User's Manual

16

RL78/G14

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

16 ビット・シングルチップ・マイクロコントローラ
RL78 ファミリ

本資料に記載の全ての情報は本資料発行時点のものであり、ルネサス エレクトロニクスは、
予告なしに、本資料に記載した製品または仕様を変更することがあります。
ルネサス エレクトロニクスのホームページなどにより公開される最新情報をご確認ください。

ルネサス エレクトロニクス
www.renesas.com

Rev.1.00 2012.6

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、お客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。

2．　本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。

3．　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に関し、当社は、何らの責任を負うものではありません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。

4．　当社製品を改造、改変、複製等しないでください。かかる改造、改変、複製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）、交通用信号機器、
　　防災・防犯装置、各種安全装置等

当社製品は、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（原子力制御システム、軍事機器等）に使用されることを意図しておらず、使用することはできません。 たとえ、意図しない用途に当社製品を使用したことによりお客様または第三者に損害が生じても、当社は一切その責任を負いません。 なお、ご不明点がある場合は、当社営業にお問い合わせください。

6．　当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。

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、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害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。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。

9．　本資料に記載されている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。また、当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途に使用しないでください。当社製品または技術を輸出する場合は、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。

10.　お客様の転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。

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。

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

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

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 ハードウェア概要と電気的特性をユーザに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、RSK 製品の機能概観を含みますが、組み込みプログラミングまたはハードウェア設計ガイドのためのマニュアルではありません。また、RSK および開発環境のセットアップに関するその他の詳細は、チュートリアルに記載しています。

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。

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。

RSKRL78G14 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	RSKRL78G14 ユーザーズマニュアル	R20UT0785JG
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	RSKRL78G14 チュートリアルマニュアル	R20UT0786JG (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	RSKRL78G14 クイックスタートガイド	R20UT0787JG
回路図	CPU ボードの回路図	RSKRL78G14 CPU ボード回路図	R20UT0784EG
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	RL78/G14 ユーザーズマニュアル ハードウェア編	R01UH0186JJ

2.　略語および略称の説明

略語／略称	英語名	備考
ADC	Analogue 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
CD	Compact Disk	コンパクトディスク
CPU	Central Processing Unit	中央処理装置
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
E20	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
LCD	Liquid Crystal Display	液晶ディスプレイ
LED	Light Emitting Diode	発光ダイオード
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
USB	Universal Serial Bus	-

目次

1. 概要 6
1.1 目的 6
1.2 特徴 6

2. はじめに 7

3. コード生成（設計ツール） 8
3.1 コード生成ツールとは 8
3.2 機能 8
3.3 コードの生成 8

4. チュートリアルプロジェクトワークスペース 10

5. プロジェクトワークスペース 11
5.1 はじめに 11
5.2 CubeSuite+の開始とE1 エミュレータの接続 11
5.3 デバッグツールの設定 14
5.4 ビルド設定 16

6. チュートリアルプログラムのビルド 17
6.1 コードのビルド 17
6.2 エミュレータの接続 18
6.3 E1 によるターゲットの接続 18

7. チュートリアルのダウンロードと実行 20
7.1 プログラムコードのダウンロード 20
7.2 コードの実行 20

8. チュートリアルレビュー 21
8.1 プログラム初期化 21
8.2 メイン関数 22

9. 追加情報 25

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 2 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。
- ‘DebugBuild’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。
- ‘ReleaseBuild’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。

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。

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、CubeSuite+、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。

3. コード生成（設計ツール）

3.1 コード生成ツールとは

コード生成は、マイクロコントローラが提供している周辺機能を制御する際に必要な情報を CubeSuite+上で選択/入力することにより、対応するソースコードを出力するツールです。

3.2 機能

コード生成には以下の機能があります：

・レポート機能
設計ドキュメント用にコード生成を使用して、設定した情報をファイル出力できます。

・リネーム機能
ファイルに割り当てられたデフォルト名を変更することができます。

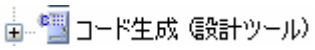
・コード生成機能
コード生成は GUI を使用して設定された情報に従ってデバイス・ドライバ・プログラムだけでなくメイン関数およびリンク指示ファイルを含むサンプル・プログラムの様なビルド環境も出力することができます。

