CS+ for CA, CX V4.04.00

リリースノート

R20UT4666JJ0100
Rev.1.00
2019.11.29

この度は，統合開発環境 CS+をご使用いただきまして，誠にありがとうございます。

この添付資料では，本製品をお使いいただく上での制限事項および注意事項等を記載しております。

ご使用の前に，必ずお読みくださいますようお願い申し上げます。

目次

第 1 章　対象デバイスについて 2

第 2 章　ユーザーズ・マニュアルについて 3

第 3 章　アンインストール方法 4

第 4 章　機能改善・変更点 5

第 5 章　注意事項 6

第 6 章　制限事項 19

第 7 章　ドキュメント訂正 20

第1章　対象デバイスについて

統合開発環境 CS+がサポートする対象デバイスに関しては，WEB サイトに掲載しています。

こちらをご覧ください。

CS+製品ページ：

https://www.renesas.com/cs+

第2章　ユーザーズ・マニュアルについて

本製品に対応したユーザーズ･マニュアルは，次のようになります。本文書と合わせてお読みください。

マニュアル名	資料番号
CubeSuite+ V2.02.00 起動編	R20UT2865JJ0100
CubeSuite+ V1.03.00 78K0 設計編	R20UT2138JJ0100
CubeSuite+ V1.03.00 78K0R 設計編	R20UT2137JJ0100
CubeSuite+ V2.01.00 RL78 設計編	R20UT2684JJ0100
CubeSuite+ V1.03.00 V850 設計編	R20UT2134JJ0100
CubeSuite+ V1.01.00 78K0 デバッグ編	R20UT0731JJ0100
CubeSuite+V1.01.00 78K0R デバッグ編	R20UT0732JJ0100
CubeSuite+ V2.02.00 RL78 デバッグ編	R20UT2867JJ0100
CubeSuite+ V2.00.00 V850 デバッグ編	R20UT2446JJ0100
CubeSuite+ V2.02.00 解析編	R20UT2868JJ0100
CubeSuite+ V2.02.00 メッセージ編	R20UT2871JJ0100

第3章　アンインストール方法

本製品をアンインストールする場合は，2 つの方法があります。

・統合アンインストーラを使用する(CS+自体をアンインストールする)

・個別にアンインストールする(本製品のみをアンインストールする)

個別にアンインストールを行なう場合，Windows の設定の「アプリと機能」や，コントロール・パネルの「プログラムと機能」から，「CS+ for CA,CX」を選択してください。

第4章　機能改善・変更点

本章では，CS+の V4.03.00 から V4.04.00 の機能改善・変更点について説明します。

4.1　CS+の全体の機能改善・変更点

4.1.1　Windows 7 のサポート終了

マイクロソフト社は 2020 年 1 月 14 日をもって Windows 7 の無償でのサポートを終了しました。
これを受けて，CS+は Windows 7 環境での評価および動作保証を終了し，今後は Windows 10，Windows 8.1 のサポートに重点をおきます。

第5章　注意事項

本章では，注意事項について説明します。

5.1　CS+全体の注意事項

5.1.1　ファイル名に関する注意事項

フォルダ名，ファイル名に関しては次の注意事項があります。

・フォルダ名，ファイル名

Windows のエクスプローラーで作成することのできないフォルダ名とファイル名は，使用しないでください。

ネットワークパス名は使用できません。ドライブ名に割り当てて使用してください。

・ソース・ファイル名とロード・モジュール・ファイル名とプロジェクト・ファイル名

ファイル名は，a-z，A-Z，0-9，.(ピリオド)，_(アンダスコア)，+，- のいずれかの文字で構成されます。

ファイル名の先頭と最後に，.(ピリオド)の文字は使えません。

ファイル名の先頭に「+」（プラス）／「-」（マイナス）は使えません。

英大文字(A - Z)，英小文字(a - z)は区別されません。

ファイル名は，パスを含めて最大 259 文字です。

ファイル名が同じソース･ファイルは使用しないで下さい。異なるパスに存在していても区別できません。

・上記以外のファイル名

Windows のファイル名規約に準拠します。

なお，ファイル名には次の文字は使えません。

¥ / : * ? " < > | ;

ファイル名の先頭と最後に.(ピリオド) とスペースは使えません。

英大文字(A - Z)，英小文字(a - z)は区別されません。

ファイル名は，パスを含めて最大 259 文字です。

・フォルダ名

Windows のファイル名規約に準拠します。

5.1.2　パネル表示に関する注意事項

使用するハードウエア環境が CS+ の推奨サポート環境を下回るスペックである場合，［プロパティ］パネルのサイズを小さくすると表示内容が乱れることがあります。

その場合には，分割パネル領域から［プロパティ］パネルを外に出してください。

・ドッキング可能を ON にして，ドッキング・パネル化する

・フローティングを ON にして，フローティング・パネル化する

5.1.3　マイクロコントローラ変更に関する注意事項

マイクロコントローラを変更する場合には，次の注意事項があります。

- 同じファミリ（V850， RL78，78K0R，78K0）内の，同じビルド・ツールに対応しているマイクロコントローラへのみ変更が可能です。
- マイクロコントローラを変更する際は，デバッグ・ツールを接続していない状態にしてください。
- マイクロコントローラを変更する前に，プロジェクトを保存する必要があります。
- 端子配置（設計ツール），コード生成（設計ツール），デバッグ・ツール（ウォッチ登録情報除く）の情報は，マイクロコントローラの変更後，引き継がれません。

5.1.4 プラグイン管理機能に関する注意事項

プラグインの管理ダイアログの基本機能タブにおいて，開発対象となるマイクロコントローラ用プラグインのチェックは，外さないことを推奨します。

開発対象ではないマイクロコントローラ用のビルド・ツール・プラグイン，デバッグ・ツール・プラグインのチェックを外してください。たとえば，ビルド・ツール・プラグインのみチェックを外すとデバッグ・ツールでダウンロードするファイルが見つからずエラーとなります。

5.1.5 エディタパネルに関する注意事項

- ページ設定ダイアログが使用できません。
- 印刷プレビューのツールバーにコピーボタンがありますが，使用できません。
- 変数，ラベルを選択して，コンテキスト・メニューの「関数へジャンプ」機能を使用した場合，変数，ラベルにジャンプしません。
- 関数へジャンプ機能で，別ファイルに定義されている static 関数には移動できません。
- メイン・プロジェクトとサブプロジェクトに，パスの違う同名のソース・ファイルが登録されていて，メイン・プロジェクトとサブプロジェクトのロード・モジュールを両方ダウンロードしたとき，次のようになります。
  - 当該ファイルでは，メイン・プロジェクトのアドレスが表示される
  - 当該ファイルの逆アセンブルから「ソースへジャンプ」を行うと，メイン・プロジェクトに登録されているファイルが開く
  - どちらのプロジェクトから当該ファイルを開いても１つのファイルしか開けない
- 無名の構造体ではスマート・エディットが正しく動作しません。
- 関数の引数に関数呼び出しが含まれる場合，ツールチップに誤った情報が表示されます。
- クラスの配列やクラスのポインタの配列に対して，メンバ変数，メンバ関数の補完が正しく動作しません。
- メンバ名を途中まで入力して，ctrl+"を入力しても補完機能が正しく動作しません。
- アウトライン（折り畳み/展開）は，プロジェクトに登録したファイルのみが対象です。そのため，プロジェクトに登録していないファイルを CS+のエディタで表示しても，アウトライン表示にはなりません。
- Windows 8.1 では，アンチエイリアスが有効になり表示が不鮮明になる場合があります。
- 混合表示モードにて，行番号を指定してジャンプすると，逆アセンブルのコードが挿入されて表示されるため，指定行が画面上に表示されない場合があります。
- 構造体がネストしている場合に，３段目以降は，スマート・エディットの機能は使用できません。また，ツールチップの情報も表示されません。

- 「#ifdef - #endif」の直後のコードでは，スマート・エディットでメンバの候補が表示されません。また，ツールチップが表示されません。
- 「#ifdef - #endif」中の最初の変数については，「#endif」以降のコードで，スマート・エディットでメンバの候補が表示されません。また，ツールチップが表示されません。
- プロジェクトに含まれないファイルを開き，ブックマークを設定し，プロジェクトを閉じます。そして，そのファイルのブックマークの設定を変更した後，プロジェクトを開きなおします。ブックマークダイアログを開いたとき，ブックマークダイアログには，プロジェクトを閉じたときのブックマークが表示され，ソース・ウインドウでは，プロジェクトを閉じた後のブックマークが表示され，差異が生じます。このような場合には，一旦，ファイルを閉じ，再度開いてください。ブックマークダイアログに表示されていたブックマークがソース・ウインドウに表示されます。
- 矩形選択（Alt キーを押下してマウスによる範囲選択）を行い，最終行以降に行が追加されて貼り付けされる場合，貼り付け位置によらず行の先頭から貼り付けられます。貼り付け後に必要なスペースを挿入してください。
- 名前を付けて保存ダイアログにおいてファイルを保存する場合，拡張子を入力しない場合には，ファイルの種類ドロップダウン・リストで選択されている最初の拡張子が自動的に付加されます。ただし，ファイルの種類ドロップダウン・リストで選択されている拡張子および Windows で登録されている拡張子を付加してファイル名を入力した場合，拡張子は付加されません。
  自動的に拡張子が付加されてしまった場合は，エクスプローラー等でファイルをリネームしてください。

5.1.6 PM+から CS+プロジェクトへの移行に関する注意事項

PM+ V6.00/V6.10/V6.11 で作成した CA850 のプロジェクトに対して，ビルド・モードを新規追加した場合，そのプロジェクトを CS+で読み込むと以下のようになります。

1)Debug Build または Release Build が選択されている場合：
新規追加したビルド・モードの情報が変換されません。
2)新規追加したビルド・モードが選択されている場合：
エラーとなります。

回避策として，PM+ V6.20 以上でプロジェクトを開いて保存し，保存後のプロジェクトを CS+で読み込んで下さい。

5.1.7 プロジェクト流用時のデバッグ・ツールの設定に関する注意事項

プロジェクトを流用作成する時，作成するプロジェクトにてデフォルトで選択されているデバッグ・ツールに対してのみ，流用した設定を反映します。

5.1.8 ヘルプに関する注意事項

ヘルプにおいて，検索タブ(S)を表示した状態で閉じ，再度ヘルプを表示し，目次(C)タブを表示した場合，コーディング編とビルド編が表示されない場合があります。

このようになった場合には，目次(C)タブを表示したままヘルプを閉じてから，再度ヘルプを表示しなおしてください。

5.1.9 プロジェクト変換時の注意事項

High-performance Embedded Workshop / PM+ / 旧 CubeSuite を開いた時の〔プロジェクト変換設定〕ダイアログで，プロジェクトの変換先デバイスを切り替えた時，〔プロジェクトの種類〕で選択されていた値を初期値であるコンボボックスの先頭の値へ戻ります。

