User's Manual

32

RX71M グループ

Renesas Starter Kit+ チュートリアルマニュアル (CS+)

ルネサス 32 ビットマイクロコントローラ
RX ファミリ/RX700 シリーズ

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 2015.01

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本文を参照してください。なお、本マニュアルの本文と異なる記載がある場合は、本文の記載が優先するものとします。

1. 未使用端子の処理
   【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。
   CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。
2. 電源投入時の処置
   【注意】電源投入時は，製品の状態は不定です。
   電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。
   同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. リザーブアドレスのアクセス禁止
   【注意】リザーブアドレスのアクセスを禁止します。
   アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレスがあります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
4. クロックについて
   【注意】リセット時は、クロックが安定した後、リセットを解除してください。
   プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。
   リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
5. 製品間の相違について
   【注意】型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。
   同じグループのマイコンでも型名が違うと、内部 ROM、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

このマニュアルの使い方

1. 目的と対象者

このマニュアルは、RSK+プラットフォーム用ソフトウェアを開発し、デバッグするためにCS+を使用する方法を理解していただくためのマニュアルです。様々な周辺装置を使用して、RSK+プラットフォーム上のサンプルコードを設計するユーザを対象にしています。

このマニュアルは、段階的にCS+中のプロジェクトをロードし、デバッグする指示を含みますが、RSK+プラットフォーム上のソフトウェア開発のガイドではありません。

このマニュアルを使用する場合、注意事項を十分確認の上、使用してください。注意事項は、各章の本文中、各章の最後、注意事項の章に記載しています。

改訂記録は旧版の記載内容に対して訂正または追加した主な箇所をまとめたものです。改訂内容すべてを記録したものではありません。詳細は、このマニュアルの本文でご確認ください。

RSK+RX71M では次のドキュメントを用意しています。ドキュメントは最新版を使用してください。最新版はルネサスエレクトロニクスのホームページに掲載されています。

ドキュメントの種類	記載内容	資料名	資料番号
ユーザーズマニュアル	RSK+ハードウェア仕様の説明	RSK+RX71M ユーザーズマニュアル	R20UT3217JG
チュートリアルマニュアル	RSK+および開発環境のセットアップ方法とデバッギング方法の説明	RSK+RX71M チュートリアルマニュアル	R20UT3218JG (本マニュアル)
コード生成支援ツール チュートリアルマニュアル	コード生成支援ツールの使用方法の説明	RSK+RX71M コード生成支援ツール チュートリアルマニュアル	R20UT3220JG
クイックスタートガイド	A4 紙一枚の簡単なセットアップガイド	RSK+RX71M クイックスタートガイド	R20UT3219JG
回路図	CPU ボードの回路図	RSK+RX71M CPU ボード回路図	R20UT3216EG
ユーザーズマニュアル ハードウェア編	ハードウェアの仕様（ピン配置、メモリマップ、周辺機能の仕様、電気的特性、タイミング）と動作説明	RX71M グループ ユーザーズマニュアル ハードウェア編	R01UH0493JJ

2. 略語および略称の説明

略語／略称	英語名	備考
ADC	Analog-to-Digital Converter	A/D コンバータ
API	Application Programming Interface	アプリケーションプログラムインタフェース
bps	Bits per second	転送速度を表す単位、ビット/秒
CMT	Compare Match Timer	コンペアマッチタイマ
COM	COMmunications port referring to PC serial port	シリアル通信方式のインタフェース
CPU	Central Processing Unit	中央処理装置
DVD	Digital Versatile Disc	ディジタルヴァーサタイルディスク
E1	Renesas On-chip Debugging Emulator	ルネサスオンチップデバッギングエミュレータ
GUI	Graphical User Interface	グラフィカルユーザインタフェース
IDE	Integrated Development Environment	統合開発環境
IRQ	Interrupt Request	割り込み要求
LCD	Liquid Crystal Display	液晶ディスプレイ
LED	Light Emitting Diode	発光ダイオード
LSB	Least Significant Bit	最下位ビット
LVD	Low Voltage Detect	電圧検出回路
MCU	Micro-controller Unit	マイクロコントローラユニット
MSB	Most Significant Bit	最上位ビット
PC	Personal Computer	パーソナルコンピュータ
Pmod™	-	Pmod は Digilent Inc.の商標です。Pmod インタフェース明細は Digilent Inc.の所有物です。Pmod 明細については Digilent Inc.の Pmod License Agreement ページを参照してください。
PLL	Phase-locked Loop	位相同期回路
RAM	Random Access Memory	ランダムアクセスメモリ
ROM	Read Only Memory	リードオンリーメモリ
RSK+	Renesas Starter Kit+	ルネサススタータキット
RTC	Realtime Clock	リアルタイムクロック
SAU	Serial Array Unit	シリアルアレイユニット
SCI	Serial Communications Interface	シリアルコミュニケーションインタフェース
SPI	Serial Peripheral Interface	シリアルペリフェラルインタフェース
TAU	Timer Array Unit	タイマアレイユニット
TFT	Thin Film Transistor	薄膜トランジスタ
TPU	Timer Pulse Unit	タイマパルスユニット
UART	Universal Asynchronous Receiver/Transmitter	調歩同期式シリアルインタフェース
USB	Universal Serial Bus	シリアルバス規格の一種
WDT	Watchdog timer	ウォッチドッグタイマ

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。

目次

1. 概要 7
1.1 目的 7
1.2 特徴 7

2. はじめに 8
2.1 コード生成プラグイン 8

3. チュートリアルプロジェクトワークスペース 9
3.1 はじめに 9
3.2 CS+の開始 9
3.3 デバッグ・ツールの設定 12
3.4 ビルド設定 13

