User's Manual

32

RZ/T1 グループ

Renesas Starter Kit+

コード生成支援ツールチュートリアルマニュアル

e^2 studio 版

RZ ファミリ RZ/T1 シリーズ

ルネサス エレクトロニクス
www.renesas.com

Rev.1.00　2016.09

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。

CAUTION

With reference to Directive 2014/30/EU Article 2, clause 2 (e) this is a custom built evaluation kit destined for professionals to be used solely at research and development facilities for such purposes. This equipment can cause radio frequency noise when used. In such cases, the user/operator of the equipment may be required to take appropriate countermeasures under his responsibility.

CAUTION

This equipment should be handled like a CMOS semiconductor device. The user must take all precautions to avoid build-up of static electricity while working with this equipment. All test and measurement tool including the workbench must be grounded. The user/operator must be grounded using the wrist strap. The connectors and/or device pins should not be touched with bare hands.

EEDT-ST-004-10

For customers in the European Union only

The WEEE (Waste Electrical and Electronic Equipment) regulations put responsibilities on producers for the collection and recycling or disposal of electrical and electronic waste. Return of WEEE under these regulations is applicable in the European Union only. This equipment (including all accessories) is not intended for household use. After use the equipment cannot be disposed of as household waste, and the WEEE must be treated, recycled and disposed of in an environmentally sound manner. Renesas Electronics Europe GmbH can take back end of life equipment, register for this service at http://www.renesas.eu/weee

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、統合開発環境 e2 studio および RZ 用コード生成プラグインを使用して RSK+ プラットフォーム用プロジェクトを作成するための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としています。

このマニュアルは、段階的に e2 studio 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。RZ/T1 グループの動作に関する詳細は、ハードウェアマニュアルおよび、サンプルプログラムを参照してください。

本マイコンは、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。

RZ/T1 グループ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+RZT1 ユーザーズマニュアル	R20UT3551JJ
チュートリアルマニュアル	RSK+の環境のセットアップ、サンプルコードの実行とデバッギング方法の説明	RSK+RZT1 チュートリアルマニュアル	R20UT3243JG
クイックスタートガイド	RSK+のセットアップおよびサンプルコード実行のための簡易ガイド	RSK+RZT1 クイックスタートガイド	R20UT3244JG
コード生成支援ツール チュートリアルマニュアル	独立型のコード生成支援ツールに関する説明書	RSK+RZT1 コード生成支援ツール チュートリアルマニュアル	R20UT3281JG （本マニュアル）
回路図	RSK+RZT1の回路図詳細	RSK+RZT1 評価ボード回路図	R20UT3241EG
ユーザーズマニュアル ハードウェア編	RZ/T1グループに関する技術的な詳細	RZT1グループユーザーズマニュアル ハードウェア編	R01UH0483JJ
NORフラッシュ ブートローダ アプリケーションノート	NORフラッシュブートローダ プログラムの操作に関する 詳細説明書	RZT1グループアプリケーションノート： NORフラッシュブートローダ	R01AN2470JG
QSPIフラッシュ ブートローダ アプリケーションノート	QSPIフラッシュブートローダ プログラムの操作に関する 詳細説明書	RZT1グループアプリケーションノート： QSPIフラッシュブートローダ	R01AN2471JG

2. 略語および略称の説明

略語／略称	英語名	日本語名
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
COM	COMmunications port referring to PC serial port	COMポート(PCのシリアルポート)
CPU	Central Processing Unit	中央処理装置
DVD	Digital Versatile Disc	デジタル多用途ディスク
E1	Renesas On-chip Debugger	ルネサス製内蔵デバッガ
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 line	割り込み要求線
LCD	Liquid Crystal Display	液晶ディスプレイ
LED	Light Emitting Diode	発光ダイオード
MCU	Micro-controller Unit	マイクロコントローラユニット
PC	Personal Computer	パーソナルコンピュータ
Pmod™	Digilent Pmod™ Compatible connector. Pmod™ is registered to Digilent Inc. Digilent-Pmod_Interface_Specification	Digilent Pmod™ 互換コネクタ。Pmod™ はDigilent社の登録商標です。 Digilent-Pmod_Interface_Specification
PLL	Phase-locked Loop	位相同期回路
QSPI	Quad Serial Peripheral Interface	クアッドシリアルペリフェラルインタフェース
RSK	Renesas Starter Kit	ルネサススタータキット
RSK+	Renesas Starter Kit+ (denotes extra functionality over standard RSK)	ルネサススタータキット+(標準RSKに機能を追加したもの)
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

目　次

1. 概要

1.1 目的

本 RSK+ は、RZ/T1 グループ用の評価ツールです。本マニュアルは、統合開発環境 e2 studio 用のコード生成プラグインを使用して、RSK+ プラットフォーム用の作業プロジェクトを作成する方法について説明します。

1.2 特徴

本マニュアルでは、以下の機能を評価するためのプロジェクト作成について説明します。

- e2 studio を使用したプロジェクトの作成
- コード生成プラグインを使用したコードの生成
- スイッチ、LED、ポテンショメータなどのユーザ回路

本 RSK+ ボードは、RZ/T1 の動作に必要な回路をすべて備えています。

2. はじめに

本マニュアルは、RZ ファミリのコード生成プラグインを統合開発環境 e2 studio と共に使用して、RSK+プラットフォームに作業プロジェクトを作成する方法について、チュートリアル形式で回答することを目的としています。チュートリアルでは以下の内容について説明します。

- e2 studio を使用したプロジェクトの生成
- e2 studio 用のコード生成プラグインの使用方法
- カスタムコードの統合
- e2 studio プロジェクトのビルドと実行

プロジェクト生成支援ツールは、下記 2 つのビルド構成のいずれかを用いてチュートリアルプロジェクトを作成します。

- 「HardwareDebug」は、デバッガサポートを含めて構築されるプロジェクトです。最適化レベルはゼロに設定されています。
- 「Release」は、製品のリリースに適したコードを作成するように最適化されたコンパイルオプションを含むプロジェクトです。

本書で使用するスクリーンショットには “RZxx” という文字が表示されているものがあります。これらは、RZ ファミリ全体に共通の一般的なスクリーンショットです。本書では、RZxx を RZ/T1 に読み替えてください。

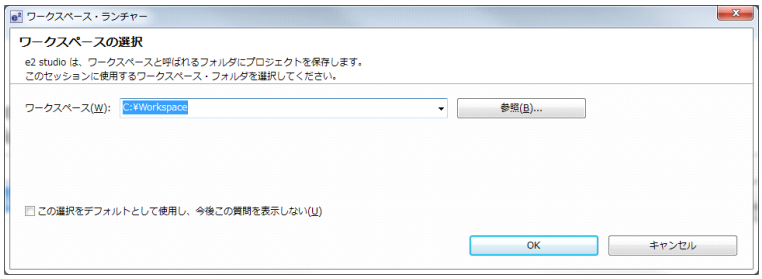
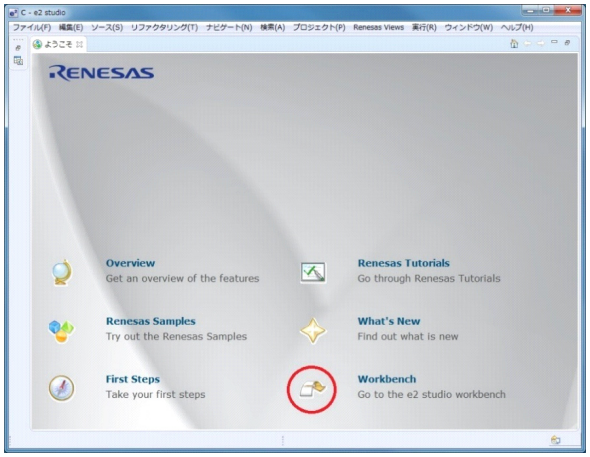
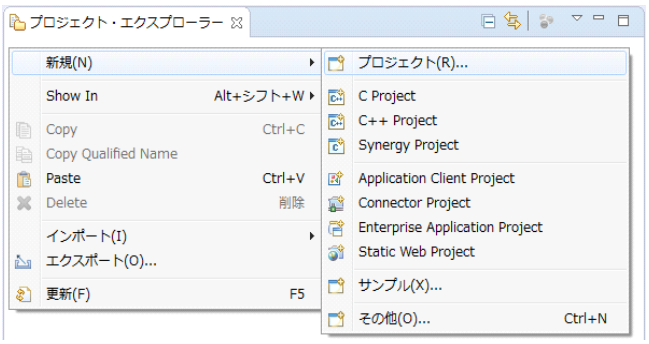
本書で説明するチュートリアルは、RSK の使用方法を示すためのものであり、e2 studio のデバッガ、コンパイラツールチェーンおよび J-Link LITE エミュレータに関する包括的な手引きではありません。詳細情報については、関連するユーザマニュアルを参照してください。