例えば，デバイスを選択し直すとプロジェクトの種類が先頭の（例えば〔アプリケーション〕）に切り替わります。

5.1.10 High-performance Embedded Workshop プロジェクト変換時の注意事項

High-performance Embedded Workshop のプロジェクトを CS+環境で読み込んだ場合，プロジェクト変換ができずエラーとなったり，ビルド実行時にエラーが発生する場合があります。

(1) CS+用のプロジェクトへ変換ができない

- ルネサス エレクトロニクス社製ツールチェインが使用されていないプロジェクト
- High-performance Embedded Workshop 環境の設定ファイル（tps ファイル）が存在していないプロジェクト（tps ファイルは，High-performance Embedded Workshop 環境で一度開くと自動生成されます。）
  プロジェクト変換前に一度プロジェクトを High-performance Embedded Workshop 環境で開くことで解決できます
- ルネサス エレクトロニクス社製リアルタイム OS の設定ファイル（CFG ファイル）が複数存在しているプロジェクト

(2) CS+用のプロジェクトへ変換はできるが，ビルド実行でエラーが発生

- プレースホルダ（$(TCINSTALL)）を使用しているプロジェクト
  $(TCINSTALL)は，変換後のプロジェクトにそのまま残ります。

  CS+は，$(TCINSTALL)を解釈できません。オプションのパラメータに$(TCINSTALL)を使用していた場合は，そのままオプションに渡されますので意図したビルド結果を得られない可能性があります。（ビルドでエラーが発生するなど）
  $(TCINSTALL)をプロジェクト変換後に，お客様自身で変更してください。

- プレースホルダ（$(WORKSPDIR)）を使用しているプロジェクト
  プロジェクト・ファイル（拡張子 hwp）を指定して変換した場合，「%ProjectDir%¥..」（プロジェクト・フォルダの 1 つ上のフォルダ）に固定で変換します。

  プロジェクト・フォルダの 1 つ上のフォルダにワークスペースがない場合は，正しいフォルダを示さなくなりますので，ビルドでエラーが発生することがあります。
  その場合，プロジェクト変換後に「%ProjectDir%¥..」を，お客様自身で変更してください。

- カスタム・ビルド・フェーズを使用しているプロジェクト

カスタム・ビルド・フェーズは，削除されます。

カスタム・ビルド・フェーズは，ビルド時に実行されなくなります。
よって，カスタム・ビルド・フェーズで生成されたファイル出力を使用している場合はビルドエラーとなる可能性があります。
プロジェクト変換後に，カスタム・ビルド・フェーズのコマンドを，各フェーズの前後実行コマンドに必要に応じて登録してください。

・ カスタムプレースホルダを使用しているプロジェクト
カスタムプレースホルダは変換しません。

CS+は，カスタムプレースホルダを解釈できません。オプションのパラメータにカスタムプレースホルダを使用していた場合は，そのままオプションに渡されますので意図したビルド結果を得られない可能性があります。（ビルドでエラーが発生するなど）
プロジェクト変換後に，カスタムプレースホルダを，お客様自身で変更してください。

(3) その他

(a)$(FILEDIR)は，%FileDir%へ変換します。
変換後，パス編集ダイアログでパス編集するとき，%FileDir%により以下のエラーが発生します。
指定したパスに存在しないフォルダが含まれています。(W0205012)
プロパティを編集する場合，%FileDir%を別のプレースホルダまたはディレクトリに置き換えてください。

(b)$(WINDIR)は，%WinDir%へ変換します。

(c)フォルダの表示順が異なる場合があります。

(d)High-performance Embedded Workshop のプロジェクトでダウンロード・ファイルを指定している場合，
変換後は各デバッグ・ツールのダウンロード・ファイル一覧の 2 番目以降に登録します。

(e)コンパイル・オプション-output=src は，-output=obj（デフォルト）へ変換します。

(f)ライブラリ・プロジェクトを変換時，ライブラリが標準ライブラリをリンクしていた場合は，そのリンク設定を破棄します。（変換ログに出力されます）

(g)ライブラリジェネレータで「既存標準ライブラリファイル指定」を指定していた場合，「標準ライブラリ・ファイル指定なし」に変更します。結果，指定していたライブラリがリンクされません。（変換ログに出力されます）

(h)High-performance Embedded Workshop の「全般」タブのオプションは変換せずに破棄します。

(i)リンカでサブコマンドファイルを指定していた場合，変換後「サブコマンドファイルを使用する」設定を破棄してリンカのオプション設定をデフォルトにします。

(j)リンカの-library，-input，-binary オプションで指定したファイルは，リンク順設定ダイアログのファイルリストに表示しません。リンク順の指定対象外になります。

(k)RTOS のコンフィグレーションファイルは，変換後に「Configuration file」カテゴリ・ノードの下に表示しません。

(l)RTOS オプションは変換せずに破棄します。オプション設定はデフォルトになります。

(m)RTOS プロジェクトのビルド・モードは，プロジェクト変換後「DefaultBuild」になります。
変換後，ビルド・モードを変更してください。

(n)RTOS プロジェクトのアセンブラ出力ファイル（ritbl.obj）のリンク順がプロジェクト変換後
High-performance Embedded Workshop と異なります。

5.1.11 チュートリアルの注意事項

チュートリアルでは，コード生成プラグイン，端子配置プラグイン，プログラム解析プラグインを使用します。プラグイン管理ダイアログで使用するプラグインを有効にして下さい。

5.1.12 CS+の複数起動の注意事項

CS+は，同じホストマシン上で複数起動が可能ですが，次の注意事項があります。

・CS+を複数起動した場合，パソコンのユーザ毎の情報ファイルは最後に書き込んだ情報が保存されます。

・CS+を複数起動した場合，スタック見積もりツール（CallWalker 含む）の情報ファイルは最後に書き込んだ情報が保存されます。

・複数起動した CS+で，同一プロジェクト・ファイルを使用した場合，最後に書き込んだ情報が保存されます。

・複数起動した CS+で，同一プロジェクト・ファイルを使用した場合，同時にビルドしないでください。出力ファイルが同一のためです。

・複数起動した CS+で，同じビルド・ツールのプロジェクト・ファイルを使用した場合，同時にビルドしないでください。ビルド時に使う一時ファイルを置くフォルダ（テンポラリフォルダ）が同一のためです。ただしテンポラリフォルダをオプションで変更すれば使用可能です。（対象ビルド・ツール：CA78K0，CA78K0R，CA850，CX）

5.1.13 CS+を起動する際のオプションの注意事項

CubeSuite+W.exe を起動する際に，オプションを指定できますが，次のオプションを指定しないでください。正常なエディタ機能が動作しなくなります。

- /npall オプション
- /np オプションでエディタを指定

5.1.14 作成したバージョンより古いバージョンで読み込む際の注意事項

作成したプロジェクトを，そのプロジェクトを作成した CS+より古いバージョンの CS+で読み込むと，一部の設定がクリアされる場合があります。これは，新たに追加された機能などの新しいバージョンでのみ認識できる項目が，古いバージョンでは認識できないためです。

5.1.15 マイクロソフト株式会社.NET Framework に関する注意事項

マイクロソフト株式会社製の.NET Framework のバージョンが 4.5.2 未満の場合に，次のメッセージを出力して終了します。

E0200010
本製品の起動に失敗しました。

お使いの PC に Microsoft .NET Framework 4.5.2 以上をインストールしてください。

マイクロソフト株式会社のホームページから.NET Framework 4.5.2 以上をダウンロードしてインストールしてから，CS+ for CA,CX を起動してください。

CS+のアップデート・マネージャでは，.NET Framework のアップデートはできませんので，ご注意ください。

5.2 ビルド・ツールの注意事項

5.2.1 ビルド・モードに関する注意事項

［ビルド・モード設定］ダイアログでビルド・モードを複製，または名前を変更する際，既存のビルド・モードと大文字小文字のみ異なる名前で複製または変更できます。

通常，ビルド・ツールのプロパティの 共通オプション タブ［中間ファイル出力フォルダ］プロパティやリンク・オプション タブ／ヘキサ出力オプションタブ［出力フォルダ］プロパティがビルド・モード名になっており，大文字小文字のみ異なる 2 つのビルド・モードの出力先が重複してしまう点，ご注意ください。

回避するには［中間ファイル出力フォルダ］プロパティ，［出力フォルダ］プロパティで異なるフォルダを設定してください。

5.3 設計ツールの注意事項

5.3.1 パッケージの変更に関する注意事項

端子配置のプロパティでパッケージ名を変更した場合，端子配置図および端子配置表の入力データはクリアされます。

5.3.2 プロジェクト保存に関する注意事項

サブプロジェクトが存在するプロジェクトにて，端子配置図または端子配置表パネルが開いた状態でプロジェクトの保存を行った場合に，プロジェクト・ツリー上の最後のサブプロジェクトの端子配置図，端子配置表が必ず表示されます。

5.4 デバッグ･ツールの注意事項

文中において以下の略称を使用しています。

OCD(シリアル) ：MINICUBE2，E1 エミュレータ(シリアル)，E20 エミュレータ(シリアル)

OCD(JTAG) ：MINICUBE，E1 エミュレータ(JTAG)，E20 エミュレータ(JTAG)

5.4.1 サブプロジェクトの追加に関する注意事項

【対 象】 全デバッグ・ツール，全デバイス共通

メインプロジェクトと異なるデバイスを扱うサブプロジェクトを追加する場合，デバッグ・ツールを切断してから行ってください。

5.4.2 ブートスワップ実行時の注意事項

【対 象】シミュレータ/OCD(JTAG)/OCD(シリアル)，V850 / 78K0 / 78K0R / RL78

ブートスワップ領域にソフトウエア・ブレークを設定した場合，フラッシュ ROM にブレーク用の命令が書き込まれるため，ブートスワップ後もブレーク用の命令が残ってしまいます。

・OCD(JTAG)/OCD(シリアル)の場合：ブレークを設定する場合は，ハードウエア・ブレークを使用してください。
・シミュレータの場合：ブートスワップ領域にブレークを設定しないでください。