4. チュートリアルプログラムのビルド 14
4.1 コードのビルド 14
4.2 エミュレータの接続 15
4.3 プロジェクトの保存 15

5. チュートリアルのダウンロードと実行 16
5.1 プログラムコードのダウンロード 16
5.2 コードの実行 16

6. チュートリアルレビュー 17
6.1 プログラム初期化 17
6.2 メイン関数 18

7. 追加情報 21

1. 概要

1.1 目的

本 RSK+はルネサスマイクロコントローラ用の評価ツールです。本マニュアルは、コードのダウンロードや基本的なデバッグ操作について説明しています。

1.2 特徴

本 RSK+は以下の特徴を含みます：

- ルネサスマイクロコントローラのプログラミング
- ユーザコードのデバッギング
- スイッチ、LED、ポテンショメータ等のユーザ回路
- サンプルアプリケーション
- 周辺機能初期化コードのサンプル

CPU ボードはマイクロコントローラの動作に必要な回路を全て備えています。

2. はじめに

本マニュアルは Renesas Starter Kit+（RSK+）をご使用の際、最も多く寄せられる質問に対し、チュートリアル形式でお答えするものです。チュートリアルでは以下の項目について説明しています。

- RSK+でプログラムをコンパイル、リンク、ダウンロードおよび実行する方法は？
- 組み込みアプリケーションの構築方法は？
- ルネサスツールの使用方法は？

プロジェクトジェネレータは、選択可能な 3 種類のビルドコンフィグレーションを持つチュートリアルプロジェクトを作成します。

- ‘DefaultBuild’はデバッガのサポートおよび最適化レベル 2 を含むプロジェクトを構築します。
- ‘Debug’はデバッガのサポートを含むプロジェクトを構築します。最適化レベルは 0 に設定されています。
- ‘Release’は最適化された製品リリース用に適したコードを構築します。最適化レベルは 2 に設定されています。

本マニュアルで引用されたファイルはチュートリアルを進めていく過程でプロジェクトジェネレータを使用してインストールされます。本チュートリアルの使用例はクイックスタートガイドに記載のインストールが完了していることを前提としています。

本マニュアル中のソースコード画面のライン番号が実際のソースコードと異なる場合がありますが、本マニュアルで記載されている内容と機能的違いはございません。

チュートリアルは RSK+の使用方法の説明を目的とするものであり、CS+、コンパイラまたは E1 エミュレータの入門書ではありません。これらに関する詳細情報は各関連マニュアルを参照してください。

2.1 コード生成プラグイン

本製品で提供しているサンプルコードの一部はコード生成を使用してコードを生成しています。コード生成は C ソースコードの生成、マイクロコントローラのプロジェクト設定を行うためのプラグインツールです。コード生成は直感的な GUI を使用することで、様々なマイクロコントローラの周辺機能や動作に必要なパラメータを設定することができ、開発工数の大幅な削減が可能です。

CS+インストール後、コード生成プラグインは有効になっておらず、本マニュアルでは必要としません。詳細はコード生成支援ツールチュートリアルマニュアルの「4.2 コード生成の有効化」を参照してください。

コード生成によって生成されるコードは、特定の周辺ごとに 3 つのコードを生成します（「r_cg_xxx.h」、「r_cg_xxx.c」、「r_cg_xxx_user.c」）。例えば A/D コンバータの場合、周辺を表す xxx は’adc’と名付けられます。これらのコードはユーザの要求を満たすために、カスタムコードを自由に加えることができます。カスタムコードを加える場合、以下に示すコメント文の間にカスタムコードを加えてください。

```
/* Start user code for adding. Do not edit comment generated here */
/* End user code. Do not edit comment generated here */
```

コード生成の GUI 上で設定した内容を変更したい場合等、再度コード生成を行う場合にコード生成はこれらのコメント文を見つけて、コメント文の間に加えられたカスタムコードを保護します。

3. チュートリアルプロジェクトワークスペース

3.1 はじめに

CS+はルネサス統合開発ツールで、ユーザはこれを使用してルネサスマイクロコントローラのソフトウェアプロジェクトをコンパイル、プログラム、デバッグすることができます。CS+は Renesas Starter Kit+製品インストール時にインストールされます。本マニュアルでは、Tutorial コードの作成およびデバッグに必要な作業を段階的に説明します。

3.2 CS+の開始

CS+起動方法は以下の通りです。

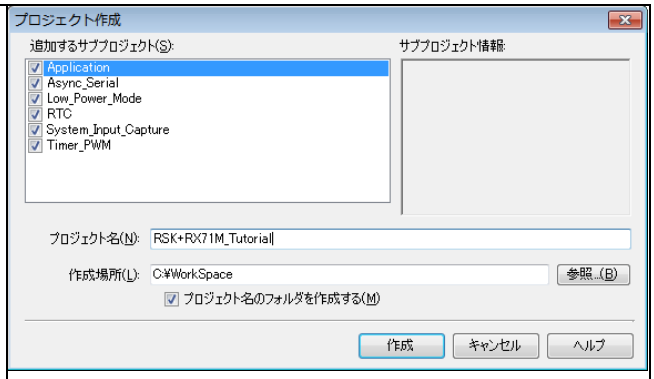
Windows™ Vista/7: スタートメニュー > すべてのプログラム > Renesas Electronics CS+ > CS+ for CC (RX, RH850)