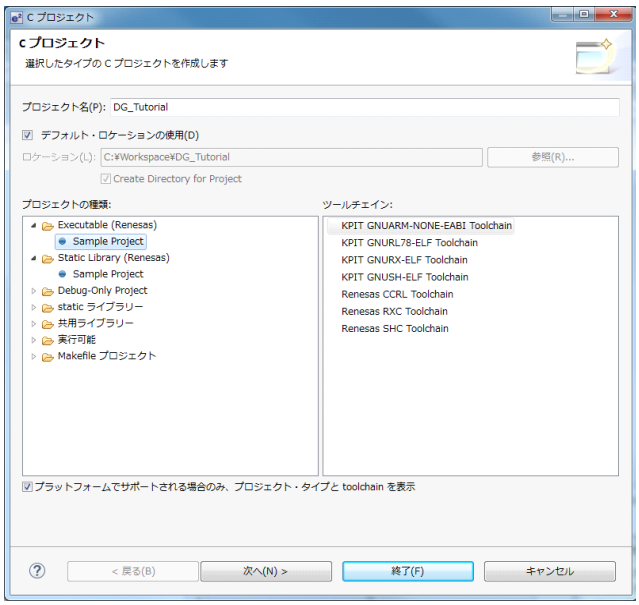
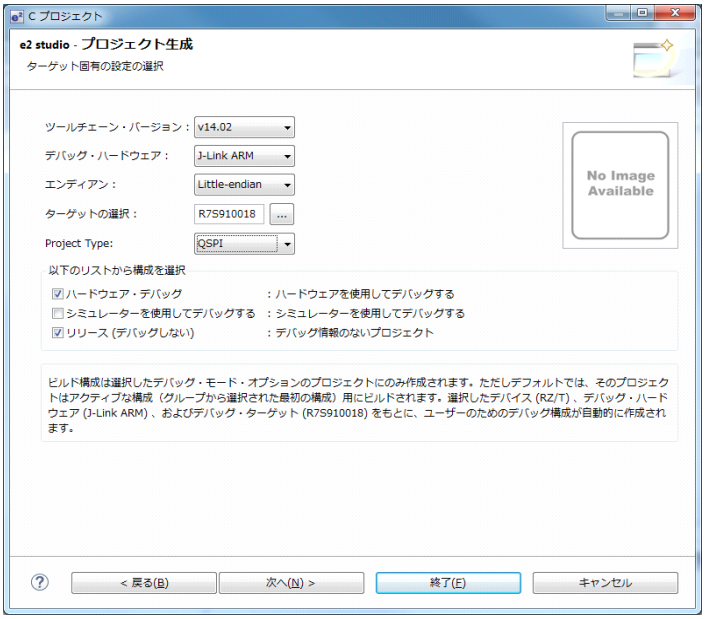
3. e2 studio を使用したプロジェクトの生成方法

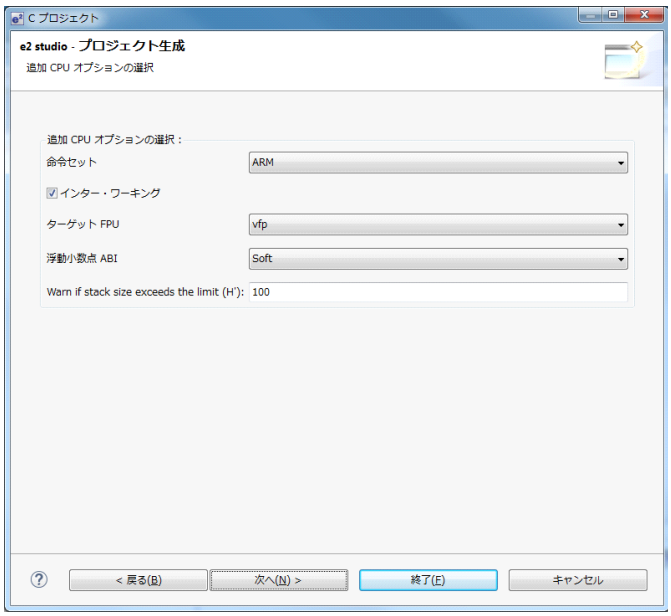
3.1 はじめに

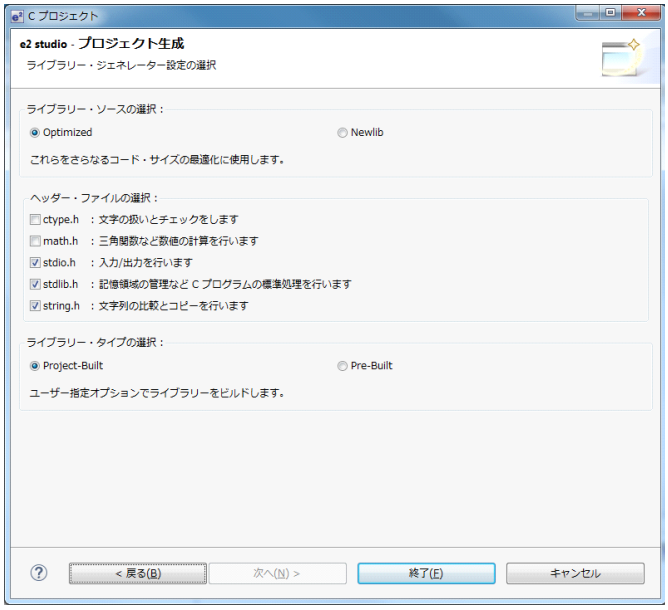
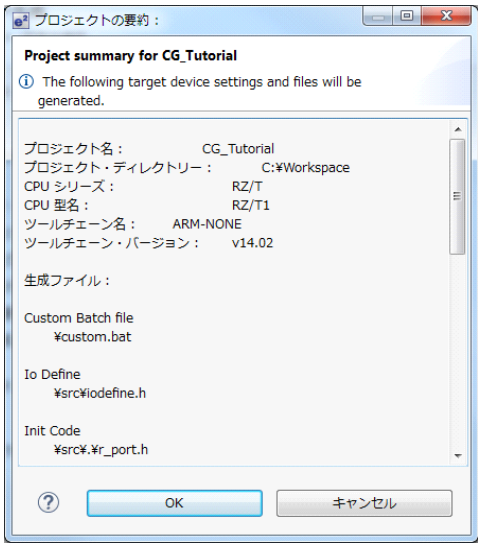
本章では、コード生成支援ツールを使用して周辺機能制御ドライバを作成および追加するための準備として、RZ/T1 用の C プロジェクトを新規に作成する手順について解説します。このプロジェクト作成手順は使用する MCU 向けの C ソースをデバッグするために必要です。