5.4.3 スタンバイ･モードの注意事項

【対　象】全デバッグ・ツール，V850 / 78K0 / 78K0R / RL78

STOP モードや HALT モードなどのスタンバイ･モード中に強制ブレークを行った場合や，ステップ実行でスタンバイ・モードに移行する命令を実行した場合，シミュレータとエミュレータ(IECUBE，OCD(JTAG)，OCD(シリアル))では以下のような動作の差があります。

・エミュレータ：強制ブレークによりスタンバイ･モードは解除されます。また，ステップ実行ではスタンバイ・モードに移行しません。
・シミュレータ：強制ブレークによりスタンバイ･モードは解除されません。また，ステップ実行ではスタンバイ・モードに移行します。

どちらの場合とも，強制ブレーク時に PC(プログラム･カウンタ)行は，HALT などのスタンバイ･モード以降命令の次命令でブレークします。このためシミュレータの場合，スタンバイ･モードが解除されているようにも見えます。スタンバイ･モードが解除されているかどうかの確認はステータス･バー行なってください。スタンバイ･モード中の場合，ステータス･バーに"Halt"や"Standby"の表示が出ます。

5.4.4 乗除算器に関する注意事項

【対　象】シミュレータ，78K0

78K0 の命令シミュレーションを行なう場合，乗除算器に対応していません。このため，プログラム内で乗算や除算を行なう場合は，ビルド･ツールのプロパティ･パネルを開き，[コンパイル･オプション]タブで[[乗除算器を使用する]ドロップダウン･リストの「いいえ」を選択してください。

5.4.5 メモリ･バンクに関する注意事項

【対　象】シミュレータ，78K0

78K0 の命令シミュレーションを行なう場合，メモリ･バンク機能に対応していません。

5.4.6 CPU 動作クロックに関する注意事項

【対　象】シミュレータ，78K0R，RL78

・78K0R の命令シミュレーションを行う場合，高速内蔵発振器の周波数は 8MHz 固定です。
・RL78 の命令シミュレーションを行う場合，CPU 動作クロックは RL78/G13 の仕様で動作します。

5.4.7 乗除算器，積和演算器に関する注意事項

【対　象】シミュレータ，78K0R / RL78

78K0R，RL78 の命令シミュレーションを行う場合，乗除算器や積和演算器の使用に関して以下の注意事項があります。

(1) 乗除算器や積和演算器を除算モードで使用した場合，除算処理は 1 クロックで終了します。
(2) 乗除算器や積和演算器を除算モードで使用した場合，除算演算完了割り込みは発生しません。ただし，除算完了を示す SFR は変化します。(乗除算コントロール・レジスタ"MDUC"の DIVST ビットが 0 にな

ります。）

5.4.8 任意区間のトレースに関する注意事項

【対　象】シミュレータ，全デバイス共通

トレース開始イベントからトレース終了イベントまでをトレースする場合，シミュレータではトレース終了イベントがトレース結果として表示されません。このため，シミュレータを使用する場合はトレース終了イベントをトレース･データとして表示させる範囲の 1 行下に設定してください。

5.4.9 任意区間の実行時間測定に関する注意事項

【対　象】シミュレータ，V850 / 78K0 / 78K0R / RL78

タイマ開始イベントからタイマ終了イベントまでを実行時間測定する場合，シミュレータではタイマ終了イベントが時間測定結果に含まれません。このため，シミュレータを使用する場合はタイマ終了イベントを時間測定する区間の 1 行下に設定してください。

5.4.10 メモリ表示パネルでの最大アドレス空間表示について

【対　象】OCD(シリアル)/IECUBE，78K0

メモリパネル等でデバイス最大サイズの内部 ROM，内部高速 RAM，内部拡張 RAM にアクセスするには，メモリ・サイズ切り替えレジスタ(IMS)と内部拡張 RAM サイズ切り替えレジスタ(IXS)をフック処理に設定してください。

5.4.11 リターン実行，コール・スタック表示について

【対　象】OCD(JTAG)／OCD(シリアル)／IECUBE，78K0R,RL78

エディタパネルで(ソース・モードで)ステップ実行した場合，デバッグ・ツールは PSW レジスタの NP，EP，ID フラグをもとに割り込み処理中かどうかを判断しています。そのため，多重割り込みを使用している場合など，上記フラグやレジスタを変更した場合は，リターン実行や，コール・スタックの表示が正常に行なわれない場合があります。

5.4.12 サブプロジェクトの追加について

【対　象】全デバッグ・ツール，全デバイス

デバッグ・ツール接続中にサブプロジェクトを追加すると，ダウンロード等に失敗することがあります。サブプロジェクトの追加は，デバッグ・ツール切断中にしてください。

5.4.13 フラッシュ・オプションの設定について

【対　象】OCD(JTAG)，V850E2M

下記フラッシュ・オプションで以下に示すビットは 1 固定になります。0 を書き込みたい場合は，フラッシュ・プログラマをお使いください。

・オンチップ・デバッグ・セキュリティ ID のビット 95(セキュリティ・ロック信号解除)

・オプション・バイト 0 のビット 31(デバッグ・インタフェース接続禁止ビット)

5.4.14 スタック・トレース表示についての注意

【対　象】全デバッグ・ツール，78K0

スタック・トレース表示機能は，スタックにフレーム・ポインタ(HL)を Push しない関数(noauto，norec 関数等)がある場合やメモリ･バンクを使っている場合には，main 関数まで正しく表示されないことがあります。

また，スタックにフレーム・ポインタ(HL)を Push しない関数(noauto，norec 関数等)や，メモリ・バンク関数からリターン実行した場合，フリーラン状態になることがあります。

5.4.15 メモリ・バンク内でステップ・インした際の注意

【対　象】全デバッグ・ツール，78K0

メモリ・バンク内のユーザ定義ライブラリ関数またはメモリ・バンク内のデバッグ情報なし関数にソース・レベルでステップ・インした場合，バンク切り替えライブラリ内でブレークします。

5.4.16 ローカル変数の表示に関する注意

【対　象】全デバッグ・ツール，78K0

スタック・トレース・パネルで，カレント PC のスコープ外のローカル変数は，正しく表示できません。

5.4.17 逆アセンブル・ウインドウについての注意

【対　象】全デバッグ・ツール，78K0

コモン領域内の命令を逆アセンブル・ウインドウで表示する際，表示される命令にメモリ・バンク領域内のシンボルが使用されていると，異なるバンクのシンボルを表示してしまう場合があります。

5.4.18 ブレークポイントの設定等が不正になる注意

【対　象】全デバッグ・ツール，全デバイス

関数名や変数名を，先頭のアンダー･バーの有無などで使い分けている場合，デバッガが誤認識してしまい，シンボル変換や，ブレークポイントの設定が不正になる場合があります。

例えば_reset と__reset という 2 つの関数が存在していた場合などが該当します。

5.4.19 クロック発生回路に関する注意事項

【対 象】シミュレータ，78K0

クロック発生回路は 78K0/Kx2 の仕様でシミュレーションしています。

5.4.20 データ・フラッシュに関する注意事項

【対 象】シミュレータ，RL78 / 78K0R

データ・フラッシュには対応していません。

5.4.21 コード・フラッシュに関する注意事項

【対 象】シミュレータ，RL78 / 78K0R

フェッチ可能なコード・フラッシュ領域，RAM 領域などの最終アドレスの 4 バイトはフェッチできません。

フラッシュは対応していません。

「ノン・マップ領域へのアクセスにより停止しました。」のエラーが発生します。

（例）コード・フラッシュ領域が 0x0～0x1ffff の場合，0x1fffc～0x1ffff が制限に該当します。

5.4.22 パイプラインに関する注意事項

【対 象】シミュレータ，RL78 / 78K0R

パイプラインには対応していません。

5.4.23 ブレーク時のターゲット・システム電源について

【対　象】OCD(JTAG)/OCD(シリアル)，78K / RL78 / V850 /V850E2M

ブレーク時にターゲット・システムの電圧を下げないようにしてください。ブレーク中に低電圧検出回路（LVI），パワーオン・クリア（POC）によるリセットが発生した場合，デバッガの不正動作や通信エラーの原因となることがあります。

また，ターゲット電源 OFF のエミュレーション中でのブレークもこれに該当します。

5.4.24 デバッガ・プロパティの注意事項

【対　象】OCD（シリアル），V850

プロパティパネルの[デバッグ・ツール設定]タブからオープンブレーク機能の有効/無効は設定できません。

Python コンソールから設定してください。

5.4.25 GHS コンパイラで生成したロードモジュール使用時の注意事項

【対　象】IECUBE，OCD(JTAG/シリアル)，シミュレータ，V850

定義されている変数を未定義の関数名でスコープ指定した時，エラーにならない場合があります。

未定義の関数名でのスコープ指定を行わないでください。

5.4.26 オプション・バイト書き換えに関する注意事項

【対　象】OCD（シリアル），RL78

メモリ・パネル等でオプション・バイトを書き換えた場合は CPU をリセットしてください。

デバッガが正常に動作しなくなる場合があります。

5.4.27 トレースパネルの DMA 表示に関する注意事項

【対　象】IECUBE, RL78, 78K0R

DMA による SFR/メモリへのアクセスがあった場合，トレースパネルに“DMA”の文字列が表示されません。（DMA アクセスによる“アドレス”，“データ”は問題なく表示されます。）

5.4.28 トレースパネルのブレーク要因表示に関する注意事項

【対　象】OCD（シリアル），RL78, 78K0R

プログラムが停止したとき，トレースパネルのブレーク要因が正しく表示されない場合があります。

ブレーク要因は，メイン・ウインドウのステータス・メッセージで確認してください。

5.4.29 周辺シミュレーション機能に関する注意事項

【対　象】シミュレータ，V850ES/Sx2, V850ES/Jx2, V850ES/Fx3, 78K0R/Lx3, 78K0R/Kx3, 78K0R/Ix3, 78K0/Kx2, RL78/G10

対象デバイスのユーザーズ･マニュアルで禁止されている IOR／SFR の設定を，ユーザ・プログラムまたはデバッグ操作で行った場合，CS+が正しく動作しない場合があります。