Windows™ 8: ⓓをクリックして[アプリ]ビューを表示 > “CS+ for CC (RX, RH850)”アイコン

CS+を初めて使用する場合、ワンポイントアドバイスのダイアログが表示されます。

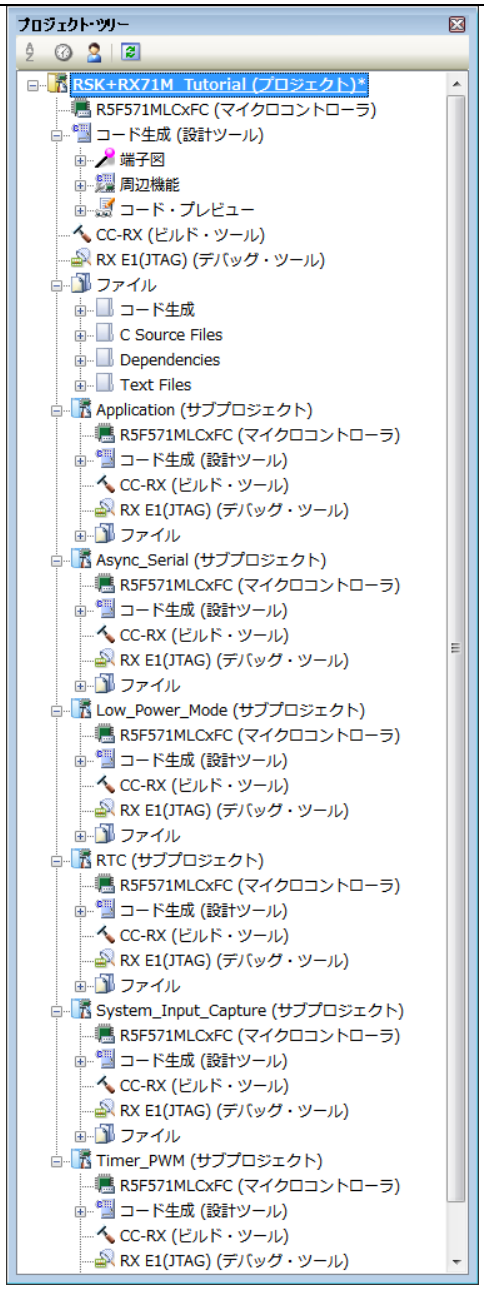
<OK>をクリックし、ダイアログを閉じてください。その後、スタートパネルが現れます。’RSK のサンプル・プロジェクトを開く’から RSK+RX71M_Tutorial を選択し、<GO>をクリックしてください。この操作によって、RSK+RX71M_Tutorial プロジェクトのコピーを保存します。

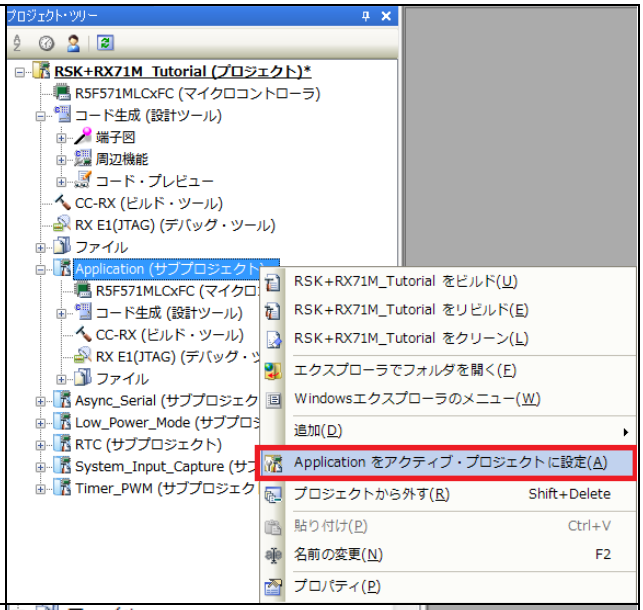
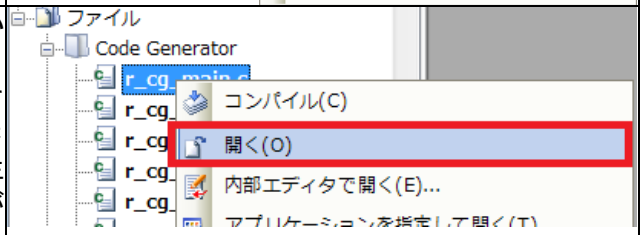
- CS+は‘プロジェクト作成’ダイアログを表示します。
- 各サブプロジェクト名のチェックボックスをチェックし、サブプロジェクトをすべて追加してください。各サブプロジェクトの情報はダイアログ上のサブプロジェクト情報の下に表示されます。
- プロジェクト名を入力し、作成場所を指定して<作成>をクリックしてください。
- コピーされたファイルを参照するには、プロジェクトツリーにリスト化されたファイルをダブルクリックしてください。新しいウィンドウが開きます。



- CS+はスクリーンショットに見られるようなプロジェクトツリーを持つプロジェクトを作成し、開きます。
- スクリーンショットのファイルフォルダはマスタープロジェクト RSK+RX71M_Tutorial に属します。
- このフォルダは個別のフォルダ構造で用意されたテキストファイルを含むプロジェクトソースおよびヘッダファイルをすべて含んでおり、リストします。
- ファイルフォルダの下にサブプロジェクトがリストされます。
- 現在アクティブなプロジェクトはプロジェクト名に下線が含まれます。初期設定によってRSK+RX71M_Tutorial がアクティブ・プロジェクトに設定されています。