3.2 プロジェクトの作成方法

• e2 studio を起動します。「参照」をクリックし、プロジェクトを保存するワークスペース・フォルダを指定します。	
• 「ようこそ」タブのページで、「Go to the e2 studio workbench」をクリックします。	
• 「プロジェクト・エクスプローラー」ウィンドウ内で右クリックし、「新規＞ C Project」の順に選択して C プロジェクトを新規作成します。もしくは、メニューバーの「ファイル＞新規＞ C Project」を選択します。	

• 新規プロジェクト名として「CG_Tutorial」を入力します。「プロジェクトの種類」の項目では「Sample Project」を選択し、「ツールチェイン」の項目では「KPIT GNUARM-NONE-EABI Toolchain」を選択します。「次へ」をクリックします。	
• 「ターゲット固有の設定の選択」ダイアログ内で、右画面の通りに項目を選択します。 • 「ターゲットの選択」の項目では、以下の順に選択し、R7S910018 MCU を指定します。「RZ/T ＞ RZ/T1 ＞ RZ/T1 - 320 pin ＞ R7S910018」 • 「次へ」をクリックします。	

• 「コード生成を使用する」にチェックを入れます。 • 「次へ」をクリックします。	
• 「追加 CPU オプションの選択」はデフォルト設定のままにしておきます。 • 「次へ」をクリックします。	

• 「ライブラリー・ジェネレーター設定の選択」ダイアログでは、デフォルト設定のままにしておきます。 • 「終了」をクリックします。	
• プロジェクトの要約が表示されるので、「OK」をクリックし、プロジェクト生成を完了してください。	

生成されたサンプルは、プログラムのビルド・実行機能を十分に備えていますが、本チュートリアルの目的外のため、サンプルの機能テストは行いません。

注 . サンプルを実行すると、本 RSK+ ボード上の LED0 が点滅します。点滅の間隔はポテンショメータ (RV1) の回転位置によって変わります。SW3 を押下することで点滅の有効／無効を切り替えることができます。ただし、本サンプルの実際の動作を保証するものではありません。

「プロジェクト」メニューの「プロジェクトのビルド」または  ボタンを使用して、エラーなくビルドできることを確認してください。

4. コード生成支援ツールの使用

4.1 はじめに

コード生成支援ツールは、C ソースコードのテンプレートを生成するための GUI ツールです。このツールは、e2 studio のプラグインとして、またはスタンドアロンのアプリケーションとしての 2 つのバージョンがあります。RSK+RZT1 ではプラグインを使用します。コード生成支援ツールを用いて、GUI によって直感的に MCU の各周辺モジュールの制御パラメータを設定することが可能です。特に、ユーザーズマニュアルハードウェア編の確認などの手間を省くことが可能です。

本チュートリアルマニュアルの記載手順に従って操作すると、CG_Tutorial という e2 studio プロジェクトを生成できます。CG_Tutorial プロジェクトの完全版は DVD にも含まれているので、クイックスタートガイドの手順に従って、e2 studio にインポートすることも可能です。本チュートリアルマニュアルは、ユーザがカスタムした e2 studio プロジェクトを生成することを目的に、コード生成支援ツールの使用方法を修得したい方を対象としたものです。

プロジェクトを生成すると、「コード生成」機能を使用して、選択された各 MCU 機能に対して 3 種類のモジュールコードファイルが生成されます。これらのファイルは、それぞれ「r_cg_xxx.h」、「r_cg_xxx.c」、「r_cg_xxx_user.c」と命名されています。「xxx」の部分は、「scifa」など該当する MCU 周辺機能の略称が入ります。このモジュールコードファイル内で、必要に応じてユーザコードを追加することが可能です。追加するユーザコードは、以下のコメントエリアの間に記述してください。

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

このコメントエリアは、コード生成支援ツールにより自動認識され、このエリア内に追加されたユーザコードは後続のコード生成作業において保持されます。コメントエリア以外に追加されたユーザコードは、後続のコード生成作業時に上書きされます。

CG_Tutorial プロジェクトでは、外部トリガによる ADC モジュール、割り込み制御ユニット (ICU)、コンペアマッチタイマ (CMT)、LED (I/O ポート端子制御) 機能を使用します。本チュートリアルマニュアルでは、以下の各項目の設定方法について具体的に説明します。

- 点滅させる LED ポートの設定
- サンプリング周期を設定する A/D チャネルの設定
- インターバル周期を生成するタイマチャネルの設定
- LED 点滅の有効／無効を切り替えるスイッチの設定

4.2 節では、コード生成支援ツールの主要なユーザインタフェース機能の概要について説明し、4.3.1 節では各周辺機能の構成について説明します。第 5 章では、テンプレートコードの詳細構成、コード生成支援ツールが提供するエリアへのユーザコードの追加方法、さらに 3.2 節で生成したプロジェクトに対するその他の変更方法について説明します。

4.2 コード生成支援ツールの概要

本章では、コード生成支援ツールの概要を紹介します。本マニュアルはスタンドアロン型のコード生成支援ツールである AP4 に対応しています。

e2 studio のメニューから、「ウィンドウ＞パースペクティブを開く＞その他」の順に選択します。**図 4.1** に示す「パースペクティブを開く」ダイアログで「コード生成」を選択し、「OK」をクリックしてください。

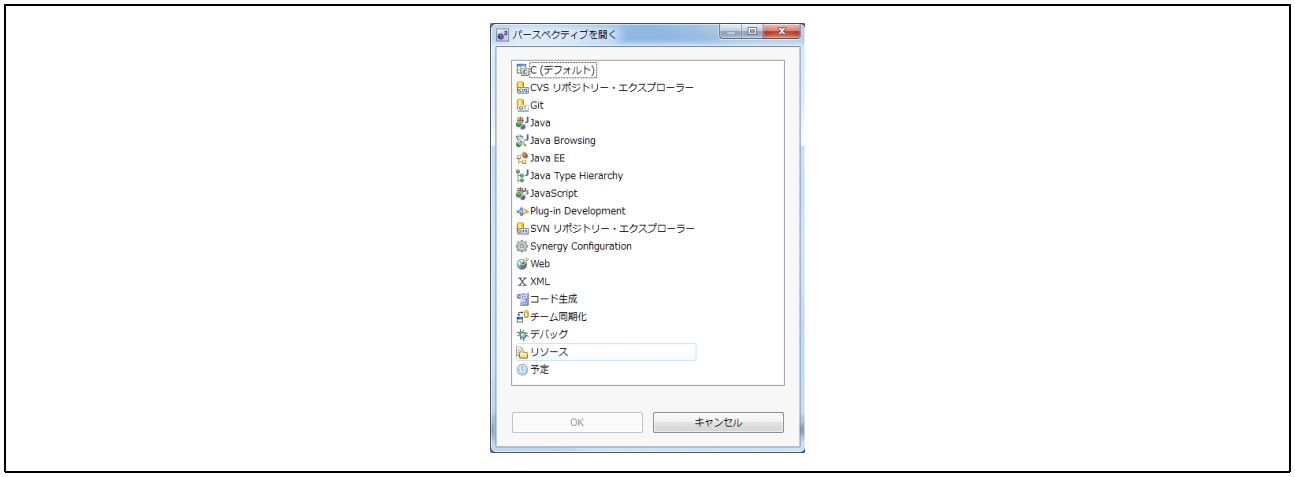


図 4.1 パースペクティブの変更

拡張子 ".cpg" のついたコード生成支援ツールのプロジェクトファイルは、CG_Tutorial プロジェクトの .settings/CodeGenerator ディレクトリにあります。また、コード生成支援ツールは、src フォルダ内に "cg_src" という名前のフォルダを作成し、生成されたソースおよびヘッダファイルを格納します。このフォルダは、コード生成支援ツール関連ファイルのみに使用し、それ以外のファイルは src フォルダ内もしくはその下に別のフォルダを作成して保存することを推奨します。