5.4.30 周辺シミュレーション機能の HALT モードに関する注意事項

【対　象】シミュレータ，　78K0R/Lx3, 78K0R/Kx3, 78K0R/Ix3, RL78/G10

シミュレータでは，ユーザ・プログラムが HALT モードに移行後，次の２つのいずれかのデバッグ操作により HALT モードが解除されます。

・プログラム停止中に，プロパティ パネルの[接続用設定]タブのメイン・クロック周波数またはサブ・クロック周波数を変更した時。

・クロック発生回路を制御するレジスタをデバッグ操作で書き込みを行った時。

ただし、RL78/G10 の場合、クロック発生回路の HOCODIV レジスタに同じ値の書き込みを行った時は該当しません。

5.5　解析ツールの注意事項

5.5.1　解析グラフパネルに関する注意事項

・　値の推移グラフにて，デバッグ・ツールにシミュレータを指定している場合，IOR／SFR のリアルタイム・サンプリング方式はサポートしていません。

5.6　Python コンソールの注意事項

5.6.1　日本語入力に関する注意事項

Python コンソールでは日本語入力機能を有効にする事ができません。日本語を入力する場合は，外部エディタ等で作成しコピーし貼り付けてください。

5.6.2　プロンプト表示に関する注意事項

Python コンソールのプロンプトが>>>であるところが>>>>>>というように複数表示される場合や>>>の後に結果が表示され，キャレットの前に>>>がない場合があります。このような状態でも継続して関数を入力することが可能です。

5.6.3　フォルダやファイルへのパスに関する注意事項

IronPython では，¥(バックスラッシュ)を制御文字として認識します。例えば，先頭が t で始まるフォルダ名やファイル名の場合¥t で TAB 文字と認識してしまいます。これを回避するには次のように，""(パス指定)の前に r を記載してください。IronPython は""の中がパスと認識します。

(例)　r"c:¥test¥test.py"

なお，パスの指定には¥(バックスラッシュ)ではなく/(スラッシュ)も使用可能です。

5.6.4　ロード･モジュールがないプロジェクトのスクリプト実行に関する注意事項

ロード・モジュール・ファイルがないプロジェクトを使用して起動オプションでスクリプト指定した場合，もしくはプロジェクト・ファイル名.py をプロジェクト・ファイルと同じフォルダにおいてある場合は，通常プロジェクト読み込み後に自動的にスクリプトを実行しますが，ロード・モジュール・ファイルがない場合は

実行しません。

5.6.5 強制終了に関する注意事項

無限ループしているようなスクリプトを実行中に以下の操作を行うと，強制的に関数の実行を終了させるため，関数の実行結果がエラーになる場合があります。

1. Python コンソールのコンテキスト・メニューの「強制終了」や Ctrl+D で強制終了
2. 複数のプロジェクトをもつプロジェクトでアクティブ・プロジェクトを変更した場合

5.6.6 強制停止に関する注意事項

コンテキスト・メニューの[強制停止]を実行した場合，実行中のスクリプトや関数を強制停止しますが，[強制停止]した時点で実行が開始していない Hook 関数や Callback 関数がある場合は，[強制停止]後順次実行します。

5.6.7 ビルド中の Python コマンドの実行に関する注意事項

ビルド中に Python コマンドを使用しないでください。

第6章　制限事項

本章では，制限事項について説明します。

6.1　デバッグ・ツールの制限事項

文中において，以下の略称を使用しています。

OCD(シリアル)　：MINICUBE2，E1 エミュレータ(シリアル)，E20 エミュレータ(シリアル)

OCD(JTAG)　　：MINICUBE，E1 エミュレータ(JTAG)，E20 エミュレータ(JTAG)

OCD(LPD)　　：E1 エミュレータ(LPD)，E20 エミュレータ(LPD)

6.1.1　デバッグ･ツールの制限事項一覧

No.	対象ツール	対象デバイス	制限事項	備考
1	OCD(シリアル) OCD(JTAG) OCD(LPD)	V850E2M V850E2S	フラッシュ･オプション設定に関する制限事項	
2	OCD(シリアル)	RL78	RL78/G14でサブクロック動作時の 高速オンチップオシレータ停止に関する制限事項	CS+ V2.02.00で制限事項解除
3	OCD(シリアル)	RL78 (RL78/G10除く)	データ・フラッシュ・メモリ書き換えに関する制限事項	CS+ V3.00.00で制限事項解除

6.1.2　デバッグ・ツールの制限事項詳細

No.1　フラッシュ・オプション設定に関する制限事項

【対　象】OCD(シリアル), OCD(JTAG)，OCD(LPD)　V850E2M, V850E2S

【内　容】フラッシュ・オプション設定プロパティのセキュリティ設定とブート・ブロック・クラスタ設定にどのような値を設定しても無効になります。

【回避策】回避策はございません。

第7章　ドキュメント訂正

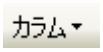
本章では，CS+のドキュメントの訂正について説明します。

7.1　エディタに関するドキュメント訂正事項

エディタに関するドキュメントの訂正について説明します。エディタの説明は，各種コーディング編，デバッグ編に記載があります。

7.1.1　ツールバーの説明追加

【追　加】

	デバッグ・ツール接続時に，通常表示モードと混合表示モードを切り替えます。 通常表示ではソース・プログラムを表示します。 混合表示ではソース・プログラムとアセンブル・コードを表示します。
	デバッグ・ツール接続時の混合表示モードにおいて，ステップ実行を行った際の動作について，ソース・レベルか，アセンブラ・レベルかを切り替えます。 ソース・レベルの場合には，プログラムカウンタ（PC）を示す表示が，ソース・プログラム行を示します。 アセンブラ・レベルの場合には，プログラムカウンタ（PC）を示す表示が，アセンブル・コード行を示します。
	デバッグ・ツール接続時の混合表示モードにおいて，現在のプログラムカウンタ（PC）位置を表示します。
	［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。
	［関数へジャンプ］を実行する前の位置へ戻ります。
	カラムの表示／非表示を切り替えます。クリックすることにより切り替えできるエリアの項目が表示されます。

7.1.2 カラム・ヘッダの説明追加

【追　加】

各カラムでは，次の情報を表示します。

カラム・ヘッダ表示	説明
行	表示しているファイルの行番号を表示します。
（表示なし）	編集状況に応じて色表示を行います。 黄色：新規または変更したが，保存していない行。 緑色：新規または変更した後，ファイルを保存済みの行。 （当カラムは，混合表示モード時以外に表示されます）
（表示なし）	ソース・ファイルの更新日時がロード・モジュール・ファイルの更新日時より新しい場合，色表示を行います。 ビルドを行った後，再度ロード・モジュール・ファイルをダウンロードした際に，このエリアの表示はクリアされます。 （当カラムは，デバッグ・ツール接続時の通常表示モード時に表示されます）
	カバレッジ機能を有効としている場合，コード・カバレッジ測定結果により色表示を行います。 （当カラムは，デバッグ・ツール接続時に表示されます）
アドレス	マイクロコントローラ上のアドレスを表示します。 （当カラムは，デバッグ・ツール接続時に表示されます）
命令コード	命令コードを表示します。 （当カラムは，デバッグ・ツール接続時の混合表示モード時に表示されます）
ラベル	ラベルを表示します。 （当カラムは，デバッグ・ツール接続時の混合表示モード時に表示されます）
	アドレス表示のある行において，タイマ／トレースなどのイベントをコンテキスト・メニューで設定する領域です。 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。 トレース・イベント有効 トレース・イベント無効 アクション・イベント有効 アクション・イベント無効 （当カラムは，デバッグ・ツール接続時に表示されます）

	デバッグ・ツール未接続時： ブックマークの表示，エラーまたはワーニングを示すマークの表示を行う領域です。 エラーまたはワーニングを示すマークは，プロジェクトに登録されているソース・ファイルにのみ表示します。プロジェクトを開いた後には表示せず，ビルド実行後に表示します。クリーンの実行の場合には，エラーまたはワーニングを示すマークは全て消します。なお，連続してビルドを実行した場合，初回に出ていたワーニングが2回目以降のビルドには出ない場合は，エディタパネル上でもマークを表示しません。エラーまたはワーニングを示すマークは，ソース・ファイルの編集（行番号の変化）には追従しません。エラーまたはワーニングを示すマーク上にマウス・カーソルを重ねると，コンパイラ／アセンブラが出力したメッセージをポップアップ表示します。 エラーが存在 ワーニングが存在 ブックマーク デバッグ・ツール接続時： ブレークポイントの状態表示，カレントPC位置，ブックマークの表示，アドレス・マークの表示を行う領域です。アドレス・マークのある行において，ブレークポイントを設定する領域です。コンテキスト・メニューで操作しますが，ブレークポイントを設定／解除する場合には，マウスの左クリックで操作できます。 現在設定しているブレークポイントがある場合，そのブレークポイント設定行にイベント・マークを表示します。また，このエリアでは，アクティブなパネル中にカレントPC位置(PCレジスタ値)が含まれる場合，カレントPC位置を示すマークも表示します（停止状態時のみ）。 ブレークポイント有効 ブレークポイント無効 カレントPC位置 ブックマーク アドレス・マーク

7.1.3 ドラッグ＆ドロップの説明追加

【追　加】

シンボルをドラッグ＆ドロップすることにより，ウォッチパネルや解析グラフ（値の遷移）へのシンボル登録が可能です。

7.1.4 強調表示に関する説明追加

【追　加】

・カレント行のハイライト表示

カレント行（キャレットが存在する行）に対して，四角で囲んで強調表示をします。
本機能を有効にするには，［オプション］ダイアログの［テキスト・エディタ］ページにて設定します。

・括弧の強調表示

キャレット位置にある括弧と，それに対応する括弧について，強調表示を行います。ファイルの種類により強調表示をする括弧は次の通りです。

ファイルの種類	対応する括弧
C / C++言語ファイル	(と) , { と }, [と]
Python言語ファイル	(と) , { と }, [と]
HTML / XML言語ファイル	< と >

対応する括弧を検索する場合に，コメント中の括弧や，文字定数／文字列／文字列定数中の括弧を考慮できません。そのため，これらが存在する場合には実際に対応する括弧とは異なる括弧が強調表示されます。

7.1.5 拡大／縮小機能に関する説明追加

【追　加】

CS+のツールバーの拡大／縮小設定により，エディタの表示も連動します。拡大率の変更は，フォーカスのあるパネルに対しておこないます。　パネルを閉じて，再度，同じファイルを開いた場合は，拡大率を 100%として開きます。通常表示モードと，デバッグ・ツール接続時の混合表示モードでは，それぞれの拡大率で表示します。

7.1.6 スマート・エディットの説明追加

【追　加】

スマート・エディット機能とは，コーディング中に関数情報や変数情報，関数の引数情報を表示し，入力を補完する機能です。

本機能は，V850E2 ファミリのプロジェクトのみサポートしています。【V850E2】

スマート・エディット機能では，次の情報を表示します。

- C 言語および C++言語におけるグローバル関数
- C 言語および C++言語におけるグローバル変数
- C++言語におけるクラスメンバー関数
- C++言語におけるクラスメンバー変数
- C++言語における複数のオーバーロード関数表示