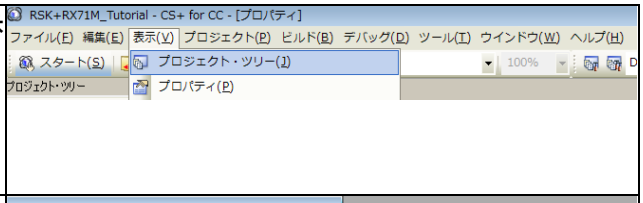
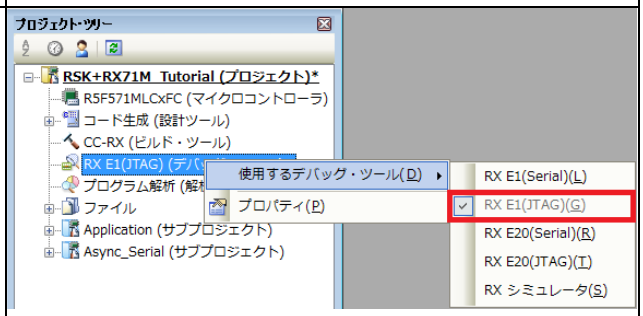
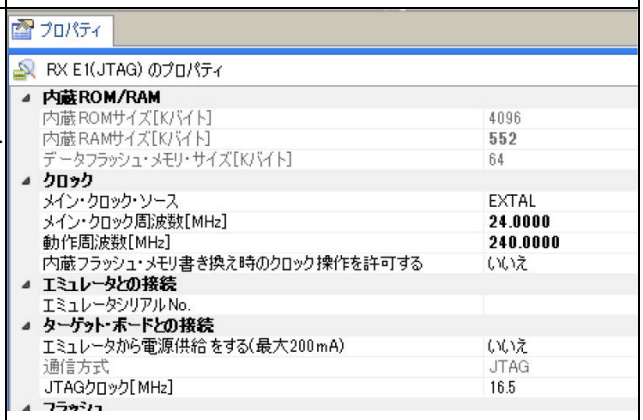
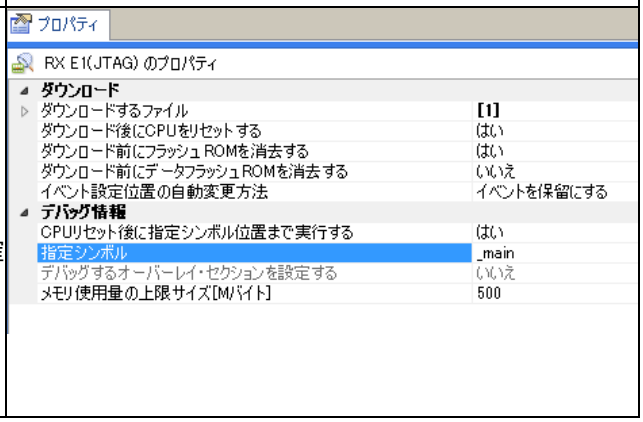
注：プラグインオプションが事前に有効になっているため、’コード生成(設計ツール)’がプロジェクト・ツリー上に表示されます。



• アクティブ・プロジェクトを変更するには、アクティブに変更したいプロジェクト/サブプロジェクトを右クリックし、”プロジェクト名/サブプロジェクト名をアクティブ・プロジェクトに設定”を選択します。 • スクリーンショットは Application サブプロジェクトをアクティブ・プロジェクトに変更する例です。	
• ファイルフォルダは 4 つのサブフォルダを含んでいます。 • いくつかのソース・ファイルはコード生成によって生成され、‘Code Generator’フォルダにグループ化されます。これらのファイルはコード生成によって生成されたことを示すため、ファイル名の前に’r_cg’が付けられます。他のユーザによって作成/インクルードされたファイルは Code Generator とは別のフォルダにリストされます。 • ファイルを参照するには、参照したいファイルを右クリックし、”開く”を選択します。ファイルをダブルクリックしても参照できます。	

3.3 デバッグ・ツールの設定

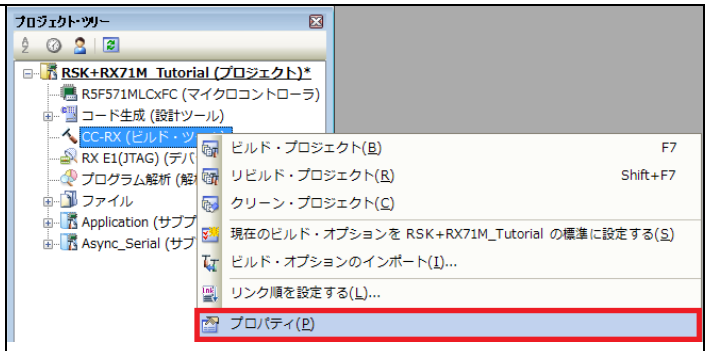
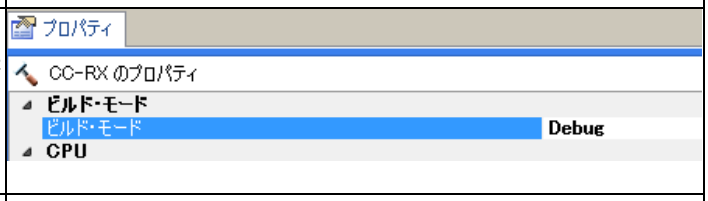
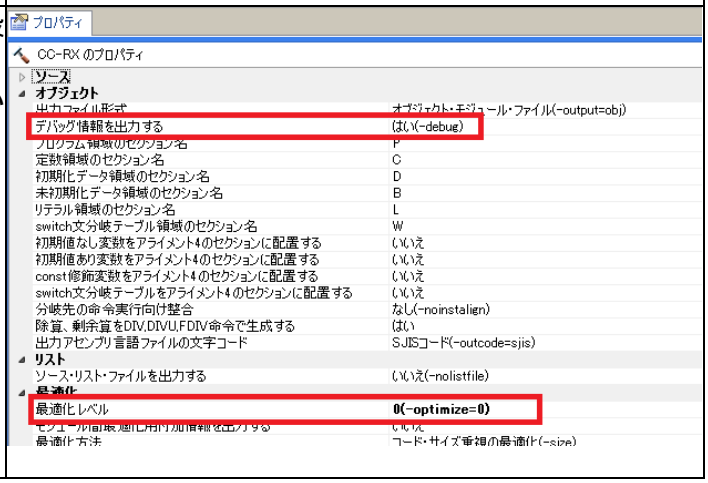
注：Tutorial プロジェクトは予めデバッグ・ツールの設定がされています。このセクションは新しいプロジェクトを作成するための説明です。

