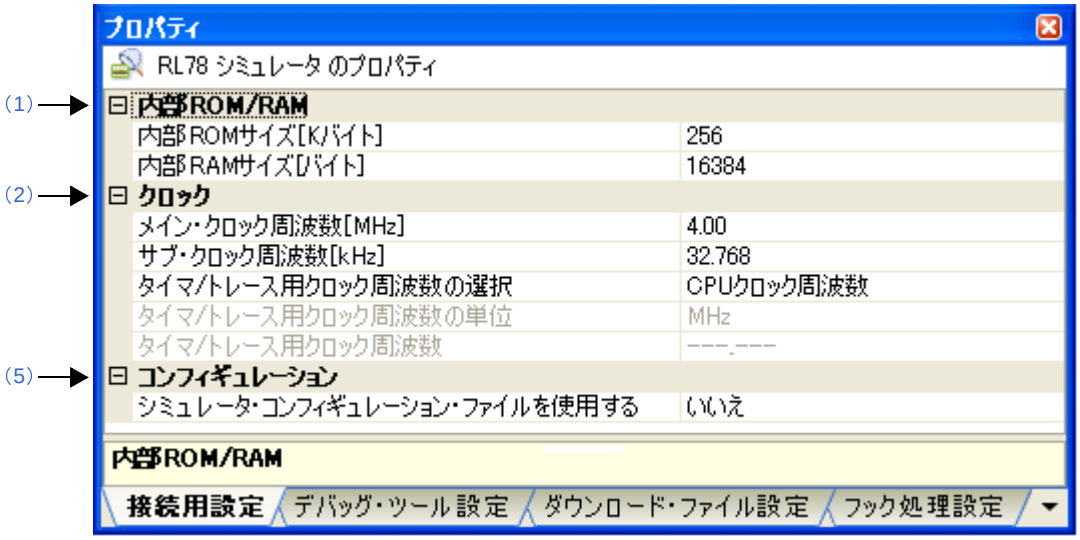


図 A—8　プロパティ パネル：［接続用設定］タブ【シミュレータ】



［各カテゴリの説明］

（1）［内部 ROM/RAM］

内部 ROM/RAM 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　設定の際には，他のメモリ・マッピング領域と重複しないよう注意が必要です。

内部 ROM サイズ [K バイト]	エミュレーション（シミュレーション）する内部 ROM サイズの表示と変更を行います。	
	デフォルト	*選択しているマイクロコントローラの内部 ROM サイズ*
	変更方法	【IECUBE】【シミュレータ】 ドロップダウン・リストによる選択 【E1】【E20】【EZ Emulator】 変更不可
	指定可能値	8，16，24，32，48，64，96，128，192，256，320，384，448，512（単位：K バイト）
内部 RAM サイズ [バイト]	エミュレーション（シミュレーション）する内部 RAM サイズの表示と変更を行います。	
	デフォルト	*選択しているマイクロコントローラの内部 RAM サイズ*
	変更方法	【IECUBE】【シミュレータ】 ドロップダウン・リストによる選択 【E1】【E20】【EZ Emulator】 変更不可
	指定可能値	512，768，1024，2048，3072，4096，5120，6144，7168，8192，10240，12288，14336，16384，18432，20480，22528，24576，26624，28672，30720，32768，34816，36864，38912，40960，43008，45056，47104，49152，51200，53248，55296，57344，59392，61440，63232（単位：バイト）

データフラッシュ・メモリ・サイズ [K バイト]（【シミュレータ】以外）	選択しているマイクロコントローラのデータフラッシュ・メモリ領域のサイズを表示します。	
	デフォルト	*選択しているマイクロコントローラのデータフラッシュ・メモリ・サイズ*
	変更方法	変更不可

(2)［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・ソース【IECUBE】	CPU に入力するメイン・クロック・ソースを指定します。		
	デフォルト	エミュレータで生成	
	変更方法	ドロップダウン・リストによる選択 ただし，デバッグ・ツールと切断中の場合のみ変更可	
	指定可能値	エミュレータで生成	IECUBE 内部で生成したクロックを使用します。
		外部	ターゲット・システム上のメイン・クロック（矩形波）を使用します。 ただし，ターゲット・システムの電源が検出不能だった場合，この項目は表示されません。
		クロック・ソケット	クロック・ソケット上の発振器のクロックを使用します。 ただし，クロック・ソケットに発振器がない場合，この項目は表示されません。

メイン・クロック周波数 [MHz]	メイン・クロックの周波数を MHz 単位で指定します。 【IECUBE】 このプロパティは，［メイン・クロック・ソース］プロパティにおいて［エミュレータで生成］を指定した場合のみ表示されます。		
	デフォルト	【IECUBE】【シミュレータ】 4.00 【E1】【E20】【EZ Emulator】 内蔵クロックを使用する	
	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	【IECUBE】 - ドロップダウン・リストによる次のいずれか 1.00，2.00，3.00，3.57，4.00，4.19，4.91，5.00，6.00，8.00，8.38，10.00，12.00，16.00，20.00（単位：MHz） 【E1】【E20】【EZ Emulator】**注1** - ドロップダウン・リストによる次のいずれか 内蔵クロックを使用する，2.00，3.00，3.57，4.00，4.19，4.91，5.00，6.00，8.00，8.38，10.00，12.00，16.00，20.00（単位：MHz） - テキスト入力による次の範囲 0.001 ～ 99.999（単位：MHz） 【シミュレータ】 - ドロップダウン・リストによる次のいずれか 2.00，3.00，3.57，4.00，4.19，4.91，5.00，6.00，8.00，8.38，10.00，12.00，16.00，20.00（単位：MHz） - テキスト入力による次の範囲 0.001 ～ 99.999（単位：MHz）	
サブ・クロック・ソース 【IECUBE】	CPU と周辺機器に入力するサブ・クロック・ソースを指定します。		
	デフォルト	エミュレータで生成	
	変更方法	ドロップダウン・リストによる選択 ただし，デバッグ・ツールと切断中の場合のみ変更可	
	指定可能値	エミュレータで生成	IECUBE 内部で生成したクロックを使用します。
		外部	ターゲット・システム上のメイン・クロック（矩形波）を使用します。

サブ・クロック周波数 [kHz]	サブ・クロックの周波数を kHz 単位で指定します。 【IECUBE】 このプロパティは，［サブ・クロック・ソース］プロパティにおいて［エミュレータで生成］を指定した場合のみ表示されます。		
	デフォルト	【IECUBE】【シミュレータ】 32.768 【E1】【E20】【EZ Emulator】 内蔵クロックを使用する	
	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	【IECUBE】 32.768，38.40（単位：kHz） 【E1】【E20】【EZ Emulator】注2 - ドロップダウン・リストによる次のいずれか 内蔵クロックを使用する，32.768，38.40（単位：kHz） - テキスト入力による次の範囲 0.001 ～ 99.999（単位：kHz） 【シミュレータ】 - ドロップダウン・リストによる次のいずれか 32.768，38.40（単位：kHz） - テキスト入力による次の範囲 0.001 ～ 99.999（単位：kHz）	
モニタ・クロック （【シミュレータ】以外）	プログラム停止中にモニタ・プログラムが動作するクロックを指定します。		
	デフォルト	システム	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	システム	メイン・クロックで動作します。
		ユーザ	プログラムで設定されているクロックで動作します。
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／トレース機能を使用する際のクロック周波数を指定します。		
	デフォルト	CPU クロック周波数	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	CPU クロック周波数	CPU クロック周波数を使用します。
		周波数の指定	任意の周波数を使用します（下段のプロパティ項目が有効となります）。
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／トレース用クロックの周波数の単位を指定します。 なお，このプロパティは，［タイマ / トレース用クロック周波数の選択］プロパティにおいて［周波数の指定］を指定した場合のみ有効となります。		
	デフォルト	MHz	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	MHz	周波数の単位を MHz とします。
		KHz	周波数の単位を kHz とします。

タイマ / トレース用クロック周波数 【シミュレータ】	このプロパティは，［タイマ / トレース用クロック周波数の選択］プロパティでの指定により，動作が異なります。 - ［周波数の指定］の場合 タイマ／トレース用クロックの周波数を指定します。 - ［CPU クロック周波数］の場合 次を表示します（変更不可）。 デバッグ・ツールと切断中：　［---_---］ デバッグ・ツールと接続中：　［*CPU クロック周波数*］	
	デフォルト	4.00
	変更方法	キーボードからの直接入力
	指定可能値	1 kHz ～ 999.999 MHz ただし，単位は［タイマ / トレース用クロック周波数の単位］プロパティの指定に依存

注 1.　X1/X2 発振の場合は，クロック周波数を指定してください。
また，PLL 搭載品で外部クロック発振の場合は，発振器／発振子の周波数（PLL 設定前）を指定してください。

2.　外部サブ・クロック発振の場合は，LIOTRM でトリミング後の周波数を指定してください。

(3)［ターゲット・ボードとの接続］（【シミュレータ】以外）

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。

ターゲット・ボードを接続している 【IECUBE】	IECUBE にターゲット・ボードを接続しているか否かを指定します。		
	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，デバッグ・ツールと切断中の場合のみ変更可	
	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。
通信方式 【E1】【E20】 【EZ Emulator】	エミュレータが，ターゲット・システム上のマイクロコントローラとシリアル通信を行う際に接続する通信方式を指定します。		
	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 ただし，デバッグ・ツールと切断中の場合のみ変更可	
	指定可能値	1 線式（TOOL0）	通信方式を 1 線式（TOOL0）とします注。
		2 線式 （TOOL0+TOOL1）	通信方式を 2 線式（TOOL0+TOOL1）とします。
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・システムに電源を供給するか否かを指定します。		
	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，デバッグ・ツールと切断中の場合のみ変更可	
	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・システムに供給する電圧を指定します。 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］を指定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 ただし，デバッグ・ツールと切断中の場合のみ変更可
	指定可能値	3.3V，5.0V

注　［実行中のメモリ・アクセス］カテゴリ内［実行を一瞬停止してアクセスする］プロパティにおいて［はい］を指定している場合，このプロパティで［1 線式 (TOOL0)］を選択すると，デバッグ・ツールの応答速度が非常に遅くなります。

(4)［フラッシュ］【E1】【E20】【EZ Emulator】

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。

セキュリティ ID	内部 ROM，または内部フラッシュ・メモリ上のコードを読み出す際のセキュリティ ID を指定します注。 なお，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（セキュリティ ID）をサポートしている場合のみ表示されます。		
	デフォルト	00000000000000000000	
	変更方法	キーボードからの直接入力 ただし，デバッグ・ツールと切断中の場合のみ変更可	
	指定可能値	0x0 ～ 0xFFFFFFFFFFFFFFFFFFFE の 16 進数	
フラッシュ書き換えを許可する	フラッシュ・メモリの書き換えを許可するか否かを指定します。		
	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	フラッシュ・メモリの書き換えを許可します。
		いいえ	フラッシュ・メモリの書き換えを許可しません。 デバッグ・ツールからフラッシュ・メモリ領域への書き換え操作は一切できなくなります。
ワイド・ボルテージ・モードを使用する	フラッシュ書き換え操作を，ワイド・ボルテージ・モードで行うか否かを指定します。 なお，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，フラッシュ書き換えのワイド・ボルテージ・モードをサポート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 ただし，デバッグ・ツールと切断中の場合のみ変更可	
	指定可能値	はい	ワイド・ボルテージ・モードでフラッシュ書き換え操作を行います。
		いいえ	通常モードでフラッシュ書き換え操作を行います。

起動時にフラッシュ ROM を消去する	デバッグ・ツールと接続する際に，フラッシュ ROM を消去するか否かを指定します。		
	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，デバッグ・ツールと切断中の場合のみ変更可	
	指定可能値	はい	デバッグ・ツールへ接続する際に，フラッシュ ROM の消去を行います。 なお，デバッグ・ツールと接続後は，自動的に［いいえ］の指定となります。
		いいえ	デバッグ・ツールと接続する際に，フラッシュ ROM の消去は行いません。

注　オンチップ・デバッグ・セキュリティ ID についての詳細は，エミュレータのユーザーズ・マニュアルを参照してください。

(5)［コンフィギュレーション］【シミュレータ】

シミュレータをカスタマイズする際の詳細情報の表示，および設定の変更を行います。

シミュレータ・コンフィギュレーション・ファイルを使用する	シミュレータに対して，ユーザ・カスタマイズ（ユーザ・モデルの追加）を行うためのシミュレータ・コンフィギュレーション・ファイルを使用するか否かを指定します。		
	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	シミュレータ・コンフィギュレーション・ファイルを使用します。
		いいえ	シミュレータ・コンフィギュレーション・ファイルを使用しません。
シミュレータ・コンフィギュレーション・ファイル	使用するシミュレータ・コンフィギュレーション・ファイルを指定します。 なお，このプロパティは，，［シミュレータ・コンフィギュレーション・ファイルを使用する］プロパティにおいて［はい］を指定した場合のみ表示されます。		
	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［...］ボタンのクリックによるオープンするシミュレータ・コンフィギュレーション・ファイルを選択 ダイアログ【シミュレータ】によるファイルの選択 ただし，デバッグ・ツールと切断中の場合のみ変更可	

［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］【IECUBE】
（4）［ブレーク］
（5）［フェイルセーフ・ブレーク］【IECUBE】
（6）［トレース］【IECUBE】【シミュレータ】
（7）［タイマ］【シミュレータ】
（8）［カバレッジ］【IECUBE】【シミュレータ】
（9）［入力信号のマスク］（【シミュレータ】以外）
（10）［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【IECUBE】

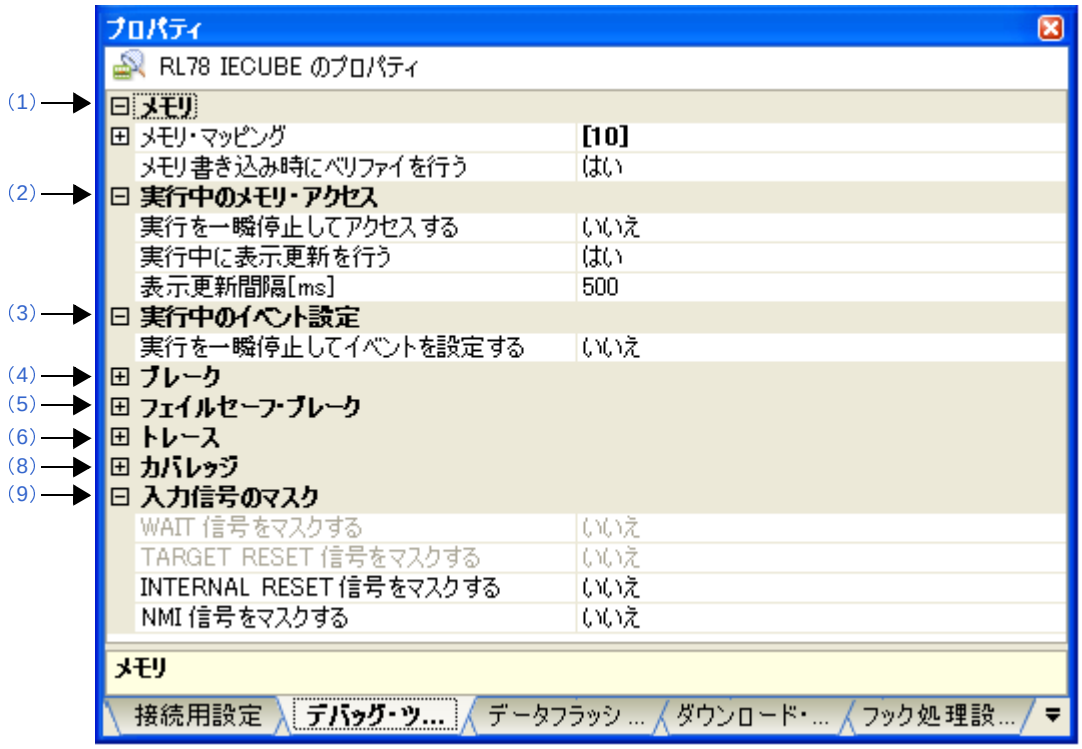


図 A—10　プロパティ パネル：［デバッグ・ツール設定］タブ【E1】【E20】【EZ Emulator】

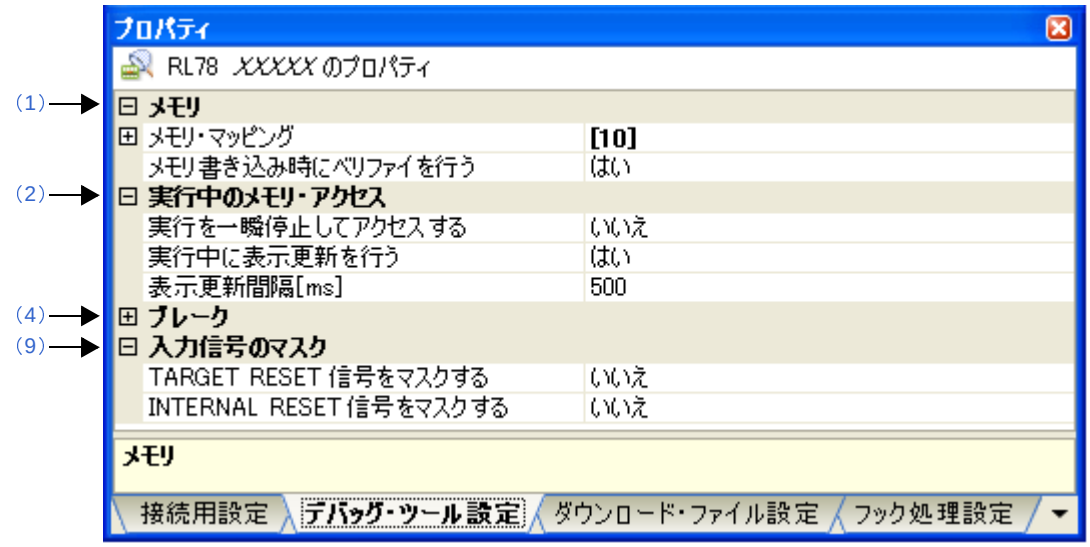


図 A—11　プロパティ パネル：［デバッグ・ツール設定］タブ【シミュレータ】

[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。

なお，表示されるメモリ種別についての詳細は，メモリ・マッピング ダイアログの項を参照してください。

メモリ・マッピング	現在のメモリ・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	メモリ・マッピング ダイアログによる指定 メモリ・マッピング ダイアログは，このプロパティを選択すると右端に表示される [...] ボタンをクリックすることでオープンします（このパネル上でマッピング値を変更することはできません）。	
	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 - アクセス幅 [ビット]	
メモリ書き込み時にベリファイを行う （【シミュレータ】以外）	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指定します。		
	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（「(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。

実行を一瞬停止してアクセスする （【シミュレータ】以外）	【IECUBE】 プログラム実行中にはアクセスできないメモリ領域（ターゲ/ット・メモリ領域 /SFR 領域 /CPU レジスタ）に対して，アクセスを許可するか否かを指定します。 【E1】【E20】【EZ Emulator】 プログラム実行中に，メモリに対してアクセスを許可するか否かを指定します注。		
	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	実行を一旦停止し，読み込み／書き込みを行います。
		いいえ	実行中にアクセスは行いません。

実行中に表示更新を行う	プログラム実行中に，ウォッチ パネル／メモリ パネルの表示内容を更新するか否かを指定します。		
	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，ウォッチ パネル／メモリ 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］を指定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）	

注　【E1】【E20】【EZ Emulator】

［ターゲット・ボードとの接続］（【シミュレータ】以外）カテゴリの［通信方式］プロパティにおいて［1 線式 (TOOL0)］を指定している場合，このプロパティで［はい］を選択すると，デバッグ・ツールの応答速度が非常に遅くなります。

(3)［実行中のイベント設定］【IECUBE】

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	プログラム実行中には設定することができないイベント（「(2) 実行中に設定／削除可能なイベント種別」参照）を，プログラムの実行を強制的に一瞬停止させることで設定を行うか否かを指定します。		
	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］

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。

優先的に使用するブレークポイントの種類 （【シミュレータ】以外）	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指定します。		
	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。
停止時にタイマ系周辺エミュレーションを停止する （【シミュレータ】以外）	実行停止時に，エミュレータのタイマ系周辺エミュレーション機能を停止するか否かを指定します。		
	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	タイマ系周辺エミュレーション機能を停止します注1。
		いいえ	タイマ系周辺エミュレーション機能を停止しません。
停止時にシリアル系周辺エミュレーションを停止する （【シミュレータ】以外）	実行停止時に，エミュレータのシリアル系周辺エミュレーション機能を停止するか否かを指定します。		
	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	シリアル系周辺エミュレーション機能を停止します。
		いいえ	シリアル系周辺エミュレーション機能を停止しません。
オープン・ブレーク機能を使用する 【IECUBE】	オープン・ブレーク機能を使用するか否かを指定します。 なお，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オープン・ブレーク機能をサポートしている場合のみ表示されます。		
	デフォルト	いいえ (Output signal)	
	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 (Hi-Z)	CPU 停止時，オープン・ブレークの対象端子は Hi-Z になります。
		いいえ (Output signal)	CPU 停止時，オープン・ブレークの対象端子は通常の状態のままで動作します。
停止時にブレーク位置の命令を実行 【シミュレータ】	ブレークポイントによるプログラム実行停止のタイミングを，ブレークポイントが設定されている位置の命令実行後とするか，または命令実行前とするかを指定します。		
	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	命令実行後にブレークします注2。
		いいえ	命令実行前にブレークします。

注 1. 【E1】【E20】【EZ Emulator】
選択しているマイクロコントローラがオープン・ブレーク機能をサポートしている場合，［はい］を指定すると，CPU 停止時にオープン・ブレークの対象端子が Hi-Z になります（［いいえ］を指定している場合では，対象端子は通常状態のままで動作します）。

2. ［はい］を指定した場合，現在設定されているアクション・イベントは，すべてブハードウエア・レーク・イベントとして動作します（「2. 14　プログラム内へのアクションの設定」参照）。

(5)［フェイルセーフ・ブレーク］【IECUBE】

フェイルセーフ・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。

フェッチ禁止領域からのフェッチ直後に停止する	フェッチ禁止領域からのフェッチ直後に，実行停止するか否かを指定します。		
	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	フェッチ直後に実行を停止します。
		いいえ	フェッチ後も実行を停止しません。
書き込み禁止領域への書き込み直後に停止する	書き込み禁止領域への書き込み直後に，実行停止するか否かを指定します。		
	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	書き込み直後に実行を停止します。
		いいえ	書き込み後も実行を停止しません。
読み込み禁止 SFR からの読み込み直後に停止する	読み込み禁止 SFR からの読み込み直後に，実行停止するか否かを指定します。		
	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	読み込み直後に実行を停止します。
		いいえ	読み込み後も実行を停止しません。
書き込み禁止 SFR への書き込み直後に停止する	書き込み禁止 SFR への書き込み直後に，実行停止するか否かを指定します。		
	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	書き込み直後に実行を停止します。
		いいえ	書き込み後も実行を停止しません。
ユーザ・スタック・オーバーフロー発生直後に停止する	ユーザ・スタック・オーバフロー発生直後に，実行停止するか否かを指定します。		
	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	発生直後に実行を停止します。
		いいえ	発生後も実行を停止しません。
ユーザ・スタック・トップ・アドレス	ユーザ・スタックのトップ・アドレスを指定します。		
	デフォルト	@STEND	
	変更方法	キーボードからの直接入力	
	指定可能値	0～“*アドレス空間の終アアドレス*”のアドレス式	

ユーザ・スタック・アンダーフロー発生直後に停止する	ユーザ・スタック・アンダーフロー発生直後に，実行停止するか否かを指定します。		
	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	発生直後に実行を停止します。
		いいえ	発生後も実行を停止しません。
ユーザ・スタック・ボトム・アドレス	ユーザ・スタックのボトム・アドレスを指定します。		
	デフォルト	@STBEG	
	変更方法	キーボードからの直接入力	
	指定可能値	0～“*アドレス空間の終アアドレス*”のアドレス式	
未初期化スタック・ポインタ操作直後に停止する	初期化していないスタック・ポインタの操作直後に，実行停止するか否かを指定します。		
	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	操作直後に実行を停止します。
		いいえ	操作後も実行を停止しません。
未初期化 RAM からの読み込み直後に停止する	初期化していない RAM からの読み込み直後に，実行停止するか否かを指定します。		
	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	読み込み直後に実行を停止します。
		いいえ	読み込み後も実行を停止しません。
非メモリ・マッピング領域へのアクセス直後に停止する	［メモリ］カテゴリ内［メモリ・マッピング］プロパティにおいて，マッピングしていない領域へのアクセス直後に，実行を停止するか否かを指定します。		
	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	アクセス直後に実行を停止します。
奇数アドレスへのワード・アクセス直後に停止する	奇数アドレスに対し，ワード幅でのアクセス直後に，実行を停止するか否かを指定します。		
	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	アクセス直後に実行を停止します。
		いいえ	アクセス後も実行を停止しません。
周辺からのフェイル・セーフ発生直後に停止する	周辺からのフェイル・セーフ発生直後に，実行を停止するか否かを指定します。		
	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	発生直後に実行を停止します。
		いいえ	発生後も実行を停止しません。

フラッシュ・イリーガル発生直後に停止する	フラッシュ・イリーガル発生直後に，実行を停止するか否かを指定します。		
	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	発生直後に実行を停止します。
		いいえ	発生後も実行を停止しません。

(6)［トレース］【IECUBE】【シミュレータ】

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。

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指定します注1。		
	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指定します。		
	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指定します。		
	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	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・タグの分周率【IECUBE】	トレースのタイム・タグに使用するタイム・タグ・カウンタの分周率を指定します。		
	デフォルト	1/1(4ns/0.3min)	
	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(4ns/0.3min)　1/2(8ns/0.6min)　1/4(17ns/1.2min) 1/8(33ns/2.4min)　1/16(67ns/4.8min)　1/32(133ns/9.5min) 1/64(267ns/19.1min)　1/128(533ns/38.2min)　1/256(1067ns/1.3h) 1/512(2133ns/2.5h)　1/1024(4267ns/5.1h)　1/2048(8533ns/10.2h) 1/4096(17067ns/20.4h) （“()”内は，分解能，および外部クロックが240MHzの場合の最大測定時間を示す）	

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【IECUBE】【シミュレータ】に表示するトレース時間の表示方法を指定します。		
	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・メモリ・サイズ[フレーム]【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2数で指定します。		
	デフォルト	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	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	

注 1. 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。

2. 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。

(7)［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指定します。		
	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。

(8)［カバレッジ］【IECUBE】【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【シミュレータ】	カバレッジ機能を使用するか否かを指定します。		
	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指定します。 【シミュレータ】 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］を指定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。

(9)［入力信号のマスク］（【シミュレータ】以外）

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【IECUBE】	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指定します。		
	デフォルト	いいえ注	
	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	WAIT 信号をマスクします。
		いいえ	WAIT 信号をマスクしません。
TARGET RESET 信号をマスクする	TARGET 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指定します。		
	デフォルト	いいえ注	
	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	TARGET RESET 信号をマスクします。
		いいえ	TARGET RESET 信号をマスクしません。
INTERNAL RESET 信号をマスクする	INTERNAL 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指定します。		
	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	INTERNAL RESET 信号をマスクします。
		いいえ	INTERNAL RESET 信号をマスクしません。
NMI 信号をマスクする【IECUBE】	NMI 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指定します。		
	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	NMI 信号をマスクします。
		いいえ	NMI 信号をマスクしません。

注　【IECUBE】

［接続用設定］タブ上の［ターゲット・ボードとの接続］（【シミュレータ】以外）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）。

（10）［シミュレータ GUI］【シミュレータ】

シミュレータ GUI 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・ツールと接続後，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 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指定します。		
	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 を使用します。
		いいえ	シミュレータ GUI を使用しません。
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指定します。 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］を指定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。

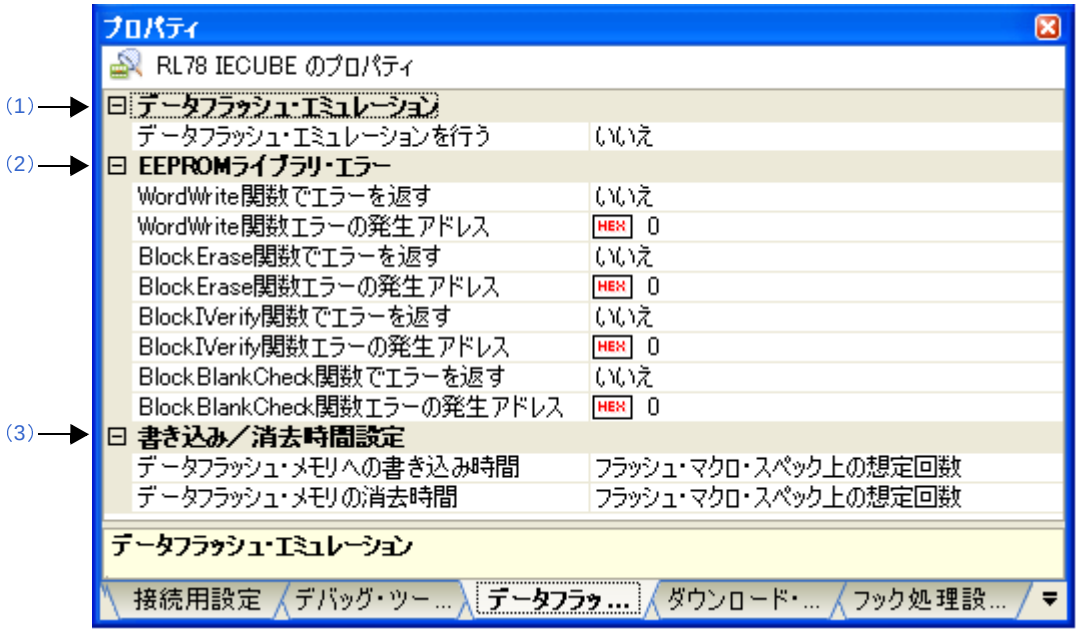
［データフラッシュ・エミュレーション設定］タブ【IECUBE】

［データフラッシュ・エミュレーション設定］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。

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データフラッシュ・メモリ内蔵品の場合のみ表示されます。

（1）［データフラッシュ・エミュレーション］
（2）［EEPROM ライブラリ・エラー］
（3）［書き込み／消去時間設定］

図 A—12　プロパティ パネル：［データフラッシュ・エミュレーション設定］タブ



［各カテゴリの説明］

（1）［データフラッシュ・エミュレーション］

データフラッシュ・エミュレーション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。

データフラッシュ・エミュレーションを行う	データフラッシュ・エミュレーション機能を使用するか否かを指定します。		
	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	データフラッシュ・エミュレーション機能を使用します。
		いいえ	データフラッシュ・エミュレーション機能を使用しません。

(2)［EEPROM ライブラリ・エラー］

EEPROM エミュレーション・ライブラ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。

このカテゴリ内の各プロパティを設定することにより，通常のエミュレーションでは返ることのないエラー値を強制的に返すことができます。

WordWrite 関数でエラーを返す	エミュレートを行う WordWrite 関数で発生するエラー値を強制的に返すか否かを指定します。		
	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	エラー値を強制的に返します。
		いいえ	エラー値を返しません。
WordWrite 関数エラーの発生アドレス	WordWrite 関数エラーを発生させるデータフラッシュ・メモリ領域内のアドレスを指定します。 なお，このプロパティは，［WordWrite 関数でエラーを返す］プロパティにおいて［はい］を指定した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	
BlockErase 関数でエラーを返す	エミュレートを行う BlockErase 関数で発生するエラー値を強制的に返すか否かを指定します。		
	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	エラー値を強制的に返します。
		いいえ	エラー値を返しません。
BlockErase 関数エラーの発生アドレス	BlockErase 関数エラーを発生させるデータフラッシュ・メモリ領域内のアドレスを指定します。 なお，このプロパティは，［BlockErase 関数でエラーを返す］プロパティにおいて［はい］を指定した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	
BlockIVerify 関数でエラーを返す	エミュレートを行う BlockIVerify 関数で発生するエラー値を強制的に返すか否かを指定します。		
	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	エラー値を強制的に返します。
		いいえ	エラー値を返しません。

BlockIVerify 関数エラーの発生アドレス	BlockIVerify 関数エラーを発生させるデータフラッシュ・メモリ領域内のアドレスを指定します。 なお，このプロパティは，［BlockIVerify 関数でエラーを返す］プロパティにおいて［はい］を指定した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	
BlockBlankCheck 関数でエラーを返す	エミュレートを行う BlockBlankCheck 関数で発生するエラー値を強制的に返すか否かを指定します。		
	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	エラー値を強制的に返します。
		いいえ	エラー値を返しません。
BlockBlankCheck 関数エラーの発生アドレス	BlockBlankCheck 関数エラーを発生させるデータフラッシュ・メモリ領域内のアドレスを指定します。 なお，このプロパティは，［BlockBlankCheck 関数でエラーを返す］プロパティにおいて［はい］を指定した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	

(3)［書き込み／消去時間設定］

データフラッシュ・メモリへの書き込み，または消去時間に関する詳細情報の表示，および設定の変更を行います。

データフラッシュ・メモリへの書き込み時間	データフラッシュ・メモリへ書き込む際の遅延時間のシミュレート値を指定します。		
	デフォルト	フラッシュ・マクロ・スペック上の想定回数	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	リトライなし	リトライ回数を 0 とし，遅延時間 =0（書き込み時間最速）とします。
		フラッシュ・マクロ・スペック上の想定回数	使用するフラッシュ・マクロのスペック上の想定回数を遅延時間とします。
		フラッシュ・マクロ・スペック上の想定最大回数	使用するフラッシュ・マクロのスペック上の想定最大回数を遅延時間とします。
		リトライ回数最大	リトライ回数を最大とし，遅延時間を最大（書き込み時間最長）とします。

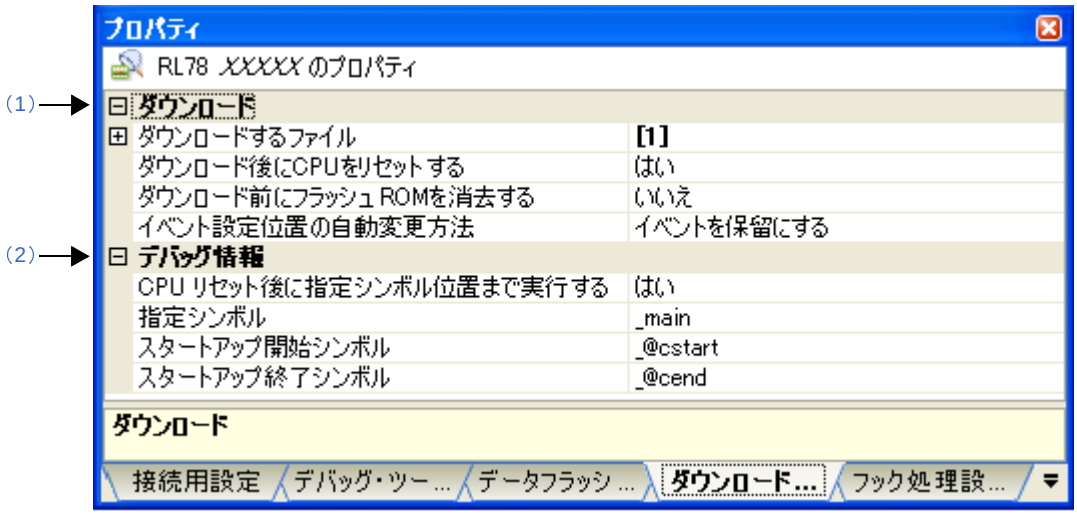
データフラッシュ・メモリの消去時間	データフラッシュ・メモリの消去を行う際の遅延時間のシミュレート値を指定します。		
	デフォルト	フラッシュ・マクロ・スペック上の想定回数	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	リトライなし	リトライ回数を 0 とし，遅延時間 =0（消去時間最速）とします。
		フラッシュ・マクロ・スペック上の想定回数	使用するフラッシュ・マクロのスペック上の想定回数を遅延時間とします。
		フラッシュ・マクロ・スペック上の想定最大回数	使用するフラッシュ・マクロのスペック上の想定最大回数を遅延時間とします。
		リトライ回数最大	リトライ回数を最大とし，遅延時間を最大（消去時間最長）とします。

［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。

（1）［ダウンロード］
（2）［デバッグ情報］

図 A—13　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（プロパティ 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指定します。		
	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	ダウンロード前にフラッシュ ROM を消去するか否かを指定します。		
	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指定します注 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。

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形式」を参照してください。

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指定します。		
	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 ～“*アドレス空間の終アアドレス*”のアドレス式	

スタートアップ開始シンボル	スタートアップ・ルーチンのテキスト領域の開始シンボルを指定します。	
	デフォルト	_@cstart
	変更方法	キーボードからの直接入力
	指定可能値	0 ～“*アドレス空間の終アアドレス*”のアドレス式
スタートアップ終了シンボル	スタートアップ・ルーチンのテキスト領域の終了シンボルを指定します。	
	デフォルト	_@cend
	変更方法	キーボードからの直接入力
	指定可能値	0 ～“*アドレス空間の終アアドレス*”のアドレス式

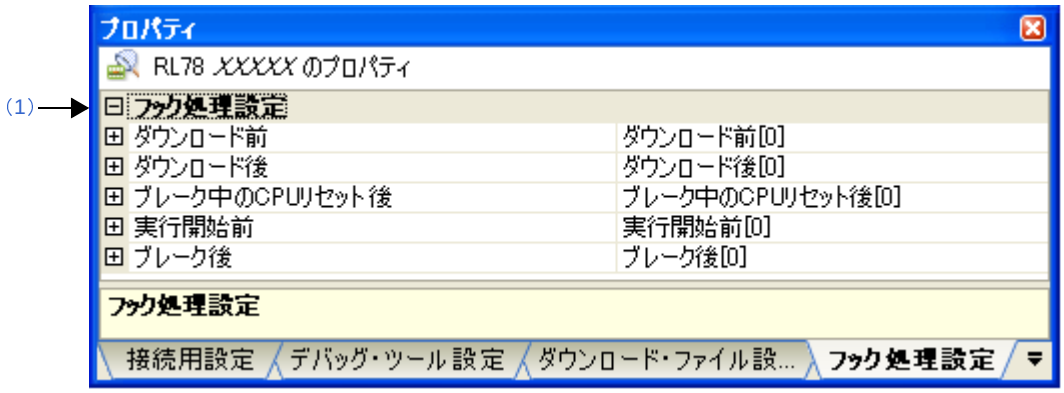
［フック処理設定］タブ

［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. 16　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	ダウンロード直前に行う処理を指定します。 ロード・モジュール・ファイルをダウンロードする直前に，ここで指定した SFR 名 / CPU レジスタ名を，指定した数値に自動的に書き換えます。	
	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	指定形式： - *SFR 名 /CPU レジスタ名* + 半角スペース + *数値* 最大指定文字数：　64 文字（1 処理につき） 最大指定処理数：　128 個

ダウンロード後	ダウンロード直後に行う処理を指定します。 ロード・モジュール・ファイルをダウンロードした直後に，ここで指定した SFR 名 / CPU レジスタ名を，指定した数値に自動的に書き換えます。	
	デフォルト	ダウンロード後 [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	指定形式： - *SFR 名 /CPU レジスタ名* + 半角スペース + *数値* 最大指定文字数：　64 文字（1 処理につき） 最大指定処理数：　128 個
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を指定します。 CPU リセット直後に，ここで指定した SFR 名 /CPU レジスタ名を，指定した数値に自動的に書き換えます。	
	デフォルト	ブレーク中の CPU リセット後 [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	指定形式： - *SFR 名 /CPU レジスタ名* + 半角スペース + *数値* 最大指定文字数：　64 文字（1 処理につき） 最大指定処理数：　128 個
実行開始前	実行開始直前に行う処理を指定します。 プログラムの実行開始直前に，ここで指定した SFR 名 /CPU レジスタ名を，指定した数値に自動的に書き換えます。	
	デフォルト	実行開始前 [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	指定形式： - *SFR 名 /CPU レジスタ名* + 半角スペース + *数値* 最大指定文字数：　64 文字（1 処理につき） 最大指定処理数：　128 個

ブレーク後	ブレーク直後に行う処理を指定します。 プログラムの実行がブレークした直後に，ここで指定した SFR 名 /CPU レジスタ名を，指定した数値に自動的に書き換えます。	
	デフォルト	ブレーク後 [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	指定形式： - *SFR 名 /CPU レジスタ名* + 半角スペース + *数値* 最大指定文字数：　64 文字（1 処理につき） 最大指定処理数：　128 個

エディタ パネル

テキスト・ファイル／ソース・ファイルの表示，編集を行います。

また，デバッグ・ツールと接続中で，ソース・ファイルを表示している場合では，ソース・レベル・デバッグ（「2. 7. 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【IECUBE】【シミュレータ】（「2. 13　カバレッジの測定【IECUBE】【シミュレータ】」参照）を行うことができます。

自動的にファイルのエンコード（Shift_JIS/EUC-JP/UTF-8）と改行コードを判別してオープンし，保存の際は元のフォーマットで保存します。ただし，ファイルの保存設定 ダイアログでエンコードと改行コードを指定した場合は，それに従って保存します。

このパネルは複数オープンすることができます（最大表示個数：100 個）。

備考 1. プロジェクトをクローズすると，該当プロジェクト内で登録されているファイルをオープンしているすべてのエディタ パネルがクローズします。

2. プロジェクトからファイルの登録を外すと，該当ファイルをオープンしているエディタ パネルがクローズします。

3. ソース・ファイルをオープンする際，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オープンするソース・ファイルの更新日時が新しい場合，メッセージを表示します。

図 A—15　エディタ パネル

図 A—16　エディタ パネル（コード・カバレッジ測定結果を表示した場合）

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（エディタ パネル専用部分）］
- ［［編集］メニュー（エディタ パネル専用部分）］
- ［コンテキスト・メニュー］

［オープン方法］

- デバッグ情報を持つロード・モジュール・ファイルのダウンロード直後，自動的にオープン
- プロジェクト・ツリー パネルにおいて，ファイルをダブルクリック
- プロジェクト・ツリー パネルにおいて，ファイルを選択したのち，コンテキスト・メニューの［開く］を選択
- プロジェクト・ツリー パネルにおいて，コンテキスト・メニューの［追加］→［新しいファイルを追加］を選択したのち，テキスト・ファイル／ソース・ファイルを作成
- 逆アセンブル パネル／コール・スタック パネル／トレース パネル【IECUBE】【シミュレータ】／イベント パネルにおいて，コンテキスト・メニューの［ソースへジャンプ］を選択
- カレント PC 値を強制的に変更した場合，またはプログラムの実行が停止した際に，カレント PC 値に対応するソース・テキスト行が存在する場合に自動的にオープン

[各エリアの説明]

(1) タイトルバー

オープンしているテキスト・ファイル／ソース・ファイルのファイル名を表示します。

なお，ファイル名の末尾に表示されるマークの意味は次のとおりです。

マーク	意味
*	編集中のテキスト・ファイルの内容が編集されている場合に表示します。
!	接続中のデバッグ・ツールにダウンロードされているロード・モジュールのデバッグ情報として含まれるテキスト・ファイルをオープンしている場合で，ダウンロードされているロード・モジュールの更新日時より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に表示します。
(編集不可)	書き込み禁止状態のテキスト・ファイルをオープンしている場合に表示します。
ID 番号	同一のテキスト・ファイルを複数オープンしている場合に表示します。

(2) イベント・エリア

デバッグ・ツールと接続中で，ダウンロードしたソース・ファイルを表示している場合のみ有効となるエリアです。

表示しているソース・テキスト行が対応するアドレスに変換可能な場合は，その行に対してイベントを設定することができ，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・マークが表示されます。また，設定したブレークポイントの詳細情報がイベント パネルに反映されます。

なお，すでにいずれかのイベント・マークが表示されている箇所において，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。

ブレークポイントの設定方法についての詳細は，「2. 8. 2　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

表示されている各種イベント・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。

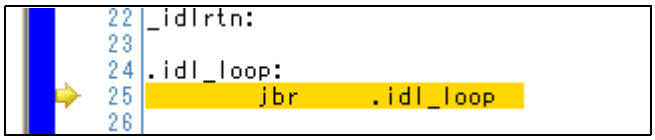
(c) ポップアップ表示

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。

備考　ダウンロードしたロード・モジュールの更新日時より，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，次のようにこのエリアをすべて警告色で表示します（警告色はオプション 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）。

図 A—17　警告色を示すイベント・エリア



(3) 行番号エリア

表示しているテキスト・ファイル／ソース・ファイルの行番号を表示します。

また，デバッグ・ツールと接続中で，ソース・ファイルを表示している場合では，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。

ただし，カレント PC マークは，カレント PC 値を変更した際，またはデバッグ・ツールが実行状態から停止状態に移行した際に，カレント PC 値がソース・テキスト行と対応する場合にのみ表示されます。

(4) 文字列エリア

テキスト・ファイル／ソース・ファイルの文字列の表示／編集を行います。

このエリアは，次の機能を備えています。

備考　オプション ダイアログの設定により，次の項目をカスタマイズすることができます。

- 表示フォント
- タブ幅
- コントロール・キャラクタ（空白記号を含む制御コード）の表示／非表示／色分け
- 予約語／コメントの色分け

(a) 文字列の編集

キーボードより，IME などの日本語入力システムを使用した文字列を入力することができます。

また，編集機能を充実させるための様々なショートカットキーを使用することができます。

(b) タグ・ジャンプ

現在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，［表示］メニュー→［タグ・ジャンプ］，またはコンテキスト・メニューの［タグ・ジャンプ］を選択することにより，該当するファイルを新たなエディタ パネルにオープンし，該当行／該当桁へジャンプすることができます（すでにオープンしている場合は該当エディタ パネルにジャンプ）。

タグ・ジャンプの動作の詳細については，「表 2—3　タグ・ジャンプの動作」を参照してください。

(c) カレント PC 行の表示

カレント PC 位置（PC レジスタ値）がソース・テキスト行と対応する場合，該当行を強調表示します（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）。

ただし，この機能は，デバッグ・ツールと接続中で，ソース・ファイルを表示している場合のみ有効となります。

(d) ブレークポイント設定行の表示

ブレークポイントが設定されている行を強調表示します（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）。

ただし，この機能は，デバッグ・ツールと接続中で，ソース・ファイルを表示している場合のみ有効となります。

(e) コード・カバレッジ測定結果の表示【IEUCBE】【シミュレータ】

カバレッジ機能を有効としている場合[注]，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. 13　カバレッジの測定【IECUBE】【シミュレータ】」を参照してください。

ただし，この機能は，デバッグ・ツールと接続中で，ソース・ファイルを表示している場合のみ有効となります。

注　【IECUBE】

カバレッジ機能は常に有効です。

(f) 変数値のポップアップ表示

変数上にマウス・カーソルを重ねることにより，変数値をポップアップ表示します。

この際の変数値の表示形式は，変数値の型に依存します（「表 A—9　ウォッチ式の表示形式（デフォルト）」と同等）。

ただし，この機能は，デバッグ・ツールと接続中で，ソース・ファイルを表示している場合のみ有効となります。

(g) 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］／［トレースの設定］／［タイマの設定］を選択することにより，現在キャレットのあるアドレス／行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ことにより，対応するイベント・マークがイベント・エリアに表示されます。また，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。

なお，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。

ただし，この機能は，デバッグ・ツールと接続中で，ソース・ファイルを表示している場合のみ有効となります。

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. 8. 3　変数 /SFR へのアクセスで停止する」
- 「2. 11. 3　任意区間の実行履歴を収集する」
- 「2. 11. 4　条件を満たした場合の実行履歴を収集する」
- 「2. 12. 2　任意区間の実行時間を計測する【IECUBE】【シミュレータ】」

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）。

(h) ウォッチ式の登録

表示されている C 言語変数 /CPU レジスタ /SFR/ 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，「(1) ウォッチ式を登録する」を参照してください。

ただし，この機能は，デバッグ・ツールと接続中で，ソース・ファイルを表示している場合のみ有効となります。

(i) ファイルの監視機能

ソース・ファイルを管理するために，次の監視機能を備えています。

- ソース・ファイルをオープンする際，ダウンロードされているロード・モジュールの更新日時よりオープンするソース・ファイルの更新日時が新しい場合，メッセージを表示します。
- ソース・ファイルを保存する際（［ファイル］メニュー→［*ファイル名*を保存］の選択），CubeSuite+ 以外によって，現在表示しているファイルの内容が変更されていた場合，ファイルを保存するか否かのメッセージを表示し，どちらかを選択することができます。

[[ファイル] メニュー（エディタ パネル専用部分）]

エディタ 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）。

*ファイル名*を閉じる	現在編集しているエディタ パネルをクローズします。 なお，パネルの内容が保存されていない場合は，確認メッセージを表示します。
*ファイル名*を保存	現在編集しているエディタ パネルの内容を上書き保存します。 なお，ファイルが一度も保存されていない，またはファイルが書き込み禁止の場合は，［名前を付けて *ファイル名*を保存］の選択と同等の動作となります。
名前を付けて *ファイル名*を保存 ...	現在編集しているエディタ パネルの内容を新規保存するために，名前を付けて保存 ダイアログをオープンします。
*ファイル名*の保存設定 ...	現在編集しているエディタ パネルでオープンしているファイルのエンコードと改行コードを変更するために，ファイルの保存設定 ダイアログをオープンします。
ページ設定 ...	Windows の印刷用ページ設定 を行うダイアログをオープンします。
印刷 ...	現在編集しているエディタ パネルの内容を印刷するために，Windows の印刷用 ダイアログをオープンします。

[[編集] メニュー（エディタ パネル専用部分）]

エディタ 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）。

元に戻す	前回行った操作をキャンセルし，文字とキャレット位置を元に戻します（最大 100 回まで）。
やり直し	前回行った［元に戻す］の操作をキャンセルし，文字とキャレット位置を元に戻します。
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。
コピー	選択範囲の文字列をクリップ・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，挿入モードの場合は挿入し，上書きモードの場合は上書きします。 ただし，クリップ・ボードの内容を文字列として認識できない場合は無効となります。
削除	キャレット位置の文字を 1 文字削除します。 ただし，範囲選択している場合は，選択されているの文字列を削除します。
すべて選択	現在編集中のテキストの先頭から最終まで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。 範囲選択している場合は，選択している範囲内での検索モードとなります。
置換 ...	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。 範囲選択している場合は，選択している範囲内での置換モードとなります。
移動 ...	指定した行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]

【文字列エリア／行番号エリア（デバッグ・ツールと切断時の場合）】

関数へジャンプ	選択している文字列，またはキャレット位置の単語を関数と判断し，該当する関数へジャンプします（「(4) 関数へジャンプする」参照）。
ジャンプ前の位置へ戻る	[関数へジャンプ] を実行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	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) タグ・ジャンプ」参照）。
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。
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。
貼り付け	クリップ・ボードの内容をキャレット位置に挿入します。
新しく開く	現在フォーカスのあるエディタ パネルと同じ内容で，新規にエディタ パネルをオープンします（新規にオープンしたエディタ パネルのタイトルバー上には，ファイル名とともに ID 番号が表示されます）。 なお，エディタ パネルは最大 100 個までオープンすることができます。

【文字列エリア／行番号エリア（デバッグ・ツールと接続中の場合）】

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）。 ただし，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無効となります。
アクション・イベントの登録 ...	キャレット位置の行に対応する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注。 ただし，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無効となります。
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注。 なお，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，下方向に有効な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＆デバッグ・ツールへダウンロード］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注。 ただし，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，プログラム実行中，または［ビルド＆デバッグ・ツールへダウンロード］実行中は無効となります。
移動 ...	指定した行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。
関数へジャンプ	選択している文字列，またはキャレット位置の単語を関数と判断し，該当する関数へジャンプします（「(4) 関数へジャンプする」参照）。
ジャンプ前の位置へ戻る	[関数へジャンプ] を実行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	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) タグ・ジャンプ」参照）。

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。 なお，イベントは，イベントの設定が可能な行のみ設定することができます（「(2) イベント・エリア」参照）。
ハード・ブレークの設定	キャレットのある行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（「2.8.2　任意の場所で停止する（ブレークポイント）」参照）注。
ソフト・ブレークの設定 （【シミュレータ】以外）	キャレットのある行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（「2.8.2　任意の場所で停止する（ブレークポイント）」参照）注。
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）/SFR に，リード・アクセスのブレーク・イベントを設定します（「(1) 変数 /SFR へのブレーク・イベントを設定する」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）/SFR に，ライト・アクセスのブレーク・イベントを設定します（「(1) 変数 /SFR へのブレーク・イベントを設定する」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）/SFR 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（「(1) 変数 /SFR へのブレーク・イベントを設定する」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 【IECUBE】【シミュレータ】	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。 なお，イベントは，イベントの設定が可能な行のみ設定することができます（「(2) イベント・エリア」参照）。
トレース開始の設定	キャレット位置の行が実行された際に，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（「2.11.3　任意区間の実行履歴を収集する」参照）注。 【シミュレータ】 プロパティ パネル上の［トレース］【IECUBE】【シミュレータ】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。
トレース終了の設定	キャレット位置の行が実行された際に，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（「2.11.3　任意区間の実行履歴を収集する」参照）注。 【シミュレータ】 プロパティ パネル上の［トレース］【IECUBE】【シミュレータ】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。
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）/SFR 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（「(1) 変数 /SFR へのアクセスが発生したとき」参照）。
トレース結果の表示	トレース パネル【IECUBE】【シミュレータ】をオープンし，取得したトレース・データの最終行を表示します。
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 【IECUBE】【シミュレータ】	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. 12. 2　任意区間の実行時間を計測する【IECUBE】【シミュレータ】」参照）。 なお，イベントは，イベントの設定が可能な行のみ設定することができます（「(2) イベント・エリア」参照）。
実行時にタイマ開始	キャレット位置の行が実行された際に，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。 【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。
実行時にタイマ終了	キャレット位置の行が実行された際に，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。 【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。
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	現在デバッグ・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。 ただし，使用するデバッグ・ツールがカバレッジ機能をサポートしていない場合，この項目は表示されません。
逆アセンブルへジャンプ	キャレット位置の行に対応するアドレスを逆アセンブル パネルでオープンします注。 ただし，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。
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。
貼り付け	クリップ・ボードの内容をキャレット位置に挿入します。
新しく開く	現在フォーカスのあるエディタ パネルと同じ内容で，新規にエディタ パネルをオープンします（新規にオープンしたエディタ パネルのタイトルバー上には，ファイル名とともに ID 番号が表示されます）。

注　ダウンロードされているロード・モジュールの更新日時よりオープンしているソース・ファイルの更新日が新しい場合，この項目を選択するとメッセージを表示します。

【イベント・エリア】（【シミュレータ】以外）

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］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］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）。