コード生成支援ツールにおける周辺機能の設定画面を**図 4.2** に示します。

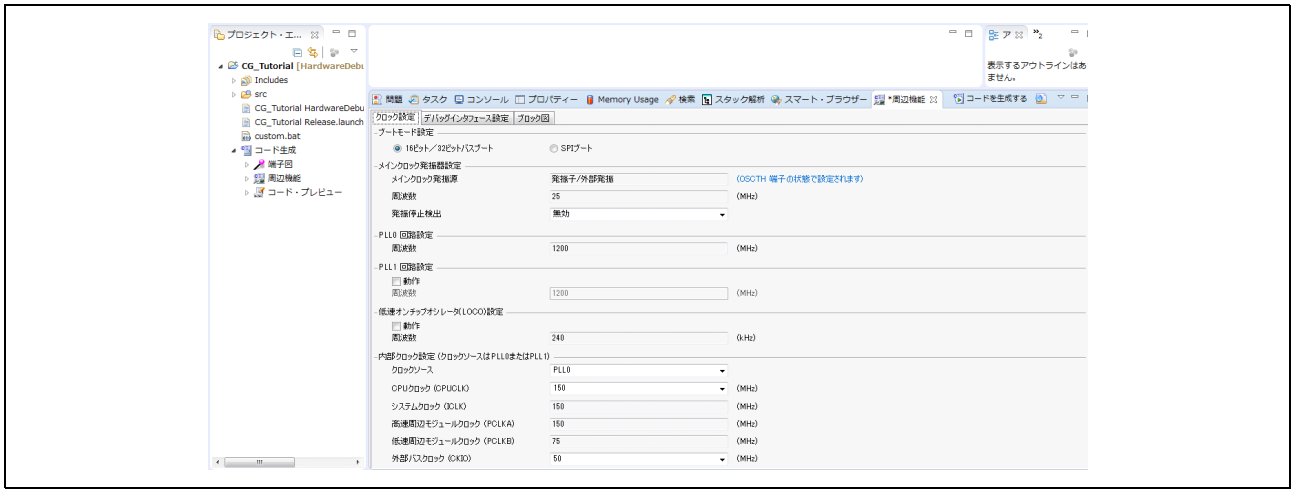


図 4.2 周辺機能の設定画面

コード生成支援ツールは、GUI 機能を使用して MCU のサブシステムと周辺機能を構成します。これらの構成が完了した後、「コードを生成する」ボタンをクリックすることで、e2 studio プロジェクトの src フォルダ内に複数のソースファイルとヘッダファイルが作成されます。この後、いくつかの手順を踏んで、プロジェクトの構成が完了し使用できるようになります。

MCU 周辺機能の構成画面へは、プロジェクト・ビューのツリーから「コード生成＞周辺機能」を選択し、目的の機能をダブルクリックしてください。

また、現在の周辺機能の設定に対して生成されるコードのプレビューを見ることもできます。画面左側の「コード生成＞コード・プレビュー」を選択し、目的の機能をダブルクリックしてください。

4.3 コード生成

本節では、必要な MCU 周辺機能の構成手順について説明します。

注 . 説明がない設定オプションについては、デフォルト設定のままにしてください。

4.3.1 周辺機能の構成

4.3.1.1 クロック発生回路

図 4.3 は、コード生成ツールのクロック発生回路に関連する設定タブのスクリーンショットです。

本チュートリアルでは、クロック発振源として 25MHz の水晶発振子を使用します。メインクロックは PLL 回路を選択します。このメインクロックまたは PLL 回路をクロック源として使用するように構成できます。

プロジェクト・ツリーの「コード生成＞周辺機能」リスト内の「クロック発生回路」エントリをダブルクリックします。

図 4.3 に示すように、「クロック設定」タブの下の「クロック発生回路」のオプションを設定します。

次ページに示すとおり、「ブロック図」タブの下に、「クロック発生回路」のブロック図があります。システムクロックと周辺クロックの各設定を確認してください

図 4.3 クロック設定タブ

4.3.1.2 I/O ポート

I/O ポートは、ユーザ LED 用の出力端子を割り当てる構成となっています。CG_Tutorial では、LED0 のみ使用します。ユーザの LED 接続ポート端子の概要を**表 4.1** に示します。

機能	MCU 端子	I/O ポート	備考
LED0	A5	PF7	
LED1	E3	P56	使用せず
LED2	K16	P77	使用せず
LED3	J18	PA0	使用せず

表4.1 I/Oポートの接続

接続の詳細は、RSK+RZT1 評価ボード回路図（R20UT3241EG）を参照してください。

「プロジェクト・ツリー＞周辺機能＞ I/O ポート」の順に選択し、エントリ「I/O Port」をダブルクリックしてリストを展開したら、**表 4.1** に記載の設定を行ってください。

図 4.4 LED ポート端子の構成に示すようにポートを設定してください。

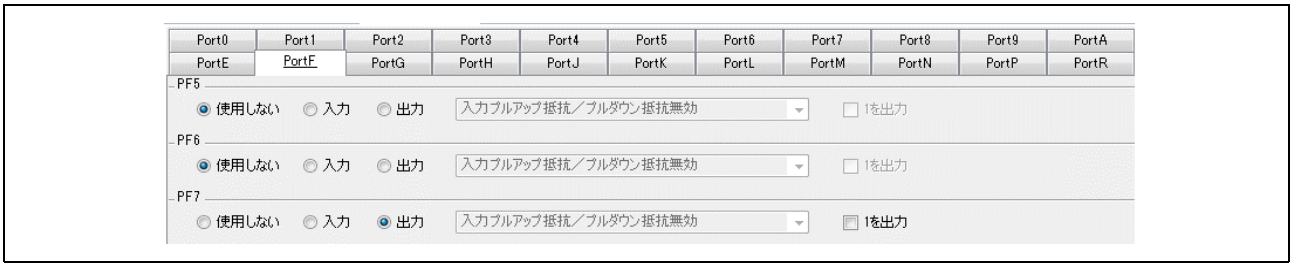


図 4.4 LED ポート端子の構成

4.3.1.3 コンペアマッチタイマ (CMT)

コンペアマッチタイマを設定し、LED0 の点滅に使用するインターバルを生成します。

「プロジェクト・ツリー＞ CG_Tutorial ＞コード生成＞周辺機能」の中のエントリ「コンペアマッチタイマ」をダブルクリックしてください。

図 4.5 に示すように CMT チャネルを設定してください。

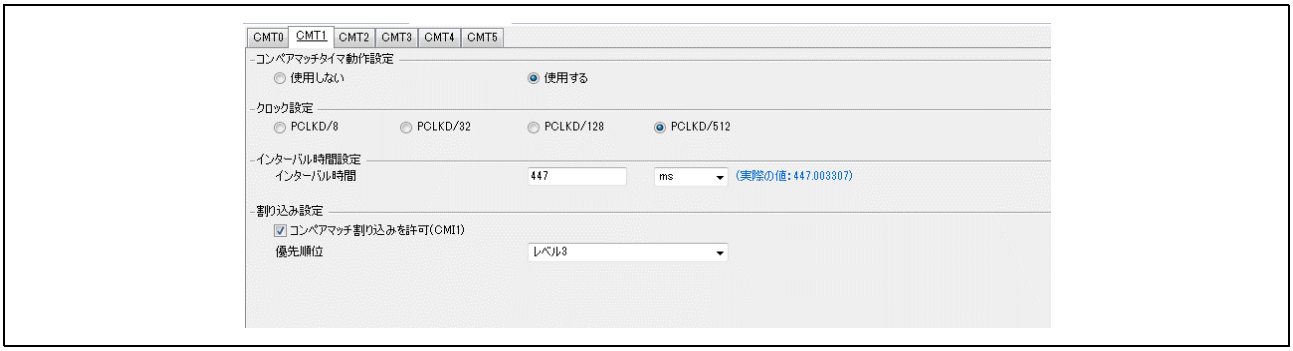


図 4.5 CMT 設定タブ

4.3.1.4 A/D コンバータ

A/D コンバータは、RV1 ポテンショメータのアナログ出力値をサンプリングする構成となっています。A/D コンバータは、マイコンの AN007 端子に接続された SW3 をユーザが押下することでサンプリングを行います。