・プロジェクト/ワークスペースファイル生成機能

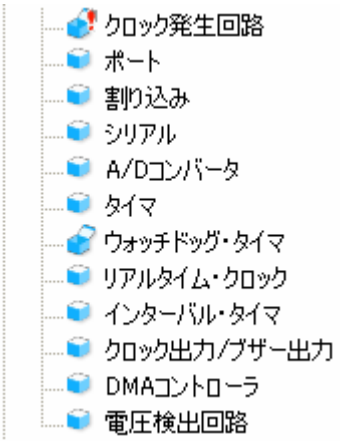
3.3 コードの生成

コード生成（設計ツール）を使用して、ソースコードを生成する簡単な手順例を示します。この手順はプロジェクトが既に存在し、プロジェクトが開かれていると仮定します。コード生成（設計ツール）の使用方法に関する詳細は、CubeSuite+ V1.02.00 RL78 設計編のユーザーズマニュアルを参照してください。

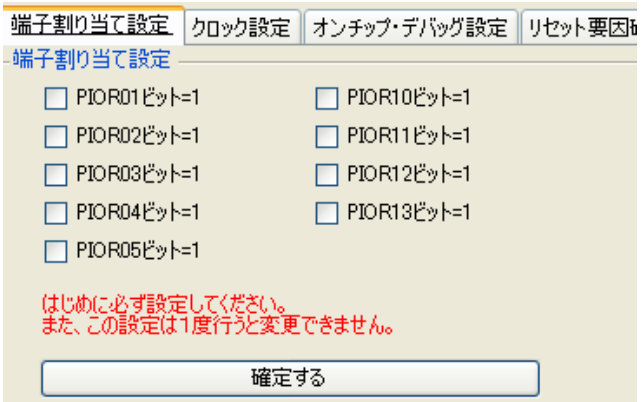
- プロジェクトツリーの‘コード生成（設計ツール）’をダブルクリックし、ツリーを展開します。



- プロジェクトツリーは選択されたデバイスの設定可能な周辺をすべて示します。
- クロック発生回路のエクスクラメーションマークは端子割り当て設定後に消えます。
- ウォッチドッグ・タイマはデフォルト許可になっています。（ツリー上の BOX が開いた状態）

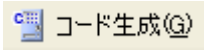


- 端子割り当て設定画面上の<確定する>をクリックします。
- ‘クロック設定’タブはクロック設定のためのオプションを含んでいます。
- ‘オンチップ・デバッグ設定’タブはデバッグの許可/禁止およびセキュリティ ID の設定を含んでいます。オンチップ・デバッグ動作設定を’使用する’にしてください。



- スクリーンショットで強調されたアイコンは設定することができる利用可能な周辺を示します。設定内容を開き、所望の周辺を設定するために各アイコンをクリックしてください。

- 所望の周辺をすべて設定した後、’コード生成’をクリックしてプロジェクトソースおよびヘッダファイルを生成します。
- 生成されたファイルはプロジェクトフォルダにコピーされます。



注：
同じ名前のファイルがプロジェクトフォルダに既に存在する場合、ファイルを上書きする前に古いファイルをバックアップすることを推奨します。

注：
Application サンプルプロジェクトは Applilet から生成されたテンポラリファイルを備えた空のサンプルプロジェクトです。アプリケーション開発に必要な API 機能を含むソースファイルを生成し、Application サンプルプロジェクトの中でそれらを使用するために Applilet を使用することを推奨します。

4. チュートリアルプロジェクトワークスペース

プロジェクトは必要なファイルをすべて含んでいます。Tutorial コードはどのようにコーディングし、デバッグすることができることを示すのを目的としてしています。

共通の C ファイルの内容はビルドコンフィグレーションのデファイン設定およびソースファイルないの#ifdef ステートメントで管理されます。プロジェクトファイルは 1 つのセットのみを取り扱うことで、管理の簡素化が図れます。

