

RL78/L1C

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。

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・システム的设计において、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っていると、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CubeSuite+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 CubeSuite+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。

RSKRL78L1C 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	RSKRL78L1C ユーザーズマニュアル	R20UT2203JG
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グ方法の説明	RSKRL78L1C チュートリアルマニュアル	R20UT2204JG (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	RSKRL78L1C コード生成支援ツール チュートリアルマニュアル	R20UT2892JG
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	RSKRL78L1C クイックスタートガイド	R20UT2205JG
回路図	CPU ボードの回路図	RSKRL78L1C CPU ボード回路図	R20UT2202EG
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/L1C ユーザーズマニュアル ハードウェア編	R01UH0409JJ

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
CPU	Central Processing Unit	中央処理装置
DVD	Digital Versatile Disc	デ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グングエミュレータ
E20	Renesas On-chip Debugging Emulator	ルネサスオンチップデバッグングエミュレータ
GUI	Graphical User Interface	グラフィカルユーザインタフェース
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
SAU	Serial Array Unit	シリアルアレイユニット
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	-

目次

1. 概要	7
1.1 目的	7
1.2 特徴	7
2. はじめに	8
2.1 Application Leading Tool	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 CubeSuite+の開始と E1 エミュレータの接続	9
3.3 デバッグ・ツールの設定	12
3.4 ビルド設定	14
4. チュートリアルプログラムのビルド	15
4.1 コードのビルド	15
4.2 エミュレータの接続	16
4.3 E1によるターゲットの接続	16
5. チュートリアルのダウンロードと実行	18
5.1 プログラムコードのダウンロード	18
5.2 コードの実行	18
6. チュートリアルレビュー	19
6.1 プログラム初期化	19
6.2 メイン関数	20
7. 追加情報	23

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グ
- スイッチ、LED、ポテンシオ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' はデバッグ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' はデバッグ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。

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。

2.1 Application Leading Tool

本製品で提供しているサンプルコードの一部は、Application Leading Tool (以下、Applilet と称す) を使用してコードを生成しています。Applilet は C ソースコード生成とマイクロコントローラの生成のための GUI ツールです。Applilet 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。