「プロジェクト・ツリー＞周辺機能＞ 12 ビット A/D コンバータ」内のエントリ「S12AD0」エントリをダブルクリックしてください。

サブのタブ「設定 1」を以下に示すように設定してください。

図 4.6 A/D コンバータ設定タブ

図 4.7 A/D コンバータ設定タブ (2)

図 4.8 A/D コンバータ設定タブ (3)

図 4.9 A/D コンバータ設定タブ (4)

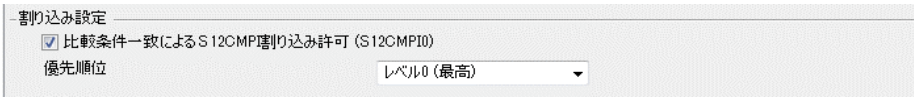


図 4.10 A/D コンバータ設定タブ (5)

4.3.1.5 割り込みコントローラユニット (ICU)

割り込みコントローラユニットは、ユーザのスイッチに接続された外部割り込み入力端子の構成に使用します。CG_Tutorial では、スイッチ SW3 のみを使用します。ユーザスイッチの接続の概要を**表 4.2** に示します。

機能	MCU 端子	I/O ポート	備考
NMI	H3	P35	使用せず
IRQ5	W3	PN5	使用せず
IRQ12	W15	P44	SW3

表4.2 ICUの接続

接続の詳細は RSK+RZT1 評価ボード回路図（R20UT3241EG）を参照してください。

「プロジェクト・ツリー＞周辺機能」リスト下のエントリ「割り込み制御」をダブルクリックし、リストを展開してください。

以下は、IRQ12 割り込みをトリガするスイッチ SW3 の設定です。

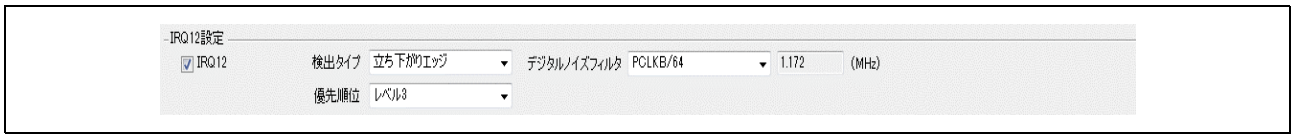


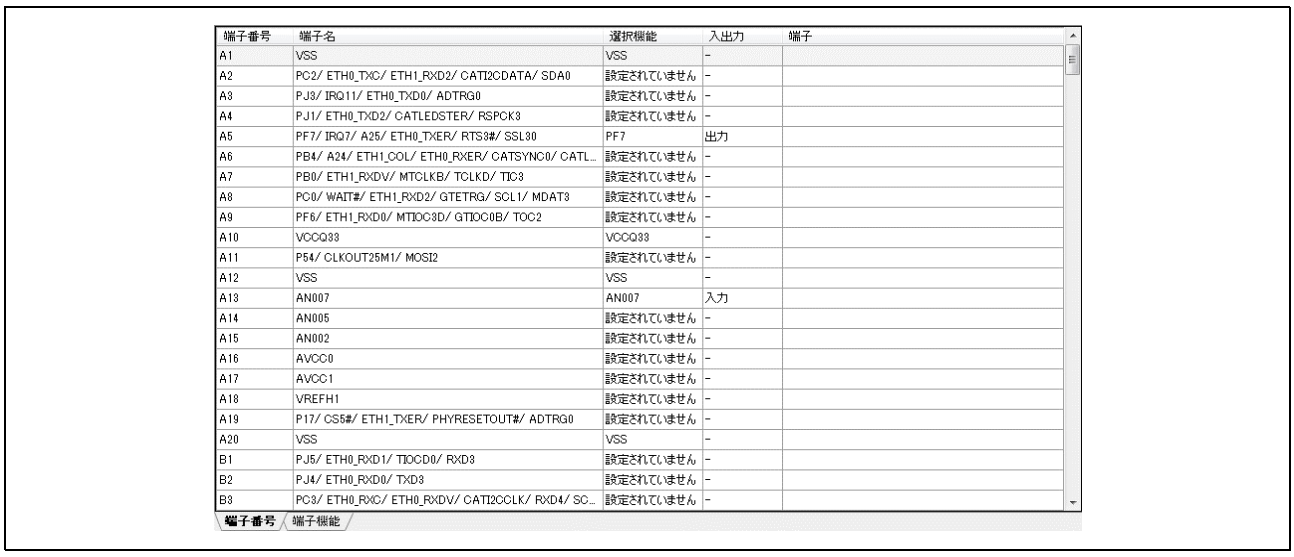
図 4.11 ICU 設定タブ

4.3.1.6 マルチファンクションピンコントローラ (MPC)

MPC は、MCU 端子のポート／周辺機能の選択とマッピングに使用します。デフォルトでは、I/O ポート、A/D モジュール、ICU モジュールのセットアップで示すように、周辺機能の構成時にマッピングを行います。MPC(マルチファンクションピンコントローラ) は、デフォルト機能のマッピングを再度行う必要があった場合に使用します。

「プロジェクト・ツリー＞端子図」内の「端子配置表」エントリをダブルクリックします。

図 4.12 および**図 4.13** に示すように、「端子番号」タブと「端子機能」タブで設定を行った各周辺機能に対するポート端子機能を必ず確認してください。



端子番号	端子名	選択機能	入出力	端子
A1	VSS	VSS	-	
A2	PC2/ ETH0_TXC/ ETH1_RXD2/ CATI2CDATA/ SDA0	設定されていません	-	
A3	PJ3/ IRQ11/ ETH0_TXD0/ ADTRG0	設定されていません	-	
A4	PJ1/ ETH0_TXD2/ CATLEDSTER/ RSPCK3	設定されていません	-	
A5	PF7/ IRQ7/ A25/ ETH0_TXER/ RTS3#/ SSL30	PF7	出力	
A6	PB4/ A24/ ETH1_COL/ ETH0_RXER/ CATSYNC0/ CATL...	設定されていません	-	
A7	PB0/ ETH1_RXDV/ MTCLKB/ TCLKD/ TIC3	設定されていません	-	
A8	PC0/ WAIT#/ ETH1_RXD2/ GTETRG/ SCL1/ MDAT3	設定されていません	-	
A9	PF6/ ETH1_RXD0/ MTIOC3D/ GTIOC0B/ TOC2	設定されていません	-	
A10	VCCQ33	VCCQ33	-	
A11	P54/ CLKOUT25M1/ MOSI2	設定されていません	-	
A12	VSS	VSS	-	
A13	AN007	AN007	入力	
A14	AN005	設定されていません	-	
A15	AN002	設定されていません	-	
A16	AVCC0	設定されていません	-	
A17	AVCC1	設定されていません	-	
A18	VREFH1	設定されていません	-	
A19	P17/ CS5#/ ETH1_TXER/ PHYRESETOUT#/ ADTRG0	設定されていません	-	
A20	VSS	VSS	-	
B1	PJ5/ ETH0_RXD1/ TIOCD0/ RXD3	設定されていません	-	
B2	PJ4/ ETH0_RXD0/ TXD3	設定されていません	-	
B3	PC3/ ETH0_RXC/ ETH0_RXDV/ CATI2CCLK/ RXD4/ SC...	設定されていません	-	