メモリ パネル

メモリの内容の表示，変更を行います（「2. 9. 1　メモリを表示／変更する」参照）。

また，選択しているマイクロコントローラがデータフラッシュ・メモリ内蔵品の場合では，データフラッシュ・メモリの内容も表示，変更が可能です。

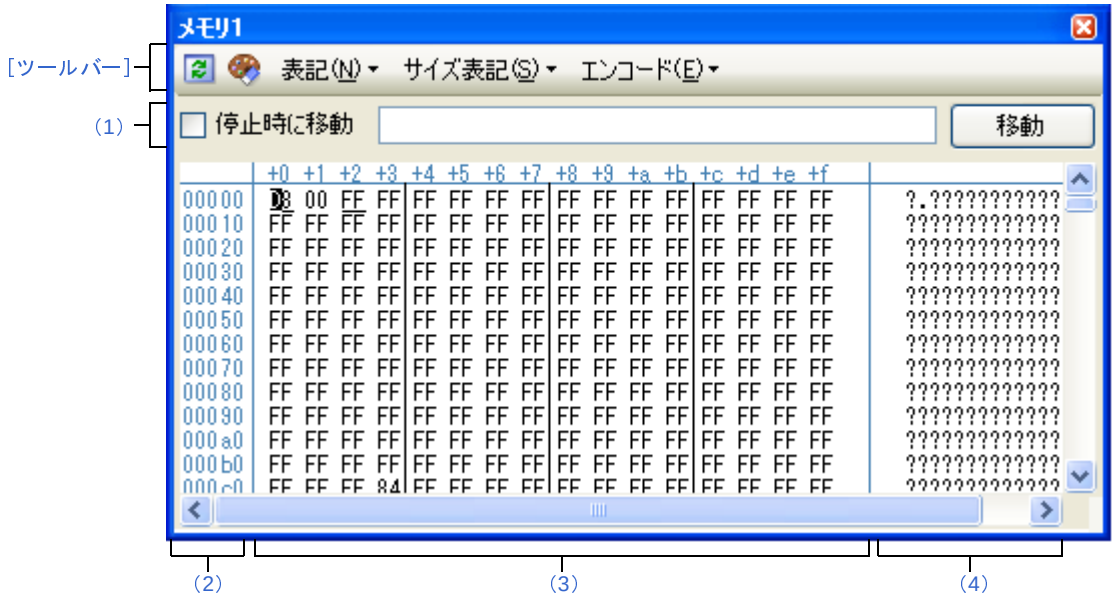
このパネルは，最大 4 個までオープンすることができます。各パネルは，タイトルバーの“メモリ 1”，“メモリ 2”，“メモリ 3”，“メモリ 4”の名称で識別されます。

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。

図 A—18　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。

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式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。

ただし，32 ビットで表現できる値より大きいアドレス式を指定することはできません。

(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。

備考　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・エリア

メモリのアドレスを表示します（16 進数表記固定）。

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。

(3) メモリ値エリア

メモリ値を表示／変更します。

デフォルトでは，“0”番地より表示を開始します。

メモリ値の表示進数／表示幅の指定は，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］の選択により行います（デフォルトでは，16 進数 /8 ビット幅で表示します）。

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	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	
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	
??	文字色	グレー	メモリがマッピングされていない領域，書き換え不可能領域（SFR 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	

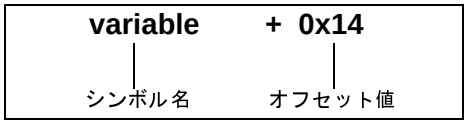
注　プログラム実行直前において，メモリ パネルで表示されていたアドレス範囲のメモリ値のみが対象となります。また，プログラムの実行前後での値の比較であるため，実行結果が同一値となった場合は強調表示を行いません。

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，「(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，「(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。

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。

(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，「(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，ツールバーのボタンのクリック，またはコンテキスト・メニューの［エンコード］の選択により行います。

このエリアは，次の機能を備えています。

(a) 文字列の変更

現在，文字コードとして［ASCII］が指定されている場合のみ，文字列を変更することができます。

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）。

(b) 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。 ただし，プログラム実行中は無効となります。
Hex	メモリ値を 16 進数で表示します（デフォルト）。
S Dec	メモリ値を符号付き 10 進数で表示します。
U Dec	メモリ値を符号なし 10 進数で表示します。
Oct	メモリ値を 8 進数で表示します。
Bin	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。 ただし，プログラム実行中は無効となります。
4	メモリ値を 4 ビット幅で表示します。
8	メモリ値を 8 ビット幅で表示します（デフォルト）。
16	メモリ値を 16 ビット幅で表示します。
32	メモリ値を 32 ビット幅で表示します。
64	メモリ値を 64 ビット幅で表示します。

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。 ただし，プログラム実行中は無効となります。
	文字列を ASCII コードで表示します（デフォルト）。
	文字列を Shift_JIS コードで表示します。
	文字列を EUC-JP コードで表示します。
	文字列を UTF-8 コードで表示します。

[[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：「(b) 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	キャレット位置の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：「(b) 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。 ただし，プログラム実行中は無効となります。
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。
符号なし 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。 ただし，プログラム実行中は無効となります。
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。
4 バイト	メモリ値を 32 ビット幅で表示します。
8 バイト	メモリ値を 64 ビット幅で表示します。
エンコード	文字列エリアの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。 ただし，プログラム実行中は無効となります。
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。
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. 7. 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【IECUBE】【シミュレータ】（「2. 13　カバレッジの測定【IECUBE】【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。

混合表示モードにすることにより，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. 7. 3　プログラムをステップ実行する」参照）。

図 A—19　逆アセンブル パネル（混合表示モードの場合）

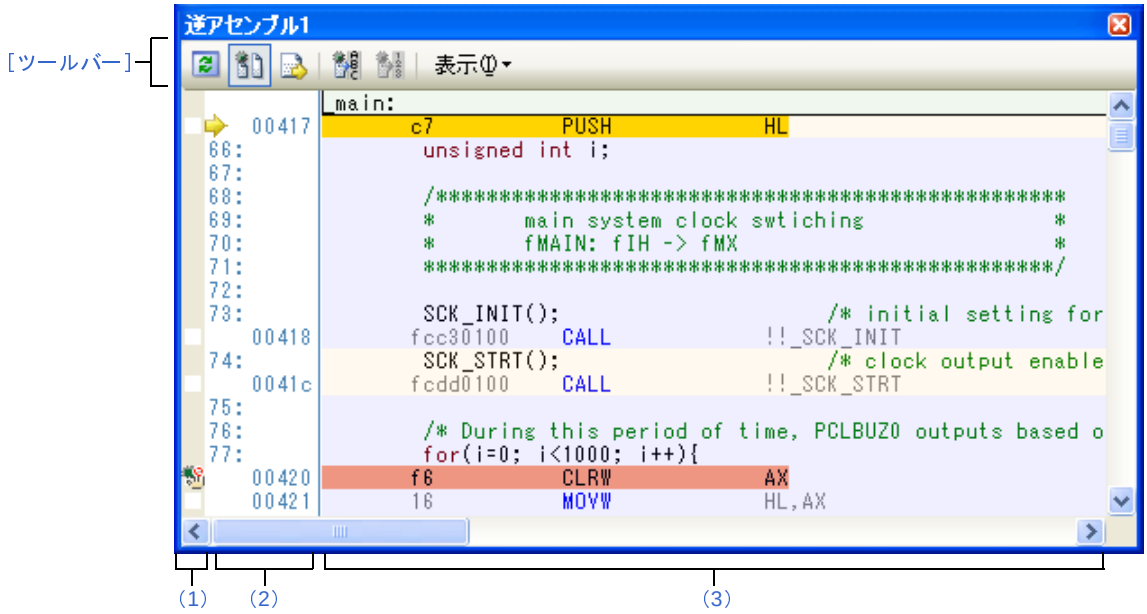


図 A—20　逆アセンブル パネル（混合表示モードを解除した場合）

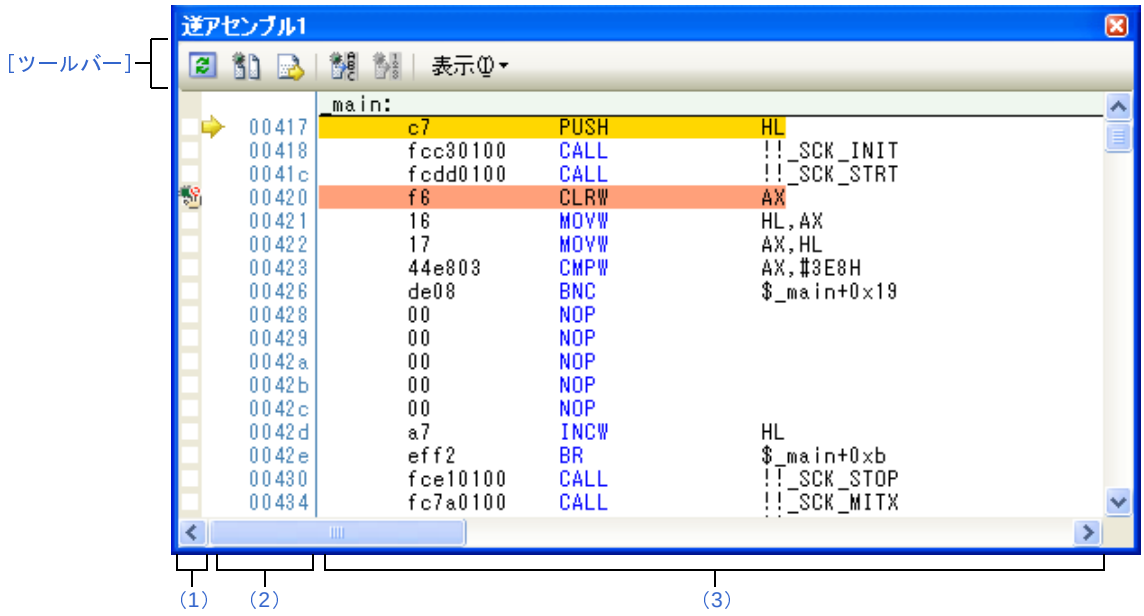
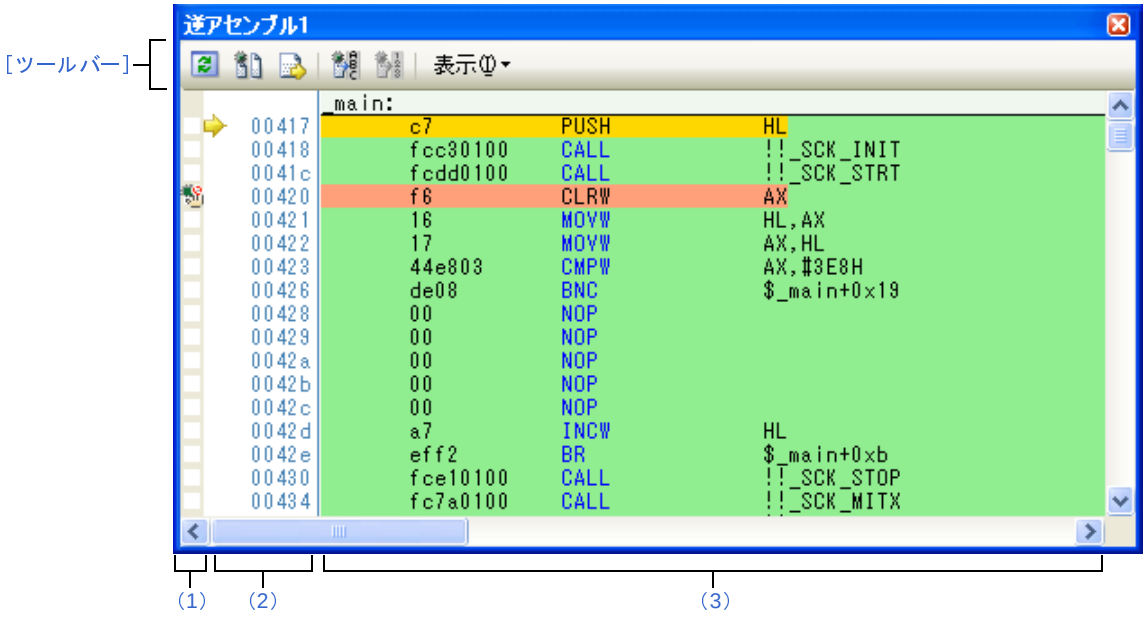


図 A—21　逆アセンブル パネル（コード・カバレッジ測定結果を表示した場合）



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・マークが表示されます。また，設定したブレークポイントの詳細情報がイベント パネルに反映されます。

なお，すでにいずれかのイベント・マークが表示されている箇所において，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。

ブレークポイントの設定方法についての詳細は，「2. 8. 2　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。

(a) ポップアップ表示

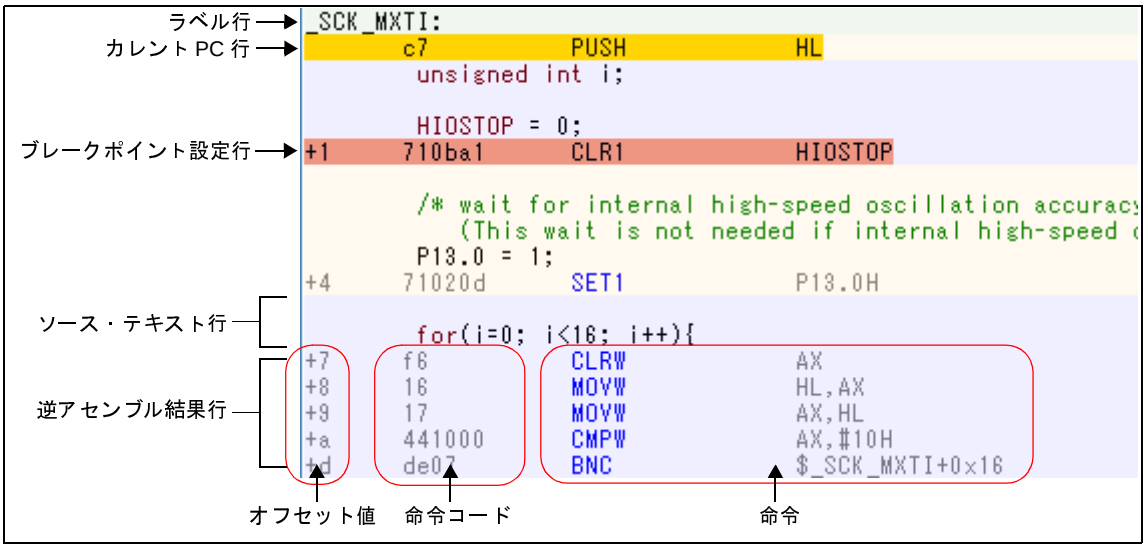
アドレス／ソース行番号にマウス・カーソルを重ねることにより，次の情報をポップアップ表示します。

アドレス	形式：　*＜ラベル名＞ + ＜オフセット値＞* 例 1：main + 0x10 例 2：subfunction + 0x20
ソース行番号	形式：　*＜ファイル名＞ # ＜行番号＞* 例 1：main.c#40 例 2：main.c#100

(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—22　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。

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）。

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

表示されている命令／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. 7. 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］を選択することにより，現在キャレットのあるアドレス／行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ことにより，対応するイベント・マークがイベント・エリアに表示されます。また，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，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。

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. 8. 3　変数 /SFR へのアクセスで停止する」
- 「2. 11. 3　任意区間の実行履歴を収集する」
- 「2. 11. 4　条件を満たした場合の実行履歴を収集する」
- 「2. 12. 2　任意区間の実行時間を計測する【IECUBE】【シミュレータ】」

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

表示されている C 言語変数 /CPU レジスタ /SFR/ 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，「(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【IEUCBE】【シミュレータ】

カバレッジ機能を有効としている場合[注]，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. 13　カバレッジの測定【IECUBE】【シミュレータ】」を参照してください。

注　【IECUBE】

カバレッジ機能は常に有効です。

(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，「(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。
	逆アセンブル結果とソース・テキストとの対応を表示する，混合表示モードに設定します（デフォルト）。
	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。

[[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）。

[[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]

【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＆デバッグ・ツールへダウンロード］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＆デバッグ・ツールへダウンロード］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。 なお，イベントは，イベントの設定が可能な行のみ設定することができます（「(1) イベント・エリア」参照）。
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. 8. 2　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 （【シミュレータ】以外）	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. 8. 2　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）/SFR に，リード・アクセスのブレーク・イベントを設定します（「(1) 変数 /SFR へのブレーク・イベントを設定する」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）/SFR に，ライト・アクセスのブレーク・イベントを設定します（「(1) 変数 /SFR へのブレーク・イベントを設定する」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）/SFR 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（「(1) 変数 /SFR へのブレーク・イベントを設定する」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 【IECUBE】【シミュレータ】	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。 なお，イベントは，イベントの設定が可能な行のみ設定することができます（「(1) イベント・エリア」参照）。
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. 11. 3　任意区間の実行履歴を収集する」参照）。 【シミュレータ】 プロパティ パネル上の［トレース］【IECUBE】【シミュレータ】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。
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. 11. 3　任意区間の実行履歴を収集する」参照）。 【シミュレータ】 プロパティ パネル上の［トレース］【IECUBE】【シミュレータ】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。
値をトレースに記録（読み書き時）	選キャレット位置，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）/SFR 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（「(1) 変数 /SFR へのアクセスが発生したとき」参照）。
トレース結果の表示	トレース パネル【IECUBE】【シミュレータ】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 【IECUBE】【シミュレータ】	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. 12. 2　任意区間の実行時間を計測する【IECUBE】【シミュレータ】」参照）。 なお，イベントは，イベントの設定が可能な行のみ設定することができます（「(1) イベント・エリア」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。 【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。
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。 【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。
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	現在デバッグ・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。 ただし，使用するデバッグ・ツールがカバレッジ機能をサポートしていない場合，この項目は表示されません。

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。
混合表示	逆アセンブル結果とソース・テキストとの対応を表示する，混合表示モードに設定します（デフォルト）。
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。

【イベント・エリア】（【シミュレータ】以外）

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］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］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）。

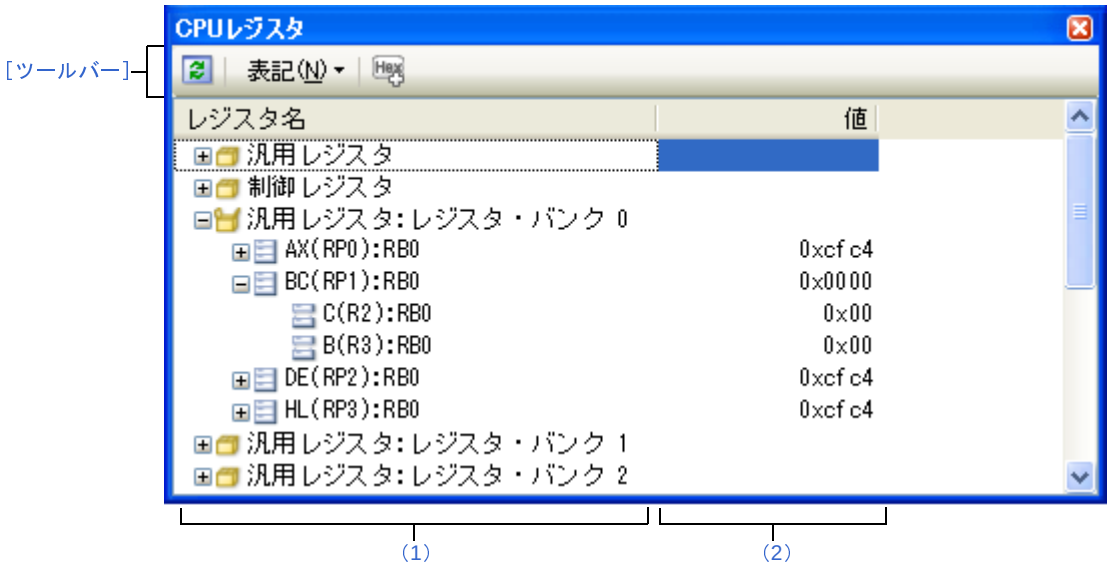
CPU レジスタ パネル

CPU レジスタ（汎用レジスタ／制御レジスタ）の内容の表示，および値の変更を行います（「2. 9. 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。

備考　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　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。

表示されるカテゴリ名／レジスタ名は次のとおりです（各レジスタ名の先頭の“+”マークの数は，表示される際の階層の深さを示します）。

表 A—3　CPU レジスタ パネルのカテゴリ名とレジスタ名

カテゴリ名	レジスタ名（別名）	ビット幅	説明
汎用レジスタ	+ AX(RP0)	16	汎用レジスタ（カレント・レジスタ・バンク）
	++ X(R0)	8	
	++ A(R1)	8	
	+ BC(RP1)	16	
	++ C(R2)	8	
	++ B(R3)	8	
	+ DE(RP2)	16	
	++ E(R4)	8	
	++ D(R5)	8	
	+ HL(RP3)	16	
	++ L(R6)	8	
	++ H(R7)	8	

カテゴリ名	レジスタ名（別名）	ビット幅	説明
制御レジスタ	+ PC	20	プログラム・カウンタ
	+ PSW	8	プログラム・ステータス・ワード
	++ IE	1	割り込み許可フラグ
	++ Z	1	ゼロ・フラグ
	++ RBS1	1	レジスタ・バンク選択フラグ
	++ AC	1	補助キャリー・フラグ
	++ RBS0	1	レジスタ・バンク選択フラグ
	++ ISP1	1	イン・サービス・プライオリティ・フラグ
	++ ISP0	1	イン・サービス・プライオリティ・フラグ
	++ CY	1	キャリー・フラグ
	+ SP	16	スタック・ポインタ
	+ ES	8	ES レジスタ
	+ CS	8	CS レジスタ
汎用レジスタ：レジスタ・バンク *n* 注	+ AX(RP0): レジスタ・バンク *n*	16	汎用レジスタ（レジスタ・バンク *n*）
	++ X(R0): レジスタ・バンク *n*	8	
	++ A(R1): レジスタ・バンク *n*	8	
	+ BC(RP1): レジスタ・バンク *n*	16	
	++ C(R2): レジスタ・バンク *n*	8	
	++ B(R3): レジスタ・バンク *n*	8	
	+ DE(RP2): レジスタ・バンク *n*	16	
	++ E(R4): レジスタ・バンク *n*	8	
	++ D(R5): レジスタ・バンク *n*	8	
	+ HL(RP3): レジスタ・バンク *n*	16	
	++ L(R6): レジスタ・バンク *n*	8	
	++ H(R7): レジスタ・バンク *n*	8	

注　“*n*”はレジスタ・バンクの番号を示します（*n*=0, 1, 2, 3）。

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，「（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。

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（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
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 レジスタ値をダブルクリックしたのち，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，「(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。
auto	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）。
Hex	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。
S Dec	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。
U Dec	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。
Oct	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。
Bin	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。
Asc	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。
Flo	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。
Dbl	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。
Hex	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。
符号なし 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。

SFR パネル

SFR の内容の表示，および値の変更を行います（「2. 9. 3　SFR 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　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 SFR 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 SFR 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］を選択してください。

備考　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—24　SFR 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（SFR パネル専用部分）］
- ［［編集］メニュー（SFR 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→［SFR］を選択

[各エリアの説明]

(1) [SFR] エリア

SFR の種別をカテゴリ（フォルダ）として分類し，各 SFR 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 SFR 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 SFR 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。
	SFR 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 SFR 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。
	SFR 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 SFR 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 SFR 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 SFR 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，SFR 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 SFR 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 SFR 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。

2.　SFR の追加／削除はできません。

(b) ウォッチ式の登録

SFR/ 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，「(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 SFR がウォッチ式として登録されます。

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

SFR 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。

SFR 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（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
0x0	文字色	青色	ユーザにより，値が変更されている SFR 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 SFR 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 SFR 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 SFR を示します。

読み込み保護対象の SFR 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 SFR と，1 バイト /2 バイト SFR に割り付けられている 1 ビット SFR 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 SFR 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) SFR 値の変更

SFR の値の変更は，対象 SFR 値をダブルクリックしたのち，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）。

SFR 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。

ただし，この操作は，プログラムが停止状態の場合のみ有効となります。

また，SFR の値には，次のように ASCII 文字による入力も可能です。

例 1. SFR 名“P0”の値に“0x41”を書き込んだ場合
→ポート P0 に，“0x41”が書き込まれます。

2. SFR 名“P0”の値に ASCII 文字“'A'”を書き込んだ場合
→ポート P0 に，“0x41”が書き込まれます。

注意　読み込み専用の SFR の値を変更することはできません。

(b) SFR 値の保存

[ファイル] メニュー→[名前を付けて SFR 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，SFR 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。

SFR 値の保存方法についての詳細は，「(5) SFR 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

各 SFR の型情報を次の形式で表示します。

- ＜*SFR の種類*＞ [＜*アクセス属性*＞ ＜*すべての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
すべてのアクセス可能サイズ	アクセス可能なサイズをビット単位で小さい順に“,”で区切り列挙します。	
サイズ	SFR 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. 「SFR [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 SFR

2. 「SFR [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 SFR

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。

(4)［アドレス］エリア

各 SFR 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

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 SFR 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 SFR 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（SFR 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。
Hex	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）。
S Dec	選択している項目の値を符号付き 10 進数で表示します。
U Dec	選択している項目の値を符号なし 10 進数で表示します。
Oct	選択している項目の値を 8 進数で表示します。
Bin	選択している項目の値を 2 進数で表示します。
Asc	選択している項目の値を ASCII コードで表示します。
Hex	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（SFR パネル専用部分）]

SFR 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。

SFR 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) SFR 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 SFR 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 SFR 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) SFR 値の保存」参照）。

[[編集] メニュー（SFR パネル専用部分）]

SFR 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（SFR/ 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。 SFR/ 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（SFR/ 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（SFR 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 SFR/ 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・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 SFR 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 SFR/ 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 SFR 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 SFR 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。 SFR/ 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。
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（SFR 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。
符号なし 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 SFR 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。

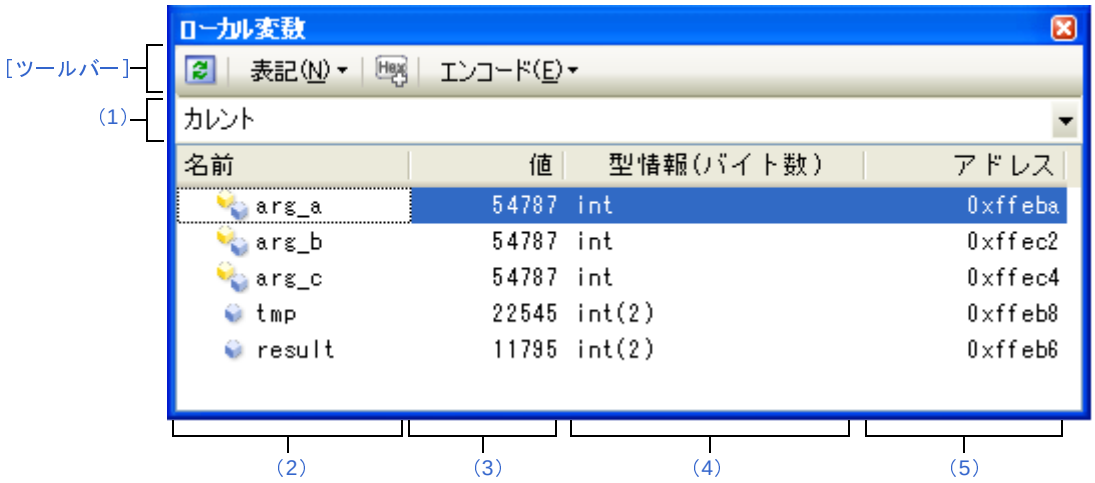
ローカル変数 パネル

ローカル変数の内容の表示，および値の変更を行います（「2.9.5　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　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。

備考　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　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1, 2	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 1. 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. 信頼性を喪失したスコープ情報の場合，リスト内の項目を無効色で表示します（無効色は，オプション ダイアログにおける［全般 - フォントと色］カテゴリカテゴリの設定に依存します）。

(2) [名前] エリア

ローカル変数名や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，「(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（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
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) 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，「(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］

ツールバーの各ボタンは，プログラム実行中は無効となります。

	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。
auto	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。
Hex	このパネル上の値を 16 進数で表示します。
Dec	このパネル上の値を 10 進数で表示します。
Oct	このパネル上の値を 8 進数で表示します。
Bin	このパネル上の値を 2 進数で表示します。
Dec	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）。
Hex	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。
Flo	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。
Dbl	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。
Hex	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。
Asc	文字列変数を ASCII コードで表示します（デフォルト）。
SJIS	文字列変数を Shift_JIS コードで表示します。
EUC	文字列変数を EUC-JP コードで表示します。
UTF	文字列変数を UTF-8 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) 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]

コンテキスト・メニューの各項目は，プログラム実行中は無効となります。

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。
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（「2.9.6　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/SFR 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）。

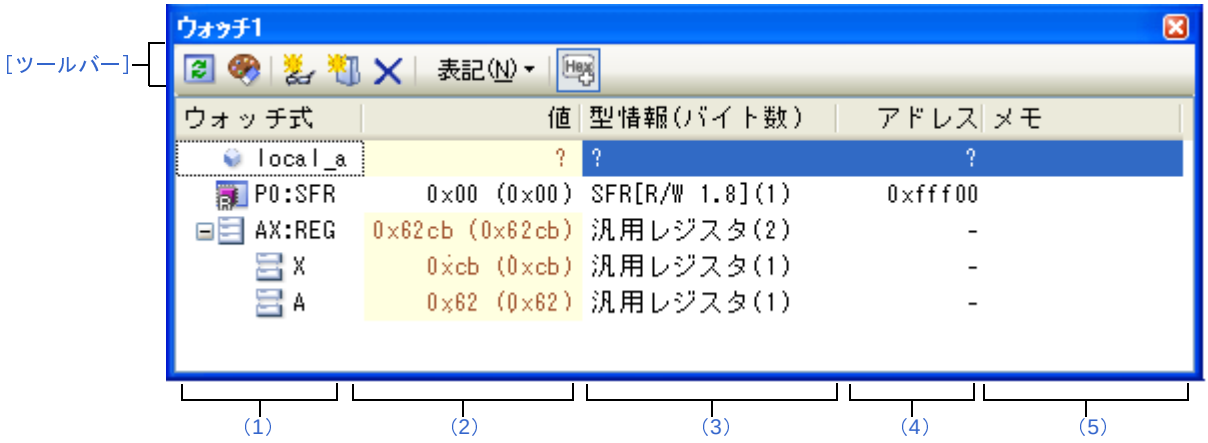
プログラムの実行後，ウォッチ式の値が変化すると表示を自動的に更新します（ステップ実行時には，ステップ実行ごとに表示を逐次更新）。

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。

備考　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—26　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 SFR 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。

2.　1 つのウォッチ パネルにおいて，カテゴリは 64 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）AX レジスタの場合 A レジスタ X 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。

- 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—4　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 /SFR/ 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 /SFR/ 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）
SFR パネル	SFR 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。

- 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。

この際のウォッチ式の入力形式は次のとおりです。

表 A—5　ウォッチ式の入力形式

ウォッチ式	表示する値
C 言語変数名	C 言語の変数の値
ウォッチ式 [*ウォッチ式*]	配列の要素値
ウォッチ式 . メンバ名	構造体／共用体のメンバ値
ウォッチ式 -> メンバ名	ポインタの指し示す構造体／共用体のメンバ値
* *ウォッチ式*	ポインタの変数の値
CPU レジスタ名	CPU レジスタの値
SFR 名	SFR の値
ラベル，EQU シンボル，即値アドレス	ラベルの値，EQU シンボルの値，即値アドレスの値
ビット・シンボル	ビット・シンボル の値

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。

表 A—6　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
prog$file#func#var	prog	file	func	var
prog$file#var	prog	file	グローバル	var
prog$var	prog	グローバル	グローバル	var
file#func#var	カレント	file	func	var
file#var	カレント	file	グローバル	var
var	カレント	カレント	カレント	var

表 A—7　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
AX:RB0	レジスタ・バンク 0	AX
AX:REG	カレント・レジスタ・バンク	AX

表 A—8　SFR 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	SFR 名
P0:SFR	P0
P0	P0

備考 1. 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。

2. 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。

3. 同名の C 言語変数 /CPU レジスタ /SFR 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＞ SFR

また，ウォッチ式の先頭に“$”を指定した場合では，次の順にシンボルを解決し，値を表示します。

CPU レジスタ＞ SFR ＞ C 言語変数

4. 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 値のスコープを元にシンボルを解決し，値を表示します。

5. SFR 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。

- 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 /SFR/ 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 つのウォッチ パネルにおいて，ウォッチ式は 128 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）。

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。

2. ウォッチ式は，同名を複数登録することができます。

（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 /SFR の場合にのみ行うことができます。

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. 8. 3　変数 /SFR へのアクセスで停止する」
- 「2. 11. 4　条件を満たした場合の実行履歴を収集する」

（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 SFR/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—9　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（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
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	

表示例（デフォルト）			説明
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（変数がスコープを外れた場合など）
	背景色	標準色	

備考 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 SFR は，読み込み保護対象となり，値の読み込みは行いません。読み込み保護対象の SFR の内容を読み込みたい場合には，コンテキスト・メニューの［値を強制読み込み］を選択してください。

2. 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 SFR 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。

3. 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，「(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，「(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，「(8) 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 SFR	＜*SFR の種類*＞（＜*アクセス属性*＞＜*アクセス・タイプ*＞＜*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・タイプ	1：　1 ビット・アクセス可能 8：　バイト・アクセス可能 16：　ワード・アクセス可能
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 SFR/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。

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 SFR 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 SFR 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 128 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 64 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。
	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—9　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）。
	選択している項目の値を 16 進数で表示します。
	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。
	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 SFR 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（「(1) 変数 /SFR へのブレーク・イベントを設定する」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 【IECUBE】【シミュレータ】	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 SFR 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（「(1) 変数 /SFR へのアクセスが発生したとき」参照）。 イベント設定後，対象となるウォッチ式の先頭にイベント・マークが表示されます。
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【IECUBE】【シミュレータ】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 SFR 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 SFR 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 128 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 64 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。

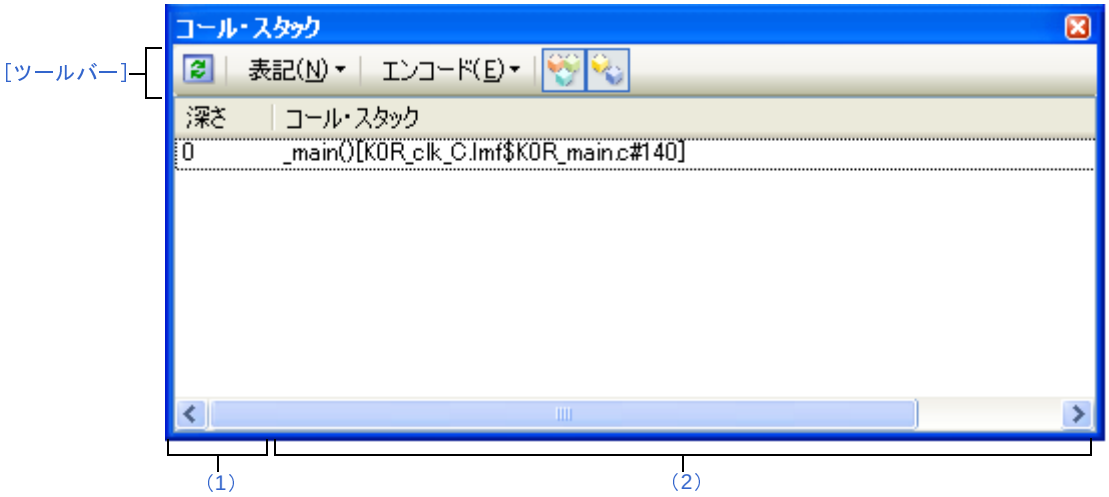
コピー	選択している範囲の文字列をクリップ・ボードにコピーします。 ウォッチ式／カテゴリが選択状態の場合は，その項目をコピーします。
貼り付け	テキストが編集状態の場合，クリップ・ボードの内容をキャレット位置に挿入します。 テキストが編集状態以外の場合で，ウォッチ式がクリップ・ボードにコピーされている場合は，コピーされているウォッチ式をキャレット位置に登録します。 ただし，ウォッチ式の展開項目を選択している場合は無効となります。
表記	表示形式を指定するため，次のカスケード・メニューを表示します。
自動	選択している項目の表記を変数ごとの規定値（「表 A—9　ウォッチ式の表示形式（デフォルト）」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。
符号なし 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—9　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—9　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（デフォルト）。
Shift_JIS	選択している項目を Shift_JIS コードで表示します。
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。
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. 10. 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　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 A—27　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 1.　信頼性を喪失したスタック・データより古い情報を取得した場合，その行の情報を無効色で表示します（無効色は，オプション ダイアログにおける［全般 - フォントと色］カテゴリカテゴリの設定に依存します）。

2.　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，現在のキャレット位置の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，「(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]

ツールバーの各ボタンは，プログラム実行中は無効となります。

	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。
auto	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）。
Hex	このパネル上の値を 16 進数で表示します。
Dec	このパネル上の値を 10 進数で表示します。
Oct	このパネル上の値を 8 進数で表示します。
Bin	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。
Asc	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）。
SJIS	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	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	このパネル上の文字列変数を UTF-8 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]

コンテキスト・メニューの各項目は，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。
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【IECUBE】【シミュレータ】

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. 11　実行履歴の収集【IECUBE】【シミュレータ】」参照）。

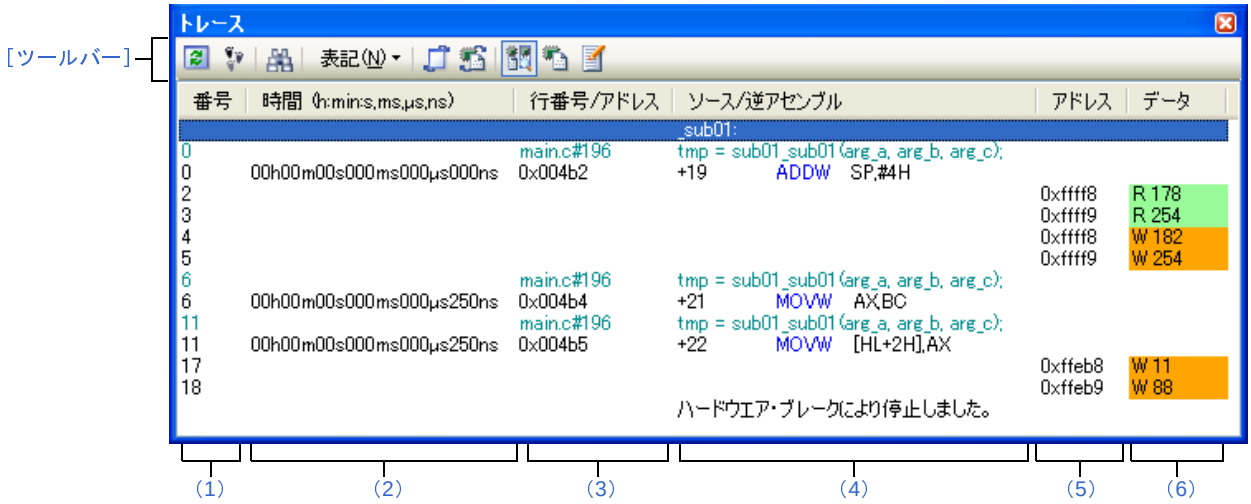
トレース・データは，デフォルトで逆アセンブル・テキストとソース・テキストを混合して表示しますが，表示モードを選択することにより，そのどちらか一方のみを表示させることもできます。

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。

備考　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　トレース 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（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

位置を示すトレース番号を表示します。

(2)［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。

なお，オーバフローした場合，このエリアは無効色（グレー）で表示されます。

備考 1. 【IECUBE】

時間の精度は，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・タグの分周率］プロパティの設定に依存します。

2. 【シミュレータ】

時間表示を積算値とするか差分値とするかは，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。

(3)［行番号 / アドレス］エリア

アセンブル命令のアドレス，またはソース・ファイルの行番号を表示します。

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
命令（逆アセンブル）	*＜アドレス＞*
ソース・テキスト	*＜ファイル名＞ # ＜行番号＞*
ラベル	－
ポイント・トレース結果	－
ブレーク要因	－

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。

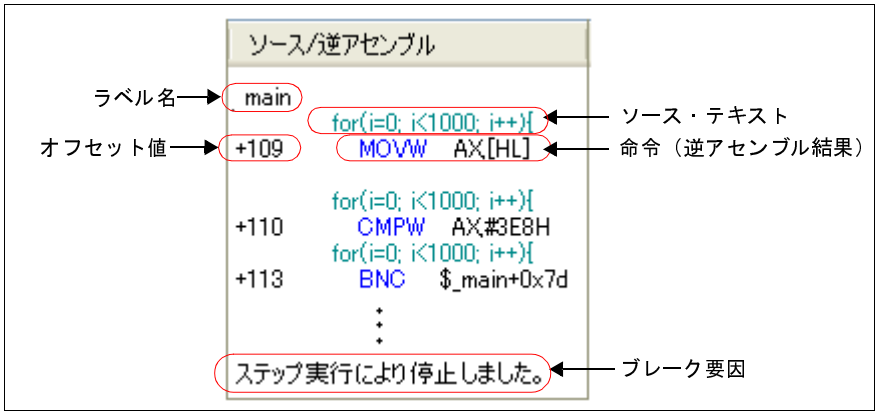
- CPU レジスタ・アクセス
- オペランド・アクセス
- 無効フェッチ

(4)［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—29 ［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。 なお，ソース行の実行時にアクセスされた変数[注1]/SFR 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* >>> - <<<*SFR 名 =SFR 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。 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。
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	プログラムがブレークした要因を表示します。
リセット要因 【IECUBE】	CPU リセットが発生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，2 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。

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／リセット要因【IECUBE】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／リセット要因【IECUBE】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／リセット要因【IECUBE】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. 11. 8　実行履歴の表示内容を保存する」を参照してください。

(5)［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，SFR へのアクセスの場合は，アドレスの代わりに SFR 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。

(6)［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（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）			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	

備考　【IECUBE】

DMA による SFR/ メモリへのアクセスの場合，“*データ値*”の直後に“DMA”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。 ただし，プログラム実行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，プログラム実行中は無効となります。
	トレース検索 ダイアログ【IECUBE】【シミュレータ】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，プログラム実行中は無効となります。
Hex	このパネル上の値を 16 進数で表示します（デフォルト）。
Dec	このパネル上の値を 10 進数で表示します。
Oct	このパネル上の値を 8 進数で表示します。
Bin	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，プログラム実行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，プログラム実行中は無効となります。
	表示モードをソース表示モードにします。 ただし，プログラム実行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。
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）。

[[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）。
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）。
検索 ...	トレース検索 ダイアログ【IECUBE】【シミュレータ】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，プログラム実行中は無効となります。
検索 ...	トレース検索 ダイアログ【IECUBE】【シミュレータ】をオープンします。 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，プログラム実行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，プログラム実行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，プログラム実行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，プログラム実行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，プログラム実行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	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。
逆アセンブルと連動	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。
逆アセンブルへジャンプ	選択している行のフェッチ・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	選択している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。

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. 15　イベントの管理」参照）。

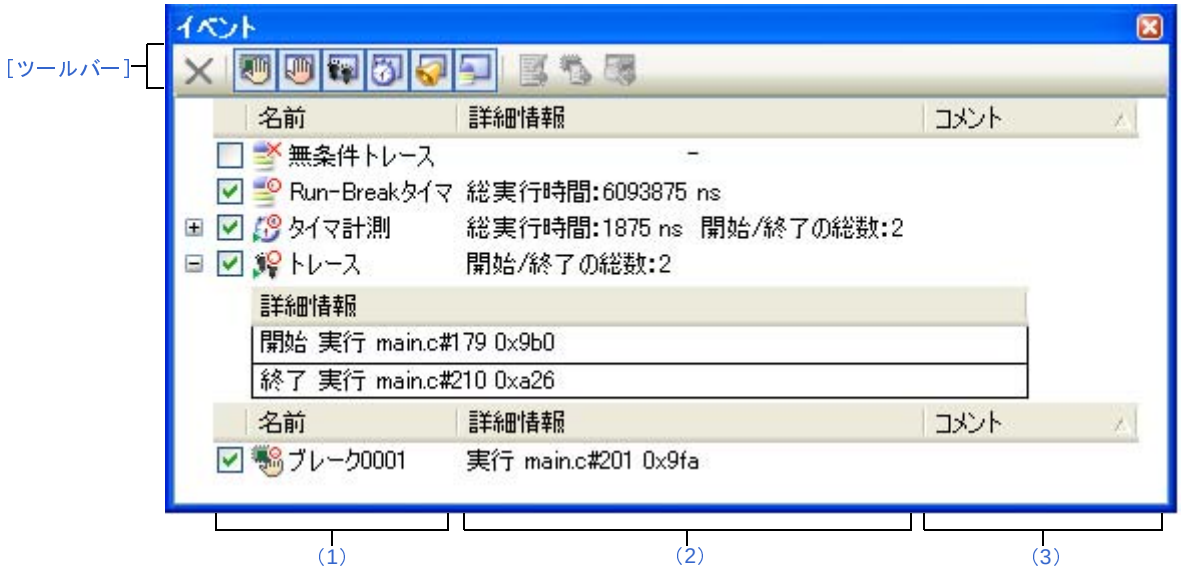
なお，このパネルは，デバッグ・ツールと接続時のみオープンすることができます。

備考 1. イベントの設定に関しては，「2. 15. 6　イベント設定に関する留意事項」を参照してください。

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されます。

3. 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—30　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］

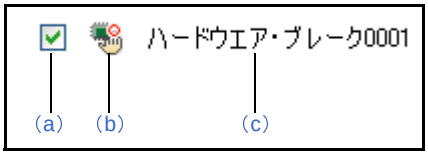
- ［表示］メニュー→［イベント］を選択
- 【IECUBE】【シミュレータ】

  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。

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。

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—10　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				—
ソフトウエア・ブレーク (【シミュレータ】以外)				—
無条件トレース 【IECUBE】【シミュレータ】			—	—
Run-Break タイマ		—	—	—
トレース 【IECUBE】【シミュレータ】				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 【IECUBE】 【シミュレータ】				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 【IECUBE】【シミュレータ】				—
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。

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。

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—11　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. 8. 2　任意の場所で停止する（ブレークポイント）」参照 →「2. 8. 3　変数 /SFR 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）（【シミュレータ】以外）	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. 8. 2　任意の場所で停止する（ブレークポイント）」参照
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. 11. 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. 12. 1　実行開始から停止までの実行時間を計測する」参照
トレース【IECUBE】【シミュレータ】	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. 11. 3　任意区間の実行履歴を収集する」参照
タイマ計測【IECUBE】【シミュレータ】	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）。 →「2. 12. 2　任意区間の実行時間を計測する【IECUBE】【シミュレータ】」参照
ポイント・トレース【IECUBE】【シミュレータ】	プログラムの実行により，指定した変数 /SFR 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. 11. 4　条件を満たした場合の実行履歴を収集する」参照

イベント種別	説明
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. 14. 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) 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。

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。

備考　上記のイベント種別のほか，解析ツールの関数 パネル／変数 パネルでイベント（ブレークポイント／ブレーク・イベント）を設定した場合，次のイベント種別を表示します。

- 関数へのブレークポイントの場合：　“関数への先頭へのブレーク”
- 変数へのブレーク・イベントの場合：　“変数のアクセス・ブレーク”

(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—12　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　ー　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　ー　0x300

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：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#variable1　0x100 - 0x101 == 0x5
		ライト　sub.c#variable2　0x200 - 0x200 == 0x7
		リード／ライト　sub2.c#variable3　0x300 - 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#variable1　0x100 - 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　variable1　0x100 - 0x101 == 0x10
ソフトウエア・ブレーク （【シミュレータ】以外）	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 【IECUBE】 【シミュレータ】	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） 【IECUBE】 【シミュレータ】	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#100　0x300 - 開始　実行後　funcA + 0x100　0x300 - 終了　実行後　main.c#200　0x100 - 終了　実行後　funcA + 0x10　0x100

イベント種別	表示内容注1	
タイマ計測 （発生条件：実行系） 【IECUBE】 【シミュレータ】	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#100　0x300 - 開始　実行後　funcA + 0x30　0x100 - 終了　実行後　main.c#100　0x300 - 終了　実行後　funcA + 0x50　0x100
ポイント・トレース （発生条件：アクセス系） 【IECUBE】 【シミュレータ】	表示形式1	＜*発生条件*＞ ＜*変数名*＞ ＜*アドレス*（*範囲*）＞
	表示例	リード　variable1　0x100 - 0x101
	表示形式2	＜*発生条件*＞ ＜*ファイル名 # 変数名*＞ ＜*アドレス*（*範囲*）＞
	表示例	ライト　sub.c#variable2　0x200 - 0x200
	表示形式3	＜*発生条件*＞ ＜*ファイル名* # 関数名 # *変数名*＞ ＜*アドレス*（*範囲*）＞
	表示例	リード／ライト　sub.c#func1#variabl30x300 - 0x303
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　0x100　aaa, bbb, ccc
		実行後　sub.c#100　0x200　aaa 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。 【シミュレータ】以外 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。 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. 12. 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。
＜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 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. 12. 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。
＜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 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）。
＜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 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）。
＜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 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）。
＜ 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。

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）。
（【シミュレータ】以外）	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）。
【IECUBE】【シミュレータ】	トレース関連のイベントを表示します（デフォルト）。
【IECUBE】【シミュレータ】	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。
無効化	選択しているイベントを無効状態にします。

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。
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。
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。

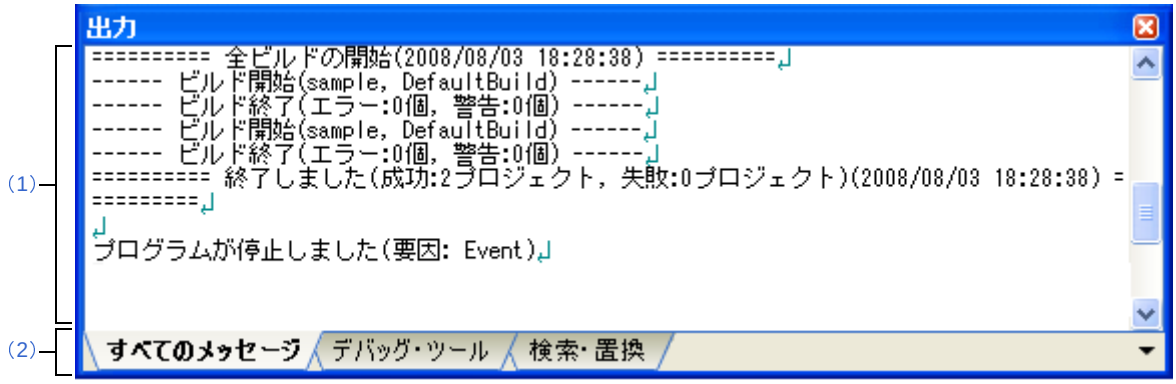
注　トレース・イベント／タイマ計測イベント／ビルトイン・イベント（無条件トレース・イベント /Run-Break タイマ・イベント）以外のイベントが対象となります。

出力 パネル

CubeSuite+ 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／解析ツールなど）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. 14. 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。

図 A—31　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。

検索結果（一括検索）の表示では，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（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）。

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	CubeSuite+ 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／解析ツールなど）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	CubeSuite+ 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／解析ツールなど）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。
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・マッピングの設定をメモリ種別ごとに行います。

注意　デバッグ・ツールと切断中の場合では，ユーザにより追加されたメモリ・マッピング領域のみが表示対象となります。デバッグ・ツールと接続することにより（「2. 4. 1　CubeSuite+ にデバッグ・ツールを接続する」），各メモリ種別ごとの詳細表示を行います。

図 A—32　メモリ・マッピング ダイアログ（【シミュレータ】以外）

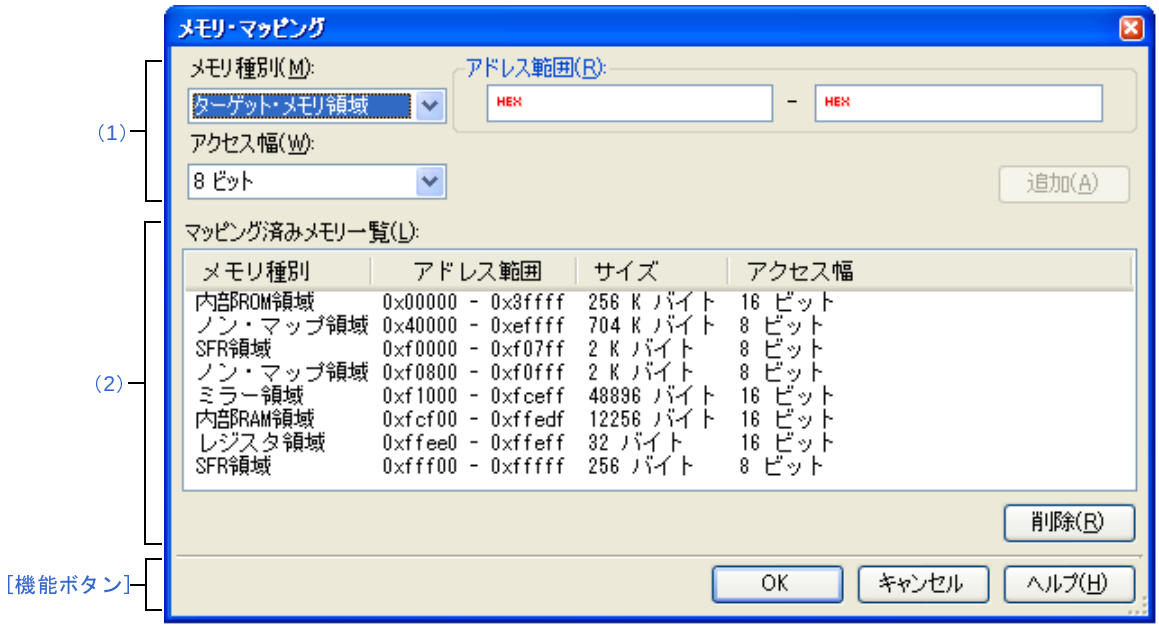
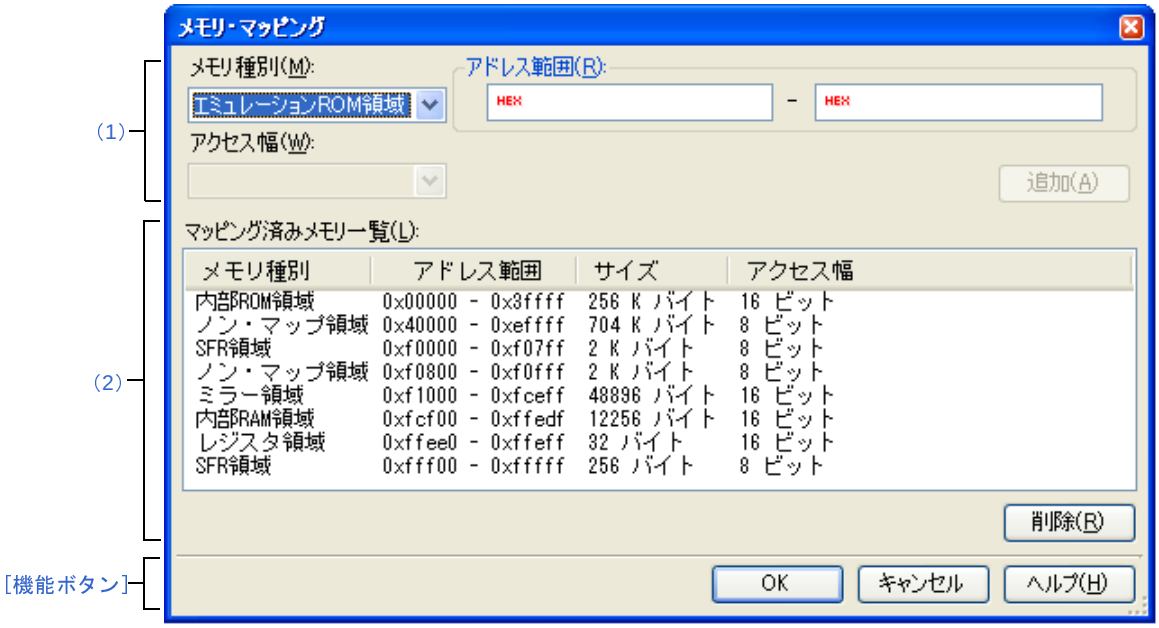


