User's Manual

32

RX671 グループ

Renesas Starter Kit+ for RX671
チュートリアルマニュアル
(e^2 studio)

ルネサス 32 ビット マイクロコントローラ
RX ファミリ／RX600 シリーズ

ルネサス エレクトロニクス
www.renesas.com

Rev. 1.00　2021.05

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。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、お客様の責任において、お客様の機器・システムを設計ください。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. 当社製品または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. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 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。
6. 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。
7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因しまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 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。
9. 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。
10. 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。
11. 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。
12. 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。
13. 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。
14. 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.5.0-1　2020.10)

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。

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、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策
   CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。
2. 電源投入時の処置
   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. 電源オフ時における入力信号
   当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。
4. 未使用端子の処理
   未使用端子は、「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。
5. クロックについて
   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。
6. 入力端子の印加波形
   入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、V_{IL}（Max.）から V_{IH}（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、V_{IL}（Max.）から V_{IH}（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。
7. リザーブアドレス（予約領域）のアクセス禁止
   リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）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。
8. 製品間の相違について
   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、フラッシュメモリ、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、ノイズ幅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e^2 studio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、e^2 studioプロジェクトのロードおよびデバッグするために段階的な手法で構成されていますが、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。
RX671 マイクロコントローラの操作に関する詳細は、「RX671 グループ ユーザーズマニュアル ハードウェア編」およびサンプルコード内に記載されています。また、RSK+ Web インストーラのセットアップ手順については「クイックスタートガイド」に記載されています。

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。

本マニュアル中のスクリーンショットと実際に表示される画面が一部異なる場合があります。読み進めるにあたって問題はありません。

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+ RX671 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	Renesas Starter Kit+ for RX671 ユーザーズマニュアル	R20UT4879JG
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	Renesas Starter Kit+ for RX671 チュートリアルマニュアル	R20UT4883JG (本マニュアル)
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	Renesas Starter Kit+ for RX671 クイックスタートガイド	R20UT4884JG
スマート・コンフィグレータ チュートリアルマニュアル	スマート・コンフィグレータの使用方法の説明	Renesas Starter Kit+ for RX671 スマート・コンフィグレータ チュートリアルマニュアル	R20UT4885JG
回路図	CPU ボードの回路図	Renesas Starter Kit+ for RX671 CPU ボード回路図	R20UT4878EG
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	RX671 グループ　ユーザーズマニュアル　ハードウェア編	R01UH0899JJ

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
E1 / E2 Lite	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
PLL	Phase-locked Loop	位相同期回路
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。
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	Real Time Clock	リアルタイムクロック
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
TFT	Thin Film Transistor	薄膜トランジスタ
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

目次

1. 概要 7
1.1 目的 7
1.2 特徴 7

2. はじめに 8
2.1 スマート・コンフィグレータについて 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 デバッガの接続 9
3.3 e² studio の起動とサンプルコードのインポート 9
3.4 ビルド構成とデバッグセッション 11
3.4.1 ビルドコンフィグレーション 11
3.4.2 デバッグ設定 12
3.5 コードの実行 13

4. チュートリアルレビュー 14
4.1 プログラム初期化 14
4.2 メイン関数 16

5. 追加情報 19

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
- 本 RSK+提供のサンプルアプリケーション

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？

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。詳細については、「クイックスタートガイド」を参照してください。
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。また、本マニュアル中のソースコード画面のアドレスが、コンパイルしたユーザコードのものと異なる場合があります。 これら本マニュアルで記載されている内容と機能的な違いはございません。

チュートリアルは RSK+の使用方法を説明するためのものであり、e^2 studio、コンパイラまたは E2 エミュレータ Lite 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ユーザーマニュアルを参照してください。

2.1 スマート・コンフィグレータについて

本製品で提供しているサンプルコードの一部はスマート・コンフィグレータを使用してコードを生成しています。スマート・コンフィグレータは 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連携ツールです。スマート・コンフィグレータは直感的な 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でき、開発工数の大幅な削減が可能です。