Applilet 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 */
```

Applilet 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 Applilet 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。

Applilet をインストールする場合、本製品に同梱されている DVD の ApplicationLeadingTool フォルダ内の Application_Leading_Tool_for_RL78_V10100.exe を実行し、インストールウィザードの指示に従ってインストールしてください。

Applilet に関する詳細情報は Renesas ウェブサイトを参照してください。

<http://japan.renesas.com/applilet> (日本サイト)

<http://www.renesas.com/applilet> (グローバルサイト)

3. チュートリアルプロジェクトワークスペース

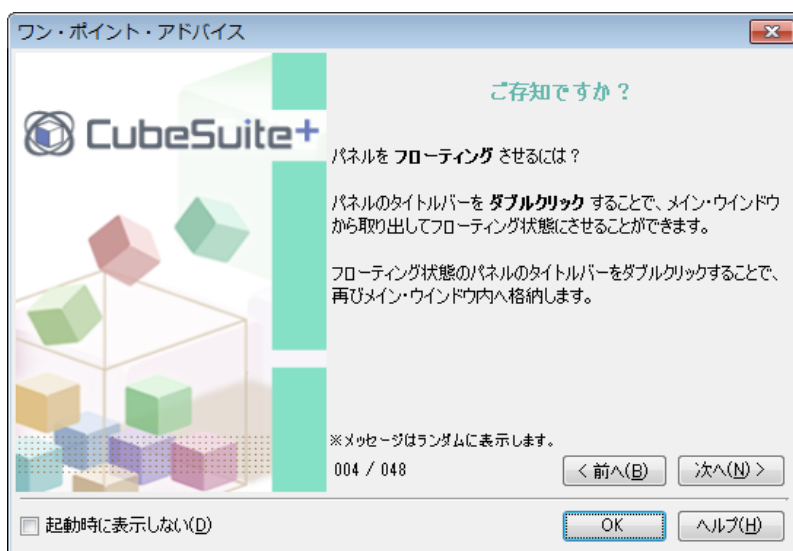
3.1 はじめに

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。

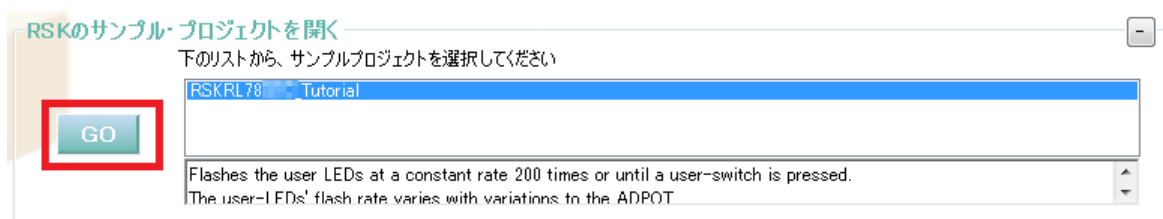
3.2 CubeSuite+の開始と E1 エミュレータの接続

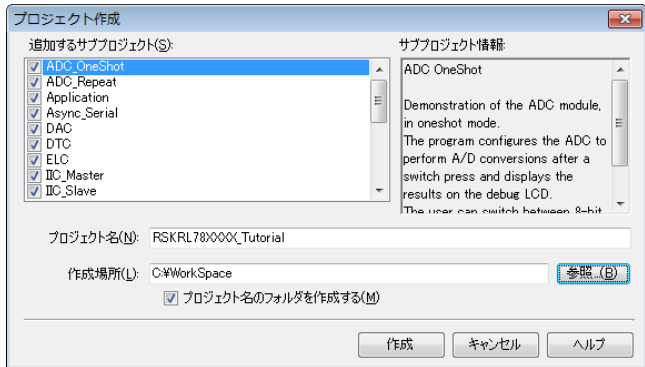
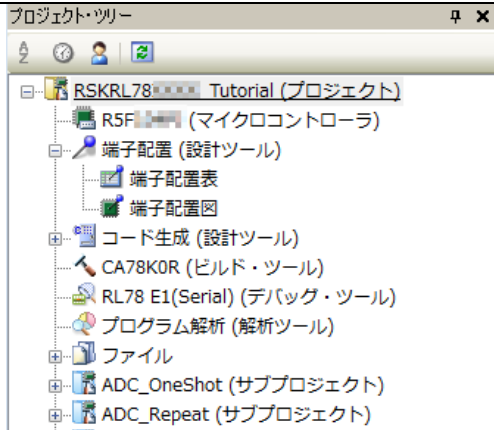
まず、Windows のスタートメニューから CubeSuite+を起動してください。

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。

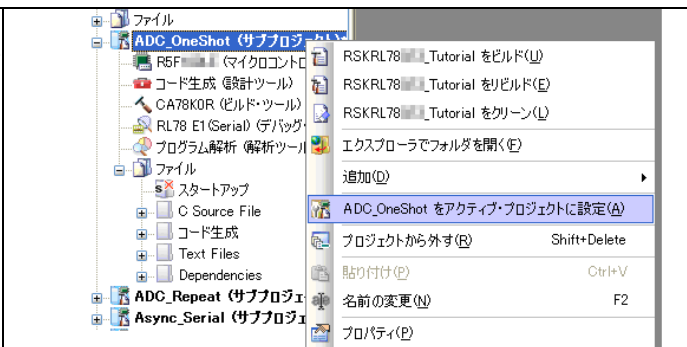


<OK>をクリックし、ダイアログを閉じてください。その後、スタートパネルが現れます。'RSK のサンプル・プロジェクトを開く'から RSKRL78L1C_Tutorial を選択し、<GO>をクリックしてください。この操作によって、RSKRL78L1C_Tutorial プロジェクトのコピーを保存します。

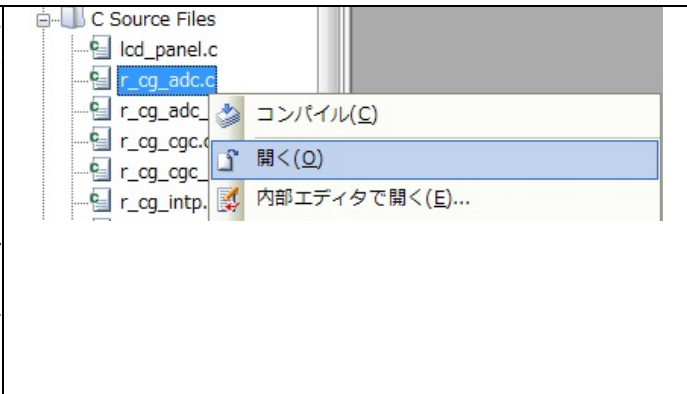


<ul style="list-style-type: none"> • CubeSuite+はプロジェクト作成ダイアログを表示します。 • 各サブプロジェクト名のチェックボックスをチェックし、サブプロジェクトをすべて追加してください。各サブプロジェクトの情報はダイアログ上のサブプロジェクト情報の下に表示されます。 • プロジェクト名を入力し、作成場所を指定して<作成>をクリックしてください。 • コピーされたファイルを参照するには、プロジェクトツリーにリスト化されたファイルをダブルクリックしてください。新しいウィンドウが開きます。 	
<ul style="list-style-type: none"> • RSKRL78L1C_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 RSKRL78L1C_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、"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 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。



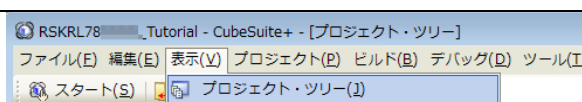
- ファイルフォルダは 3 つのサブフォルダを含んでいます。この構造はすべてのプロジェクト共通です。
- ソースファイルのうちいくつかは Applilet によって生成されたソースファイルです。これらのファイルは Applilet によって生成されたことを示すため、ファイル名の前に 'r_cg' が付けられます。