• プロジェクトツリーは CS+の左側ウィンドウに表示されます。 • これはメニューバーから起動することができます。（表示 -> プロジェクト・ツリー）	
• RX xxx (デバッグ・ツール)を右クリックし、RX E1 (JTAG)を選択してください。スクリーンショットは予め RX E1 (JTAG)が選択されています。	
• RX E1 (JTAG) (デバッグ・ツール)を右クリックし、プロパティを選択してください。 • 接続用設定タブをクリックしてください。 設定内容がスクリーンショットと同じであることを確認してください。	
プロジェクトはコードをダウンロードした後にメイン関数の先頭でコード実行を停止させる設定になっています。エントリポイントを別の関数に指定する場合： • ‘ダウンロード・ファイル設定’タブをクリックしてください。 • 指定シンボルを別の関数に変更してください。 • 関数名の前にアンダースコア（”_”）があることを確認してください。 注：割り込みハンドラをエントリポイントとして指定しないでください。	

3.4 ビルド設定

ビルド設定は CC-RX (ビルド・ツール)のプロパティから選択できます。利用可能なオプションは‘DefaultBuild’、‘Debug’、‘Release’です。‘DefaultBuild’および‘Debug’はデバッガを備えた設定になっています。‘Release’は最終の ROM 化用プログラムのために設定されます。

3 つのビルド間の共通の違いは、最適化セットおよびデバッガ設定です。最適化が有効の場合、デバッガがコードを予想外の順序で実行するようなケースがあります。デバッグをスムーズに処理する為には、デバッグされるコードの最適化を無効にすることを推奨します。

• CC-RX (ビルド・ツール)を右クリックし、プロパティを選択してください。	
• 共通オプションタブを選択してください。 • ビルド・モードを‘Debug’に設定してください。	
• ‘コンパイル・オプション’タブを選択してください。	
• デバッグ情報を出力するが’はい(-debug)’に設定されていることを確認してください。 • 最適化レベルが’ 0(-optimize=0)’に設定されていることを確認してください。	

4. チュートリアルプログラムのビルド

Tutorial プロジェクトのビルド設定は、ツールチェインオプションで既に設定されています。ツールチェインオプションを表示するためには、プロジェクトツリーの CC-RX (ビルド・ツール)をダブルクリックし、利用可能なタブを選択してください。

• 各タブで利用可能なオプションを確認してください。ここでは、デフォルトのオプション設定にしてください。 • 選択終了後に<X>をクリックしてプロパティ画面を閉じます。	

4.1 コードのビルド

プロジェクトのビルド用に 3 つのショートカットがあります。

• ツールバーの’プロジェクトをビルドします。’ボタンです。プロジェクトツリー中の全プロジェクトをビルドします。	
• キーボードの’F7’ボタンです。上記のボタン選択の場合と同じです。	
• ツールバーの’プロジェクトをリビルドします’ボタンです。プロジェクトファイルをすべてリビルドします。	
• ツールバーの’ビルド後デバッグ・ツールへプログラムをダウンロードします。(F6)”ボタンです。プロジェクトのビルドを行い，ビルド後にアクティブ・プロジェクトで現在選択しているデバッグ・ツールにダウンロードを実行します。	
• キーボードの’F6’ボタンはツールバーの’ビルド後デバッグ・ツールへプログラムをダウンロードします。’ボタンと同じです。	

ここで、キーボードの'F7'ボタンを押すか、または上記アイコンの 1 つを選択し、プロジェクトをビルドしてください。ビルド中の各段階で、アウトプットウィンドウにビルド状況が表示されます。ビルド終了時、ビルド中に発生したエラーおよび警告の表示がされます。

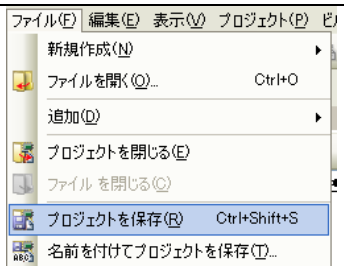
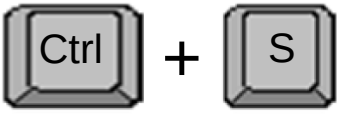
4.2 エミュレータの接続

本チュートリアルでは、外部から CPU ボードに電源を供給してください。外部電源を供給する際、極性および電源電圧が適切であることを必ず確認してください。

E1 のホストコンピュータへの接続方法は、クイックスタートガイドに詳しく記載されています。以下は、クイックスタートガイドの手順が踏まれ、E1 用のドライバが既にインストールされていることを前提としています。

- Pmod LCD を CPU ボードの PMOD1 コネクタに取り付け、コネクタの全てのピンがきちんとソケットに収まっていることを確認してください。
- E1 をご使用のコンピュータの USB ポートに接続してください。
- E1 を CPU ボードに接続します。'E1'のシルク印字のある E1 コネクタに接続してください。
- 外部電源を CPU ボードに供給します。'PWR'のシルク印字のある PWR コネクタに接続してください。

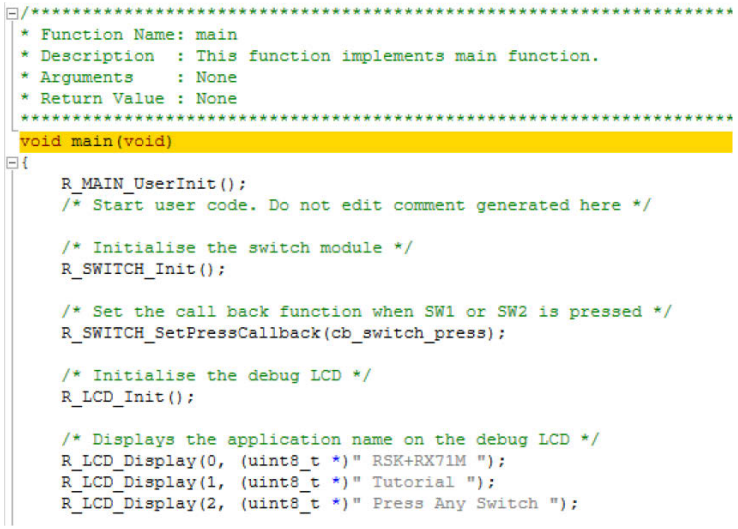
4.3 プロジェクトの保存

プロジェクトの設定を変更した場合、プロジェクトを保存することを推奨します。 • 'ファイル' | 'プロジェクトを保存' を選択します。	
CS+中のファイルを変更した場合、次の操作で保存することができます。 • 'ファイル' | 'すべてを保存' を選択します。	
ツールバーの'保存'または'すべてを保存'ボタンでファイルを保存することもできます。	
また、キーボードでファイルを保存することもできます。	