端子番号 端子機能

図 4.12 端子配置表の端子番号タブ

図 4.13 端子配置表の端子機能タブ

図 4.13 では、IRQ12（端子名）機能が MCU 端子番号 V4 に割り当てられていますが、RSK+RZT1 評価ボード回路図では、V4 は I/O ポート端子 P44 に接続されています。したがって、IRQ12 を P44(MCU 端子番号 W15) に再度割り当てる必要があります。

端子の割り当て／再割り当てを行う際は、該当端子をクリックしてプルダウンメニューを表示させてください。図 4.14 に示すように、メニュー内の使用可能な端子リストから W15 を選択します。

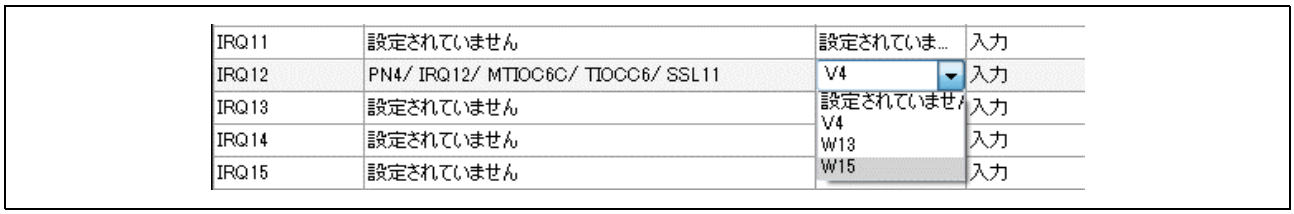


図 4.14 端子機能の配置／再配置

MPC は、周辺機能が構成された順に端子を割り当てます。既に機能を割り当てられた端子に他の機能を割り当てると、ソフトウェアの端子は兼用端子となります。また、MPC による機能の自動マッピングは、RSK+RZT1 の接続性に基づいていないため、必ず端子配置表内で端子機能を確認してください。

設定エラーが存在する場合、端子名が赤字で表示されます。

ある端子機能が、その機能を使用する周辺機器を構成する前に割り当てられた場合、端子配置表の「備考」欄にワーニングが表示されます。

4.3.2 コードの生成方法

周辺機能の構成が完了したら、「周辺機能」タブの右上にある「コードを生成する」ボタンをクリックします。図 4.15 に示すように、「出力」ウィンドウに、「The operation of generating file was successful」と表示されます。

図 4.15 コード生成支援ツールの出力ウィンドウ

5. 生成済みファイルへのコード追加

通常のプロジェクト開発では、ユーザはこの段階で、生成されたコードを展開して必要なアプリケーションを作成することができます。

コード生成支援ツールで生成されたファイルにコードを挿入する場合、以下のようなコメントで区切られたエリア内に配置してください。

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

上記において、_xxxxx_ は特定のエリアを示します。たとえば、ユーザ関数およびプロトタイプ宣言を挿入する場合は「function」、ユーザのグローバル変数の宣言を挿入する場合は「global」、プリプロセッサのインクルードディレクティブを挿入する場合は「include」となります。コメントエリア内に挿入されたユーザコードは、ユーザがコードを再度生成する場合に、コード生成支援ツールによる上書きから除外されます。

5.1 ファイルの除外

全てのサンプルコードは一つのメインファイルしか持てません。3.2 節で生成された init_main.c ファイルとコード生成支援ツールで生成された r_cg_main.c ファイルはどちらにもメイン関数が含まれます。init_main.c ファイルは、コードが生成されると自動的に除外されます。プロジェクトからファイルを除外する手順を以下に示します。

1. 「プロジェクト・エクスプローラー」ウィンドウ内で除外するファイルを探します。
2. 除外するファイル上で右クリックし、「ビルドから除外」を選択します。
3. 「すべて選択」をクリックし、該当する全ビルド構成に変更を適用します。
4. 「OK」をクリックします。

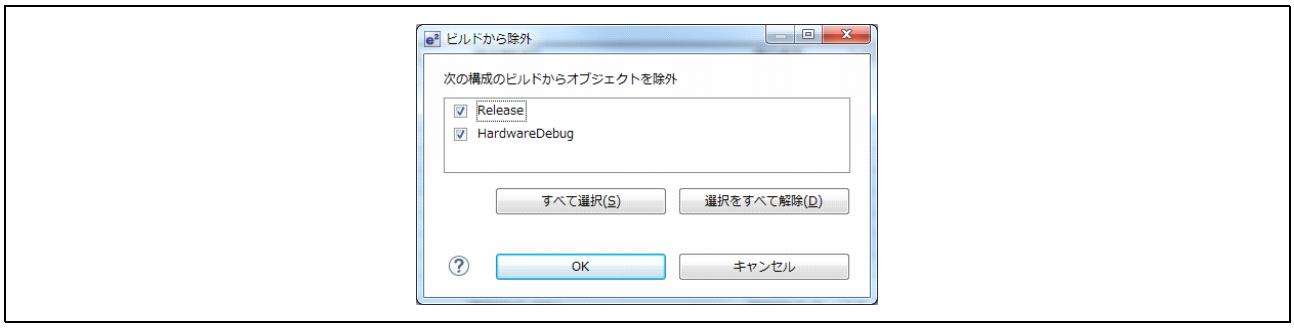


図 5.1 プロジェクトからファイルを除外

手順 1 で該当するファイルを複数選択し、一度に処理を行うことができます。以下のファイルを除外してください。

loader_param.c
r_ecm.c
r_ecm.h
r_icu_init.c
r_icu_init.h
r_system.h
typdefine.h

ファイルを再度インクルードするには、上記手順において、「選択をすべて解除」を選択し、「OK」をク

リックしてください。

5.2 生成ファイルへのコード挿入方法

本節では、新規に生成されたコード生成ファイルへのコード挿入方法について説明します。

各節は、コード生成支援ツールで生成されたソースファイルごとに構成され、これらのファイルを開く際には、src フォルダ下の e2 studio の「プロジェクト・ツリー」ウィンドウ内で該当ファイル名をダブルクリックしてください。

各節に記述されているコードは、本文からコピーし、表示された位置にある該当ファイル内に貼り付けてください。

5.2.1 r_cg_userdefine.h へのコード挿入

e2 studio の「プロジェクト・ツリー」ウィンドウ内のファイル名をダブルクリックして、本ファイルを開いてください。

ファイルの最後にあるユーザコードの記述可能エリアに以下のコードを挿入してください。

```
/* Start user code for function. Do not edit comment generated here */
#define LED0        (PORTF.PODR.BIT.B7)
/* End user code. Do not edit comment generated here */
```

5.2.2 r_cg_icu_user.c へのコード挿入

e2 studio の「プロジェクト・ツリー」ウィンドウ内のファイル名をダブルクリックして、本ファイルを開いてください。

特定のユーザコードの記述可能エリア（計 2ヶ所）に以下のコードをそれぞれ挿入してください。

```
/* Start user code for global. Do not edit comment generated here */
volatile uint8_t g_switch_press = 0;
/* End user code. Do not edit comment generated here */
/* Start user code. Do not edit comment generated here */
/* Invert the flag */
g_switch_press = (~g_switch_press);
/* End user code. Do not edit comment generated here */
```