スマート・コンフィグレータを使用して、選択された特定の周辺U機能ごとに3つのコードを生成します。これらのコードモジュールは、名前 'Config_xxx.h'、 'Config_xxx.c'、および 'Config_xxx_user.c'です。ここで、'xxx'は、周辺機能の頭字語です（例： 'CMT'）。
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 */
```

スマート・コンフィグレータの 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スマート・コンフィグレータ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。

RSK+サンプルプロジェクトでは、一部の関数のみが使用されています。
その他の便利な機能については、以下の URL を参照してください。
https://www.renesas.com/smart-configurator

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

e² studio はオープンソースの統合開発ツールであり、多くのルネサス製デバイスでソフトウェア製品の作成、コンパイル、プログラムおよびデバッグが可能です。

3.2 デバッガの接続

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。

E2 Lite 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、E2 Lite 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。

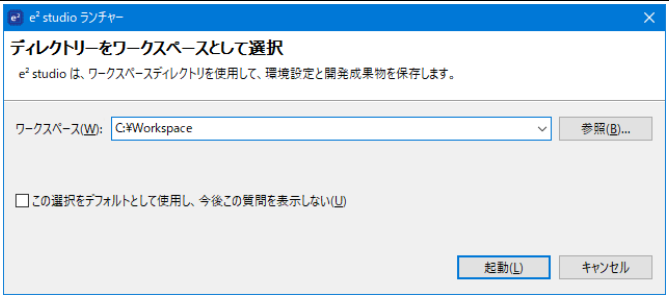
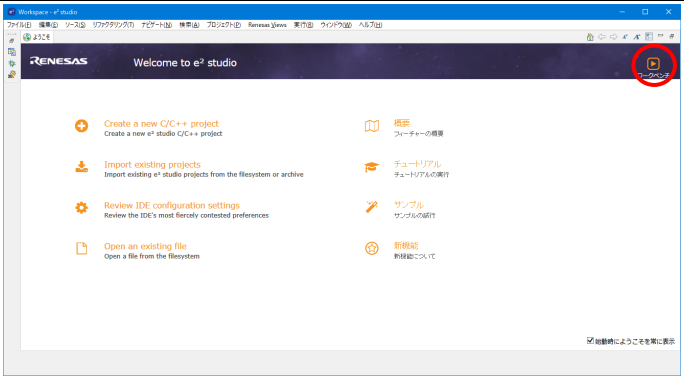
- Pmod LCD を CPU ボードの PMOD1 コネクタに取り付け、コネクタの全てのピンが正しくソケットに収まっていることを確認してください。
- E2 Lite をご使用のコンピュータの USB ポートに接続してください。
- E2 Lite を CPU ボードに接続します。‘E2 Lite’のシルク印字のある E2 Lite コネクタに接続してください。
- 外部電源を CPU ボードに供給します。’PWR’のシルク印字のある PWR コネクタに接続してください。

3.3 e² studio の起動とサンプルコードのインポート

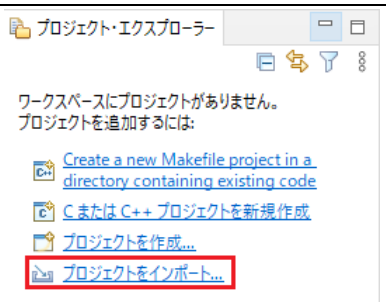
e² studio の起動:

Windows™ 8.1: ⓦをクリックして[アプリ]ビューを表示 > Renesas Electronics e2studio > e2 studio

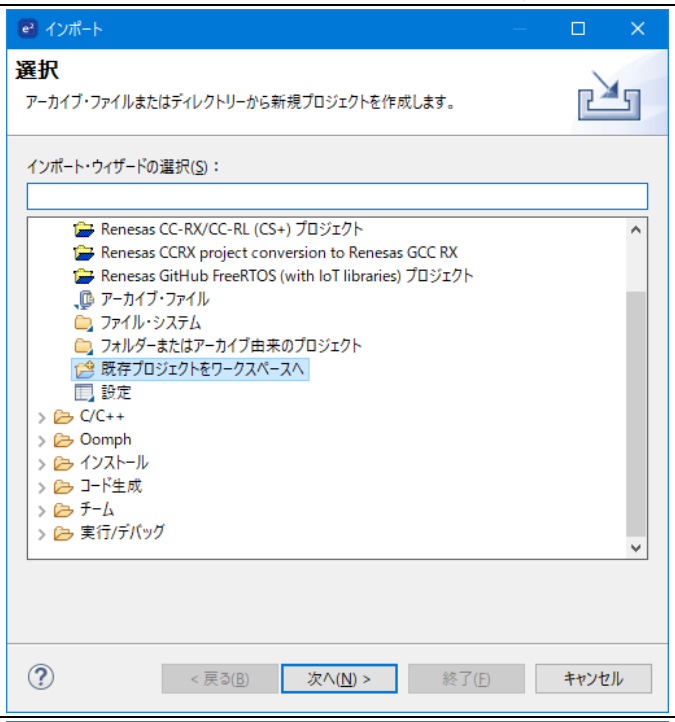
Windows™ 10: スタートメニュー > すべてのアプリ > Renesas Electronics e2studio > e2 studio

• Windows™ のスタートメニューから e² studio を選択して起動します。最初にワークスペースランチャーのダイアログボックスが表示されます。 • 必要に応じて[新しいフォルダを作成]オプションを使用して、[参照]をクリックし、ワークスペースを格納する適切な場所を選択、 <起動(L)>をクリックします。	
• Welcome to e² studio 画面が表示されます。 右端にある「ワークベンチ」の矢印ボタンをクリックします（スクリーンショットの円囲み部分）。	

- 環境が初期化されたら、’プロジェクト・エクスプローラー’ウィンドウから[プロジェクトをインポート...]を選択します。



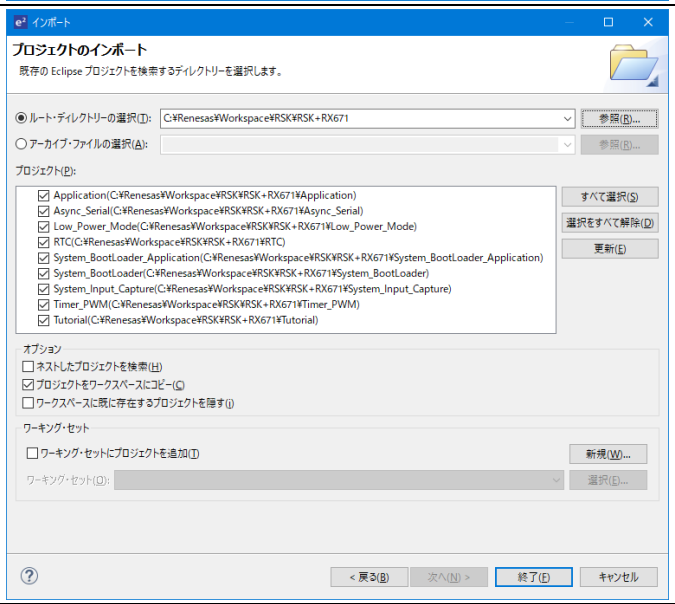