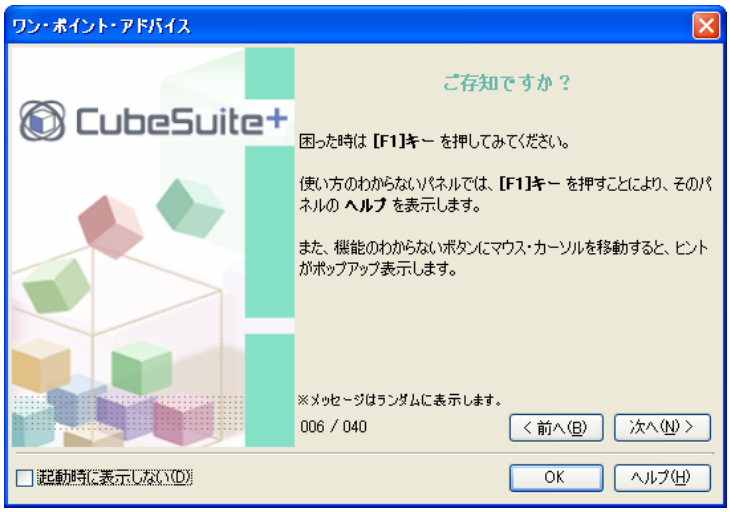
5. プロジェクトワークスペース

5.1 はじめに

CubeSuite+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。CubeSuite+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。

5.2 CubeSuite+の開始とE1 エミュレータの接続

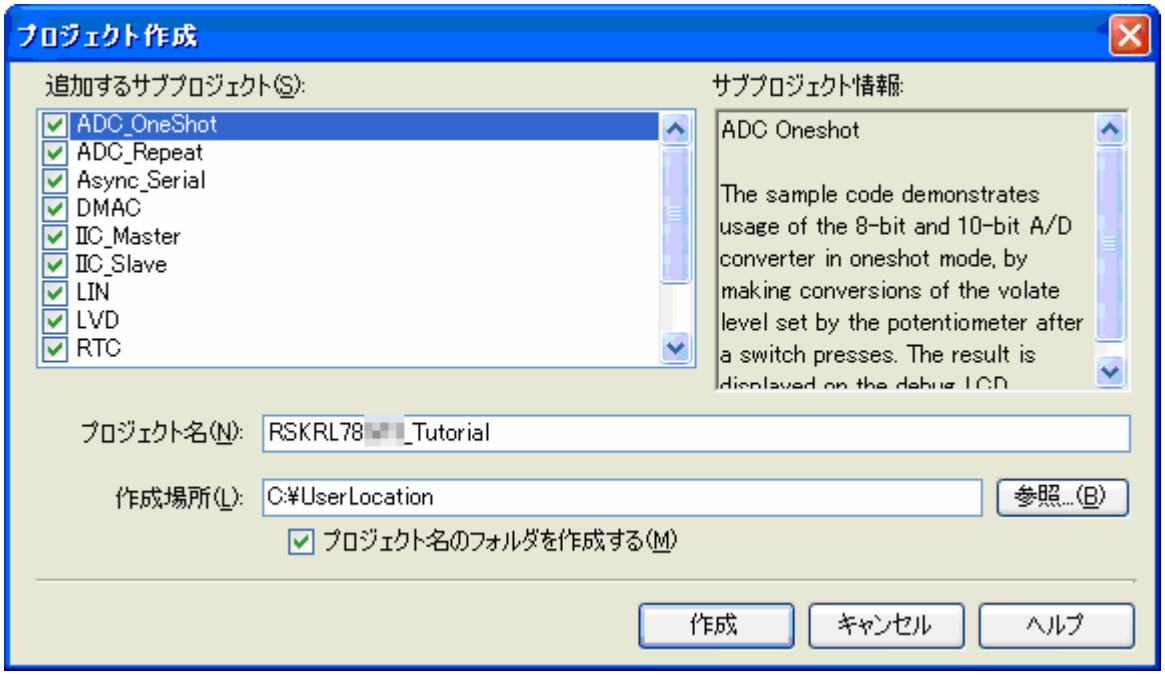
先ず、Windows のスタートメニューから CubeSuite+を起動してください。
CubeSuite+を初めて使用する場合、ワンポイントアドバイスのダイアログが表示されます。



<OK>をクリックし、ダイアログを閉じてください。その後、6 つのセクションを含む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 RSKRL78G14_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、RSKRL78G14_Tutorial プロジェクトのコピーを保存します。

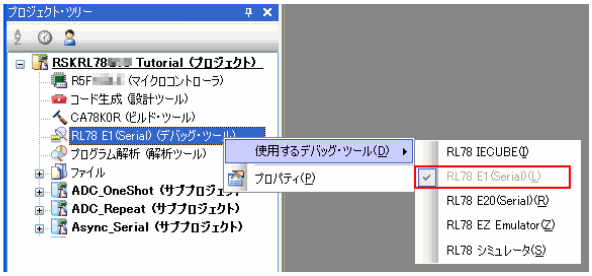
CubeSuite+はプロジェクト作成ダイアログを表示します。各サブプロジェクト名のチェックボックスをチェックし、サブプロジェクトをすべて追加してください。各サブプロジェクトの情報はダイアログ上のサブプロジェクト情報の下に表示されます。

