

RL78開発環境 移行ガイド

R8C、M16CからRL78への移行(オンチップ・デバッグ編)

ルネサス エレクトロニクス株式会社

MCU事業本部 ソフトウェア統括部

MCUツール技術部

2012/06/29 Rev. 1.00

R20UT2150JJ0100

はじめに

本資料はR8C/M16Cファミリ用High-performance Embedded WorkshopとRL78ファミリ用CubeSuite+との違いやCubeSuite+でのE1,E20エミュレータ操作方法について記載しています。

本資料は**CubeSuite+V1.02.00**をベースに説明しています。

ツールチェーン等について下記資料を参照ください。

- ・「RL78開発環境移行ガイド R8C/M16C,H8S/H8SXから RL78への移行 (コーディング編)」
- ・「RL78開発環境移行ガイド R8C/M16C,H8S/H8SXから RL78への移行 (ビルド編)」
- ・「RL78開発環境移行ガイド R8C/M16C,H8S/H8SXから RL78への移行 (起動編)」

また、CubeSuite+はツールの使い方を記載したチュートリアルガイドを用意していますので参照ください。

チュートリアルガイドはCubeSuite+のメニューから [ヘルプ] → [チュートリアル] を選択で参照できます。

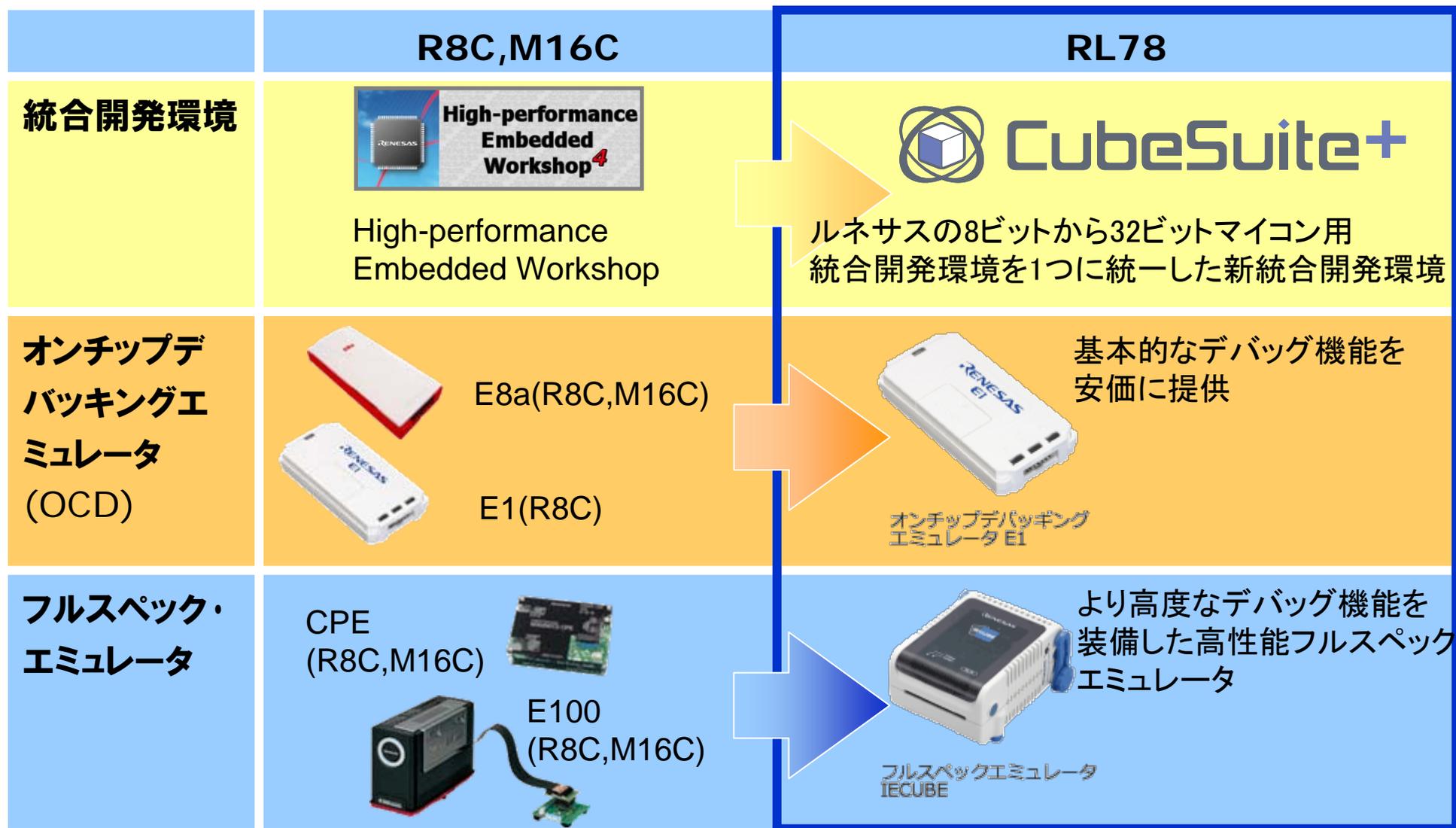


チュートリアルガイド

目次

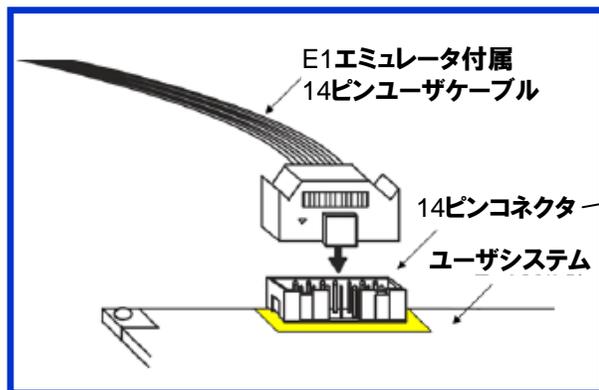
- 1. 統合開発環境とエミュレータ
- 2. ターゲットインタフェースの違い(OCD)
- 3. デバッガの変更方法
- 4. IDコード入力方法
- 5. リソース確保の方法
- 6. オンチップ・デバッグ・オプションバイトの設定方法
- 7. エミュレータ接続時の設定はどこで行うのか？
- 8. エミュレータの接続方法
- 9. エミュレータの接続解除方法
- 10. プログラムのダウンロード方法
- 11. ダウンロードファイルを追加登録する方法
- 12. プログラムの実行、停止方法
- 13. ブレーク時のマイコンの動作の違い
(周辺ブレーク機能)
- 14. プログラム実行中にメモリや変数を参照、変更する方法
- 15. プログラム実行中にメモリや変数を自動更新する方法
- 16. ブレークポイントの設定方法
- 17. 変数へのアクセスでブレークする方法
- 18. メモリをフィルする方法
- 19. メモリ内容をセーブする方法
- 20. フラッシュ・セルフ・プログラミングについて
- 21. デバイス単体で動作確認するための書き込み方法
- 22. アクション・イベント機能(Printfイベント)
- 23. 変数と関数の一覧表示機能
- 24. 解析グラフ機能
- 25. エミュレータデバッグ機能比較(OCD)

1. 統合開発環境とエミュレータ



*この資料ではOCDを使用する場合の移行方法を紹介しています。

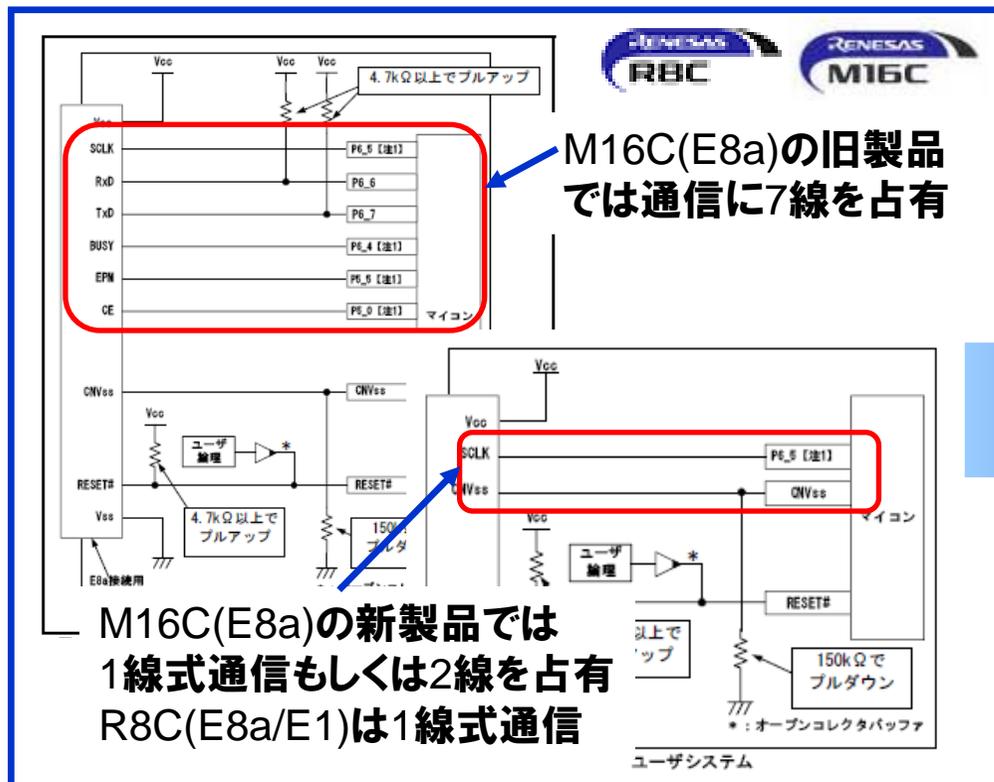
2. ターゲットインタフェースの違い(OCD)



RL78とE1エミュレータは14ピンコネクタを使用して接続します。

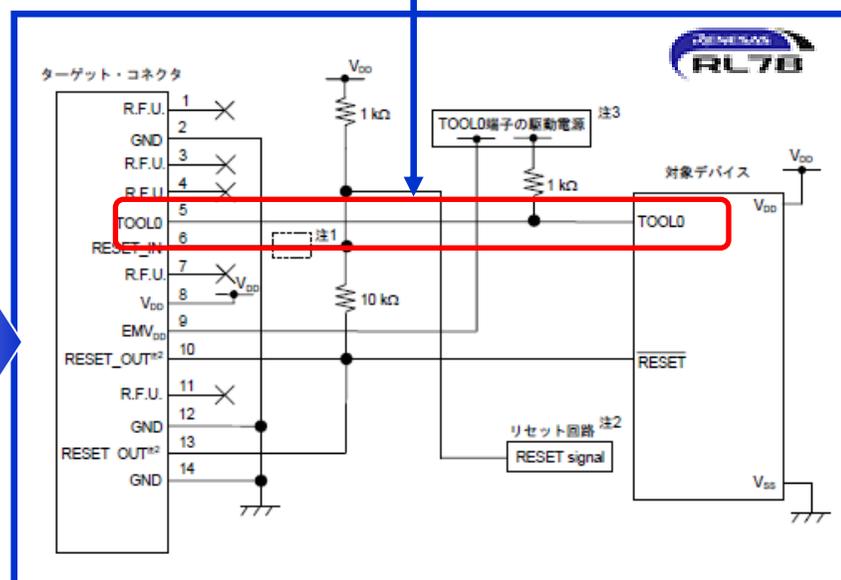
	型名	メーカー	仕様
14ピンコネクタ	7614-6002	住友スリーエム株式会社	14ピンストレートタイプ (国内推奨)
	2514-6002	3M Limited	14ピンストレートタイプ (海外推奨)