スマート・エディット機能での関数，変数の候補の表示をさせるには，次の様に操作してください。

(1) 自動で表示するケース

- C 言語および C++言語において‘. ’を入力した時点で左辺に対して該当するメンバがある場合キャレット位置に自動的に候補を表示します。
- C 言語および C++言語において‘->’を入力した時点で左辺に対して該当するメンバがある場合キャレット位置に自動的に候補を表示します。
- C++言語において“::”を入力した時点で左辺に対して該当するメンバがある場合，キャレット位置に自動的に候補を表示します。
- メソッド（関数）の，‘（’を入力した時点で‘（’の左辺に該当メソッド（関数）がある場合には，引数候補を自動的に表示します。

(2) キー，マウスで操作することにより表示させるケース

- キーボードより Ctrl+Space を入力した場合，キャレット位置にすべての候補を表示します。なお，候補がひとつしかない場合，候補表示を行わず，該当する文字列を貼り付けます。
- メソッド（関数）の引数位置にキャレットがある状態で，Ctrl+Shift+Space を入力した場合，引数リストを表示します。
- メソッド（関数）および変数にマウス・カーソルをあてるとメソッド（関数）および変数の情報をツールチップで表示します（デバッグ・ツール未接続の状態の時のみ）。

スマート・エディットの候補等の表示は，次の操作により表示されなくなります。

- ESC キーの入力
  スマート・エディットの関数，変数候補表示中に ESC キーを入力した場合，候補表示を中止します。

- 候補リストで何も選択していない状態での英数字以外の入力
  英数字以外を入力した時点で関数，変数の候補を何も選択していない場合，候補表示を中止します。
- 候補リスト選択している状態での英数字以外の入力
  英数字以外を入力した時点で関数，変数候補を選択していた場合，選択していた候補を貼り付けます。

スマート・エディット機能を有効にするには，次の設定をしてください。

- ［オプション］ダイアログの［テキスト・エディタ］ページで，「スマート・エディット」をチェック･オンに設定してください。
- ビルド・ツールのプロパティで，クロスリファレンスを出力する設定にしてください。
- クロスリファレンスの情報を使用して候補を表示するので，ビルドを実行し完了させてください。
  なお，ビルド時にエラーが発生した場合，エラー発生前のクロスリファレンス情報が存在すれば，それを使用します。（ビルド・ツールのクロスリファレンス出力を無効にした場合，クロスリファレンス情報がクリアされるため，スマート・エディット機能が使用できなくなります。）

スマート・エディット機能では，次の点に注意してください。

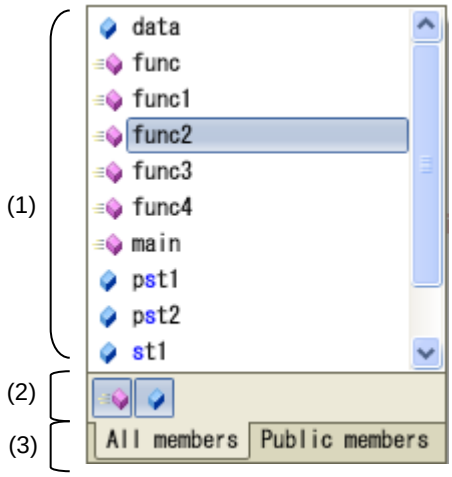
- クロスリファレンス情報が必要です。
- デバッグ・ツール接続状態で混合表示モードを使用している場合，スマート・エディット機能は使用できません。
- マクロ定義は対応していません。
- ローカル変数は対応していません。
- typedef で定義された型のメンバの候補表示には対応していません。【V850E2】
- C++の namespace およびサブクラス（クラス中で宣言したクラス）は，サポートしていません。
- 関数中に構造体宣言，共用体宣言および C++のクラス宣言を行った場合，宣言以降関数内で候補表示は行いません。
- const，mutable の属性表示は行いません。
- C++において，“ (*class) ” または“(*this)”入力を行っても関数，変数候補の表示は行いません。
- C++において，配列宣言したクラスでは，関数，変数候補の表示は行いません。
- C++において，左辺にクラスを指定し，メソッド名を途中まで入力した状態で，Ctrl+Space により関数，変数候補を表示させた場合，クラス内の関数，変数候補を表示せずにグローバルな関数，変数候補を表示します。
- 変数のサイズに影響するコンパイル・オプションを設定した場合，表示される変数の型が実際の宣言と異なる場合があります。
- ツールチップ表示は，デバッグ・ツール接続時には表示しません。
- ツールチップ表示は，C++の friend 属性はサポートしていません。
- ツールチップ表示は，const，static，volatile，virtual 属性は表示しません。
- ツールチップ表示は，ヘッダファイル中で宣言されている構造体，共用体および C++のメンバ関数については，表示しません。
- ツールチップ表示は，ビルド・ツールが CX の場合，共用体の宣言において，共用体タグ名にマウス・カーソルを置いても表示しません。

7.1.7 スマート・エディットの表示の説明追加

【追　加】

関数，変数の候補表示

スマート・エディット機能で関数，変数の候補を表示します。



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］

(1) 自動で表示するケース

・ C 言語および C++言語において‘．’を入力した時点で左辺に対して該当するメンバがある場合キャレット位置に自動的に候補を表示

・ C 言語および C++言語において‘->’を入力した時点で左辺に対して該当するメンバがある場合キャレット位置に自動的に候補を表示

・ C++言語において“::”を入力した時点で左辺に対して該当するメンバがある場合，キャレット位置に自動的に候補を表示

(2) キーを押すことにより表示させるケース

・ キーボードより Ctrl+Space を入力した場合，キャレット位置にすべての候補を表示します。なお，候補がひとつしかない場合，候補表示を行わず，該当する文字列を貼り付けます。

［各エリアの説明］

(1) 候補表示エリア

関数および，変数候補を，アルファベット順に表示します。

スマート・エディット表示中にキー入力を行った場合，一致する文字列を強調表示します。

候補表示の先頭には，以下のアイコンを表示します。

アイコン	説明
	候補がtypedefであることを示します
	候補が関数であることを示します
	候補が変数であることを示します
	候補がクラスであることを示します
	候補が構造体であることを示します

	候補が共用体であることを示します
{ }	候補が名前空間であることを示します
	候補がProtectedメンバであることを示します
	候補がPrivateメンバであることを示します

候補表示のリストから候補を選択し，Enter キーもしくは TAB キーを入力することにより，キャレット位置に候補の文字列を挿入します。

(2) ツールバーエリア

関数および変数の候補の表示／非表示を切替えます。

アイコン	説明
	選択状態： メソッド（関数）候補を表示します 非選択状態：メソッド（関数）候補を表示しません
	選択状態： 変数候補を表示します 非選択状態：変数候補を表示しません

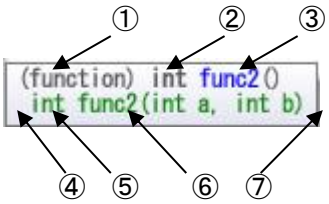
(3) タブエリア

表示するメンバを切替えます。

タブ名	説明
All members	すべての候補を表示します
Public members	Public属性の候補のみ表示します

詳細候補表示

スマート・エディット機能で，関数，変数の詳細候補を表示します。



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］

関数，変数の候補表示にて，候補中のメソッド（関数），変数を選択

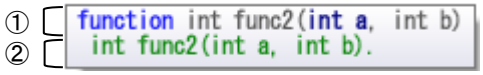
［各エリアの説明］

項番	説明	
①	種別表示	選択中の項目がメソッド（関数）か変数かを表示します。 （function）：メソッド（関数） （variable）：変数
②	型表示	関数もしくは変数の型を表示します

③	名称表示	関数もしくは変数の名称を表示します。
④	属性表示	属性（public，protected，private）を表示します。 この項目は属性が定義されていない場合表示しません。
⑤	型表示	関数もしくは変数の型を表示します。
⑥	名称と引数表示	関数もしくは変数の名称を表示します。関数の場合，引数も表示します
⑦	Overload情報表示	Overloadされている個数を表示します （例）　(+1 overloads)

引数の候補表示

スマート・エディット機能で，引数の候補を表示します。

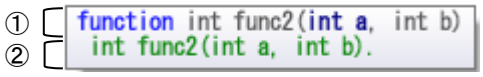


ここでは，次の項目について説明します。
- ［オープン方法］
- ［各エリアの説明］

［オープン方法］

(1) 自動で表示するケース
・ メソッド（関数）の，‘(’を入力した時点で‘(’の左辺に該当メソッド（関数）がある場合には，引数候補を自動的に表示します。
(2) キーを押すことにより表示させるケース
・ メソッド（関数）の引数位置にキャレットがある状態で，Ctrl+Shift+Space を入力した場合，引数リストを表示します。

［各エリアの説明］



項番	説明	
①	候補表示	候補名と引数を表示します。
②	候補属性示	候補の属性表示を表示します。

引数候補を表示する際，キャレット位置の引数を強調表示します。（上記の図の例では，第一引数が強調表示されています）

ツールチップ表示

スマート・エディット機能で関数，変数の情報を表示します。

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]

関数，および，変数にマウス・カーソルをあてる

[各エリアの説明]

マウス・カーソルをあてた対象に応じて，以下のアイコンを表示します。

アイコン	説明
	対象がtypedefであることを示します
	対象が関数であることを示します
	対象が変数であることを示します
	対象がクラスであることを示します
	対象が構造体であることを示します
	対象が共用体であることを示します
{ }	対象が名前空間であることを示します
	対象が列挙型であることを示します

クラス型／構造体型／共用体型／列挙型の変数にマウス・カーソルをあてた場合，ツールチップにメンバの一覧を表示します。

- クラス型／構造体型／共用体の場合，メンバの型とメンバ名を表示します。
- クラス型の場合に，メンバがメソッド（関数）の場合は，返却値の型とメソッド名（関数名）を表示して，メソッド名（関数名） の最後に，「 ‘(’ ‘)’ 」を付加します。
- 列挙型の場合は，メンバ名のみを表示します。
- 1 行に 1 つのメンバを表示して，表示するメンバは，20 個までです。20 個を超える場合は，21 個目に，「…」を表示します。
- 表示するメンバの順番は，ソース・ファイルに定義されている順番です。