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。

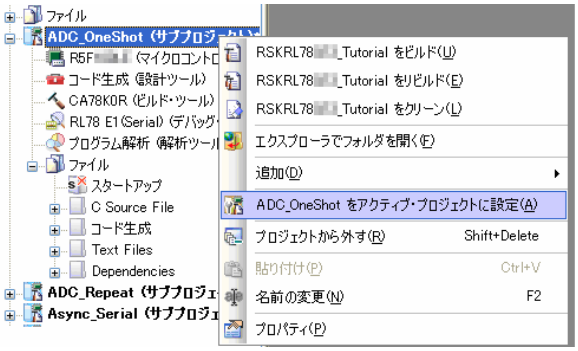


- RSKRL78G14_Tutorial はマスタープロジェクトで、サブプロジェクトとして各サンプルを含んでいます。
- スクリーンショットのファイルフォルダはマスタープロジェクト RSKRL78G14_Tutorial に属します。
- このフォルダは個別のフォルダ構造で用意されたテキストファイルを含むプロジェクトソースおよびヘッダファイルをすべて含んでおりリストします。
- ファイルフォルダの下にサブプロジェクトがリストされます。
- 各サブプロジェクトフォルダを展開すると、マスタープロジェクトと同様にツールとフォルダ構成になっています。
- 現在アクティブなプロジェクトはプロジェクト名に下線が含まれます。
- 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、”____をアクティブ・プロジェクトに設定”を選択します。

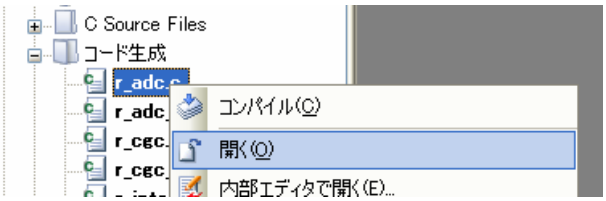
- プロジェクトはデバッグツールを選択できます。
- RL78 XXX（デバッグ・ツール）を右クリックし、RL78 E1（Serial）を選択してください。スクリーンショットは予め RL78 E1（Serial）が選択されています。



- スクリーンショットは ADC_OneShot 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。この構造はすべてのプロジェクト共通です。
- コード生成はコード生成フォルダの下でグループ化されたすべてのソースファイルを生成します。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_'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コード生成とは別のフォルダにリストされます。



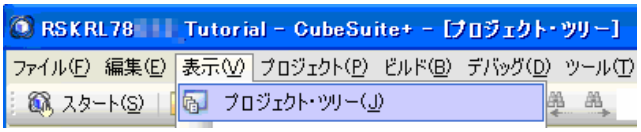
- ファイルを参照するには、参照したいファイルを右クリックし、"開く"を選択します。ファイルをダブルクリックしても参照できます。



5.3 デバッグツールの設定

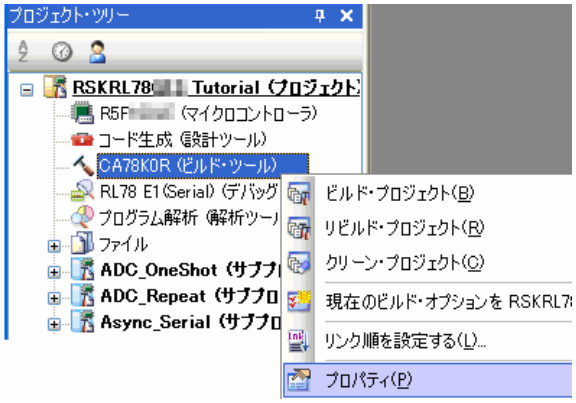
注：
Tutorial プロジェクトは予めデバッグツールの設定がされています。このセクションは新しいプロジェクトを作成するための説明です。