5.2.3 r_cg_icu.h へのコード挿入

e2 studio の「プロジェクト・ツリー」ウィンドウ内のファイル名をダブルクリックして、本ファイルを開いてください。

ファイルの最後にあるユーザコードの記述可能エリアに以下のコードを挿入してください。

```
/* Start user code for function. Do not edit comment generated here */
extern volatile uint8_t g_switch_press;
/* End user code. Do not edit comment generated here */
```

5.2.4 r_cg_cmt_user.c へのコード挿入

e2 studio の「プロジェクト・ツリー」ウィンドウ内のファイル名をダブルクリックして、本ファイルを開いてください。

Global 変数及び関数を指定するファイルセクション内で、ユーザコードの記述可能エリアに以下のコードを挿入してください。

```
/* Start user code for include. Do not edit comment generated here */
#include "r_cg_icu.h"
#include "r_cg_cmt.h"
#include "r_cg_s12ad.h"
/* End user code. Do not edit comment generated here */


    /* Start user code for global. Do not edit comment generated here */

    /* Function prototype for scaling a value */
    static uint32_t scale_value (const uint32_t value, const uint32_t in_max, const uint32_t
out_max);

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

関数“void r_cmt_cmi1_interrupt(void)”内に以下のコードを挿入してください。

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

    /* Update the period based on the flag set in the switch handler interrupt */
    if (0 == g_switch_press)
     {
           /* scale the ADC value. ADC range: 0-4095, CMT range: 0 - 65478 */
           CMT1.CMCOR = scale_value ((uint32_t)(S12ADC0.ADDR7), 4095, 65478);
     }
     else
     {
         /* Do not update the CMT period */
     }

     LED0 = (~LED0);

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

ファイルの最後にあるユーザコードの記述可能エリアに以下を挿入してください。

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

/***********************************************************************************************
* Function Name: scale_value
* Description  : This function is CMI1 interrupt service routine.
*                The formula used
*                output = 1 + (value - 0) * (out_max - 0) / (in_max - 0)
*
*                Note - The actual and desired ranges' minimum value is assumed to be 0.
* Arguments    : uint32_t value   - value to scale
*                           uint32_t in_max  - maximum range of value to scale
*                           uint32_t out_max - maximum range of desired scale
* Return Value : None
***********************************************************************************************/
static uint32_t scale_value (const uint32_t value, const uint32_t in_max, const uint32_t out_max)
{
          uint32_t output;

          output = (out_max - 0) / (in_max - 0);
          output = (value - 0) * output;
          output = (1 + output);

          return output;
}

/***********************************************************************************************
* End of function scale_value
***********************************************************************************************/

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

5.2.5 r_cg_main.c へのコード挿入

関数 “void main (void)” 内に以下のコードを挿入してください。

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

    /* The rest of the code is executed in interrupt handlers */

    while (1U)
    {
        asm("nop");
    }
    /* End user code. Do not edit comment generated here */
```

関数 “void R_MAIN_UserInit (void)” 内に以下のコードを挿入してください。

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

    uint32_t delay = 0x3FFFF;

    /* Clear the switches' interrupt flags before enabling the interrupts */
    VIC.PIC0.LONG = 0x00000200UL;
    VIC.PIC0.LONG = 0x00010000UL;

    /* Enable the switch interrupts */
    R_ICU_IRQ12_Start();

    /* Enabling interrupts can cause generation of an interrupt which should
       be ignored. Allow some delay to catch the interrupt should it occur. */
    while ((0 == g_switch_press) && (delay--))
    {
        asm("nop");
    }

    /* Ensure the switch pressed flag is cleared to enable timer period updates */
    g_switch_press = 0;

    /* Enable continuous A/D conversions */
    R_S12AD0_Start();

    /* Enable the timer's count */
```

```
    R_CMT1_Start();
    /* End user code. Do not edit comment generated here */
```

5.3 追加のインクルードパス

プロジェクトをビルドするために、コンパイラに対しインクルードパスを追加する必要があります。「プロジェクト・ツリー」ウィンドウで CG_Tutorial プロジェクトを選択してください。ツールバーの ボタンを使用してプロジェクト設定を開きます。**図 5.2** に示すように、「C/C++ ビルド＞設定＞ツール設定＞ Compiler ＞ソース」の順に操作し、 ボタンをクリックします。

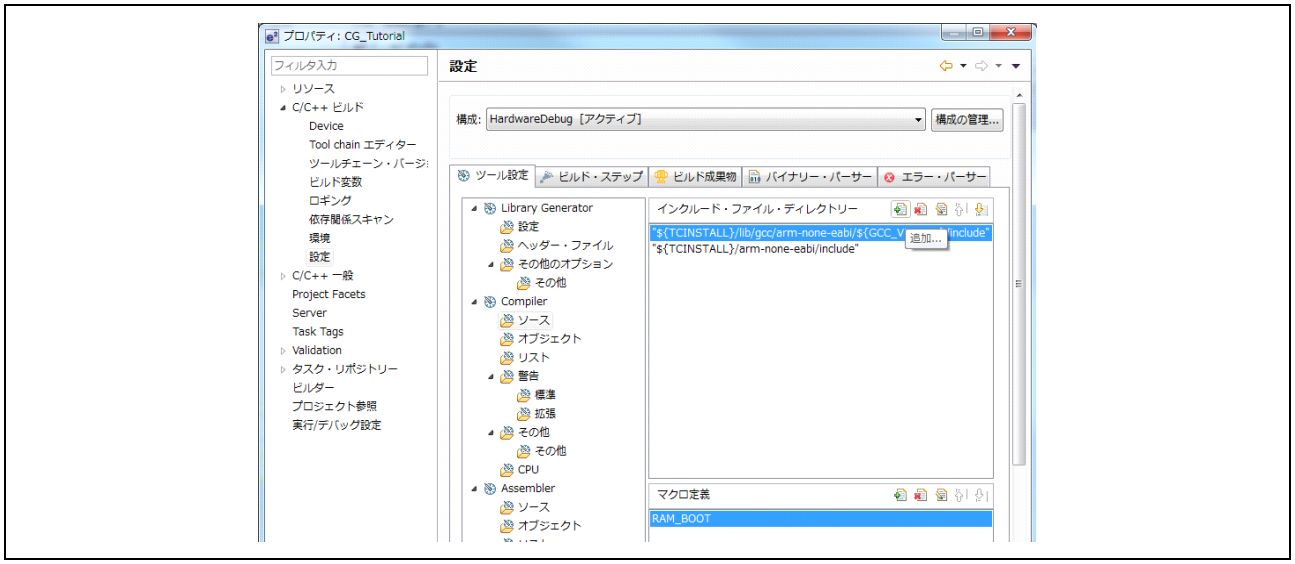


図 5.2 追加サーチパスの追加

「ディレクトリー・パスの追加」ダイアログ内の「ワークスペース」ボタンをクリックすると、「フォルダ選択」ダイアログが表示されます。ここで「CG_Tutorial/src」のフォルダ（ディレクトリー・パス）を選択し、「OK」をクリックすると、**図 5.3** に示すディレクトリー・パスが表示されます。

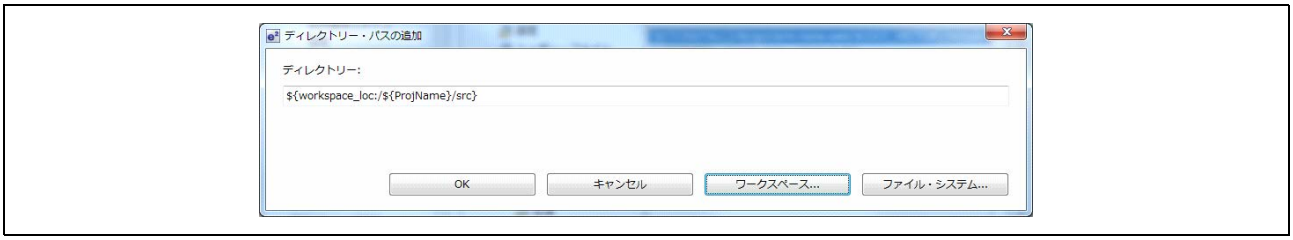


図 5.3 ディレクトリー・パスの追加

インクルードパスをさらに追加するには、上記手順を繰り返してください。「プロパティ」ダイアログを終了するには「OK」をクリックします。

「プロジェクト」メニューから「プロジェクトのビルド」を選択または ボタンを使用して、プロジェクトをビルドします。これにより、第 7 章に記載するデバッガを用いたプロジェクトの実行が可能になります。

5.4 Release ビルドセクションマップ

コード生成支援ツールは、コード生成中に Linker Section アドレスを変更します。この変更は、現在選択中のビルド構成にのみ適用されます。

上記手順を行うことにより、作業を行う「HardwareDebug」ビルド構成を作成します。「Release」ビルド構成を作成するには以下の手順を行ってください。ビルド構成の詳細は、第 2 章を参照してください。

以下の順にクリックし、「Release」ビルド構成を選択してください。

「プロジェクト＞ビルド構成＞アクティブにする＞ Release」

「プロジェクト・エクスプローラー」ツリーにおいて、「コード生成」エントリを右クリックし、「コードを生成する」を選択してください。これにより、生成されたコードが更新され、Linker Section アドレスに必要な変更がなされます。

以下から「Release」構成を開きます。
「デバッグ構成＞ Renesas GDB Hardware Debugging . CG_Tutorial Release」

「Startup」タブを選択します。オプション「ブレークポイント設定先：」のチェックマークが外れていることを確認してください。

注 . 第 6 章の内容は、両方のビルド構成に対し実施する必要があります。

6. 外部リンカファイル

e2 studio では、リンカが使用するリンカファイルを変更することができます。デフォルトのリンカマップは、以下にあります。

「プロジェクトのプロパティ＞ C/C++ ビルド＞設定＞ツール設定＞ Linker ＞セクション」

CG_Tutorial プロジェクトでは、デフォルトのリンカマップを利用しません。loader_init.asm ファイルは、デフォルトのリンカマップで指定されたセクション変数を使用します。これらの変数は、リンカファイル内の特定のアドレスを格納するために使用されます。loader_init.asm を開き、#if 0 になっていることを確認してください。

6.1 リンカファイルの上書き

以下の手順でリンカファイルを新規に作成し、RZ/T1 デバイスのリンカセクションを定義し、そのファイルを使用するための GNU リンカをセットしてください。

- プロジェクト内にファイルを新規作成する。
- ソースフォルダ src を右クリックする。
- 新規＞ファイルの順に選択する。
- 名前を指定する：linker_file.ld
- ファイルをダブルクリックして開く。
- 以下のテキストをコピーして貼り付ける。