コネクタはR8C, M16C用E8a/E1で使用しているコネクタと同じですが、下図のように**通信インタフェースは異なります。**



M16C(E8a)の新製品では1線式通信もしくは2線を占有
R8C(E8a/E1)は1線式通信

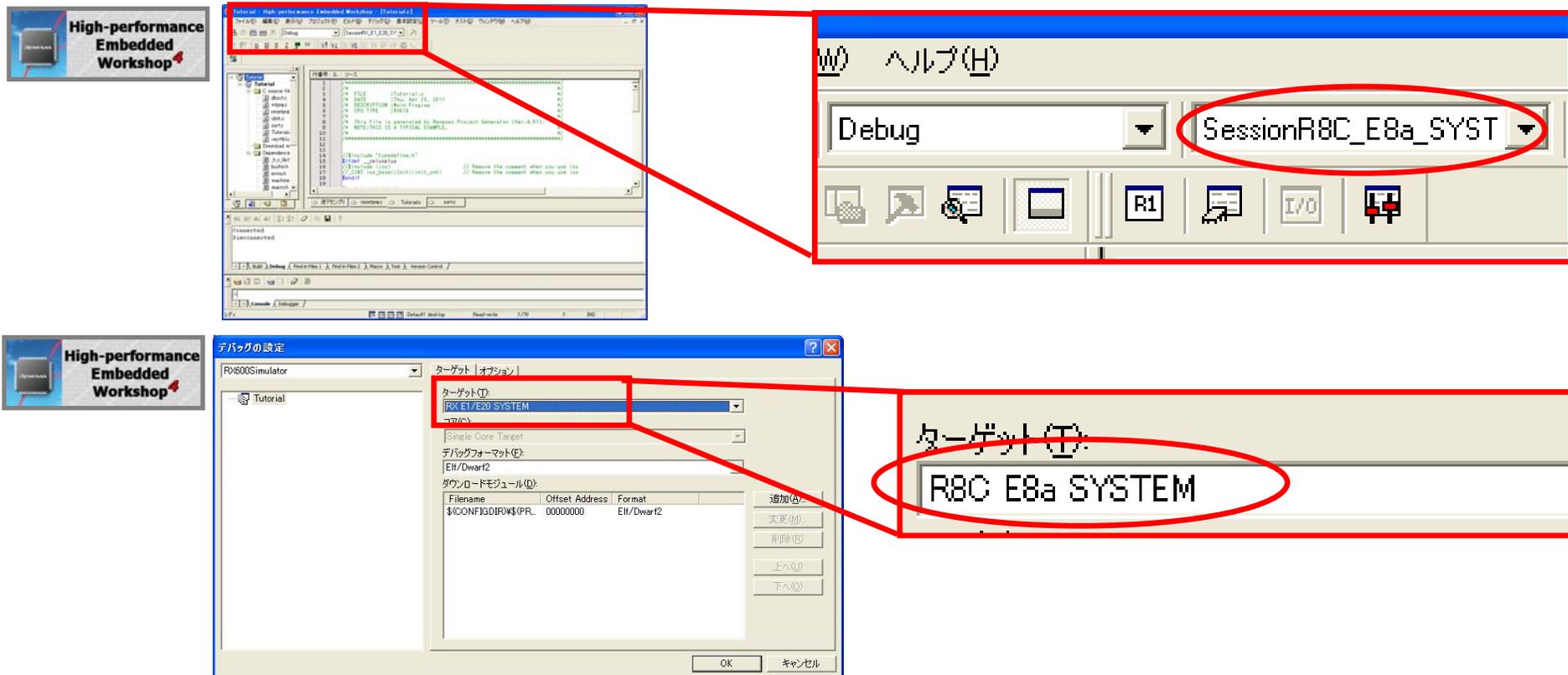
RL78(E1)は1線式通信



デバイスによっては結線回路が異なる場合があるので詳細はE1/E20エミュレータユーザズマニュアル 別冊 (RL78接続時の注意事項)を確認してください。

3. デバッガの変更方法

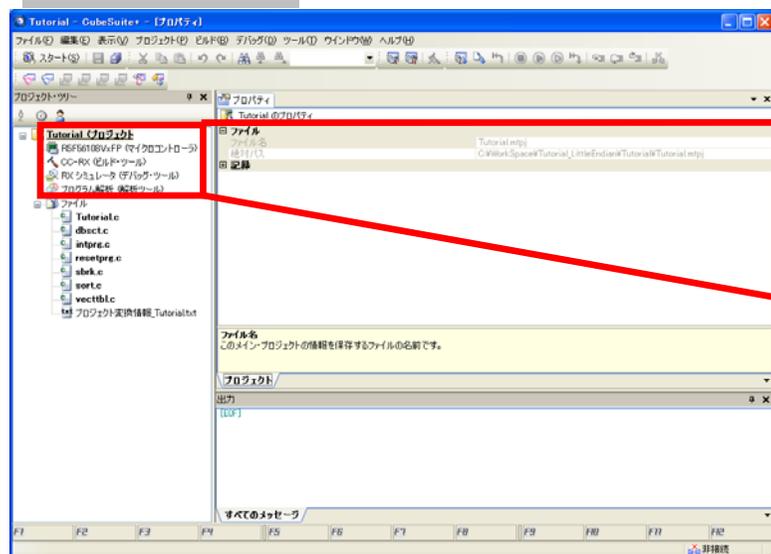
HEWではデバッグセッションの変更や [デバッグの設定] ダイアログのターゲットの変更でデバッガ (E8a、E1エミュレータやシミュレータ) の選択を行っていましたが、CubeSuite+ではプロジェクトツリー上でデバッガ (E1エミュレータやシミュレータ) の変更をおこないます。次ページにその手順を示します。



HEWでのデバッガ (E8a,E1エミュレータやシミュレータ) 選択方法

3. デバッガの変更方法

- (1) CubeSuite+では現在選択しているデバッガはプロジェクトツリーパネル上のデバッグ・ツール名 (デバッグ・ツール) で確認できます。
以下の例ではE1エミュレータが選択されています。

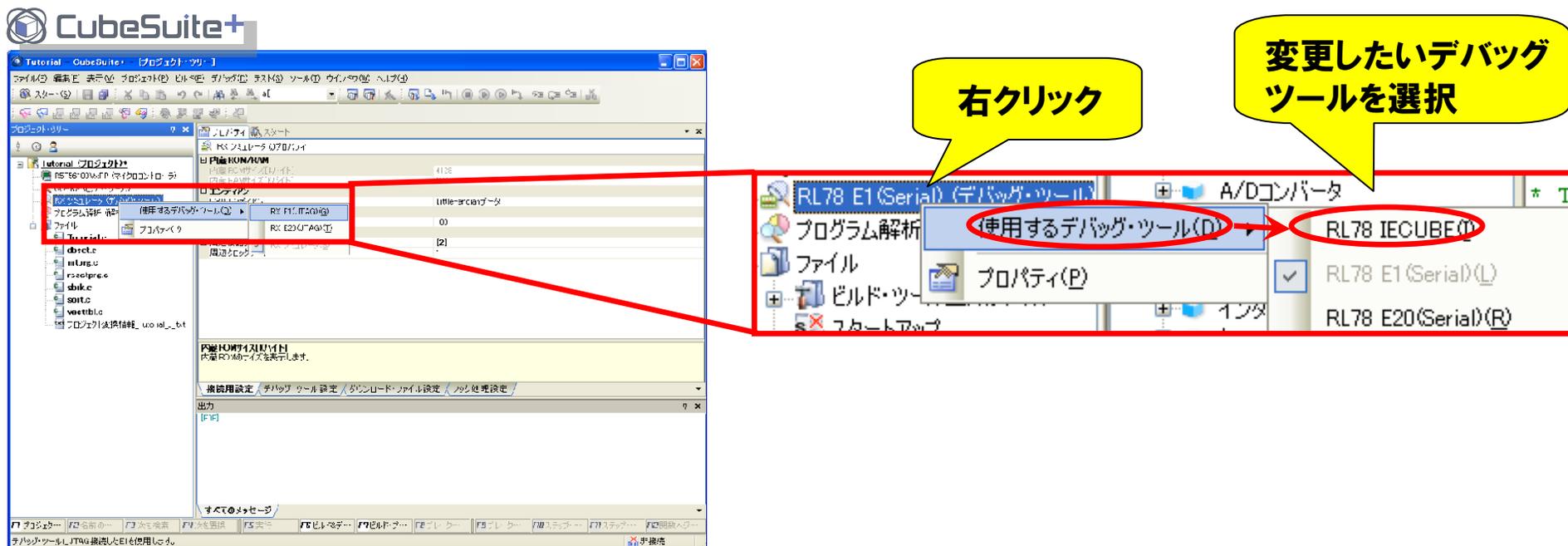


選択しているデバッガ

- CA78M1R (ビルド・ツール)
- RL78 E1 (Serial) (デバッグ・ツール)**
- プログラム解析 (解析ツール)

3. デバッガの変更方法

(2) デバッガを変更する場合、デバッグ・ツール名 (デバッグ・ツール) を右クリックしてポップアップメニューを開いて、[使用するデバッグツール] を選択して変更したいデバッグツールを選択してください。



4.IDコード入力方法

R8C,M16CとRL78ではエミュレータ使用時にどちらもIDコードの設定を行いますが、設定方法、照合方法、不一致だった場合の動作は異なります。
以下の仕様の違いを記載します。

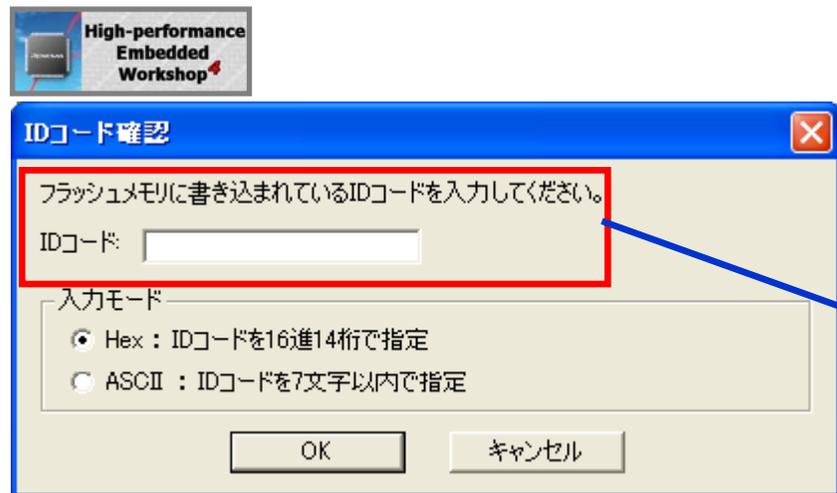
	IDコード桁数 (16進数)	IDコードの アドレス	設定方法	照合方法	不一致だった 場合の動作	オンボード・ ライターでの有効性
R8C M16C	14桁	0xFFDF, E3,EB,EF, F3,F7,FB番 地	ビルド時にユーザ プログラムに埋め 込む	デバッガ起動時に IDがオールFFhの 場合は自動照合、 それ以外の場合は ダイアログで入力	デバッガは起動 しない(フラッ シュメモリの内 容は保持され る)	有り (オンボード・ライタ 起動時もIDの照 合が必要)
RL78	20桁	0xC4~ 0xCD 番地	ビルド時にユーザ プログラムに埋め 込む	デバッガに予め照 合するIDコードを 設定しておく	オンチップ・デ バッグ・オプショ ンバイトの設定 による*	無し (デバッグ時のみ 有効)

*詳細はE1/E20エミュレータユーザーズマニュアル 別冊(RL78接続時の注意事項)を参照してください。

4.IDコード入力方法

マイコンにIDコードが書き込まれている場合、HEWではエミュレータ起動時に [IDコード確認] ダイアログが表示されますが、CubeSuite+ではエミュレータ**起動前**に [プロパティ] パネルでIDコードを設定しておく必要があります。

以下を参考にIDコードを設定してください。



RL78 E1 (Serial) のプロパティ	
内部ROM/RAM	
内部ROMサイズ[Kバイト]	64
内部RAMサイズ[バイト]	4096
データフラッシュ・メモリ・サイズ[Kバイト]	4
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
モニタークロック	システム
ターゲット・ボードとの接続	
通信方式	1線式(TOOL0)
エミュレータから電源供給をする(最大200mA)	はい
供給電圧	5.8V
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	
内蔵ROMまたは内蔵フラッシュ・メモリ上のコードを読み出す際のセキュリティID(16進数で20桁)を設定します。セキュリティユーザーズ・マニュアルを参照してください。	