- ファイルを参照するには、参照したいファイルを右クリックし、"開く"を選択します。ファイルをダブルクリックしても参照できます。



3.3 デバッグ・ツールの設定

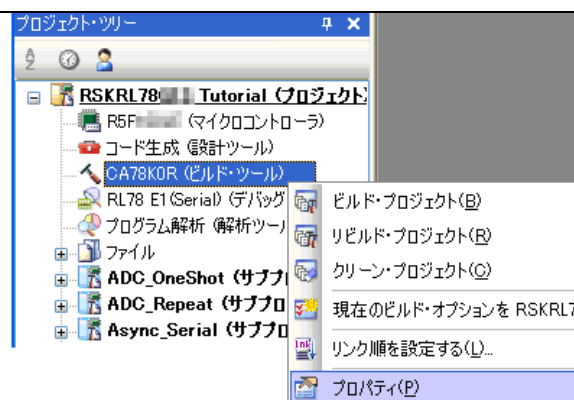
注：Tutorial プロジェクトは予めデバッグ・ツールの設定がされています。このセクションは新しいプロジェクトを作成するための説明です。

- プロジェクトツリーは 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せん。（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：

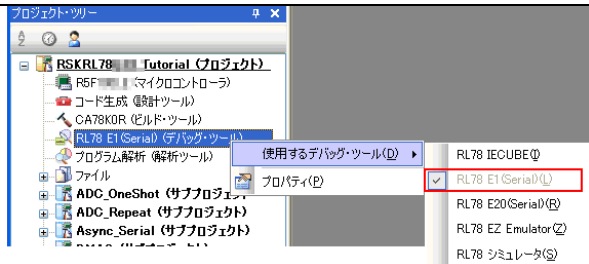
- CA78K0R（ビルド・ツール）を右クリックしてください。
- プロパティを選択してください。



- 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。

プロパティ	
RL78 E1(Serial) のプロパティ	
内部ROM/RAM	
内部ROMサイズ[Kバイト]	256
内部RAMサイズ[Kバイト]	16384
データフラッシュ・メモリ・サイズ[Kバイト]	0
クロック	
メイン・クロック周波数 [MHz]	12.00
サブ・クロック周波数 [kHz]	32.768
モニタ・クロック	システム
ターゲット・ボードとの接続	
エミュレータから電源供給をする(最大200mA)	はい
供給電圧	3.3V
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・ボルトテージ・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

プロジェクトはコードをダウンロードした後、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。

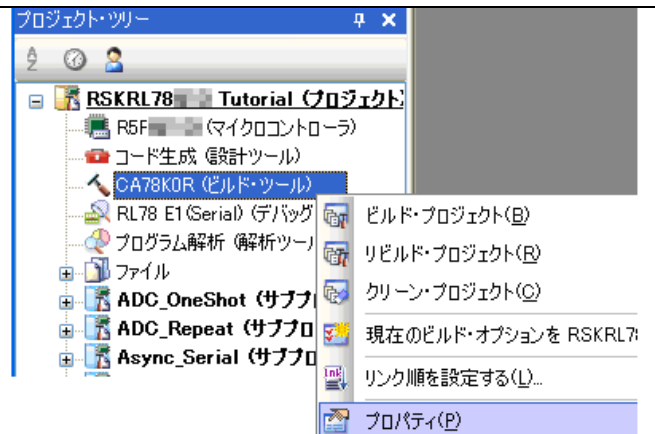
プロパティ	
RL78 E1(Serial) のプロパティ	
ダウンロード	
ダウンロードするファイル	[1]
ダウンロード後にCPUをリセットする	はい
ダウンロード前にフラッシュROMを消去する	いいえ
イベント設定位置の自動変更方法	イベントを保留にする
デバッグ情報	
CPUリセット後に指定シンボル位置まで実行する	はい
指定シンボル	._main
スタートアップ開始シンボル	._cstart
スタートアップ終了シンボル	._cend

3.4 ビルド設定

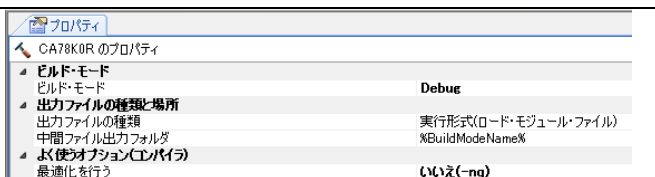
ビルド設定は CA78K0R（ビルド・ツール）のプロパティから選択できます。利用可能なオプションは DefaultBuild、Debug、Release です。DefaultBuild および Debug はデバッグを備えた設定になっています。Release は最終の ROM 化用プログラムのために設定されます。

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グ設定です。最適化が有効の場合、デバッグ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。



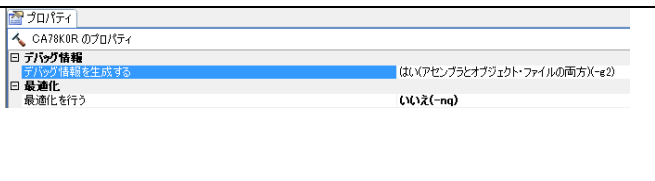
- 共通オプションタブを選択してください。
- ビルド・モードを Debug に設定してください。



- 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。








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 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。

<ul style="list-style-type: none"> 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。