5. チュートリアルのダウンロードと実行

5.1 プログラムコードのダウンロード

CS+でコードのビルドが完了したら、プログラムを CPU ボード上のマイクロコントローラにダウンロードする必要があります。

• ツールバーの'ダウンロード'ボタンをクリックしてください。	
• ダウンロード完了後、デバッガおよびコードは実行準備ができています。プログラムカウンタ表示はメイン関数内の最初のインストラクションを示します。これは'r_cg_main.c'のプログラムエントリポイントです。	

```
/***********************************************************************
* Function Name: main
* Description  : This function implements main function.
* Arguments    : None
* Return Value : None
***********************************************************************/
void main(void)
{
    R_MAIN_UserInit();
    /* Start user code. Do not edit comment generated here */

    /* Initialise the switch module */
    R_SWITCH_Init();

    /* Set the call back function when SW1 or SW2 is pressed */
    R_SWITCH_SetPressCallback(cb_switch_press);

    /* Initialise the debug LCD */
    R_LCD_Init();

    /* Displays the application name on the debug LCD */
    R_LCD_Display(0, (uint8_t *)" RSK+RX71M ");
    R_LCD_Display(1, (uint8_t *)" Tutorial ");
    R_LCD_Display(2, (uint8_t *)" Press Any Switch ");
```

コードを実行する前に、コンピュータの USB ポートと CPU ボード上の USB シリアルポート（シルク印字'G1CUSB0'）を USB ケーブルで接続する必要があります。はじめて接続した場合、コンピュータの画面にドライバのインストールメッセージが表示され、自動的にデバイスドライバはインストールされます。デバイスマネージャ上のポート(COM と LPT)に'RSK USB Serial Port (COMx)'が現れますので、COM ポート番号を確認し、ターミナルソフトを起動して確認した COM ポート番号の設定を行ってください(ボーレート：19200、データ長：8、パリティ：なし、ストップビット：1、フロー制御：なし)。

5.2 コードの実行

プログラムが CPU ボード上のマイクロコントローラにダウンロードされると、プログラムを実行することができます。現在のプログラムカウンタ位置からプログラムを始めるため'実行'ボタンまたは、F5 を押してください。	

6. チュートリアルレビュー

本章では、CS+中の Tutorial コードの各セクションおよび基礎的なデバッグ機能を確かめます。

6.1 プログラム初期化

メインプログラムが実行される前にマイクロコントローラは初期化されます。Tutorial プロジェクトおよび残りのサンプルプロジェクトはデバッグ・ツールの設定により、ユーザはハードウェア初期化コードの実行工程を見ることができません。プログラムダウンロード後のエントリポイントを変更する場合はセクション 3.3 を参照してください。ハードウェアの初期化を見る場合、関数名は'_R_Systeminit'を指定してください。

チュートリアルコードの以下の部分は、主要機能が正確に実行できるように、CPU ボード上のマイクロコントローラを初期化するために使用されます。マイクロコントローラはリセットスイッチまたはパワーオンリセットによってリセットされるごとに、初期化コードが実行されます。

Tutorial コードがマイクロコントローラにダウンロードされていることを確認し、デバッグツールバーの'CPU リセット'をクリックしてください。	

- コード表示をメニューバーの'逆アセンブル'ボタン、'混合'ボタンで表示を切り替えることができます。

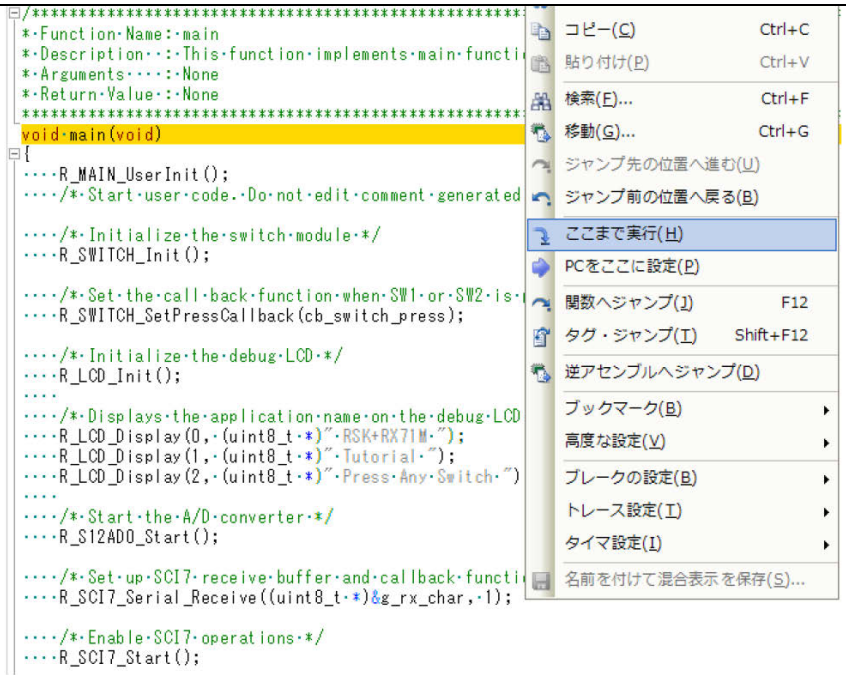
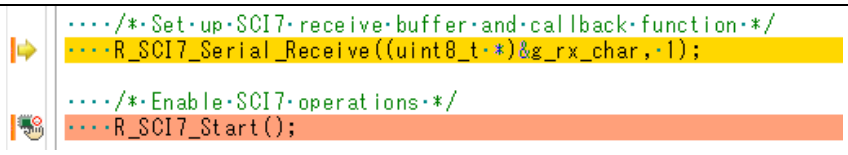
← 逆アセンブルボタン