図 A—33　メモリ・マッピング ダイアログ【シミュレータ】



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］
- ［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

新たに追加するメモリ・マッピングの情報を指定します。

（a）［メモリ種別］

追加するメモリ・マッピングのメモリ種別を次のドロップダウン・リストより選択します（デフォルトで選択される項目は使用するデバッグ・ツールに依存）。

エミュレーション ROM 領域 【シミュレータ】	エミュレーション ROM 領域を追加します。 シミュレータ代替 ROM を使用します。
エミュレーション RAM 領域 【シミュレータ】	エミュレーション RAM 領域を追加します。 シミュレータ代替 RAM を使用します。
ターゲット・メモリ領域	ターゲット・メモリ領域を追加します。
I/O 保護領域	I/O 保護領域を追加します注。

注　I/O 保護領域は，デバッグ機能によるアクセスを保護する領域です。
I/O 保護領域に設定されたアドレス範囲は，メモリ パネルなどからアクセスできません（ロード・モジュールの実行によるアクセスのみ可能）。I/O 保護領域は，ターゲット・メモリ領域として指定されている範囲内でのみ設定可能です。

備考　マイクロコントローラの外部メモリ領域と使用禁止領域（ガード領域）は，ノン・マップ領域として扱われます。このため，使用禁止領域に重なってマッピングを指定した場合，ノン・マップ領域と重なることを警告するメッセージを表示します。
なお，外部メモリ領域と使用禁止領域のマッピング情報については，使用するマイクロコントローラのマニュアルを参照してください。

なお，メモリ・マッピングが可能な属性／サイズは次のとおりです。

表 A—13　設定可能なメモリ・マッピング属性

属性	デバッグ・ツール		
	IECUBE	E1/E20 EZ Emulator	シミュレータ
エミュレーション ROM 領域	○	－	○
エミュレーション RAM 領域	－	－	○
ターゲット・メモリ領域	○注	○	○
I/O 保護領域	○	○	○

○ ：可能（マッピング単位：2 バイト）

－ ：不可

注　ターゲット・メモリ領域は合計 4 つまでマッピング可能

(b)［アドレス範囲］

追加するメモリ・マッピングの開始アドレスと終了アドレスを指定します。それぞれのテキスト・ボックスに，16 進数を直接入力します。

ただし，次の指定の場合，新たにメモリ・マッピングを追加することはできません（このエリアの［追加］ボタンをクリックした際に，メッセージを表示します）。

- メモリ種別として［ターゲット・メモリ領域］を選択した際に，指定したアドレス範囲が他のメモリ領域と重複している場合
- メモリ種別として［I/O 保護］を選択した際に，指定したアドレス範囲が 1 つのターゲット・メモリ領域内で収まらない場合

(c)［アクセス幅］（【シミュレータ】以外）

追加するメモリ・マッピングのアクセス幅を次のドロップダウン・リストより選択します（直接入力不可）。

なお，メモリ種別として［I/O 保護領域］を選択した場合，アクセス幅は，対象となるターゲット・メモリ領域のアクセス幅と同値にする必要があります。

8 ビット	追加するメモリ・マッピングのアクセス幅を 8 ビットにします（デフォルト）。
16 ビット	追加するメモリ・マッピングのアクセス幅を 16 ビットにします注。

注　メモリ・マッピングに追加される際に，終了アドレスに指定したアドレス値の端数を 2 の倍数に自動的に切り上げます。

(d) ボタン

ボタン	機能
追加	このエリアで指定した内容をメモリ・マッピングに追加します。 追加されたメモリ・マッピングは，［マッピング済みメモリ一覧］エリアに表示されます。 なお，［OK］ボタンを押下するまでは，変更内容の設定は行われません。

(2)［マッピング済みメモリ一覧］エリア

(a) 一覧の表示

追加メモリ・マッピング指定エリアで追加したメモリ・マッピングと，マイクロコントローラ内のメモリ・マッピングの情報を表示します。このエリアを編集することはできません。

メモリ種別	次のメモリ種別を表示します。 - 内部 ROM 領域注1 - ミラー領域 - 内部 RAM 領域 - データフラッシュ領域注2（【シミュレータ】以外） - その他の RAM 領域 - SFR 領域 - ターゲット・メモリ領域 - エミュレーション ROM 領域【シミュレータ】 - エミュレーション RAM 領域【シミュレータ】 - ノン・マップ領域 - I/O 保護領域
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3）。
アクセス幅	アクセス幅を表示します（単位：ビット）。

注 1. 選択しているマイクロコントローラが ROM レス品の場合は表示しません。
ただし，デバッグ・ツールにエミュレーション内部 ROM 領域が存在する場合で，プロパティ パネルの［接続用設定］タブ上の［内部 ROM/RAM］カテゴリ内［内部 ROM サイズ [K バイト]］プロパティに“0”より大きな値を設定した場合にかぎり，“内部 ROM 領域”を表示します。

2. 選択しているマイクロコントローラがデータフラッシュ・メモリ内蔵品の場合のみ表示します。

3. 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	このエリアで選択しているメモリ・マッピングを削除します。 削除できるメモリ領域は，ターゲット・メモリ領域/I/O 保護領域／エミュレーション ROM 領域【シミュレータ】／エミュレーション RAM 領域【シミュレータ】のいずれかです（マイクロコントローラ内のメモリ・マッピングを削除することはできません）。 ただし，I/O 保護領域が設定されているターゲット・メモリ領域を削除しようとした場合は，メッセージを表示し，［OK］ボタンがクリックされた場合のみ，選択されたターゲット・メモリ領域とその領域にマッピングされている I/O 保護領域をすべて削除します。

［機能ボタン］

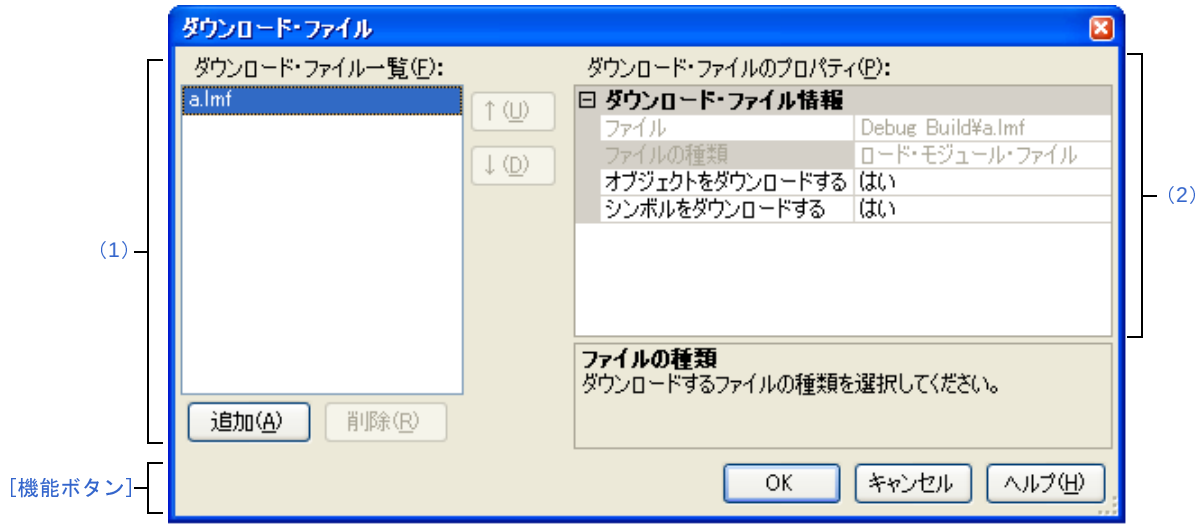
ボタン	機能
OK	現在設定されているメモリ・マッピングをデバッグ・ツールに設定し，このダイアログをクローズします。
キャンセル	メモリ・マッピングの変更を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

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）。

図 A—34　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

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) [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することにより行います。

(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。

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（最大指定文字数：259 文字）。		
	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形式」参照	
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指定します。		
	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	次のいずれか - ロード・モジュール・ファイル - ヘキサ・ファイル - バイナリ・データ・ファイル	
オフセット	この項目は，ダウンロードするファイルがヘキサ・フォーマットの場合のみ表示されます。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。		
	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 ～ 0xFFFFFF の 16 進数	
開始アドレス	この項目は，ダウンロードするファイルがバイナリ・データ・フォーマットの場合のみ表示されます。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。		
	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 ～ 0xFFFFFF の 16 進数	
オブジェクトをダウンロードする	この項目は，ダウンロードするファイルがロード・モジュール・フォーマットの場合のみ表示されます。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指定します。		
	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	この項目は，ダウンロードするファイルがロード・モジュール・フォーマットの場合のみ表示されます。 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指定します注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。

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。

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。

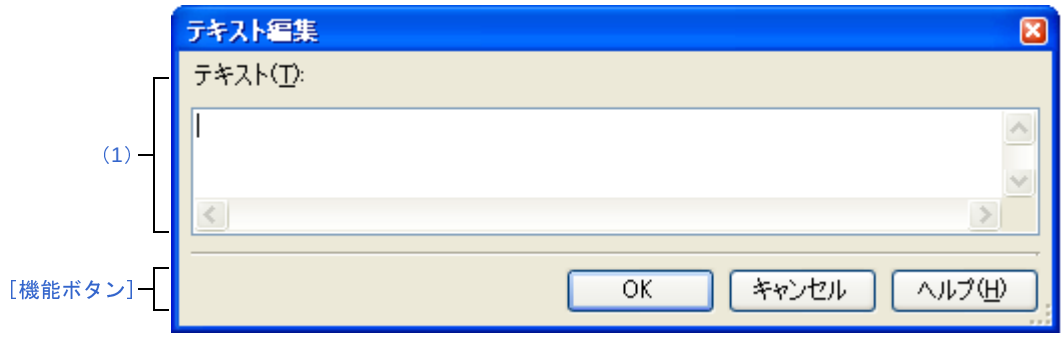
［機能ボタン］

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。

テキスト編集 ダイアログ

複数行のテキストの入力，編集を行います。

図 A—35　テキスト編集 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［フック処理設定］タブにおいて，［フック処理設定］カテゴリ内の各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)［テキスト］エリア

複数行のテキストの編集を行います。

［機能ボタン］

ボタン	機能
OK	入力したテキストをこのダイアログの呼び出し元に反映し，このダイアログをクローズします。
キャンセル	入力したテキストをこのダイアログの呼び出し元に反映せずに，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。

アクション・イベント ダイアログ

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. 14　プログラム内へのアクションの設定」参照）。

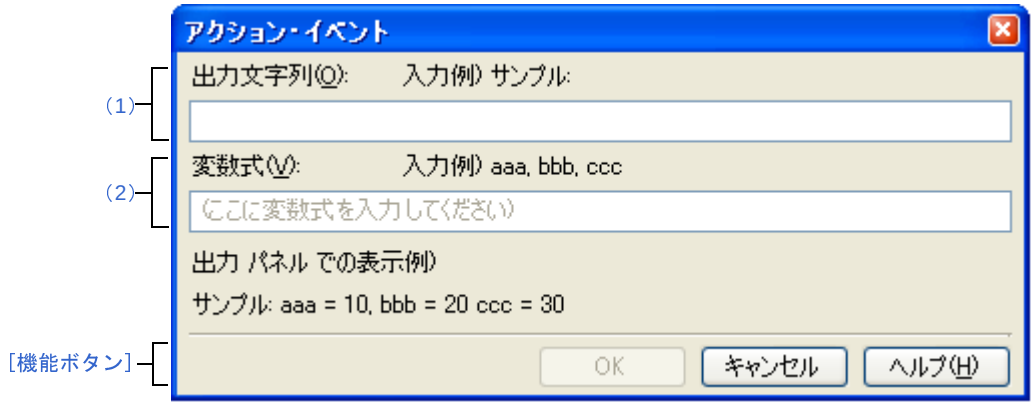
Printf イベントとは，プログラムの実行を指定した箇所で一瞬停止させ，ソフトウエア処理によりコマンド（printf）を実行させる機能です。Printf イベントを設定すると，このイベントを設定した箇所の命令実行直前にプログラムが一瞬停止し，このダイアログで指定した変数式の値を出力 パネルに出力します。

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 1.　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. 15. 6　イベント設定に関する留意事項」も参照してください。

2.　一度設定した Printf イベントの変数式を編集することはできません。
編集が必要な場合は，該当する Printf イベントを削除したのち，新たに Printf イベントを設定してください。

図 A—36　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，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

[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—14　変数式と出力される値の関係（Printf イベント）

変数式	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
CPU レジスタ名	CPU レジスタの値
SFR 名	SFR の値
ラベル名 /EQU シンボル名／即値アドレス	ラベルの値 /EQU シンボルの値／即値アドレスの値
ビット・シンボル	ビット・シンボル の値

注意　算術式（“+” ／ “-” など）を使用した変数式を指定することはできません。

備考　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—37　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（10 進数値に続き“()”内に 16 進数値を併記） ただし，指定された変数式を取得できない場合は，“?”を表示します。 また，変数が char 型の場合は，文字として表示します。

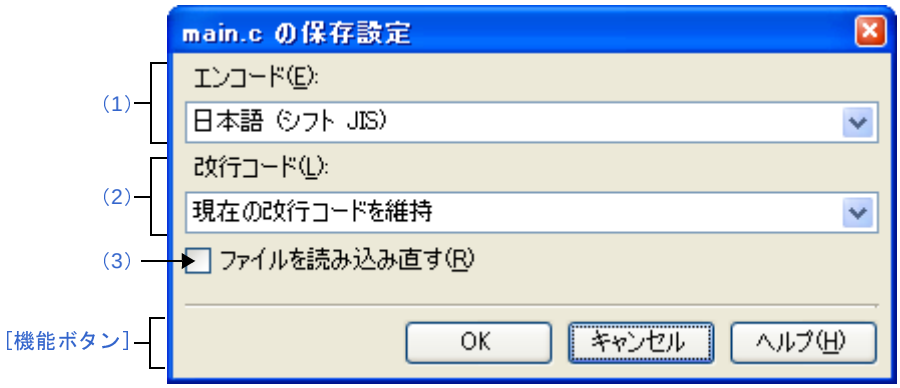
[機能ボタン]

ボタン	機能
OK	Printf イベントの設定を終了し，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／アドレスに設定します。
キャンセル	Printf 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。

ファイルの保存設定 ダイアログ

エディタ パネルで編集中のファイルのエンコードと改行コードの設定を行います。

図 A—38　ファイルの保存設定 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］

- エディタ パネルにフォーカスがある状態で，［ファイル］メニュー→［*ファイル名*の保存設定 ...］を選択

［各エリアの説明］

(1)［エンコード］エリア

設定するエンコードをドロップダウン・リストにより選択します。

ドロップダウン・リストの項目は，次の順番で表示されます。

ただし，同じエンコード名，および現在の OS が対応していないエンコード名は表示されません。

- *現在のファイルのエンコード名*（デフォルト）
- *現在の OS の既定のエンコード名*
- 日本語 (シフト JIS)
- 日本語 (JIS 1 バイト カタカナ可 - SO/SI)
- 日本語 (EUC)
- Unicode (UTF-8)
- *現在の OS が対応する上記以外のエンコード名*

(2)［改行コード］エリア

設定する改行コードをドロップダウン・リストにより選択します。

次の項目を選択することができます。

- 現在の改行コードを維持（デフォルト）
- Windows (CR LF)
- Macintosh (CR)
- Unix (LF)

(3)［ファイルを読み込み直す］

☑	［OK］ボタンをクリックした際に，指定したエンコード，および改行コードでファイルの再読み込みを行います。
☐	［OK］ボタンをクリックした際に，ファイルの再読み込みを行いません（デフォルト）。

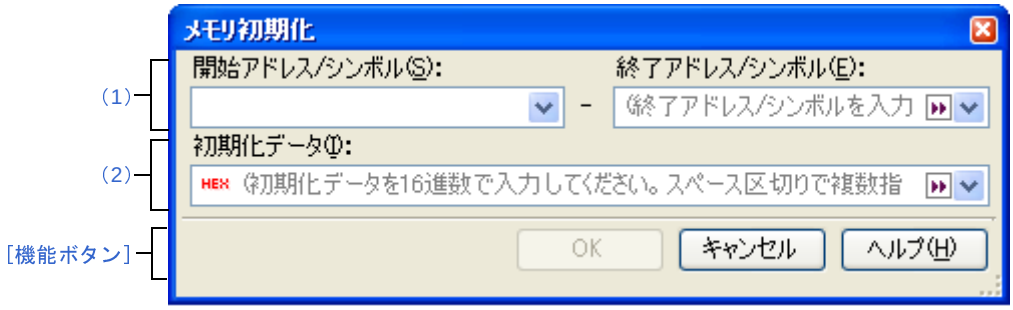
［機能ボタン］

ボタン	機能
OK	指定したエンコード，および改行コードを対象ファイルに設定し，このダイアログをクローズします。 ［ファイルを読み込み直す］をチェックした場合，指定したエンコード，および改行コードを対象ファイルに設定し，ファイルを読み込み直したのち，このダイアログをクローズします。
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。

メモリ初期化 ダイアログ

メモリ値の初期化を行います（「(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。

(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，次のようにリトル・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）
1	01
0 12	00 12
00 012 345	00 12 00 45 03
000 12 000345	00 00 12 45 03 00

[機能ボタン]

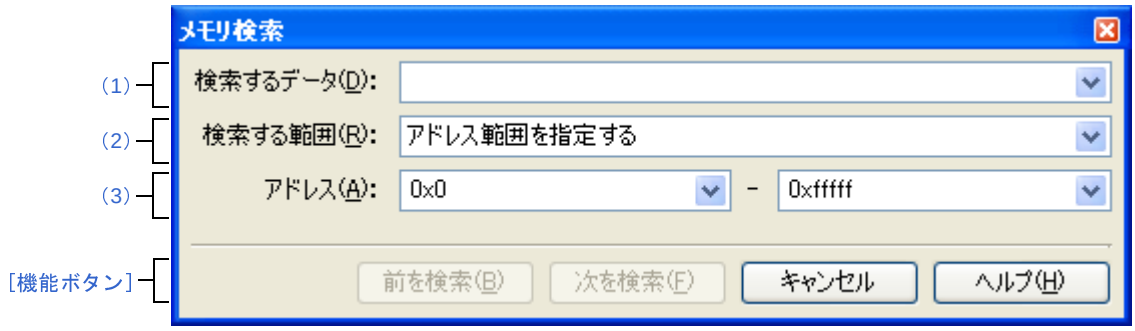
ボタン	機能
OK	指定したアドレス範囲のメモリ領域に，指定した初期化データのパターンを繰り返し書き込みます（パターンの途中で終了アドレスに達した場合は書き込みを終了します）。
キャンセル	メモリ値の初期化の設定を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

メモリ検索 ダイアログ

メモリ値の検索を行います（「(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。

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（ノン・マップ領域を除く）。 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，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。

備考 1. “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。

2. “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。

トレース検索 ダイアログ【IECUBE】【シミュレータ】

トレース・データの検索を行います（「2. 11. 7　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—41　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【IECUBE】【シミュレータ】において，ツールバーの ボタンを選択
- トレース パネル【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】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—42　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］
- ［機能ボタン］

［オープン方法］

- トレース パネル【IECUBE】【シミュレータ】において，ツールバーの ボタンを選択
- トレース パネル【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】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。

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。

(指定なし)
リード／ライト
リード
ライト
ベクタ・リード
DMA

(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【IECUBE】【シミュレータ】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。

[機能ボタン]

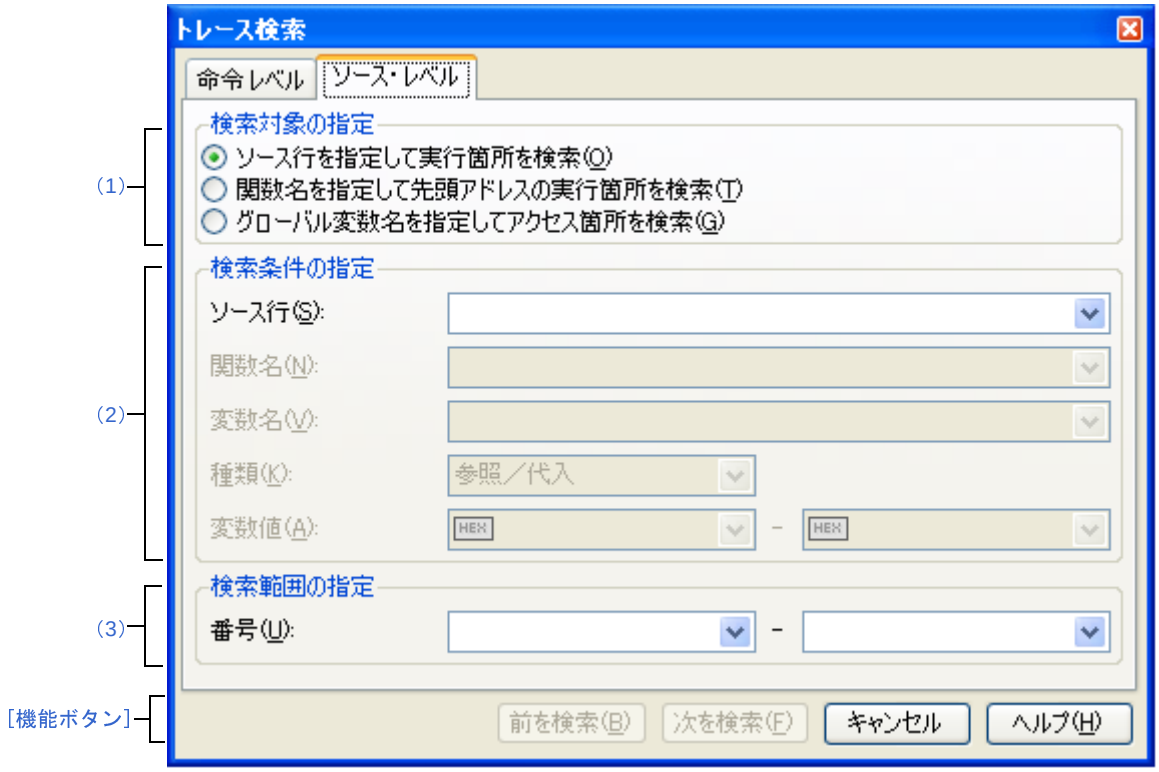
ボタン	機能
前を検索	指定した範囲内で，番号の小さい方向に検索を行います。 検索結果箇所をトレース パネル【IECUBE】【シミュレータ】上で選択状態にします。 ただし，不正な値を指定している場合はメッセージを表示し，トレース・データの検索は行いません。また，トレース パネルが非表示の場合，または他のパネルにフォーカスがある状態からこのダイアログへフォーカスを移動した場合，このボタンは無効となります。
次を検索	指定した範囲内で，番号の大きい方向に検索を行います。 検索結果箇所をトレース パネル【IECUBE】【シミュレータ】上で選択状態にします。 ただし，不正な値を指定している場合はメッセージを表示し，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。

［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【IECUBE】【シミュレータ】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—43　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］
- ［機能ボタン］

［オープン方法］

- トレース パネル【IECUBE】【シミュレータ】において，ツールバーの ボタンを選択
- トレース パネル【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】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
2. main.c
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【IECUBE】【シミュレータ】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。

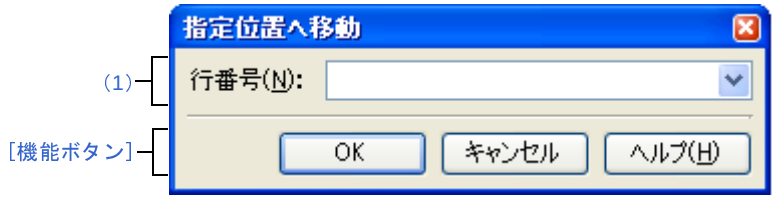
［機能ボタン］

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【IECUBE】【シミュレータ】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【IECUBE】【シミュレータ】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。

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—44　指定位置へ移動 ダイアログ（エディタ パネルを対象とする場合）



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］

- ［編集］メニュー→［移動 ...］を選択
- エディタ パネルにおいて，コンテキスト・メニューの［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
- SFR 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)［行番号］／［アドレス / シンボル］／［SFR］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，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
エディタ パネル	行番号（10 進数）
逆アセンブル パネル	アドレス式
SFR パネル	SFR 名

[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【IECUBE】【シミュレータ】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—45　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【IECUBE】【シミュレータ】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

選択できるファイル形式は，「表 2—2　アップロード可能なファイル形式」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 【IECUBE】【シミュレータ】	- 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。

保存対象	説明
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。

［機能ボタン］

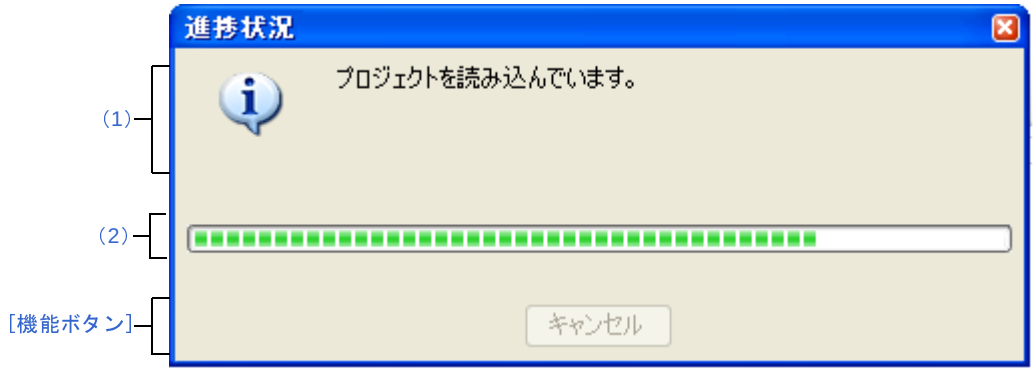
ボタン	機能
保存	指定したファイルに，指定した形式でデータを保存します。
キャンセル	データ保存の設定を無効とし，このダイアログをクローズします。
ヘルプ	このダイアログのヘルプを表示します。

処理中表示 ダイアログ

時間を要する処理を行っている際に，その進捗状況の表示を行います。

このダイアログは，実行中の処理が完了した場合，自動的にクローズします。

図 A—46　処理中表示 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］

- 時間を要する処理において，メッセージが発生した際に自動的に表示

［各エリアの説明］

(1) メッセージ表示エリア

処理中に発生したメッセージを表示します（編集不可）。

(2) プログレスバー

現在実行中の処理の進捗状況をバーの長さで表示します。

なお，進捗率が 100% に達した場合（右端までバーの長さが達した場合），このダイアログは自動的にクローズします。

［機能ボタン］

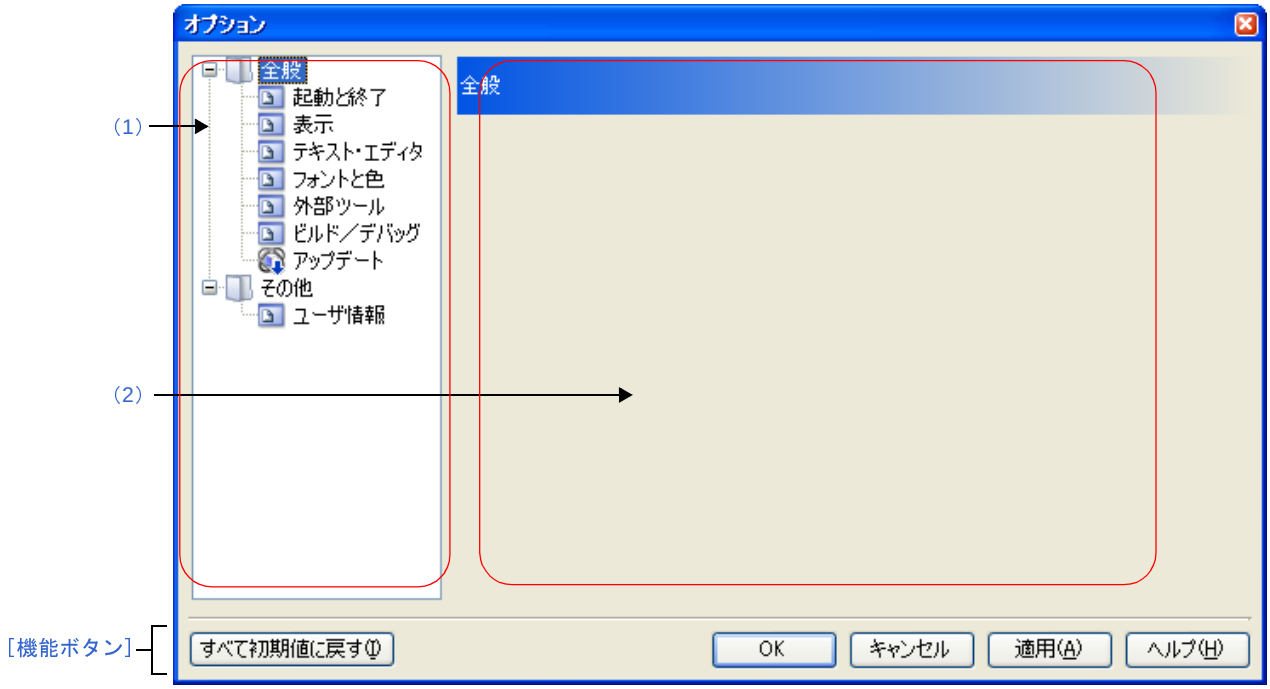
ボタン	機能
キャンセル	現在実行中の処理を中断し，このダイアログをクローズします。 ただし，実行中の処理の中断が不可能な場合，このボタンは無効となります。

オプション ダイアログ

CubeSuite+ の各種環境設定を行います。

このダイアログでの設定は，使用中のユーザの設定として保存されます。

図 A—47　オプション 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］

- ［ツール］メニュー→［オプション ...］を選択

[各エリアの説明]

(1) カテゴリ選択エリア

設定したい項目を次のカテゴリから選択します。

カテゴリ	設定内容
[全般 - 起動と終了] カテゴリ	起動，または終了時に関連した設定を行います。
[全般 - 表示] カテゴリ	表示に関連した設定を行います。
[全般 - テキスト・エディタ] カテゴリ	テキスト・エディタに関連した設定を行います。
[全般 - フォントと色] カテゴリ	各パネルで表示するフォントと色に関連した設定を行います。
[全般 - 外部ツール] カテゴリ	外部ツールを起動する際の設定を行います。
[全般 - ビルド／デバッグ] カテゴリ	ビルド，またはデバッグに関連した設定を行います。
[全般 - アップデート] カテゴリ	アップデートに関連した設定を行います。
[その他 - ユーザ情報] カテゴリ	ユーザ情報に関連した設定を行います。

(2) 設定エリア

選択したカテゴリに対して，各種オプションを設定するエリアです。

各カテゴリの設定方法についての詳細は，該当するカテゴリの項を参照してください。

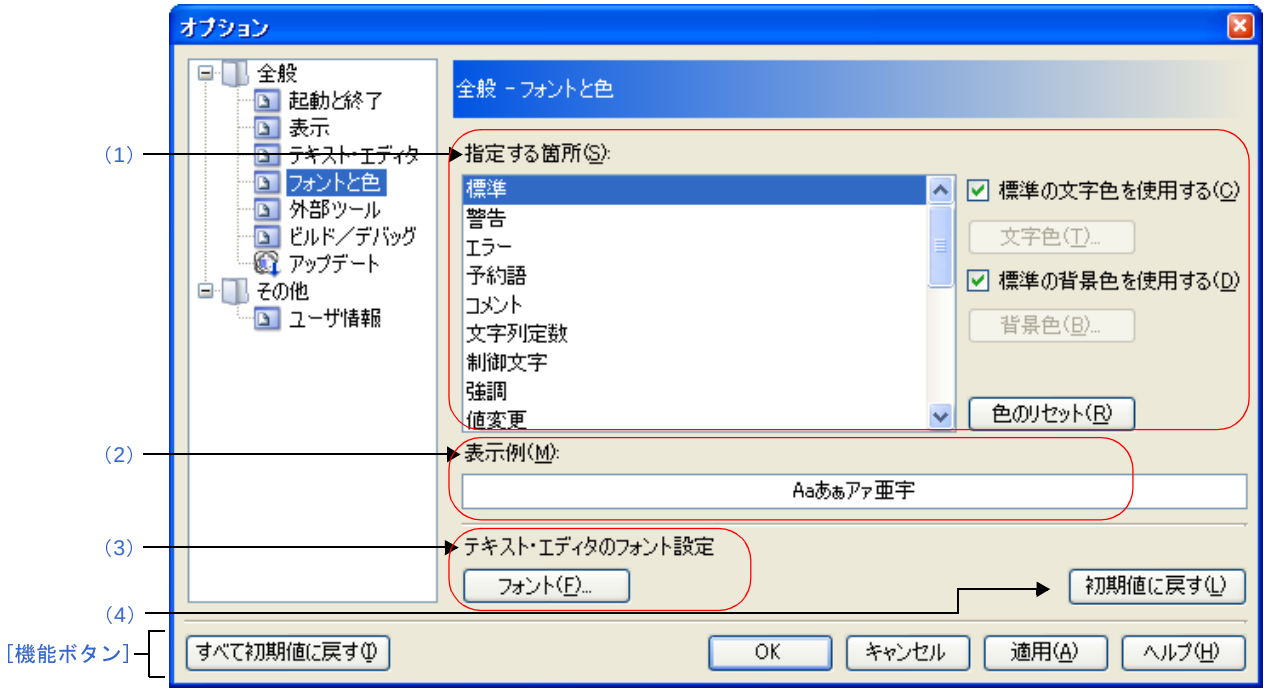
[機能ボタン]

ボタン	機能
すべて初期値に戻す	このダイアログのすべての設定項目をデフォルトの状態に戻します。 ただし，[全般 - 外部ツール] カテゴリでは，新規登録した内容の削除は行いません。
OK	変更した設定内容を適用し，このダイアログをクローズします。
キャンセル	変更した設定内容を無効とし，このダイアログをクローズします。
適用	変更した設定内容を適用します（このダイアログをクローズしません）。
ヘルプ	このダイアログのヘルプを表示します。

［全般 - フォントと色］カテゴリ

全般に関わる設定のうち，各パネルで表示するフォントと色に関連した設定を行います。

図 A—48　オプション ダイアログ（［全般 - フォントと色］カテゴリ）



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］

- ［ツール］メニュー→［オプション ...］を選択

［各エリアの説明］

（1）色の設定エリア

色の設定を行うエリアです。

（a）［指定する箇所］エリア

表示色を指定したい箇所をリスト一覧から選択します。

表示されるリスト一覧の項目と，デフォルトで設定されている色の関係は次のとおりです。

項目	表示例			説明
標準注	AaBbCc	文字色	黒	すべてのウインドウ／パネルにおいて，標準となる表示色
		背景色	白	
警告	AaBbCc	文字色	青	出力 パネルにおいて，警告メッセージの表示色，およびプロジェクト・ツリー パネルにおける“警告あり”ファイル名の表示色
		背景色	標準色	
エラー	AaBbCc	文字色	赤	出力 パネルにおいて，エラー・メッセージの表示色，およびプロジェクト・ツリー パネルにおける“エラーあり”ファイル名の表示色
		背景色	薄グレー	
予約語	AaBbCc	文字色	茶	エディタ パネルにおいて，使用するコンパイラ／アセンブラの予約語の表示色
		背景色	標準色	
コメント	AaBbCc	文字色	緑	エディタ パネルにおいて，コメント部（C ソース・ファイルの場合，“/* ～ */”）の表示色
		背景色	標準色	
制御文字	AaBbCc	文字色	青緑	エディタ パネルにおいて，制御文字の表示色
		背景色	標準色	
文字列定数	AaBbCc	文字色	グレー	エディタ パネルにおいて，文字列定数の表示色
		背景色	標準色	
強調	AaBbCc	文字色	白	プラグイン製品などにおいて，強調箇所の表示色
		背景色	赤紫	
値変更	AaBbCc	文字色	薄茶	メモリ パネル /CPU レジスタ パネル／ローカル変数パネル /SFR パネル／ウォッチ パネルにおいて，プログラムの実行により値が変更した箇所の表示色
		背景色	クリーム	
値編集	AaBbCc	文字色	青	メモリ パネル /CPU レジスタ パネル／ローカル変数パネル /SFR パネル／ウォッチ パネルにおいて，ユーザが強制的に値を変更した箇所の表示色
		背景色	標準色	
PC 位置	AaBbCc	文字色	黒	エディタ パネル／逆アセンブル パネルにおいて，カレント PC 位置のある行の表示色
		背景色	山吹	
ブレークポイント	AaBbCc	文字色	黒	エディタ パネル／逆アセンブル パネルにおいて，ブレークポイントが設定されている行の表示色
		背景色	サーモンピンク	
リアルタイム更新中	AaBbCc	文字色	ピンク	メモリ パネル／ウォッチ パネルにおいて，リアルタイム表示更新に設定されている領域の表示色
		背景色	標準色	
リード／フェッチ	AaBbCc	文字色	標準色	トレース パネル【IECUBE】【シミュレータ】において，リード，またはフェッチされた箇所の表示色
		背景色	薄緑	
ライト	AaBbCc	文字色	標準色	トレース パネル【IECUBE】【シミュレータ】において，ライトされた箇所の表示色
		背景色	オレンジ	
リード＆ライト	AaBbCc	文字色	標準色	トレース パネル【IECUBE】【シミュレータ】において，リードとライトされた箇所の表示色
		背景色	薄青	
カバレッジ 100%	AaBbCc	文字色	標準色	エディタ パネル／逆アセンブル パネルにおいて，コード・カバレッジ率 100% の行の表示色
		背景色	ライトグリーン	

項目	表示例			説明
カバレッジ 1 ～ 99%	AaBbCc	文字色	標準色	エディタ パネル／逆アセンブル パネルにおいて，コード・カバレッジ率 1 ～ 99% の行の表示色
		背景色	ライトピンク	
カバレッジ 0%	AaBbCc	文字色	標準色	エディタ パネル／逆アセンブル パネルにおいて，コード・カバレッジ率 0%（未実行）の行の表示色
		背景色	ライトグレー	
無効	AaBbCc	文字色	グレー	メモリ パネルにおいて，メモリ・マッピングされていない領域，およびプロジェクト・ツリー パネル上で実際存在しないファイル名の表示色
		背景色	標準色	

注　［標準］の文字色／背景色は，使用するホスト・マシンにおける Windows の設定に依存します。ここでは，Windows のデフォルト設定である“文字色：黒”，“背景色：白”を表記しています。

(b)［標準の文字色を使用する］

☑	［指定する箇所］エリアで選択している項目を，標準の文字色を使用して表示します。
☐	［指定する箇所］エリアで選択している項目の文字色を，任意に指定します。 ［文字色 ...］ボタンが有効となります。

(c)［標準の背景色を使用する］

☑	［指定する箇所］エリアで選択している項目を，標準の背景色を使用して表示します。
☐	［指定する箇所］エリアで選択している項目の背景色を，任意に指定します。 ［背景色 ...］ボタンが有効となります。

(d) ボタン

ボタン	機能
文字色 ...	色の設定 ダイアログがオープンし，［指定する箇所］エリアで選択している項目の文字色を指定します。 ただし，［標準の文字色を使用する］をチェックしている場合は，無効となります。
背景色 ...	色の設定 ダイアログがオープンし，［指定する箇所］エリアで選択している項目の背景色を指定します。 ただし，［標準の背景色を使用する］をチェックしている場合は，無効となります。
色のリセット	［指定する箇所］エリアで選択している項目の色情報をリセットし，デフォルトの設定に戻します。

図 A—49　色の設定 ダイアログ

(2)［表示例］エリア

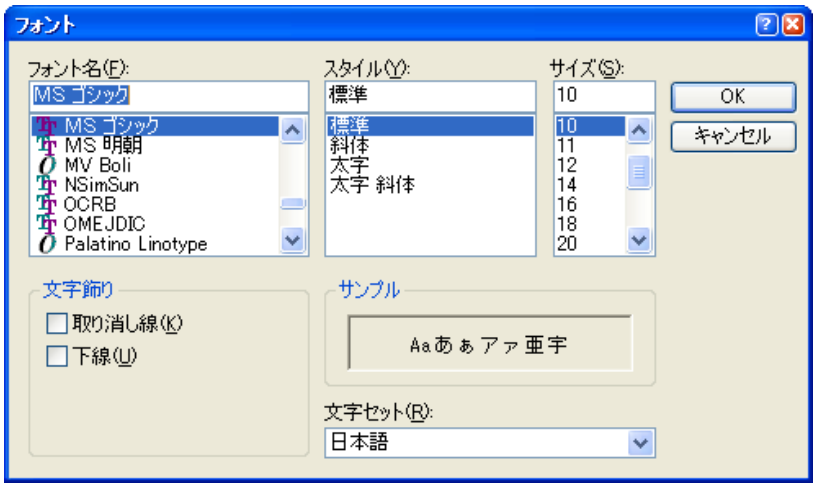
色の設定エリア，および［テキスト・エディタのフォント設定］エリアで指定した色とフォントの表示例を示します。

デフォルトでは，文字列“Aa あぁアァ亜宇”を表示しますが，テキスト・ボックスに任意の文字列を直接入力することができます。

(3)［テキスト・エディタのフォント設定］エリア

［フォント ...］ボタンを押下することにより，次のフォント ダイアログをオープンし，使用するテキスト・エディタで使用するフォントを設定します。

図 A—50　フォント ダイアログ



(4) ボタン・エリア

初期値に戻す	現在表示している項目の指定をすべてデフォルトに戻します。

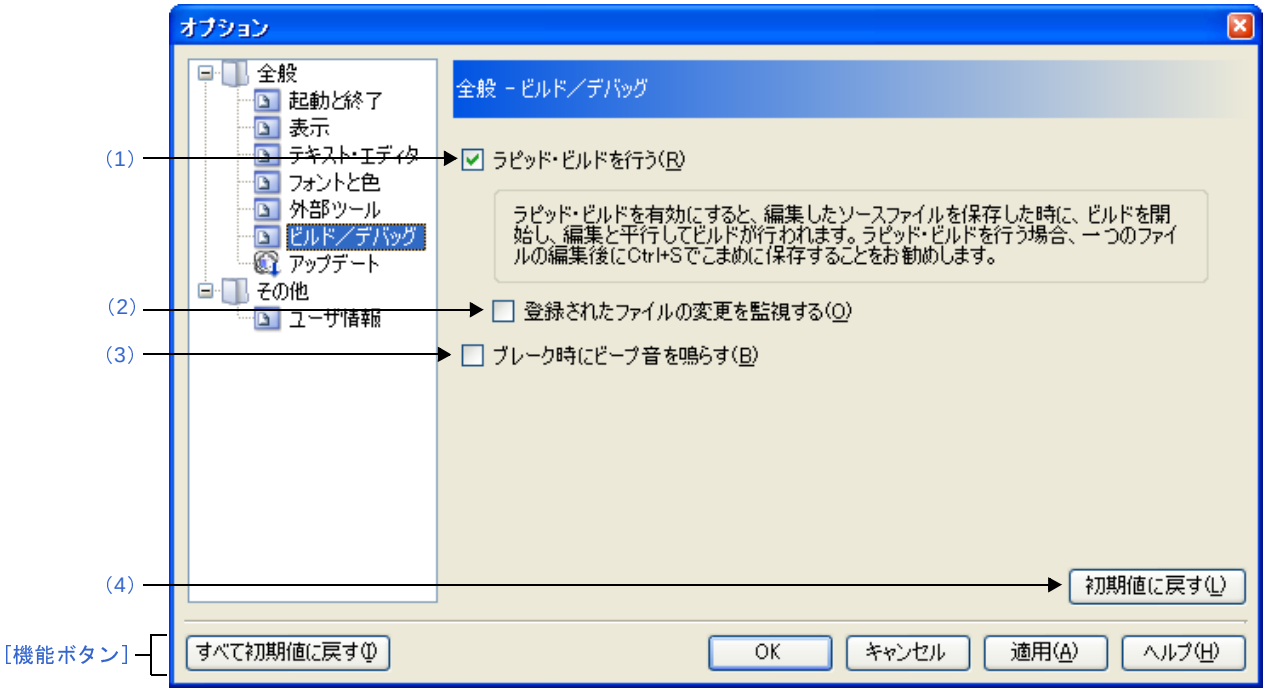
［機能ボタン］

ボタン	機能
すべて初期値に戻す	このダイアログのすべての設定項目をデフォルトの状態に戻します。 ただし，［全般 - 外部ツール］カテゴリでは，新規登録した内容の削除は行いません。
OK	変更した設定内容を適用し，このダイアログをクローズします。
キャンセル	変更した設定内容を無効とし，このダイアログをクローズします。
適用	変更した設定内容を適用します（このダイアログをクローズしません）。
ヘルプ	このダイアログのヘルプを表示します。

［全般 - ビルド／デバッグ］カテゴリ

全般に関わる設定のうち，ビルド，またはデバッグに関連した設定を行います。

図 A—51　オプション ダイアログ（［全般 - ビルド／デバッグ］カテゴリ）



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］

- ［ツール］メニュー→［オプション ...］を選択

［各エリアの説明］

(1)［ラピッド・ビルドを行う］

☑	ラピッド・ビルド機能注を有効にします（デフォルト）。
☐	ラピッド・ビルド機能を使用しません。

注　編集したソース・ファイルの保存時に，ビルドを自動で開始する機能です。
この機能を有効にすることにより，ソース・ファイルの編集と同時にビルドを行うことができます。
なお，この機能を使用する場合，ソース・ファイル編集後，こまめに上書き保存することを推奨します。

(2)［登録されたファイルの変更を監視する］

☑	プロジェクトに登録されたソースファイルの変更を監視し，外部エディタなどで編集／保存されたときに，ラピッド・ビルドを開始します。
☐	プロジェクトに登録されたソースファイルの変更を監視し，外部エディタなどで編集／保存されたときに，ラピッド・ビルドを開始しません（デフォルト）。

備考　［ラピッド・ビルドを行う］チェック・ボックスにチェックが付いている場合のみ有効です。

注意　この項目をチェックし，かつ，プロジェクトに登録されたソース・ファイルで存在しないファイル（プロジェクト・ツリーでグレー表示されたファイル）がある場合，エクスプローラなどでファイルを再登録しても，監視状態にはなりません。監視状態にするためには，プロジェクト・ファイルを読み込み直すか，またはこの項目のチェックを一旦外してダイアログを閉じた後，再度この項目をチェックしてください。

(3)［ブレーク時にビープ音を鳴らす］

☑	プログラムの実行が，ブレーク・イベント（ハードウエア・ブレーク／ソフトウエア・ブレーク）により停止した際，ビープ音を鳴らします。
☐	プログラムの実行が，ブレーク・イベント（ハードウエア・ブレーク／ソフトウエア・ブレーク）により停止した際，ビープ音を鳴らしません（デフォルト）。

(4) ボタン・エリア

初期値に戻す	現在表示している項目をすべてデフォルトに戻します。

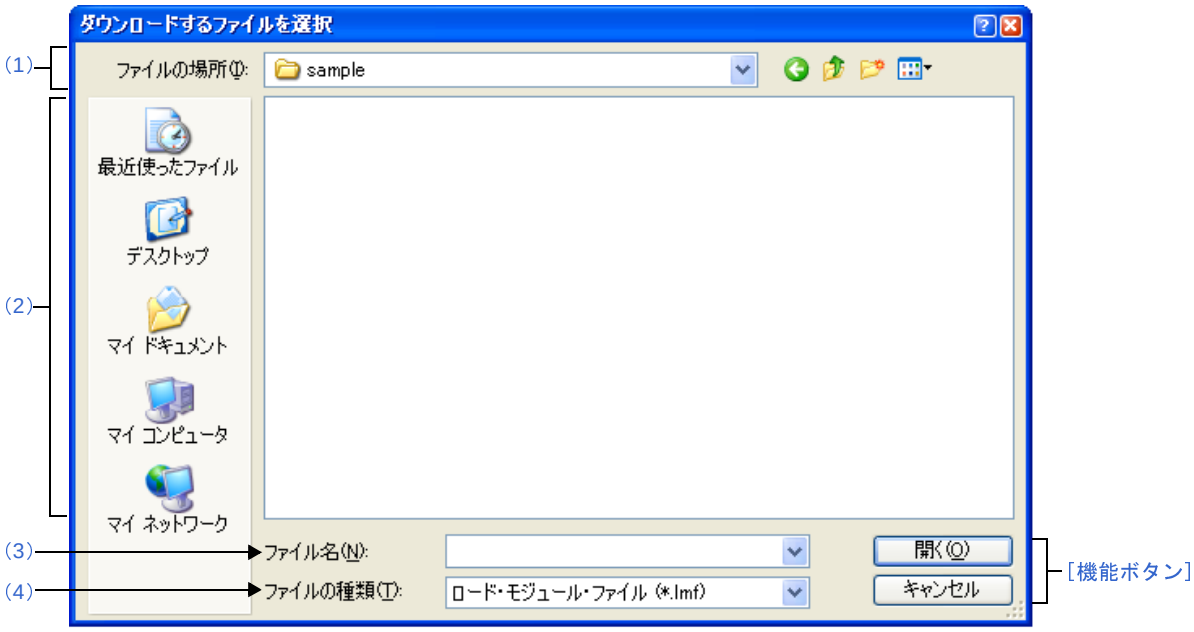
［機能ボタン］

ボタン	機能
すべて初期値に戻す	このダイアログのすべての設定項目をデフォルトの状態に戻します。 ただし，［全般 - 外部ツール］カテゴリでは，新規登録した内容の削除は行いません。
OK	変更した設定内容を適用し，このダイアログをクローズします。
キャンセル	変更した設定内容を無効とし，このダイアログをクローズします。
適用	変更した設定内容を適用します（このダイアログをクローズしません）。
ヘルプ	このダイアログのヘルプを表示します。

ダウンロードするファイルを選択 ダイアログ

ダウンロードするファイルの選択を行います。

図 A—52　ダウンロードするファイルを選択 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］

- ダウンロード・ファイル ダイアログの［ダウンロード・ファイルのプロパティ］エリアにおいて，［ファイル項目］内の［...］ボタンのクリック

［各エリアの説明］

(1)［ファイルの場所］エリア

ダウンロードするファイルの存在するフォルダをドロップダウン・リストにより選択します。

(2) ファイル一覧エリア

［ファイルの場所］，および［ファイルの種類］で選択された条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

ダウンロードするファイル名を指定します。

(4)［ファイルの種類］エリア

ダウンロードするファイルの種類（ファイル・タイプ）を次のドロップダウン・リストにより選択します。

ロード・モジュール・ファイル (*.lmf)	ロード・モジュール・フォーマット（デフォルト）
ヘキサ・ファイル (*.hex;*.hxb;*.hxf)	ヘキサ・フォーマット
バイナリ・データ・ファイル (*.bin)	バイナリ・データ・フォーマット
すべてのファイル (*.*)	すべてのファイル・フォーマット

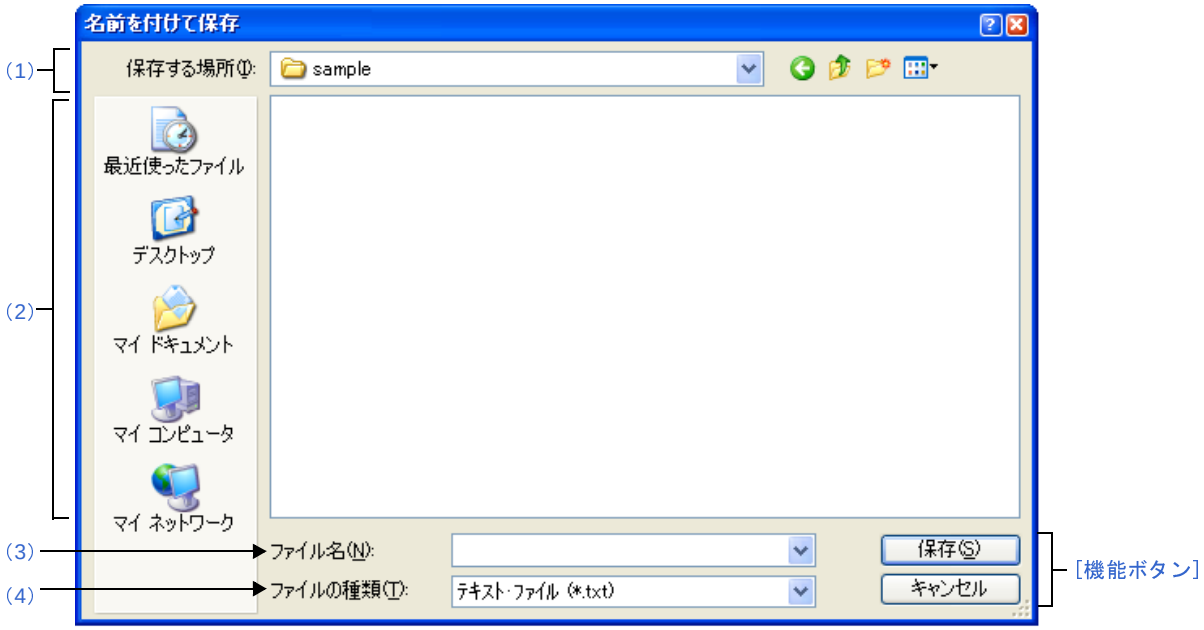
［機能ボタン］

ボタン	機能
開く	指定したファイルをダウンロード・ファイル ダイアログに追加します。
キャンセル	このダイアログをクローズします。

名前を付けて保存 ダイアログ

編集中のファイル，または各パネルの内容を名前を付けてファイルに保存します。

図 A—53　名前を付けて保存 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］

- エディタ 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 *ファイル名*を保存 ...］を選択
- CPU レジスタ 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 CPU レジスタ・データを保存 ...］を選択
- 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
- SFR 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 SFR データを保存 ...］を選択
- コール・スタック パネルにフォーカスがある状態で，［ファイル］メニュー→［名前を付けてコール・スタック・データを保存 ...］を選択
- ローカル変数 パネルにフォーカスがある状態で，［ファイル］メニュー→［名前を付けてローカル変数データを保存 ...］を選択
- 出力 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 *タブ名*を保存 ...］を選択

[各エリアの説明]

(1) [保存する場所] エリア

ファイルを保存するフォルダを選択します。

(2) ファイル一覧エリア

[保存する場所] エリア，および [ファイルの種類] エリアで選択された条件に合致するファイルの一覧を表示します。

(3) [ファイル名] エリア

保存する際のファイル名を指定します。

(4) [ファイルの種類] エリア

(a) エディタ パネルの場合

編集中のファイルの種類に依存して，次のファイルの種類（ファイル・タイプ）を表示します。

テキスト・ファイル (*.txt)	テキスト形式
C ソース・ファイル (*.c)	C ソース・ファイル
ヘッダ・ファイル (*.h;*.inc)	ヘッダ・ファイル
アセンブル・ファイル (*.asm)	アセンブラ・ソース・ファイル
リンク・ディレクティブ・ファイル (*.dr;*.dir)	リンク・ディレクティブ・ファイル
変数／関数情報ファイル (*.vfi)	変数，および関数情報ファイル
マップ・ファイル (*.map)	マップ・ファイル
シンボル・テーブル・ファイル (*.sym)	シンボル・テーブル・ファイル
ヘキサ・ファイル (*.hex;*hxb;*hxf)	ヘキサ・ファイル