- プロジェクトツリーは CubeSuite+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）

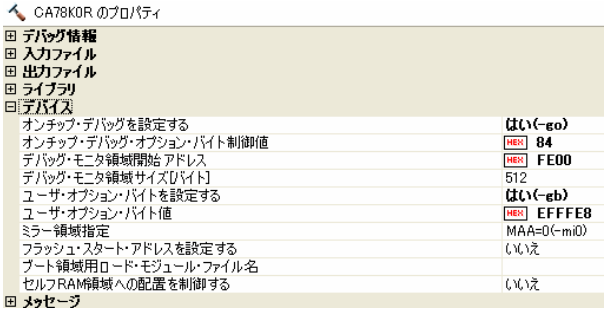


このリストはソースコードをリストするのと同様にデバイスをプログラムしデバッグするための IDE を形成するのに使用される多くのツールを含んでいます。既に設定された内容を確認するために、次の指示に従ってください：

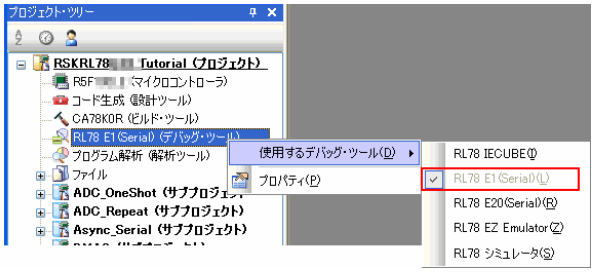
- CA78K0R（ビルド・ツール）を右クリックしてください。
- プロパティを選択してください。



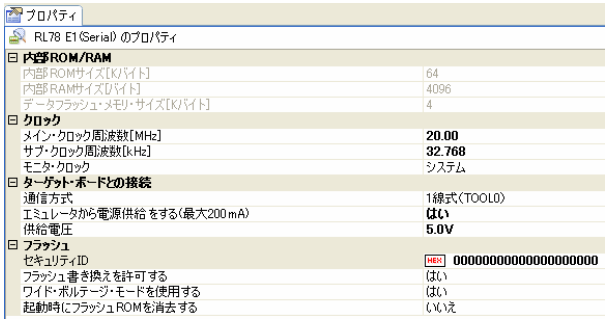
- リンク・オプションタブをクリックしてください。
- デバイスオプションを展開してください。
- デバッグ・モニタの開始アドレスがスクリーンショットと同じであることを確認してください。



- RL78 XXX（デバッグ・ツール）を右クリックし、RL78 E1（Serial）を選択してください。スクリーンショットは予め RL78 E1（Serial）が選択されています。

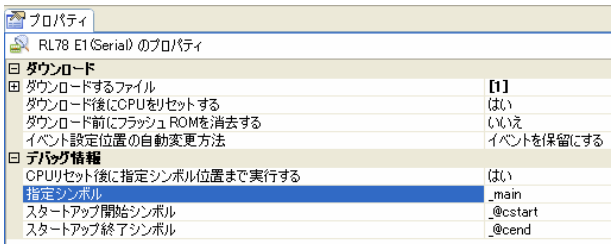


- RL78 E1（Serial）を右クリックし、プロパティを選択してください。
- 接続用設定タブをクリックしてください。
- 設定内容がスクリーンショットと同じであることを確認してください。



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。



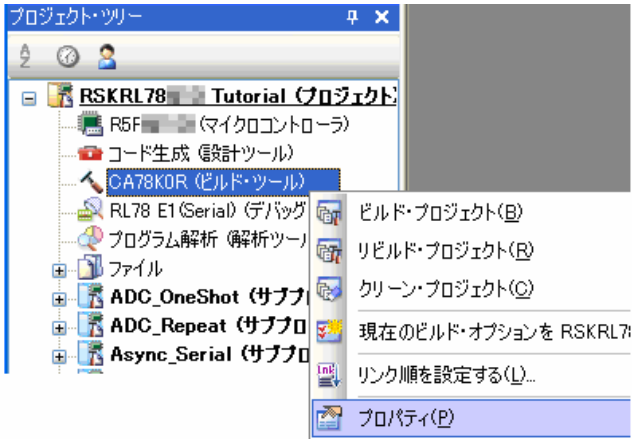
注：割り込みハンドラをエントリポイントとして指定しないでください。

5.4 ビルド設定

ビルド設定は CA78K0R（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、DebugBuild、ReleaseBuild です。DefaultBuild および DebugBuild はデバッガを備えた設定になっています。ReleaseBuild は最終の ROM 化用プログラムのために設定されます。