7.1.8 混合表示の説明追加

【追　加】

混合表示とは，デバッグ作業時，通常であればソース・プログラムのみの表示であるが，アセンブラ・コードも合わせて表示する機能です。

混合表示の設定は，ツールバーで行います。

混合表示モードで開いている［エディタ］パネルを閉じた後，再度同じファイルを開いた場合，混合表示モードで表示せずに通常表示モードで表示します。また，デバッグ・ツール切断後，再度，デバッグ・ツールに接続した場合も通常表示モードで表示します。

混合表示モードを使用する際には，次の点にご注意ください。

- 編集することはできません。
- 切り取り，貼り付け，削除，やり直し，置換，アウトライン，インデント操作等，内容を変更する機能は使用できません。
- すべて選択機能を使用することができません。
- パネルを分割表示することはできません。

以下のいずれかの方法で，混合表示の内容を，テキスト・ファイル，または，CSV ファイルへ保存することができます。

- 混合表示しているエディタパネルをアクティブにして，CS+の［ファイル］－［名前をつけて混合表示を保存］メニューを選択
- 混合表示しているエディタパネルのコンテキスト・メニューの「名前をつけて混合表示を保存」メニューを選択

7.1.9 リサイクル・モードの説明追加

【追　加】

リサイクル・モードとは，デバッグ作業時，ステップ実行等で，プログラム・カウンタ（PC）が複数のソース・ファイルをまたいで移動する場合，複数のソース・ファイルのエディタパネルを表示するところを，1 枚のエディタ パネルで順に複数のソース・ファイルを表示するモードです。

リサイクル・モードの設定は［ツール］－［オプション］の［テキスト・エディタ］パネルにて行います。

既に該当するソース・ファイルのエディタパネルが開いている場合，リサイクル・モードのウインドウには表示せずに，既に開いているソース・ファイルのエディタパネルを表示します。

リサイクル・モード時もエディタパネル上で編集が可能である。編集を行ったリサイクル・モードのエディタパネル上にプログラム・カウンタ（PC）が存在する状態で，プログラムを実行した場合，編集を行ったエディタパネルは，リサイクル・モードを解除し，新たに開くエディタパネルをリサイクル・モードとして開きます。

リサイクル・モードで動作するエディタパネルを閉じた場合，次にアクティブになるエディタパネルがリサイクル・モードで動作します。

7.1.10 分割バーの説明変更

【変更前】

縦と横の分割バーを使うことにより，エディタ パネルを分割して表示することができます。分割の上限は，**4 分割**までです。

【変更後】

縦と横の分割バーを使うことにより，エディタ パネルを分割して表示することができます。分割の上限は，**縦 2 分割，横 2 分割**までです。

7.1.11 関数へジャンプの説明追加

【追　加】

関数へジャンプ機能は，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) ビルド・ツールが CA78K0R，CA850，CA78K0 の場合

・対象がアクティブ・プロジェクト内の関数である。

・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（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）。

注意：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できません。

(b) ビルド・ツールが CX の場合

・デバッグ・ツールに接続していない場合
- アクティブ・プロジェクトの種類が“アプリケーション”である
- ［ダウンロードするファイル］の 1 番目に指定されたファイルに対象の関数が定義されている
- 上記ファイルにシンボル情報が存在する
- 対象の関数がグローバル関数である

・デバッグ・ツールに接続し，ロード・モジュールをダウンロードしている場合
- ダウンロードしたロード・モジュール内にシンボル情報が存在する
- プログラムカウンタ(PC)の指すアドレスから呼び出し可能な関数である
  ※例えば，PC の指すアドレスのファイル以外で定義した static 関数へはジャンプできない。

・C++言語プログラムで関数へジャンプする機能を使う場合は，関数を特定するための以下の注意事項があります。
- 選択された関数名の文字列で関数が特定できない場合，ジャンプができないか，別の同名関数へジャンプする可能性があります。

(1)クラスのメンバ関数
対象の関数の所属するクラス名を含む必要があります。
また，同名で引数の異なる関数が存在する場合は，引数の型名も含める必要があります。
例："memfunc" ： ジャンプできません
"Class::memfunc(short)" ： ジャンプできます

(2)名前空間内に定義した関数
対象の関数の所属する名前空間を全て含む必要があります。
また，同名で引数の異なる関数が存在する場合は，引数の型名も含める必要があります。
例："func" ： ジャンプできません
"Namespace1::Namespace2::func(int)" ： ジャンプできます

(3)テンプレート関数
コンパイラが生成した関数の引数の型名を含める必要があります。
例："template" ： ジャンプできません
"template(int, short)" ： ジャンプできます

(c) 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である。
・シンボル情報を持つファイルが［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（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）。

注意：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できません。

7.1.12 コンテキスト・メニューの説明追加

【追　加】

解析グラフに登録	解析グラフの値の遷移 タブに変数を登録します

7.1.13 Print Preview ダイアログの説明追加

【変更前】

（1） プレビュー エリア
印刷イメージをプレビュー表示します。

【変更後】

（1） プレビュー エリア
印刷イメージをプレビュー表示します。**デバッグ・ツール未接続時,デバッグ・ツール接続時（通常表示モード），デバッグ・ツール接続時（混合表示モード）により表示が異なります。なお，アウトラインで折りたたんでいる行についてもアウトラインを開いた状態で印刷プレビューを表示します。**

（a）デバッグ・ツール未接続時
次の様に表示します。
左端：行番号
ただし，エディタ パネル上で非表示に設定している場合，表示しません。
ページヘッダ：ファイル名（フルパス）
ページフッタ：ページ番号

（b）デバッグ・ツール接続時（通常表示モード）
次の様に表示します。
左端：行番号，アドレス
ただし，エディタ パネル上で非表示に設定している場合，表示しません。
ページヘッダ：ファイル名（フルパス）
ページフッタ：ページ番号

（c）デバッグ・ツール接続時（混合表示モード）

次の様に表示します。

左端：行番号，アドレス，命令コード

ただし，エディタ パネル上で行番号を非表示に設定している場合，表示しません。

ページヘッダ：ファイル名（フルパス）

ページフッタ：ページ番号

7.1.14 サイズが大きなファイルの表示の説明追加

【追　加】

ファイルサイズが 24MB を超えるファイル（以後，巨大ファイル）を開く場合，メッセージ Q2000007 を表示しますので，エディタパネルの次の機能を無効にするかどうかを選択してください。

・シンタックスの色付け
・スマート・エディット
・コードのアウトライン

選択した状態は，プロジェクトを開いている期間のみ有効です。また，本確認メッセージは，プロジェクトを開いてから初めて巨大ファイルを開いた場合のみです。2 つ目以降の巨大ファイルを開いた場合には，初回に選択した設定でエディタパネルを開きます。

7.2　フック処理に関するドキュメント訂正事項

フック処理に関するドキュメント訂正について説明します。フック処理の説明は各種デバッグ編に記載があります。

フック処理の設定は，デバッグ・ツールプロパティパネルの［フック処理設定］タブで行います。

フック処理を使用すると，ダウンロード前，実行開始前，リセット後等様々なタイミングで I/O レジスタの設定，CPU レジスタの設定，python スクリプトの実行が行え，以下が実現できます。

(1)プログラム開発中で，マイコンの I/O レジスタの設定プログラムが未完成でも，実行開始前に I/O レジスタ設定を行うことによりデバッグを行うことができます。
(2)ダウンロード前に I/O レジスタを設定することにより，ダウンロードを高速に行うことができます。
(3)ダウンロード前に I/O レジスタを設定することにより外部 RAM へのダウンロードが容易に行えます。

デバッガのフック処理から Python スクリプトを実行する場合，以下のコマンドが記載可能です。

```
debugger.Register.GetValue
debugger.Register.SetValue
debugger.Memory.GetValue
debugger.Memory.SetValue
```

それ以外の Python コマンドを使用したい場合 Python コンソールの Hook コマンドを使用してください。

7.3　ビルド編のドキュメント訂正事項

ビルド編(資料番号：R20UT0783JJ0100, R20UT2143JJ0100)のドキュメントの訂正について説明します。

7.3.1　リンクオプションタブの説明変更（R20UT2143JJ0100 のみ）

【場　所】　232 ページ

【変更前】 (5) ［デバイス］
オンチップ・デバッグを設定する

【変更後】 (5) ［デバイス］
オンチップデバッグの許可/禁止をリンク・オプションで設定する

7.3.2 スタック見積もりツールの注意事項の説明追加

【場 所】 351 ページ →解析対象関数
【追加後】 したがって，ユーザが記述したアセンブラ・ソース・ファイル，およびユーザが作成したライブラリ・ファイルに内包されている関数については，解析対象外となるため，スタックサイズ変更ダイアログを用いて該当情報を設定する必要があります。
また，割り込み関数も解析対象外となるため，スタックサイズ変更ダイアログを用いて該当情報を設定する必要があります。

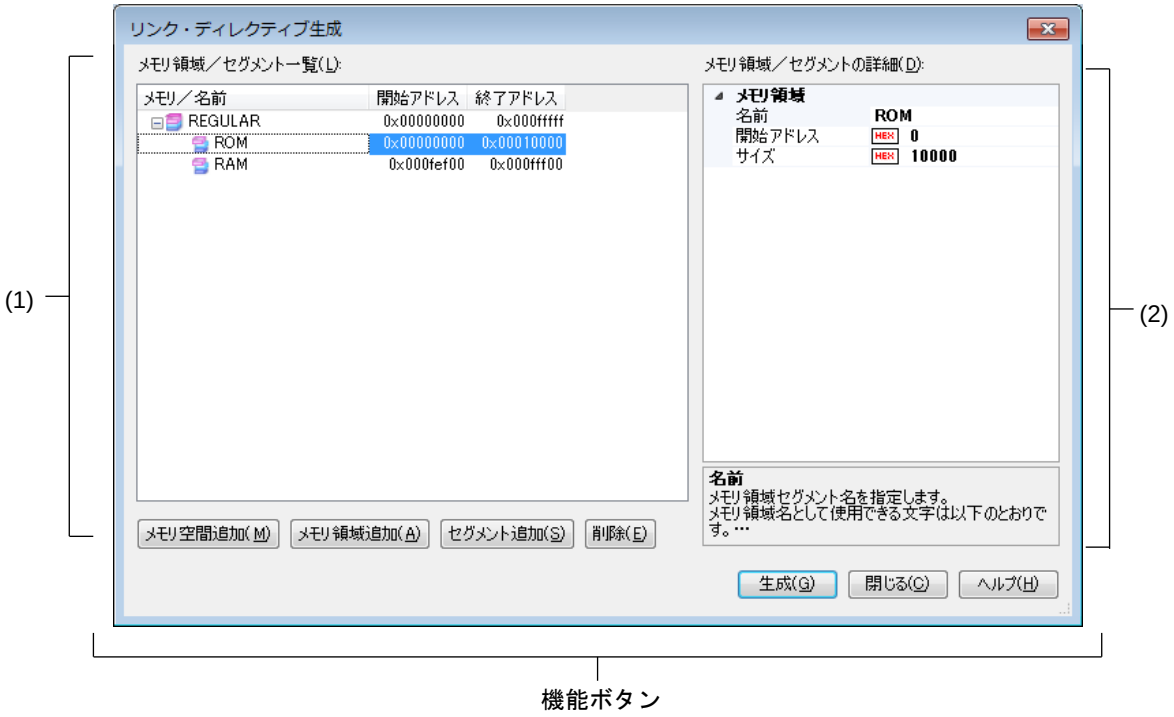
【場 所】 362 ページ →解析対象関数
【追加後】 したがって，ユーザが記述したアセンブラ・ソース・ファイル，およびユーザが作成したライブラリ・ファイルに内包されている関数については，解析対象外となるため，スタックサイズ変更ダイアログを用いて該当情報を設定する必要があります。
また，割り込み関数も解析対象外となるため，スタックサイズ変更ダイアログを用いて該当情報を設定する必要があります。