- [インポート]ダイアログボックスが表示されます。’一般’フォルダアイコンを展開し、’既存プロジェクトをワークスペースへ’を選択して, ‘次へ’ボタンをクリックします。



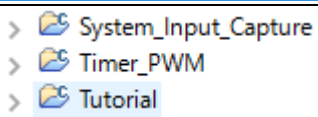
- インポートダイアログボックスでは、インポートするプロジェクトを指定できます。‘参照’ボタンをクリックし、以下のディレクトリを検索します。

  C:\Renesas\Workspace\RSK\RSK+RX671

- [プロジェクトをワークスペースにコピー]オプションが選択されていることを確認し、[終了]をクリックします。



- 左側にある「プロジェクト・エクスプローラー」のプロジェクトリストから「Tutorial」をクリックします。



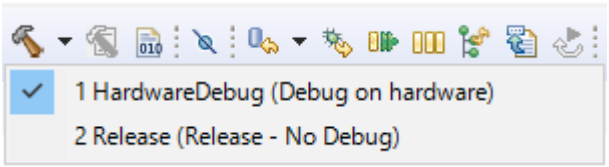
3.4 ビルド構成とデバッグセッション

3.4.1 ビルドコンフィグレーション

e² studio のワークスペースは、 'HardwareDebug'と 'Release'という 2 つのビルド構成で作成されます。

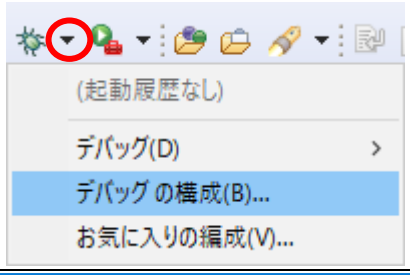
Release
このビルドモードでは、製品リリース用に適したコードを構築します。
最適化レベル 2 とし、デバッグ情報を出力しないよう設定されています。 コンパイラによるコードの最適化のため、コードが不規則に実行される場合があります。

HardwareDebug
このビルドモードでは、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 0 とし、デバッグ情報を出力するよう設定されています。 コンパライラによるコードの最適化を行わないため、コードが規則的に実行されます。

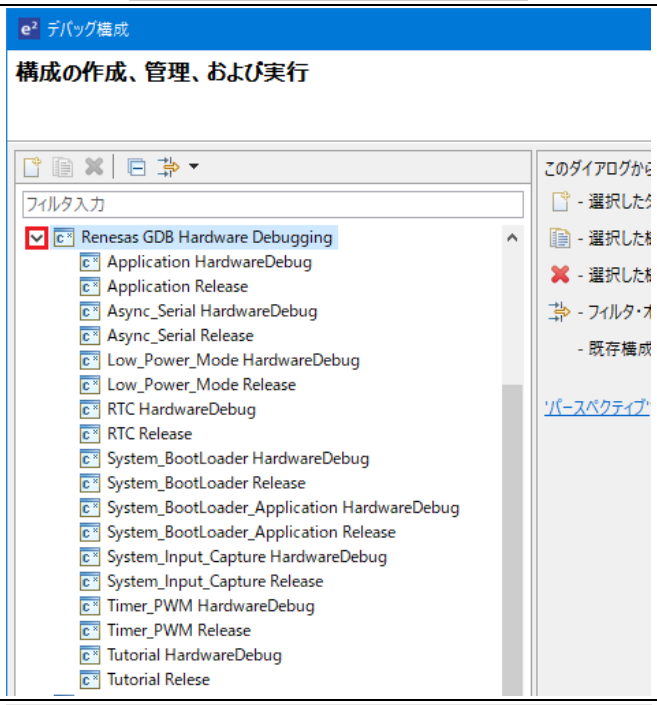
• トップの「Tutorial」フォルダをもう一度クリックし、ビルドボタン（ハンマーアイコン）  の横にある矢印をクリックし、「HardwareDebug」オプションを選択します。 • e² studio がビルドを開始します。	

3.4.2 デバッグ設定

- •デバッグボタン（バグアイコン）の横にある矢印（赤囲み部分）をクリックし、 [デバッグ構成]を選択します。



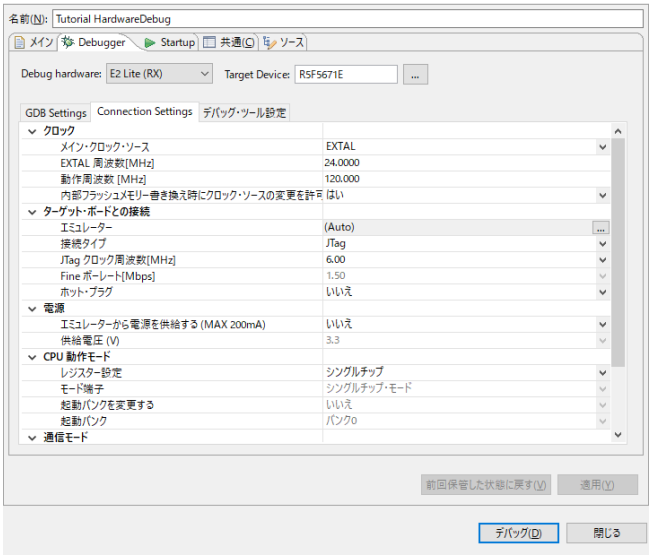
- [デバッグ構成]ダイアログボックスが表示されます。 「Renesas GDB Hardware Debugging」オプションの横にある小さな矢印（赤囲み部分）をクリックします。
- 各プロジェクトのデバッグ設定が表示されたら 'Tutorial HardwareDebug'のエントリを選択します。



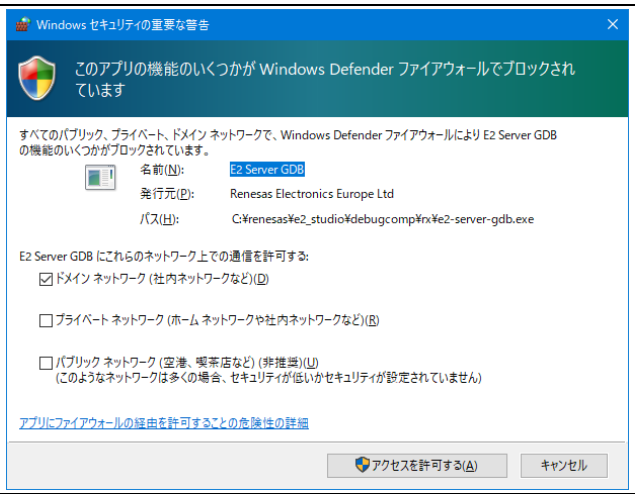
- Tutorial プロジェクトのデバッグ構成ページが表示されます。 メインタブを「Debugger」に変更して表示されるセカンダリタブを「Connection Settings」を選択します。
- Tutorial プロジェクトは事前に設定されているので、デバッガの設定を変更する必要はありません。「エミュレータから電源を供給する (MAX 200mA)」が「いいえ」となっていることを確認してください。
- 電源設定の詳細については、RSK+RX671 ユーザーズマニュアルを参照してください。

注) 誤った電源設定で接続した場合、e² studioは警告表示を行います

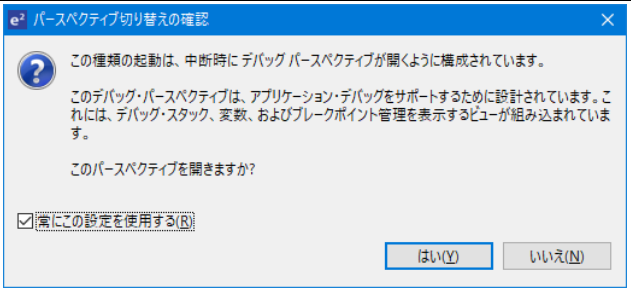
- デバッグボタンをクリックして続行します。e² studio がデバッガに接続し、コードをターゲットにダウンロードします。



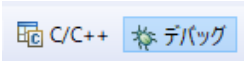