逆アセンブル表示または混合表示から C ソース表示に戻る場合、プログラムカウンタが指しているコード行を右クリックして、'ソースへジャンプ'をクリックしてください。

```
89:          void main(void)
             _main:
  ffc007c9          7ea6            PUSH.L          R6
  ffc007cb          6040            SUB             #4H,R0
  ffc007cd          fb6204040000    MOV.L           #00000404H,R6
91:              R_MAIN_UserInit();
  ffc007d3          39bd00          BSR.W           _R_MAIN_UserInit
95:              R_SWITCH_Init();
  ffc007d6          05be1200        BSR.A           _R_SWITCH_Init
98:              R_SWITCH_SetPressCallback(cb_switch_press);
  ffc007da          fb129108c0ff    MOV.L           #-003FF76FH,R1
  ffc007e0          05ba1200        BSR.A           _R_SWITCH_SetPressCallback
101:             R_LCD_Init();
  ffc007e4          05d90d00        BSR.A           _R_LCD_Init
104:             R_LCD_Display(0, (uint8_t *)" RSK+RX71M ");
  ffc007e8          fb221c07c0ff    MOV.L           #-003FF8E4H,R2
  ffc007ee          6601            MOV.L           #0H,R1
  ffc007f0          05e00d00        BSR.A           _R_LCD_Display
105:             R_LCD_Display(1, (uint8_t *)" Tutorial ");
  ffc007f4          fb222807c0ff    MOV.L           #-003FF8D8H,R2
  ffc007fa          6611            MOV.L           #1H,R1
  ffc007fc          05d40d00        BSR.A           _R_LCD_Display
106:             R_LCD_Display(2, (uint8_t *)" Press Any Switch ");
  ffc00800          fb223307c0ff    MOV.L           #-003FF8CDH,R2
  ffc00806          6621            MOV.L           #2H,R1
  ffc00808          05c80d00        BSR.A           _R_LCD_Display
```

6.2 メイン関数

このセクションでは、メイン関数がコールされたプログラムコードがどのように動作するかを見ます。

• 'R_SCI7_Serial_Receive' 関数を右クリックして、'ここまで実行' を選択してください。'R_LCD_Init'関数は LCD を初期化し、'R_LCD_Display' 関数はストリングデータを LCD に以下のように表示します。 1 行目：RSK+RX71M 2 行目：Tutorial 3 行目：Press Any Switch	
• 'R_SCI7_Start' 関数にソフトウェア・ブレークを設定してください（行数の左側にあるブレークポイント行）。	
• 'ステップ・イン'ボタンをクリックまたは'F11'ボタンを押して'R_SCI7_Serial_Receive' 関数にエントリします。	

- プログラムカウンタは‘R_SCI7_Serial_Receive’関数に移動します。この関数は、コード生成によって生成され、受信バイトカウンタ、受信バイト数、バッファアドレスを設定します。そして、SCI7 割り込み処理内で設定した受信バイト数に達したとき、コールバック関数に呼び出されます。
- コード生成を使用したプロジェクトにつきましては、コード生成支援ツールチュートリアルマニュアルを参照ください。
- ‘実行’ボタンをクリックしてプログラム実行を再開してください。

```
MD_STATUS R_SCI7_Serial_Receive(uint8_t * const rx_buf, uint16_t rx_num)
{
    MD_STATUS status = MD_OK;

    if (1U > rx_num)
    {
        status = MD_ARGERROR;
    }
    else
    {
        g_sci7_rx_count = 0U;
        g_sci7_rx_length = rx_num;
        gp_sci7_rx_address = rx_buf;
        SCI7.SCR.BIT.RIE = 1U;
        SCI7.SCR.BIT.RE = 1U;
    }

    return (status);
}
```

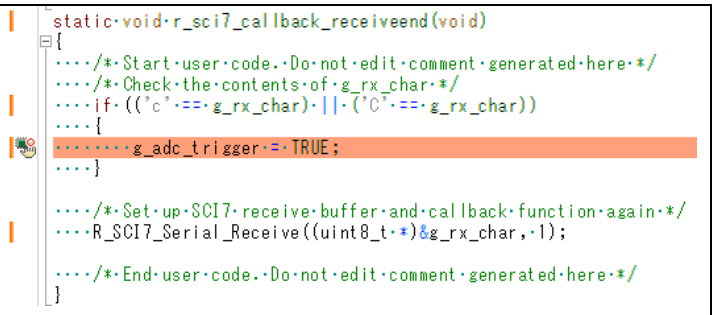
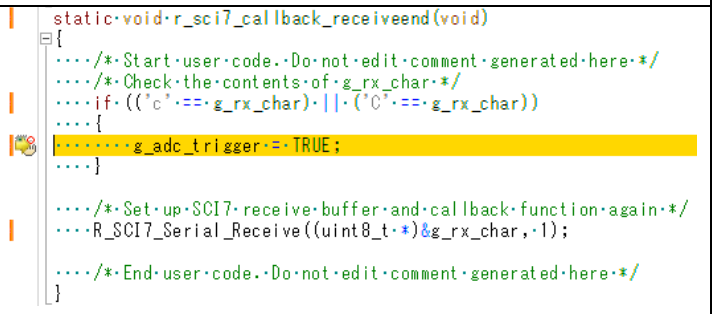
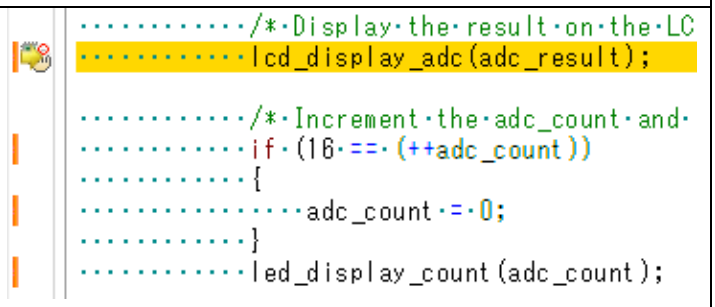
- プログラムカウンタは‘R_SCI7_Start’関数で停止します。
- ‘ステップ・オーバー’ボタンをクリックまたは’F10’ボタンを押してください。