7.3.3 リンク・ディレクティブ生成 ダイアログの説明追加

【追 加】

リンク・ディレクティブ生成 ダイアログ

指定したメモリ，セグメントから，リンク・ディレクティブ・ファイルを生成します。



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］
- プロジェクト・ツリー パネル上において，ビルド・ツール・ノードを選択したのち，コンテキスト・メニュー→［リンク・ディレクティブ・ファイルを生成する...］を選択
- ビルド・ツールのプロパティ パネルのリンク・オプション タブ ［入力ファイル］－［リンク・ディレクティブ・ファイルを生成する］プロパティの［…］ボタンを選択

［各エリアの説明］

(1) ［メモリ領域／セグメント一覧］エリア

リンク・ディレクティブ・ファイルに生成するメモリ空間，メモリ領域およびセグメントのリストを表示します。

（a）［メモリ／名前］

メモリ空間，メモリ領域，セグメントの名前を表示します。

メモリ空間は，以下のうち，該当するメモリ空間の名前を表示します。
- REGULAR
- EX1，EX2，EX3，EX4，EX5，EX6，EX7，EX8，EX9，EX10
- EX11，EX12，EX13，EX14，EX15

メモリ領域，セグメントについては，この項目を直接編集することができます。メモリ領域名，セグメント名を変更すると，［メモリ領域／セグメントの詳細］エリアの［名前］も変更されます。

（b）［開始アドレス］

メモリ空間，メモリ領域，セグメントの開始アドレスを表示します。

メモリ領域，セグメントについては，この項目を直接編集することができます。開始アドレスを変更すると，［メモリ領域／セグメントの詳細］エリアの［開始アドレス］も変更されます。

（c）［終了アドレス］

メモリ空間，メモリ領域の終了アドレスを表示します。

セグメントの行については，“－”が表示されます。

（d）ボタン

メモリ空間追加	一覧の最後の行の直下に，新しいメモリ空間を追加します。 メモリ空間名は，作成時に登録可能なもっとも小さい数値のメモリ空間 EX*n* を追加します。（*n*：0～15の10進数） なお，このボタンは，EX1～EX15の15メモリ空間を登録している場合は無効となります。
メモリ領域追加	一覧で選択している行の直下に，新しいメモリ領域を追加します。 メモリ領域名は，デフォルトで“NewMemoryArea_XXX”となります（XXX：0～255の10進数）。 メモリ領域の詳細設定は，［メモリ領域／セグメントの詳細］エリアで行います。 なお，このボタンは，セグメントの行を選択している場合，および一覧に256個のメモリ領域を登録している場合は無効となります。

セグメント追加	一覧で選択している行の直下に，新しいセグメントを追加します。 セグメント名は，デフォルトで“Seg_XXX”となります（XXX：0～255の10進数）。 セグメントの詳細設定は，［メモリ領域／セグメントの詳細］エリアで行います。 なお，このボタンは，メモリ空間の行を選択している場合，および一覧に256個のセグメントを登録している場合は無効となります。
削除	一覧で選択しているメモリ空間（REGULAR以外），メモリ領域，またはセグメントを削除します。 メモリ領域を削除する場合は，メモリ領域に含まれているセグメントも削除します。 メモリ空間 REGULAR は削除することはできません。

また，このエリアは，次の機能を備えています。

- 行の展開／折りたたみ表示の切り替え
  行をダブルクリック，または行の先頭にある+マーク／-マークをクリックすることにより，各行の展開／折りたたみ表示の切り替えを行うことができます。

- メモリ領域，およびセグメントの行の移動
  ドラッグ・アンド・ドロップにより，メモリ領域，およびセグメントの行を移動することができます。
  備考 メモリ領域を移動する場合は，メモリ領域に含まれるセグメントも移動します。

- メモリ領域，およびセグメントのコピー
  メモリ領域，およびセグメントを選択したのち，［Ctrl］＋［C］キーの押下によりコピー，［Ctrl］＋［V］キーの押下により貼り付けを行うことができます。
  貼り付け位置は，［Ctrl］＋［V］キーの押下時に選択している行の直下となります。
  コピー後のメモリ領域，およびセグメントの名前には，先頭に“C_”が付加され，文字数がオーバする場合には末尾をカットします。
  備考 1. メモリ領域をコピーする場合，メモリ領域に含まれるセグメントはコピーしません。
  2. コピー後のメモリ領域，およびセグメントの開始アドレスは，空欄となります。
  3. コピー先のメモリ領域の属性により，コピーできない場合は，エラーとなります。

(2) ［メモリ領域／セグメントの詳細］エリア
［メモリ領域／セグメント一覧］エリアで選択したメモリ領域／セグメントの詳細情報の表示，および編集を行います。

（a）メモリ領域の詳細情報

名前	メモリ領域名を指定します。 使用可能な文字は，数字（0～9），英大文字（A～Z），英小文字（a～z），アンダスコア（_），クエスチョン（?），アットマーク（@）です。	
	デフォルト	NewMemoryArea_*XXX*（*XXX*：0～255の10進数）
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集
	指定可能値	31文字までの文字列
開始アドレス	メモリ領域の開始アドレスを指定します。 空欄，もしくは，16進数の値のみ入力可能です。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0x0～ 0xFFFF（16進数）【78K0（バンク品除く）】 0x0～ 0xFFFFF（16進数）【RL78，78K0R，78K0（バン

		ク品）】 ただし，デバイスにより指定可能値は異なります。
サイズ	メモリ領域のサイズを指定します。 空欄，もしくは，16進数の値のみ入力可能です。 ただし，空欄の場合はリンク時にエラーとなります。	
	デフォルト	空欄
	変更方法	テキスト・ボックスによる直接入力
	指定可能値	0x1～ 0xFFFF（16進数）【78K0（バンク品除く）】 0x1～ 0xFFFFF（16進数）【RL78，78K0R，78K0（バンク品）】 ただし，デバイスにより指定可能値は異なります。

（b）セグメントの詳細情報

名前	セグメント名を指定します。 使用可能な文字は，数字（0～9），英大文字（A～Z），英小文字（a～z），アンダスコア（_），クエスチョン（?），アットマーク（@）です。		
	デフォルト	NewMemoryArea_*XXX*（*XXX*：0～255の10進数）	
	変更方法	テキスト・ボックスによる直接入力，または［...］ボタンをクリックし，文字列入力 ダイアログによる編集	
	指定可能値	8文字までの文字列	
開始アドレスを指定	セグメントの開始アドレスを指定するかどうか指定します。		
	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	セグメントの開始アドレスを指定します。
		いいえ	セグメントの開始アドレスを指定しません。
開始アドレス	セグメントの開始アドレスを指定します。 空欄，もしくは，16進数の値のみ入力可能です。 本プロパティは，［開始アドレスを指定する］プロパティが“はい”の時のみ		
	デフォルト	空欄	
	変更方法	テキスト・ボックスによる直接入力	
	指定可能値	0x0～ 0xFFFF（16進数）【78K0（バンク品除く）】 0x0～ 0xFFFFF（16進数）【RL78，78K0R，78K0（バンク品）】 ただし，デバイスにより指定可能値は異なります。	
結合属性	セグメントの結合属性を指定します。 同名のセグメントが複数存在した場合にどのように結合するかを指定します。		
	デフォルト	自動（なし）	
	変更方法	ドロップダウン・リストによる選択	
	指定可能値	自動（なし）	順次結合（SEQUENT）の処理と同じですが，同名の入力セグメントのうちいずれかに，アセンブル時の再配置属性としてATが指定されていた場合，エラー（COMPLETE）の処理になります。
		順次結合（SEQUENT）	同名のセグメントが複数存在する場合，セグメントを出現順に，順次空きを作らないようにマージします。 BSEGはビット単位で出現順にマージします。
		エラー（COMPLETE）	同名のセグメントが複数存在する場合はエラーとします。

［機能ボタン］

ボタン	機能
生成	指定したメモリ領域，セグメント情報を元に，リンク・ディレクティブ・ファイル（ファイル名：プロジェクト名.dir）を生成し，プロジェクトに登録します。 リンク・ディレクティブ・ファイルの生成先は，プロジェクト・フォルダとなります。生成したリンク・ディレクティブ・ファイルは，プロジェクト・ツリーのファイル・ノードにも表示されます。 生成したリンク・ディレクティブ・ファイルはビルド対象となります。すでにリンク・ディレクティブ・ファイルをプロジェクトに登録していた場合，登録済みのリンク・ディレクティブ・ファイルはビルド対象外となります。
閉じる	本ダイアログをクローズします。
ヘルプ	本ダイアログのヘルプを表示します。 本バージョンではサポートされていません。