(b) CPU レジスタ パネル／ウォッチ パネル /SFR パネル／コール・スタック パネル／ローカル変数 パネルの場合

次のファイルの種類（ファイル・タイプ）を表示します。

ドロップダウン・リストより選択したファイル形式でパネルの内容をファイルに保存します。

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。

(c) 出力パネルの場合

次のファイルの種類（ファイル・タイプ）を表示します。

テキスト形式でのみ保存することができます。

テキスト・ファイル (*.txt)	テキスト形式（デフォルト）

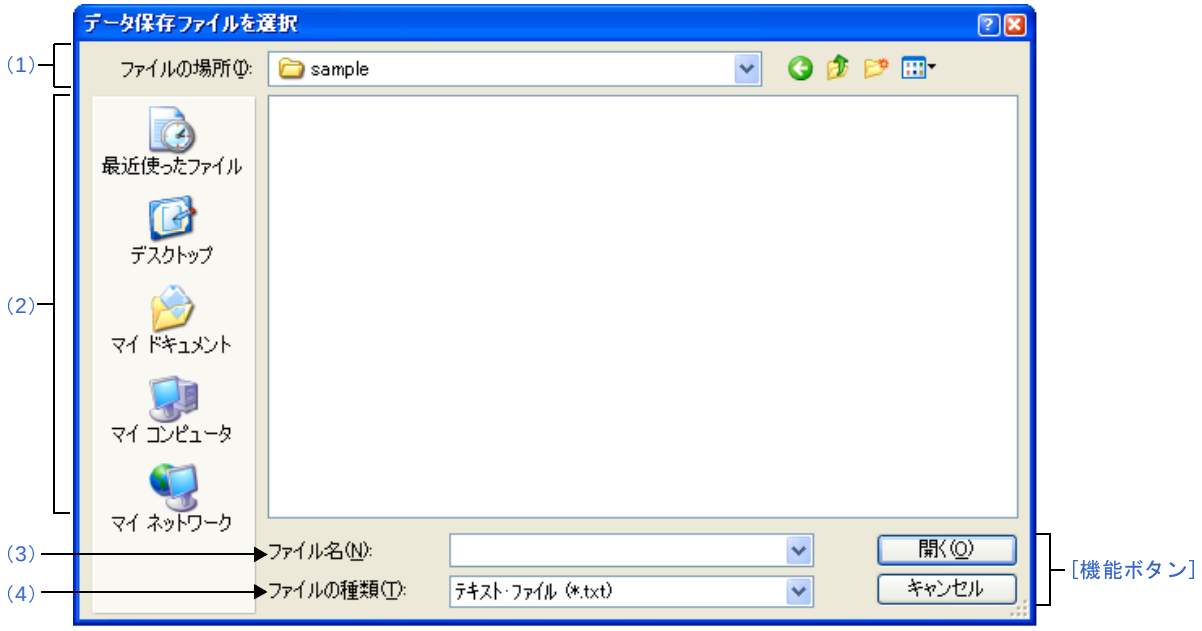
[機能ボタン]

ボタン	機能
保存	指定したファイル名でファイルを保存します。
キャンセル	このダイアログをクローズします。

データ保存ファイルを選択 ダイアログ

データを保存するファイルの選択を行います。

図 A—54　データ保存ファイルを選択 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］

- データ保存 ダイアログの［ファイル名］エリアにおいて［...］ボタンをクリック

［各エリアの説明］

(1)［ファイルの場所］エリア

保存するファイルの存在するフォルダをドロップダウン・リストにより選択します。

(2) ファイル一覧エリア

［ファイルの場所］，および［ファイルの種類］で選択された条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

保存するファイル名を指定します。

(4)［ファイルの種類］エリア

保存するファイルの種類（ファイル・タイプ）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

選択できるファイル形式は，「表 2—2　アップロード可能なファイル形式」を参照してください。

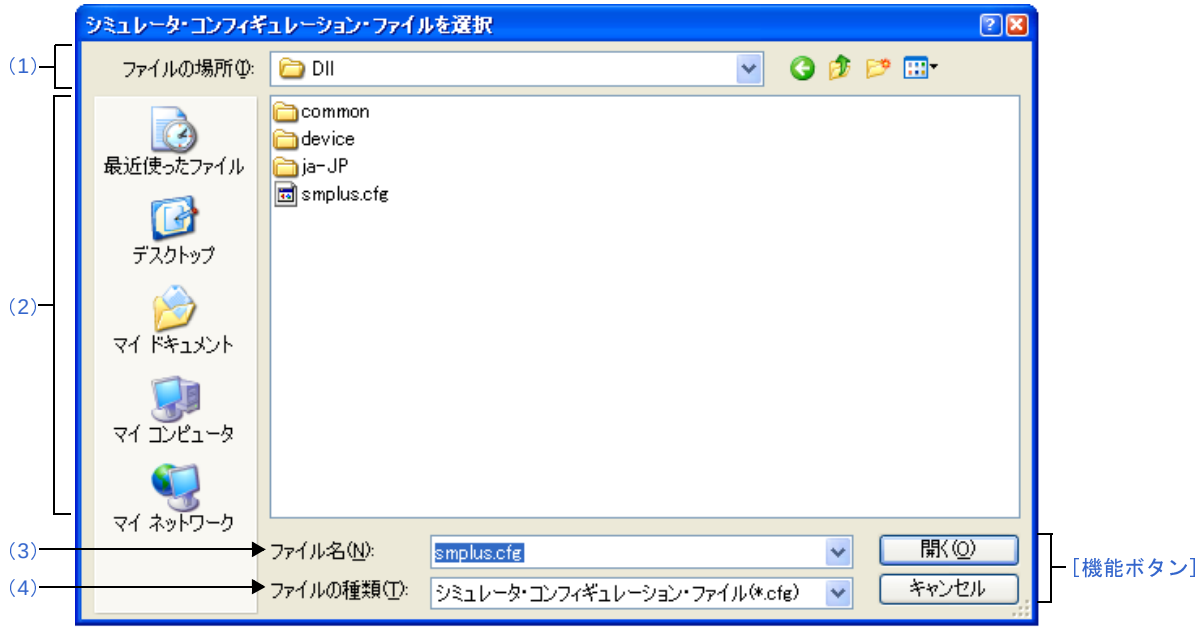
［機能ボタン］

ボタン	機能
開く	指定したファイルをデータ保存 ダイアログに指定します。
キャンセル	このダイアログをクローズします。

シミュレータ・コンフィギュレーション・ファイルを選択 ダイアログ【シミュレータ】

シミュレータを使用する場合において，ユーザ・カスタマイズ（ユーザ・モデルの追加）を行うためのシミュレータ・コンフィギュレーション・ファイルの選択を行います。

図 A—55　シミュレータ・コンフィギュレーション・ファイルを選択 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］

- プロパティ パネルにおいて，［接続用設定］タブ→［コンフィギュレーション］カテゴリ→［シミュレータ・コンフィギュレーション・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)［ファイルの場所］エリア

シミュレータ・コンフィギュレーション・ファイルが存在するフォルダをドロップダウン・リストにより選択します。

デフォルトでは空欄です。

(2) ファイル一覧エリア

［ファイルの場所］，および［ファイルの種類］で選択された条件に合致するファイルの一覧を表示します。

(3)［ファイル名］エリア

使用するシミュレータ・コンフィギュレーション・ファイル名を指定します。

(4)［ファイルの種類］エリア

ファイルの種類（ファイル・タイプ）を選択します。

ただし，“シミュレータ・コンフィギュレーション・ファイル (*.cfg)”に固定です。

［機能ボタン］

ボタン	機能
開く	指定したシミュレータ・コンフィギュレーション・ファイルを使用します。
キャンセル	このダイアログをクローズします。

シミュレータ GUI ウインドウ

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器シミュレーションをサポートしている場合で，かつ使用するデバッグ・ツールに“シミュレータ”を選択している場合，デバッグ・ツールと接続すると自動的にオープンするウインドウです（「2. 17　シミュレータ GUI の使用【シミュレータ】」参照）。

シミュレータ GUI では，このウインドウを中心に，各種ウインドウ（信号データエディタ ウインドウ／タイミングチャート ウインドウ／入出力パネル ウインドウ／シリアル ウインドウ）を操作します。

注意 1.　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，このウインドウをオープンすることはできません。

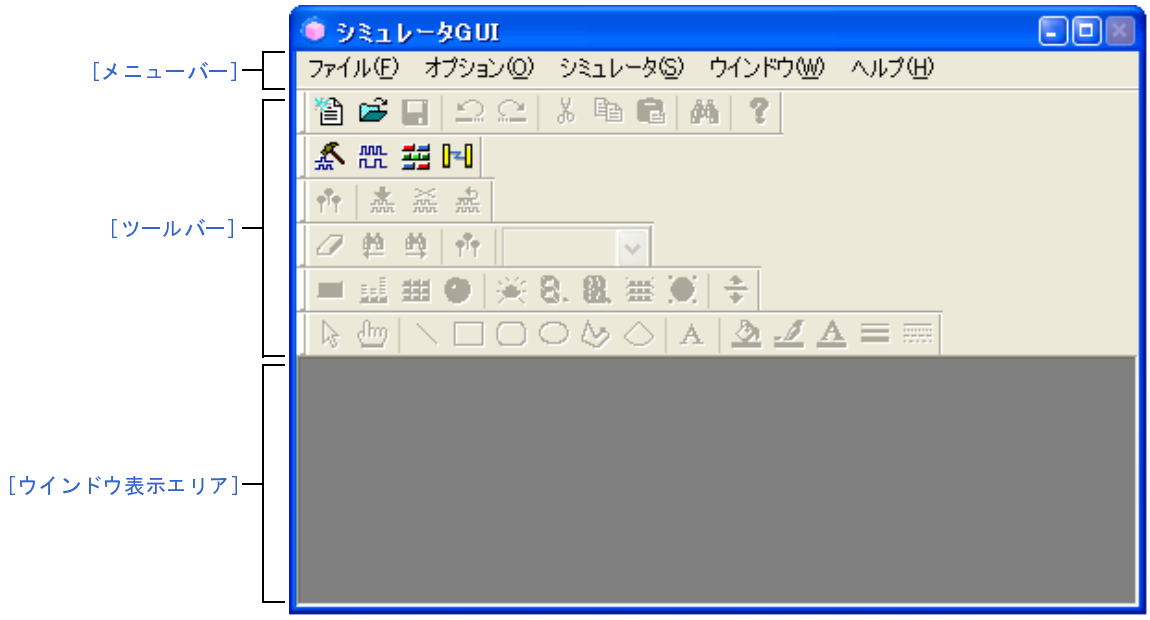
2.　このウインドウ，およびこのウインドウよりオープンする各種ウインドウは，CubeSuite+ のメイン・ウインドウとドッキング表示することはできません。

3.　このウインドウよりオープンする各種ウインドウを何もオープンしていない状態で［F1］キーを押下してもこのウインドウのヘルプは表示されません。このウインドウのヘルプを表示する場合は，このウインドウ上の［ヘルプ］メニュー→［メイン・ウインドウ］を選択してください。

4.　このウインドウ右上の［x］ボタンは無効です（Windows Vista の Aero 機能を有効にしている場合も無効）。このウインドウを非表示にする場合は，プロパティ パネルの設定により行ってください（「2. 17　シミュレータ GUI の使用【シミュレータ】」参照）。
なお，［Alt］+［F4］キーにより，このウインドウをクローズすることができますが，この操作は行わないでください。

備考　このウインドウ，およびこのウインドウよりオープンする各種ウインドウのタイトルバー／メニューバーの表記は，使用するホスト・マシンの［コントロール パネル］→［地域と言語のオプション］設定に依存します（この設定を日本以外／日本語以外に指定すると，タイトルバー／メニューバーが英語表記となります）。

図 A—56　シミュレータ GUI ウインドウ



ここでは，次の項目について説明します。

- ［メニューバー］
- ［ツールバー］
- ［ウインドウ表示エリア］

［メニューバー］

（1）［ファイル］メニュー
（2）［編集］メニュー
（3）［表示］メニュー
（4）［部品］メニュー
（5）［図形］メニュー
（6）［オプション］メニュー
（7）［シミュレータ］メニュー
（8）［ウインドウ］メニュー
（9）［ヘルプ］メニュー

（1）［ファイル］メニュー

新規作成 ...	新規作成のための選択ダイアログをオープンします。 ボタンのクリックと同様です。
開く ...	シミュレータ GUI ウインドウが扱うファイルを開きます。 ボタンのクリックと同様です。
閉じる	現在フォーカスのあるウインドウをクローズします。
上書き保存	シミュレータ GUI ウインドウが扱うファイルに，現在フォーカスのあるウインドウの内容を上書き保存します。 ボタンのクリックと同様です。
名前を付けて保存 ...	指定したファイルに，現在フォーカスのあるウインドウの内容を保存します。

（2）［編集］メニュー

このメニューの内容は，現在フォーカスのあるウインドウの種類により異なります。

このメニューについての詳細は，信号データエディタ ウインドウ／タイミングチャート ウインドウ／入出力パネル ウインドウ／シリアル ウインドウの［専用メニュー］の項を参照してください。

（3）［表示］メニュー

このメニューの内容は，現在フォーカスのあるウインドウの種類により異なります。

このメニューについての詳細は，信号データエディタ ウインドウ／タイミングチャート ウインドウ／入出力パネル ウインドウ／シリアル ウインドウの［専用メニュー］の項を参照してください。

（4）［部品］メニュー

このメニューは，入出力パネル ウインドウをオープンした際に追加されます。

このメニューについての詳細は，［部品］メニュー／［部品］ツールバーを参照してください。

(5) ［図形］メニュー

このメニューは，入出力パネル ウインドウをオープンした際に追加されます。

このメニューについての詳細は，［図形］メニュー／［図形］ツールバーを参照してください。

(6) ［オプション］メニュー

ツールバー	カスケード・メニューに対応するツールバーの表示／非表示を切り替えます。
シミュレータスタンダード	［シミュレータスタンダード］ツールバーの表示／非表示を切り替えます。
シミュレータツール	［シミュレータツール］ツールバーの表示／非表示を切り替えます。
信号データエディタ	［信号データエディタ］ツールバーの表示／非表示を切り替えます。
タイミングチャート	［タイミングチャート］ツールバーの表示／非表示を切り替えます。
部品	［部品］ツールバーの表示／非表示を切り替えます。
図形	［図形］ツールバーの表示／非表示を切り替えます。
ウインドウのカスタマイズ ...	書式設定 ダイアログをオープンします。

(7) ［シミュレータ］メニュー

信号データエディタ	信号データエディタ ウインドウをオープンします。 ボタンのクリックと同様です。
タイミングチャート	タイミングチャート ウインドウをオープンします。 ボタンのクリックと同様です。
入出力パネル	入出力パネル ウインドウをオープンします。 ボタンのクリックと同様です。
シリアル	シリアル ウインドウをオープンします。 ボタンのクリックと同様です。

(8) ［ウインドウ］メニュー

すべてのウインドウを閉じる	このウインドウを除く，すべてのウインドウをクローズします。
重ねて表示	このウインドウ内のウインドウをカスケード表示します。
並べて表示	このウインドウ内のウインドウを並べて表示します。
アイコンの整列	このウインドウの下部にアイコンを並べて表示します。

(9) ［ヘルプ］メニュー

メイン・ウインドウ	このウインドウのヘルプを表示します。
カレント ウインドウ	カレント・ウインドウのヘルプを表示します。

［ツールバー］

（1）［シミュレータスタンダード］ツールバー
（2）［シミュレータツール］ツールバー
（3）［信号データエディタ］ツールバー
（4）［タイミングチャート］ツールバー
（5）［部品］ツールバー
（6）［図形］ツールバー

（1）［シミュレータスタンダード］ツールバー

	シミュレータ GUI ウインドウを新規にオープンします。
	シミュレータ GUI ウインドウが扱うファイルを開きます。
	シミュレータ GUI ウインドウが扱うファイルに，現在フォーカスのあるウインドウの内容を上書き保存します。
	直前に行った操作を取り消し，元の状態に戻します。
	ボタンで戻した状態を復帰します。
	選択範囲を切り取り，クリップボードに保存します。
	選択範囲をコピーし，クリップボードに保存します。
	クリップボードの内容を貼り付けます。
	データ検索 ダイアログをオープンします。
	ヘルプの目次を表示します。

（2）［シミュレータツール］ツールバー

	信号データエディタ ウインドウをオープンします。
	タイミングチャート ウインドウをオープンします。
	シリアル ウインドウをオープンします。
	入出力パネル ウインドウをオープンします。

（3）［信号データエディタ］ツールバー

このツールバーは，信号データエディタ ウインドウにフォーカスがある場合にのみ選択可能となります。

このツールバーについての詳細は，［［信号データエディタ］ツールバー］を参照してください。

（4）［タイミングチャート］ツールバー

このツールバーは，タイミングチャート ウインドウにフォーカスがある場合にのみ選択可能となります。

このツールバーについての詳細は，［［タイミングチャート］ツールバー］を参照してください。

（5）［部品］ツールバー

このツールバーは，入出力パネル ウインドウにフォーカスがある場合にのみ選択可能となります。

このツールバーについての詳細は，［部品］メニュー／［部品］ツールバーを参照してください。

(6)［図形］ツールバー

このツールバーは，入出力パネル ウインドウにフォーカスがある場合にのみ選択可能となります。

このツールバーについての詳細は，［図形］メニュー／［図形］ツールバーを参照してください。

［ウインドウ表示エリア］

各種ウインドウ（信号データエディタ ウインドウ／タイミングチャート ウインドウ／入出力パネル ウインドウ／シリアル ウインドウ）を表示するエリアです。

表示されたウインドウは，このエリアの中でウインドウ・サイズの変更，アイコン化などを行います。

書式設定 ダイアログ

信号データエディタ ウインドウ，タイミングチャート ウインドウ，およびシリアル ウインドウの色やフォントの設定，変更を行います。

図 A—57　書式設定 ダイアログ：［色］タブ（タイミングチャート ウインドウの場合）

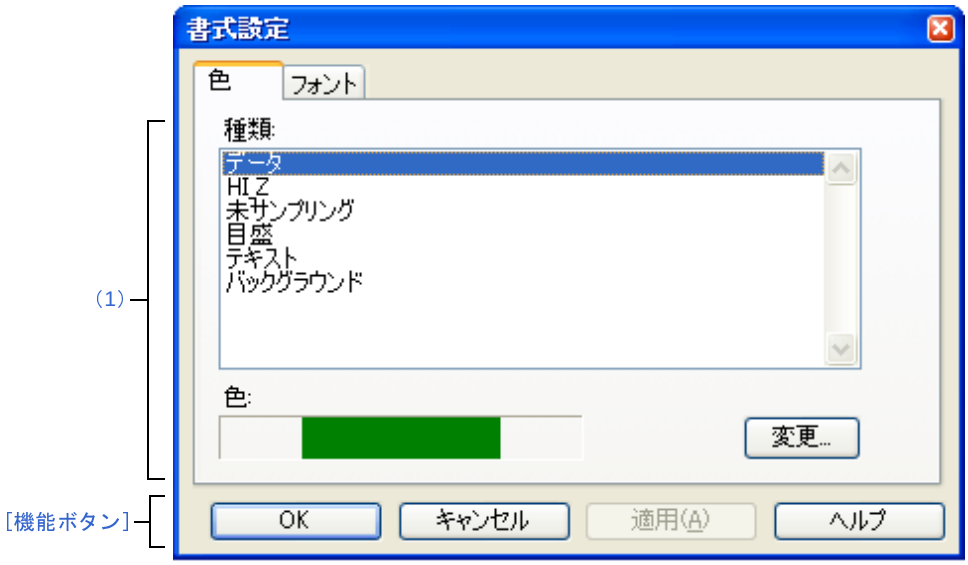
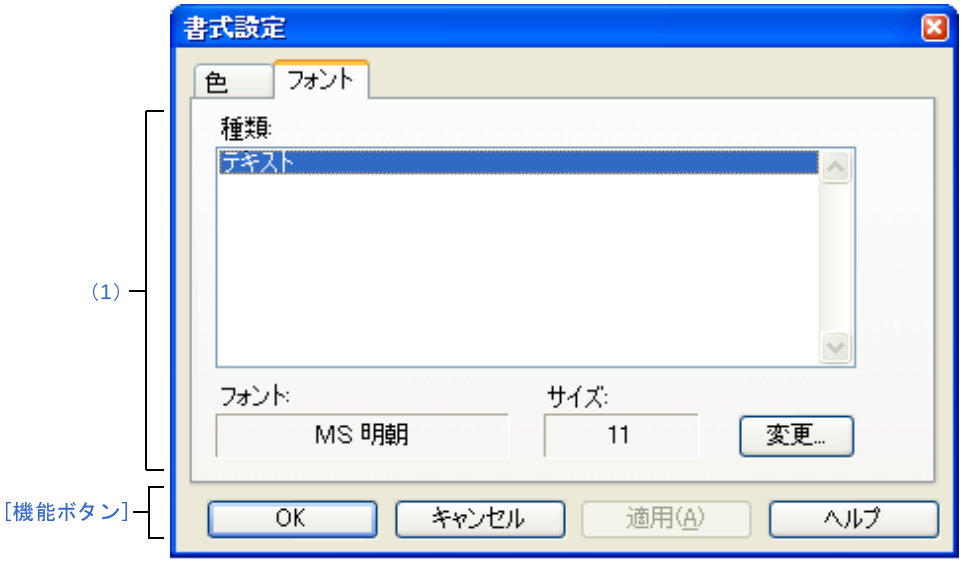


図 A—58　書式設定 ダイアログ：［フォント］タブ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［［色］タブ］
- ［［フォント］タブ］
- ［機能ボタン］

[オープン方法]

- 信号データエディタ ウインドウ／タイミングチャート ウインドウ／シリアル ウインドウのいずれかにフォーカスがある状態で，［オプション］メニュー→［ウインドウのカスタマイズ ...］を選択

[［色］タブ]

（1）色設定エリア

対象ウインドウ内の各部位の色を設定／変更します。

種類	色の変更が可能な部位がリスト表示されます。 表示される部位は，対象となるウインドウにより異なります。
色	リスト内を選択することにより，該当部位の現在の設定色が表示されます。
［変更 ...］ボタン	リスト内の該当部位の現在の設定色を変更します。

[［フォント］タブ]

（1）フォント設定エリア

対象ウインドウ内で使用する各部位のテキストのフォントを設定／変更します。

種類	フォントの変更が可能な部位がリスト表示されます。
フォント	リスト内を選択することにより，該当部位の現在のフォント名が表示されます。
サイズ	リスト内を選択することにより，該当部位の現在のフォント・サイズが表示されます。
［変更 ...］ボタン	リスト内の該当部位の現在の設定フォントを変更します。

[機能ボタン]

ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。
適用	選択不可
ヘルプ	このダイアログのヘルプを表示します。

信号データエディタ ウインドウ

入力端子に入力する信号データの作成，編集を行います。

作成した信号データは，［編集］メニュー→［信号入力］→［開始］を選択することにより，シミュレーション中の入力端子へ入力されます。また，［ファイル］メニュー→［上書き保存］／［名前を付けて保存 ...］の選択により，信号データ・ファイル（*.wvi），およびプロジェクト・ファイルへ保存することができます。

なお，保存した信号データは，［ファイル］メニュー→［開く ...］の選択，またはプロジェクト・ファイルのロードにより復元することができます。

注意 1.　保存した信号データ・ファイルをオープンする際，またはプロジェクト・ファイルをオープンする際に，信号データ・ファイルを作成した時点でのマイクロコントローラとは異なるマイクロコントローラでシミュレータ GUI が起動されていた場合，そのマイクロコントローラに存在しない端子名の設定は復元されません。

2.　このウインドウから，メイン・クロック入力，およびサブ・クロック入力を行うことはできません。メイン・クロック，およびサブ・クロックの発振周波数の設定は，プロパティ パネルの［接続用設定］タブで行ってください。

3.　プログラムがブレークしている間に信号データの入力を開始した場合，実際に信号入力が開始されるのはプログラムの実行直後になります。

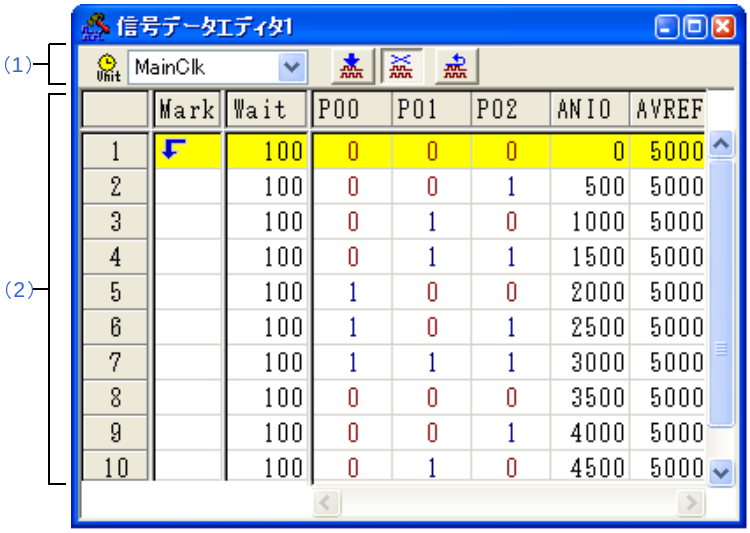
備考 1.　このウインドウでは，次の信号データの表示，および編集を行うことができます。

- 新規に作成する信号データ
- 以前に作成した信号データ・ファイル
- 以前にシミュレーションし，出力信号データとして保存した信号データ・ファイル

2.　このウインドウのタイトルバー上には，プロジェクト・ファイルを読み込んだ場合，“*プロジェクト・ファイル名＋数字*（0 から連番）wvi”を表示します。

ただし，PM+ のプロジェクト・ファイルを読み込み，CubeSuite+ のプロジェクト・ファイルに保存した場合は，それ以降，“*プロジェクト・ファイル名*＋ CS ＋*数字*（0 から連番）.wvi”と表示します。

図 A—59　信号データエディタ 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］
- ［専用メニュー（信号データエディタ ウインドウ）］
- ［［信号データエディタ］ツールバー］
- ［コンテキスト・メニュー］
- ［操作方法］

［オープン方法］

- ボタンをクリック
- ［シミュレータ］メニュー→［信号データエディタ］を選択

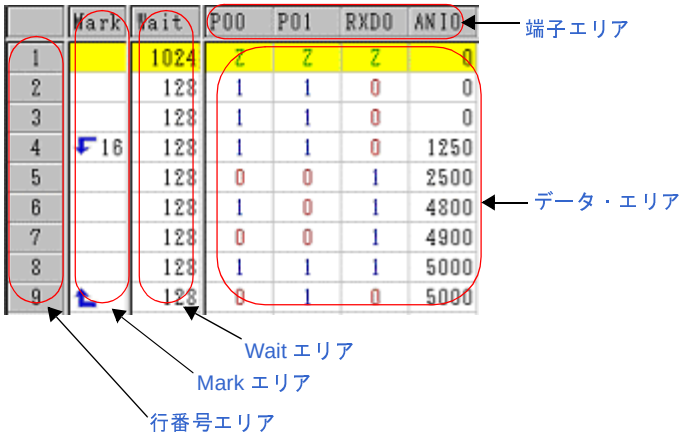
［各エリアの説明］

（1）インフォメーション・バー

このエリアは［表示］メニュー→［インフォメーション バー］により，表示／非表示の選択が可能です。

MainClk	ウエイト時間の単位をドロップダウン・リストから変更します。 ウエイト時間の単位は，［編集］メニュー→［時間単位］からも変更可能です。
	プログラム実行中に，このボタンをクリックすると信号入力を開始します。 プログラム停止中に，このボタンをクリックしておくと，次回プログラム実行開始のタイミングで信号入力が自動的に開始されます。
	プログラム実行中に，このボタンをクリックすると信号入力を停止します。 プログラム停止中に，このボタンをクリックしておくと，プログラム実行を開始しても信号入力が自動的には行われません。
	入力カレント行（黄色の行）を先頭に戻します。

（2）クライアント・エリア



端子エリア	入力端子名を示します。 使用する入力端子の選択は，ツールバーの ボタンのクリック，または［編集］メニュー→［端子選択 ...］によりオープンする端子選択 ダイアログで行います。 なお，［編集］メニュー→［端子状態］により端子へのデータ入力の有効／無効の制御が可能です。		
行番号エリア	行番号を示します。 このエリアは行単位で編集する際に使用します。 なお，信号データとして，最大 1,048,576（= 1M）行まで入力することができます。		
Mark エリア	設定した入力値に対するループ情報を示します。 ループ情報の設定は，対象枠内におけるコンテキスト・メニューからの選択，または［編集］メニュー→［マーク設定］により行います。 ループ情報を設定すると，次のマークが表示されます。		
		ループの開始位置（無限ループ）	
		ループの開始位置（ループカウント付き）	
		ループの終了位置	
Wait エリア	設定した入力値が端子に入力されるタイミングをウエイト時間として示します。ウエイト時間の設定は，対象枠内に直接数値を書き込むことにより行います。 0 ～ 4,294,967,295 までの整数値（10 進数）が指定可能です（4,294,967,295 を越える数値を設定する場合は，もう 1 段を使用することで設定可）。 なお，ウエイト時間の単位は，［編集］メニュー→［時間単位］により選択可能です。		
データ・エリア	端子に入力する入力値を示します。 入力値の設定は，対象枠内に直接数値を書き込むことにより行います。 なお，端子の種類によって次のように入力規則が異なります。		
	デジタル端子	次のいずれかの 1 文字列	
		0	LOW 信号
		1	HIGH 信号
		Z	Hi-Z 信号（大小文字不問）
	アナログ端子	0 ～ 5000 の範囲の値（10 進数）（単位：mV）	

［専用メニュー（信号データエディタ ウインドウ）］

（1）［編集］メニュー

元に戻す	選択不可
やり直し	選択不可
切り取り	選択範囲を切り取りクリップボードに保存します。
コピー	選択範囲をコピーしクリップボードに保存します。
貼り付け	クリップボードの内容を選択位置に貼り付けます。
削除	選択範囲を削除します。
すべて選択	すべての表示データを選択します。
検索	選択不可

端子選択 ...	端子選択 ダイアログをオープンします。データを作成／編集する端子を選択します。
時間単位	ウエイト時間の単位を選択します。
メインクロック	ウエイト時間の単位をメイン・クロックとします（デフォルト）。
マイクロ秒	ウエイト時間の単位をマイクロ秒とします。
ミリ秒	ウエイト時間の単位をミリ秒とします。
端子状態	選択した端子の入力状態を選択します。
有効	端子へのデータ入力を有効にします（デフォルト）。
無効	端子へのデータ入力を無効にします。
マーク設定	選択した Mark エリアにループ情報を設定します。
ループ開始	ループ開始マークを設定します。
ループ終了	ループ終了マークを設定します。
ループ詳細設定	ループ設定 ダイアログをオープンします。ループ情報の詳細を設定します。
信号入力	信号データをシミュレータに入力します。
開始	信号入力を開始します。
停止	信号入力を停止します。
リセット	入力カレント行を先頭に戻します。

(2)［表示］メニュー

インフォメーション バー	インフォメーション・バーの表示／非表示を切り替えます。

(3)［オプション］メニュー

ウインドウのカスタマイズ ...	書式設定 ダイアログをオープンします。

[[信号データエディタ］ツールバー]

	端子選択 ダイアログをオープンします。データを作成／編集する端子を選択します。
	プログラム実行中に，このボタンをクリックすると信号入力を開始します。 プログラム停止中に，このボタンをクリックしておくと，次回プログラム実行開始のタイミングで信号入力が自動的に開始されます。
	プログラム実行中に，このボタンをクリックすると信号入力を停止します。 プログラム停止中に，このボタンをクリックしておくと，プログラム実行を開始しても信号入力が自動的には行われません。
	入力カレント行（黄色の行）を先頭に戻します。

[コンテキスト・メニュー]

クライアント・エリアの各エリアにおいて，次のコンテキスト・メニューを表示します。

(1) 端子エリア

有効	端子へのデータ入力を有効にします（デフォルト）。
無効	端子へのデータ入力を無効にします。
端子選択 ...	端子選択 ダイアログをオープンします。データを作成／編集する端子を選択します。

(2) 行番号

切り取り	選択範囲を切り取りクリップボードに保存します。
コピー	選択範囲をコピーしクリップボードに保存します。
貼り付け	クリップボードの内容を選択位置に貼り付けます。
削除	選択範囲を削除します。

(3) Mark エリア

切り取り	選択セルを切り取りクリップボードに保存します。
コピー	選択セルをコピーしクリップボードに保存します。
貼り付け	クリップボードの内容を選択位置に貼り付けます。
削除	選択セルを削除します。
ループ開始	ループ開始マークを設定します。
ループ終了	ループ終了マークを設定します。
ループ詳細設定	ループ設定 ダイアログをオープンします。ループ情報の詳細を設定します。

(4) Wait エリア

切り取り	選択セルのデータを切り取りクリップボードに保存します。切り取られたデータは“0”になります。
コピー	選択セルのデータをコピーしクリップボードに保存します。
貼り付け	クリップボードの内容を選択位置に貼り付けます。
削除	選択セルのデータを削除します。削除されたデータは“0”になります。

(5) データ・エリア

切り取り	選択セルのデータを切り取りクリップボードに保存します。切り取られたデータは“Z（Hi-Z）”になります。
コピー	選択セルのデータをコピーしクリップボードに保存します。
貼り付け	クリップボードの内容を選択位置に貼り付けます。
削除	選択セルのデータを削除します。削除されたデータは“Z（Hi-Z）”になります。
信号入力開始	信号入力を開始します。

信号入力停止	信号入力を停止します。
信号入力リセット	入力カレント行を先頭に戻します。

[操作方法]

（1）端子の選択
（2）信号データの作成
（3）データのコピーと貼り付け編集
（4）行単位の編集
（5）信号の入力
（6）CPU リセット時の動作

（1）端子の選択

信号データを作成するためには，最初に使用する端子を選択する必要があります。

端子の選択は，ツールバーの ボタンのクリック，または［編集］メニュー→［端子選択 ...］の選択によりオープンする端子選択 ダイアログで行います。これにより，端子エリアに選択した端子名が表示されます。

（2）信号データの作成

各端子へ入力する信号データを作成します。

（a）入力値の設定

データ・エリアにおいて，各端子へ入力する値を設定します（「データ・エリア」参照）。

（b）入力タイミングの設定

Wait エリアにおいて，各端子へ入力するタイミングをウエイト時間として設定します（「Wait エリア」参照）。

（c）ループ情報の設定

上記（a）～（b）で設定した信号データをループ処理させたい場合は，ループ情報を設定します。

ループ情報の設定は，Mark エリアのループ開始位置で，コンテキスト・メニューの［ループ開始］を選択し，ループ終了位置で［ループ終了］を選択します。

なお，この際に，ループ・カウントを指定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［ループ詳細設定 ...］を選択することでオープンするループ設定 ダイアログにより，ループ・カウントの設定を行います。

ループ情報の設定が完了すると，該当するループ情報のマークが表示されます（「Mark エリア」参照）。

（3）データのコピーと貼り付け編集

Mark エリア /Wait エリア／データ・エリア上の設定値は，コピーと貼り付けを行うことができます。

ただし，コピーしたデータは同一エリア内でのみの貼り付けとなります。

コピー	1 つ，または複数（範囲）のセルを選択した状態で，［編集］メニュー→［コピー］の選択（または［Ctrl］キー +［C］の入力）により行います。

貼り付け	1つ，または複数（範囲）のセルを選択した状態で，［編集］メニュー→［貼り付け］の選択（または［Ctrl］キー＋［V］の入力）により行います。 複数（範囲）のセルを選択した場合は，コピーしたデータを繰り返して貼り付けます。

（4）行単位の編集

行単位の編集は，行番号エリアを選択することにより行います。

方法は「（3）データのコピーと貼り付け編集」と同様です。

なお，行の貼り付け（挿入）時に貼り付けられるデータは選択行の直前に挿入されます。

（5）信号の入力

作成した信号データをシミュレーション実行時にシミュレータの入力端子へ入力します。

この際，プログラムがブレークすると，現在信号入力中の行が黄色く強調表示され（書式設定 ダイアログの［入力カレント行］で変更可能）信号入力の進捗状況を表示します。

信号データの入力操作には次の種類があります。

信号入力の開始	ボタンのクリック，または［編集］メニュー→［信号入力］→［開始］を選択します。 これにより，入力カレント行（強調表示されている行）から信号入力が開始されます。
信号入力の停止	ボタンのクリック，または［編集］メニュー→［信号入力］→［停止］を選択します。 これにより信号入力が停止されます。
信号のリセット	ボタンのクリック，または［編集］メニュー→［信号入力］→［リセット］を選択します。 これにより入力カレント行が先頭に戻ります。なお，信号入力中に信号のリセットを行った場合には先頭から引き続き入力が継続されます。

備考　端子名を選択したのち，［編集］メニュー→［端子状態］→［有効］／［無効］を選択することにより，端子への信号データ入力を制御することができます。

（6）CPU リセット時の動作

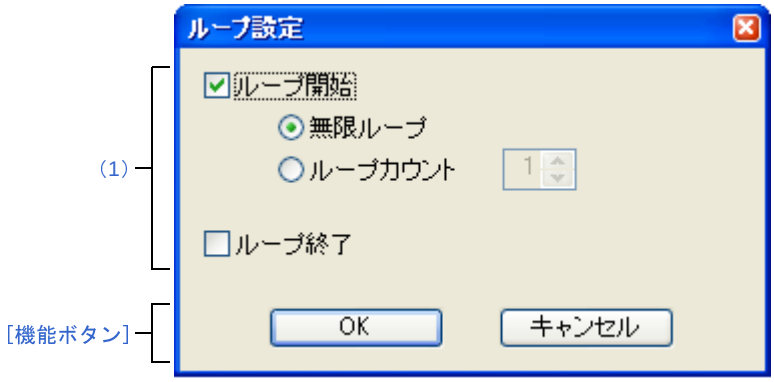
CPU リセットが発生した場合，入力カレント行は先頭に戻ります。

信号入力中に CPU リセットが発生した場合は，先頭から引き続き入力が継続されます（ボタンのクリックと同等）。

ループ設定 ダイアログ

信号データエディタ ウインドウのループ情報に関する詳細設定（ループ開始／終了，ループ・カウントなど）を行います。

図 A—60　ループ設定 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］

信号データエディタ ウインドウにおける次のいずれか

- Mark エリアをダブルクリック
- Mark エリアを選択したのち，［編集］メニュー→［マーク設定］→［ループ詳細設定］を選択

［各エリアの説明］

(1) ループ情報設定エリア

ループ開始	ループ開始を設定する場合，チェックします。		
	無限ループ	無限ループを設定する場合，選択します。	
	ループカウント	ループ・カウントを設定する場合，選択します。 スピン・ボタンにより，次のカウント値を指定します。	
		0	ループをスキップします。
		1 ～ 99	指定カウント分のループをします。
ループ終了	ループ終了を設定する場合，チェックします。		

[機能ボタン]

ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。

端子選択 ダイアログ

信号データエディタ ウインドウ，およびタイミングチャート ウインドウで表示する端子を設定します。

設定した端子情報は，［保存］ボタンにより端子情報ファイル（*.pin）として保存することができます。また，保存した端子情報は，［読み込み］ボタンにより復元することができます。

図 A—61　端子選択 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］

信号データエディタ ウインドウ／タイミングチャート ウインドウにフォーカスがある状態で次のいずれか

- ボタンをクリック
- ［編集］メニュー→［端子選択 ...］を選択

[各エリアの説明]

(1) 接続端子設定エリア

エリア右側のスクロール・バーを操作することにより，最大 256 端子まで設定することができます。

接続端子名	接続する端子名を指定します。 指定は，ドロップダウン・リストからの選択，または直接入力により行います。
アナログ	指定した端子をアナログ端子として使用する場合，対応するチェック・ボックスをチェックします。
機能名	指定した端子に機能名を設定します。 入力した文字列を端子名としてウインドウ上に表示します。 何も指定しない場合，端子名を表示します。

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

[機能ボタン]

ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。 このダイアログを呼び出したウインドウの Pin 欄に端子名（または表示名）が表示されます。
保存	表示内容を端子情報ファイル（*.pin）に保存します。
読み込み	指定した端子情報ファイル（*.pin）を読み込みます。
クリア	設定内容をすべて削除します。
キャンセル	設定を無視し，このダイアログをクローズします。

タイミングチャート ウインドウ

端子に対する入力信号と出力信号をタイミング・チャートで表示します。

このウインドウでは，メイン・クロック単位で時間計測を行います。

ブラウズした信号データは，［ファイル］メニュー→［上書き保存］／［名前を付けて保存 ...］により，タイミングチャート・ファイル（*.wvo）として保存することができます。

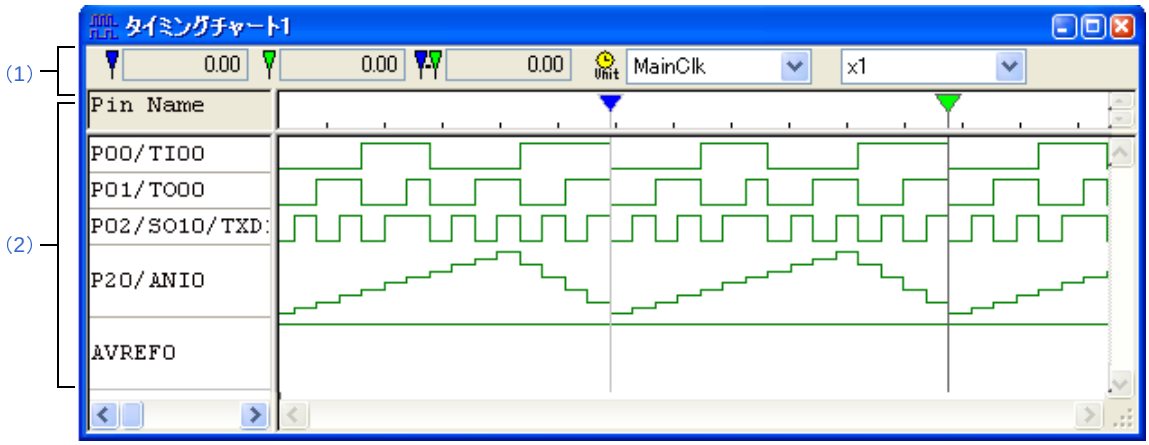
保存した信号データは，［ファイル］メニュー→［開く ...］の選択により復元することができます。

なお，プロジェクト・ファイルとして保存した場合，信号データは保存されませんが，設定した端子情報は保存されます（測定結果を保存する必要がない場合，この方法で問題ありません）。

注意 1.　保存したタイミングチャート・ファイルをオープンする際，またはプロジェクト・ファイルをオープンする際に，タイミングチャート・ファイルを保存した時点でのマイクロコントローラとは異なるマイクロコントローラでシミュレータ GUI が起動されていた場合，そのマイクロコントローラに存在しない端子名の設定は復元されません。

2.　このウインドウでメイン・クロック波形，およびサブ・クロック波形を表示することはできません。また，外部バス・インタフェース機能使用時に，外部バス・インタフェース機能で使用する端子の波形を表示することはできません。

図 A—62　タイミングチャート 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］
- ［専用メニュー（タイミングチャート ウインドウ）］
- ［［タイミングチャート］ツールバー］
- ［コンテキスト・メニュー］
- ［操作方法］

[オープン方法]

- ボタンをクリック
- ［シミュレータ］メニュー→［タイミングチャート］を選択

[各エリアの説明]

(1) インフォメーション・バー

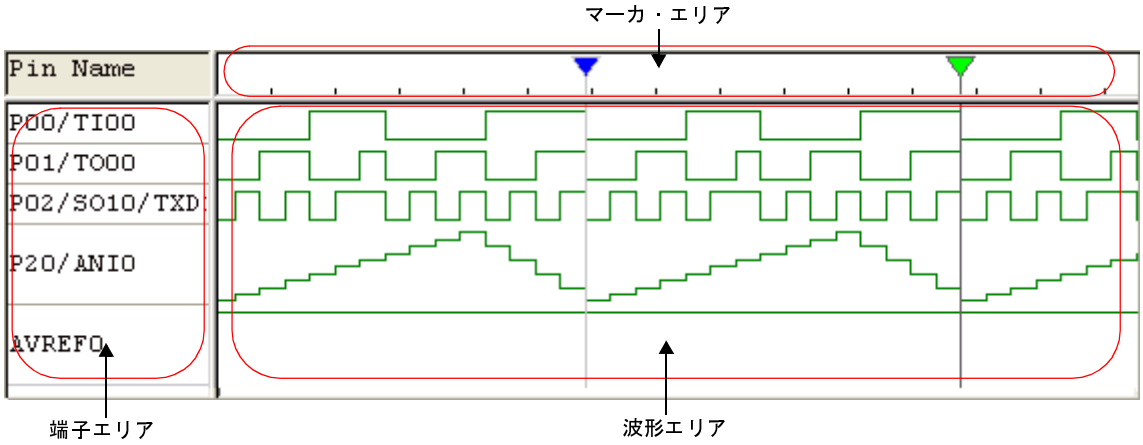
このエリアは［表示］メニュー→［インフォメーション バー］により，表示／非表示の選択が可能です。

5553364.00	シミュレーション開始からマーカ A の位置までの時間を表示します。
5554860.00	シミュレーション開始からマーカ B の位置までの時間を表示します。
1496.00	マーカ A，B 間の時間を絶対値で表示します。
MainClk	マーカ A，B の位置情報の単位をドロップダウン・リストから選択します。［編集］メニュー→［時間単位］，およびコンテキスト・メニューの［時間単位］からも変更可能です。
x1/2	波形データの表示倍率をドロップダウン・リストから選択します。なお，表示倍率を変更する際，波形データの一部が消去される場合には，確認 ダイアログが表示されます。

備考 1. シミュレーション開始から，最大 4,294,967,262 クロックまでカウントすることができます。カウントが最大値に達した場合 0 に戻り，再度カウントを開始します。

2. プログラム実行中は表示倍率の設定欄が淡色表示になり変更できません。

(2) クライアント・エリア



端子エリア	このウインドウで表示する端子名を表示します。 端子の選択は，［編集］メニュー→［端子選択 ...］によりオープンする端子選択 ダイアログで行います。

マーカエリア	マーカ A，B のヘッド部分を表示します。 これらのマーカはドラッグにより移動可能です。	
		マーカ A
		マーカ B
波形エリア	端子エリアに指定した端子のデータをタイミング・チャート表示します。 なお，信号の種類により，デフォルトで次のように色分けされます。	
	緑	端子の HIGH，LOW 信号
	赤	ハイ・インピーダンス信号
	青	未サンプリング状態の信号

備考 1. 端子データを格納するバッファはリング・バッファ形式のため，バッファがいっぱいになると最古のデータは，最新のデータによって上書きされます。
なお，バッファのサイズは，次のいずれかまでです。
- 端子変化点： 4,096 箇所
- クロック数： 2,147,483,631
- 描画横幅： 134,217,711 ピクセル

2. ［オプション］メニュー→［ウインドウのカスタマイズ ...］の選択でオープンする書式設定 ダイアログにより，このエリア内の色／フォントを変更することができます。

［専用メニュー（タイミングチャート ウインドウ）］

（1）［編集］メニュー

クリア	すべての波形データを削除します。
検索 ...	データ検索 ダイアログがオープンします。 データの検索を行います。
後方検索	選択端子の変化点を後方（左方向）に検索します。
前方検索	選択端子の変化点を前方（右方向）に検索します。
端子選択 ...	端子選択 ダイアログをオープンします。データを表示する端子を選択します。
時間単位	マーカ A，B の位置情報の単位を選択します。
メインクロック	ウエイト時間の単位をメイン・クロックとします（デフォルト）。
マイクロ秒	ウエイト時間の単位をマイクロ秒とします。
ミリ秒	ウエイト時間の単位をミリ秒とします。

(2)［表示］メニュー

波形表示	マーカ・エリアと波形エリアの表示／非表示を切り替えます。
インフォメーション バー	インフォメーション・バーの表示／非表示を切り替えます。
ズーム	波形データの表示倍率を選択します。なお，表示倍率を変更する際，波形データの一部が消去される場合には，確認 ダイアログが表示されます。
x 1/32	表示倍率を 1/32 倍にします。
x 1/16	表示倍率を 1/16 倍にします。
x 1/8	表示倍率を 1/8 倍にします。
x 1/4	表示倍率を 1/4 倍にします。
x 1/2	表示倍率を 1/2 倍にします。
x 1	表示倍率を 1 倍にします。
x 2	表示倍率を 2 倍にします。
x 4	表示倍率を 4 倍にします。
x 8	表示倍率を 8 倍にします。
x 16	表示倍率を 16 倍にします。
x 32	表示倍率を 32 倍にします。

(3)［オプション］メニュー

ウインドウのカスタマイズ ...	書式設定 ダイアログをオープンします。

［[タイミングチャート］ツールバー］

	すべての波形データを削除します。
	選択端子の変化点を後方（左方向）に検索します。
	選択端子の変化点を前方（右方向）に検索します。
	端子選択 ダイアログをオープンします。 データを表示する端子を選択します。
x 1	波形データの表示倍率をドロップダウン・リストから選択します。なお，表示倍率を変更する際，波形データの一部が消去される場合には，確認 ダイアログが表示されます。

[コンテキスト・メニュー]

クライアント・エリアにおいて，次のコンテキスト・メニューを表示します。

クリア	すべての波形データを削除します。
検索	データ検索 ダイアログをオープンします。 データの検索を行います。
後方検索	選択端子の変化点を後方（左方向）に検索します。
前方検索	選択端子の変化点を前方（右方向）に検索します。
端子選択 ...	端子選択 ダイアログをオープンします。 データを表示する端子を選択します。
波形表示	マーカ・エリアと波形エリアの表示／非表示を切り替えます。
時間単位	マーカ A，B の位置情報の単位を選択します。
ズーム	波形データの表示倍率を選択します。なお，表示倍率を変更する際，波形データの一部が消去される場合には，確認 ダイアログが表示されます。
マーカ A の配置	マーカ A をマウス・カーソル位置に配置します。 ［Shift］キー＋クリックでも同様の操作が可能です。
マーカ B の配置	マーカ B をマウス・カーソル位置に配置します。 ［Ctrl］キー＋クリックでも同様の操作が可能です。

[操作方法]

（1）端子の選択
（2）タイミング・チャートの表示
（3）タイミング・チャートのクリア
（4）タイミング・チャートのタイミング計測
（5）データの検索
（6）リセット時の動作

（1）端子の選択

タイミング・チャートを表示するためには，最初に表示する端子を選択する必要があります。

端子の選択は，ツール・バーの ボタンのクリック，または［編集］メニュー→［端子選択 ...］の選択によりオープンする端子選択 ダイアログで行います。これにより，端子エリアに選択した端子名が表示されます。

（2）タイミング・チャートの表示

プログラムを実行することにより，選択した端子の波形がタイミング・チャート形式で表示されます。

備考　タイミング・チャートを非表示にすることでシミュレーション速度を速めることができます。
非表示にするには，［表示］メニュー→［波形表示］を選択します（チェックなしにする）。

非表示時は，マーカ・エリア，および波形エリアは淡色表示され，中央に“Display OFF”と表示されます。

(3) タイミング・チャートのクリア

［編集］メニュー→［クリア］の選択により，タイミング・チャートの表示波形はすべてクリアされます。

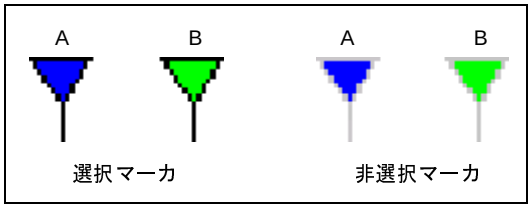
(4) タイミング・チャートのタイミング計測

マーカ A，B を使用し，2 箇所をマーキングすることで 2 点間のタイミング計測を行います。各マーカの時間，およびマーカ間の時間はインフォメーション・バーに表示されます。

マーカの配置は，マーカ・ヘッドをドラッグすることにより目的の位置への移動することができます。また，コンテキスト・メニュー→［マーカ A の配置］／［マーカ B の配置］の選択によっても，マーカが現在のマウス・カーソルの位置に移動します。

なお，最後にクリックしたマーカは選択マーカとなり，データの検索の対象となります。

図 A—63　マーカ A とマーカ B



(5) データの検索

タイミング・チャートのデータ検索機能には，次の 2 種類があります。

(a) 単純検索

単純検索は 1 つの端子の変化点を検索する機能です。

検索したい端子名を端子エリアで 1 つ選択し，［編集］メニュー→［後方検索］／［前方検索］を選択します。これにより，変化点が検索されたデータ位置に選択マーカが移動します。

(b) 詳細検索

詳細検索では複数端子の様々なデータの組み合わせによる検索を行います。

検索データの設定は，［編集］メニュー→［検索 ...］の選択によりオープンするデータ検索 ダイアログで行います。検索結果は単純検索と同様に，ヒットしたデータ位置に選択マーカが移動します。

(6) リセット時の動作

CPU リセット，またはシミュレータ GUI のリセットが発生した場合，タイミング・チャートの表示波形はすべてクリアされます。

データ検索 ダイアログ

タイミングチャート ウインドウで表示されている信号データの検索を行います。

注意 1.　アナログ入出力信号は検索できません。

2.　プログラム実行中に，このダイアログをオープンすることはできません。

図 A—64　データ検索 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］

タイミングチャート ウインドウにフォーカスがある状態で次のいずれか

- ボタンをクリック
- ［編集］メニュー→［検索 ...］を選択

[各エリアの説明]

(1) 検索端子設定エリア

検索条件を複数個指定した場合，すべての検索条件を満たす信号データを検索します。

右側のスクロール・バーを操作することにより，検索条件は最大 48 個指定できます。

検索端子	検索する端子名を指定します。 指定は，ドロップダウン・リストからの選択，または直接入力により行います。 空白を入力するとデータ検索の対象外となり［検索データ］を入力不可にします。	
検索データ	指定した端子で検索するデータをドロップダウン・リストから選択します。	
	------	指定しません。
	Rising Edge	信号データの立ち上がりを検索します。
	Falling Edge	信号データの立ち下がりを検索します。
	Rise/Fall Edge	信号データの立ち上がり／立ち下がりを検索します。
	High	信号データが HIGH の状態を検索します。
	Low	信号データが LOW の状態を検索します。
	Hi Z	信号データがハイ・インピーダンスの状態を検索します。
検索する方向	データ検索する方向をオプション・ボタンにより選択します。 ［次を検索］ボタンをクリックした際，このエリアで選択した方向へ検索を行います。	
	後方	後方（現在位置より古い時間）のデータを検索します。
	前方	前方（現在位置より新しい時間）のデータを検索します（デフォルト）。

[機能ボタン]

ボタン	機能
次を検索	選択している方向へ検索を行います。 検索終了後，再度クリックすることにより次のデータを検索します。
キャンセル	データ検索を中止し，このダイアログをクローズします。