2 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、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。

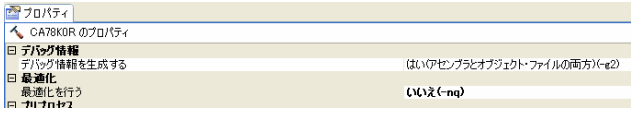
- CA78K0R（ビルド・ツール）を右クリックし、プロパティを選択してください。



- 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 DebugBuild に設定してください。スクリーンショットは DefaultBuild 例です

- コンパイル・オプションタブを選択してください。

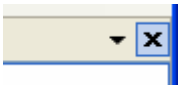
- デバッグ情報オプションが’はい(アセンブラとオブジェクト・ファイルの両方)(-g2)’に設定されていることを確認してください。
- 最適化オプションが’いいえ(-nq)’に設定されていることを確認してください。



6. 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 CA78K0R（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。



6.1 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。

- 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。

6.2 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する必要はありません。電源は USB ポートから供給されます。USB ポートに多くのデバイスが接続している場合、Windows がシャットダウンするかもしれないので注意してください。この問題が発生した場合、一部のデバイスを削除して、もう一度やり直してください。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。

LVD（電圧検出）回路サンプルコードは電圧検出のために可変電源を必要とします。その場合には、外部電源を使用する必要があります。詳細は RSKRL78G14 ユーザーズマニュアルを参照してください。

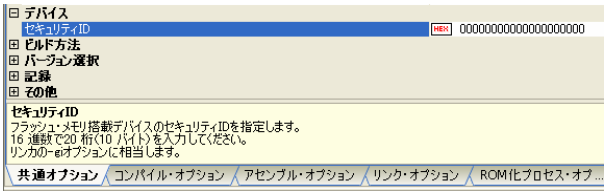
E1 のホストコンピュータへの接続方法は、クイックスタートガイドに詳しく記載されています。以下は、クイックスタートガイドの手順が踏まれ、E1 用のドライバが既にインストールされていることを前提としています。

- LCD モジュールを CPU ボードの LCD コネクタに取り付け、コネクタの全てのピンが正しく接続され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 ボードに供給する場合、セクション 5.3 を参照し、エミュレータから電源供給するオプションを解除してください。

6.3 E1 によるターゲットの接続

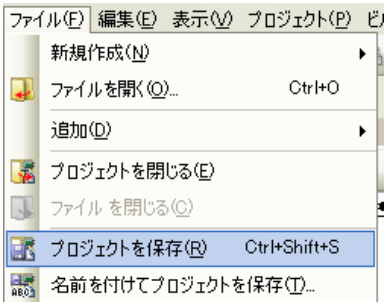
ここでは、デバイスへの接続、フラッシュへのプログラミングおよびコード実行について説明します。

- プロジェクトツリーの CA78K0R（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。
- デバイスオプションを展開して、セキュリティ ID が 00000000000000000000 に設定されていることを確認してください。



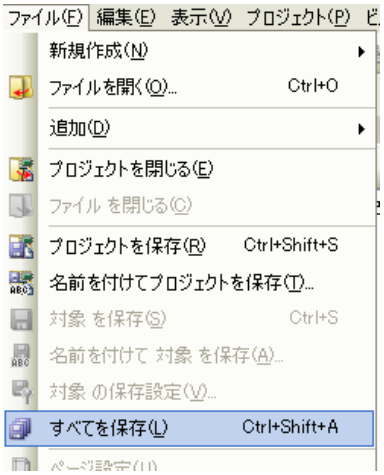
プロジェクトの設定を変更した場合、プロジェクトを保存することを推奨します。

- 'ファイル' | 'プロジェクトを保存' を選択します。



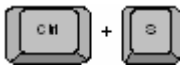
CubeSuite+中のファイルを変更した場合、次の操作で保存することができます。

- '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。



7. チュートリアルのダウンロードと実行

7.1 プログラムコードのダウンロード

CubeSuite+でコードのビルドが完了したら、それを 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。

- ダウンロードの完了後、デバッガおよび個度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。これはプログラムのエントリポイントです。

```
77 void main(void)
78 {
79         /* Start user code. Do not edit comment generated here */
80
81         /* Initialise the LCD module. */
82         Init_LCD();
83
84         /* Display RSK and sample name */
85         DisplayString(LCD_LINE1, "Renesas");
86         DisplayString(LCD_LINE2, NICKNAME);
```