- 'e2-server-gdb.exe'のファイアウォール警告が表示されることがあります。 [自宅や職場のネットワークなどのプライベートネットワーク]チェックボックスをオンにし、[アクセスを許可する]をクリックします。
- ユーザーアカウント制御ダイアログが表示されることがあります。 管理者パスワードを入力し、「はい」をクリックします。



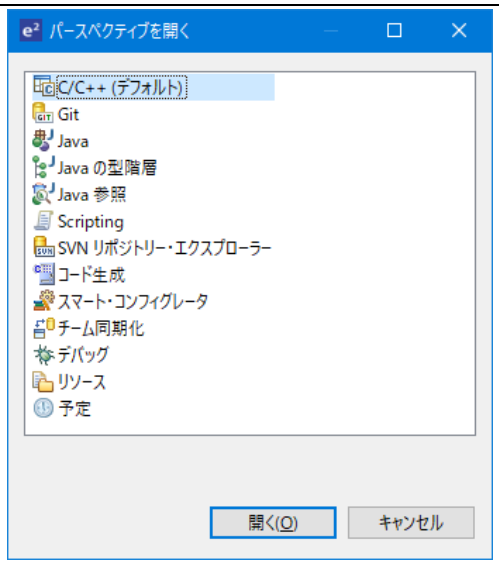
- コードをダウンロードした後に、「デバッグパースペクティブ」に切り替えを確認するダイアログボックスが表示されます。 このダイアログボックスが今後表示されないようにするには、[常にこの設定を使用する]をクリックし、[はい]をクリックします。
- e² studio はデバッグ用に最適化された新しいパースペクティブをロードします。



- デフォルトの 'C / C ++'パースペクティブに戻すには、メニューバーから[ウィンドウ]> [パースペクティブ]> [パースペクティブを開く]> [その他]を選択するか、パースペクティブを開くボタン  をクリックします。
- 「パースペクティブを開く」ダイアログボックスが表示されます。 目的のパースペクティブをクリックして選択し、<開く>をクリックします。
- パースペクティブは以下のショートカットでも切り替えることができます。



- パースペクティブは次の章のために「デバッグ」にします。



3.5 コードの実行

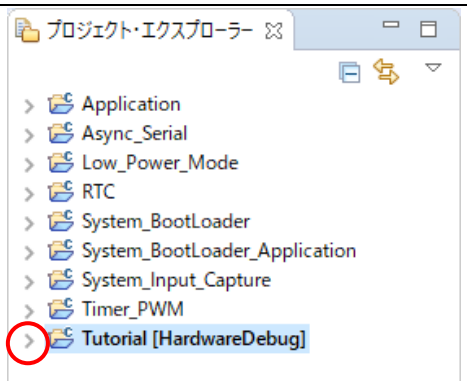
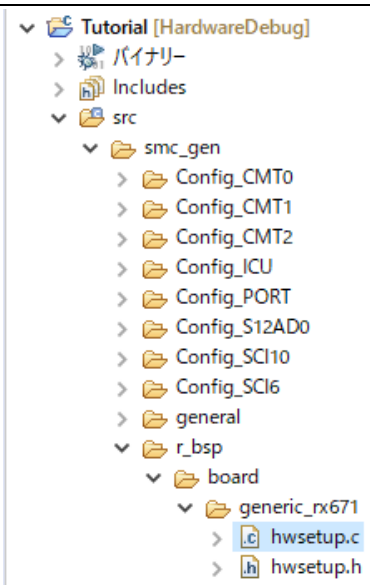
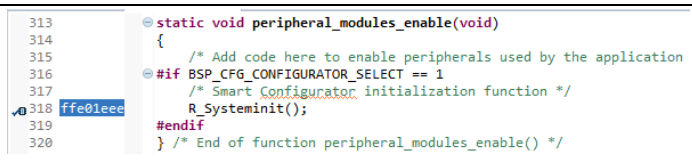
- CPU ボードの設定およびサンプルコードを実行する方法については、Tutorial プロジェクトの doc フォルダの description.txt ファイルを参照してください。
- コードがダウンロードされたら、'再開'ボタン  をクリックしてコードをメイン関数まで実行します。 main 関数は、プログラムのエントリポイントとして初期設定されます。 プログラムカウンタは、メイン関数の最初の命令で停止します。
- 「デバッグ」パースペクティブの'再開'ボタン  をクリックして、残りのコードを実行します。
- チュートリアルのデモを最初に実行してから、デバッグを続行することをお勧めします。

4.チュートリアルレビュー

この章では、Tutorial コードの各セクションと e² studio の基礎的なデバッグ機能について説明します。

4.1 プログラム初期化

メインプログラムを実行する前に、マイクロコントローラを設定する必要があります。Tutorial 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。

• 3.4.章に示すようにコードをビルドしてダウンロードします。 • プロジェクト・エクスプローラータブで、赤い円で強調表示されているフォルダアイコンの横にある矢印をクリックして、「Tutorial」フォルダを展開します。	
• 「src」フォルダの横にある矢印をクリックすると、ソースファイルが表示されます。 • 同じ方法で 'smc_gen'-> 'r_bsp'->'board' ->'generic_rx671'フォルダの順に展開し、'hwsetup.c'をダブルクリックしてファイルを開きます。	
• ブレークポイントは、ソースウィンドウの左端をダブルクリックすることで設定できます。 命令 'R_Systeminit（）;'が表示されている行で、行番号の左をダブルクリックしてブレークポイントを設定します。	

- デバッグパースペクティブの’再開’ボタンをクリックする(または[F8]を押す)と、このブレークポイントまでコードを実行します。

注）プログラムカウンタは、ブレークポイントの横に青い矢印で示されます。