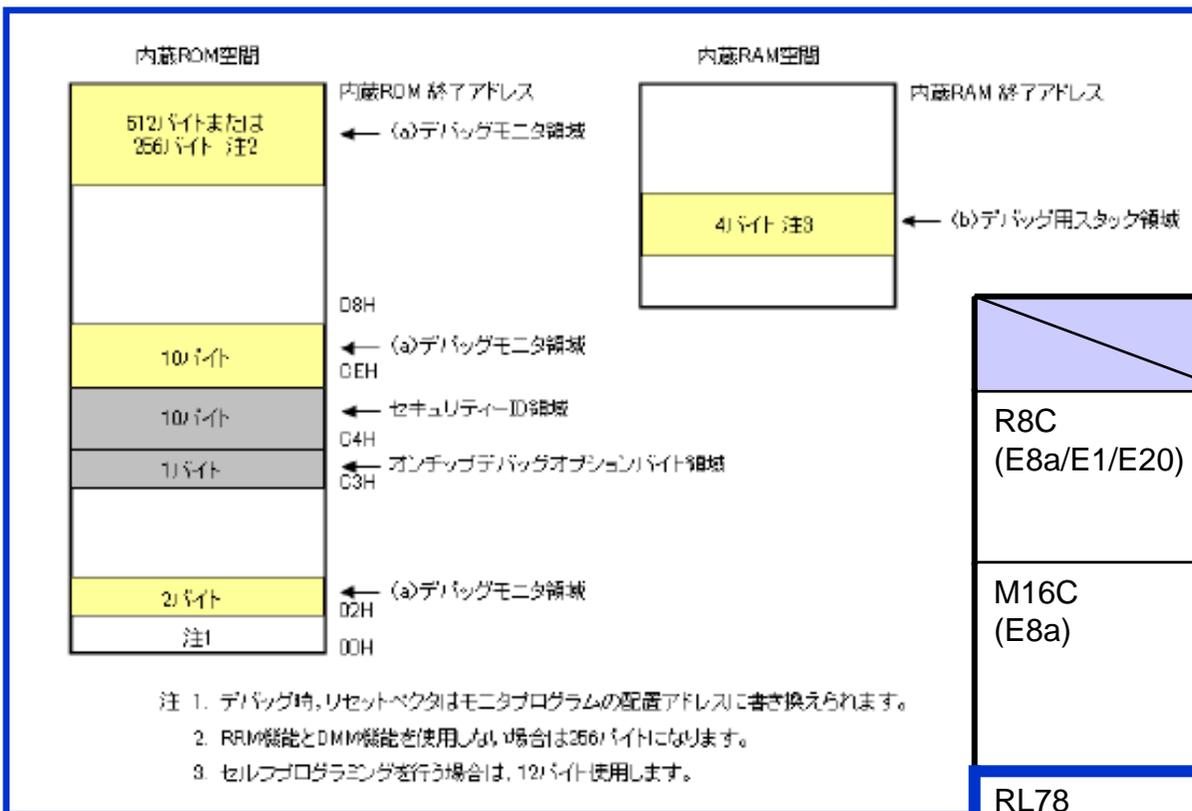
接続用設定 | デバッグ・ツール設定 | ダウンロード・ファイル設定 | フック処理設定

注:R8C,M16C(HEW)ではIDがFFFFFFFFFFFFFFF(全てFFh)の場合は接続操作時に自動的に照合されるのでIDコード確認ダイアログは表示されない

5. リソース確保の方法

RL78ではOCD使用時にユーザリソースを占有します。それらの領域はユーザプログラムで使用しないようにビルドツール等で領域を確保する必要があります。

E1/E20で使用する予約領域(RL78)*



CubeSuite+での設定方法は次ページを参照してください。

R8C, M16Cとの比較

	リソース占有	占有容量	確保方法
R8C (E8a/E1/E20)	有り (デバイスによる)	最大ROM2K スタック8バイト ベクタの一部	ビルド時に確保 (リンカで占有領域のアドレスを避けるよう設定する)
M16C (E8a)	有り (デバイスによる)	最大ROM2K RAM128バイト スタック14バイト ベクタの一部	
RL78 (E1/E20)	有り	左図参照	ビルド時に確保 (CubeSuite+のGUIで設定可能)

*詳細はE1/E20エミュレータユーザーズマニュアル 別冊 (RL78接続時の注意事項)を参照してください。

5. リソース確保の方法

デバッグ・モニタ領域のアドレスはビルド・ツールのプロパティからリンク・オプションで設定できます。

The screenshot shows the CubeSuite+ interface. On the left, the project tree is visible, with 'CA78K0R (ビルド・ツール)' selected. The main window displays the properties for 'CA78K0R'. The 'デバイス' (Device) section is highlighted with a red box, containing the following settings:

Property	Value
オンチップ・デバッグを設定する	はい(-go)
オンチップ・デバッグ・オプション・バイト制御値	HEX 84
デバッグ・モニタ領域開始アドレス	HEX FE00
デバッグ・モニタ領域サイズ[バイト]	512
ユーザ・オプション・バイトを設定する	はい(-gb)
ユーザ・オプション・バイト値	HEX EFFFEB
ミラー領域指定	MAA=0(-mi0)
フラッシュ・スタート・アドレスを設定する	(いいえ)
ブート領域用ロード・モジュール・ファイル名	(いいえ)
セルフRAM領域への配置を制御する	(いいえ)

At the bottom of the window, the 'リンク・オプション' (Link Options) tab is highlighted with a red circle.

6. オンチップ・デバッグ・オプションバイトの設定方法

オンチップ・デバッグ・オプションバイトはビルド・ツールのプロパティからリンク・オプションで設定できます。

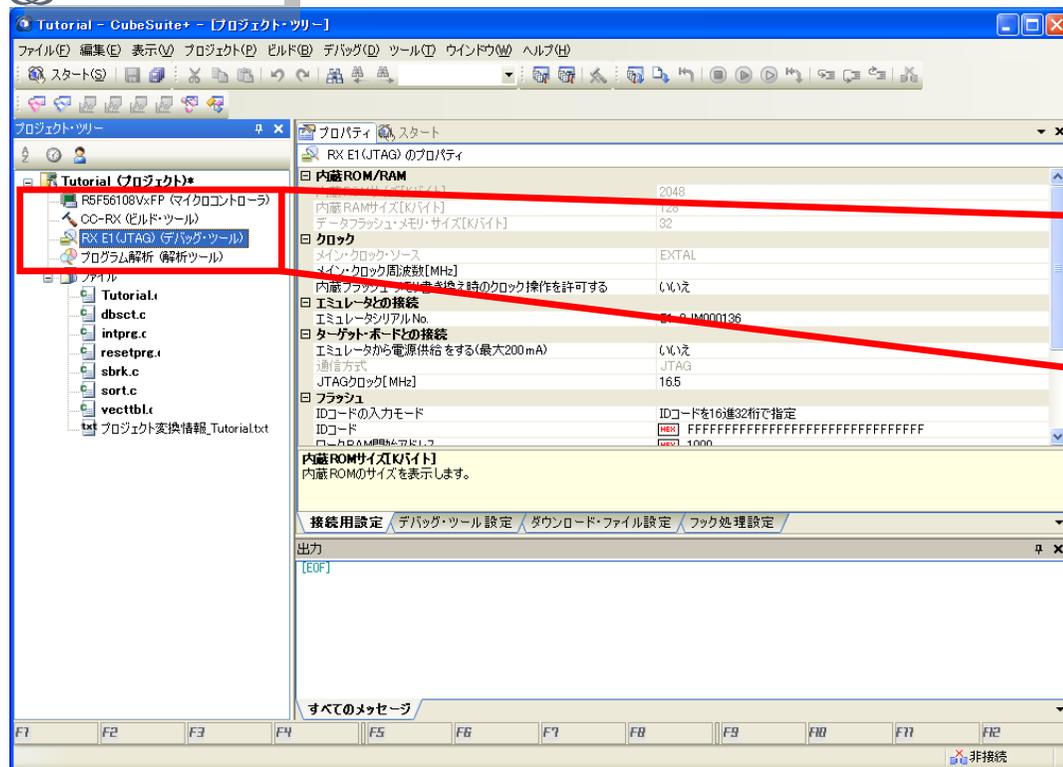
The screenshot shows the CubeSuite+ interface. On the left, a project tree is visible with 'CA78K0R (ビルド・ツール)' selected. The main window displays the properties for 'CA78K0R のプロパティ'. The 'デバイス' (Device) section is expanded, and the 'オンチップ・デバッグ・オプション・バイト制御値' (On-chip Debug Option Byte Control Value) is highlighted with a red box and set to 'HEX: 84'. The 'リンク・オプション' (Link Options) tab at the bottom is also highlighted with a red circle.

項目	値
使用するリンク・ディレクティブ・ファイル	r_lk.dr
出力ファイル	
出力フォルダ	%BuildModeName%
出力ファイル名	%ProjectName%.lbf
強制リンクを行う	(いいえ)
ライブラリ	
使用するライブラリ・ファイル	使用するライブラリ・ファイル[0]
システム・ライブラリ・ファイル	システム・ライブラリ・ファイル[0]
追加のライブラリ・パス	追加のライブラリ・パス[0]
システム・ライブラリ・パス	システム・ライブラリ・パス[0]
デバイス	
オンチップ・デバッグを設定する	はい(-go)
オンチップ・デバッグ・オプション・バイト制御値	HEX: 84
デバッグ・モニタ領域開始アドレス	HEX: FE00
デバッグ・モニタ領域サイズ[バイト]	512
ユーザ・オプション・バイトを設定する	はい(-gb)
ユーザ・オプション・バイト値	HEX: EFFFEB
ミラー領域指定	MAA=0(-mi0)
フラッシュ・スタート・アドレスを設定する	(いいえ)
ブート領域用ロード・モジュール・ファイル名	
セルフRAM領域への配置を制御する	(いいえ)
メッセージ	
スタック	
デバイス	

共通オプション / コンパイル・オブ ... / アセンブル・オブ ... / **リンク・オプション** / ROM化プロセス ...

7. エミュレータ接続時の設定はどこで行うのか？

HEWではエミュレータ接続時に [起動設定] ダイアログ、[コンフィグレーションプロパティ] ダイアログが表示され設定を行います。CubeSuite+ではエミュレータ**接続前**に [プロパティ] パネルで設定を行う必要があります。以下に手順を示します。
[プロジェクト・ツリー] パネルのデバッグツール名 (デバッグ・ツール) をダブルクリックしてください。デバッグツールのプロパティが開きます。



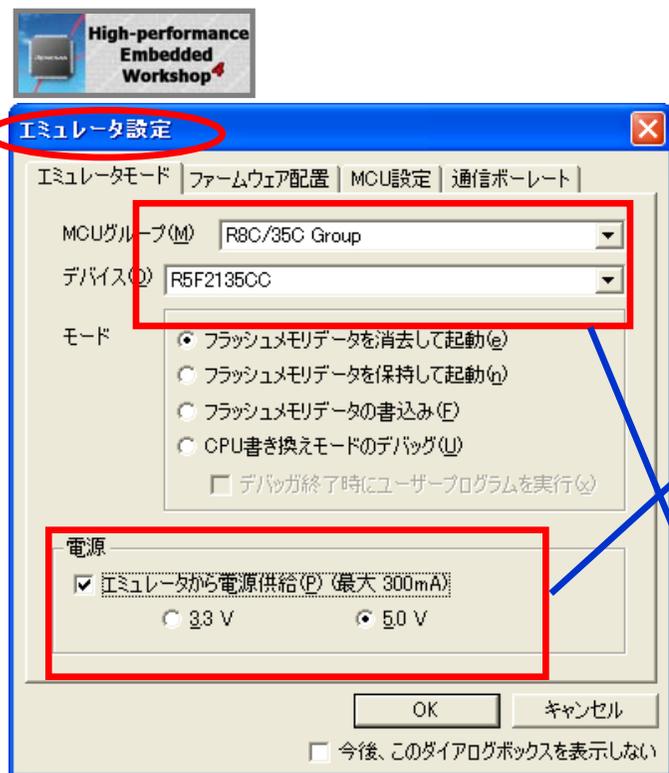
ダブルクリック

CA78K0R (ビルド・ツール)
RL78 E1 (Serial) (デバッグ・ツール)
プログラム解析 (解析ツール)

7. エミュレータ接続時の設定はどこで行うのか？

HEWはエミュレータ接続時に [エミュレータ設定] ダイアログで行いますが、CubeSuite+ではエミュレータ接続前にデバッガの [プロパティ] パネルで設定しておきます。