入出力パネル ウインドウ

疑似的なターゲット・システムの構築，および作成した接続部品の操作を行います。

このウインドウでは，使用する接続部品（図形オブジェクト／部品オブジェクト）の作成／設定を行うことで疑似的なターゲット・システムを構築することができます。設定を行った接続部品はこのウインドウ内の任意の位置へ配置することができ，シミュレーション中に信号処理としてそれらを操作することができます。

ウインドウ内に配置した接続部品の情報は，［ファイル］メニュー→［上書き保存］／［名前を付けて保存 ...］の選択により，入出力パネル・ファイル（*.pnl），またはプロジェクト・ファイルへ保存することができます。

なお，保存した部品情報ファイのル内容は［ファイル］メニュー→［開く ...］の選択，またはプロジェクト・ファイルのロードにより復元することができます。

注意 1.　保存した入出力パネル・ファイルをオープンする際，そのファイルを作成した時点のマイクロコントローラとは異なるマイクロコントローラでシミュレータ GUI が起動されていた場合，そのマイクロコントローラに存在しない端子に接続されていた部品の端子設定情報は復元されません（各部品の設定ダイアログ内の［接続端子］が空欄になります）。

2.　プログラムがブレークしている間に信号を入力した場合（ボタンを押した場合など），実際に信号が変化するのはプログラムの実行直後になります。

備考　このウインドウのタイトルバー上には，プロジェクト・ファイルを読み込んだ場合，“*プロジェクト・ファイル名＋数字*（0 から連番）.pnl”を表示します。
ただし，PM+ のプロジェクト・ファイルを読み込み，CubeSuite+ のプロジェクト・ファイルに保存した場合は，それ以降，“*プロジェクト・ファイル名*＋ CS ＋*数字*（0 から連番）.pnl”と表示します。

図 A—65　入出力パネル 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］
- ［専用メニュー／ツールバー（入出力パネル ウインドウ）］
- ［コンテキスト・メニュー］
- ［操作方法］

［オープン方法］

- ボタンをクリック
- ［シミュレータ］メニュー→［入出力パネル ...］を選択

［各エリアの説明］

（1）クライアント・エリア

擬似的なターゲット・システムを構築するために，使用する接続部品（図形オブジェクト／部品オブジェクト）の作成，および設定を行うエリアです（［操作方法］参照）。

［専用メニュー／ツールバー（入出力パネル ウインドウ）］

入出力パネル ウインドウに関する操作を行うメニュー項目，ツールバー上のボタンは次のとおりです。

（1）［編集］メニュー

作成したオブジェクトに対して基本的な編集操作を行う場合に選択します。

元に戻す	オブジェクトの移動等，直前に行った操作を元に戻します。 変更内容の復帰は，5 回前の状態まで可能です。
やり直し	［元に戻す］で戻した状態を復帰します。
切り取り	選択範囲を切り取りクリップボードに保存します。
コピー	選択範囲をコピーしクリップボードに保存します。
貼り付け	クリップボードの内容を貼り付けます。
削除	選択範囲を削除します。
すべて選択	ウインドウ上のすべてのオブジェクトを選択します。
グループ化	選択しているオブジェクトをグループ化します。
グループ解除	選択しているオブジェクトのグループ状態を解除します。
最前面へ移動	選択しているオブジェクトをパネルの最前面に移動します。
最背面へ移動	選択しているオブジェクトをパネルの最背面へ移動します。
前面へ移動	選択しているオブジェクトを一つ前面へ移動します。
背面へ移動	選択しているオブジェクトを一つ背面へ移動します。

(2)［表示］メニュー

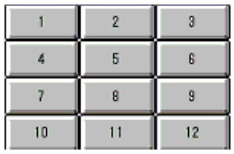
ツールバー，ステータスバーの表示状態の切り替え，およびウインドウ内の各種情報を表示／非表示する場合に選択します。

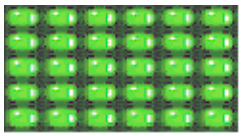
ツール バー	［図形］ツールバー／［部品］ツールバーの表示／非表示を切り替えます。
ステータス バー	ステータスバーの表示／非表示を切り替えます。
接続部品一覧 ...	部品一覧 ダイアログをオープンします。 このウインドウ上に存在するすべての図形オブジェクト／部品オブジェクトの一覧を表示します。
グリッド	グリッドの表示／非表示をします。
プロパティ	選択している図形オブジェクト／部品オブジェクトの設定ダイアログをオープンします。

(3)［部品］メニュー／［部品］ツールバー

シミュレータ GUI が提供する接続部品（部品オブジェクト）を新規作成／配置する際に選択します（「(3) 部品オブジェクトの作成」参照）。

なお，［部品］メニューの各項目は，［部品］ツールバーのボタンにより同様の動作を行うことができます。

メニュー項目	ボタン	機能
ボタン		デジタル入力用スイッチです。 任意の端子に対する接続が可能で，表示されたボタンをクリックすることで接続端子へデジタル入力値を与えることができます。
例）		
アナログボタン		アナログ入力用スイッチです。 任意の端子に対する接続が可能で，表示されたボタンをクリックすることで接続端子へアナログ入力値を与えることができます。
例）		
キーマトリクス		複数の端子をマトリクス状に接続し，その接点を各種のキーとみなし，キーをクリックするとある種の状態になる部品です。 任意の端子に対する接続が可能で，複数のキーを使用した入力が可能です。
例） 		
レベルゲージ		電圧源などのアナログ・データの入力用として，ある一定範囲のデータを加減に設定できる部品です。 A/D コンバータを接続した端子に対して，指定した範囲内の任意の値を与えることができます。
例）		
LED		発光ダイオード (Light Emitting Diode）です。 任意の端子に対する接続が可能で，端子の出力を LED の点灯／消灯で表示します。
例）		
7 セグメント LED		LED7 個を数字の字画に近似させ 1 つのパッケージにした部品です。 桁信号に割り当てた端子の出力がアクティブ時に，対応する 7 セグメント LED を点灯／消灯で表示します。
例）		

メニュー項目	ボタン	機能
14 セグメント LED		LED14 個をアルファベットの字画に近似して 1 つのパッケージにした部品です。 桁信号に割り当てた端子の出力がアクティブ時に，対応する 14 セグメント LED を点灯／消灯で表示します。
例）		
マトリクス LED		複数の LED をマトリクス状に配置して 1 つのパッケージにした部品です。 割り当てた端子の出力がアクティブ時に，対応する 14 セグメント LED を点灯／消灯で表示します。
例） 		
ブザー		ブザーです。 端子と接続したブザーは，接続した端子からの出力情報をビットマップやブザー音で表します。
例）		
プルアップ／プルダウン設定 ...		プルアップ／プルダウン設定 ダイアログをオープンします。 端子にプルアップ抵抗／プルダウン抵抗を接続することができます。

(4)［図形］メニュー／［図形］ツールバー

このウインドウの動作モードの設定，および接続部品（図形オブジェクト）を新規作成／配置する際に選択します（「(2) 図形オブジェクトの作成」参照）。

なお，［図形］メニューの各項目は，［図形］ツールバーのボタンにより同様の動作を行うことができます。

メニュー項目	ボタン	
選択		このウインドウの動作モードを編集モードにします。 マウス・カーソルが矢印になり，オブジェクトの編集を可能にします。
入力シミュレーション		このウインドウの動作モードを入力シミュレーション・モードにします。 マウス・カーソルが手の形になり，接続部品（部品オブジェクト）への入力操作を可能にします。
線		マウス・カーソルが十字（+）になり，線の作画を可能にします。
四角形		マウス・カーソルが十字（+）になり，四角形の作画を可能にします。
丸四角形		マウス・カーソルが十字（+）になり，丸みを帯びた四角形の作画を可能にします。
楕円		マウス・カーソルが十字（+）になり，楕円の作画を可能にします。
多角形		マウス・カーソルが十字（+）になり，多角形の作画を可能にします。
扇型		マウス・カーソルが十字（+）になり，扇型の作画を可能にします。
文字		マウス・カーソルが十字（+）になり，文字の作成を可能にします。
ビットマップの貼り付け ...	－	選択しているビットマップ・ファイルをこのウインドウに貼り付けます。

メニュー項目	ボタン	
線の色 ...		次の色の設定 ダイアログをオープンします。 選択しているオブジェクトの線の色を，選択した色に変更します。
塗りつぶしの色 ...		色の設定 ダイアログをオープンします。 選択しているオブジェクトの塗りつぶしの色を，選択した色に変更します。
フォントの指定 ...	－	次のフォント ダイアログをオープンします。 選択しているオブジェクトのフォントを，選択したフォントに変更します。
線のスタイル		選択しているオブジェクトの線のスタイルを変更します。
16pt		線の太さを 16pt に設定します。
12pt		線の太さを 12pt に設定します。
8pt		線の太さを 8pt に設定します。
4pt		線の太さを 4pt に設定します。
2pt		線の太さを 2pt に設定します。
1pt		線の太さを 1pt に設定します。
なし		線を描画しません。

メニュー項目	ボタン	
点線のスタイル		選択しているオブジェクトの線のスタイルを変更します。
実線		実線を描画します。
破線		破線を描画します。
点線		点線を描画します。
一点鎖線		一点鎖線を描画します。
二点鎖線		二点鎖線を描画します。

[コンテキスト・メニュー]

編集モード選択時，次のコンテキスト・メニューを表示します。

コピー	選択しているオブジェクトをコピーします。
貼り付け	クリップボードの内容を貼り付けます。
削除	選択しているオブジェクトを削除します。
グループ化	カスケード・メニューより選択します。
グループ化	選択しているオブジェクトをグループ化します。
グループ化解除	選択しているオブジェクトのグループ状態を解除します。
順序	カスケード・メニューより選択します。
最前面へ移動	選択しているオブジェクトをパネルの最前面に移動します。
最背面へ移動	選択しているオブジェクトをパネルの最背面へ移動します。
前面へ移動	選択しているオブジェクトを一つ前面へ移動します。
背面へ移動	選択しているオブジェクトを一つ背面へ移動します。
プロパティ	選択している図形／部品オブジェクトの設定ダイアログをオープンします。

[操作方法]

擬似的なターゲット・システムを構築するためのオブジェクト（図形オブジェクト／部品オブジェクト）の作成方法，およびそのシミュレーション方法は次のとおりです。

（1）編集モード
（2）図形オブジェクトの作成
（3）部品オブジェクトの作成
（4）オブジェクトの配置
（5）文字の入力
（6）オブジェクトの一覧表示
（7）オブジェクトの詳細設定
（8）入力シミュレーション・モード

(1) 編集モード

オブジェクトの作成を行うためには，このウインドウの動作モードを“編集モード”（デフォルト）に設定します。

編集モードの設定は，次のいずれかの方法により行います。

- ［図形］メニュー→［選択］を選択
- 図形ツールバーの ボタンをクリック
- ［編集］メニュー→［すべて選択］を選択

(2) 図形オブジェクトの作成

(a) 線の描画

［図形］メニュー→［線］を選択，またはツールバーの ボタンをクリックします。

→マウス・カーソルが十字（+）に変わり，線の描画が可能になります。

線の開始位置からドラッグし，終了位置でドロップします。

→線の開始位置と終了位置が直線で結ばれます（線の太さ，形状はデフォルトになります）。

(b) 四角形／丸四角形／楕円／扇型の描画

［図形］メニュー→［四角形］／［丸四角］／［楕円］／［扇型］を選択，またはツールバーの ／ ／ ／ ボタンをクリックします。

→マウス・カーソルが十字（+）に変わり，それぞれの描画が可能になります。

描画領域（長方形領域）の左上隅から右下隅の方向へドラッグします。

→マウス位置を右下隅とする描画領域に該当図形が表示されます。

ドロップすることで図形のサイズが確定されます。

→四角形は長方形領域と同じサイズに，その他の図形は長方形領域に納まるサイズで描画されます（線の太さ，形状はデフォルトになります）。

(c) 多角形の描画

［図形］メニュー→［多角形］を選択，またはツールバーの ボタンをクリックします。

→マウス・カーソルが十字（+）に変わり，多角形の描画が可能になります。

多角形の各頂点を描画したい位置でクリックします。

→クリックした順番に各頂点が直線で結ばれます。

ダブルクリックすることで多角形の描画が終了します。

→多角形の線の太さ，形状はデフォルトになります。

(d) ビットマップの貼り付け

図形オブジェクトとして，任意のビットマップを使用することができます。

［図形］メニュー→［ビットマップの貼り付け ...］を選択したのち，貼り付けたいビットマップ・ファイル（*.bmp）を選択します。

→このウインドウ上のデフォルト位置に該当ビットマップ・ファイルが貼り付けられます。

(e) 図形オブジェクトのスタイル変更

次のいずれかの方法により，作成した図形オブジェクトの色／線の種類などを変更することができます。

- 対象図形オブジェクトをダブルクリックすることによりオープンする Object Properties ダイアログの［スタイル］タブ上で操作
- 対象図形オブジェクトを選択したのち，［図形］メニュー→［線の色］／［塗りつぶしの色］／［線のスタイル］／［点線のスタイル］のいずれかを選択，またはツールバーの ／ ／ ／ ボタンをクリック

(3) 部品オブジェクトの作成

シミュレータ GUI が提供する接続部品を利用して，部品オブジェクトを作成することができます。

(a) 部品オブジェクトの選択

［部品］メニュー，またはツールバーから作成する部品オブジェクトを選択します。

→マウス・カーソルが十字（+）に変わります。

任意の位置をクリックします。

→クリック位置を左上隅の位置として，該当部品オブジェクトが作成／配置されます（デフォルト・サイズ）。

(b) 部品オブジェクトのスタイル変更

対象部品オブジェクトをダブルクリックすることによりオープンする設定ダイアログの［スタイル］タブにより，作成した部品オブジェクトのスタイルを変更することができます。

なお，変更可能な項目についての詳細は，各部品オブジェクトに対応する設定ダイアログの項を参照してください（対象となる部品オブジェクトに依存）。

(4) オブジェクトの配置

(a) グリッドの表示

［表示］メニュー→［グリッド］の選択により，このウインドウ上にグリッドが表示されます。

(b) オブジェクトの選択

次のいずれかの方法により，作成したオブジェクトが選択状態になります。

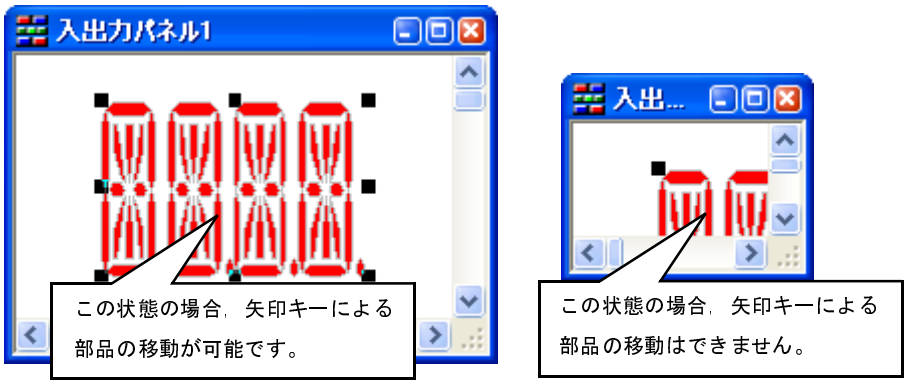
なお，選択状態となったオブジェクトは，周囲にトラッカーが表示されます。

- 個別選択
  選択したいオブジェクトをクリック
- 複数選択
  ［Shift］キーを押しながら選択したいオブジェクトをクリック
- 範囲選択
  選択したいオブジェクトを含む領域の左上隅からドラッグし，右下隅でドロップ
- すべてを選択
  ［編集］メニュー→［すべて選択］を選択

(c) オブジェクトの移動

対象オブジェクトを選択したのち（複数選択可），そのままドラッグし，移動先でドロップします。

備考　オブジェクトの移動は，矢印キーを使用することもできます。
ただし，ウインドウを縮小し選択部品を半分以上隠した状態では，矢印キーでの選択部品の移動はできません。



(d) オブジェクトのサイズ変更

対象オブジェクトを選択したのち，表示されるトラッカーをそのままドラッグします。

(e) オブジェクトの切り取り／コピー／貼り付け／削除／グループ化／グループ解除

対象オブジェクトを選択したのち，［編集］メニューから該当項目を選択することで行います。

(f) オブジェクトの順序変更（最前面へ移動／最背面へ移動／前面へ移動／背面へ移動）

対象オブジェクトを選択したのち，［編集］メニューから該当項目を選択することで行います。

(5) 文字の入力

［図形］メニュー→［文字］を選択，またはツールバーの A ボタンをクリックします。

→マウス・カーソルが十字（+）に変わります。

文字描画領域（長方形領域）の左上隅からドラッグし右下隅でドロップします。

→この長方形領域が文字描画領域となります。

文字描画領域内をクリックします。

→カーソルが表示され，文字入力が可能になります。

(6) オブジェクトの一覧表示

このウインドウ上で作成した図形オブジェクト，および部品オブジェクトは，このウインドウ上での表示以外に［表示］メニュー→［接続部品一覧 ...］の選択により，一覧表示することができます。

(7) オブジェクトの詳細設定

作成したオブジェクトには，使用するターゲット・システムに準じた詳細設定（端子接続情報など）が必要です。

(a) 図形オブジェクト

詳細設定は，図形オブジェクトをダブルクリックことによりオープンする Object Properties ダイアログの［端子接続］タブで行います。

オブジェクトと出力端子を接続することにより，接続端子の出力状態により図形オブジェクトの表示／非表示を切り替えることができます。

(b) 部品オブジェクト

詳細設定は，部品オブジェクトをダブルクリックことによりオープンする設定ダイアログの［*xxx* 端子接続］タブで行います。

設定可能な項目についての詳細は，各部品オブジェクトに対応する設定ダイアログの項を参照してください（対象となる部品オブジェクトに依存）。

(8) 入力シミュレーション・モード

詳細設定が完了した部品オブジェクトは，ユーザがシミュレーション中に操作することができるため（シミュレータに対して入力値を与えることができます），その入出力結果をこのウインドウ上で確認することができます。

部品オブジェクトの操作を行うためには，このウインドウの動作モードを入力シミュレーション・モードに設定します。

入力シミュレーション・モードの設定は，次のいずれかの方法により行います（マウス・カーソルが手の形になります）。

- ［図形］メニュー→［入力シミュレーション］を選択
- 図形ツールバーの ボタンをクリック

備考　入力操作についての詳細は，各部品オブジェクトに対応する設定ダイアログの項を参照してください。

Parts Button Properties ダイアログ

入出力パネル ウインドウの接続部品の一つであるボタンの端子接続情報の設定，変更を行います。

入力シミュレーション・モード時，端子と接続したボタンからシミュレータに対して入力操作が可能になります。

なお，ボタンの表示スタイルには，図形とビットマップの 2 種類があり，これらスタイルの変更は［[スタイル］タブ］で行います。

図 A—66　Parts Button Properties ダイアログ：［ボタン端子接続］タブ

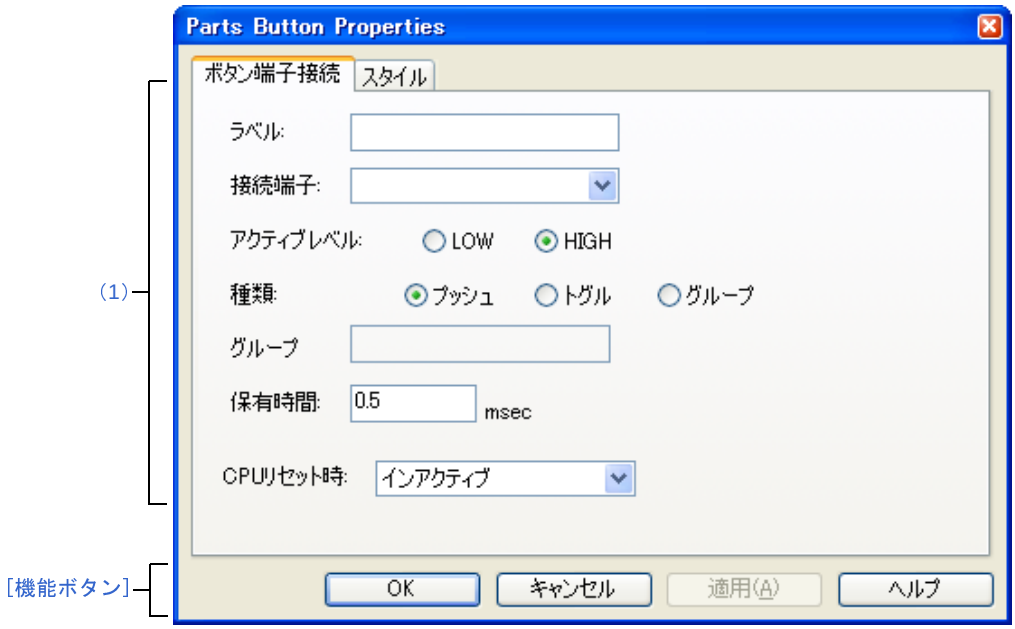
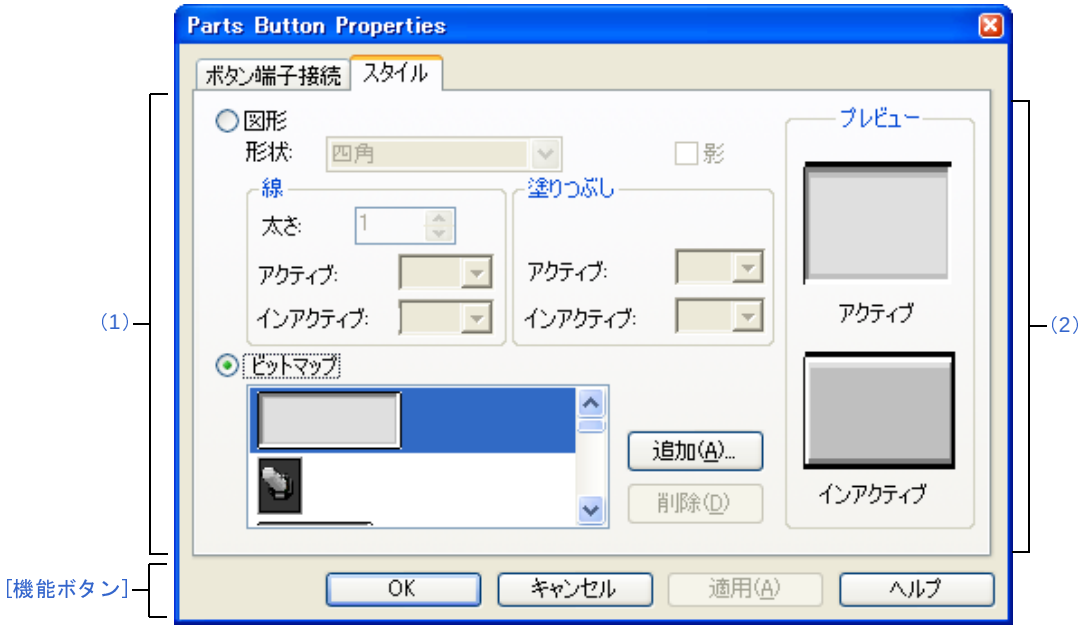


図 A—67　Parts Button Properties ダイアログ：［スタイル］タブ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［［ボタン端子接続］タブ］
- ［［スタイル］タブ］
- ［機能ボタン］
- ［表示例と操作方法］

［オープン方法］

入出力パネル ウインドウにおける次のいずれか

- 部品オブジェクト“ボタン”をダブルクリック
- 部品オブジェクト“ボタン”のコンテキスト・メニューより［プロパティ ...］を選択
- 部品オブジェクト“ボタン”を選択したのち，［表示］メニュー→［プロパティ ...］を選択

［［ボタン端子接続］タブ］

（1）ボタン端子接続設定エリア

ラベル	部品に名前を付ける際に指定します。 ここで指定した名前はボタン上に表示されます。また，部品一覧 ダイアログ上でラベルとして表示されます。	
端子接続	接続する端子名を指定するエリアです。 指定は，ドロップダウン・リストからの選択，または直接入力により行います。	
アクティブレベル	アクティブ状態をオプション・ボタンにより選択します。	
	LOW	アクティブ・レベルを LOW に設定します。
	HIGH	アクティブ・レベルを HIGH に設定します（デフォルト）。
種類	ボタンの種類をオプション・ボタンにより選択します。	
	プッシュ	プッシュ・ボタンにします（デフォルト）。 ［保有時間］での指定が必要となります。
	トグル	トグル・ボタンにします。
	グループ	グループ・ボタンにします。 ［グループ］での指定が必要となります。
グループ	ボタンのグループ名を入力します。 このエリアは，［種類］で［グループ］を選択した時のみ有効です。	
保有時間	入力した値を保持させる時間（保有時間）を指定します（デフォルト：0.5 ミリ秒）。 指定可能範囲は 0.001 ～ 999 ミリ秒です。 このエリアは，［種類］で［プッシュ］を選択した時のみ有効です。	
CPU リセット時	CPU リセット時のボタンの状態を指定します。	
	CPU リセット前を維持	CPU リセット時，ボタンの状態を維持します。
	インアクティブ	CPU リセット時，ボタンが押されていない状態にします（デフォルト）。
	アクティブ	CPU リセット時，ボタンが押された状態にします。

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

[[スタイル] タブ]

(1) スタイル情報設定エリア

図形	ボタンを図形で表示する場合，このオプション・ボタンを選択します。		
	形状	図形の形状（四角，楕円のいずれか）を選択します。	
	影	選択不可	
	線	図形の線に関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		太さ	線の太さを指定します。 スピン・ボタンでの選択，または直接入力により行います。 1 ～ 100 までの範囲での指定が可能です。
		アクティブ	アクティブ表示時の線の色を指定します。
		インアクティブ	インアクティブ表示時の線の色を指定します。
	塗りつぶし	図形の塗りつぶしに関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		アクティブ	アクティブ表示時の塗りつぶしの色を指定
		インアクティブ	インアクティブ表示時の塗りつぶしの色を指定
ビットマップ	ボタンを指定したビットマップで表示する場合，このオプション・ボタンを選択します（デフォルト）。		
	選択リスト	使用するビットマップを選択します（リストには，現在選択可能なビットマップが表示されます）。	
	[追加] ボタン	次のビットマップの追加 ダイアログがオープンし，選択リストに新規にビットマップを追加します。[...] ボタンによるファイル選択，または直接入力によりファイルを指定します。 	
	[削除] ボタン	現在選択リストで選択しているビットマップを削除します。 ただし，ユーザにより追加されたビットマップのみ削除可能です。	

(2) プレビュー・エリア

現在設定しているボタンのスタイルを表示します。

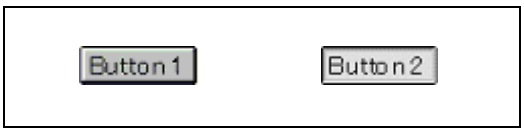
[機能ボタン]

ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。
適用	選択不可
ヘルプ	このダイアログのヘルプを表示します。

[表示例と操作方法]

入力シミュレーション・モード時，ボタンの押下により，接続した端子へのデータ入力が可能になります。
なお，ボタンの種類により入力形態が異なります。

図 A—68　接続部品表示例（ボタン）



プッシュ・ボタン	ボタンの押下により，接続した端子にはアクティブ値が取り込まれます。 アクティブ値は，保有時間の間，保持され，保有時間を過ぎると元に戻ります。
トグル・ボタン	ボタンの押下により，接続した端子にはアクティブ値が取り込まれます。 アクティブ値は，再度同一のボタンが押されるまでの間，保持されます。
グループ・ボタン	ボタンの押下により，接続した端子にはアクティブ値が取り込まれます。 同じグループ名を持つグループ・ボタンの値は元に戻ります。

Analog Button Properties ダイアログ

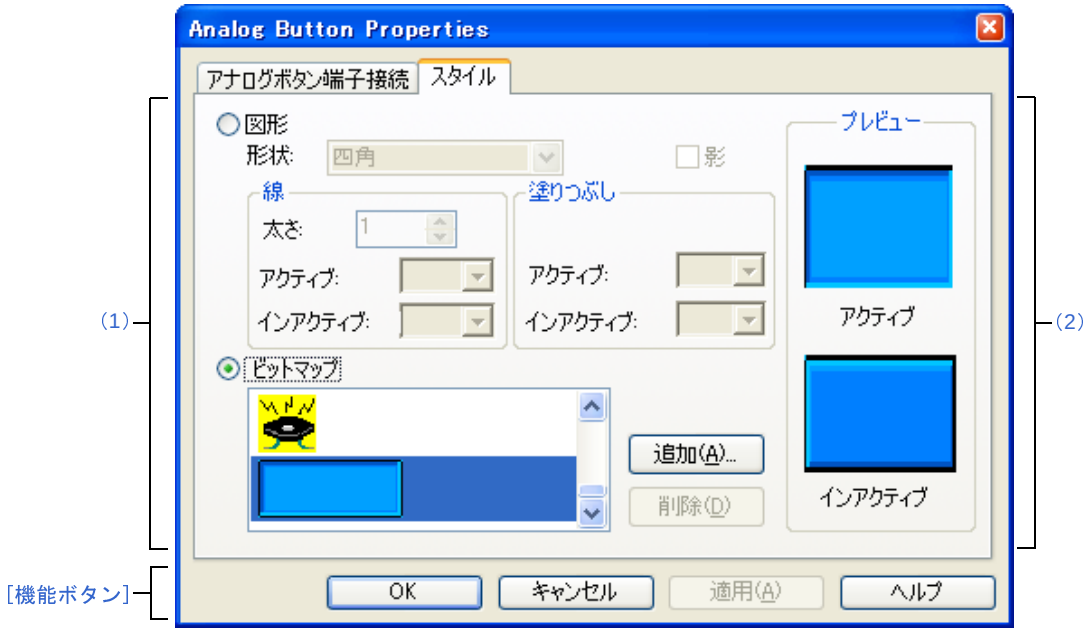
入出力パネル ウインドウの接続部品の一つであるアナログ・ボタンの端子接続情報の設定，変更を行います。

入力シミュレーション・モード時，端子と接続したアナログ・ボタンからは，シミュレータに対して入力操作が可能になります。

なお，アナログ・ボタンの表示スタイルには，図形とビットマップの 2 種類があり，これらスタイルの変更は［［スタイル］タブ］で行います。

図 A—69　Analog Button Properties ダイアログ：［アナログボタン端子接続］タブ

図 A—70　Analog Button Properties ダイアログ：［スタイル］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］
- ［［アナログボタン端子接続］タブ］
- ［［スタイル］タブ］
- ［機能ボタン］
- ［表示例と操作方法］

［オープン方法］

入出力パネル ウインドウにおける次のいずれか

- 部品オブジェクト“アナログ・ボタン”をダブルクリック
- 部品オブジェクト“アナログ・ボタン”のコンテキスト・メニューより［プロパティ ...］を選択
- 部品オブジェクト“アナログ・ボタン”を選択したのち，［表示］メニュー→［プロパティ ...］を選択

［［アナログボタン端子接続］タブ］

(1) アナログ・ボタン端子接続設定エリア

ラベル	部品に名前を付ける際に指定します。 ここで指定した名前はボタン上に表示されます。また，部品一覧 ダイアログ上でラベルとして表示されます。	
端子接続	接続する端子名を指定するエリアです。 指定は，ドロップダウン・リストからの選択，または直接入力により行います。	
アクティブ	アクティブ状態を指定します。	
	チェック・ボックス	チェックした数だけアナログ・ボタンが作成されます。
	ラベル	各アナログ・ボタンに表示する名前を直接入力します。
	電圧（mV）	各アナログ・ボタンをクリックした際，入力される電圧（単位：mV）を直接入力します。
配置	ボタンの並びをオプション・ボタンで指定します。 上記アクティブ・エリアで作成したアナログ・ボタン数が 2 個以上の場合，この設定が有効になります。アナログ・ボタン数が 1 個以下の場合，この設定は無視されます。	
	横列	アナログ・ボタンを横並びに配置します（デフォルト）。
	縦列	アナログ・ボタンを縦並びに配置します。
CPU リセット時	CPU リセット時のアナログ・ボタンの状態を指定します。	
	CPU リセット前を維持	CPU リセット時，ボタンの状態を維持します。
	インアクティブ	CPU リセット時，ボタンが押されていない状態にします（デフォルト）。
	'xxx'yyy(mV) をアクティブ	“*'xxx'yyy*(mV)”で指定されたアナログ・ボタンが CPU リセット後に押された状態になります。
インアクティブ	すべてのアナログ・ボタンが押されていない場合の入力レベルを指定します（単位：mV）。	

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

[[スタイル] タブ]

(1) スタイル情報設定エリア

図形	アナログ・ボタンを図形で表示する場合，このオプション・ボタンを選択します。		
	形状	図形の形状（四角，楕円のいずれか）を選択します。	
	影	選択不可	
	線	図形の線に関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		太さ	線の太さを指定します。 スピン・ボタンでの選択，または直接入力により行います。 1 ～ 100 までの範囲での指定が可能です。
		アクティブ	アクティブ表示時の線の色を指定します。
		インアクティブ	インアクティブ表示時の線の色を指定します。
	塗りつぶし	図形の塗りつぶしに関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		アクティブ	アクティブ表示時の塗りつぶしの色を指定します。
		インアクティブ	インアクティブ表示時の塗りつぶしの色を指定します。
ビットマップ	アナログ・ボタンを指定したビットマップで表示する場合，このオプション・ボタンを選択します（デフォルト）。		
	選択リスト	使用するビットマップを選択します（リストには，現在選択可能なビットマップが表示されます）。	
	[追加] ボタン	次のビットマップの追加 ダイアログがオープンし，選択リストに新規にビットマップを追加します。[...] ボタンによるファイル選択，または直接入力によりファイルを指定します。 	
	[削除] ボタン	現在選択リストで選択しているビットマップを削除します。 ただし，ユーザにより追加されたビットマップのみ削除可能です。	

(2) プレビュー・エリア

現在設定しているアナログ・ボタンのスタイルを表示します。

[機能ボタン]

ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。
適用	選択不可
ヘルプ	このダイアログのヘルプを表示します。

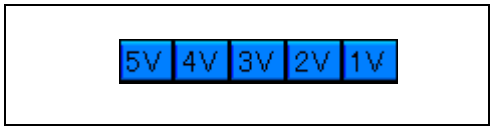
[表示例と操作方法]

入力シミュレーション・モード時，アナログ・ボタンを押下することにより，接続した端子へのアナログ電圧値入力が可能になります。

なお，一度に押すことができるボタンは１つのみです。

アナログ・ボタンが押されている間，設定した端子へ設定したアナログ電圧値が入力されます。押下状態のアナログ・ボタンは同じボタンを再度押すことで元の状態に戻ります。

図 A—71　接続部品表示例（アナログ・ボタン）



Parts Key Properties ダイアログ

入出力パネル ウインドウの接続部品の一つであるキー・マトリクスの端子接続情報の設定，変更を行います。

入力シミュレーション・モード時，端子と接続したキーからは，シミュレータに対して入力操作が可能になります。

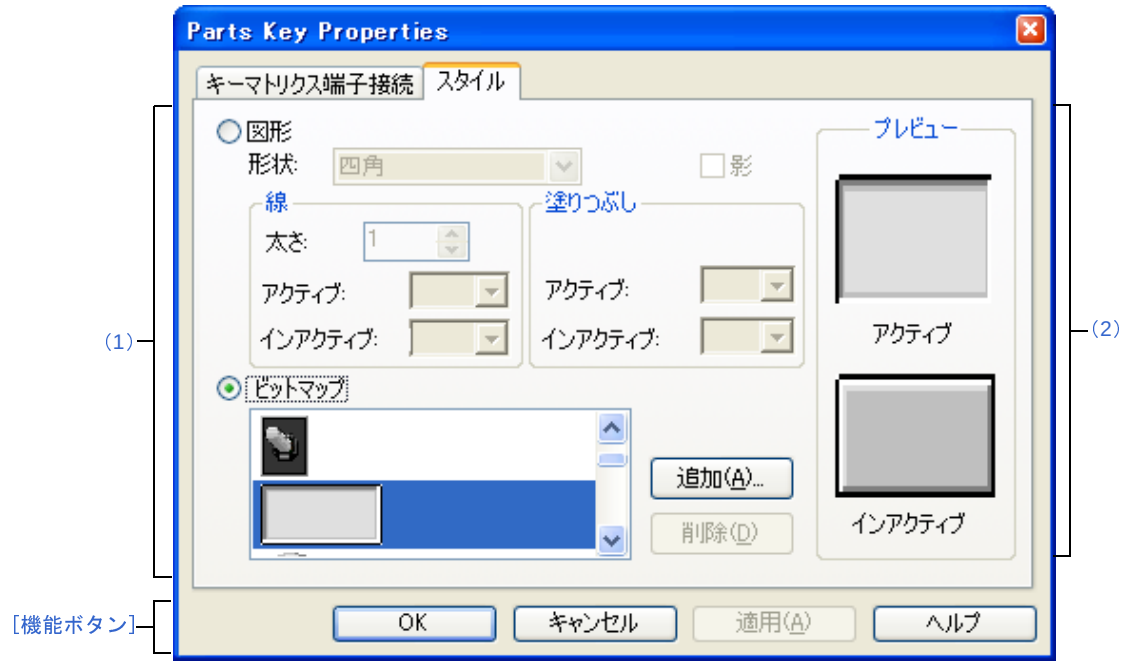
入力端子と出力端子によるキー・マトリクスは，最大 16 × 16 まで設定可能です。

なお，キー・マトリクスの表示スタイルには，図形とビットマップの 2 種類があり，これらスタイルの変更は［［スタイル］タブ］で行います。

注意　キー・マトリクスを端子に接続する際は，接続端子のプルアップ／プルダウン設定も合わせて行う必要があります。キーの押下時，キーと接続された入力端子には，該当キーに接続された出力端子の出力値が入力されます。なお，キーの非押下時の値は，プルアップ／プルダウン設定 ダイアログで指定した値になります。
プルアップ／プルダウンの設定を行わない場合，入力端子はハイ・インピーダンス状態になります。このため，入力端子に接続された機能の動作は不定となります。

図 A—72　Parts Key Properties ダイアログ：［キーマトリクス端子接続］タブ

図 A—73　Parts Key Properties ダイアログ：［スタイル］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］
- ［［キーマトリクス端子接続］タブ］
- ［［スタイル］タブ］
- ［機能ボタン］
- ［表示例と操作方法］

［オープン方法］

入出力パネル ウインドウにおける次のいずれか

- 部品オブジェクト“キー・マトリクス”をダブルクリック
- 部品オブジェクト“キー・マトリクス”のコンテキスト・メニューより［プロパティ ...］を選択
- 部品オブジェクト“キー・マトリクス”を選択したのち，［表示］メニュー→［プロパティ ...］を選択

［［キーマトリクス端子接続］タブ］

（1）端子接続情報設定エリア

ラベル	部品に名前を付ける際に指定します。 ここで指定した名前は部品一覧 ダイアログ上でラベルとして表示されます。

接続端子	接続する端子名（入力端子，出力端子）を指定するエリアです。 指定は，ドロップダウン・リストからの選択，または直接入力により行います。 このエリアは，スクロール・バーにより 16 × 16 端子の設定が可能です。	
	In0 ～ In15	入力端子を指定します。
	Out0 ～ Out15	出力端子を指定します。
	N00 ～ Nff	キー・マトリクス上に表示する文字列を直接入力により指定します。 任意の長さの文字列が指定可能です。 デフォルトの記述文字列（N 数字）は，キー上には表示されません。
保有時間	入力した値を保持させる時間（保有時間）を指定します（デフォルト：0.5 ミリ秒）。 指定可能範囲は 0.001 ～ 999 ミリ秒です。 なお，保有時間内に同じ入力端子に入力されるキーを複数個，押下した場合には，最後にクリックしたキーが有効になります。	
CPU リセット時	CPU リセット時のキー・マトリクスの動作を指定します。	
	CPU リセット前を維持	CPU リセット時，キー・マトリクスの状態が変化しません。
	インアクティブ	CPU リセット時，キー・マトリクスがすべて押されていない状態になります（デフォルト）。

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

[[スタイル] タブ]

(1) スタイル情報設定エリア

図形	キー・マトリクスを図形で表示する場合，このオプション・ボタンを選択します。		
	形状	図形の形状（四角，楕円のいずれか）を選択します。	
	影	選択不可	
	線	図形の線に関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		太さ	線の太さを指定します。 スピン・ボタンでの選択，または直接入力により行います。 1 ～ 100 までの範囲での指定が可能です。
		アクティブ	アクティブ表示時の線の色を指定します。
		インアクティブ	インアクティブ表示時の線の色を指定します。
	塗りつぶし	図形の塗りつぶしに関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		アクティブ	アクティブ表示時の塗りつぶしの色を指定します。
		インアクティブ	インアクティブ表示時の塗りつぶしの色を指定します。

ビットマップ	キー・マトリクスを指定したビットマップで表示する場合，このオプション・ボタンを選択します（デフォルト）。	
	選択リスト	使用するビットマップを選択します（リストには，現在選択可能なビットマップが表示されます）。
	［追加］ボタン	次のビットマップの追加 ダイアログがオープンし，選択リストに新規にビットマップを追加します。[...] ボタンによるファイル選択，または直接入力によりファイルを指定します。
	［削除］ボタン	現在選択リストで選択しているビットマップを削除します。 ただし，ユーザにより追加されれたビットマップのみ削除可能です。

(2) プレビュー・エリア

現在設定しているキー・マトリクスのスタイルを表示します。

[機能ボタン]

ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。
適用	選択不可
ヘルプ	このダイアログのヘルプを表示します。

[表示例と操作方法]

入力シミュレーション・モード時，次の操作を行うことができます。

(1) 同時に複数個のキーを操作する
(2) キーの入力値をロックする

(1) 同時に複数個のキーを操作する

同時に入力したいキーの片方をマウスの右ボタンでクリックし，待ち状態にします。続いて残りのキーをクリックすることにより，先の待ち状態が解除され両方のキーを同時に入力することができます。複数のキーを待ち状態にすることにより，複数のキーの同時入力が可能になります。

ただし，同じ入力端子への入力になる場合には，後から入力したキーが有効となります。

(2) キーの入力値をロックする

任意のキーに対してマウスの右ボタンを押しながら，マウスの左ボタンをクリックすることにより，その時のキーの入力値がロックされます。ロック状態中に，ロックされたキーと同じ入力端子への入力になるキーがクリックされた場合には，後から入力したキーの入力値が有効になりますが，そのキーの保有時間が経過すると再度ロック状態時の入力値になります。

ロック状態のキーをクリックすることによりロックが解除され，右ボタンでクリックすることにより待ち状態になります。

図 A—74　接続部品表示例（キー・マトリクス）

Parts Level Gauge Properties ダイアログ

入出力パネル ウインドウの接続部品の一つであるレベル・ゲージの端子接続情報の設定，変更を行います。

入力シミュレーション・モード時，端子と接続したレベル・ゲージからは，シミュレータに対して入力操作が可能になります。ただし，接続する端子はアナログ入力用端子に限ります。

なお，レベル・ゲージの表示スタイルには，スライド式とダイアル式の 2 種類があり，これらスタイルの変更は［［スタイル］タブ］で行います。

図 A—75　Parts Level Gauge Properties ダイアログ：［レベルゲージ端子接続］タブ

図 A—76　Parts Level Gauge Properties ダイアログ：［スタイル］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］
- ［［レベルゲージ端子接続］タブ］
- ［［スタイル］タブ］
- ［機能ボタン］
- ［表示例と操作方法］

［オープン方法］

入出力パネル ウインドウにおける次のいずれか

- 部品オブジェクト“レベル・ゲージ”をダブルクリック
- 部品オブジェクト“レベル・ゲージ”のコンテキスト・メニューより［プロパティ ...］を選択
- 部品オブジェクト“レベル・ゲージ”を選択したのち，［表示］メニュー→［プロパティ ...］を選択

［［レベルゲージ端子接続］タブ］

（1）端子接続情報設定エリア

ラベル	部品に名前を付ける際に指定します。 ここで指定した名前は部品一覧 ダイアログ上でラベルとして表示されます。	
接続端子	接続する端子名を指定するエリアです。 指定は，ドロップダウン・リストからの選択，または直接入力により行います。	
最大入力値	レベル・ゲージ入力の最大値を指定します（デフォルト：5000 mV）。 指定は mV 単位で行ってください。 指定可能範囲は 0 ～ 65535 です。 この指定値により，入出力パネル ウインドウに表示されたレベル・ゲージの動作範囲が決定します。	
CPU リセット時	CPU リセット時のレベル・ゲージの動作を指定します。	
	CPU リセット前を維持	CPU リセット直前の状態を CPU リセット後も維持します。
	初期電圧を指定	CPU リセット時，レベル・ゲージが指定した値に設定されます（デフォルト）。 指定は mV 単位で行ってください。 指定可能範囲は 0 ～最大入力値で指定した値です（デフォルト：0 mV）。

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

[[スタイル] タブ]

(1) スタイル情報設定エリア

スライド式	レベル・ゲージをスライド式で表示する場合，このオプション・ボタンを選択します。	
	方向	スライド方向（垂直，水平）をドロップダウン・リストから選択します。
	色	スライドの色を指定，変更します。 プルダウン・ボタンをクリックすることにより色の指定が可能です。
ダイアル式	レベル・ゲージをダイアル式で表示する場合このオプション・ボタンを選択します（デフォルト）。	
	マークの色	動作点を示すマークの色を指定，変更します。 プルダウン・ボタンをクリックすることにより色の指定が可能です。
	選択リスト	使用するビットマップを選択します（リストには，現在選択可能なビットマップが表示されます）。
	[追加] ボタン	次のビットマップの追加 ダイアログがオープンし，選択リストに新規にビットマップを追加します。[...] ボタンによるファイル選択，または直接入力によりファイルを指定します。
	[削除] ボタン	現在選択リストで選択しているビットマップを削除します。 ただし，ユーザにより追加されたビットマップのみ削除可能です。

(2) プレビュー

現在設定しているレベル・ゲージのスタイルを表示します。

[機能ボタン]

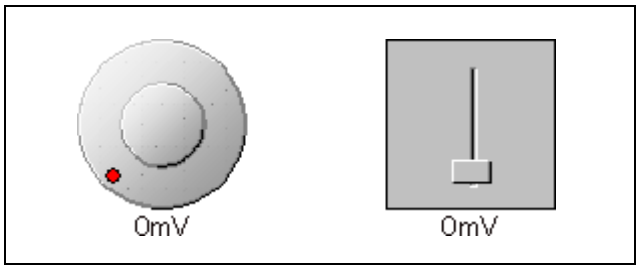
ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。
適用	選択不可
ヘルプ	このダイアログのヘルプを表示します。

[表示例と操作方法]

入力シミュレーション・モード時，表示されたスライダ，またはダイアルを操作することにより，レベル・ゲージからのアナログ入力を行います。

［最大入力値］で指定した値が入力できる最大値になります。

図 A—77　接続部品表示例（レベル・ゲージ）



ダイアル式レベル・ゲージ	設ダイアル上の動作点（赤い丸）をドラッグすることにより，表示しているアナログ値が変化します。このアナログ値が入力したい値になった時，動作点からマウスを離します。これにより，表示しているアナログ値を入力が可能になります。なお，動作点の移動は任意の位置をクリックすることによっても可能です。
スライド式レベル・ゲージ	スライダのつまみをドラッグし動かすことにより表示しているアナログ値が変化します。このアナログ値が入力したい値になった時，つまみからマウスを離します。これにより，表示しているアナログ値の入力が可能になります。なお，つまみの移動は任意の位置をクリックすることによっても可能です。

注意　ダイアル上の動作点（赤い丸），またはスライダのつまみをドラッグしたのち，レベル・ゲージから離れた場所でドロップすると，レベル・ゲージに表示される電圧は変更されますが実際にレベル・ゲージから出力される電圧は変更されません。ドラッグ・アンド・ドロップは，必ずレベル・ゲージの上で行ってください。

Parts Led Properties ダイアログ

入出力パネル ウインドウの接続部品の一つである LED の端子接続情報の設定，変更を行います。

入力シミュレーション・モード時，端子と接続した LED は，シミュレータからの出力情報を点灯／消灯で表示します。

なお，LED の表示スタイルには，図形とビットマップの 2 種類があり，これらスタイルの変更は［[スタイル］タブ］で行います。

図 A—78　Parts Led Properties ダイアログ：［LED 端子接続］タブ

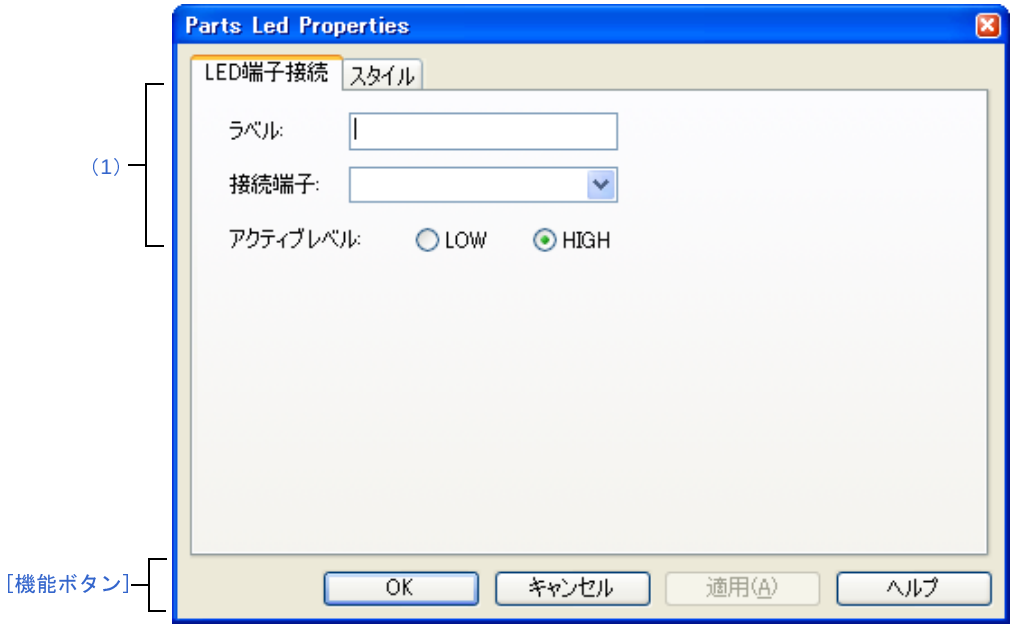


図 A—79　Parts Led Properties ダイアログ：［スタイル］タブ

ここでは，次の項目について説明します。

- ［オープン方法］
- ［［LED 端子接続］タブ］
- ［［スタイル］タブ］
- ［機能ボタン］
- ［表示例と操作方法］

［オープン方法］

入出力パネル ウインドウにおける次のいずれか

- 部品オブジェクト“LED”をダブルクリック
- 部品オブジェクト“LED”のコンテキスト・メニューより［プロパティ ...］を選択
- 部品オブジェクト“LED”を選択したのち，［表示］メニュー→［プロパティ ...］を選択

［［LED 端子接続］タブ］

（1）端子接続情報設定エリア

ラベル	部品に名前を付ける際に指定します。 ここで指定した名前は部品一覧 ダイアログ上でラベルとして表示されます。	
接続端子	接接続する端子名（出力端子）を指定するエリアです。 指定は，ドロップダウン・リストからの選択，または直接入力により行います。	
アクティブレベル	アクティブ状態をオプション・ボタンにより選択します。	
	LOW	アクティブ・レベルを LOW に設定します。
	HIGH	アクティブ・レベルを HIGH に設定します（デフォルト）。

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

[[スタイル] タブ]

(1) スタイル情報設定エリア

図形	LED を図形で表示する場合，このオプション・ボタンを選択します。		
	形状	図形の形状（四角，楕円のいずれか）を選択します。	
	影	選択不可	
	線	図形の線に関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		太さ	線の太さを指定します。 スピン・ボタンでの選択，または直接入力により行います。 1 ～ 100 までの範囲での指定が可能です。
		アクティブ	アクティブ表示時の線の色を指定します。
		インアクティブ	インアクティブ表示時の線の色を指定します。
	塗りつぶし	図形の塗りつぶしに関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		アクティブ	アクティブ表示時の塗りつぶしの色を指定します。
		インアクティブ	インアクティブ表示時の塗りつぶしの色を指定します。
ビットマップ	LED を指定したビットマップで表示する場合，このオプション・ボタンを選択します（デフォルト）。		
	選択リスト	使用するビットマップを選択します（リストには，現在選択可能なビットマップが表示されます）。	
	[追加] ボタン	次のビットマップの追加 ダイアログがオープンし，選択リストに新規にビットマップを追加します。[...] ボタンによるファイル選択，または直接入力によりファイルを指定します。 	
	[削除] ボタン	現在選択リストで選択しているビットマップを削除します。 ただし，ユーザにより追加されれたビットマップのみ削除可能です。	

(2) プレビュー・エリア

現在設定している LED のスタイルを表示します。

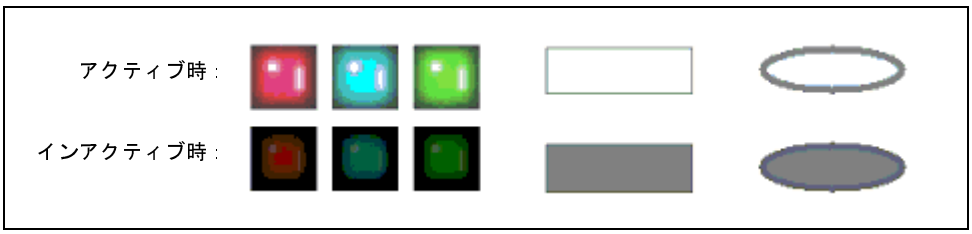
[機能ボタン]

ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。
適用	選択不可
ヘルプ	このダイアログのヘルプを表示します。

[表示例と操作方法]

入力シミュレーション・モード時，接続した端子の出力状態（アクティブ／インアクティブ）を 2 種類のビットマップ，または図形でリアルタイムに表示します。

図 A—80　接続部品表示例（LED）



Parts Segment LED Properties ダイアログ

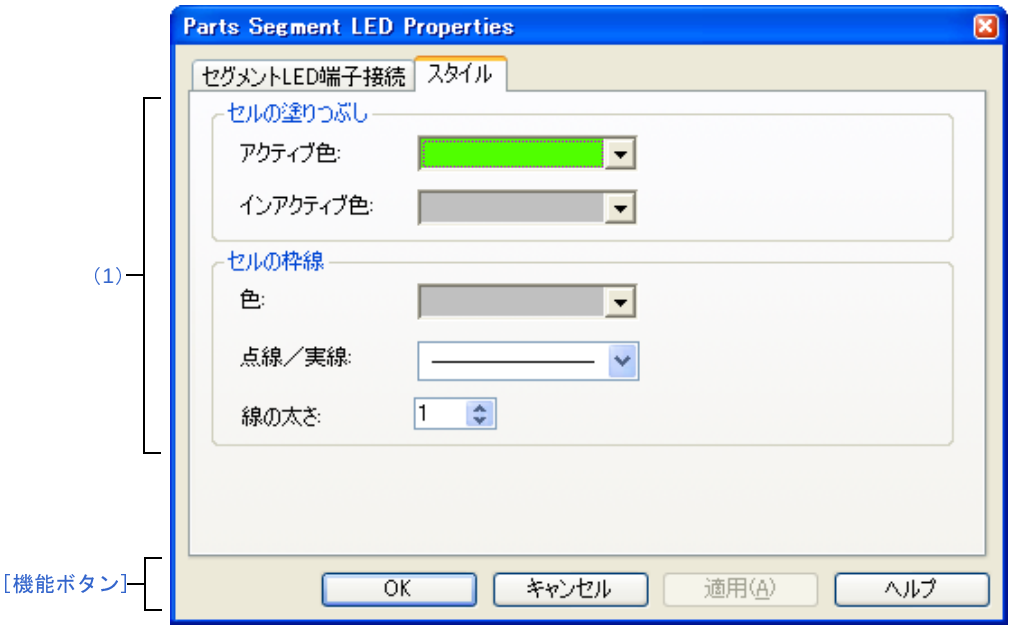
入出力パネル ウインドウの接続部品の一つである 7 セグメント LED，および 14 セグメント LED の端子接続情報の設定，変更を行います。