```
313          static void peripheral_modules_enable(void)
314          {
315              /* Add code here to enable peripherals used by the application */
316          #if BSP_CFG_CONFIGURATOR_SELECT == 1
317              /* Smart Configurator initialization function */
318  ffe01eee    R_Systeminit();
319          #endif
320          } /* End of function peripheral_modules_enable() */
```

- ‘ステップイン’ボタンをクリックするか（または[F5]を押す）と、’R_Systeminit’関数を実行します。

- 'R_Systeminit'関数は、MCU を通常動作用に設定するいくつかの初期化関数を呼び出します。 これには入出力ポートとシステムクロックも含まれます。
- ユーザは、「ステップイン」アイコンをクリックすることで、初期化コードをステップ実行できますが、このマニュアルではスキップします。
- メイン機能までコードを実行するには、’再開’ボタンをクリックします。

```
 78 ffe01df0  void R_Systeminit(void)
 79           {
 80               /* Enable writing to registers related to operating modes, LPC, CGC
 81 ffe01df5       SYSTEM.PRCR.WORD = 0xA50BU;
 82
 83               /* Enable writing to MPC pin function control registers */
 84 ffe01dfe       MPC.PWPR.BIT.B0WI = 0U;
 85 ffe01e05       MPC.PWPR.BIT.PFSWE = 1U;
 86
 87               /* Write 0 to the target bits in the POECR2 registers */
 88 ffe01e0c       POE3.POECR2.WORD = 0x0000U;
 89
 90               /* Initialize clocks settings */
 91 ffe01e0f       R_CGC_Create();
 92
 93               /* Set peripheral settings */
 94 ffe01e13       R_Config_PORT_Create();
 95 ffe01e17       R_Config_CMT0_Create();
 96 ffe01e1b       R_Config_CMT1_Create();
 97 ffe01e1f       R_Config_CMT2_Create();
 98 ffe01e23       R_Config_ICU_Create();
 99 ffe01e27       R_Config_SCI10_Create();
100 ffe01e2b       R_Config_SCI6_Create();
101 ffe01e2f       R_Config_S12AD0_Create();
```

ハードウェア構成の詳細については、「Renesas Starter Kit+ for RX671　ユーザーズマニュアル」および「RX671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。

4.2 メイン関数

本章では、メイン関数を使用して呼び出されたプログラムコードとその動作を見ていきます。RSK+G1CUSB0 を PC の USB ポートに接続し、Tera Term などのターミナルエミュレータに以下の設定を行います：

ボーレート：19200、データ長：8、パリティビット：なし、ストップビット：1、フロー制御：なし

RSK+仮想 COM ポートドライバのインストールについては、e² studio Tutorial プロジェクトの doc フォルダ内の 'description.txt'を参照ください。

<table>
<tr>
<td>

- 'R_Config_SCI10_Serial_Receive'関数を選択して右クリックし、[指定行まで実行]を選択すると、この行までプログラムが実行されます。 'R_LCD_Init'関数は LCD パネルを有効に設定し、 'R_LCD_Display'は "RSK+RX671 Tutorial Press Any Switch"を LCD に表示します。

</td>
<td>