(1) HEW [エミュレータ設定] ダイアログ⇒CubeSuite+ [接続用設定タブ] に相当



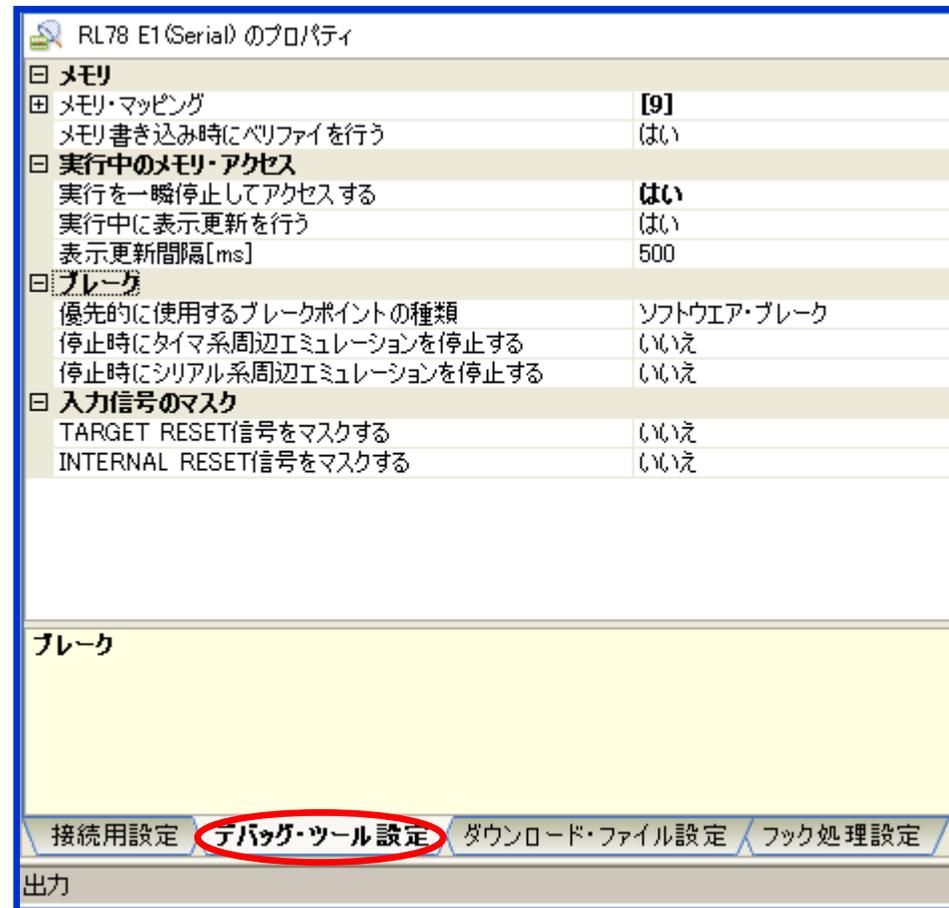
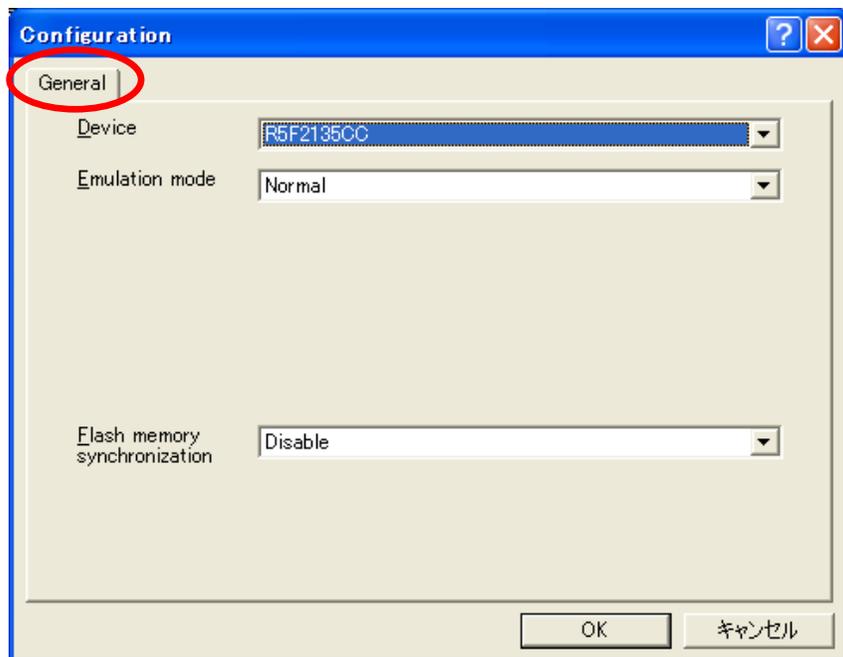
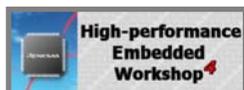
電源供給機能など同じ機能はそれぞれ対応

例: R8C用E8aでのエミュレータ設定ダイアログ

CubeSuite+ではプロジェクト作成時にデバイスを設定。

7. エミュレータ接続時の設定はどこで行うのか？

(2) HEW [コンフィグレーション] ダイアログ⇒CubeSuite+ [デバッグ・ツール設定] に相当



例：R8C用E8aでのエミュレータ設定ダイアログ

8. エミュレータの接続方法

(1) CubeSuite+のメニューから [デバッグ] → [デバッグ・ツールへ接続] を選択すると設定したエミュレータ (デバッグ・ツール) に接続します。
接続が完了すると画面右下のステータスバーにデバッグツール名が表示されます。

CubeSuite+

The image shows the CubeSuite+ interface. A red box highlights the 'Debug' menu, with a callout pointing to the 'Connect to Debug Tool' option. Another red box highlights the status bar at the bottom, showing the transition from 'Not Connected' to 'RL78 E1 (Serial)'.

接続前

接続前: ステータスバーに「非接続」が表示されています。

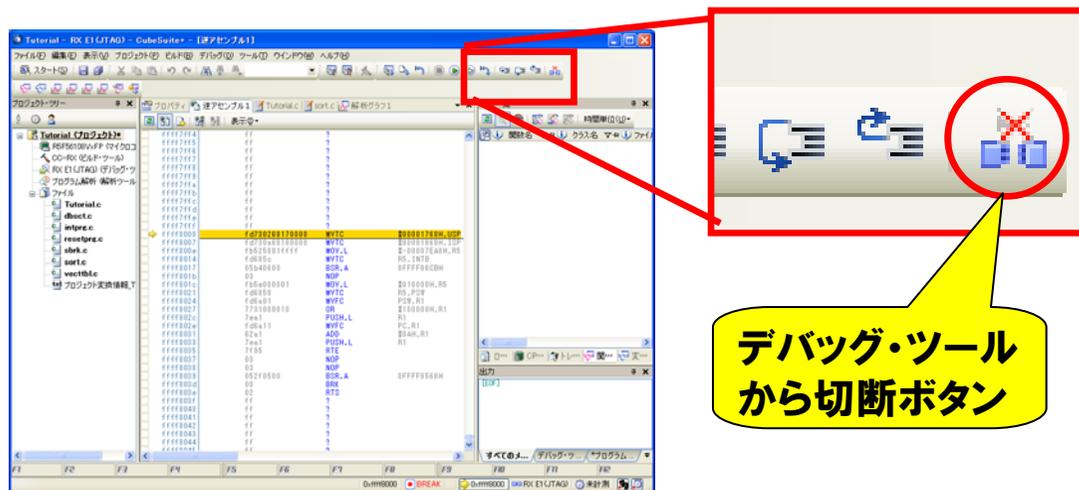
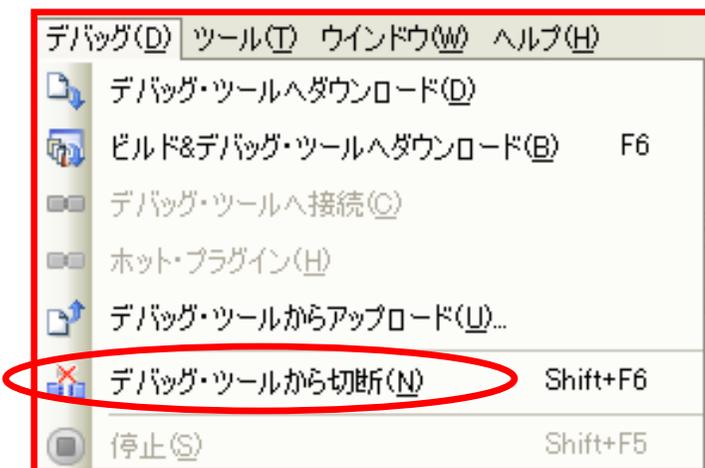
接続後

接続後: ステータスバーに「RL78 E1 (Serial)」が表示されています。

注: マイコンにIDコードを書き込んでいる場合は、「2.IDコード入力方法」を参考にして事前にIDコードを設定してください。

9. エミュレータの接続解除方法

(2) 接続を解除する場合はメニューから [デバッグ・ツールから切断] を選択するかデバッグツールバーの  ボタンをクリックしてください。



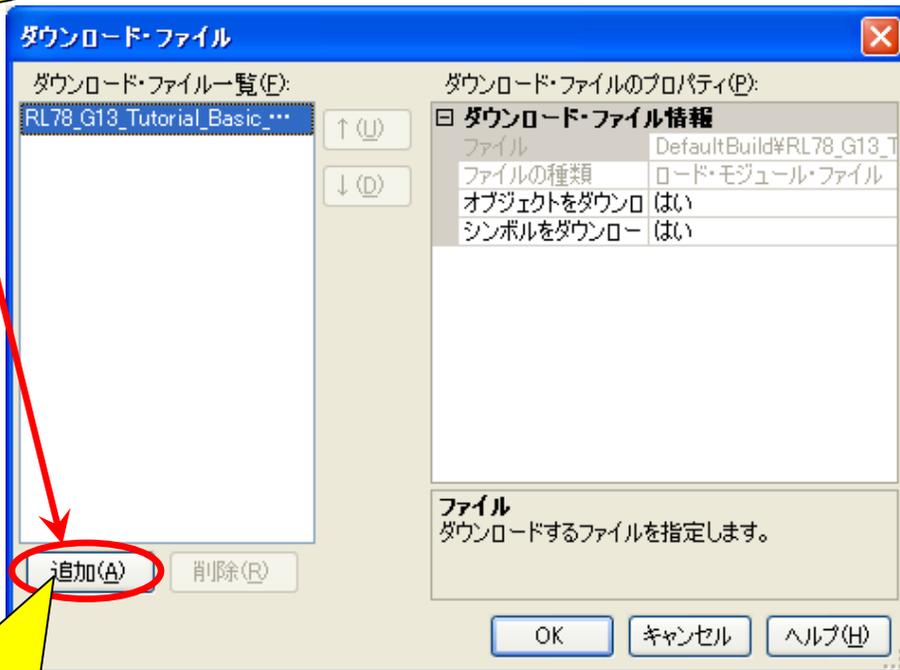
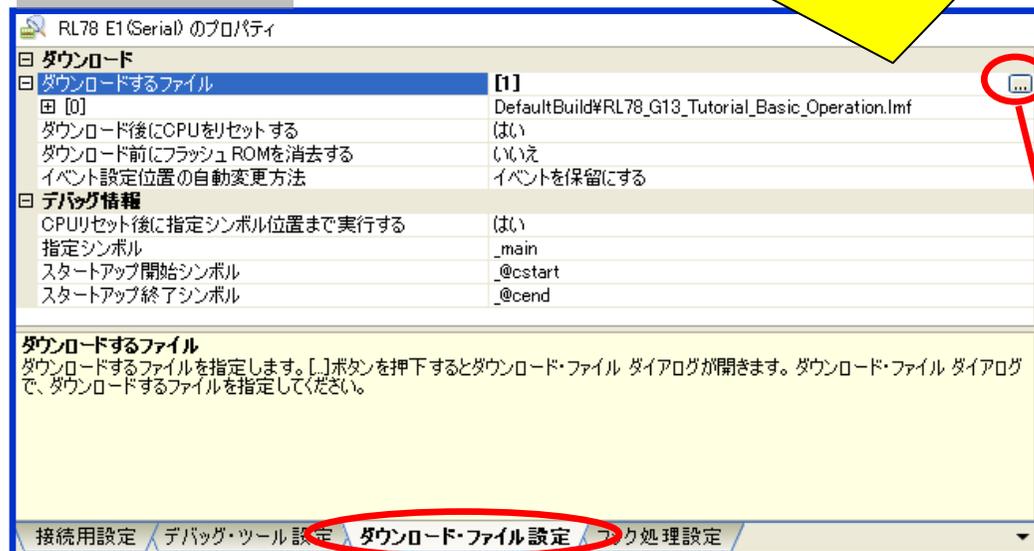
11. ダウンロードファイルを追加登録する方法