入力シミュレーション・モード時，端子と接続した各 LED は，シミュレータからの出力情報を表示します。

セグメント LED の表示スタイルの変更は［[スタイル］タブ］で行います。

図 A—81　Parts Segment LED Properties ダイアログ：［セグメント LED 端子接続］タブ

図 A—82　Parts Segment LED Properties ダイアログ：［スタイル］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］
- ［［セグメント LED 端子接続］タブ］
- ［［スタイル］タブ］
- ［機能ボタン］
- ［表示例と操作方法］

［オープン方法］

入出力パネル ウインドウにおける次のいずれか

- 部品オブジェクト“7 セグメント LED”／“14 セグメント LED”をダブルクリック
- 部品オブジェクト“7 セグメント LED”／“14 セグメント LED”のコンテキスト・メニューより［プロパティ ...］を選択
- 部品オブジェクト“7 セグメント LED”／“14 セグメント LED”を選択したのち，［表示］メニュー→［プロパティ ...］を選択

［［セグメント LED 端子接続］タブ］

（1）端子接続情報設定エリア

ラベル	部品に名前を付ける際に指定します。 ここで指定した名前は部品一覧 ダイアログ上でラベルとして表示されます。	
グリッドの割り当て	グリッド信号の割り当て方法を次の中から選択します。 この選択により，桁信号設定エリアでの設定が変化します。	
	桁	セグメント LED の 1 桁を 1 つのグリッド端子と接続します（デフォルト）。 桁信号設定エリアでは，桁信号の設定を行います。グリッド信号は 16 桁の指定が可能で，1 つのセグメント LED 部品で最大 16 桁のセグメント LED を作成可能です。
	カスタマイズ	選択不可

(2) セグメント信号設定エリア

セグメント信号	7 セグメント LED/14 セグメント LED のセグメント信号と接続する端子（出力端子），およびアクティブ・レベルを指定するエリアです。		
	図示	左上には 7 セグメント LED/14 セグメント LED のビットマップを表示します。 ［接続端子］を入力する際，対応する位置を示します。	
	接続端子	接続する端子をドロップダウン・リストからの選択，または直接入力により指定します。 接続するセグメント端子数は，7 セグメント LED では 8 個，14 セグメント LED では 15 個となります。 なお，右側にあるスクロール・バーを操作することにより，すべてのセグメント端子に接続可能です。	
	アクティブレベル	アクティブ状態をオプション・ボタンにより選択します。	
		LOW	アクティブ・レベルを LOW に設定します。
		HIGH	アクティブ・レベルを HIGH に設定します（デフォルト）。

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

(3) 桁信号設定エリア

桁信号	7 セグメント LED/14 セグメント LED の桁，またはグリッド信号と接続する端子（出力端子），およびアクティブ・レベルを指定するエリアです。 ［グリッドの割り当て］での指定により，接続の仕方が次のように変化します。 - ［桁］選択時 桁信号の設定を行います。接続する桁端子の数は最大 16 です。 エリア右側にあるスクロール・バーを操作することにより，すべての桁端子に接続可能です。 - ［カスタマイズ］選択時 選択不可		
	接続端子	接続する端子名をドロップダウン・リストからの選択，または直接入力により指定します。 なお，設定信号は，最下位桁からの連続端子を指定してください。	
	アクティブレベル	アクティブ状態をオプション・ボタンにより選択します	
		LOW	アクティブ・レベルを LOW に設定します。
		HIGH	アクティブ・レベルを HIGH に設定します（デフォルト）。

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

[[スタイル] タブ]

(1) スタイル情報設定エリア

セルの塗りつぶし	各セルの塗りつぶしに関する設定，変更を行うエリアです。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
	アクティブ色	アクティブ表示時の塗りつぶしの色を指定します。
	インアクティブ色	インアクティブ表示時の塗りつぶしの色を指定します。
セルの枠線	セルの枠線の形状に関する設定，変更を行うエリアです。	
	色	線の色を指定，変更します。 プルダウン・ボタンをクリックすることにより色の指定が可能です。
	点線／実線	線の形状（点線／実線）を指定，変更します。 ドロップダウン・リストからの選択により行います。 [線の太さ] での指定が“1”の時のみ指定可能です。
	線の太さ	線の太さを指定，変更します。 スピン・ボタンでの選択，または直接入力により行います。 0.1 ～ 100 までの範囲での指定が可能です。

[機能ボタン]

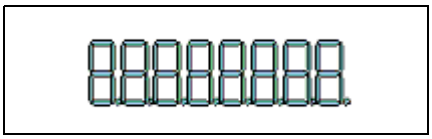
ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。
適用	選択不可
ヘルプ	このダイアログのヘルプを表示します。

[表示例と操作方法]

入力シミュレーション・モード時，1 シミュレーションの結果，接続端子の出力情報を受け取りその値にしたがって表示します。

桁／グリッド信号，およびセグメント信号ともにアクティブ出力の際，対応する桁／グリッドのセグメント LED が点灯します。

図 A—83　接接続部品表示例（7 セグメント LED）



Parts Matrix Led Properties ダイアログ

入出力パネル ウインドウの接続部品の一つであるマトリクス LED の端子接続情報の設定，変更を行います。

入力シミュレーション・モード時，端子と接続したマトリクス LED は，シミュレータからの出力情報を点灯／消灯で表示します。

なお，マトリクス LED の表示スタイルには，図形とビットマップの 2 種類があり，これらスタイルの変更は［[スタイル］タブ］で行います。

図 A—84　Parts Matrix Led Properties ダイアログ：［マトリクス LED 端子接続］タブ

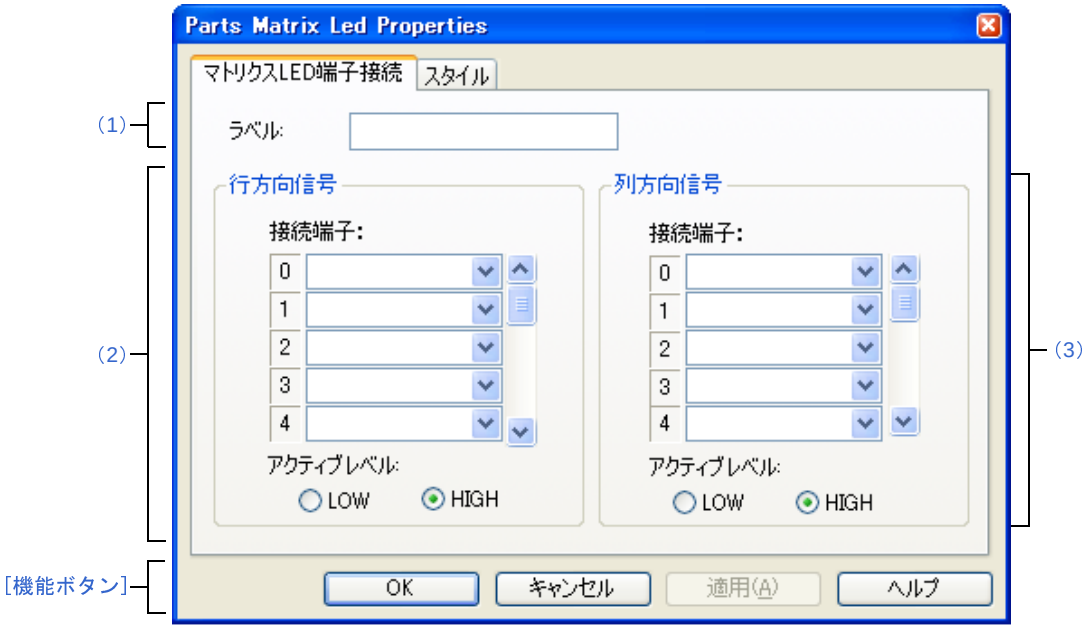


図 A—85　Parts Matrix Led Properties ダイアログ：［スタイル］タブ

ここでは，次の項目について説明します。

- ［オープン方法］
- ［［マトリクス LED 端子接続］タブ］
- ［［スタイル］タブ］
- ［機能ボタン］
- ［表示例と操作方法］

［オープン方法］

入出力パネル ウインドウにおける次のいずれか

- 部品オブジェクト“マトリクス LED”をダブルクリック
- 部品オブジェクト“マトリクス LED”のコンテキスト・メニューより［プロパティ ...］を選択
- 部品オブジェクト“マトリクス LED”を選択したのち，［表示］メニュー→［プロパティ ...］を選択

［［マトリクス LED 端子接続］タブ］

（1）ラベル設定エリア

ラベル	部品に名前を付ける際に指定します。 ここで指定した名前は部品一覧 ダイアログ上でラベルとして表示されます。

（2）行方向信号設定エリア

行方向信号	マトリクス LED の行方向の信号と接続する端子（出力端子），およびアクティブ・レベルを指定するエリアです。		
	接続端子	接続する端子名をドロップダウン・リストからの選択，または直接入力により指定します。 接続する端子数は最大 16 です。右側にあるスクロール・バーを操作することにより，すべての行方向信号に接続可能です。	
	アクティブレベル	アクティブ状態をオプション・ボタンオプション・ボタンにより選択します。	
		LOW	アクティブ・レベルを LOW に設定します。
		HIGH	アクティブ・レベルを HIGH に設定します（デフォルト）。

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

(3) 列方向信号設定エリア

列方向信号	マトリクス LED の列方向信号と接続する端子（出力端子），およびアクティブ・レベルを指定するエリアです。		
	接続端子	接続する端子名をドロップダウン・リストからの選択，または直接入力により指定します。接続する端子数は最大 16 です。右側にあるスクロール・バーを操作することにより，すべての列方向信号に接続可能です。	
	アクティブレベル	アクティブ状態をオプション・ボタンにより選択します。	
		LOW	アクティブ・レベルを LOW に設定します。
		HIGH	アクティブ・レベルを HIGH に設定します（デフォルト）。

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

[[スタイル] タブ]

(1) スタイル情報設定エリア

図形	マトリクス LED を図形で表示する場合，このオプション・ボタンを選択します。		
	形状	図形の形状（四角，楕円のいずれか）を選択します。	
	影	選択不可	
	線	図形の線に関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		太さ	線の太さを指定します。 スピン・ボタンでの選択，または直接入力により行います。 1 ～ 100 までの範囲での指定が可能です。
		アクティブ	アクティブ表示時の線の色を指定します。
		インアクティブ	インアクティブ表示時の線の色を指定します。
	塗りつぶし	図形の塗りつぶしに関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		アクティブ	アクティブ表示時の塗りつぶしの色を指定します。
		インアクティブ	インアクティブ表示時の塗りつぶしの色を指定します。

ビットマップ	マトリクス LED を指定したビットマップで表示する場合，このオプション・ボタンを選択します（デフォルト）。	
	選択リスト	使用するビットマップを選択します（リストには，現在選択可能なビットマップが表示されます）。
	［追加］ボタン	次のビットマップの追加 ダイアログがオープンし，選択リストに新規にビットマップを追加します。[...] ボタンによるファイル選択，または直接入力によりファイルを指定します。
	［削除］ボタン	現在選択リストで選択しているビットマップを削除します。 ただし，ユーザにより追加されれたビットマップのみ削除可能です。

(2) プレビュー・エリア

現在設定しているマトリクス LED のスタイルを表示します。

[機能ボタン]

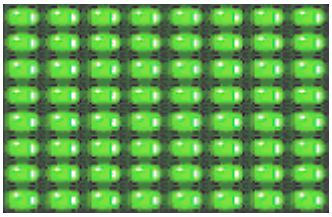
ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。
適用	選択不可
ヘルプ	このダイアログのヘルプを表示します。

[表示例と操作方法]

入力シミュレーション・モード時，1 シミュレーションの結果，接続端子の出力情報を受け取りその値にしたがって表示します。

行方向端子と列方向端子のマトリクス上での交点で両方の端子がアクティブの際，対応する LED が点灯します。

図 A—86　接続部品表示例（マトリクス LED）



Parts Buzzer Properties ダイアログ

入出力パネル ウインドウの接続部品の一つであるブザーの端子接続情報の設定，変更を行います。

入力シミュレーション・モード時，端子と接続したブザーは，接続した端子からの出力情報をビットマップで表示します（表示確認のみ）。

なお，ブザーの表示スタイルには，図形とビットマップの 2 種類があり，これらスタイルの変更は［[スタイル］タブ］で行います。

図 A—87　Parts Buzzer Properties ダイアログ：［ブザー端子接続］タブ

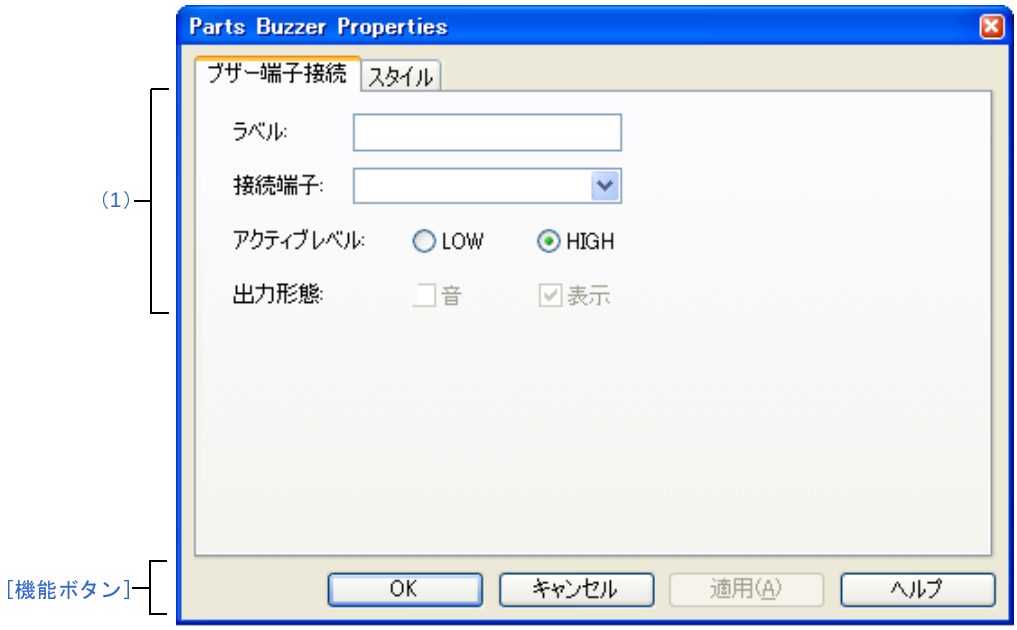


図 A—88　Parts Buzzer Properties ダイアログ：［スタイル］タブ

ここでは，次の項目について説明します。

- ［オープン方法］
- ［［ブザー端子接続］タブ］
- ［［スタイル］タブ］
- ［機能ボタン］
- ［表示例と操作方法］

［オープン方法］

入出力パネル ウインドウにおける次のいずれか

- 部品オブジェクト“ブザー”をダブルクリック
- 部品オブジェクト“ブザー”のコンテキスト・メニューより［プロパティ ...］を選択
- 部品オブジェクト“ブザー”を選択したのち，［表示］メニュー→［プロパティ ...］を選択

［［ブザー端子接続］タブ］

（1）ブザー端子接続設定エリア

ラベル	部品に名前を付ける際に指定します。 ここで指定した名前は部品一覧 ダイアログ上でラベルとして表示されます。	
接続端子	接続する端子名（出力端子）を指定するエリアです。 指定は，ドロップダウン・リストからの選択，または直接入力により行います。	
アクティブレベル	アクティブ状態をオプション・ボタンにより選択します。	
	LOW	アクティブ・レベルを LOW に設定します。
	HIGH	アクティブ・レベルを HIGH に設定します（デフォルト）。
出力形態	この項目は，変更不可です。	

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

[[スタイル] タブ]

(1) スタイル情報設定エリア

図形	ブザーを図形で表示する場合，このオプション・ボタンを選択します。		
	形状	図形の形状（四角，楕円のいずれか）を選択します。	
	影	選択不可	
	線	図形の線に関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		太さ	線の太さを指定します。 スピン・ボタンでの選択，または直接入力により行います。 1 ～ 100 までの範囲での指定が可能です。
		アクティブ	アクティブ表示時の線の色を指定します。
		インアクティブ	インアクティブ表示時の線の色を指定します。
	塗りつぶし	図形の塗りつぶしに関する指定，変更を行います。 プルダウン・ボタンをクリックすることにより色の指定が可能です。	
		アクティブ	アクティブ表示時の塗りつぶしの色を指定します。
		インアクティブ	インアクティブ表示時の塗りつぶしの色を指定します。
ビットマップ	ブザーを指定したビットマップで表示する場合，このオプション・ボタンを選択します（デフォルト）。		
	選択リスト	使用するビットマップを選択します（リストには，現在選択可能なビットマップが表示されます）。	
	[追加] ボタン	次のビットマップの追加 ダイアログがオープンし，選択リストに新規にビットマップを追加します。[...] ボタンによるファイル選択，または直接入力によりファイルを指定します。 	
	[削除] ボタン	現在選択リストで選択しているビットマップを削除します。 ただし，ユーザにより追加されたビットマップのみ削除可能です。	

(2) プレビュー・エリア

現在設定しているブザーのスタイルを表示します。

[機能ボタン]

ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。
適用	選択不可
ヘルプ	このダイアログのヘルプを表示します。

[表示例と操作方法]

入力シミュレーション・モード時，接続した端子のアクティブ・レベル出力をビットマップで表示します。
端子の出力値（アクティブ／インアクティブ）により，次のようなビットマップとして表示されます。

図 A—89　接続部品表示例（ブザー）

プルアップ／プルダウン設定 ダイアログ

入出力パネル ウインドウの接続部品の一つであるプルアップ／プルダウン抵抗の端子接続情報の設定，変更を行います。

この接続部品の設定方法は他の部品とは異なり，このダイアログで全端子の接続情報を一括管理します。

図 A—90　プルアップ／プルダウン設定 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］

入出力パネル ウインドウにフォーカスがある状態で次のいずれか

- ボタンのクリック
- ［部品］メニュー→［プルアップ／プルダウン設定 ...］の選択

[各エリアの説明]

(1) 接続情報表示エリア

端子名	プルアップ／プルダウン抵抗と接続できる端子名を表示します。	
Pull Up/Pull Down	端子の接続状態を表示します。	
	Pull Up	プルアップ抵抗接続状態を示します。
	Pull Down	プルダウン抵抗接続状態を示します。
	—	プルアップ抵抗／プルダウン抵抗未接続状態を示します（デフォルト）。
ボタン	プルアップ／プルダウン抵抗の端子接続情報を変更します。	
	プルアップ	選択した端子をプルアップ抵抗と接続します。 接続すると“Pull Up”が表示されます。
	プルダウン	選択した端子をプルダウン抵抗と接続します。 接続すると“Pull Down”が表示されます。
	解除	選択した端子の接続状態を解除します。 解除が完了すると“-”が表示されます。

[機能ボタン]

ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。

Object Properties ダイアログ

入出力パネル ウインドウの図形オブジェクト（文字／ビットマップを含む）の端子接続情報の設定，変更を行います。

入力シミュレーション・モード時，端子と接続した各オブジェクトは，接続端子の出力状態により，表示／非表示を切り替えます。

なお，各信号のアクティブ状態は，アクティブ HIGH です。

表示スタイルの変更は［[スタイル］タブ］で行います。

図 A—91　Object Properties ダイアログ：[端子接続］タブ

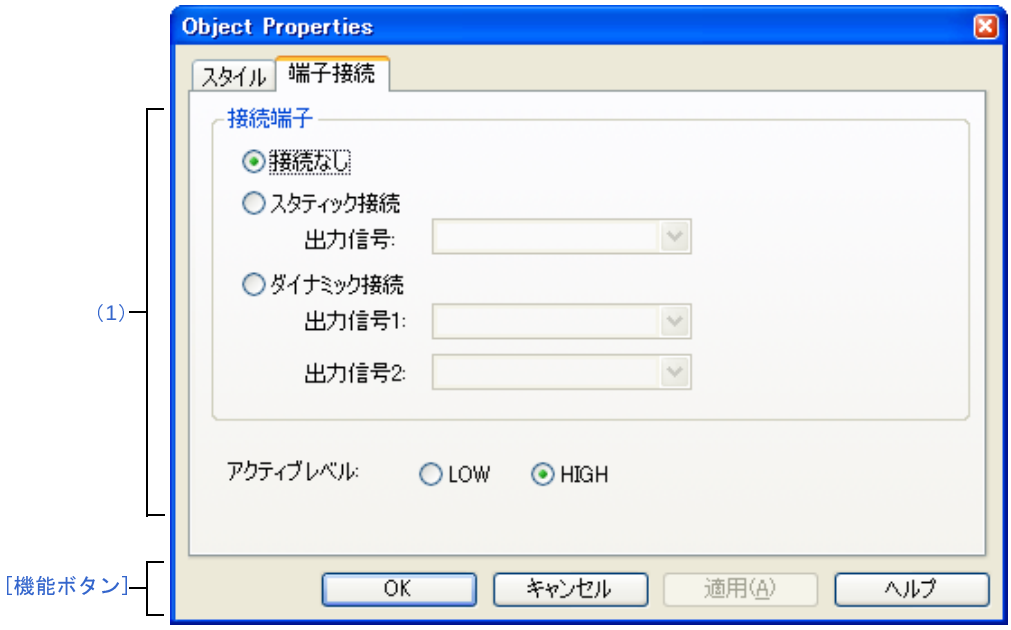
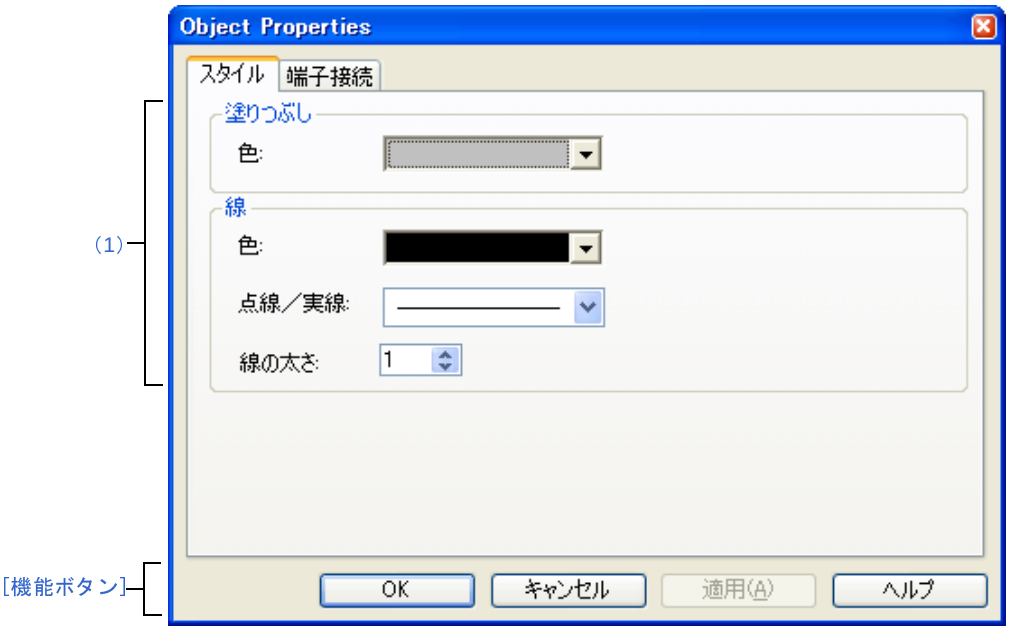


図 A—92　Object Properties ダイアログ：[スタイル］タブ



ここでは，次の項目について説明します。

- ［オープン方法］
- ［［端子接続］タブ］
- ［［スタイル］タブ］
- ［機能ボタン］

［オープン方法］

入出力パネル ウインドウにおける次のいずれか

- 図形オブジェクトをダブルクリック
- 図形オブジェクトのコンテキスト・メニューより［プロパティ ...］を選択
- 図形オブジェクトを選択したのち，［表示］メニュー→［プロパティ ...］を選択

［［端子接続］タブ］

（1）端子接続設定エリア

接続端子	オブジェクトと出力端子との接続方法をオプション・ボタンにより選択し，各出力端子名を指定します。 接続することにより，接続した出力端子の ON/OFF 状態で図形の表示が切り替わります。	
	接続なし	オブジェクトを端子と接続しません（デフォルト）。 端子に接続していないオブジェクトは，常に表示状態です。
	スタティック接続	オブジェクトを 1 つの出力端子と接続します。 ［出力信号］に，接続する端子名をドロップダウン・リストからの選択，または直接入力により指定します。 シミュレーション中に，指定した端子の出力信号データがアクティブの際，オブジェクトが表示されます。
	ダイナミック接続	オブジェクトを 2 つの出力端子と接続します。 ［出力信号 1］／［出力信号 2］に接続する端子名をドロップダウン・リストからの選択，または直接入力により指定します。 シミュレーション中に，指定した端子の出力信号データがともにアクティブの際，オブジェクトが表示されます。
アクティブレベル	各出力信号共通のアクティブ状態をオプション・ボタンにより選択します。	
	LOW	アクティブ・レベルを LOW に設定します。
	HIGH	アクティブ・レベルを HIGH に設定します（デフォルト）。

備考　指定する端子名に関しては，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

[[スタイル] タブ]

(1) スタイル情報設定エリア

塗りつぶし	各オブジェクトの塗りつぶしに関する設定，変更を行うエリアです注。 対象オブジェクトにより，塗りつぶしをする範囲は次のようになります。 - 線 対象外 - 四角，楕円，丸四角 輪郭となる線で囲まれた範囲内 - 多角形 各頂点を結ぶ線で囲まれた範囲内 - 文字 テキスト・ボックス内 - ビットマップ 図形描画領域内	
	色	塗りつぶしの色を指定，変更します。 プルダウン・ボタンをクリックすることにより色の指定が可能です。
線	各オブジェクトの線の形状に関する設定，変更を行うエリアです。 各オブジェクトにより，線の定義は次のようになります。 - 線 すべて - 四角，楕円，丸四角 輪郭となる線 - 多角形 各頂点を結ぶ線 - 文字 テキスト・ボックスの輪郭線 - ビットマップ 図形描画領域の輪郭線	
	色	線の色を指定，変更します。 プルダウン・ボタンをクリックすることにより色の指定が可能です。
	点線／実線	線の形状（点線／実線）をドロップダウン・リストにより指定，変更します。 ただし，[線の太さ] での指定が“1”の時のみ変更可能です。
	線の太さ	線の太さを指定，変更します。 スピン・ボタンでの選択，または直接入力により行います。 1 ～ 100 までの範囲での指定が可能です。

注　対象オブジェクトがビットマップ・ファイルからの貼り付けの場合，表示していたビットマップは見えなくなります。

[機能ボタン]

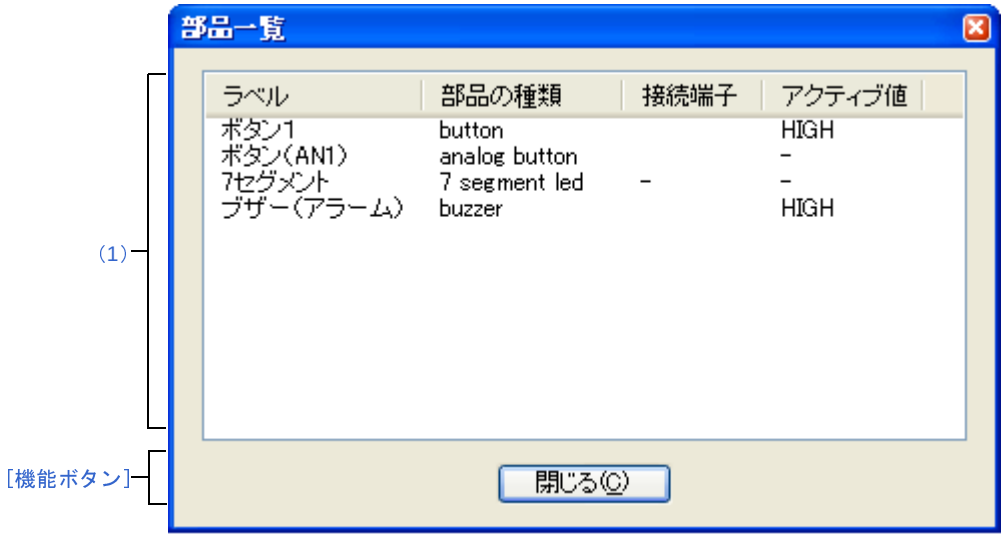
ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。
適用	選択不可
ヘルプ	このダイアログのヘルプを表示します。

部品一覧 ダイアログ

入出力パネル ウインドウ上に作成したすべての図形オブジェクト，および部品オブジェクトの端子接続状況を一覧表示します。

なお、各オブジェクトの端子接続の設定内容を変更するには，一覧内のオブジェクトをダブルクリック，または，一覧内のオブジェクトを選択した状態で，［表示］メニュー→［プロパティ ...］を選択することによりオープンする設定ダイアログで行います。

図 A—93　部品一覧 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］

- 入出力パネル ウインドウにフォーカスがある状態で，［表示］メニュー→［接続部品一覧 ...］を選択

[各エリアの説明]

(1) 端子接続状況表示エリア

ラベル	オブジェクトに付けたラベル（名前）を表示します。ラベルがないオブジェクトでは何も表示されません。	
部品の種類	部品の種類を表示します。	
	rectangle	直線，四角形，楕円，丸四角，扇型
	polygon	多角形
	text	テキスト
	bitmap	ビットマップ
	button	プッシュ・ボタン，トグル・ボタン，グループ・ボタン
	analog button	アナログ・ボタン
	key	キー・マトリクス
	level gauge	レベル・ゲージ
	led	LED
	7segment led	7 セグメント LED
	14segment led	14 セグメント LED
	matrix led	マトリクス LED
	buzzer	ブザー
	groups	グループ化された部品
接続端子	部品と接続されている端子を表示します。 なお，複数端子と接続されている部品では“-”が，接続されていない部品では“(空白)”が表示されます。	
アクティブ値	部品に設定されているアクティブ値を表示します。 なお，複数端子と接続されている部品では“-”が，接続されていない部品では“(空白)”が表示されます。	

[機能ボタン]

ボタン	機能
閉じる	このダイアログをクローズします。

シリアル ウインドウ

CPU に搭載されているシリアル・インタフェースと通信を行います。

このウインドウは CPU の他局側のシリアル・インタフェースとして動作するため，CPU からの送信データがこのウインドウでの受信データに，このウインドウからの送信データが CPU での受信データになります。

このウインドウでは，次の 2 種類のファイルを扱うことができます。

ウインドウ上部エリアで表示された送受信データは，［ファイル］メニュー→［上書き保存］／［名前を付けて保存...］の選択により，シリアル・ログデータ・ファイル（*.log）（CSV 形式）として保存することができます。

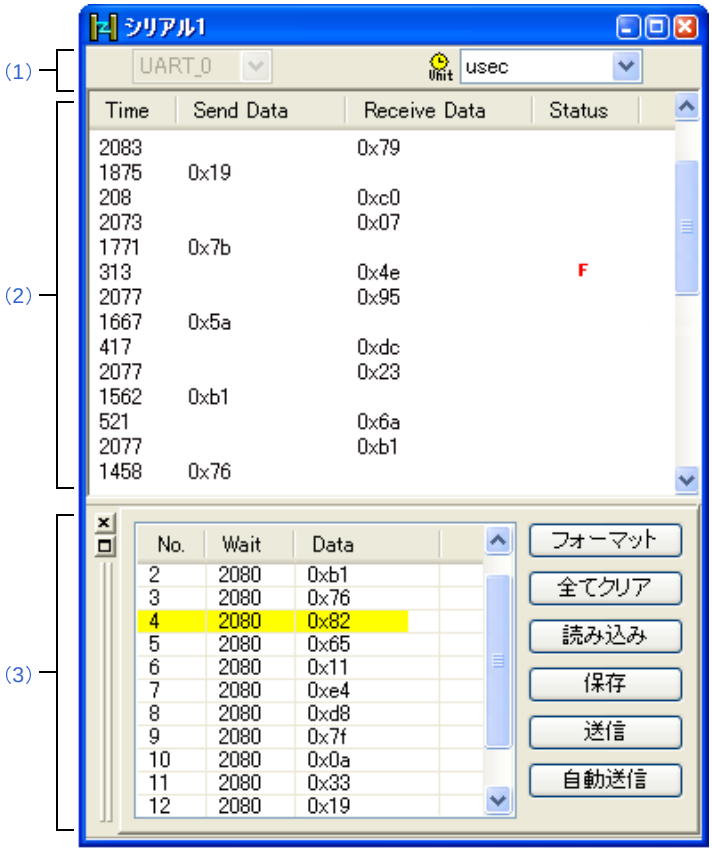
また，ウインドウ下部エリアで作成した送信データは，［保存］ボタンによりシリアル送信データ・ファイル（*.ser）（CSV 形式）として保存することができます（［読み込み］ボタンにより復元可能）。

なお，作成したデータの保存／復元はプロジェクト・ファイルの保存／ロードでも行うことができますが，この場合，送信データは CSV 形式のテキスト・ファイルとして保存されず，プロジェクト・ファイル内に保存されます。

注意 1.　保存したシリアル送信データ・ファイルをオープンする際，またはプロジェクト・ファイルをオープンする際に，シリアル送信データ・ファイルを作成した時点のマイクロコントローラとは異なるマイクロコントローラでシミュレータ GUI が起動されていた場合，そのマイクロコントローラに存在しないシリアルの設定は復元されません。

2.　このウインドウは複数オープン可能です。ウインドウのオープン後に，シリアル選択エリアで検証したいシリアル・インタフェースを選択してください。

図 A—94　シリアル 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］
- ［専用メニュー（シリアル ウインドウ）］
- ［コンテキスト・メニュー］

［オープン方法］

- ボタンをクリック
- ［シミュレータ］メニュー→［シリアル］を選択

［各エリアの説明］

（1）シリアル選択エリア

通信するシリアル・インタフェースを選択します。

UART00	搭載されているシリアル・インタフェースの一覧（ドロップダウン・リスト）から通信するシリアル・インタフェースを選択します注。
usec	シリアル・エディタ・エリアの［Wait］に使用される時間情報の単位をドロップダウン・リストから選択します。 なお，時間情報の単位は［編集］メニュー→［時間単位］からも変更できます。

注　一度選択すると，変更することはできません。
　　変更する場合は，新たなシリアル ウインドウをオープンしてください。

（2）ログ表示エリア

送受信データを表示します。

表示タイミングは，データを構成する全ビットの受信，または送信が完了した時点です。

このエリアには，スタート・ビット，ストップ・ビット，およびパリティ・ビットを削除したデータのみが表示されます。

データの表示方法は，［表示］メニュー→［数値表現］→［2 進表示］／［16 進表示］の選択により変更することができます。

なお，デバッガ，またはシミュレータのリセット発生によりログ表示はクリアされます。

Time	前データの送受信終了から今回の送受信終了までの時間を表示します。 時間情報の単位は，［編集］メニュー→［時間単位］により選択した単位となります。	
Send Data	このウインドウが送信したデータ（CPU 側が受信したデータ）を表示します。	
Receive Data	このウインドウが受信したデータ（CPU 側が送信したデータ）を表示します。	
Status	データ受信時の状態を表示します。 エラーが発生時には次のマークを表示します。正常時は何も表示しません。	
	P	パリティ・エラー（パリティ・ビットの不一致）
	F	フレーミング・エラー（ストップ・ビットが検出されない）

(3) シリアル・エディタ・エリア

送信データを作成するエリアです。

このエリアは，［表示］メニュー→［シリアル・エディタ］により表示／非表示の選択が可能です。

No.	先頭からシーケンシャルに付けられている行番号です。直接書き込みはできません。 最大 9999 行まで設定可能です。	
Wait	ひとつ前のデータ送信完了から次のデータを送信開始するまでの間の時間を指定します。［自動送信］ボタンでの送信時に有効です。 時間情報の単位は，［編集］メニュー→［時間単位］により選択した単位となります。 Wait 値の入力は，編集を行う Wait 欄にカーソルを置きダブルクリックすることにより行います。 1 回の操作で 1 つ分の Wait 値の書き込みが可能です。	
Data	送信データを編集するエリアです。 Data 欄にカーソルを置きダブルクリックすることにより，直接書き込み可能です。 先頭に“0x”と付与すると 16 進数として，“0b”を付与すると 2 進数として扱われます。デフォルト進数は 16 進数です。 フォーマット（UART）ダイアログ，またはフォーマット（CSI）ダイアログで設定したビット長と異なるビット長を指定した場合，下位ビットからのデータが有効になります。 1 回の操作で 1 つ分のデータ書き込みが可能です。	
ボタン	フォーマット	フォーマット（UART）ダイアログ，またはフォーマット（CSI）ダイアログをオープンします。
	全てクリア	シリアル・エディタ・エリアをすべて空白にします。
	読み込み	以前に保存したシリアル送信データ・ファイル（*.ser）の内容を読み込み，シリアル・エディタ・エリアに再現します。 ただし，UART 用として作成されたファイルを CSI 用として読み込むことはできません。また，CSI 用として作成されたファイルを UART 用として読み込むこともできません。
	保存	シリアル・エディタ・エリアで設定した送信データの内容を指定したシリアル送信データ・ファイル（*.ser）に保存します。
	送信	シリアル・エディタ・エリアで選択されているデータを 1 つ送信します。 送信の完了により次のデータが選択状態になります。 データが選択されてない場合，先頭のデータを送信します。
	自動送信	シリアル・エディタ・エリアで選択されているデータを先頭にし，下方向にデータを自動送信します。データ送信の時間間隔は Wait に指定した時間となります。

注意　このエリア内にカーソルがある場合，［F1］キーを押下してもこのウインドウのヘルプは表示されません。

備考　このウインドウの CSI をマスタ・モードとして設定する場合，受信時にクロックが必要になるため，受信動作を行うためにダミー・データを送信する必要があります。

[専用メニュー（シリアル ウインドウ）]

(1)［編集］メニュー

挿入	選択行の直前に新しい行を挿入します。
切り取り	選択範囲を切り取りクリップボードに保存します。
コピー	選択範囲をコピーしクリップボードに保存します。
貼り付け	クリップボードの内容を選択位置に貼り付けます。
削除	選択範囲を削除します。
時間単位	時間の単位を選択します。
メインクロック	ウエイト時間の単位をメイン・クロックとします（デフォルト）。
マイクロ秒	ウエイト時間の単位をマイクロ秒とします。
ミリ秒	ウエイト時間の単位をミリ秒とします。
フォーマット設定 ...	フォーマット（UART）ダイアログ，またはフォーマット（CSI）ダイアログをオープンします。

(2)［表示］メニュー

シリアル・エディタ	シリアル・エディタ・エリアの表示／非表示を切り換えます。
数値表現	ログ表示エリアの表示方法を変更します。
2 進数表示	2 進数表示します。
16 進数表示	16 進数表示します。

(3)［オプション］メニュー

ウインドウのカスタマイズ ...	書式設定 ダイアログをオープンします。

[コンテキスト・メニュー]

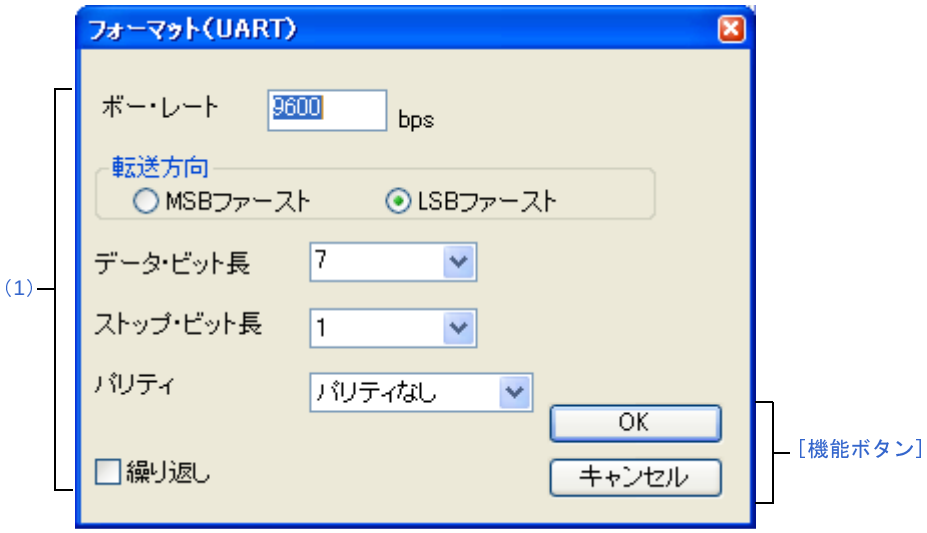
シリアル・エディタ・エリアにおいて，次に示すコンテキスト・メニューを表示します。

挿入	選択行の直前に新しい行を挿入します。
切り取り	選択範囲を切り取りクリップボードに保存します。
コピー	選択範囲をコピーしクリップボードに保存します。
貼り付け	クリップボードの内容を選択位置に貼り付けます。
削除	選択範囲を削除します。

フォーマット（UART）ダイアログ

アシンクロナス・シリアル・インタフェース（UART）用のシリアル・フォーマットの設定，変更を行います。

図 A—95　フォーマット（USRT）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］

UART 系シリアル・インタフェースを選択しているシリアル ウインドウにおける次のいずれか

- ［フォーマット］ボタンをクリック
- ［編集］メニュー→［フォーマット設定 ...］を選択

[各エリアの説明]

(1) フォーマット（UART）設定エリア

ボー・レート	シリアルのボー・レート値（単位：bps）を整数で直接入力します。	
転送方向	転送方向を選択します。	
	MSB ファースト	転送方向を MSB ファーストとします。
	LSB ファースト	転送方向を LSB ファーストとします（デフォルト）。
データ・ビット長	送信データのビット長をドロップダウン・リストからの選択，または直接入力により指定します（デフォルト：7）。	
ストップ・ビット長	ストップ・ビット長をドロップダウン・リストから選択します（デフォルト：1）。	
パリティ	パリティ情報（パリティなし（デフォルト）／奇数パリティ／偶数パリティ／0 パリティ）を選択します。	
繰り返し	シリアル ウインドウの［自動送信］ボタンをクリックした際に，データ転送の繰り返しを行う場合，チェックします。	
	☑	自動送信で最後のデータを送信した後，データの先頭に戻って自動送信を続けます。
	☐	自動送信で最後のデータを送信した後，送信を停止します（デフォルト）。

備考　選択可能な範囲については，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

[機能ボタン]

ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。

フォーマット（CSI）ダイアログ

3 線式シリアル・インタフェース（CSI）用のシリアル・フォーマットの設定，変更を行います。

図 A—96　フォーマット（CSI）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］
- ［3 線式シリアル・インタフェース（CSI）選択時の送受信動作に関して］

［オープン方法］

CSI 系シリアル・インタフェースを選択しているシリアル ウインドウにおける次のいずれか

- ［フォーマット］ボタンをクリック
- ［編集］メニュー→［フォーマット設定 ...］を選択

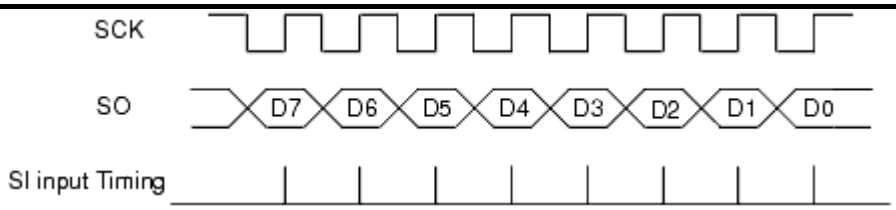
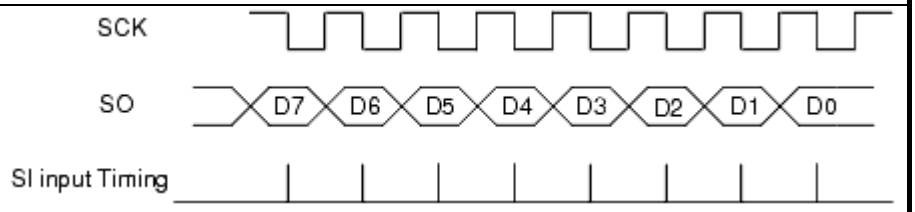
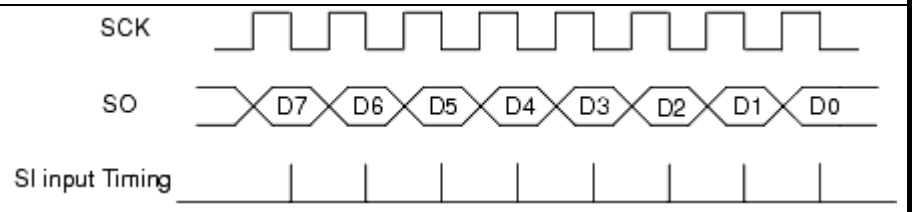
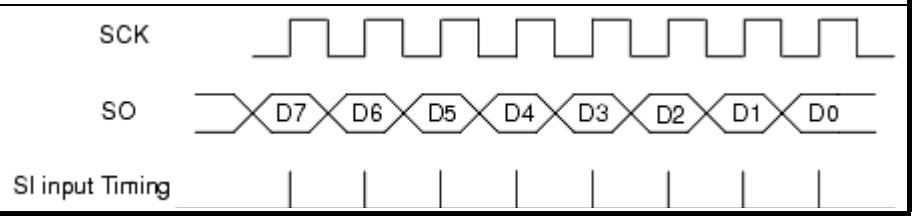
[各エリアの説明]

(1) シリアル・フォーマット設定エリア

マスタ，スレーブ	転送モードを選択します。	
	マスタ	このウインドウ側をマスタとして動作します。 通信の際クロックを生成するため［転送クロック］の設定が必要です。
	スレーブ	このウインドウ側をスレーブとして動作します（デフォルト）。 CPU 搭載のシリアル・インタフェースのクロックで通信します。
転送クロック	転送クロック値を直接入力します（単位：kHz）。 小数点付き数値の設定も可能です。 マスタを選択時には必ず設定します。	
転送方向	転送方向を選択します。	
	MSB ファースト	転送方向を MSB ファーストとします（デフォルト）。
	LSB ファースト	転送方向を LSB ファーストとします。
データ・ビット長	送信データのビット長をドロップダウン・リストからの選択，または直接入力により指定します（デフォルト：8）。	
データ位相	送受信のタイミングを選択することでデータ位相を設定します。 ［クロック位相］との組み合わせにより，「表 A—15　データ・クロック位相設定表」で示すデータ・クロック位相となります。	
	通常	3 線式シリアルの通常の送受信タイミングでデータの送受信を行います（デフォルト）。
	先行	3 線式シリアルの通常の送受信タイミングより，動作クロックの半クロック分先行したタイミングでデータの送受信を行います。
クロック位相	送受信のクロック波形を選択することでクロック位相を設定します。 ［データ位相］との組み合わせにより，「表 A—15　データ・クロック位相設定表」で示すデータ・クロック位相となります。	
	通常	3 線式シリアルの通常のクロックで動作します（クロックの立ち下がりを転送開始とします）（デフォルト）。
	反転	3 線式シリアルの通常のクロックを反転したクロックで動作します（クロックの立ち上がりを転送開始とします）。
繰り返し	シリアル ウインドウの［自動送信］ボタンをクリックした際に，データ転送の繰り返しを行う場合，チェックします。	
	☑	自動送信で最後のデータを送信したのち，データの先頭に戻って自動送信を続けます。
	☐	自動送信で最後のデータを送信したのち，送信を停止します（デフォルト）。

備考　選択可能な範囲については，使用するマイクロコントローラのユーザーズ・マニュアルを参照してください。

表 A—15　データ・クロック位相設定表

データ位相設定	クロック位相設定	データ・クロック位相
通常	通常	
先行	通常	
通常	反転	
先行	反転	

[機能ボタン]

ボタン	機能
OK	設定を有効にし，このダイアログをクローズします。
キャンセル	設定を無視し，このダイアログをクローズします。

[3 線式シリアル・インタフェース（CSI）選択時の送受信動作に関して]

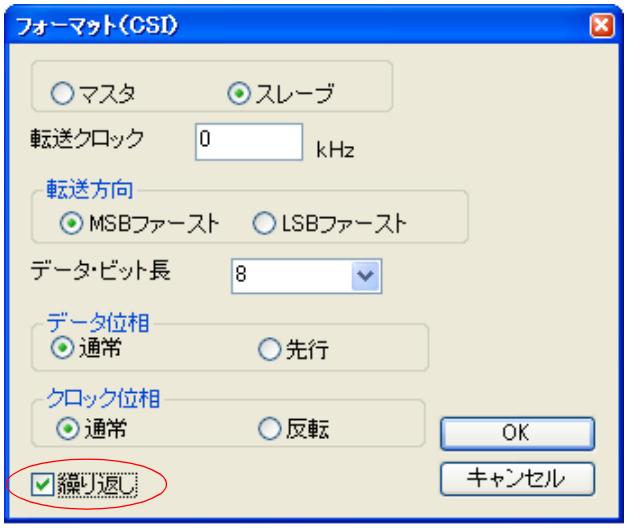
CSI 系シリアル・インタフェース選択時のシリアル ウインドウは，このダイアログで［マスタ］，または［スレーブ］のどちらを選択していても，次のように常に送受信モードで動作します。

［マスタ］選択時	シリアル・エディタ・エリアの［送信］ボタン，または［自動送信］ボタンの押下後すぐにデータの送受信が行われます。
［スレーブ］選択時	シリアル・エディタ・エリアの［送信］ボタン，または［自動送信］ボタンの押下によりデータ送受信レディ状態となります。 データ送受信レディ状態では，CSI クロック信号を受け取ることによりデータの送受信が開始され，データの送受信の終了によりこの状態が解除されます（データ送受信レディ状態以外の状態では，CSI クロック信号を受け取ってもデータの送受信は行われません）。

このため，［スレーブ］選択時に，シリアル ウインドウでデータ受信のみを行いたい場合には，次の手順で操作してください（CSI シリアル・スレーブ選択時の受信設定方法）。

(1)［繰り返し］の指定

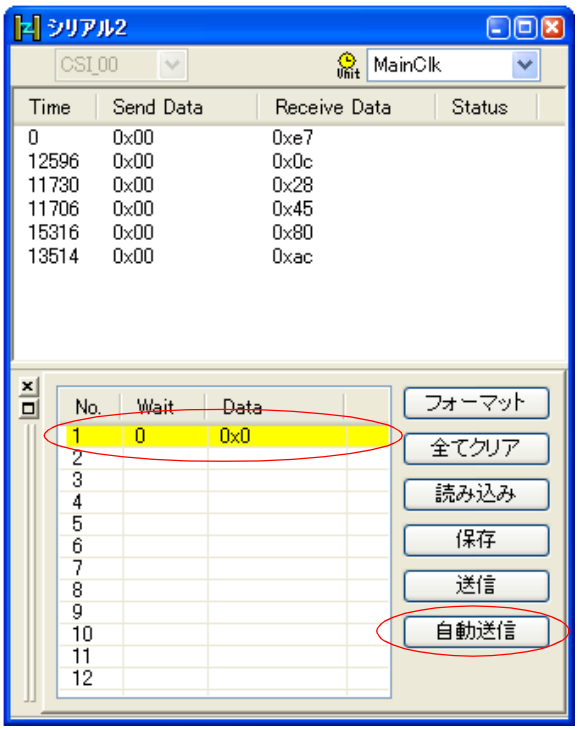
［繰り返し］ボタンをチェックします。



(2) Wait 時間の設定

シリアル ウインドウで Wait 時間 0 のダミー設定を行います。

(3)［自動送信］ボタンのクリック



付録 B　ユーザ・オープン・インタフェース

この付録では，シミュレータ GUI が提供する機能の 1 つである，ユーザ・オープン・インタフェースについての詳細を説明します。

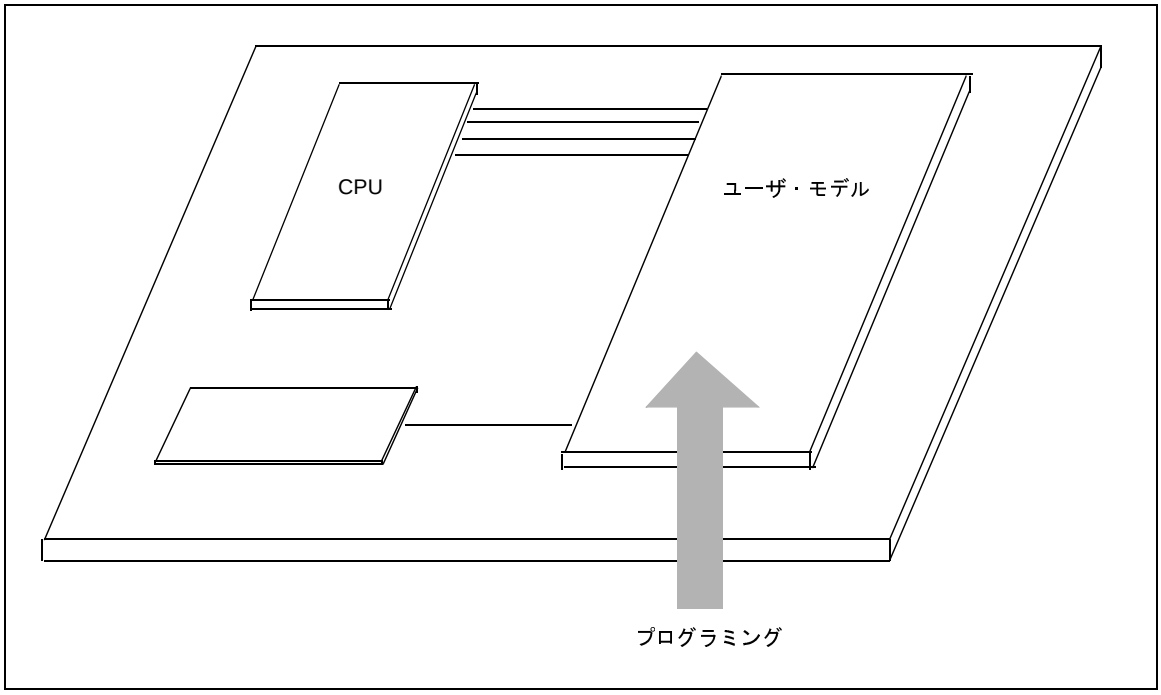
B. 1　概　　要

シミュレータ GUI では，CPU（CPU コア＋内蔵周辺）のシミュレーションに加え，次に挙げる 2 つのターゲット・システムのシミュレーション環境構築手段を用意しています。