7.4 78K0 デバッグ編ドキュメント訂正

78K0 デバッグ編(資料番号：R20UT0731JJ0100)のドキュメントの訂正について説明します。

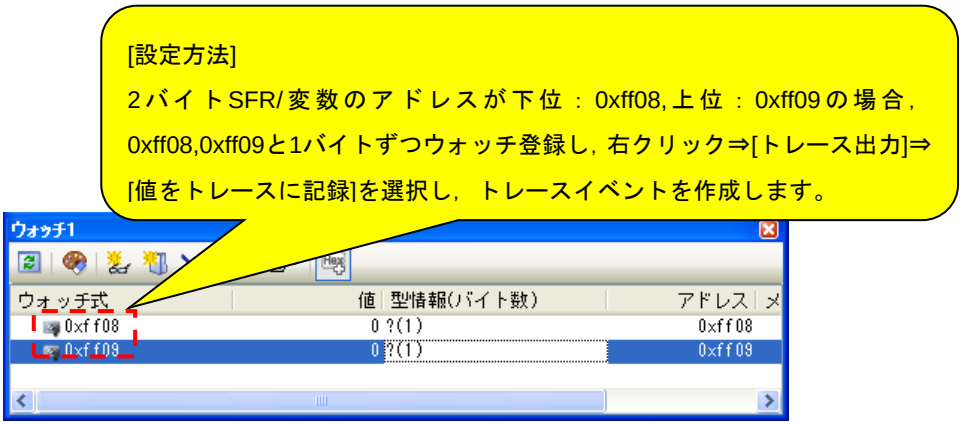
7.4.1 2 バイト SFR/変数のポイントトレースの説明追加

[場所] 138 ページ 「2.11.5 実行履歴を表示する」の前に追加

[追加後]

2.11.4(2) 2 バイト変数/SFR へのアクセスが発生したとき [IECUBE]

2 バイト SFR/変数のポイントトレースを行うには，上位 8 ビット，下位 8 ビットのアドレスを直接ウォッチパネルに登録し，トレースイベントを作成します。(図参照)



この設定を行った場合のトレース結果は以下のようになります。

7.5 V850 デバッグ編ドキュメント訂正

V850 デバッグ編(資料番号：R20UT2446JJ0100)のドキュメントの訂正について説明します。

7.5.1 GHS コンパイラ使用上の注意点の説明変更

[場所] 113 ページ 注 1．GHS コンパイラ（米国 Green Hills Software, LLC 製）使用上の注意点

[追加前]

－ 対応オプション
　　－ その他： -prepare_dispose，-call

[追加後]
- 対応オプション
　　- その他： -prepare_dispose，-callt

7.6 CubeSuite+ V2.02.00 解析編ドキュメント訂正

CubeSuite+ V2.02.00 解析編(R20UT2868JJ0100)のドキュメントの訂正について説明します。

7.6.1 動的解析の設定方法に関して

(1)変更箇所 1

[場所] 10 ページ 1.1.2 解析情報の種別 (2)動的解析情報 (a) トレース機能 備考 2 の後

[変更前]
なし。

[変更後]
以下の内容を追加。
トレース機能を利用した動的解析の結果を取得するには，デバッグ・ツールのプロパティにて，設定を変更する必要があります。以下に，各デバッグ・ツールと設定が必要なプロパティの一覧は以下のとおりです。

デバッグ・ツール	プロパティ	説明
【RL78】 【シミュレータ】 【V850E2】 【シミュレータ】	[デバッグ・ツール設定]タブ → [トレース]カテゴリ → [トレース機能を使用する]プロパティ(※)	「はい」に設定してください。
【78K0】 【シミュレータ】 【78K0R】 【シミュレータ】	[デバッグ・ツール設定]タブ → [トレース]カテゴリ → [トレース・メモリ・サイズ[フレーム]]プロパティ	リストから，トレースのサイズを選択してください。
【RL78】 【IECUBE】 【78K0】 【IECUBE】 【78K0R】 【IECUBE】	－	設定不要です。トレース・サイズ固定で常に有効。
【V850E2】 【MINICUBE】／ 【MINICUBE2】／ 【E1】／【E20】 【78K0R】 【MINICUBE2】／ 【E1】／【E20】 ／ 【EZ Emulator】 【78K0】	－	トレース機能はサポートしていません。

【MINICUBE2】／ 【E1】／【E20】／ 【EZ Emulator】		
【RL78】 【E1】／【E20】／ 【EZ Emulator】	[デバッグ・ツール設定]タブ → [トレース]カテゴリ → [トレース機能を使用する]プロパティ(※)	「はい」に設定してください。 トレース機能をサポートしていないデバイスの場合は，本プロパティは表示されません。
	[デバッグ・ツール設定]タブ → [トレース]カテゴリ → [トレース・メモリ・サイズ[フレーム]]プロパティ	設定不要です。
【V850E2】 【IECUBE2】	[デバッグ・ツール設定]タブ → [トレース]カテゴリ → [トレース・データの用途]プロパティ(※)	「トレース」を選択してください。
	[デバッグ・ツール設定]タブ → [トレース]カテゴリ → [トレース・データ種別]プロパティ	関数情報のみを取得する場合は，「分岐命令」を選択してください。 変数情報のみを取得する場合は，[アクセス・アドレス／データ値]を選択してください。 関数／変数情報を取得する場合は，[分岐命令とアクセス・アドレス／データ値]を選択してください。 ただし，選択可能な設定は，デバイスに依存します。
	[デバッグ・ツール設定]タブ → [トレース]カテゴリ → [トレース・メモリ・サイズ[フレーム]]プロパティ	リストから，トレースのサイズを選択してください。

(※)プログラム解析プラグインの[設定]タブ→[全般]カテゴリ→[動的解析を有効にする]プロパティを「はい」に設定すると，デバッグ・ツールへの接続時に，自動的に設定を変更します。

(2)変更箇所 2

[場所] 11 ページ 1.1.2 解析情報の種別 (2)動的解析情報 (b) RRM 機能／RAM モニタ（疑似 RRM）

機能 注意の後

[変更前]
なし。

[変更後]
以下の内容を追加。
< RRM機能／RAMモニタ（疑似RRM）の設定>
RRM機能／RAMモニタ（疑似RRM）機能を利用した動的解析の結果を取得するには，デバッグ・ツールのプロパティにて，設定を変更する必要があります。以下に，各デバッグ・ツールと設定が必要なプロパティの一覧を示します。

デバッグ・ツール	プロパティ	説明
【RL78】 【シミュレータ】 【V850E2】 【シミュレータ】 【78K0】 【シミュレータ】 【78K0R】 【シミュレータ】	[デバッグ・ツール設定]タブ → [実行中のメモリ・アクセス]カテゴリ → [実行中に表示更新を行う]プロパティ(※)	「はい」に設定してください。
	[デバッグ・ツール設定]タブ → [実行中のメモリ・アクセス]カテゴリ → [表示更新間隔[ms]]プロパティ	更新間隔を指定してください。 100～65500 の範囲で指定可能です。
【RL78】 【IECUBE】 ／【E1】／ 【E20】 ／【EZ Emulator】 【V850E2】 【IECUBE2】／ 【MINICUBE】／ 【MINICUBE2】／【E1】／ 【E20】 【78K0R】 【IECUBE】／ 【MINICUBE2】／【E1】／ 【E20】／ 【EZ Emulator】 【78K0】 【IECUBE】／ 【MINICUBE2】／【E1】／ 【E20】／ 【EZ Emulator】	[デバッグ・ツール設定]タブ → [実行中のメモリ・アクセス]カテゴリ → [実行中に表示更新を行う]プロパティ(※)	「はい」に設定してください。
	[デバッグ・ツール設定]タブ → [実行中のメモリ・アクセス]カテゴリ → [実行中に表示更新を行う]プロパティ	「はい」に設定してください。
	[デバッグ・ツール設定]タブ → [実行中のメモリ・アクセス]カテゴリ → [表示更新間隔[ms]]プロパティ	更新間隔を指定してください。 100～65500 の範囲で指定可能です。

(※)プログラム解析プラグインの[設定]タブ→[全般]カテゴリ→[動的解析を有効にする]プロパティを「はい」に設定すると，デバッグ・ツールへの接続時に，自動的に設定を変更します。

(3)変更箇所 3

[場所] 11 ページ 解析情報の種別 (2)動的解析情報 (c) カバレッジ機能 注意 3 の後

[変更前]
なし。

[変更後]
以下の内容を追加。
カバレッジ機能を利用した動的解析の結果を取得するには，デバッグ・ツールのプロパティにて，設定を変更する必要があります。以下に，各デバッグ・ツールと設定が必要なプロパティの一覧を示します。

デバッグ・ツール	プロパティ	説明
【RL78】 【シミュレータ】 【78K0】 【シミュレータ】 【78K0R】 【シミュレータ】	[デバッグ・ツール設定]タブ → [カバレッジ]カテゴリ → [カバレッジ機能を使用する]プロパティ(※)	「はい」に設定してください。
【V850E2】 【シミュレータ】	[デバッグ・ツール設定]タブ → [カバレッジ]カテゴリ → [カバレッジ機能を使用する]プロパティ(※)	「はい」に設定してください。
	[デバッグ・ツール設定]タブ → [カバレッジ]カテゴリ → [カバレッジ測定領域(1M バイト単位)]プロパティ	カバレッジの測定領域の開始アドレスを指定してください。
【V850E2】 【IECUBE2】	[デバッグ・ツール設定]タブ → [トレース]カテゴリ → [トレース・データの用途]プロパティ	「カバレッジ」を選択してください。
【RL78】 【IECUBE】 【78K0R】 【IECUBE】 【78K0】 【IECUBE】	ー	設定不要です。 常に有効です。
【RL78】 【E1】／【E20】／ 【EZ Emulator】 【V850E2】 【MINICUBE】／ 【MINICUBE2】／ 【E1】／【E20】 【78K0R】 【MINICUBE2】／【E1】／ 【E20】 ／【EZ Emulator】 【78K0】 【MINICUBE2】／【E1】／ 【E20】 ／【EZ Emulator】	ー	カバレッジ機能はサポートしていません。

(※)プログラム解析プラグインの[設定]タブ→[全般]カテゴリ→[動的解析を有効にする]プロパティを「はい」に設定すると，デバッグ・ツールへの接続時に，自動的に設定を変更します。

7.7 起動編ドキュメント訂正

起動編(資料番号：R20UT2865JJ0100)のドキュメントの訂正について説明します。

7.7.1 高速化ユーティリティの説明削除

[場所] 105 ページ 2.10.2 高速化ユーティリティを使用する

7.7.2 動作環境に関して

[場所] 11 ページ 1. 4 動作環境 （2）ソフトウエア環境

［変更前］

- Microsoft Visual C++ 2010 SP1 ランタイム・ライブラリ

［変更後］

- Microsoft Visual C++ 2015 Update 3 ランタイム・ライブラリ(x86)

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品、本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

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

   高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等

   当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
6. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
9. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
10. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
12. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

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直接的、間接的に支配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。

(Rev.4.0-1　2017.11)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。

© 2019 Renesas Electronics Corporation. All rights reserved.