```
 83                 void main (void);
 84
 86               * Function Name: main
 91 ffe0092c       void main(void)
 92                 {
 93                     /* Initialize the switch module */
 94 ffe0092e         R_SWITCH_Init();
 95
 96                     /* Set the call back function when SW1 or SW2 is pressed */
 97 ffe00932          R_SWITCH_SetPressCallback(cb_switch_press);
 98
 99                     /* Initialize the debug LCD */
100 ffe0093c          R_LCD_Init();
101
102                    /* Displays the application name on the debug LCD.
103                       Casting for use as characters. */
104 ffe00940          R_LCD_Display(0, (uint8_t *)" RSK+RX671");
105
106                    /* Casting for use as characters. */
107 ffe0094c          R_LCD_Display(1, (uint8_t *)" Tutorial ");
108
109                    /* Casting for use as characters. */
110 ffe00958          R_LCD_Display(2, (uint8_t *)" Press Any Switch ");
111
112                    /* Start the A/D converter */
113 ffe00964          R_Config_S12AD0_Start();
114
```

</td>
</tr>
<tr>
<td>

- 'R_Config_SCI10_Start'関数コール行にてブレークポイント列でダブルクリックしてブレークポイントを設定します。
- ‘ステップイン’ボタンをクリックして、’R_Config_SCI10_Serial_Receive’関数にエントリします。

</td>
<td>