ダウンロードファイルの追加は [プロパティ] パネルの [ダウンロード・ファイル設定] タブで行います。

- (1) 「ダウンロードするファイル」を選択して右側の ... をクリックしてください。
- (2) ダウンロード・ファイルダイアログが開きますので追加ボタンを押してください。

CubeSuite+

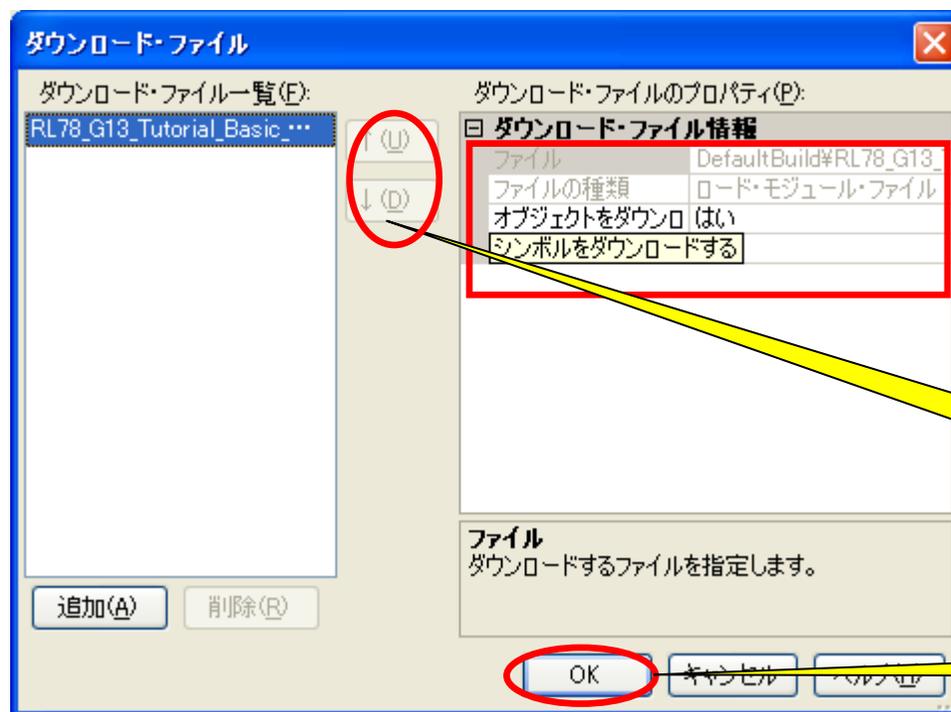
「ダウンロードするファイル」を選択後クリック



追加ボタンをクリック

11. ダウンロードファイルを追加登録する方法

(3) ダウンロード・ファイル情報にダウンロードするファイル名、ファイルの種類等を設定後OKボタンを押してください。



注:ダウンロード実行時、登録したファイルはすべてダウンロードされます。
任意のファイルのみダウンロードする場合は上記設定画面でダウンロードしたいファイルのみ「オブジェクトをダウンロードする」、「シンボルをダウンロードする」を「はい」に設定してください。

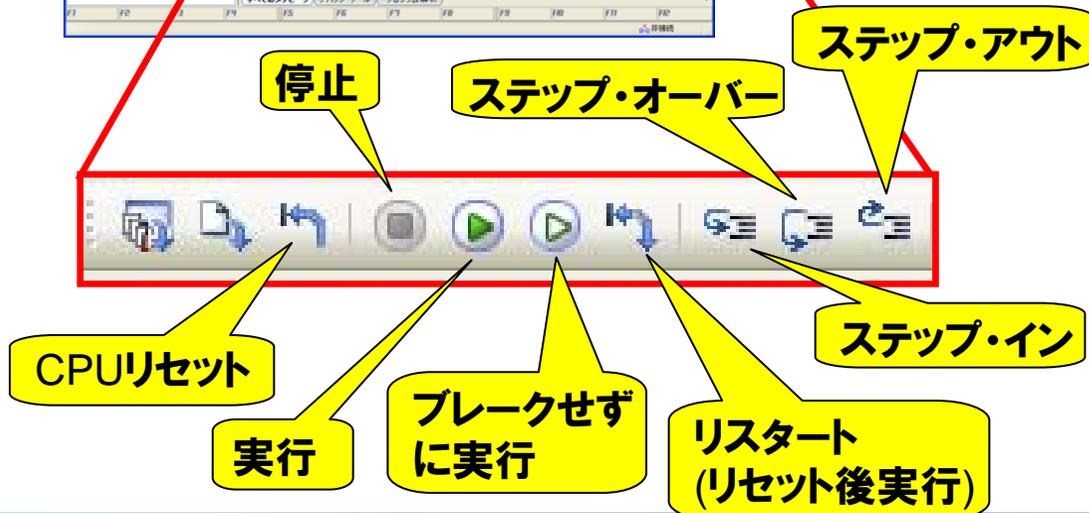
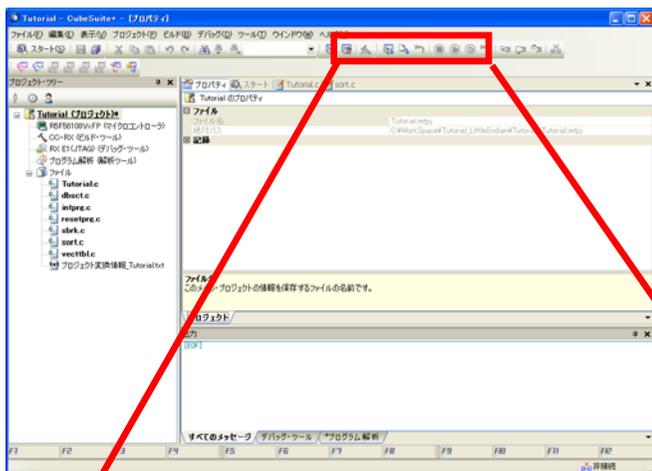
11. ダウンロードファイルを追加登録する方法

(4) R8C,M16CとRL78でダウンロード可能なファイルフォーマット(拡張子)が異なります。詳細は下図を参照してください。

	ロードモジュール フォーマット	ヘキサファイル	バイナリファイル
R8C,M16C	*.x30 *.abs	*.mot *.hex	*.bin
RL78	*.lmf	*.hex	*.bin
用途	ソースレベルデバッグ 時にダウンロードする ファイル	ROMライター等で書き込 む場合に使用するファ イル	データファイル

12. プログラムの実行、停止方法

プログラムの実行、停止やCPUリセットはHEWと同様にメニューおよびツールバーから行うことができます。以下を参照ください。

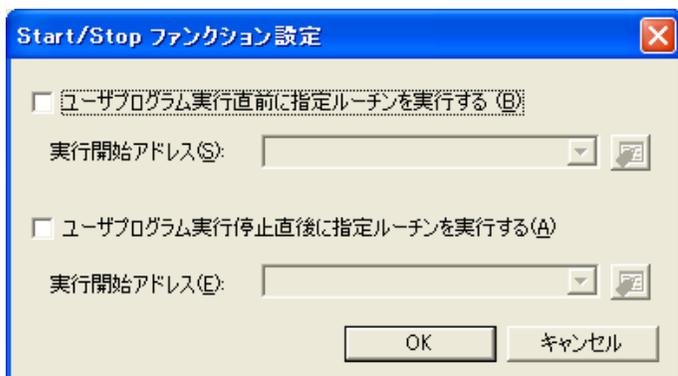
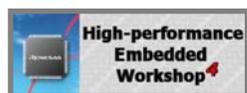


CubeSuite+のメニュー

デバッグ(D)	ツール(T)	ウインドウ(W)	ヘルプ(H)
デバッグ・ツールヘダダウンロード(D)			
ビルド&デバッグ・ツールヘダダウンロード(B)			F6
デバッグ・ツールへ接続(C)			
ホット・プラグイン(H)			
デバッグ・ツールからアップロード(U)...			
デバッグ・ツールから切断(N)			Shift+F6
停止(S)			Shift+F5
実行(Q)			F5
ステップ・イン(I)			F11
ブレークせずに実行(E)			F8
ステップ・オーバー(O)			F10
リターン・アウト(R)			Shift+F11
CPUリセット(T)			Ctrl+F5
リスタート(A)			

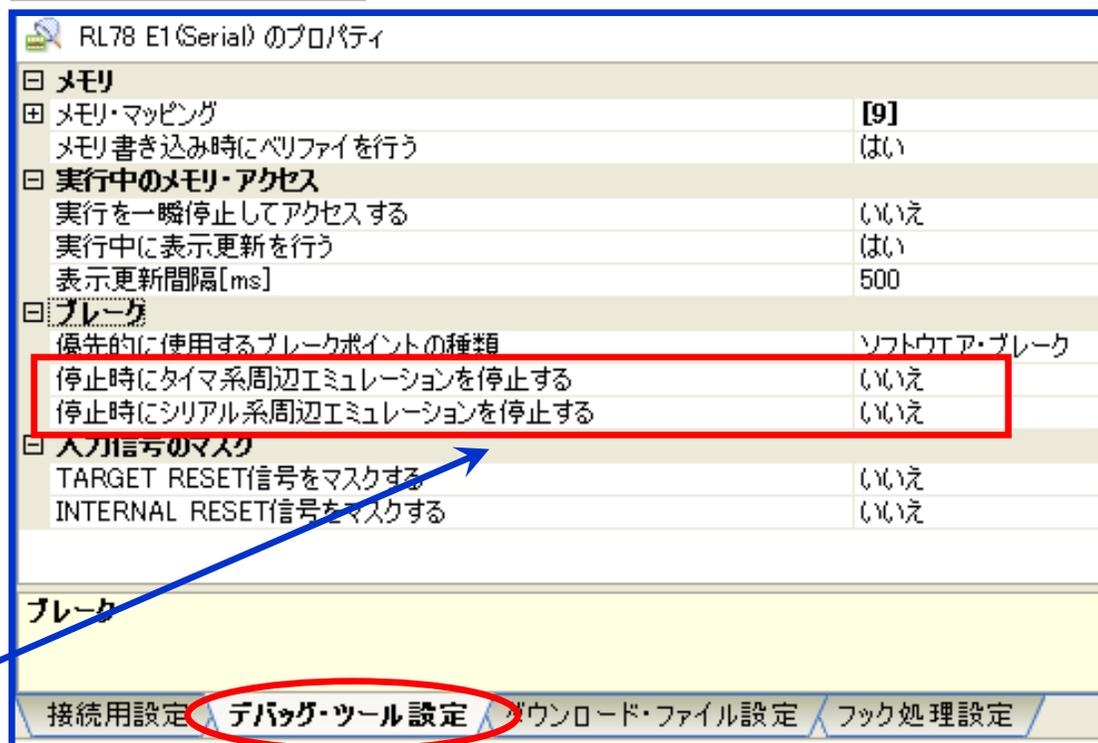
13. ブレーク時のマイコンの動作の違い(周辺ブレーク機能)

R8C,M16Cではエミュレータでブレーク時にタイマ、シリアルは継続して動作していましたが*、RL78ではCubeSuite+の[デバッグ・ツール設定]でブレーク時に動作か停止の選択ができるようになっています。