<ul style="list-style-type: none"> 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。 	
<ul style="list-style-type: none"> 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。 	
<ul style="list-style-type: none"> 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。 	
<ul style="list-style-type: none"> 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。 	
<ul style="list-style-type: none"> 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 RSKRL78L1C ユーザーズマニュアルを参照してください。

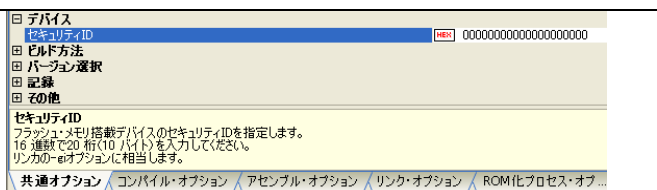
E1 のホストコンピュータへの接続方法は、クイックスタートガイドに詳しく記載されています。以下は、クイックスタートガイドの手順が踏まれ、E1 用のドライバが既にインストールされていることを前提としています。

- LCD Application Board V2(拡張基板)を CPU ボードの JA4(50pin ソケット)に取り付け、コネクタの全てのピンがソケットに収まっていることを確認してください。
- E1 をご使用のホストコンピュータの 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 3.3 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。

4.3 E1 によるターゲットの接続

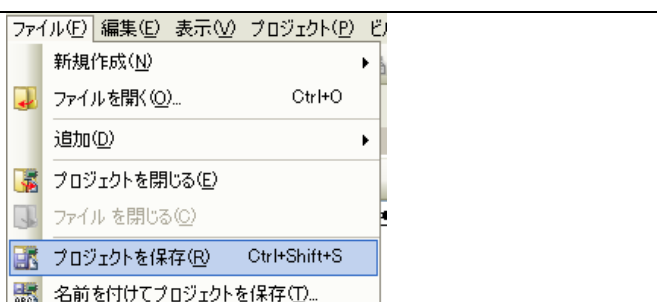
ここでは、デバイスへの接続、フラッシュへのプログラミングおよびコード実行について説明します。

- 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。

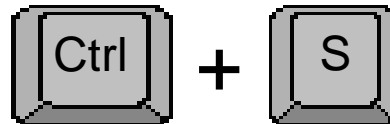
- 'ファイル' | 'すべてを保存' を選択します。



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。



5. チュートリアル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

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。これはプログラムの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 */
    Switch_Init();

    /* Set the call back function when SW3 is pressed */
    SetSwitchPressCallback(cb_switch_press);

    /* Enable and configure LCD display. */
    Init_Display_Panel();

    /* Display the device family name on LCD.*/
    Display_Panel_String(PANEL_LCD_LINE1, " RL78");

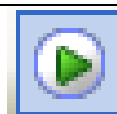
    /* Set up UART1 receive buffer */
    R_UART1_Receive((uint8_t *)&g_rx_char, 1);

    /* Enable UART1 operations */
    R_UART1_Start();
}