```
115                    /* Set up SCI10 receive buffer and callback function */
116 ffe0096a          R_Config_SCI10_Serial_Receive((uint8_t *)&g_rx_char, 1);
117
118                    /* Enable SCI10 operations */
119 ffe00974          R_Config_SCI10_Start();
120
```

</td>
</tr>
<tr>
<td>

- プログラムカウンタは‘R_Config_SCI10_Serial_Receive’関数に移動します。この関数は、スマート・コンフィグレータによって生成され、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、SCI6 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。
- スマート・コンフィグレータを使用したプロジェクトにつきましては、スマート・コンフィグレータ チュートリアルマニュアルを参照ください。
- ‘再開’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。

</td>
<td>

```
166 ffe01a08  MD_STATUS R_Config_SCI10_Serial_Receive(uint8_t * const rx_buf, uint16_t rx_num)
167           {
168 ffe01a0e       MD_STATUS status = MD_OK;
169
170 ffe01a11       if (1U > rx_num)
171               {
172 ffe01a16           status = MD_ARGERROR;
173               }
174               else
175               {
176 ffe01a1b           g_sci10_rx_count = 0U;
177 ffe01a24           g_sci10_rx_length = rx_num;
178 ffe01a2d           gp_sci10_rx_address = rx_buf;
179 ffe01a3b           SCI10.SCR.BYTE |= 0x50U;
180               }
181
182 ffe01a42       return (status);
183 ffe01a48   }
```

</td>
</tr>
</table>

- プログラムカウンタは‘R_Config_SCI10_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[F6]ボタンを押してください。

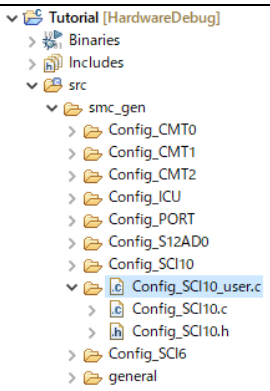
‘R_Config_SCI10_Start’関数は、SCI の開始、割り込みを許可します。
プログラムは、CPUボードのスイッチ入力、またはSCI割り込みが発生するまでwhileループを実行します。スイッチ入力かSCI割り込み発生で、A/D変換を実行します。

```
115                 /* Set up SCI10 receive buffer and callback function */
116 ffe0096a        R_Config_SCI10_Serial_Receive((uint8_t *)&g_rx_char, 1);
117
118                 /* Enable SCI10 operations */
119 ffe00974        R_Config_SCI10_Start();
```

- While ループ内の‘lcd_display_adc’関数をコールします。
- ブレークポイント列をダブルクリックして、'lcd_display_adc'関数コール行にブレークポイントを設定してください。

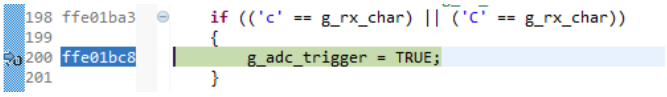
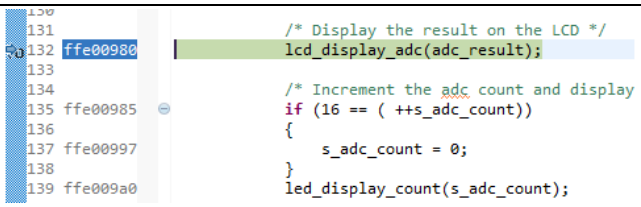
```
121                 while (1U)
122                 {
123                     uint16_t adc_result;
124
125                     /* Wait for user requested A/D conversion flag to be
126 ffe00a19          if (TRUE == g_adc_trigger)
127                     {
128                         /* Call the function to perform an A/D conversion
129 ffe0097b            adc_result = get_adc();
130
131                     /* Display the result on the LCD */
132 ffe00980            lcd_display_adc(adc_result);
```

- プロジェクト・エクスプローラーで、'Config_SCI10_user.c'ファイルを見つけてダブルクリックしてソースファイルを開きます。‘r_Config_SCI10_callback_receiveend’関数までスクロールしてください。

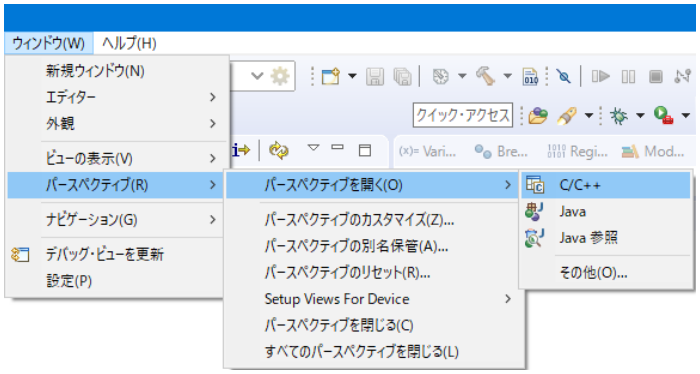


- ‘r_Config_SCI10_callback_receiveend’ 関数で、画像の箇所にブレークポイントを設定してください。
- ‘再開’ボタンをクリックして、プログラムの実行を再開してください。