R8C,M16Cでは上図にあるようなStart/Stopファンクション機能を使用し周辺を停止させるコードを作成し埋め込む必要があった。

RL78ではCubeSuite+で設定可能



*R8C/5xシリーズはブレーク時に停止させることが可能です

14. プログラム実行中にメモリや変数を参照・変更する方法

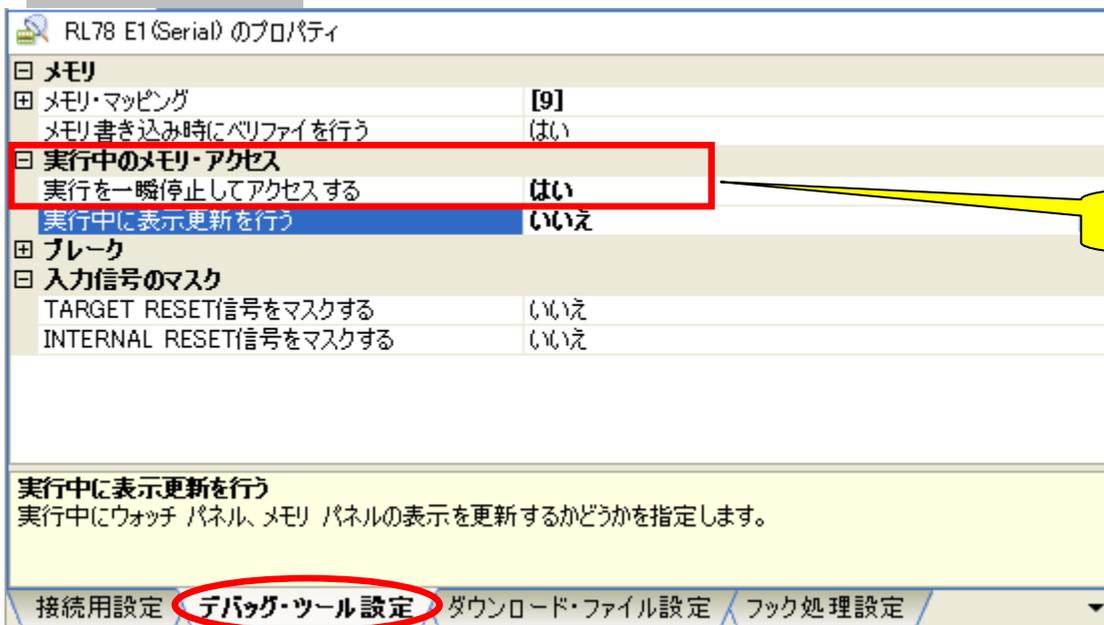
CubeSuite+でプログラム実行中にメモリや変数の内容を参照、変更するには [プロパティ] パネルの設定が必要です。

以下の手順で設定を行ってください。

(1) デバッグツールの [プロパティ] パネルの [デバッグ・ツール設定] タブを開いてください。

(2) 実行中のメモリ・アクセスの [実行を一瞬停止してアクセスする] を [はい] に設定してください。プログラム実行中にメモリ、変数の内容を参照・変更できるようになります。

 CubeSuite+



RL78 E1 (Serial) のプロパティ	
<input type="checkbox"/> メモリ	
<input type="checkbox"/> メモリ・マッピング	[9]
メモリ書き込み時にベリファイを行う	(はい)
<input type="checkbox"/> 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	はい
実行中に表示更新を行う	いいえ
<input type="checkbox"/> ブレーク	
<input type="checkbox"/> 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
実行中に表示更新を行う 実行中にウォッチ パネル、メモリ パネルの表示を更新するかどうかを指定します。	
接続用設定	デバッグ・ツール設定
ダウンロード・ファイル設定	フック処理設定

ここを「はい」に変更

本設定を「いいえ」にした場合、プログラムの実行中メモリパネルは**を表示します。

15. プログラム実行中にメモリや変数を自動更新する方法

CubeSuite+でメモリや変数の内容を自動更新するには [プロパティ] パネルの設定が必要です。以下の手順で設定を行ってください。

(1) デバッグツールの [プロパティ] パネルの [デバッグ・ツール設定] タブを開いてください。

(2) 実行中のメモリ・アクセスの [実行を一瞬停止してアクセスする] と [実行中に表示更新を行う] を [はい] に設定してください。

プログラム実行中にメモリパネル、ウォッチパネルの表示内容を自動更新します。表示間隔を変更したい場合は [表示更新間隔] の値を修正してください。

2箇所を「はい」に変更

項目	設定
メモリ	
メモリ・マッピング	[9]
メモリ書き込み時にベリファイを行う	(はい)
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	はい
実行中に表示更新を行う	はい
表示更新間隔[ms]	500
ブレーク	
入力信号のマスク	

表示更新間隔[ms]
実行中にウォッチ パネル、メモリ パネルの表示を更新する間隔を100ms単位で指定します。100から65500までの数値の指定が可能です。端数は切り上げます。

表示間隔を設定

ウォッチ1

ウォッチ式	値	型情報
count2	15 (0x0000000f)	int (4)
led3	0 (0x00000000)	int (4)

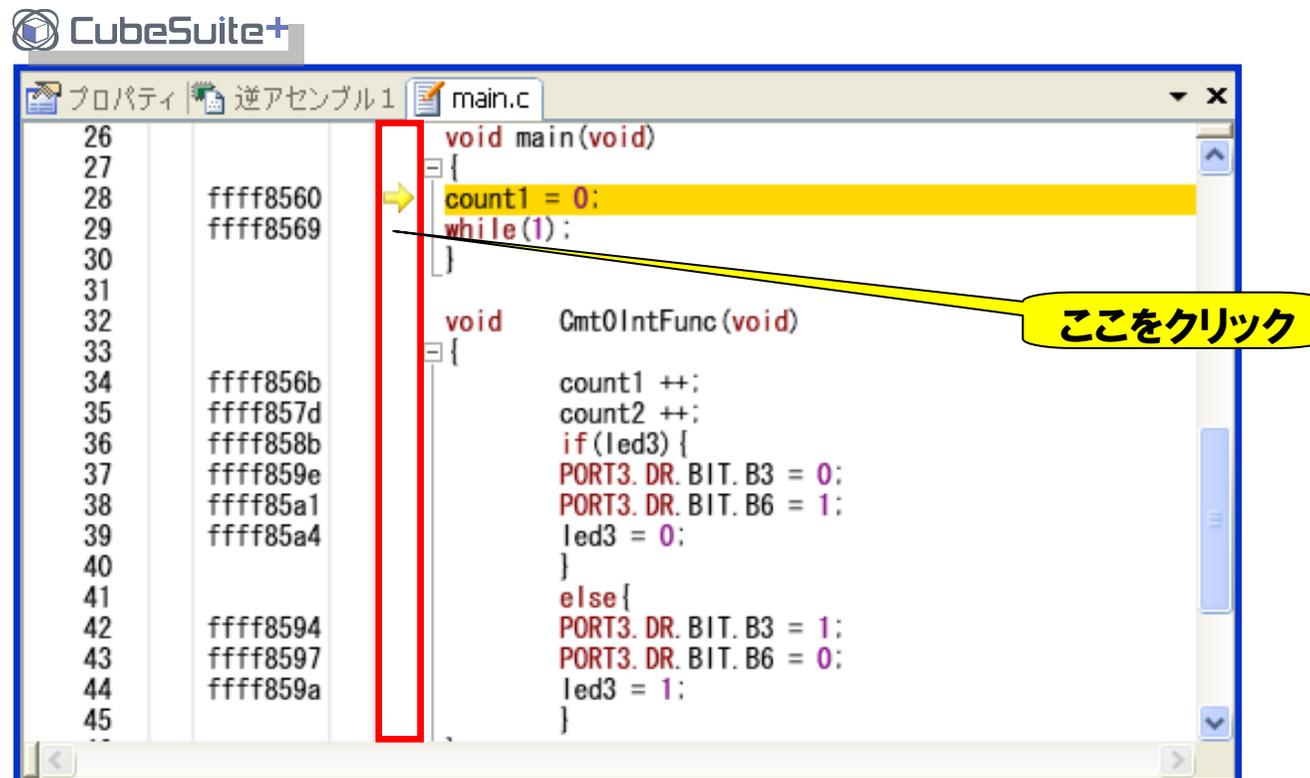
プログラム実行中にウォッチパネルに登録した変数やメモリパネルの内容を自動的に更新して表示します。

16. ブレークポイントの設定方法

(1) CubeSuite+でのブレークポイント設定はエディタパネルのメイン・エリア (下図の赤で囲っている部分) で行います。

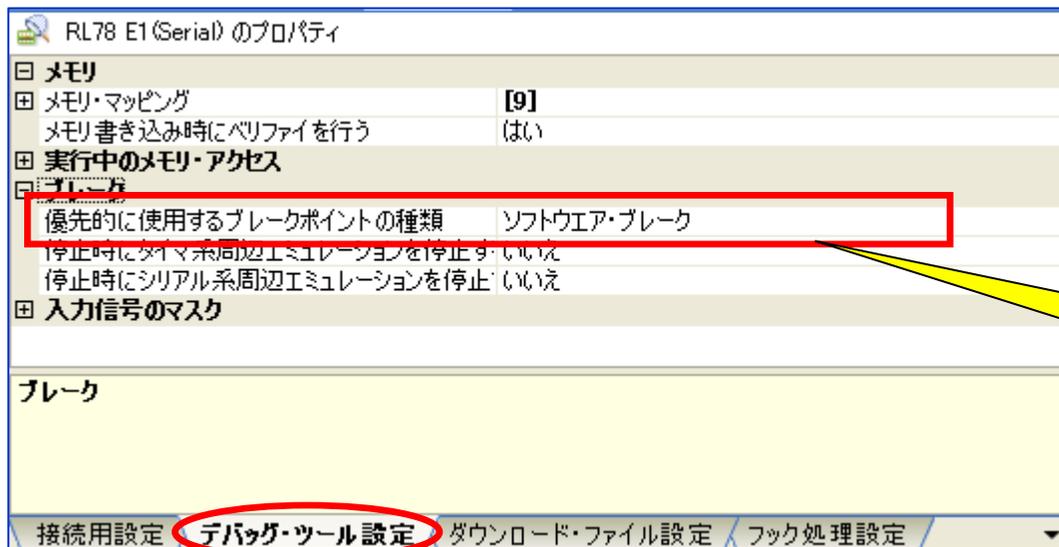
アドレス表示のある行を1クリックすることでブレークポイントを設定できます。

ブレークポイント設定済みの行を1クリックした場合はブレークポイントを削除します。



16. ブレークポイントの設定方法

(2) ソフトウェアブレーク、ハードウェアブレークのうちどちらを設定するかは [プロパティ] パネルの [デバッグ・ツールの設定] タブの [優先的に使用するブレークポイント] で選択します。(下図の例ではソフトウェア・ブレークを設定しています)



優先的に使用する
ブレークポイントを設定

(3) 上記で選択したブレークポイントの設定数が制限を越える場合、もう一方の種類のブレークポイントが使用されます。

どちらのブレークが割り当てられたかは、イベントマークで区別できます。

 がソフトウェアブレーク、 がハードウェアブレークのイベントマークです。

16. ブレークポイントの設定方法

- (4) ブレークポイントの設定状態は [イベント] パネルで確認できます。
[イベント] パネルはCubeSuite+のメニューから [表示] → [イベント] で開きます。
[イベント] パネルで不要なブレークポイントを削除したり、無効にすることができます。



17. 変数へのアクセスでブレークする方法

変数へのアクセスでブレークする設定はウォッチパネルまたはエディタパネルで行います。

(1) ウォッチパネルまたはエディタパネル上のアクセス時にブレークしたい変数を右クリックしてポップアップメニューを開いてください。

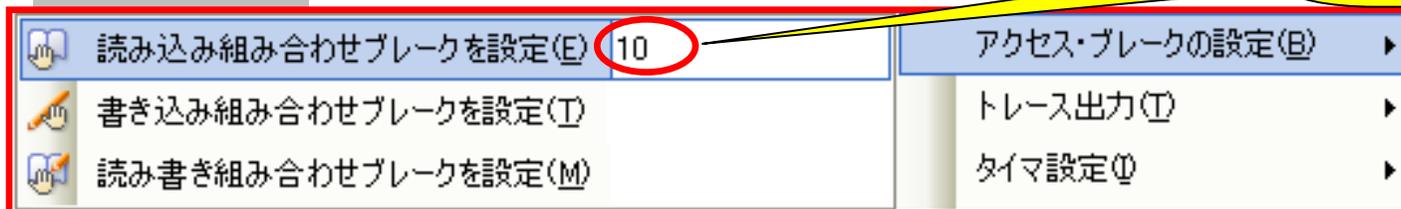
(2) アクセス・ブレークの設定 (エディタパネルの場合はブレークの設定) を選択して、「読み込み組み合わせブレークを設定」「書き込み組み合わせブレークを設定」「読み書き組み合わせブレークを設定」のいずれかを設定してください。

The screenshot shows the CubeSuite+ interface. On the left, the 'ウォッチ1' (Watch 1) window displays a table with columns for 'ウォッチ式' (Watch Expression), '値' (Value), and '型情報(バイト数)' (Type Information (Byte Count)). The entry for 'g_count' is highlighted with a red circle. On the right, the code editor shows the function 'interrupt void R_TAU0_Channel10_Interrupt' with the line 'g_count++;' circled in red. A yellow callout bubble points to this line with the text '変数を右クリックしてポップアップメニューを開く' (Right-click the variable to open the pop-up menu). Another yellow callout bubble points to the 'g_count++;' line with the text 'アクセス・ブレークの設定を選択' (Select the access breakpoint setting). At the bottom, a context menu is open, showing options: '読み込み組み合わせブレークを設定(B)' (Set load combination breakpoint), '書き込み組み合わせブレークを設定(I)' (Set store combination breakpoint), '読み書き組み合わせブレークを設定(M)' (Set read/write combination breakpoint), 'アクセス・ブレークの設定(B)' (Access breakpoint setting), 'ブレーク条件(C)' (Breakpoint condition), and 'タイマ設定(T)' (Timer setting). A red circle highlights the 'アクセス・ブレークの設定(B)' option, with a yellow callout bubble pointing to it that says 'ブレーク条件(読み込み、書き込み、読み書き)を選択' (Select breakpoint condition (load, store, read/write)).