```

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'ボタンをクリックしてください。



6. 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認します。

Tutorial コードでは、CPU ボードとホストコンピュータでシリアル通信を行います。そのため、ターミナルソフトと RS232 ケーブルが必要となります。ターミナルソフトの設定を以下に示します。

- ボー・レート : 19200bps
- データ長 : 8bit
- パリティ : なし
- ストップビット : 1bit
- フロー制御 : なし



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ー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。

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。

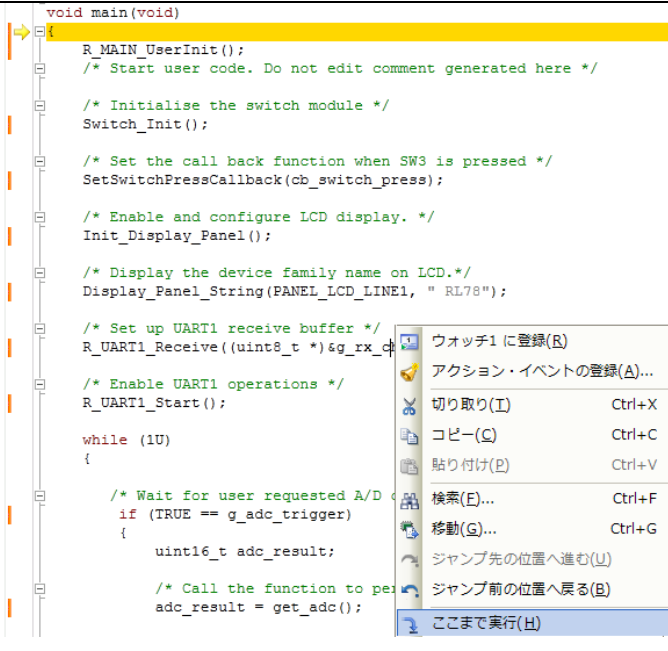
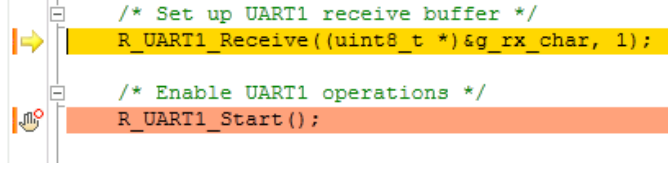
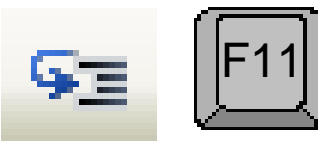
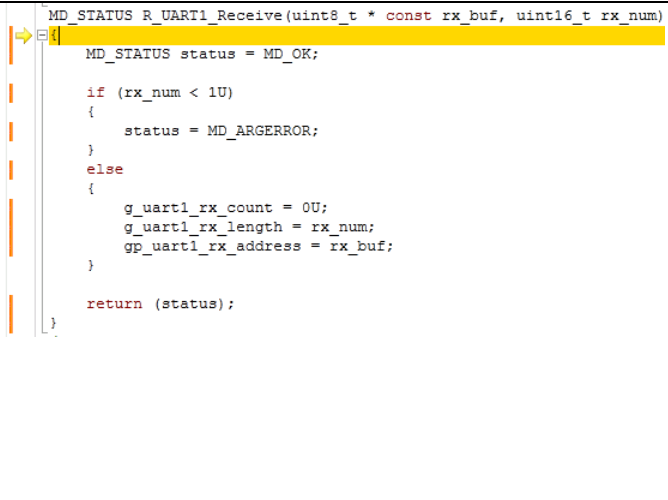
Tutorial コードがマイクロコントローラにダウンロードされていることを確認し、デバッグツールバーの'CPU リセット'をクリックしてください。