7.2 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'ボタンをクリックしてください。

8. チュートリアルレビュー

本章では、Tutorial コードがどのように動作し、より複雑なコードへ実装されるためにどのようにそれを変更することができるかを確かめます。

8.1 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 5.3 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。

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。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。

```
77:         void main(void)
78:         {
79:             /* Start user code. Do not edit comment generated here */
80:
81:             /* Initialise the LCD module. */
82:             Init_LCD();
            _main:
  0070d         fc300200    CALL            !!_Init_LCD
83:
84:             /* Display information on the debug LCD. */
85:             Display_LCD(LCD_LINE1, "Renesas");
    00711       300630      MOVW            AX,#3006H
    00714       c1          PUSH            AX
    00715       f6          CLRW            AX
    00716       fcc00200    CALL            !!_Display_LCD
    0071a       c0          POP             AX
86:             Display_LCD(LCD_LINE2, NICKNAME);
    0071b       300e30      MOVW            AX,#300EH
    0071e       c1          PUSH            AX
    0071f       301000      MOVW            AX,#10H
    00722       fcc00200    CALL            !!_Display_LCD
    00726       c0          POP             AX
```

8.2 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。

- Flash_LED 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。Init_LCD 関数は LCD モジュールの初期化し、Display_LCD 関数はストリングデータ'Renesas'および'RL78G14'を LCD の 1 行目および 2 行目にライトします（表示します）。

```
void main(void)
{
    /* Start user code. Do not edit comment generated here */

    /* Initialise the LCD module. */
    Init_LCD();

    /* Display information on the debug LCD. */
    Display_LCD(LCD_LINE1, "Renesas");
    Display_LCD(LCD_LINE2, NICKNAME);

    /* Flash the user LEDs for some time or until a push button is pressed. */
    Flash_LEDs();

    /* Flash the user LEDs at a rate set by the user potentiometer (ADC) using
    interrupts. */
    Timer_ADC();
```

- Timer_ADC 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）。

```
    /* Flash the user LEDs for some time
    Flash_LEDs();

    /* Flash the user LEDs at a rate set
    interrupts. */
    Timer_ADC();
```

- '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 Flash_LEDs 関数にエントリします。

- プログラムカウンタは Flash_LEDs 関数に移動します。この関数はユーザスイッチに接続されている外部割り込みを許可し、ユーザ LED を点滅させます。LED は 200 回点滅するかユーザスイッチが押されるまで点滅を繰り返します。
- '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。
- ループを出ると直ぐに関数はスイッチ割り込みを禁止にし、メイン関数に戻ります。

```
void Flash_LEDs(void)
{
    /* Declare loop iteration count variable */
    uint32_t led_delay=0;
    uint8_t flash_count=0;

    /* Enable SW1 interrupts */
    R_INTC8_Start();

    /* Enable SW2 interrupts */
    R_INTC9_Start();

    /* Enable SW3 interrupts */
    R_INTC10_Start();

    /* Flash the LEDs for 200 times or until a user switch is
    while((0 == gSwitch_Flag) && (flash_count++ < 200))
    {
        /* LED flashing Delay */
        for(led_delay=0; led_delay<400000; led_delay++)
        {
                /* Delay */
        }

        /* Toggles the LEDs after a specific delay. */
        Toggle_LEDs();
    }
```