17. 変数へのアクセスでブレークする方法

(3) データ条件を設定する場合は数値を入力してください。(不要な場合は入力しなくてもかまいません)

CubeSuite+



注:ここで入力する数値は10進数です。16進数で数値入力する場合は、0xAAのように先頭に"0x"を付けてください。

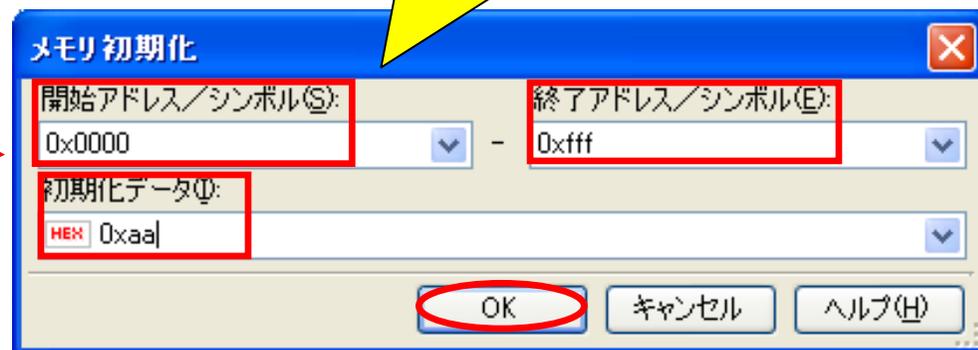
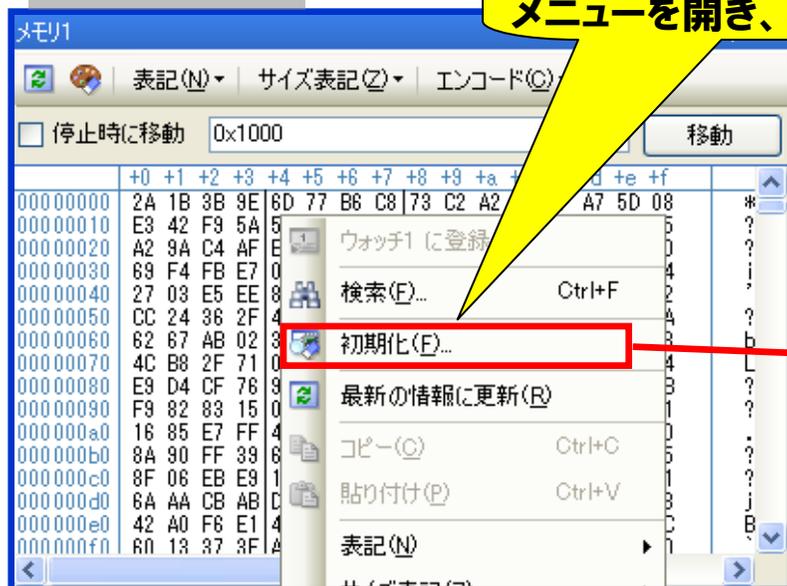
18. メモリをフィルする方法

メモリフィル（一括して変更）はメモリ初期化ダイアログで行います。

(1) [メモリ] パネル内を右クリックしてポップアップメニューを開いて、[初期化] を選択してください

(2) メモリ初期化ダイアログが開きますので、初期化したいアドレス（開始アドレスと終了アドレス）を初期化データを設定後OKボタンを押してください。

CubeSuite+



注:ここで入力する数値は10進数です。16進数で数値入力する場合は、先頭に"0x"を付けてください。

19. メモリ内容をセーブする方法

CubeSuite+でメモリ内容をセーブする場合は、[デバッグ・ツールからアップロード]を使用します。

メニューから[デバッグ] → [デバッグ・ツールからアップロード]を選択してください。データ保存ダイアログが開きますので、保存したいファイル名、ファイルの種類、保存範囲を設定して、保存ボタンを押してください。



The screenshot shows the 'データ保存 - デバッグ・ツールからアップロード' dialog box. It has three main sections highlighted with red boxes: the 'ファイル名(N):' field, the 'ファイルの種類(T):' dropdown menu, and the '保存範囲 アドレス/シンボル(B):' section. The '保存' button is circled in red. Three yellow callout boxes point to these sections with the following text:

- 保存ファイル名を設定します。
- ファイルの種類(インテルヘキサ、モトローラS、バイナリ)を設定します。
- 保存範囲を設定します。

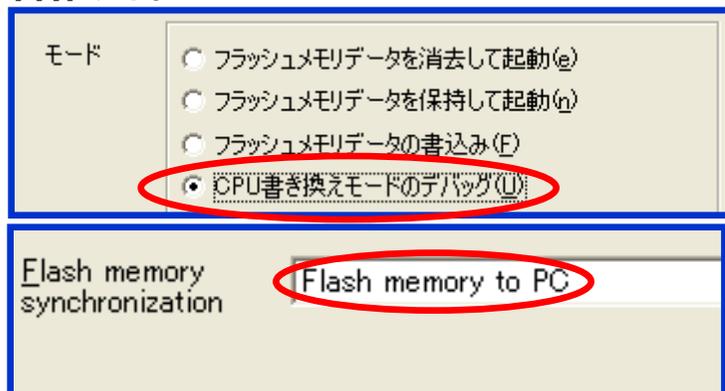
注:ここで入力する数値は10進数です。16進数で数値入力する場合は、先頭に"0x"を付けてください。

20. フラッシュ・セルフ・プログラミングについて

RL78は、ユーザ・プログラムでフラッシュ・メモリの書き換えを行うためのセルフ・プログラミング機能をサポートしています。この機能はRL78セルフ・プログラミング・ライブラリを利用することによりユーザ・アプリケーションでフラッシュ・メモリの書き換えを可能にします。

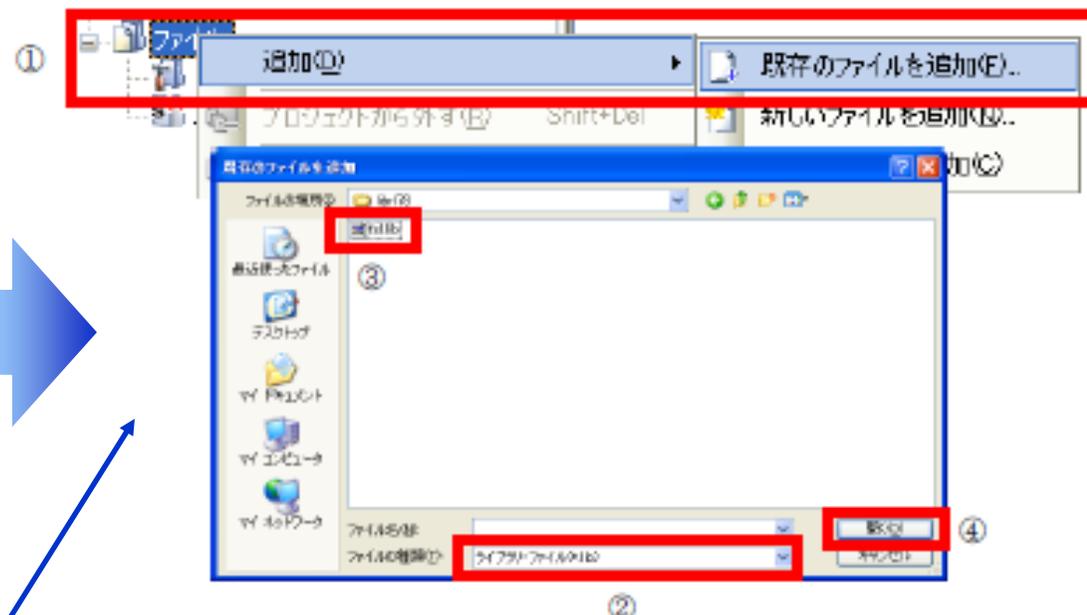


R8C,M16C(HEW)ではデバッガを専用のモードに変更し、フラッシュ書き換えプログラムは基本的には自作する。



RL78(CubeSuite+)では無償で提供されるセルフ・プログラミング・ライブラリをプロジェクトに組み込む。
組み込み方法の詳細は以下のURLを参照。

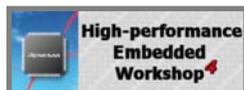
http://japan.renesas.com/products/tools/flash_programming/flash_libraries/



CubeSuite+の[プロジェクト・ツリー]にある, [ファイル] を右クリックして表示されるリストから[追加]を選択し [既存のファイルを追加] 画面を表示する。次に[ファイルの種類]のプルダウンメニューをクリックしファイルの種類を選択しフラッシュ・セルフ・プログラミング関連のファイルを登録する。

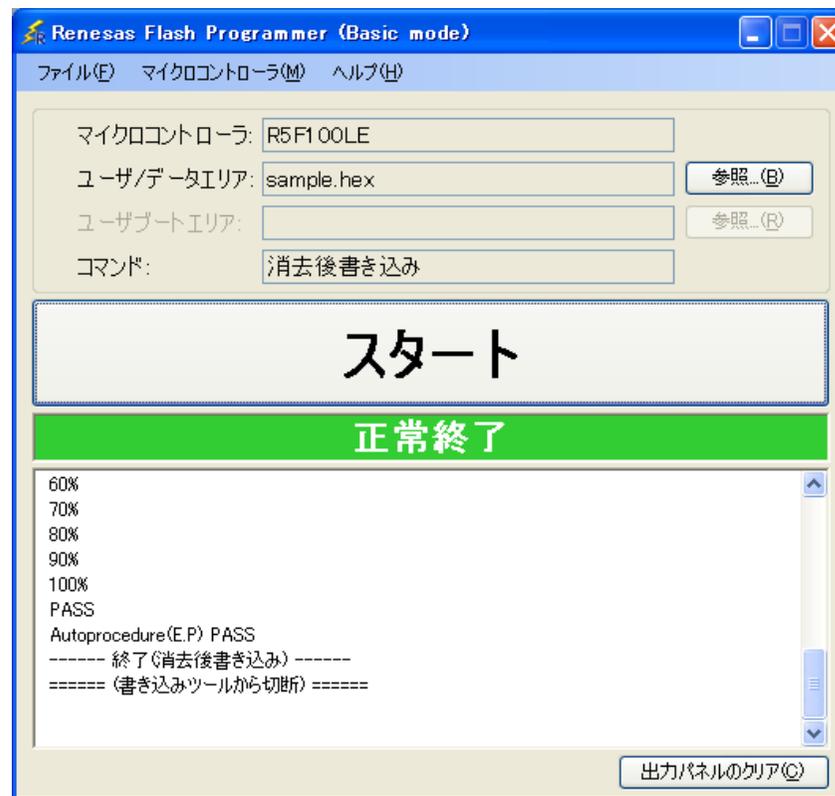
21. デバイス単体で動作確認するための書き込み方法

RL78ではデバッグ後にデバイス単体で動作確認を行う場合は、CubeSuite+ではなくフラッシュ書き込みソフトRenesas Flash Programmerを使用して書き込みを行ってください。



R8C,M16C(HEW)ではデバッガ用のモードの他にオンボードライターと同等の機能を持つモードを選択する。

モード	<input type="radio"/> フラッシュメモリデータを消去して起動(e)
	<input type="radio"/> フラッシュメモリデータを保持して起動(n)
	<input checked="" type="radio"/> フラッシュメモリデータの書き込み(F)
	<input type="radio"/> CPU書き換えモードのデバッグ(U)
	<input type="checkbox"/> デバッガ終了時にユーザープログラムを実行(x)