‘R_SCI7_Start’関数は、SCI の開始、割り込みを許可します。
プログラムは、CPU ボードのスイッチ入力、または SCI 割り込みが発生するまで while ループを実行します。その後、スイッチ入力か SCI 割り込み発生で、A/D 変換を実行します。

```
    /* Set up SCI7 receive buffer and callback function */
    R_SCI7_Serial_Receive((uint8_t *)&g_rx_char, 1);

    /* Enable SCI7 operations */
    R_SCI7_Start();

    while (1U)
    {
```

- while ループ内の‘lcd_display_adc’関数を呼び出します。
- ‘lcd_display_adc’処理内の最初の処理上で右クリックし、ハードウェア・ブレークを設定してください。
- プロジェクトツリーにある、‘r_cg_sci_user.c’ファイルを開いてください。‘r_sci7_callback_receiveend’関数までスクロールしてください。

```
    while (1U)
    {
        uint16_t adc_result;

        /* Wait for user requested A/D conversion flag to be set */
        if (TRUE == g_adc_trigger)
        {
            /* Call the function to perform an A/D conversion */
            adc_result = get_adc();

            /* Display the result on the LCD */
            lcd_display_adc(adc_result);
```

• ‘r_sci7_callback_receiveend’関数で、画像の箇所にハードウェア・ブレークを設定してください。 • ‘実行’ボタンをクリックまたは‘F5’ボタンを押してプログラムを実行してください。 	
• ターミナルソフト画面で、キーボードの‘c’キーを押してください。 • プログラムは‘r_sci7_callback_receiveend’関数のハードウェア・ブレークポイントで停止します。 • ハードウェア・ブレークのアイコンをクリックしてブレークポイントを削除してください。 • ‘実行’ボタンをクリックまたは‘F5’ボタンを押してプログラムを実行してください。 	
• プログラムは‘main’関数の while ループ内のハードウェア・ブレークポイントで停止します。 • ハードウェア・ブレークのアイコンをクリックしてブレークポイントを削除してください。 • ‘実行’ボタンをクリックまたは‘F5’ボタンを押してプログラムを実行してください。 	

プログラムは、CPU ボード上のユーザスイッチを押すことで A/D 変換を実行します。そして、ポテンショメータでコントロールされた電圧値の A/D 変換結果を LCD およびターミナル画面に表示します。さらに、CPU ボード上のユーザ LED で A/D 変換回数をバイナリ形式で点灯表示します。

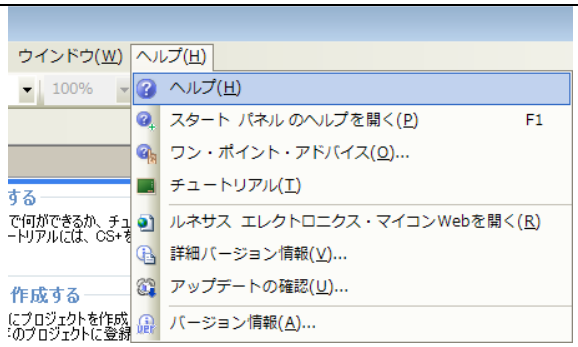
• ‘停止’ボタンをクリックし、プログラム実行を停止してください。	

ハードウェアに関する詳細は、RSK+RX71M ユーザーズマニュアルおよび RX71M グループ ユーザーズマニュアルハードウェア編を参照してください。

E1 エミュレータは本マニュアルでは説明していない高度な機能を持っています。E1 エミュレータの詳細情報は、E1/E20 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。

7. 追加情報

サポート

CS+の使用方法等の詳細情報は、CS+のヘルプメニューを参照してください。	

RSK+RX71M で提供されるサンプルコードの一部はコード生成プラグインを使用しております。コード生成によって生成されたソース・ファイルには「“R_”や“r_”」がプリフィックスされています。

RX71M グループマイクロコントローラに関する詳細情報は、RX71M グループユーザーズマニュアルハードウェア編を参照してください。
アセンブリ言語に関する詳細情報は、RX ファミリユーザーズマニュアルソフトウェア編を参照してください。

オンラインの技術サポート、情報等は以下のウェブサイトより入手可能です：
http://japan.renesas.com/rskrx71m （日本サイト）
http://www.renesas.com/rskrx71m （グローバルサイト）

オンライン技術サポート
技術関連の問合せは、以下を通じてお願いいたします。
日本： csc@renesas.com
グローバル： csc@renesas.com

ルネサスのマイクロコントローラに関する総合情報は、以下のウェブサイトより入手可能です：
http://japan.renesas.com/ （日本サイト）
http://www.renesas.com/ （グローバルサイト）

商標
本書で使用する商標名または製品名は、各々の企業、組織の商標または登録商標です。

著作権
本書の内容の一部または全てを予告無しに変更することがあります。
本書の著作権はルネサス エレクトロニクス株式会社にあります。ルネサス エレクトロニクス株式会社の書面での承諾無しに、本書の一部または全てを複製することを禁じます。

© 2015 Renesas Electronics Europe Limited. All rights reserved.
© 2015 Renesas Electronics Corporation. All rights reserved.
© 2015 Renesas System Design Co., Ltd. All rights reserved.

改訂記録	RSK+RX71M チュートリアルマニュアル(CS+)

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2015.01.23	ー	初版発行

RSK+RX71M チュートリアルマニュアル(CS+)

発行年月日　2015 年 01 月 23 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

RX71M グループ

R20UT3218JG0100

ルネサス エレクトロニクス株式会社