<ul style="list-style-type: none"> • 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。 	<pre> 98: void main(void) 99: { main: ➔ 00781 c7 PUSH HL 00782 c1 PUSH AX 00783 fbf8ff MOVW HL,SP 100: R_MAIN_UserInit(); 00786 fced0700 CALL !!_R_MAIN_UserInit 101: /* Start user code. Do not edit comment generated here */ 102: 103: /* Initialise the switch module */ 104: Switch_Init(); 0078a fcd30c00 CALL !!_Switch_Init 105: 106: /* Set the call back function when SW3 is pressed */ 107: SetSwitchPressCallback(cb_switch_press); 0078e 30f107 MOVW AX,#7F1H 00791 5200 MOV C,#0H 00793 f3 CLRB B 00794 fcd90c00 CALL !!_SetSwitchPressCallback 108: 109: /* Enable and configure LCD display. */ 110: Init_Display_Panel(); 00798 fc500e00 CALL !!_Init_Display_Panel 111: 112: /* Display the device family name on LCD.*/ 113: Display_Panel_String(PANEL_LCD_LINE1, " RL78"); 0079c 300030 MOVW AX,#3000H </pre>
 ← 逆アセンブルボタン  ← 混合ボタン	
<p>逆アセンブル表示または混合表示から C ソース表示に戻ります。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、"ソースヘジャンプ"をクリックしてください。</p>	

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。

<ul style="list-style-type: none"> R_UART1_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。Init_Display_Panel 関数は LCD モジュールを初期化し、Display_Panel_String 関数はストリングデータ'RL78'を LCD の最下位セグメント位置に表示します。 	 <pre> void main(void) { R_MAIN_UserInit(); /* Start user code. Do not edit comment generated here */ /* Initialise the switch module */ Switch_Init(); /* Set the call back function when SW3 is pressed */ SetSwitchPressCallback(cb_switch_press); /* Enable and configure LCD display. */ Init_Display_Panel(); /* Display the device family name on LCD.*/ Display_Panel_String(PANEL_LCD_LINE1, " RL78"); /* Set up UART1 receive buffer */ R_UART1_Receive((uint8_t *)&g_rx_char, 1); /* Enable UART1 operations */ R_UART1_Start(); while (1U) { /* Wait for user requested A/D conversion */ if (TRUE == g_adc_trigger) { uint16_t adc_result; /* Call the function to perform A/D conversion */ adc_result = get_adc(); } } } </pre>
<ul style="list-style-type: none"> R_UART1_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）。 	 <pre> /* Set up UART1 receive buffer */ R_UART1_Receive((uint8_t *)&g_rx_char, 1); /* Enable UART1 operations */ R_UART1_Start(); </pre>
<ul style="list-style-type: none"> '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_UART1_Receive 関数にエントリします。 	
<ul style="list-style-type: none"> プログラムカウンタは、R_UART1_Receive 関数に移動します。 この関数は、Applilet 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、UART 受信割り込み処理内で設定した受信バイト数に達したとき、UART コールバック関数に呼び出されます。 Applilet 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。 	 <pre> MD_STATUS R_UART1_Receive(uint8_t * const rx_buf, uint16_t rx_num) { MD_STATUS status = MD_OK; if (rx_num < 1U) { status = MD_ARGERROR; } else { g_uart1_rx_count = 0U; g_uart1_rx_length = rx_num; gp_uart1_rx_address = rx_buf; } return (status); } </pre>