- プログラムカウンタは Timer_ADC 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。

Timer_ADC 関数は連続的な A/D 変換および周期タイマ（周期は A/D 変換結果によって更新されます）を開始します。

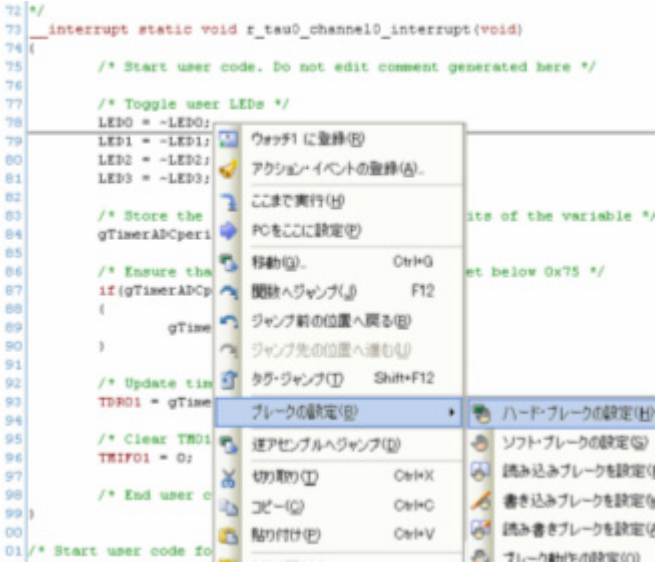
```
/* Flash the user LEDs for some time or until a
Flash_LEDs();

/* Flash the user LEDs at a rate set by the use
interrupts. */
Timer_ADC();

/* Demonstration of initialised variables. Use
debugger.*/
Statics_Test();

/* Halt program in an infinite while loop */
while(1U);
```

- ‘r_timer_user.c’ファイルを開いてください。r_tau0_channel0_interrupt 割り込みハンドラ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。
- ‘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。



- プログラムはタイマの周期経過によってハードウェア・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。

```
__interrupt static void r_tau0_channel0_interrupt(void)
{
    /* Start user code. Do not edit comment generated here */

    /* Toggle user LEDs */
    LED0 = ~LED0;
    LED1 = ~LED1;
    LED2 = ~LED2;
    LED3 = ~LED3;
```

- ‘F5’ボタンを押し、プログラム実行を再開してください。Statics_Test 関数が実行されることで、LCD の 2 行目のストリング表示が「STATIC」から「TESTTEST」に置き換わることを確認してください。
- ストリング表示が置き換わった後、2 行目の表示は初期表示「RL78G14」に戻ります。

```
126 void Statics_Test(void)
127 {
128     /* Declare loop count variable */
129     uint8_t uiCount;
130     /* Declare string variable to hold the string to be copied */
131     uint8_t ucStr[] = "STATIC \0";
132     /* Declare variable buffer to store the copied string */
133     uint8_t ucReplace[] = "TESTTEST\0";
134
135     /* Declare a delay count variable */
136     uint32_t ulDelay;
137
138     /* Write ucStr varaible, "STATIC" to LCD */
139     DisplayString(LCD_LINE2, (int8_t*)ucStr);
140
141     /* Begin for loop which writes one letter of ucReplace to the LCD at a time
142        The nested while loops generate the delay bewteen each letter change */
143     for (uiCount=0; uiCount<8; uiCount++)
144     {
145         /* Replace letter number uiCount of ucStr from ucReplace */
146         ucStr[uiCount] = ucReplace[uiCount];
147
148         /* Display the character on the debug LCD */
149         DisplayString(LCD_LINE2, (int8_t*)ucStr);
150
151         /* LED Flashing Delay */
152         for(ulDelay=0; ulDelay<700000; ulDelay++)
153         {
154                 /* Delay */
155         }
156     }
157
158     /* Clear LCD Display */
159     ucStr[uiCount] = '\0';
160
161     /* Write MCU nickname to LCD again */
162     DisplayString(LCD_LINE2, NICKNAME);
163 }
```

- ‘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、RSKRL78G14 ユーザーズマニュアルおよび RL78/G14 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。

9. 追加情報

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

RL78/G14 マイクロコントローラに関する詳細情報は、RL78/G14 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、RL78 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/rskrl78g14 （日本サイト）
http://www.renesas.com/rskrl78g14 （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。

© 2012 Renesas Electronics Europe Limited. All rights reserved.
© 2012 Renesas Electronics Corporation. All rights reserved.
© 2012 Renesas Solutions Corp. All rights reserved.

改訂記録	RSKRL78G14 チュートリアルマニュアル(CubeSuite+)

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2012.06.08	ー	初版発行

RSKRL78G14 チュートリアルマニュアル(CubeSuite+)

発行年月日　2012 年 6 月 8 日　Rev.1.00

発行　株式会社ルネサスソリューションズ
〒532-0003　大阪府大阪市淀川区宮原 4-1-6

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口

http://www.renesas.com

※営業お問合せ窓口の住所・電話番号は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル）　(03)5201-5307

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/contact/

© 2012 Renesas Electronics Corporation. All rights reserved.
Colophon 1.1

RL78/G14

R20UT0786JG0100

RENESAS

ルネサス エレクトロニクス株式会社