```
193 ffe01ba1    static void r_Config_SCI10_callback_receiveend(void)
194             {
195                 /* Start user code for r_Config_SCI10_callback_receiveend. Do not
196
197                 /* Check the contents of g_rx_char */
198 ffe01ba3        if (('c' == g_rx_char) || ('C' == g_rx_char))
199                 {
200 ffe01bc8            g_adc_trigger = TRUE;
201                 }
202
203                 /* Set up SCI10 receive buffer and callback function again */
204 ffe01bd3        R_Config_SCI10_Serial_Receive((uint8_t *)&g_rx_char, 1);
205
206                 /* End user code. Do not edit comment generated here */
207             }
```

• ターミナルエミュレータ画面で、キーボードの'c'キーを押してください。 • プログラムは'r_Config_SCI10_callback_receiveend'関数のブレークポイントで停止します。ブレークポイント列のアイコンをダブルクリックしてブレークポイントを削除してください。 • '再開'ボタンをクリックしてプログラムの実行を再開してください。 	
• プログラムは'main'関数の while ループ内のブレークポイントで停止します。 • ブレークポイント列のアイコンをダブルクリックしてブレークポイントを削除します。 '再開'ボタンをクリックしてプログラムの実行を再開してください。  •	

プログラムは 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、実行された A/D 変換の実行カウントは、CPU ボードの LED 0～3 を使用してバイナリ形式で表示されます。 ポテンショメータを調整し、SW1 - SW3 の何れかを押すと、A/D 変換が再度実行されます。

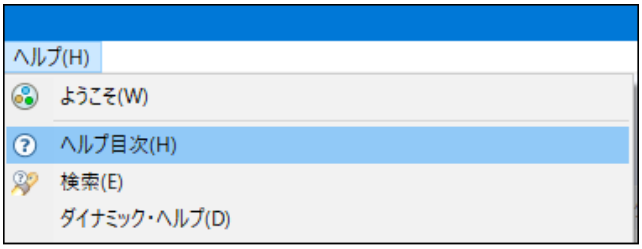
• プログラムの実行を停止するには、'中断'ボタンを押します。	
• デフォルトの 'C/C ++'パースペクティブに戻すには、メニューバーから[ウィンドウ]> [パースペクティブ]> [パースペクティブを開く]> [C/C++]を選択します。	
• または、画面の右上にある'C/C++'ボタンをクリックしてください。	

ハードウェアに関する詳細については、「Renesas Starter Kit+ for RX671 ユーザーズマニュアル」および、「RX671 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。

E2エミュレータLite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。E2エミュレータLiteの詳細情報は、E2エミュレータLiteのユーザーズマニュアルを参照してください。

5. 追加情報

サポート

e² studio の使用方法の詳細については、e² studio のメニューバーから’ヘルプ(H)’-> ‘ヘルプ目次(H)’を選択して、ヘルプファイルを参照してください。	

RSK+RX671 で提供されるサンプルコードの一部はスマート・コンフィグレータプラグインを使用しています。スマート・コンフィグレータは e² studio とのプラグインとして付属しています。
スマート・コンフィグレータで生成されたソースファイル名と関数名にはそれぞれ、’r_’と’R_’または'Config_'が付加されます。

RX671 マイクロコントローラに関する詳細情報は、RX671 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 https://www.renesas.com/rskrx671 より入手できます。

オンライン技術サポート
技術関連のお問合せは、https://www.renesas.com/support/contact.html を通じてお願いいたします。

ルネサスのマイクロコントローラに関する総合情報は、https://www.renesas.com/をご利用ください。

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

改訂記録	RX671 グループ Renesas Starter Kit+ for RX671 チュートリアルマニュアル(e^2 studio)

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	May.10.21	—	初版発行

RX671 グループ
Renesas Starter Kit+ for RX671 チュートリアルマニュアル(e^2 studio)

発行年月日　2021 年 05 月 10 日　Rev.1.00

発行　　ルネサス エレクトロニクス株式会社
　　〒135-0061　東京都江東区豊洲 3-2-24（豊洲フォレシア）

RX671 グループ

ルネサス エレクトロニクス株式会社

R20UT4883JG0100