- プログラムカウンタは R_UART1_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_UART1_Start 関数は、UART の開始、割り込みを許可します。プログラムは、RSK のスイッチ入力、または UART 割り込みが発生するまで while ループを実行します。

```

/* Set up UART1 receive buffer */
R_UART1_Receive((uint8_t *)&g_rx_char, 1);

/* Enable UART1 operations */
R_UART1_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_sau_user.c’ファイルを開いてください。r_uart1_callback_receiveend 関数までスクロールしてください。

```

/* Enable UART1 operations */
R_UART1_Start();

while (1U)
{
    /* Wait for user requested A/D conversion
    if (TRUE == g_adc_trigger)
    {
        uint16_t adc_result;

        /* Call the function to perform an A/
        adc_result = get_adc();

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

```

- r_uart1_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。



```

static void r_uart1_callback_receiveend(void)
{
    /* Start user code. Do not edit comment generated here */

    /* Check the contents of g_rx_char */
    if (('c' == g_rx_char) || ('C' == g_rx_char))
    {
        g_adc_trigger = TRUE;
    }

    /* Set up UART1 receive buffer and callback function again */
    R_UART1_Receive((uint8_t *)&g_rx_char, 1);

    /* End user code. Do not edit comment generated here */
}

```

- 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_uart1_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。



```
static void r_uart1_callback_receiveend(void)
{
    /* Start user code. Do not edit comment generated here */

    /* Check the contents of g_rx_char */
    if (('c' == g_rx_char) || ('C' == g_rx_char))
    {
        g_adc_trigger = TRUE;
    }

    /* Set up UART1 receive buffer and callback function again */
    R_UART1_Receive((uint8_t *)&g_rx_char, 1);

    /* End user code. Do not edit comment generated here */
}
```

- 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。



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

/* Increment the adc_count and
if (16 == ++adc_count)
{
    adc_count = 0;
}
led_display_count(adc_count);
```

プログラムは、SW3 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 パネルおよびターミナル画面に表示します。さらに、RSK で A/D 変換回数を LED でバイナリ形式表示を行います。

- ‘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。



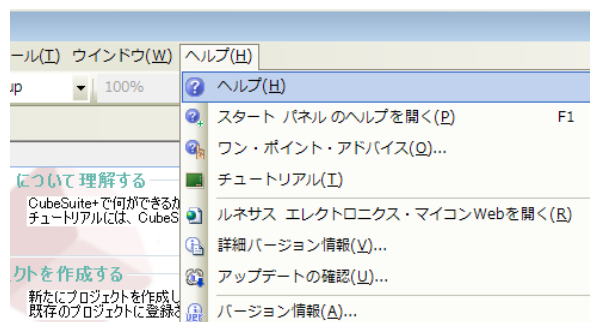
ハードウェアに関する詳細は、RSKRL78L1C ユーザーズマニュアルおよび RL78/L1C ユーザーズマニュアルハードウェア編を参照してください。

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

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/L1C マイクロコントローラに関する詳細情報は、RL78/L1C 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 ファミリー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/rskrl78l1c> (日本サイト)
<http://www.renesas.com/rskrl78l1c> (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、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。

© 2014 Renesas Electronics Europe Limited. All rights reserved.
 © 2014 Renesas Electronics Corporation. All rights reserved.
 © 2014 Renesas Solutions Corp. All rights reserved.

改訂記録	RSKRL78L1C 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	2014.01.15	—	初版発行
1.01	2014.03.19	—	「2. 略語および略称の説明」を更新
		8	・ビルドコンフィグレーション”Debug”、”Release”に最適化レベル情報を追加 ・Appliletの実行ファイル情報を追加
		19	逆アセンブル表示または混合表示からCソース表示へ戻す方法を追加
1.02	2014.04.04	10 - 22	一部の説明文、図に枠線を追加
		10	ページ最後の説明文を更新 “ _____をアクティブ・プロジェクトに設定” ↓ “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”
		19	・逆アセンブルボタン、混合ボタンを区別化 ・語句右クリック修正（右くりっく→右クリック）

RSKRL78L1C チュートリアルマニュアル(CubeSuite+)

発行年月日 2014年4月4日 Rev.1.02

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 エレクトロニクス株式会社 〒100-0004 千代田区大手町2-6-2（日本ビル）

■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/>

RL78/L1C