RL78ではCubeSuite+ではなく
Renesas Flash Programmerを使用して書き込む

Renesas Flash Programmerは、ルネサス製フラッシュ内蔵マイコンのフラッシュメモリに対し、書き込みを行うためのソフトウェアです。書き込みに特化した操作性・機能を提供します。

22. アクション・イベント機能(Printfイベント)

CubeSuite+ではアクション・イベントとしてPrintfイベントの設定が可能です。

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、ソフトウェ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。

CubeSuite+

```

69      00196      outreg = inreg      1;
70      0019e      P6.2 = outreg;
71      001a4      P6.3 = inreg;
72      001ab      g_count++;
73      /* End user code. Do not edit comment generated here */
74      001ae
75

```

この行に`printf("サンプル: g_count = %d, P6 = %d¥n",g_count,P6)`に相当するコードを挿入したい ⇒ アクションイベントの設定

アクション・イベント

Printf イベント

出力文字列(O): 入力例) サンプル:
 サンプル:|

変数式(V): 入力例) aaa, bbb, ccc
 g_count,P6

アドレス(A):

出力パネルに結果が出力

出力

```

↓
サンプル: g_count=0(0x0000), P6=0x9(0x09) ↓
サンプル: g_count=1(0x0001), P6=0x5(0x05) ↓
サンプル: g_count=2(0x0002), P6=0x9(0x09) ↓
サンプル: g_count=3(0x0003), P6=0x5(0x05) ↓
サンプル: g_count=4(0x0004), P6=0x9(0x09) ↓
サンプル: g_count=5(0x0005), P6=0x5(0x05) ↓
サンプル: g_count=6(0x0006), P6=0x9(0x09) ↓
ソフトウェア・ブレークにより停止しました。 ↓
[EOF]

```

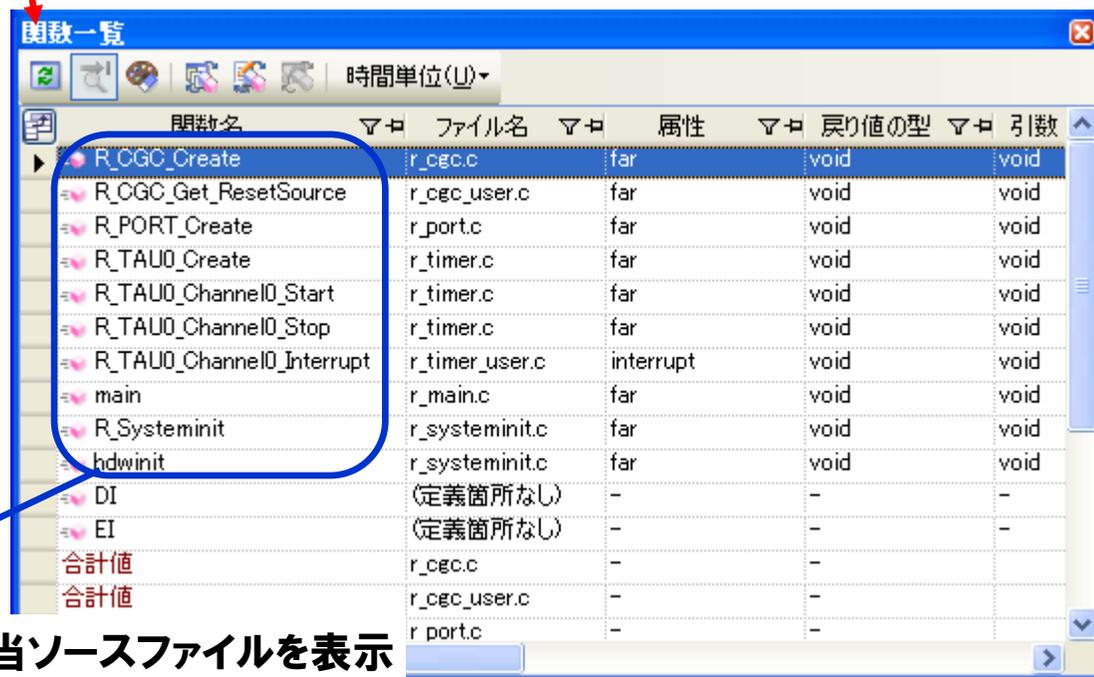
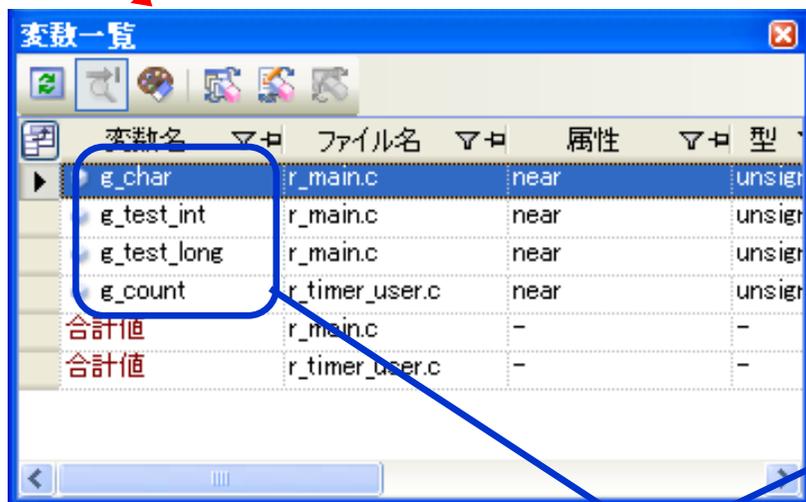
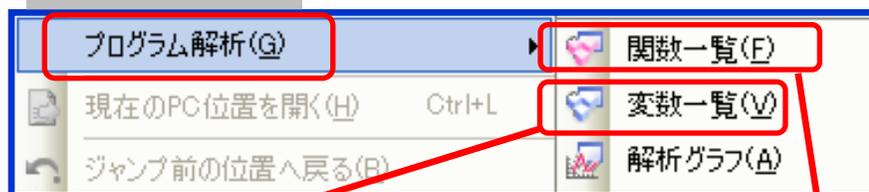
すべてのメッセージ *ビルド・ツール *デバッグ・ツール

23. 変数と関数の一覧表示機能

CubeSuite+でプロジェクトで使用している変数と関数の一覧を自動的に表示することができます。

メニューから[表示]→[プログラム解析]を選択してください。

CubeSuite+

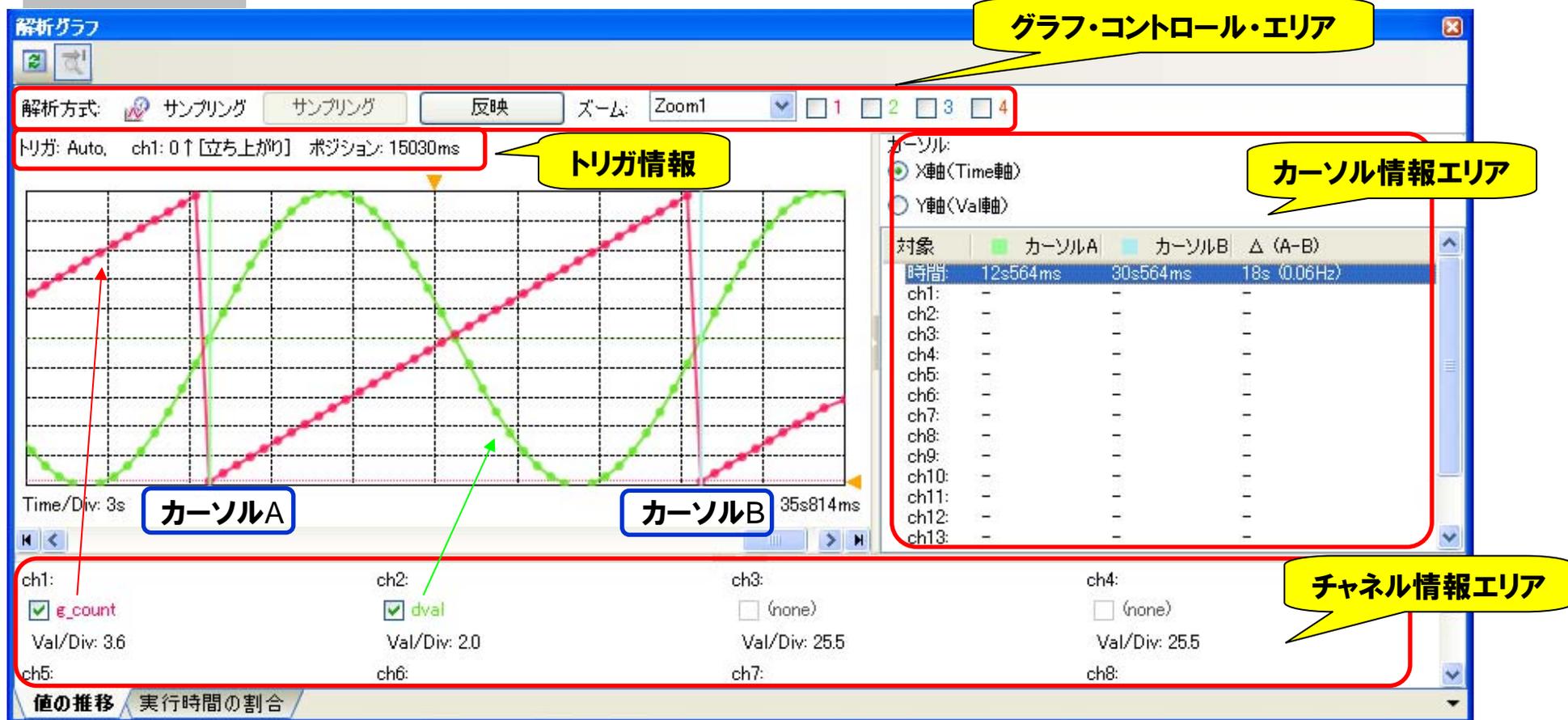


変数名や関数名をクリックすると該当ソースファイルを表示

24. 解析グラフ機能

CubeSuite+にはグラフ化の対象として登録した変数／レジスタ／アドレス等の値と時間の関係を折れ線グラフで表示する解析グラフ機能があります。

RL78のオンチップ・デバッグでは疑似RRM機能により取得したデータを基にグラフの表示を行います。



25. エミュレータデバッグ機能比較(OCD)

デバッグ機能		RL78(E1/E20)	R8C(E8a/E1/E20)	M16C(E8a)
ブレーク	ソフトウェア・ブレーク	2000 点	255点	255点
	ハードウェア・ブレーク	実行・アクセス兼用で 1~2 点*	実行・アクセス兼用で 2点~10点*	実行・アクセス兼用で 6点~10点*
	強制ブレーク	可能	可能	可能
イベント	設定可能数	実行・アクセス兼用で 1~2 点*	1点~2点*	0点~2点*
	イベント使用機能	ハードウェア・ブレークのみ	ハードウェア・ブレークのみ	ハードウェア・ブレークのみ
トレース		分岐トレース*	分岐トレース/ データトレース*	分岐トレース/ データトレース*
パフォーマンス測定	測定項目	実行開始~停止	実行開始~停止	実行開始~停止
	性能	分解能 100 μ s, 最大測時間 100 時間	E8a : 分解能 1ms 注: リソースはPCのタイマ E1/E20 : 分解能 1 μ s	分解能 1ms 注: リソースはPCのタイマ
疑似リアルタイムRAMモニタ (RRM)		可能 (モニタ時にCPU を占有)	可能(モニタ時にCPU を占有/ デバッグDMAC)	可能(モニタ時にCPU を占有/ デバッグDMAC)
Dynamic Memory Modification (DMM)		可能 (変更時に CPU を占有)	可能(変更時に CPU を占有/ デバッグDMAC)	可能(変更時に CPU を占有/ デバッグDMAC)
ホットプラグイン		不可	不可	不可
セキュリティ		10 バイト ID 認証 **	7バイトID認証**	7バイトID認証**
占有端子数		1本(TOOL0端子)	1本(MODE端子)	1本or2本or7本*
周辺ブレーク		可能	不可(R8C/5xIは可)	不可

*対象デバイスにより機能が異なります。 **IDコードの仕様の違いに関しては本資料の「IDコードの入力方法」を参照してください。

RENESAS

ルネサス エレクトロニクス株式会社