1 つは，入出力パネル ウインドウを使用する方法で，標準的な接続部品とその操作環境を提供することにより，GUI 操作による簡単なシミュレーション環境を構築することができます。

もう 1 つが，この付録で説明するユーザ・オープン・インタフェースを使用する方法で，ユーザが外部のユーザ・モデルをプログラミングすることにより，入出力パネル ウインドウでは実現不可能なシミュレーション環境を構築することができます。

図 B—1　ユーザ・モデルのプログラミング・イメージ



B. 1. 1　インタフェース関数の種類

シミュレータ GUI のユーザ・オープン・インタフェースでは，次の種類のインタフェース関数を提供しています。各関数についての詳細は，「B. 4　提供インタフェース関数」を参照してください。

表 B—1　提供インタフェース関数の種類

種類	概要
基本インタフェース関数	シミュレータの基本機能 - 初期化通知 - リセット通知など
時間インタフェース関数	ユーザ・モデルの時系列処理を行うための周期タイマ機能 - タイマの設定 - タイマの解除 - タイマの時間通知など
端子インタフェース関数	端子の入出力機能 - 端子への信号出力 - 端子への信号入力通知など
外部バス・インタフェース関数注	外部バスのスレーブ機能 - 外部バス・リード・アクセス通知 - 外部バス・ライト・アクセス通知など
シリアル・インタフェース関数	シリアルの送受信機能 - シリアル・データの送信 - シリアル・データの受信通知など
信号出力器インタフェース関数	信号データ・ファイルに従った信号出力機能 - 信号データ・ファイルに従った信号出力など

注　外部バス・インタフェース関数を使用する場合は，使用する外部メモリ領域を，メモリ・マッピング ダイアログ上の［メモリ種別］エリアにおいて［ターゲット・メモリ領域］に指定する必要があります。

B. 1. 2　インタフェース方式

シミュレータ GUI が提供するユーザ・オープン・インタフェースのインタフェース方式は，次のとおりです。

(1) C 言語インタフェース

C 言語の API（Application Program Interface）関数セットで構成されています。

したがって，ユーザ・モデルのプログラミングは，C 言語で行います。

(2) コールバック関数方式

システム側からプログラムを呼び出す手段として，コールバック関数方式を採用しています。

コールバック関数方式とは，プログラムで作成した関数へのポインタをあらかじめシステム側に通知しておき，システム側は，その関数へのポインタを使用してプログラムで作成した関数を呼び出す方式です。

プログラムからシステム側を呼び出す API 関数に対し，コールバック関数は，システム側からプログラムを呼び出す場合（たとえば，端子への信号入力など）に使用します。

(3) イベント・ドリブン方式

事象（イベント）の発生に従い処理を記述する，イベント・ドリブン方式を採用しています。

したがって，シミュレータ GUI 側で，初期化 /CPU リセット／端子への信号入力／外部バス・アクセスなどの事象が発生した時点で，ユーザ・モデル側で用意したコールバック関数を呼び出します。

また，ユーザ・モデルにおいて時系列処理を行うために設けている“時間インタフェース”（タイマ機能）でも，設定した時間に到達した時点で，ユーザ・モデル側で容易したコールバック関数を呼び出します。

B. 1. 3　開発環境

シミュレータ GUI が提供するユーザ・オープン・インタフェースでプログラミングを行い，DLL ファイルを作成する際は，次の開発ツールを使用してください。

- Microsoft Visual C++ （Ver. 6.00 以上）

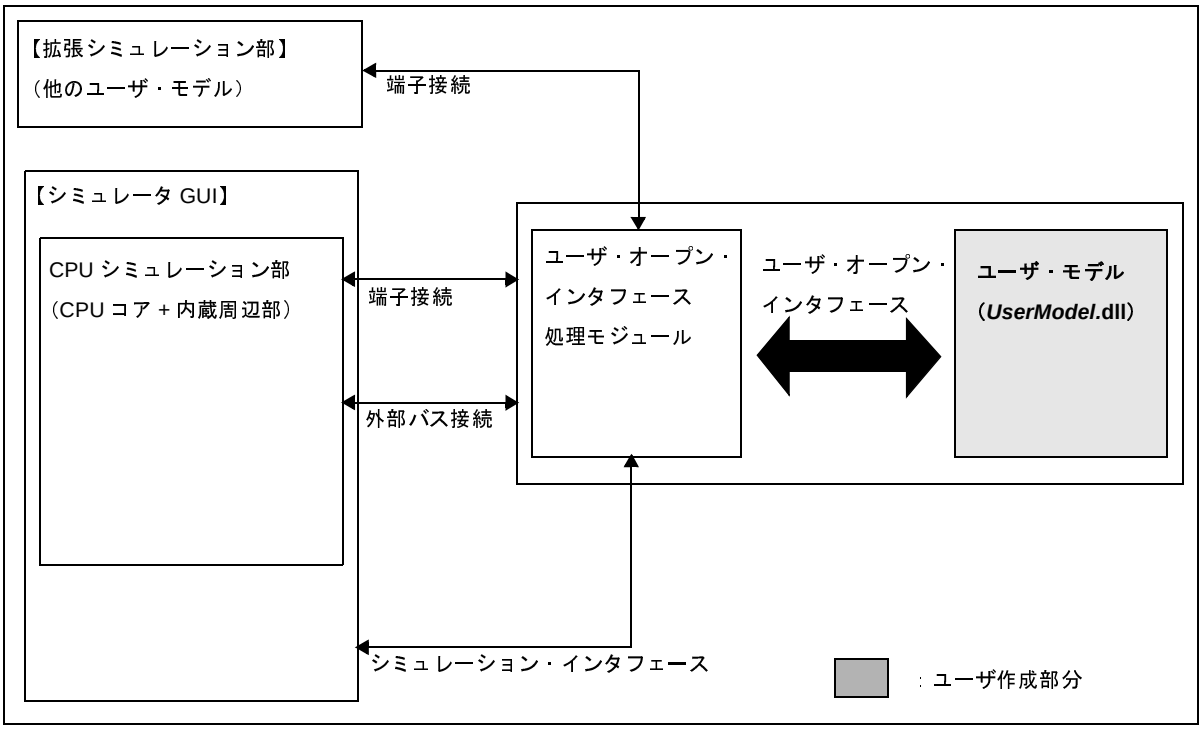
B. 2　ユーザ・モデルの作成

この節では，ユーザ・モデル（*UserModel*.dll）の作成方法について説明します。

B. 2. 1　プログラム構成

次に，シミュレータ GUI が提供するユーザ・オープン・インタフェースを使用し，システムを拡張する際のプログラム構成を示します。

図 B—2　プログラム構成図



システムを拡張するためには，まず，ユーザ・モデルを作成する必要があります。

ユーザ・モデルは，シミュレーション・システムと連携動作をするため，ユーザ・オープン・インタフェース処理モジュールとインタフェースをとります。このインフェースが，ユーザ・オープン・インタフェースとなります。

ユーザ・モデルは，ユーザ・オープン・インタフェースを介することで，コンフィギュレーション（シミュレータ GUI 起動時に行うシミュレータ構築処理）時に端子や外部バス・スレーブなどの資源を生成します。これにより生成された端子や外部バス・スレーブを CPU シミュレーション部の端子や外部バス・マスタに接続することにより，CPU シミュレーション部の端子の入出力，および CPU シミュレーション部からの外部バス・アクセスの処理が可能となります。

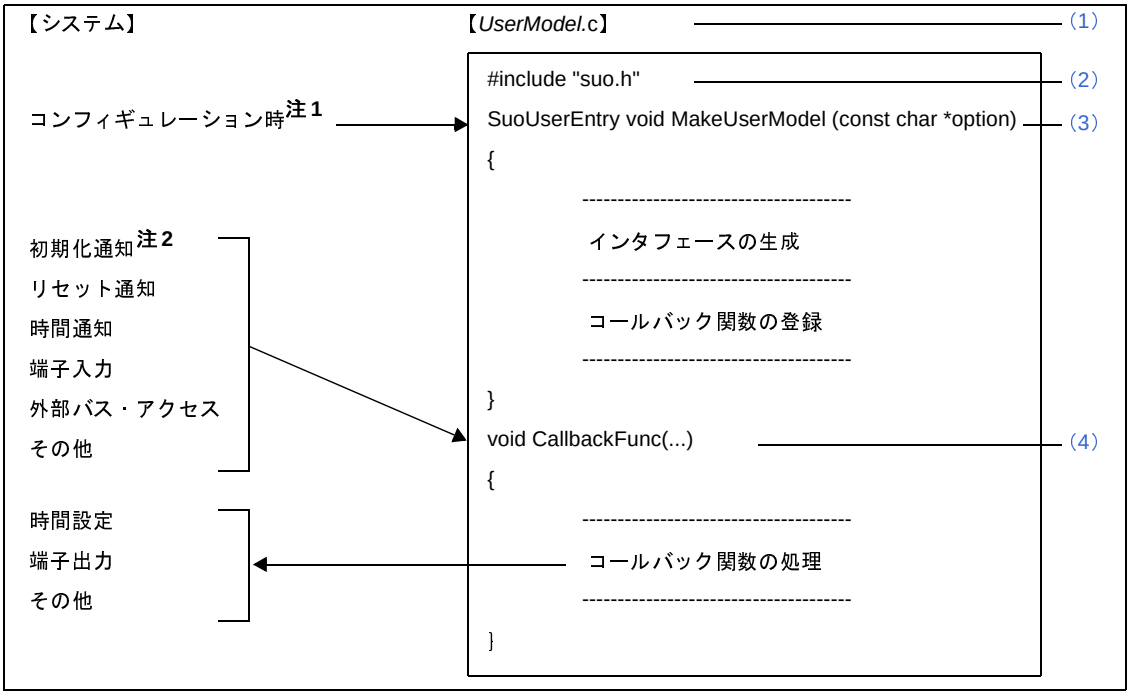
なお，生成された端子や外部バス・スレーブは，CPU シミュレーション部だけではなく拡張シミュレーション部（他のユーザ・モデル）とも接続することができます。

B. 2. 2　ユーザ・モデルのプログラミング

ユーザ・モデルは，WIN32 ダイナミック・リンク・ライブラリ（DLL）形式でプログラミングします。

次に，プログラム・ファイル（*UserModel*.c）のテンプレートを示します。

図 B—3　プログラム・ファイルのテンプレート



注 1.　シミュレータ GUI 起動の際に行うシミュレータ構築処理時

2.　シミュレータ構築処理が完了し，シミュレータ GUI が起動した直後の 1 回のみ通知されます。

（1）ファイル名

作成するユーザ・モデルのファイル名を示します。

ファイル名は任意に指定することができます。

ただし，C ソース・ファイルであるため，拡張子は“*.c”固定です。

（2）インクルード・ファイル

インクルード・ファイルを示します。

ユーザ・オープン・インタフェースを使用するためには，システム・ヘッダ・ファイル“suo.h”をインクルードする必要があります。

（3）MakeUserModel 関数

MakeUserModel 関数（シミュレータ GUI のコンフィギュレーション時にシステム側から呼び出される関数）を示します。関数名は，“MakeUserModel”である必要があります。

【指定形式】

```
SuoUserEntry void MakeUserModel(const char *option);
```

なお，この関数の中では，次の 2 つの処理を記述します。

(a) インタフェースの生成

シミュレータ GUI 起動時のコンフィギュレーション処理において端子やバス接続を行うため，コンフィギュレーションのタイミングで，接続する端子やバスなどの資源を生成しておく必要があります。

これを行うために，MakeUserModel 関数の中でインタフェースの生成関数を呼び出し，インタフェースの生成を行います（「B. 4　提供インタフェース関数」参照）。これに伴い必要な資源の生成も行われます。

(b) コールバック関数の登録

必要に応じ，コールバック関数を登録することができます。

注意　初期化のコールバック関数を記述する場合は必ずこのタイミングで登録してください。このタイミングで登録しない場合，コールバックが機能しません。
これは，初期化通知が，MakeUserModel 関数呼び出しの次のタイミングであるためです。

(4) コールバック関数

コールバック関数を示します。

ここでは，初期化通知，リセット通知，時間通知，端子入力，外部バス・アクセスなどの複数のコールバック関数を作成することができます。コールバック関数の中ではコールバック内容に応じた処理を記述します（「B. 5　ユーザ定義関数」参照）。

なお，作成したコールバック関数は，システムから呼び出しできるように事前の登録が必要です（「B. 4　提供インタフェース関数」参照）。コールバック関数の名前は任意に決定することができ，その関数の形式はコールバックの種類によって異なります。

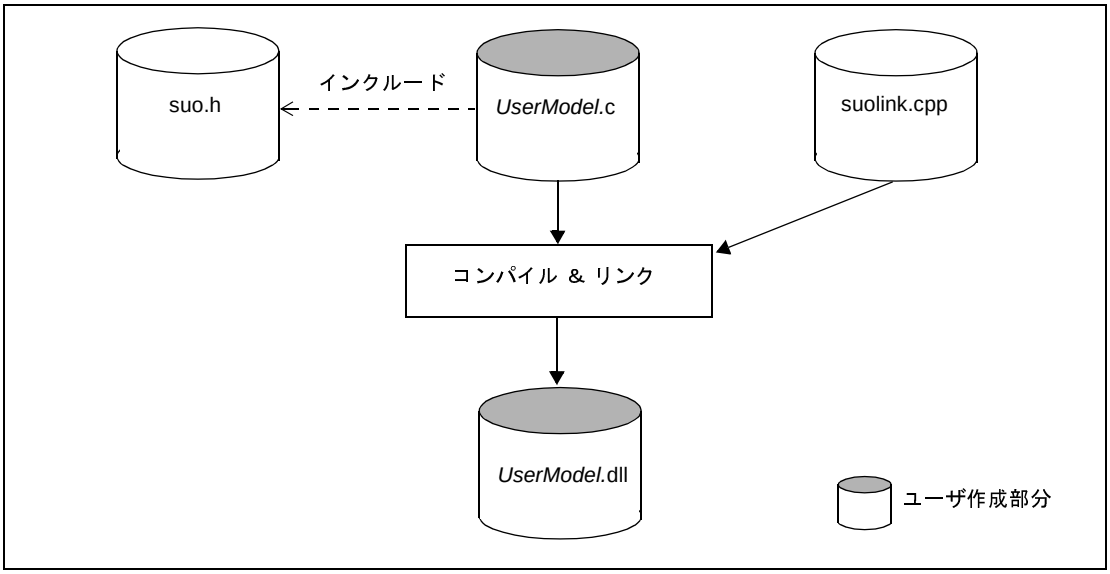
B. 2. 3　プログラム・ファイル（*UserModel*.c）の記述例

```
#include "suo.h"
#include <memory.h>

void Init(void);
void InputP00(SuoHandle handle, int pinValue);
void ReadBUS1(SuoHandle handle, unsigned long addr, int accessSize, unsigned char data[]);
void WriteBUS1(SuoHandle handle, unsigned long addr, int accessSize, const unsigned char data[]);

SuoHandle p00;
SuoHandle p01;
SuoHandle bus1;
unsigned char mem[0x100];

/* MakeUserModel */
SuoUserEntry void MakeUserModel(const char *option)
{
    SuoCreatePin("P00", &p00);
    SuoCreatePin("P01", &p01);
    SuoCreateExtbus("BUS1", 0x200000, 0x100, &bus1);

    SuoSetInitCallback(Init);
    SuoSetInputDigitalPinCallback(p00, InputP00);
    SuoSetReadExtbusCallback(bus1, ReadBUS1);
    SuoSetWriteExtbusCallback(bus1, WriteBUS1);
}

/* callbacks */
void Init(void)
{
    memset(mem, 0, 0x100);
}

void InputP00(SuoHandle handle, int pinValue)
{
    SuoOutputDigitalPin(p01, pinValue);
}

void ReadBUS1(SuoHandle handle, unsigned long addr, int accessSize, unsigned char data[])
{
    memcpy(data, &mem[addr-0x200000], accessSize);
}

void WriteBUS1(SuoHandle handle, unsigned long addr, int accessSize, const unsigned char data[])
{
    memcpy(&mem[addr-0x200000], data, accessSize);
}
```

B. 2. 4　コンパイルとリンク

作成した *UserModel*.c と suolink.cpp をコンパイル，およびリンクすることにより，*UserModel*.dll を作成します。

図 B—4　コンパイルとリンク



ファイル名	説明
suo.h	ユーザ・オープン・インタフェース用のシステム・ヘッダ・ファイルです。 プログラム（*UserModel*.c）がインクルードするだけで，コンパイル対象ではありません。 なお，suo.h は，デフォルトで次のフォルダに格納されています。 - *インストール・フォルダ*¥ CubeSuite+ ¥ DebugTools ¥ DebugTool78K0RSimulator ¥ useropen ¥ sys
suolink.cpp	システムのユーザ・オープン・インタフェース処理モジュールとのダイナミック・リンク処理を行うファイルです。 なお，suolink.cpp は，デフォルトで次のフォルダに格納されています。 - *インストール・フォルダ*¥ CubeSuite+ ¥ DebugTools ¥ DebugTool78K0RSimulator ¥ useropen ¥ sys
UserModel.c	作成するユーザ・モデルの C ソース・ファイルです。 ファイル名は任意に指定することができます。
UserModel.dll	ユーザ・モデルのバイナリ・ファイル（DLL ファイル）です。 ファイル名は任意に指定することができます。

注意　Microsoft Visual C++ がインストールされていない環境で DLL ファイルを動作させるには，DLL ファイルをリリース版で作成する必要があります。

B. 3　ユーザ・モデルの組み込み

この節では，作成したユーザ・モデル（*UserModel*.dll）をシミュレータ GUI に組み込む方法について説明します。

シミュレータ GUI への組み込みは，シミュレータ・コンフィギュレーション・ファイル（*.cfg）を使用します。

したがって，作成したユーザ・モデルを組み込んでシミュレータ GUI を使用する場合は，プロパティ パネルの［接続用設定］タブ上［コンフィギュレーション］カテゴリ内［シミュレータ・コンフィギュレーション・ファイルを使用する］プロパティにおいて，［はい］を指定したのち，同カテゴリ内の［シミュレータ・コンフィギュレーション・ファイル］プロパティにおいて，使用するシミュレータ・コンフィギュレーション・ファイルを指定する必要があります。

B. 3. 1　シミュレータ・コンフィギュレーション・ファイルへの記述

シミュレータ・コンフィギュレーション・ファイルには，作成したユーザ・モデルの生成処理，および端子／外部バスの接続処理などを記述します。

（1）ユーザ・モデルの生成処理
（2）端子の接続処理
（3）外部バスの接続処理
（4）その他の処理

（1）ユーザ・モデルの生成処理

```
UserModel1 = Device("USEROPEN", "UserModel1.dll UserOption1");
```

（a）*UserModel1*

生成したユーザ・モデルを表す変数です。変数名は任意に指定することができます。

（b）Device 関数

ユーザ・モデルを生成する関数です。

（c）"USEROPEN"

ユーザ・オープン・インタフェース処理モジュール（システム・モジュール）です。

（d）*UserModel1*.dll

「B. 2　ユーザ・モデルの作成」で作成したユーザ・モデルのバイナリ・ファイル（DLL 形式）です。ファイル名は任意に指定することができます。

なお，ファイルのパスは，シミュレータ・コンフィギュレーション・ファイルが存在するフォルダからの相対パス，または絶対パスを指定します。

注意　パスの指定に，半角スペースを使用することはできません。
半角スペースを使用した場合，ユーザ・モデルを生成することはできません。

(e) *UserOption1*

UserModel1.dll に対するオプション文字列です。このオプションは MakeUserModel 関数の引数 "*option*" にそのまま渡されます。

(2) 端子の接続処理

```
wire1 =  Wire(1);                                          ---  (a)
wire1 += cpu.Port("PinName1");                             ---  (b)
wire1 += UserModel1.Port("UserPinName1");                  ---  (c)
```

(a) ワイヤの生成

Wire 関数を使用し，ワイヤ（端子同士を接続する線）を生成します。

Wire 関数の引数には，必ず "1" を指定してください。

なお，*wire1* は，生成したワイヤを表す変数です。変数名は任意に指定することができます。

(b) ワイヤと CPU の接続

ワイヤの一方を CPU の端子に接続します。

"*PinName1*" には，接続したい CPU の外部端子名を大文字アルファベットで指定します（小文字アルファベットは使用不可）。必ず "" で囲んで指定してください。

(c) ワイヤとユーザ・モデルの接続

ワイヤのもう一方をユーザ・モデルの端子に接続します。

"*UserPinName1*" には，接続したいユーザ・モデルの端子名（MakeUserModel 関数の中で生成した端子名）を指定します。必ず "" で囲んで指定してください。

なお，ユーザ・モデルの複数の端子を同一のワイヤに接続する場合は，この行を追加します。

(3) 外部バスの接続処理

```
extbus1 =  BUS(n);                                         ---  (a)
extbus1 += cpu.BusMasterIF("EXTBUS");                      ---  (b)
extbus1 += UserModel1.BusSlaveIF("UserExtbusName1");       ---  (c)
```

(a) バスの生成

BUS 関数を使用し，バスを生成します。

BUS 関数の引数 *n* は，データ・バス・ビット幅を示し，8/16/32 のいずれかを指定することができます。

なお，*extbus1* は，生成したバスを表す変数です。変数名は任意に指定することができます。

(b) バスと CPU の接続

バスの一方を CPU の外部バス・マスタに接続します。

引数には，外部バス・マスタ "EXTBUS" を指定してください。

(c) バスとユーザ・モデルの接続

バスのもう一方をユーザ・モデルの外部バスに接続します。

"*UserExtbusName1*" には，接続したいユーザ・モデルの外部バス名（MakeUserModel 関数の中で生成した外部バス名）を指定します。必ず "" で囲んで指定してください。

なお，ユーザ・モデルの複数の外部バスを接続する場合は，この行を追加します。

(4) その他の処理

上記の処理以外に，ユーザ・オープン・インタフェースを動作させるためには，メイン・クロック通知端子とリセット通知端子の接続が必要となります。

```
clock1 =  Wire(1);                                                    ---  (a)
clock1 += cpu.DebuggerPseudoPort("debugger_pseudo_pin_main_clkout");  ---  (b)
clock1 += UserModel1.Port("gui_pseudo_pin_clock_notice");             ---  (c)
reset1 =  Wire(1);                                                    ---  (d)
reset1 += cpu.DebuggerPseudoPort("debugger_pseudo_pin_reset_notice"); ---  (e)
reset1 += UserModel1.Port("gui_pseudo_pin_reset_notice");             ---  (f)
```

(a) ワイヤの生成

Wire 関数を使用し，ワイヤ（端子同士を接続する線）を生成します。

Wire 関数の引数には，必ず“1”を指定してください。

なお，*clock1* は，生成したワイヤを表す変数です。変数名は任意に指定することができます。

(b) ワイヤとメイン・クロック通知端子の接続

ワイヤの一方をシミュレータ GUI のメイン・クロック通知端子に接続します。

引数には，"debugger_pseudo_pin_main_clkout" を指定してください。

(c) ワイヤとユーザ・モデルの接続

ワイヤのもう一方をユーザ・モデルの端子に接続します。

引数には，"gui_pseudo_pin_clock_notice" を指定してください。

(d) ワイヤの生成

Wire 関数を使用し，ワイヤ（端子同士を接続する線）を生成します。

Wire 関数の引数には，必ず“1”を指定してください。

なお，*reset1* は，生成したワイヤを表す変数です。変数名は任意に指定することができます。

(e) ワイヤとリセット通知端子の接続

ワイヤの一方をシミュレータ GUI のリセット通知端子に接続します。

引数には，"debugger_pseudo_pin_reset_notice" を指定してください。

(f) ワイヤとユーザ・モデルの接続

ワイヤのもう一方をユーザ・モデルの端子に接続します。

引数には，"gui_pseudo_pin_reset_notice" を指定してください。

B. 3. 2　シミュレータ・コンフィギュレーション・ファイルの記述例

次に，シミュレータ・コンフィギュレーション・ファイルの記述例を示します。

この例では，次表に示す接続処理を行っています。

接続の種類	CPU		ユーザ・モデル（SampleModel.dll）	
端子	"P00/INTP0"	P00 端子	"P00"	P00 の操作端子
	"P30/TXD1"	シリアルの出力端子	"RXD"	シリアルの入力端子
	"P31/RXD1"	シリアルの入力端子	"TXD"	シリアルの出力端子
外部バス	"EXTBUS"	外部バス・マスタ	"EXTBUS1"	外部バス・スレーブ 1
	"EXTBUS"	外部バス・マスタ	"EXTBUS2"	外部バス・スレーブ 2

```
cpu = CPU('a');
# ---------------------------------------------------------------
# SampleModel description
# ---------------------------------------------------------------

# Generate SampleModel.dll
model = Device("USEROPEN", "SampleModel.dll -a -b");

# Connect PIN (CPU.P00-MODEL.P00)
wire_P00 =  Wire(1);
wire_P00 += cpu.Port("P00/INTP0");
wire_P00 += model.Port("P00");

# Connect PIN (CPU.TXD1-MODEL.RXD)
wire_RXD =  Wire(1);
wire_RXD += cpu.Port("P30/TXD1");
wire_RXD += model.Port("RXD");

# Connect PIN (CPU.RXD1-MODEL.TXD)
wire_TXD =  Wire(1);
wire_TXD += cpu.Port("P31/RXD1");
wire_TXD += model.Port("TXD");

# Connect BUS (CPU.EXTBUS-MODEL.EXTBUS1)
extbus =  BUS(32);
extbus += cpu.BusMasterIF("EXTBUS");
extbus += model.BusSlaveIF("EXTBUS1");
extbus += model.BusSlaveIF("EXTBUS2");

# Connect Pseudo PIN
wire_clock =  Wire(1);
wire_clock += cpu.DebuggerPseudoPort("debugger_pseudo_pin_main_clkout");
wire_clock += model.Port("gui_pseudo_pin_clock_notice");
wire_reset =  Wire(1);
wire_reset += cpu.DebuggerPseudoPort("debugger_pseudo_pin_reset_notice");
wire_reset += model.Port("gui_pseudo_pin_reset_notice");
```

B. 4　提供インタフェース関数

この節では，ユーザ・オープン・インタフェースとしてシミュレータ GUI が提供するインタフェース関数について説明します。

B. 4. 1　概　　要

次に，シミュレータ GUI が提供するインタフェース関数の一覧を示します。

表 B—2　提供インタフェース関数一覧

種類	関数名	機能概要
基本インタフェース関数	SuoSetInitCallback	初期化処理のコールバック登録
	SuoSetResetCallback	リセット処理のコールバック登録
	SuoGetMainClock	シミュレーションのメイン・クロック周期の取得
時間インタフェース関数	SuoCreateTimer	タイマ・インタフェースの生成
	SuoGetTimerHandle	タイマ・インタフェースのハンドルの取得
	SuoSetTimer	周期タイマの設定
	SuoKillTimer	周期タイマの停止
	SuoSetNotifyTimerCallback	タイマの時間通知処理のコールバック登録
端子インタフェース関数	SuoCreatePin	端子インタフェースの生成
	SuoGetPinHandle	端子インタフェースのハンドルの取得
	SuoOutputDigitalPin	端子のデジタル・データの出力
	SuoOutputAnalogPin	端子のアナログ・データ出力
	SuoOutputHighImpedance	端子のハイ・インピーダンス出力
	SuoSetInputDigitalPinCallback	端子のデジタル値入力処理のコールバック登録
	SuoSetInputAnalogPinCallback	端子のアナログ値入力処理のコールバック登録
	SuoSetInputHighImpedanceCallback	端子のハイ・インピーダンス状態通知処理のコールバック登録
外部バス・インタフェース関数注	SuoCreateExtbus	外部バス・インタフェースの生成
	SuoGetExtbusHandle	外部バス・インタフェースのハンドルの取得
	SuoSetReadExtbusCallback	外部バスのリード・アクセスのコールバック登録
	SuoSetWriteExtbusCallback	外部バスのライト・アクセスのコールバック登録

種類	関数名	機能概要
シリアル・インタフェース関数	SuoCreateSerialUART	シリアル・インタフェース（UART タイプ）の生成
	SuoCreateSerialCSI	シリアル・インタフェース（CSI タイプ）の生成
	SuoGetSerialHandle	シリアル・インタフェースのハンドルの取得
	SuoSetSerialParameterUART	シリアルのパラメータ（UART タイプ）の設定
	SuoSetSerialParameterCSI	シリアルのパラメータ（CSI タイプ）の設定
	SuoGetSerialParameterUART	シリアルのパラメータ（UART タイプ）の取得
	SuoGetSerialParameterCSI	シリアルのパラメータ（CSI タイプ）の取得
	SuoSendSerialData	シリアル・データの送信（1 データ）
	SuoSendSerialDataList	シリアル・データの送信（複数データ）
	SuoSendSerialFile	シリアル・データの送信（シリアル送信データ・ファイル）
	SuoSetNotifySentSerialCallback	シリアルの送信完了通知のコールバック登録
	SuoSetReceiveSerialCallback	シリアルの受信のコールバック登録
信号出力器インタフェース関数	SuoCreateWave	信号出力器インタフェースの生成
	SuoGetWaveHandle	信号出力器インタフェースのハンドルの取得
	SuoSendWaveFile	信号出力器による信号データの送信
	SuoSetNotifySentWaveCallback	信号出力器の送信完了通知のコールバック登録

注　外部バス・インタフェース関数を使用する場合は，使用する外部メモリ領域を，メモリ・マッピング ダイアログ上の［メモリ種別］エリアにおいて［ターゲット・メモリ領域］に指定する必要があります。

B. 4. 2　基本インタフェース関数

基本インタフェース関数には，次のものがあります。

関数名	機能概要
SuoSetInitCallback	初期化処理のコールバック登録
SuoSetResetCallback	リセット処理のコールバック登録
SuoGetMainClock	シミュレーションのメイン・クロック周期の取得

SuoSetInitCallback

初期化処理のコールバック登録を行います。

注意　この関数を MakeUserModel 関数内で呼び出さない場合，コールバック関数は機能しません。

[指定形式]

```
#include    "suo.h"
void    SuoSetInitCallback(SuoInitCallback func);
```

[引数]

引数	説明
func	初期化処理を行うユーザ定義関数へのポインタ（「InitFunc」参照）

[戻り値]

なし

[詳細説明]

- 初期化処理を行うユーザ定義関数を登録します。
- ここで登録された関数は，シミュレータ GUI 起動時の 1 回のみ呼び出されます。
- *func* に NULL を指定した場合は，登録を解除します。

[使用例]

```
#include    "suo.h"
void InitFunc(void);

/* MakeUserModel */
SuoUserEntry void MakeUserModel(const char *option)
{
    .........
    SuoSetInitCallback(InitFunc);        /* Set initialize function */
}

/* Initialize function */
void InitFunc(void){
    .........
}
```

SuoSetResetCallback

リセット処理のコールバック登録を行います。

[指定形式]

```
#include    "suo.h"
void    SuoSetResetCallback(SuoResetCallback func);
```

[引数]

引数	説明
func	リセット処理を行うユーザ定義関数へのポインタ（「ResetFunc」参照）

[戻り値]

なし

[詳細説明]

- リセット処理を行うユーザ定義関数を登録します。
- ここで登録された関数は，CPU リセット時に呼び出されます。
- *func* に NULL を指定した場合は，登録を解除します。

[使用例]

```
#include    "suo.h"
void ResetFunc(void);

void func1(void)
{
    .........
    SuoSetResetCallback(ResetFunc);        /* Set reset function */
}

/* Reset function */
void ResetFunc(void){
    .........
}
```

SuoGetMainClock

シミュレーションのメイン・クロック周期を取得します。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoGetMainClock(unsigned long* time);
```

[引数]

引数	説明
time	メイン・クロックの周期（単位：ピコ秒）の格納場所

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 現在動作中のシミュレーション環境のメイン・クロック周期を取得します。

[使用例]

```
#include    "suo.h"
unsigned long time;

void func1(void)
{
    .........
    SuoGetMainClock(&time);         /* Get main clock */
}
```

B. 4. 3　時間インタフェース関数

時間インタフェース関数には，次のものがあります。

関数名	機能概要
SuoCreateTimer	タイマ・インタフェースの生成
SuoGetTimerHandle	タイマ・インタフェースのハンドルの取得
SuoSetTimer	周期タイマの設定
SuoKillTimer	周期タイマの停止
SuoSetNotifyTimerCallback	タイマの時間通知処理のコールバック登録

SuoCreateTimer

タイマ・インタフェースを生成します。

注意　この関数は，**MakeUserModel** 関数内でのみ呼び出すことができます。他のタイミングで呼び出した場合はエラーになります。

[指定形式]

```
#include    "suo.h"
int     SuoCreateTimer(const char* timerName, SuoHandle* handle);
```

[引数]

引数	説明
timerName	タイマの名前
handle	タイマ・インタフェースのハンドルの格納場所

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- タイマ・インタフェースを生成します。
- 生成したタイマ・インタフェースは，*timerName* で指定した名前と関連付けられます。
- この関数が正常終了すると，生成したタイマ・インタフェースのハンドルを取得できます。後のタイマ・インタフェースに対する制御は，ここで取得したハンドルを指定して行います。
  なお，ハンドルは，SuoGetTimerHandle 関数により取得することもできます。

［使用例］

```
#include    "suo.h"
SuoHandle hTim1;

SuoUserEntry void MakeUserModel(const char *option)
{
    .........
    SuoCreateTimer("TIM1", &hTim1);        /* Create "TIM1" */
}
```

SuoGetTimerHandle

タイマ・インタフェースのハンドルを取得します。

[指定形式]

```
#include    "suo.h"
SuoHandle   SuoGetTimerHandle(const char* timerName);
```

[引数]

引数	説明
timerName	タイマの名前

[戻り値]

マクロ	説明
タイマ・インタフェースのハンドル	正常終了
NULL	不正終了

[詳細説明]

- 指定したタイマ・インタフェースのハンドルを取得します。
- *timerName* には，SuoCreateTimer 関数で指定した名前を指定してください（異なる名前を指定した場合は“NULL ”が返ります）。

[使用例]

```
#include    "suo.h"
SuoHandle hTim1;

void func1(void)
{
    .........
    hTim1 = SuoGetTimerHandle("TIM1");        /* Get handle of "TIM1" */
}
```

SuoSetTimer

周期タイマの設定を行います。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoSetTimer(SuoHandle handle, int timeUnit, unsigned long timeValue);
```

[引数]

引数	説明
handle	タイマ・インタフェースのハンドル
timeUnit	次のいずれかの時間単位 - SUO_MAINCLK：　メイン・クロック周期の単位 - SUO_USEC：　マイクロ秒単位
timeValue	タイマ周期時間

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 指定したタイマ・インタフェースに対して周期タイマの設定を行います。
- 周期時間は，*timeUnit* の単位で *timeValue* の値で指定します（*timeValue* に 0 は指定できません）。
- この関数の呼び出し直後からタイマは動作を開始し，SuoKillTimer 関数でタイマ動作を停止するまでタイマは動作し続けます。
- SuoSetNotifyTimerCallback 関数でタイマ通知関数が登録されていれば，1 周期毎にタイマ通知関数が呼び出されます。
- 現在動作しているタイマに対してこの関数を呼び出した場合は，タイマはリセットされ新たに指定した周期時間で動作を開始します。

［使用例］

```
#include    "suo.h"
SuoHandle hTim1;

void func1(void)
{
.........
    SuoSetTimer(hTim1, SUO_USEC, 20);     /* Invoke 20us cyclic timer */
}
```

SuoKillTimer

周期タイマの停止を行います。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoKillTimer(SuoHandle handle);
```

[引数]

引数	説明
handle	タイマ・インタフェースのハンドル

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 指定したタイマ・インタフェースに対して周期タイマの停止を行います。
- タイマが動作している場合はタイマ動作を停止し，タイマが停止している場合は何もしません（エラーにはなりません）。

[使用例]

```
#include    "suo.h"
SuoHandle hTim1;

void func1(void)
{
    .........
    SuoKillTimer(hTim1);        /* Stop timer */
}
```

SuoSetNotifyTimerCallback

タイマの時間通知処理のコールバック登録を行います。

[指定形式]

```
#include    "suo.h"
int     SuoSetNotifyTimerCallback(SuoHandle handle, SuoNotifyTimerCallback func);
```

[引数]

引数	説明
handle	タイマ・インタフェースのハンドル
func	タイマの時間通知を行うユーザ定義関数へのポインタ（「NotifyTimerFunc」参照）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- タイマの時間通知処理を行うユーザ定義関数を登録します。
- ここで登録された関数は，指定したタイマ・インタフェースのタイマ周期毎に呼び出されます。
- *func* に NULL を指定した場合は，登録を解除します。

［使用例］

```
#include    "suo.h"
void NotifyTimerFunc(SuoHandle handle);
SuoHandle hTim1;
void func1(void)
{
    .........
    SuoSetNotifyTimerCallback(hTim1, NotifyTimerFunc);    /* Set notify-timer function */
}

/* Notify-timer function */
void NotifyTimerFunc(SuoHandle handle)
{
    .........
}
```

B. 4. 4　端子インタフェース関数

端子インタフェース関数には，次のものがあります。

関数名	機能概要
SuoCreatePin	端子インタフェースの生成
SuoGetPinHandle	端子インタフェースのハンドルの取得
SuoOutputDigitalPin	端子のデジタル・データの出力
SuoOutputAnalogPin	端子のアナログ・データの出力
SuoOutputHighImpedance	端子のハイ・インピーダンス出力
SuoSetInputDigitalPinCallback	端子のデジタル・データ入力処理のコールバック登録
SuoSetInputAnalogPinCallback	端子のアナログ・データ入力処理のコールバック登録
SuoSetInputHighImpedanceCallback	端子のハイ・インピーダンス状態通知処理のコールバック登録

SuoCreatePin

端子インタフェースを生成します。

注意　この関数は，MakeUserModel 関数内でのみ呼び出すことができます。他のタイミングで呼び出した場合はエラーになります。

[指定形式]

```
#include    "suo.h"
int     SuoCreatePin(const char* pinName, SuoHandle* handle);
```

[引数]

引数	説明
pinName	端子の名前
handle	端子インタフェースのハンドルの格納場所

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 端子インタフェースを生成します。
- 生成した端子インタフェースは，*pinName* で指定した名前と関連付けられます。また，*pinName* で指定した端子が生成されます。
- この関数が正常終了すると，生成した端子インタフェースのハンドルを取得できます。後の端子インタフェースに対する制御は，ここで取得したハンドルを指定して行います。
  なお，ハンドルは，SuoGetPinHandle 関数により取得することもできます。

［使用例］

```
#include    "suo.h"
SuoHandle hPinP00;
SuoHandle hPinABC;

SuoUserEntry void MakeUserModel(const char *option)
{
    .........
    SuoCreatePin("P00", &hPinP00);        /* Create "P00" */
    SuoCreatePin("ABC", &hPinABC);        /* Create "ABC" */
}
```

SuoGetPinHandle

端子インタフェースのハンドルを取得します。

[指定形式]

```
#include    "suo.h"
SuoHandle   SuoGetPinHandle(const char* pinName);
```

[引数]

引数	説明
pinName	端子の名前

[戻り値]

マクロ	説明
端子インタフェースのハンドル	正常終了
NULL	不正終了

[詳細説明]

- 指定した端子インタフェースのハンドルを取得します。
- *pinName* には SuoCreatePin 関数で指定した名前を指定してください（異なる名前を指定した場合は NULL が返ります）。

[使用例]

```
#include    "suo.h"
SuoHandle hPinP00;

void func1(void)
{
    .........
    hPinP00 = SuoGetPinHandle("P00");        /* Get handle of "P00" */
}
```

SuoOutputDigitalPin

端子インタフェースに対し，デジタル・データを出力します。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoOutputDigitalPin(SuoHandle handle, int pinValue);
```

[引数]

引数	説明
handle	端子インタフェースのハンドル
pinValue	次のいずれかの出力値（デジタル・データ） - SUO_HIGH (=1)：　HIGH 値 - SUO_LOW (=0)：　LOW 値

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 指定した端子インタフェースに対して，*pinValue* で指定したデジタル・データを出力します。
- アナログ・データの出力は，SuoOutputAnalogPin 関数を使用してください。

［使用例］

```
#include    "suo.h"
SuoHandle hPinP00;

void func1(void)
{
    .........
    SuoOutputDigitalPin(hPinP00, SUO_HIGH);        /* Output HIGH */
}
```

SuoOutputAnalogPin

端子インタフェースに対し，アナログ・データを出力します。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoOutputAnalogPin(SuoHandle handle, double pinValue);
```

[引数]

引数	説明
handle	端子インタフェースのハンドル
pinValue	出力値（アナログ・データ）（単位：V）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 指定した端子インタフェースに対して，*pinValue* で指定したアナログ・データを出力します。
- *pinValue* の単位は“V”で，浮動小数点データで指定してください。
- デジタル・データの出力は，SuoOutputDigitalPin 関数を使用してください。

[使用例]

```
#include    "suo.h"
SuoHandle hPinP00;

void func1(void)
{
    .........
    SuoOutputAnalogPin(hPinP00, 3.5);        /* Output 3.5V */
}
```

SuoOutputHighImpedance

端子インタフェースに対し，ハイ・インピーダンスを出力します。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoOutputHighImpedance(SuoHandle handle);
```

[引数]

引数	説明
handle	端子インタフェースのハンドル

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 指定したデジタル／アナログ端子インタフェースに対して，ハイ・インピーダンスを出力します。

[使用例]

```
#include    "suo.h"
SuoHandle hPinP00;

void func1(void)
{
    .........
    SuoOutputHighImpedance(hPinP00);        /* Output High Impedance */
}
```

SuoSetInputDigitalPinCallback

端子のデジタル・データ入力処理のコールバック登録を行います。

[指定形式]

```
#include    "suo.h"
int     SuoSetInputDigitalPinCallback(SuoHandle handle, SuoInputDigitalPinCallback func);
```

[引数]

引数	説明
handle	端子インタフェースのハンドル
func	端子のデジタル・データ入力処理を行うユーザ定義関数へのポインタ（「InputDigitalPinFunc」参照）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 端子のデジタル・データ入力処理を行うユーザ定義関数を登録します。
- ここで登録された関数は，指定した端子に信号が入力された時に呼び出されます。
- *func* に NULL を指定した場合は，登録を解除します。

[使用例]

```
#include    "suo.h"
void InputDigitalPinFunc(SuoHandle handle, int pinValue);
SuoHandle hPinP00;

void func1(void)
{
    .........
    SuoSetInputDigitalPinCallback(hPinP00, InputDigitalPinFunc); /* Set input-digital-pin function */
}

/* Input-digital-pin function */
void InputDigitalPinFunc(SuoHandle handle, int pinValue)
{
    .........
}
```

SuoSetInputAnalogPinCallback

端子のアナログ・データ入力処理のコールバック登録を行います。

[指定形式]

```
#include    "suo.h"
int     SuoSetInputAnalogPinCallback(SuoHandle handle, SuoInputAnalogPinCallback func);
```

[引数]

引数	説明
handle	端子インタフェースのハンドル
func	端子のアナログ・データ入力処理を行うユーザ定義関数へのポインタ（「InputAnalogPinFunc」参照）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 端子のアナログ・データ入力処理を行うユーザ定義関数を登録します。
- ここで登録された関数は，指定した端子に信号が入力された時に呼び出されます。
- *func* に NULL を指定した場合は，登録を解除します。

［使用例］

```
#include    "suo.h"
void InputAnalogPinFunc(SuoHandle handle, double pinValue);
SuoHandle hPinP00;

void func1(void)
{
    .........
    SuoSetInputAnalogPinCallback(hPinP00, InputAnalogPinFunc); /* Set input-analog-pin function */
}

/* Input-analog-pin function */
void InputAnalogPinFunc(SuoHandle handle, double pinValue)
{
    .........
}
```

SuoSetInputHighImpedanceCallback

端子のハイ・インピーダンス状態通知処理のコールバック登録を行います。

[指定形式]

```
#include    "suo.h"
int  SuoSetInputHighImpedanceCallback(SuoHandle handle, SuoInputHighImpedanceCallback func);
```

[引数]

引数	説明
handle	端子インタフェースのハンドル
func	接続されたすべての端子がハイ・インピーダンス状態になった際の処理を行う ユーザ定義関数へのポインタ（「InputHighImpedanceFunc」参照）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- デジタル／アナログ端子に接続されたすべての端子がハイ・インピーダンス状態になった際の処理を行うユーザ定義関数を登録します。
- *func* に NULL を指定した場合は，登録を解除します。

［使用例］

```
#include    "suo.h"
void InputHighImpedanceFunc(SuoHandle handle);
SuoHandle hPinP00;

void func1(void)
{
    .........
    SuoSetInputHighImpedanceCallback(hPinP00,InputHighImpedanceFunc);
                                                    /* Set input-high-impedance function */
}

/* Input-high-impedance function */
void InputHighImpedanceFunc(SuoHandle handle)
{
    .........
}
```

B. 4. 5　外部バス・インタフェース関数

外部バス・インタフェース関数には，次のものがあります。

関数名	機能概要
SuoCreateExtbus	外部バス・インタフェースの生成
SuoGetExtbusHandle	外部バス・インタフェースのハンドルの取得
SuoSetReadExtbusCallback	外部バスのリード・アクセスのコールバック登録
SuoSetWriteExtbusCallback	外部バスのライト・アクセスのコールバック登録

注意　外部バス・インタフェース関数を使用する場合は，使用する外部メモリ領域を，メモリ・マッピング ダイアログ上の［メモリ種別］エリアにおいて［ターゲット・メモリ領域］に指定する必要があります。

SuoCreateExtbus

外部バス・インタフェースを生成します。

注意 1.　この関数は，MakeUserModel 関数内でのみ呼び出すことができます。他のタイミングで呼び出した場合はエラーになります。

2.　外部バス・インタフェース関数を使用する場合は，使用する外部メモリ領域を，メモリ・マッピング ダイアログ上の［メモリ種別］エリアにおいて［ターゲット・メモリ領域］に指定する必要があります。

[指定形式]

```
#include    "suo.h"

int     SuoCreateExtbus(const char* extbusName, unsigned long addr, unsigned long size,
SuoHandle* handle);
```

[引数]

引数	説明
extbusName	外部バスの名前
addr	外部メモリ領域の先頭アドレス
size	外部メモリ領域のサイズ
handle	外部バス・インタフェースのハンドルの格納場所

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 外部バス・インタフェースを生成します。
- 生成した外部バス・インタフェースは，*extbusName* で指定した名前と関連付けられます。
- この関数が正常終了すると，生成した外部バス・インタフェースのハンドルを取得できます。後の外部バス・インタフェースに対する制御は，ここで取得したハンドルを指定して行います。
  なお，ハンドルは，SuoGetExtbusHandle 関数により取得することもできます。

［使用例］

```
#include    "suo.h"
SuoHandle hExtbus1;

SuoUserEntry void MakeUserModel(const char *option)
{
    .........
    SuoCreateExtbus("EXTBUS1", 0x200000, 0x1000, &hExtbus1);    /* Create "EXTBUS1" */
}
```

SuoGetExtbusHandle

外部バス・インタフェースのハンドルを取得します。

[指定形式]

```
#include    "suo.h"
SuoHandle   SuoGetExtbusHandle(const char* extbusName);
```

[引数]

引数	説明
extbusName	外部バスの名前

[戻り値]

マクロ	説明
外部バス・インタフェースのハンドル	正常終了
NULL	不正終了

[詳細説明]

- 指定した外部バス・インタフェースのハンドルを取得します。
- *extbusName* には SuoCreateExtbus 関数で指定した名前を指定してください（異なる名前を指定した場合は NULL が返ります）。

[使用例]

```
#include    "suo.h"
SuoHandle hExtbus1;

void func1(void)
{
    .........
    hExtbus1 = SuoGetExtbusHandle("EXTBUS1");    /* Get handle of "EXTBUS1" */
}
```

SuoSetReadExtbusCallback

外部バスのリード・アクセス処理のコールバック登録を行います。

[指定形式]

```
#include    "suo.h"
int     SuoSetReadExtbusCallback(SuoHandle handle, SuoReadExtbusCallback func);
```

[引数]

引数	説明
handle	外部バス・インタフェースのハンドル
func	外部バスのリード・アクセス処理を行うユーザ定義関数へのポインタ（「ReadExtbusFunc」参照）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 外部バスのリード・アクセス処理を行うユーザ定義関数を登録します。
- ここで登録された関数は，指定した外部バス（登録したアドレス範囲）に対してリード要求が発生した時に呼び出されます。
- *func* に NULL を指定した場合は，登録を解除します。

[使用例]

```
#include    "suo.h"
void ReadExtbusFunc(SuoHandle handle, unsigned long addr, int accessSize, unsigned char data[]);
SuoHandle hExtbus1;

void func1(void)
{
    .........
    SuoSetReadExtbusCallback(hExtbus1, ReadExtbusFunc);    /* Set read-external-bus function */
}

/* Read-external-bus function */
void ReadExtbusFunc(SuoHandle handle, unsigned long addr, int accessSize, unsigned char data[])
{
    .........
}
```

SuoSetWriteExtbusCallback

外部バスのライト・アクセス処理のコールバック登録を行います。

[指定形式]

```
#include    "suo.h"
int     SuoSetWriteExtbusCallback(SuoHandle handle, SuoWriteExtbusCallback func);
```

[引数]

引数	説明
handle	外部バス・インタフェースのハンドル
func	外部バスのライト・アクセス処理を行うユーザ定義関数へのポインタ（「WriteExtbusFunc」参照）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 外部バスのライト・アクセス処理を行うユーザ定義関数を登録します。
- ここで登録された関数は，指定した外部バス（登録したアドレス範囲）に対してライト要求が発生した時に呼び出されます。
- *func* に NULL を指定した場合は，登録を解除します。

[使用例]

```
#include    "suo.h"
void WriteExtbusFunc(SuoHandle handle, unsigned long addr, int accessSize, const unsigned char data[]);
SuoHandle hExtbus1;

void func1(void)
{
    .........
    SuoSetWriteExtbusCallback(hExtbus1, WriteExtbusFunc);  /* Set write-external-bus function */
}

/* Write-external-bus function */
void WriteExtbusFunc(SuoHandle handle, unsigned long addr, int accessSize, const unsigned char data[])
{
    .........
}
```

B. 4. 6　シリアル・インタフェース関数

シリアル・インタフェース関数には，次のものがあります。

関数名	機能概要
SuoCreateSerialUART	シリアル・インタフェース（UART タイプ）の生成
SuoCreateSerialCSI	シリアル・インタフェース（CSI タイプ）の生成
SuoGetSerialHandle	シリアル・インタフェースのハンドルの取得
SuoSetSerialParameterUART	シリアルのパラメータ（UART タイプ）の設定
SuoSetSerialParameterCSI	シリアルのパラメータ（CSI タイプ）の設定
SuoGetSerialParameterUART	シリアルのパラメータ（UART タイプ）の取得
SuoGetSerialParameterCSI	シリアルのパラメータ（CSI タイプ）の取得
SuoSendSerialData	シリアル・データの送信（1 データ）
SuoSendSerialDataList	シリアル・データの送信（複数データ）
SuoSendSerialFile	シリアル・データの送信（シリアル・ファイル）
SuoSetNotifySentSerialCallback	シリアルの送信完了通知のコールバック登録
SuoSetReceiveSerialCallback	シリアルの受信のコールバック登録

SuoCreateSerialUART

シリアル・インタフェース（UART タイプ）を生成します。

注意　この関数は，**MakeUserModel** 関数内でのみ呼び出すことができます。他のタイミングで呼び出した場合はエラーになります。

[指定形式]

```
#include    "suo.h"

int     SuoCreateSerialUART(const char* serialName, const char* pinNameTXD, const char*
pinNameRXD, SuoHandle* handle);
```

[引数]

引数	説明
serialName	シリアルの名前
pinNameTXD	シリアルで使用する送信データ用端子の名前
pinNameRXD	シリアルで使用する受信データ用端子の名前
handle	シリアル・インタフェースのハンドルの格納場所

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- シリアル・インタフェース（UART タイプ）を生成します。
- 生成したシリアル・インタフェースは，*sirialName* で指定した名前と関連付けられます。また，*pinNameTXD*，および *pinNameRXD* で指定した端子が生成されます。
- この関数が正常終了すると，生成したシリアル・インタフェースのハンドルを取得できます。後の外部バス・インタフェースに対する制御は，ここで取得したハンドルを指定して行います。
  なお，ハンドルは，SuoGetSerialHandle 関数により取得することもできます。

［使用例］

```
#include    "suo.h"
SuoHandle hUart1;

SuoUserEntry void MakeUserModel(const char *option)
{
    .........
    SuoCreateSerialUART("UART1", "TXD1", "RXD1", &hUart1);    /* Create "UART1" */
}
```

SuoCreateSerialCSI

シリアル・インタフェース（CSI タイプ）を生成します。

注意　この関数は，**MakeUserModel** 関数内でのみ呼び出すことができます。他のタイミングで呼び出した場合はエラーになります。

[指定形式]

```
#include    "suo.h"

int     SuoCreateSerialCSI(const char* serialName, const char* pinNameSO, const char*
pinNameSI, const char* pinNameSCK, SuoHandle* handle);
```

[引数]

引数	説明
serialName	シリアルの名前
pinNameSO	シリアルで使用する送信データ用端子の名前
pinNameSI	シリアルで使用する受信データ用端子の名前
pinNameSCK	シリアルで使用するクロック用端子の名前
handle	シリアル・インタフェースのハンドルの格納場所

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- シリアル・インタフェース（CSI タイプ）を生成します。
- 生成したシリアル・インタフェースは，*sirialName* で指定した名前と関連付けられます。また，*pinNameSO*，*pinNameSI*，および *pinNameSCK* で指定した端子が生成されます。
- この関数が正常終了すると，生成したシリアル・インタフェースのハンドルを取得できます。後の外部バス・インタフェースに対する制御は，ここで取得したハンドルを指定して行います。
  なお，ハンドルは，SuoGetSerialHandle 関数により取得することもできます。

［使用例］

```
#include    "suo.h"
SuoHandle hCsi1;

SuoUserEntry void MakeUserModel(const char *option)
{
    .........
    SuoCreateSerialCSI("CSI1", "SO1", "SI1", "SCK1", &hCsi1);    /* Create "CSI1" */
}
```

SuoGetSerialHandle

シリアル・インタフェースのハンドルを取得します。

[指定形式]

```
#include    "suo.h"
SuoHandle   SuoGetSerialHandle(const char* serialName);
```

[引数]

引数	説明
serialName	シリアルの名前

[戻り値]

マクロ	説明
シリアル・インタフェースのハンドル	正常終了
NULL	不正終了

[詳細説明]

- 指定したシリアル・インタフェースのハンドルを取得します。
- *sirialName* には，SuoCreateSerialUART 関数，または SuoCreateSerialCSI 関数で指定した名前を指定してください（異なる名前を指定した場合は NULL が返ります）。

[使用例]

```
#include    "suo.h"
SuoHandle hSerial1;

void func1(void)
{
    .........
    hSerial1 = SuoGetSerialHandle("SERIAL1");    /* Get handle of "SERIAL1" */
}
```

SuoSetSerialParameterUART

シリアルのパラメータ（UART タイプ）を設定します。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoSetSerialParameterUART(SuoHandle handle, const SuoSerialParameterUART* param);
```

[引数]

引数	説明
handle	シリアル・インタフェースのハンドル
param	シリアルのパラメータ（UART タイプ）の格納場所を示す SuoSerialParameterUART 構造体[注]へのポインタ

注　SuoSerialParameterUART 構造体の構成は次のとおりです。

```
typedef struct {
    unsigned long   baudrate;           /* ボー・レート値 */
    int             direction;          /* 転送方向 */
    int             dataLength;         /* データ・ビット長 */
    int             stopLength;         /* ストップ・ビット長 */
    int             parity;             /* パリティ */
} SuoSerialParameterUART;
```

パラメータ（UART タイプ）	値	説明
ボー・レート	*ボー・レート値*	単位：bps
転送方向	SUO_MSBFIRST	MSB ファースト
	SUO_LSBFIRST	LSB ファースト
データ・ビット長	1 ～ 32	－
ストップ・ビット長	1，または 2	－
パリティ	SUO_NONEPARITY	パリティなし
	SUO_ZEROPARITY	0 パリティ （送信時：パリティ 0，受信時：パリティ・チェックなし）
	SUO_ODDPARITY	奇数パリティ
	SUO_EVENPARITY	偶数パリティ

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 指定したシリアル・インタフェースに対してシリアル動作に関するパラメータ（UART タイプ）を設定します。
  デフォルトの設定値は次の通りです。
  - ボー・レート：　9600 bps
  - 転送方向：　LSB ファースト
  - データ・ビット長：　7 ビット
  - ストップ・ビット長：　1 ビット
  - パリティ：　なし

[使用例]

```
#include    "suo.h"
SuoHandle hUart1;

void func1(void)
{
    SuoSerialParameterUART param;
    .........
    param.baudrate   = 19200;                    /* 19200 bps */
    param.direction  = SUO_LSBFIRST;             /* LSB First */
    param.dataLength = 8;                        /* databit 8 bit */
    param.stopLength = 1;                        /* stopbit 1 bit */
    param.parity     = SUO_EVENPARITY;           /* even parity */
    SuoSetSerialParameterUART(hUart1, &param);   /* Set parameter of UART1 */
}
```

SuoSetSerialParameterCSI

シリアルのパラメータ（CSI タイプ）を設定します。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoSetSerialParameterCSI(SuoHandle handle, const SuoSerialParameterCSI* param);
```

[引数]

引数	説明
handle	シリアル・インタフェースのハンドル
param	シリアルのパラメータ（CSI タイプ）の格納場所を示す SuoSerialParameterCSI 構造体注へのポインタ

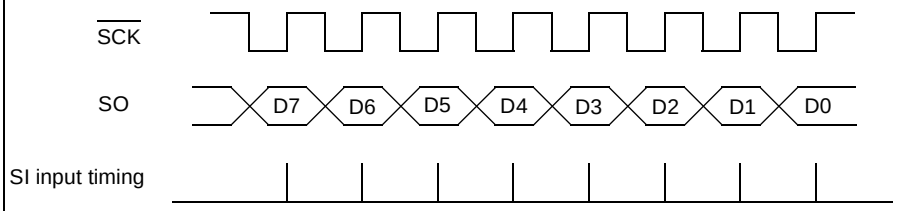
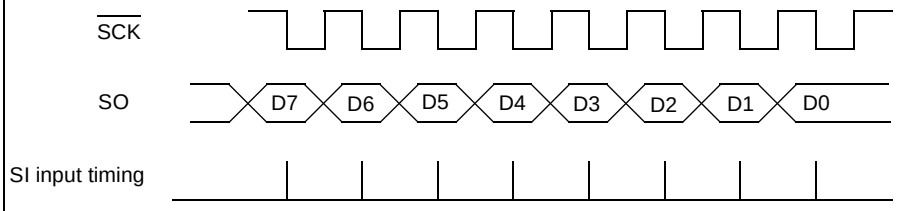
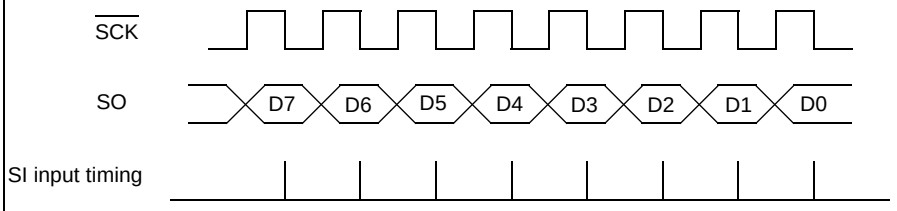
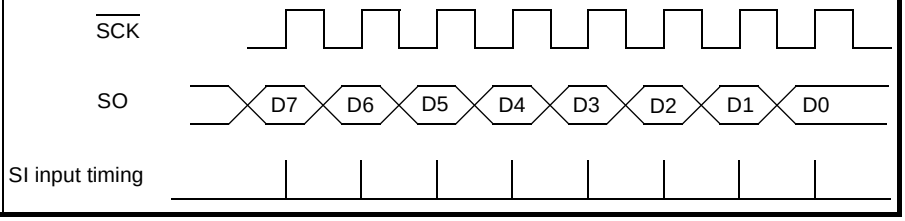
注　SuoSerialParameterCSI 構造体の構成は次のとおりです。

```
typedef struct {
    int             mode;           /* 動作モード */
    unsigned long   frequency;      /* 転送クロックの周波数 */
    int             phase;          /* 位相 */
    int             direction;      /* 転送方向 */
    int             datalength;     /* データ・ビット長 */
} SuoSerialParameterCSI;
```

パラメータ（CSI タイプ）	値	説明	
動作モード	SUO_MASTER	マスタ動作	
	SUO_SLAVE	スレーブ動作	
転送クロックの周波数	*周波数*	単位：Hz ただし，マスタ動作の場合は“0”の指定不可	
位相	0	通常の位相	表 B—3 参照
	SUO_PRECEDEDATA	データ先行出力	
	SUO_REVERSECLOCK	クロック反転	
	SUO_PRECEDEDATA | SUO_REVERSECLOCK	データ先行出力，およびクロック反転の両方を指定	
転送方向	SUO_MSBFIRS	MSB ファースト	
	SUO_LSBFIRST	LSB ファースト	

パラメータ（CSI タイプ）	値	説明
データ・ビット長	1 ～ 32	－

表 B—3　CSI 位相タイプ（SuoSetSerialParameterCSI 関数）

値	位相
0	
SUO_PRECEDEDATA	
SUO_REVERSECLOCK	
SUO_PRECEDEDATA| SUO_REVERSECLOCK	

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 指定したシリアル・インタフェースに対してシリアル動作に関するパラメータ（CSI タイプ）を設定します。デフォルトの設定値は次の通りです。
  - 動作モード：　スレーブ動作
  - 転送クロック：　0
  - 位相：　通常の位相
  - 転送方向：　MSB ファースト
  - データ・ビット長：　8 ビット
- CSI をマスタ動作で使用する場合，受信を行うためにはダミー・データの送信を行う必要があります（CSI はマスタ側がクロックを出力することで送受信を行う通信方式であるため）。

備考　CSI の通信が行われていない状況で，タイミングチャート ウインドウで CSI の端子波形を確認した場合，次のように本来の期待値と異なったレベルが観測されます。
なお，通信が開始すると期待通りのレベルになるため，実際の動作には影響しません。

- SCK 端子 (スレーブ動作時)：　ハイ・レベル（本来はハイ・インピーダンス）
- SO 端子：　ハイ・レベル（本来はロー・レベル）

[使用例]

```
#include    "suo.h"

SuoHandle hCsi1;
void func1(void)
{
    SuoSerialParameterCSI param;
    .........
    param.mode          = SUO_SLAVE;            /* slave */
    param.frequency     = 1000000;              /* 1MHz */
    param.phase         = 0;                    /* normal */
    param.direction     = SUO_LSBFIRST;         /* LSB First */
    param.dataLength    = 8;                    /* databit 8bit */
    SuoSetSerialParameterCSI(hCsi1, &param);    /* Set parameter of CSI1 */
}
```

SuoGetSerialParameterUART

シリアルのパラメータ（UART タイプ）を取得します。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int      SuoGetSerialParameterUART(SuoHandle handle, SuoSerialParameterUART* param);
```

[引数]

引数	説明
handle	シリアル・インタフェースのハンドル
param	シリアルのパラメータ（UART タイプ）の格納場所を示す SuoSerialParameterUART 構造体[注]へのポインタ

注　SuoSerialParameterUART 構造体についての詳細は，SuoSetSerialParameterUART 関数を参照してください。

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 指定したシリアル・インタフェースに対して，シリアル動作に関するパラメータ（UART タイプ）を取得します。

［使用例］

```
#include    "suo.h"
SuoHandle hUart1;

void func1(void)
{
    SuoSerialParameterUART param;
    .........
    SuoGetSerialParameterUART(hUart1, &param);    /* Get parameter of UART1 */
    .........
}
```

SuoGetSerialParameterCSI

シリアルのパラメータ（CSI タイプ）を取得します。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoGetSerialParameterCSI(SuoHandle handle, SuoSerialParameterCSI* param);
```

[引数]

引数	説明
handle	シリアル・インタフェースのハンドル
param	シリアルのパラメータ（CSI タイプ）の格納場所を示す SuoSerialParameterCSI 構造体注へのポインタ

注　SuoSerialParameterCSI 構造体についての詳細は，SuoSetSerialParameterCSI 関数を参照してください。

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 指定したシリアル・インタフェースに対して，シリアル動作に関するパラメータ（CSI タイプ）を取得します。

［使用例］

```
#include    "suo.h"
SuoHandle hCsi1;

void func1(void)
{
    SuoSerialParameterCSI param;
    .........
    SuoGetSerialParameterCSI(hCsi1, &param);    /* Get parameter of CSI1 */
    .........
}
```

SuoSendSerialData

シリアル・データ（1 データ）の送信を行います。

注意 1.　UART の連続送信を行う場合は，必ず SuoSendSerialDataList 関数（複数データ用シリアル送信関数）を使用してください。

2.　この関数を NotifySentSerialFunc 関数（シリアル送信完了のコールバック関数）で呼び出した場合，送信開始が UART のボー・レート半周期分遅れます。

3.　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoSendSerialData(SuoHandle handle, unsigned long data);
```

[引数]

引数	説明
handle	シリアル・インタフェースのハンドル
data	送信データ（1 データ）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 1 つのシリアル・データの送信を開始します。
- シリアル・データの送信が完了するまで時間がかかります。送信完了のタイミングを通知する場合は，SuoSetNotifySentSerialCallback 関数で送信完了通知関数を設定してください。
- 現在送信中のシリアル・インタフェースに対してこの関数を呼び出した場合はエラーとなります。

［使用例］

```
#include    "suo.h"
SuoHandle hSerial1;

void func1(void)
{
    .........
    SuoSendSerialData(hSerial1, 0x80);    /* Send 0x80 */
}
```

SuoSendSerialDataList

シリアル・データ（複数データ）の送信を行います。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoSendSerialDataList(SuoHandle handle, long count, unsigned long dataList[]);
```

[引数]

引数	説明
handle	シリアル・インタフェースのハンドル
count	送信データの個数（1 ～ 32767）
dataList[]	送信データ（配列によりデータ個数分指定）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 複数個のシリアル・データの送信を開始します。
- シリアル・データの送信が完了するまで時間がかかります。送信完了のタイミングを通知する場合は，SuoSetNotifySentSerialCallback 関数で送信完了通知関数を設定してください。
- 現在送信中のシリアル・インタフェースに対してこの関数を呼び出した場合はエラーとなります。

［使用例］

```
#include    "suo.h"
SuoHandle hSerial1;

void func1(void)
{
    unsigned long dataList[6] = {0x73, 0x65, 0x72, 0x69, 0x61, 0x6c};
    .........
    SuoSendSerialDataList(hSerial1, 6, dataList);        /* Send dataList */
}
```

SuoSendSerialFile

シリアル・データ（シリアル送信データ・ファイル）の送信を行います。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoSendSerialFile(SuoHandle handle, const char* serialFile);
```

[引数]

引数	説明
handle	シリアル・インタフェースのハンドル
serialFile	シリアル ウインドウで編集したのち保存したシリアル送信データ・ファイル名 なお，ファイル名を相対パスで指定した場合，ユーザ・モデル（*UserModel*.dll）のパスからの相対として扱われます。

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- シリアル ウインドウで編集したのち保存したシリアル送信データ・ファイル（*.ser）に記録されているシリアル・データの送信を開始します。
- シリアル・データの送信が完了するまで時間がかかります。送信完了のタイミングを通知する場合は，SuoSetNotifySentSerialCallback 関数で送信完了通知関数を設定してください。
- 現在送信中のシリアル・インタフェースに対してこの関数を呼び出した場合はエラーとなります。

［使用例］

```
#include    "suo.h"
SuoHandle hSerial1;

void func1(void)
{
    .........
    SuoSendSerialFile(hSerial1, "foo.ser");    /* Send serial data on "foo.ser" */
}
```

SuoSetNotifySentSerialCallback

シリアルの送信完了通知のコールバック登録を行います。

[指定形式]

```
#include    "suo.h"
int     SuoSetNotifySentSerialCallback(SuoHandle handle, SuoNotifySentSerialCallback func);
```

[引数]

引数	説明
handle	シリアル・インタフェースのハンドル
func	シリアルの送信完了時の処理を行うユーザ定義関数へのポインタ（「NotifySentSerialFunc」参照）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- シリアルの送信完了時の処理を行うユーザ定義関数を登録します。
- ここで登録された関数は，送信指定した１つ，または複数のシリアル・データがすべて送信完了した際に呼び出されます。
- *func* に NULL を指定した場合は，登録を解除します。

［使用例］

```
#include    "suo.h"
void NotifySentSerialFunc(SuoHandle handle);
SuoHandle hSerial1;

void func1(void)
{
    .........
    SuoSetNotifySentSerialCallback(hSerial1, NotifySentSerialFunc);
                                                    /* Set notify-sent-serial function */
}
/* Notify-sent-serial function */
void NotifySentSerialFunc(SuoHandle handle)
{
.........
}
```

SuoSetReceiveSerialCallback

シリアルの受信のコールバック登録を行います。

[指定形式]

```
#include    "suo.h"
int     SuoSetReceiveSerialCallback(SuoHandle handle, SuoReceiveSerialCallback func);
```

[引数]

引数	説明
handle	シリアル・インタフェースのハンドル
func	シリアルの受信時の処理を行うユーザ定義関数へのポインタ（「ReceiveSerialFunc」参照）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- シリアルの受信時の処理を行うユーザ定義関数を登録します。
- ここで登録された関数は，シリアル・データを 1 つ受信した際に呼び出されます。
- *func* に NULL を指定した場合は，登録を解除します。

［使用例］

```
#include    "suo.h"
void ReceiveSerialFunc(SuoHandle handle, unsigned long data, int status);
SuoHandle hSerial1;

void func1(void)
{
    .........
    SuoSetReceiveSerialCallback(hSerial1, ReceiveSerialFunc);
                                                        /* Set receive-serial function */
}
/* Receive-serial function */
void ReceiveSerialFunc(SuoHandle handle, unsigned long data, int status)
{
    .........
}
```

B. 4. 7　信号出力器インタフェース関数

信号出力器インタフェース関数には，次のものがあります。

関数名	機能概要
SuoCreateWave	信号出力器インタフェースの生成
SuoGetWaveHandle	信号出力器インタフェースのハンドルの取得
SuoSendWaveFile	信号出力器による信号データの送信
SuoSetNotifySentWaveCallback	信号出力器の送信完了通知のコールバック登録

SuoCreateWave

信号出力器インタフェースを生成します。

注意　この関数は，**MakeUserModel** 関数内でのみ呼び出すことができます。他のタイミングで呼び出した場合はエラーになります。

[指定形式]

```
#include    "suo.h"
int  SuoCreateWave(const char* waveName, int count, const char* pinNameList[], SuoHandle* handle);
```

[引数]

引数	説明
waveName	信号出力器の名前
count	信号出力器で使用する端子の個数
pinNameList[]	信号出力器で使用する端子の名前（配列により端子数分指定）
handle	信号出力器インタフェースのハンドルの格納場所

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 信号出力器インタフェースを生成します。
- 生成した信号出力器インタフェースは，*waveName* で指定した名前と関連付けられます。また，*count*，および *pinNameList* で指定した端子が生成されます。
- この関数が正常終了すると，生成した信号出力器インタフェースのハンドルを取得できます。後の信号出力器インタフェースに対する制御は，ここで取得したハンドルを指定して行います。
  なお，ハンドルは，SuoGetWaveHandle 関数により取得することもできます。

［使用例］

```
#include    "suo.h"
SuoHandle hWave1;

SuoUserEntry void MakeUserModel(const char *option)
{
    .........
    char* pinNameList[4] = {"P00", "P01", "P02", "P03"};
    SuoCreateWave("WAVE1", 4, pinNameList, &hWave1);        /* Create "WAVE1" */
}
```

SuoGetWaveHandle

信号出力器インタフェースのハンドルを取得します。

[指定形式]

```
#include    "suo.h"
SuoHandle   SuoGetWaveHandle(const char* waveName);
```

[引数]

引数	説明
waveName	信号出力器の名前

[戻り値]

マクロ	説明
信号出力器インタフェースのハンドル	正常終了
NULL	不正終了

[詳細説明]

- 指定した信号出力器インタフェースのハンドルを取得します。
- *waveName* には SuoCreateWave 関数で指定した名前を指定してください（異なる名前を指定した場合は NULL が返ります）。

[使用例]

```
#include    "suo.h"
SuoHandle hWave1;

void func1(void)
{
    .........
    hWave1 = SuoGetWaveHandle("WAVE1");        /* Get handle of "WAVE1" */
}
```

SuoSendWaveFile

信号出力器による信号データ（信号データ・ファイル）の送信を行います。

注意　この関数を MakeUserModel 関数内で呼び出すことはできません。コールバック関数内でのみ呼び出すことができます。

[指定形式]

```
#include    "suo.h"
int     SuoSendWaveFile(SuoHandle handle, const char* waveFile);
```

[引数]

引数	説明
handle	信号出力器インタフェースのハンドル
waveFile	信号データエディタ ウインドウで編集したのち保存した信号データ・ファイル名 なお，ファイル名を相対パスで指定した場合，ユーザ・モデル（*UserModel*.dll）のパスからの相対として扱われます。

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 信号データエディタ ウインドウで編集したのち保存した信号データ・ファイル（*.wvi）に記録されている信号データの送信を開始します。
- 信号データの送信が完了するまで時間がかかります。送信完了のタイミングを通知する場合は，SuoSetNotifySentWaveCallback 関数で送信完了通知関数を設定してください。
- 現在送信中の信号出力器インタフェースに対してこの関数を呼び出した場合は，送信中の信号データはキャンセルされ，新たに指定した信号データが送信されます。

［使用例］

```
#include    "suo.h"
SuoHandle hWave1;

void func1(void)
{
    .........
    SuoSendWaveFile(hSerial1, "foo.wvi");        /* Send pin data on "foo.wvi" */
}
```

SuoSetNotifySentWaveCallback

信号出力器の送信完了通知のコールバック登録を行います。

[指定形式]

```
#include    "suo.h"
int     SuoSetNotifySentWaveCallback(SuoHandle handle, SuoNotifySentWaveCallback func);
```

[引数]

引数	説明
handle	信号出力器インタフェースのハンドル
func	信号出力器の送信完了時の処理を行うユーザ定義関数へのポインタ（「NotifySentWaveFunc」参照）

[戻り値]

マクロ	説明
SUO_NOERROR	正常終了
エラー番号	不正終了（「B. 4. 8　エラー番号一覧」参照）

[詳細説明]

- 信号出力器の送信完了時の処理を行うユーザ定義関数を登録します。
- ここで登録された関数は，送信指定した信号データがすべて送信完了した際に呼び出されます。
- *func* に NULL を指定した場合は，登録を解除します。

［使用例］

```
#include    "suo.h"
void NotifySentWaveFunc(SuoHandle handle);
SuoHandle hWave1;

void func1(void)
{
    .........
    SuoSetNotifySentWaveCallback(hWave1, NotifySentWaveFunc);
                                                    /* Set notify-sent-wave function */
}
/* Notify-sent-wave function */
void NotifySentWaveFunc(SuoHandle handle)
{
    .........
}
```

B. 4. 8　エラー番号一覧

提供インタフェース関数からの戻り値として返されるエラー番号（マクロ）の意味は，次のとおりです。
なお，エラー番号は，ヘッダ・ファイル（suo.h）で定義されているマクロ名称で示されます。

表 B—4　エラー番号（マクロ）一覧

エラー番号（マクロ）	説明
SUO_NOERROR	正常終了
SUO_CANTALLOC	メモリが確保できませんでした。
SUO_ILLIFNAME	インフェース名が不正です。 インタフェース名に NULL，または“""”が指定されました。 または，ハンドル取得関数において，未生成のインタフェース名が指定されました。
SUO_ILLHANDLE	ハンドルが不正です。 生成したインタフェースのハンドル以外のハンドルが指定されました。
SUO_ILLPARAM	引数が不正です。 引数として指定できる値以外が指定されました。
SUO_CANTCALL	関数呼び出しができません。 MakeUserModel 関数でのみ呼び出すことができる関数が，MakeUserModel 関数以外の関数で呼び出されました。 または，MakeUserModel 関数以外で呼び出すことができない関数が，MakeUserModel 関数で呼び出されました。
SUO_CONFLICTRES	生成する資源が競合しています。 MakeUserModel 関数内で生成したインタフェース名，または端子名の中に同名のも存在します。
SUO_ILLFILENAME	ファイル名が不正です。 ファイル名に NULL，または無効な文字を含む名前が指定されました。
SUO_CANTOPENFILE	信号データ・ファイルをオープンすることができませんでした。 指定した信号データ・ファイルが存在しません。 または，指定した信号データ・ファイルがリード許可されていません。
SUO_ILLFILEFMT	【シリアル送信データ・ファイルの場合】 ファイルをオープンすることができませんでした。 ファイルが存在しない，ファイルがリード許可されていない，またはファイル名が不正であるかのいずれかです。 【信号データ・ファイルの場合】 ファイルの形式が不正です。 ファイル名に NULL，または無効な文字を含む名前が指定されました。
SUO_ILLFILECONT	ファイルの内容が不正です。 ファイルに記録されているデータの内容に矛盾がある，またはデータが 1 つも存在しません。
SUO_ILLPINNAME	端子名が不正です。 端子名に NULL，または“""”が指定されました。
SUO_ILLADDRRANGE	アドレス範囲が不正です。 有効なアドレス範囲ではありません。

エラー番号（マクロ）	説明
SUO_UNDERSENDING	すでに送信中です。 すでに送信中のため，新たに送信開始することができません。

B. 5　ユーザ定義関数

この節では，ユーザが作成するユーザ定義関数について説明します。

次に，ユーザ定義関数（ユーザ・モデルのエントリ関数，およびコールバック関数）の一覧を示します。

表 B—5　ユーザ定義関数一覧

関数名	機能概要
MakeUserModel	ユーザ・モデルのエントリ関数
InitFunc	初期化のコールバック関数
ResetFunc	リセットのコールバック関数
NotifyTimerFunc	タイマの時間通知のコールバック関数
InputDigitalPinFunc	端子のデジタル・データ入力のコールバック関数
InputAnalogPinFunc	端子のアナログ・データ入力のコールバック関数
InputHighImpedanceFunc	端子のハイ・インピーダンス状態通知のコールバック関数
ReadExtbusFunc	外部バスのリード・アクセスのコールバック関数
WriteExtbusFunc	外部バスのライト・アクセスのコールバック関数
NotifySentSerialFunc	シリアルの送信完了通知のコールバック関数
ReceiveSerialFunc	シリアルの受信のコールバック関数
NotifySentWaveFunc	信号出力器の送信完了通知のコールバック関数

MakeUserModel

ユーザ・モデルのエントリ関数として，使用する資源の生成を行います。

注意　MakeUserModel はユーザ・モデルの静的エントリ関数であるため，この関数名を使用する必要があります。

[指定形式]

```
#include    "suo.h"
SuoUserEntry    void MakeUserModel(const char *option);
```

[引数]

引数	説明
option	シミュレータ・コンフィギュレーション・ファイルで指定したオプション文字列 なお，シミュレータ・コンフィギュレーション・ファイルでオプションを指定しなかった場合，NULL 文字（" "）が入ります。

[戻り値]

なし

[詳細説明]

- この関数では，ユーザ・モデルで使用する資源の生成を行う必要があります（この関数以外の関数で資源の生成を行うことができません）。
- この関数では，必要に応じてコールバック関数の登録をする必要があります。
  特に初期化のコールバック関数の登録は，この関数内で登録する必要があります（この関数以外での登録では，初期化タイミングを過ぎているため意味を持ちません）。

［使用例］

```
#include    "suo.h"
SuoHandle hTim1;
SuoHandle hPinP00;
SuoHandle hExtbus1;

void InitFunc(void);
void ResetFunc(void);

SuoUserEntry void MakeUserModel(const char *option)
{
    /* Create source */
    SuoCreateTimer("TIM1", &hTim1);                          /* Create "TIM1" */
    SuoCreatePin("P00", &hPinP00);                           /* Create "P00" */
    SuoCreateExtbus("EXTBUS1", 0x200000, 0x1000, &hExtbus1); /* Create "EXTBUS1" */

    /* Set callbacks */
    SuoSetInitCallback(InitFunc);                              /* Set initialize function */
    SuoSetResetCallback(ResetFunc);                          /* Set reset function */
}
```

InitFunc

コールバック関数として，初期化処理を行います。

注意　InitFunc はユーザ定義の関数名のプレース・ホルダであるため，この関数名を使用する必要はありません。

[指定形式]

```
#include    "suo.h"
void    InitFunc (void);
```

[引数]

なし

[戻り値]

なし

[詳細説明]

- InitFunc では，初期化処理を記述します。
- InitFunc をコールバック関数として登録するためには，SuoSetInitCallback 関数を使用します。

ResetFunc

コールバック関数として，リセット処理を行います。

注意　ResetFunc はユーザ定義の関数名のプレース・ホルダであるため，この関数名を使用する必要はありません。

[指定形式]

```
#include    "suo.h"
void    ResetFunc (void);
```

[引数]

なし

[戻り値]

なし

[詳細説明]

- ResetFunc では，リセット処理を記述します。
- ResetFunc をコールバック関数として登録するためには，SuoSetResetCallback 関数を使用します。

NotifyTimerFunc

コールバック関数として，タイマの時間通知の処理を行います。

注意　NotifyTimerFunc はユーザ定義の関数名のプレース・ホルダであるため，この関数名を使用する必要はありません。

[指定形式]

```
#include    "suo.h"
void    NotifyTimerFunc (SuoHandle handle);
```

[引数]

引数	説明
handle	タイマ・インタフェースのハンドル

[戻り値]

なし

[詳細説明]

- NotifyTimerFunc では，タイマの時間通知時の処理を記述します。
- NotifyTimerFunc をコールバック関数として登録するためには，SuoSetNotifyTimerCallback 関数を使用します。

InputDigitalPinFunc

コールバック関数として，端子のデジタル入力処理を行います。

注意　InputDigitalPinFunc はユーザ定義の関数名のプレース・ホルダであるため，この関数名を使用する必要はありません。

[指定形式]

```
#include    "suo.h"
void    InputDigitalPinFunc (SuoHandle handle, int pinValue);
```

[引数]

引数	説明
handle	端子インタフェースのハンドル
pinValue	次のいずれかの端子への入力値（デジタル・データ） - SUO_HIGH (=1)：　HIGH 値 - SUO_LOW (=0)：　LOW 値

[戻り値]

なし

[詳細説明]

- InputDigitalPinFunc では，端子のデジタル入力処理を記述します。
- InputDigitalPinFunc をコールバック関数として登録するためには，SuoSetInputDigitalPinCallback 関数を使用します。

InputAnalogPinFunc

コールバック関数として，端子のアナログ入力処理を行います。

注意　InputAnalogPinFunc はユーザ定義の関数名のプレース・ホルダであるため，この関数名を使用する必要はありません。

[指定形式]

```
#include    "suo.h"
void    InputAnalogPinFunc (SuoHandle handle, double pinValue);
```

[引数]

引数	説明
handle	端子インタフェースのハンドル
pinValue	端子への入力値（アナログ・データ）（単位：V）

[戻り値]

なし

[詳細説明]

- InputAnalogPinFunc では，端子のアナログ入力処理を記述します。
- InputAnalogPinFunc をコールバック関数として登録するためには，SuoSetInputAnalogPinCallback 関数を使用します。

InputHighImpedanceFunc

コールバック関数として，端子のハイ・インピーダンス状態通知の処理を行います。

注意　InputHighImpedanceFunc はユーザ定義の関数名のプレース・ホルダであるため，この関数名を使用する必要はありません。

[指定形式]

```
#include    "suo.h"
void    InputHighImpedanceFunc (SuoHandle handle);
```

[引数]

引数	説明
handle	端子インタフェースのハンドル

[戻り値]

なし

[詳細説明]

- InputHighImpedanceFunc では，デジタル／アナログ端子に接続されたすべての端子がハイ・インピーダンス状態になった際の処理を記述します。
- InputHighImpedanceFunc をコールバック関数として登録するためには，SuoSetInputHighImpedanceCallback 関数を使用します。

ReadExtbusFunc

コールバック関数として，外部バスのリード・アクセスの処理を行います。

注意　ReadExtbusFunc はユーザ定義の関数名のプレース・ホルダであるため，この関数名を使用する必要はありません。

[指定形式]

```
#include    "suo.h"

void    ReadExtbusFunc (SuoHandle handle, unsigned long addr, int accessSize, unsigned char
data[]);
```

[引数]

引数	説明
handle	外部バス・インタフェースのハンドル
addr	アドレス
accessSize	アクセス・サイズ
data[]	データ格納領域 なお，アクセス・サイズ分のデータを格納する必要があります。

[戻り値]

なし

[詳細説明]

- ReadExtbusFunc では，外部バスのリード・アクセスの処理を記述します。
- *data[]* へのデータ格納処理を行う必要があります。
- ReadExtbusFunc をコールバック関数として登録するためには，SuoSetReadExtbusCallback 関数を使用します。

WriteExtbusFunc

コールバック関数として，外部バスのライト・アクセスの処理を行います。

注意　WriteExtbusFunc はユーザ定義の関数名のプレース・ホルダであるため，この関数名を使用する必要はありません。

[指定形式]

```
#include    "suo.h"

void    WriteExtbusFunc (SuoHandle handle, unsigned long addr, int accessSize, const unsigned
char data[]);
```

[引数]

引数	説明
handle	外部バス・インタフェースのハンドル
addr	アドレス
accessSize	アクセス・サイズ
data[]	データ格納領域 なお，アクセス・サイズ分のデータを格納する必要があります。

[戻り値]

なし

[詳細説明]

- WriteExtbusFunc では，外部バスのライト・アクセスの処理を記述します。
- WriteExtbusFunc をコールバック関数として登録するためには，SuoSetWriteExtbusCallback 関数を使用します。

NotifySentSerialFunc

コールバック関数として，シリアルの送信完了通知の処理を行います。

注意　NotifySentSerialFunc はユーザ定義の関数名のプレース・ホルダであるため，この関数名を使用する必要はありません。

[指定形式]

```
#include    "suo.h"
void    NotifySentSerialFunc (SuoHandle handle);
```

[引数]

引数	説明
handle	シリアル・インタフェースのハンドル

[戻り値]

なし

[詳細説明]

- NotifySentSerialFunc では，シリアルの送信完了時の処理を記述します。
- NotifySentSerialFunc をコールバック関数として登録するためには，SuoSetNotifySentSerialCallback 関数を使用します。

ReceiveSerialFunc

コールバック関数として，シリアルの受信時の処理を行います。

注意　ReceiveSerialFunc はユーザ定義の関数名のプレース・ホルダであるため，この関数名を使用する必要はありません。

[指定形式]

```
#include    "suo.h"
void    ReceiveSerialFunc (SuoHandle handle, unsigned long data, int status);
```

[引数]

引数	説明
handle	シリアル・インタフェースのハンドル
data	受信したシリアル・データ
status	次のいずれかの受信ステータス - 0：　正常受信 - SUO_PARITYERR：　パリティ・エラー（パリティ・ビットの不一致の場合） - SUO_FRAMINGERR：　フレーミング・エラー（ストップ・ビットが検出されない場合）

[戻り値]

なし

[詳細説明]

- ReceiveSerialFunc では，シリアル受信時の処理を記述します。
- ReceiveSerialFunc をコールバック関数として登録するためには，SuoSetReceiveSerialCallback 関数を使用します。

NotifySentWaveFunc

コールバック関数として，信号出力器の送信完了通知の処理を行います。

注意　NotifySentWaveFunc はユーザ定義の関数名のプレース・ホルダであるため，この関数名を使用する必要はありません。

[指定形式]

```
#include    "suo.h"
void    NotifySentWaveFunc (SuoHandle handle);
```

[引数]

引数	説明
handle	信号出力器インタフェースのハンドル

[戻り値]

なし

[詳細説明]

- NotifySentWaveFunc では，信号出力器の送信完了通知の処理を記述します。
- NotifySentWaveFunc をコールバック関数として登録するためには，SuoSetNotifySentWaveCallback 関数を使用します。

B. 6　サンプル・プログラム（Timer モデル）

この節では，シミュレータ GUI のユーザ・オープン・インタフェースを使用したユーザ・モデルのサンプル・プログラム（Timer モデル）について説明します。

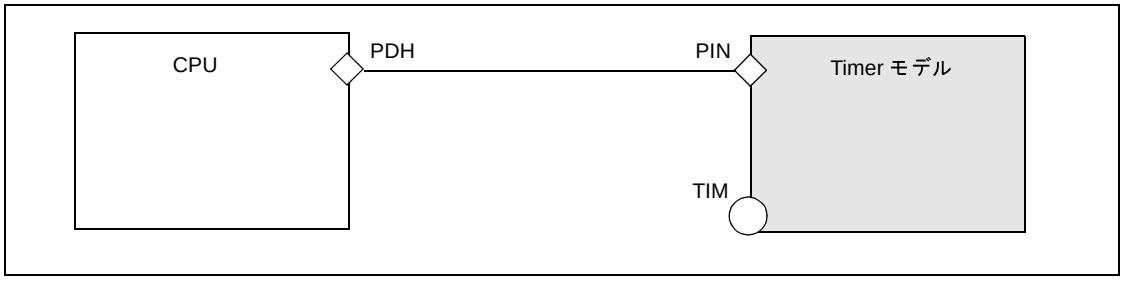
B. 6. 1　概　　要

Timer モデルは，タイマ・インタフェースを使用し，決められた時間間隔で端子に対して値を出力させるサンプル・プログラムです。

B. 6. 2　構　　成

Timer モデルは，PIN 端子と TIM タイマを生成します。生成した PIN 端子は CPU の PDH 端子に接続します。

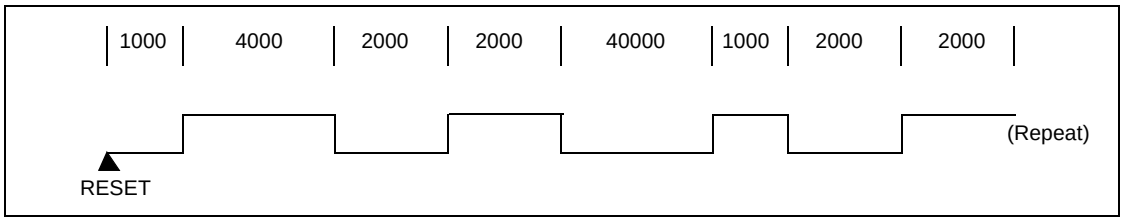
図 B—5　Timer モデルの構成



B. 6. 3　動　　作

タイマ・インタフェースを使用して決められた時間を計測し，端子 PDH に対して LOW/HIGH を交互に出力します。出力する値と時間の関係は次のとおりです。

図 B—6　Timer モデルの動作



B. 6. 4　プロジェクト・ファイル

Timer モデルのプロジェクト・ファイル（Microsoft Visual C++）の設定情報は次のとおりです。

表 B—6　Timer モデルの設定情報

設定情報	内容
プロジェクト・タイプ	Win32 Dynamic-Link Library
構成ソース・ファイル	suolink.c，uo_timer.c
インクルード・ファイルのパス	suo.h の格納フォルダ

B. 6. 5　プログラム詳細

次に，Timer モデルのサンプル・プログラムを示します。

（1）Timer モデルのソース・ファイル（uo_timer.c）
（2）シミュレータ・コンフィギュレーション・ファイル（smplus.cfg）
（3）ターゲット・プログラムのソース・ファイル（lm_timer.c）

（1）Timer モデルのソース・ファイル（uo_timer.c）

```
#include <windows.h>
#include "suo.h"

/* Handle */
SuoHandle hTIM;
SuoHandle hPIN;

/* Wave-Table */
#define MAXWAVE 8
struct _WaveTable {
    unsigned longtime;              /* Wait Time [usec] */
    int pinValue;                   /* Pin Value (SUO_HIGH or SUO_LOW) */
} waveTable[MAXWAVE] = {
    1000, SUO_HIGH,
    4000, SUO_LOW,
    2000, SUO_HIGH,
    2000, SUO_LOW,
    4000, SUO_HIGH,
    1000, SUO_LOW,
    2000, SUO_HIGH,
    2000, SUO_LOW
};
int waveIndex;

/* Declare */
void Reset(void);
void NotifyTimer(SuoHandle handle);
void puterr(int error);

/* MakeUserModel */
void SuoUserEntry MakeUserModel(const char *option)
{
    int error;
```

```
    /* Create interface */
    if((error = SuoCreateTimer("TIM", &hTIM)) != SUO_NOERROR){
        puterr(error);
        return;
    }
    if((error = SuoCreatePin("PIN", &hPIN)) != SUO_NOERROR){
        puterr(error);
        return;
    }

    /* Set callback */
    SuoSetResetCallback(Reset);
    SuoSetNotifyTimerCallback(hTIM, NotifyTimer);
    }

/* Reset callback */
void Reset(void)
{
    int error;

    /* Initialize Wave-Tabel index */
    waveIndex = 0;

    /* Output LOW(initial value) to PIN */
    if((error = SuoOutputDigitalPin(hPIN, SUO_LOW)) != SUO_NOERROR){
        puterr(error);
        return;
    }

    /* Set wait time */
    if((error = SuoSetTimer(hTIM, SUO_USEC, waveTable[waveIndex].time)) != SUO_NOERROR){
        puterr(error);
        return;
    }
}

/* NotifyTimer callback */
void NotifyTimer(SuoHandle handle)
{
    int error;

    /* Output value to PIN */
    if((error = SuoOutputDigitalPin(hPIN, waveTable[waveIndex].pinValue)) != SUO_NOERROR){
        puterr(error);
        return;
    }
```

```
    /* Set next Wave-Tabel index */
    waveIndex++;
    if(waveIndex >= MAXWAVE){
        waveIndex = 0;
    }

    /* Set wait time */
    if((error = SuoSetTimer(hTIM, SUO_USEC, waveTable[waveIndex].time)) != SUO_NOERROR){
        puterr(error);
        return;
    }
}

/* Report error */
void puterr(int error)
{
    char message[80];
    wsprintf(message, "The user open interface error (0x%04x) occurred.", error);
    MessageBox(NULL, message, "ERROR", MB_OK|MB_ICONERROR);
}
```

(2) シミュレータ・コンフィギュレーション・ファイル（smplus.cfg）

```
cpu = CPU('a');

#=====================================================================
# UO_TIMER description (CPU=uPD70F3261Y)
#=====================================================================

#---- Create UserOpen ---------------------------------------------
uo_timer = Device("USEROPEN", "Release\uo_timer.dll");

#---- Pseudo Pin connection ---------------------------------------
wire_clock =  Wire(1);
wire_clock += cpu.DebuggerPseudoPort("debugger_pseudo_pin_main_clkout");
wire_clock += uo_timer.Port("gui_pseudo_pin_clock_notice");
wire_reset =  Wire(1);
wire_reset += cpu.DebuggerPseudoPort("debugger_pseudo_pin_reset_notice");
wire_reset += uo_timer.Port("gui_pseudo_pin_reset_notice");

#---- PIN connection ----------------------------
# UO_TIMER.PIN <--> CPU.PDH0
wire_PIN =  Wire(1);
wire_PIN += uo_timer.Port("PIN");
wire_PIN += cpu.Port("PDH0");
```

(3) ターゲット・プログラムのソース・ファイル（lm_timer.c）

```
/* Target Program for UO_TIMER */

#pragma ioreg

void main( )
{
    unsigned char value;

    PMDH.0 = 1;                 /* set port-input mode */
    PMDH.1 = 0;                 /* set port-output mode */

    while( 1 ){
        value = PDH.0;         /* input signal from "PDH0" */
        PDH.1 = value;         /* output signal to "PDH1" */
    }
}
```

付録 C 索 引

【A】

Analog Button Properties ダイアログ … 390

Auto 変数 … 256

【C】

CPU レジスタ パネル … 242

【D】

DMA … 282

DMA 機能 … 282

DMM 機能 … 103

【E】

EEPROM ライブラリ … 27

【I】

I/O 保護領域 … 299

InitFunc … 524

InputAnalogPinFunc … 528

InputDigitalPinFunc … 527

InputHighImpedanceFunc … 529

【M】

MakeUserModel … 522

【N】

NotifySentSerialFunc … 532

NotifySentWaveFunc … 534

NotifyTimerFunc … 526

【O】

Object Properties ダイアログ … 421

【P】

Parts Button Properties ダイアログ … 386

Parts Buzzer Properties ダイアログ … 415

Parts Key Properties ダイアログ … 394

Parts Led Properties ダイアログ … 403

Parts Level Gauge Properties ダイアログ … 399

Parts Matrix Led Properties ダイアログ … 411

Parts Segment LED Properties ダイアログ … 407

PC をここに設定 … 88

Printf イベント … 151, 287, 289, 308

【R】

ReadExtbusFunc … 530

ReceiveSerialFunc … 533

Register 変数 … 256

ResetFunc … 525

ROM レス品 … 301

RRM 機能 … 103

Run-Beak 時間 … 143

Run-Break タイマ … 287, 288

Run-Break タイマ・イベント … 143

【S】

SFR パネル … 248

SuoCreateExtbus … 479

SuoCreatePin … 465

SuoCreateSerialCSI … 489

SuoCreateSerialUART … 487

SuoCreateTimer … 456

SuoCreateWave … 512

SuoGetExtbusHandle … 481

SuoGetMainClock … 454

SuoGetPinHandle … 467

SuoGetSerialHandle … 491

SuoGetSerialParameterCSI … 499

SuoGetSerialParameterUART … 497

SuoGetTimerHandle … 458

SuoGetWaveHandle … 514

SuoKillTimer … 461

SuoOutputAnalogPin … 470

SuoOutputDigitalPin … 468

SuoOutputHighImpedance … 471

SuoSendSerialData … 501

SuoSendSerialDataList … 503

SuoSendSerialFile … 505

SuoSendWaveFile … 515
SuoSetInitCallback … 452
SuoSetInputAnalogPinCallback … 474
SuoSetInputDigitalPinCallback … 472
SuoSetInputHighImpedanceCallback … 476
SuoSetNotifySentSerialCallback … 507
SuoSetNotifySentWaveCallback … 517
SuoSetNotifyTimerCallback … 462
SuoSetReadExtbusCallback … 482
SuoSetReceiveSerialCallback … 509
SuoSetResetCallback … 453
SuoSetSerialParameterCSI … 494
SuoSetSerialParameterUART … 492
SuoSetTimer … 459
SuoSetWriteExtbusCallback … 484

【W】

WriteExtbusFunc … 531

【あ行】

アクション・イベント … 151, 291
アクション・イベント ダイアログ … 308
アクセス幅 … 300
アップロード … 63, 71
アップロード可能なファイル形式 … 71
アドレス範囲 … 300
イベント・エリア … 217, 234
イベント種別 … 288
イベントの管理 … 154
　イベントのアドレスにジャンプする … 156
　イベントの設定状態を変更する … 154
　実行中に設定／削除可能なイベント種別 … 157
　設定したイベントを削除する … 156
　特定のイベント種別のみ表示する … 155
　有効イベント数の制限 … 157
イベントの設定状態 … 155
イベント パネル … 285
イベント・マーク … 287
インタフェース関数 … 449
　外部バス・インタフェース関数 … 478
　基本インタフェース関数 … 451
　時間インタフェース関数 … 455
　シリアル・インタフェース関数 … 486
　信号出力器インタフェース関数 … 511
　端子インタフェース関数 … 464
ウインドウ・リファレンス … 168
ウォッチ式 … 117, 261
ウォッチ式として登録可能な対象 … 118
ウォッチ式の登録 … 261
ウォッチ式の登録方法 … 118
ウォッチ式の入力形式 … 264
ウォッチ式の表示形式 … 267
ウォッチ パネル … 261
エディタ パネル … 215
エンコード … 311
オープン・ブレーク機能 … 198
オプション ダイアログ … 332
　[全般 - ビルド／デバッグ] カテゴリ … 339
　[全般 - フォントと色] カテゴリ … 334
オフセット値 … 236, 280
オンチップ・デバッグ・セキュリティ ID … 33, 42, 50

【か行】

改行コード … 311
外部バス・インタフェース関数 … 478
　SuoCreateExtbus … 479
　SuoGetExtbusHandle … 481
　SuoSetReadExtbusCallback … 482
　SuoSetWriteExtbusCallback … 484
カバレッジ測定 … 147
　カバレッジ測定結果を表示する … 148
　カバレッジ測定の設定をする … 147
カレント PC 位置 … 218, 235
カレント PC マーク … 218, 235
関数のエピローグ … 256
関数のプロローグ … 256
基本インタフェース関数 … 451
　SuoGetMainClock … 454
　SuoSetInitCallback … 452
　SuoSetResetCallback … 453
逆アセンブル結果の表示 … 79
逆アセンブル・テキスト … 79, 232

逆アセンブル パネル … 232
逆アセンブル表示モード … 133, 281
共用体 … 256, 263
グローバル変数 … 114
クロックの設定 … 21, 31, 40, 48, 56
構造体 … 256, 263
コード … 236
コード・カバレッジ測定 … 147
コール・スタック情報 … 274
コール・スタック パネル … 274
コールバック関数 … 438
ここまで実行 … 88
混合表示モード … 133, 281
コンフィギュレーション … 57

【さ行】

サブ・クロック … 189
サンプル・プログラム … 535
時間インタフェース関数 … 455
SuoCreateTimer … 456
SuoGetTimerHandle … 458
SuoKillTimer … 461
SuoSetNotifyTimerCallback … 462
SuoSetTimer … 459
実行時間の計測 … 143
実行開始から停止までの実行時間を計測する … 143
測定可能時間の範囲 … 146
任意区間の実行時間を計測する … 144
実行履歴の収集 … 125
実行開始から停止までの実行履歴を収集する … 128
実行履歴の表示内容を保存する … 141
実行履歴を表示する … 132
条件を満たした場合の実行履歴を収集する … 130
トレース・データを検索する … 134
トレース動作の設定をする … 125
トレース・メモリをクリアする … 134
任意区間の実行履歴を収集する … 128
指定アドレスへ移動 … 80
指定位置へ移動 ダイアログ … 326
指定行へ移動 … 76
シミュレータ GUI ウインドウ … 350
シミュレータ・コンフィギュレーション・ファイル … 445
シミュレータ・コンフィギュレーション・ファイルを選択 ダイアログ … 348
出力 パネル … 295
初期化データ … 313
書式設定 ダイアログ … 355
処理中表示 ダイアログ … 331
シリアル・インタフェース関数 … 486
SuoGetSerialHandle … 491
SuoCreateSerialCSI … 489
SuoCreateSerialUART … 487
SuoGetSerialParameterCSI … 499
SuoGetSerialParameterUART … 497
SuoSendSerialData … 501
SuoSendSerialDataList … 503
SuoSendSerialFile … 505
SuoSetNotifySentSerialCallback … 507
SuoSetReceiveSerialCallback … 509
SuoSetSerialParameterCSI … 494
SuoSetSerialParameterUART … 492
シリアル ウインドウ … 427
シリアル送信データ・ファイル … 505
信号出力器インタフェース関数 … 511
SuoCreateWave … 512
SuoGetWaveHandle … 514
SuoSendWaveFile … 515
SuoSetNotifySentWaveCallback … 517
信号データエディタ ウインドウ … 357
信号データ・ファイル … 357
シンボル定義箇所へ移動 … 81
スコープの指定 … 114
スタックからの関数呼び出し情報の表示 … 123
コール・スタック情報を表示する … 123
コール・スタック情報を保存する … 124
ステータスバー … 176
ステップ・イン実行 … 89, 175
ステップ・オーバー実行 … 90, 175

ステップ実行 … 89
制御レジスタ … 109, 242
セキュリティ ID … 192
接続方法 … 62
切断方法 … 62
［全般 - ビルド／デバッグ］カテゴリ … 339
［全般 - フォントと色］カテゴリ … 334
ソース表示モード … 133, 281
ソース・レベル・デバッグ … 73, 215
ソフトウエア・ブレーク … 92, 287, 288

【た行】

ターゲット・ボード … 22
タイマ … 60
タイマ開始 … 287
タイマ開始イベント … 144, 288
タイマ計測 … 287, 288
タイマ計測イベント … 144
タイマ終了 … 287
タイマ終了イベント … 144, 288
タイミングチャート ウインドウ … 368
タイミングチャート・ファイル … 368
ダウンロード … 29, 38, 46, 63, 209
ダウンロード可能なファイル形式 … 66
ダウンロード条件 … 303
ダウンロード条件の変更 … 67
ダウンロードするファイルを選択 ダイアログ … 341
ダウンロード・ファイル ダイアログ … 303
タグ・ジャンプ … 77, 219, 296
端子インタフェース関数 … 464
SuoCreatePin … 465
SuoGetPinHandle … 467
SuoOutputAnalogPin … 470
SuoOutputDigitalPin … 468
SuoOutputHighImpedance … 471
SuoSetInputAnalogPinCallback … 474
SuoSetInputDigitalPinCallback … 472
SuoSetInputHighImpedanceCallback … 476
端子選択 ダイアログ … 366
データ・カバレッジ測定 … 147
データ検索 ダイアログ … 374
データフラッシュ・エミュレーション機能 … 27
データフラッシュ・メモリ … 31, 40, 48
データ保存 ダイアログ … 328
データ保存ファイルを選択 ダイアログ … 346
テキスト編集 ダイアログ … 307
デバッグ情報 … 210
デバッグ専用プロジェクト … 63
デバッグ・ツールとの接続／切断 … 62
デバッグ・ツールバー … 174
動作環境設定 … 19
特長 … 11
トレース・イベント … 128
トレース開始 … 287
トレース開始イベント … 128, 288
トレース検索 ダイアログ … 317
トレース終了 … 287
トレース終了イベント … 129, 288
トレース動作の設定 … 125
トレース パネル … 278
トレース番号 … 279
トレース・フレーム … 127
トレース・メモリ … 125, 201
トレース・タイム・タグ … 201

【な行】

内部スタティック変数 … 256
名前を付けて保存 ダイアログ … 343
入出力パネル ウインドウ … 376
入力信号のマスク … 26, 37, 46, 54

【は行】

ハードウエア・ブレーク … 92, 287, 288
バイナリ・データ・フォーマット … 68
配列 … 256, 263
汎用レジスタ … 109, 242
ビルトイン・イベント … 128, 288
ファイルの監視機能 … 220
ファイルの保存設定 ダイアログ … 311
フェイルセーフ・ブレーク機能 … 26, 98
フォーマット（CSI）ダイアログ … 433
フォーマット（UART）ダイアログ … 431

不正なアクセス … 98
フック処理 … 212
フック処理を設定する … 159
部品一覧 ダイアログ … 425
フラッシュ … 33, 42, 50
フラッシュ書き換え … 33, 42, 50
プルアップ／プルダウン設定 ダイアログ … 419
ブレーク … 37, 45, 53, 59
ブレーク・イベント … 94
ブレークポイント … 91
ブレークポイントの削除 … 94
ブレークポイントの種別 … 91, 198
ブレーク要因 … 99, 280
プログラム内にアクションを挿入する … 151
printf を挿入する … 151
プログラムの実行 … 86
プログラムを実行する … 87
プログラムをステップ実行する … 89
プログラムの停止 … 91
任意の場所で停止する … 91
プログラムの実行を手動で停止する … 91
変数へのアクセスで停止する … 94
プログラムの表示と変更 … 73
逆アセンブルを表示する … 79
ソース・ファイルを表示する … 74
他の処理と平行してビルドを実行する … 83
ライン・アセンブルを行う … 84
プロジェクト・ツリー パネル … 179
プロパティ パネル … 182
［接続用設定］タブ … 185
［ダウンロード・ファイル設定］タブ … 209
［データ・フラッシュ・エミュレーション設定］タブ … 205
［デバッグ・ツール］タブ … 194
［フック処理設定］タブ … 212
ヘキサ・フォーマット … 68
ベリファイ … 196
変数値のポップアップ表示 … 220
変数へのアクセス … 94
ポインタ型変数 … 256, 263
ポイント・トレース … 288
ポイント・トレース・イベント … 130, 223

【ま行】

マクロ … 519
無条件トレース … 288
無条件トレース・イベント … 128, 155
命令レベル・デバッグ … 73
メイン・ウインドウ … 171
メイン・クロック … 188
メニューバー … 172
メモリ・アクセス … 24, 36, 44, 52, 59
メモリ検索 ダイアログ … 315
メモリ種別 … 299
メモリ初期化 ダイアログ … 313
メモリの設定 … 23, 58
メモリ・マッピング … 23, 35, 43, 51, 58, 299
メモリ・マッピング ダイアログ … 298
メモリ，レジスタ，変数の表示／変更 … 100
CPU レジスタを表示／変更する … 109
SFR を表示／変更する … 112
ウォッチ式を表示／変更する … 117
グローバル変数・スタティック変数を表示／変更する … 114
メモリを表示／変更する … 100
ローカル変数を表示／変更する … 114

【や行】

有効イベント数の制限 … 157
ユーザ定義関数 … 521
ユーザ定義関数関数
InitFunc … 524
InputAnalogPinFunc … 528
InputDigitalPinFunc … 527
InputHighImpedanceFunc … 529
MakeUserModel … 522
NotifySentSerialFunc … 532
NotifySentWaveFunc … 534
NotifyTimerFunc … 526
ReadExtbusFunc … 530
ReceiveSerialFunc … 533
ResetFunc … 525

WriteExtbusFunc … 531
ユーザ・オープン・インタフェース … 437
ユーザ・モデル … 440
読み込み保護対象 … 250

【ら行】

ライン・アセンブル … 84, 236
ラピッド・ビルド機能 … 83
ラベル名 … 236, 280
リアルタイム表示更新機能 … 103, 121
リセット … 86
リターン・アウト実行 … 90, 176
ループ設定 ダイアログ … 364
レジスタ・バンク … 244
ローカル変数 … 114
ローカル変数 パネル … 255
ロード・モジュール・ファイル … 63

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2011.04.01	―	初版発行

CubeSuite+ V1.00.00 ユーザーズマニュアル
RL78 デバッグ編

発行年月日　2011 年 4 月 1 日　Rev.1.00
発行　ルネサス エレクトロニクス株式会社
〒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・電話番号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 エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル）　(03)5201-5307

■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/inquiry

© 2011 Renesas Electronics Corporation. All rights reserved.
Colophon 1.0

CubeSuite+ V1.00.00

RENESAS

ルネサス エレクトロニクス株式会社

R20UT0561JJ0100