```
/********************************** ここからコピー ************************************/

OUTPUT_FORMAT("elf32-littlearm", "elf32-bigarm", "elf32-littlearm")
OUTPUT_ARCH(arm)
ENTRY(_PowerON_Reset)

MEMORY
{
   /* Internal RAM address range H'2000_0000 to H'2001_FFFF is configured as data retention RAM */
   /* Write access to this address range has to be enabled by writing to registers SYSCR1 and SYSCR2 */
   ATCM      (rwx)   : ORIGIN = 0x00000000, LENGTH = 0x00080000  /* (512KB)  H'00000000 to H'0007FFFF */
   BTCM      (rwx)   : ORIGIN = 0x00800000, LENGTH = 0x00800000  /* (32KB)   H'00800000 to H'00807FFF */
   BUFFER_RAM (rwx)   : ORIGIN = 0x08000000, LENGTH = 0x10000000  /* (128MB) H'08000000 to H'10000000  */
   DATA_RAM   (rwx)   : ORIGIN = 0x20000000, LENGTH = 0x00080000  /* (512KB)  H'20000000 to H'2007FFFF */

   /* Mapped memory type */
   SPI_ROM    (rw)    : ORIGIN = 0x30000000, LENGTH = 0x04000000
   CS0_ROM    (rw)    : ORIGIN = 0x40000000, LENGTH = 0x04000000
   CS1_ROM    (rw)    : ORIGIN = 0x44000000, LENGTH = 0x04000000
   SDRAM0_EXT (rw)    : ORIGIN = 0x48000000, LENGTH = 0x04000000
   SDRAM1_EXT (rw)    : ORIGIN = 0x4C000000, LENGTH = 0x04000000
}

SYS_STACK_SIZE       = 0x200;     /* Application stack size          */
SVC_STACK_SIZE       = 0x200;     /* SVC mode stack                  */
IRQ_STACK_SIZE       = 0x100;     /* IRQ mode stack                  */
FIQ_STACK_SIZE       = 0x100;     /* FRQ mode stack                  */
UND_STACK_SIZE       = 0x100;     /* SVC mode stack                  */
ABT_STACK_SIZE       = 0x100;     /* ABT mode stack                  */
HEAP_STACK_SIZE      = 0x1000;    /* Heap stack size                 */

ATCM_BASE           = 0x00000000;  /* User application located here        */
BTCM_BASE           = 0x00800000;  /* BTCM base address                    */
USER_EXEC_BASE      = 0x00000000;  /* Application loads and runs from here  */
```

```
USER_RAM            = 0x20000000;  /* Application's RAM base              */
STACK_BASE          = 0x00807800;  /* Stacks located in BTCM              */

SECTIONS
{
   .loader_text USER_EXEC_BASE :
   {
      reset_start = .;
      *(.loader_text);
      .   = ALIGN(0x4);
      reset_end = .;
   } > ATCM

   .text :
   {
      text_start = .;
      *(.text)
      *(.text.startup)
      text_end = .;
   } > ATCM

   .rodata :
   {
      rodata_start = .;
      _start_data_ROM = .;
      *(.rodata)
      *(.rodata.*)
      .   = ALIGN(0x8);
      *(.data)
      *(.data.*)
      _end_data_ROM = .;
      *(.got.plt)
      *(.got)
      .   = ALIGN(0x8);
      rodata_end = .;
      PROVIDE(end = .);
   } > ATCM

   _ram_data_size = (_end_data_ROM - _start_data_ROM);

   .data USER_RAM :
   {
      _start_data_RAM = .;
      data_start = .;
      start_data_RAM = .;
      . += _ram_data_size;
      data_end = .;
   }

   .bss data_end :
   {
       bss_start = .;
       PROVIDE(__bss_start__ = .);
       *(.bss)
       *(.bss.**)
       *(COMMON)
       . = ALIGN(0x4);
       PROVIDE(__bss_end__ = .);
       ebss_end = .;
       _end = .;
       PROVIDE(end = .);
   }

   .heap :
   {
      heap_start = .;
```

```
        .   = ALIGN(0x8);
        *(.heap_stack)
        . += HEAP_STACK_SIZE;
        heap_end = .;
    } > ATCM

    .sys_stack 0x807800 : AT (0x807800)
    {
        sys_stack_start = .;
        .   = ALIGN(0x8);
        *(.sys_stack)
        . += SYS_STACK_SIZE;
        sys_stack_end = .;
        _sys_stack = .;
    } > BTCM

    .svc_stack 0x807A00 : AT (0x807A00)
    {
        svc_stack_start = .;
        .   = ALIGN(0x8);
        *(.svc_stack)
        . += SVC_STACK_SIZE;
        svc_stack_end = .;
        _svc_stack = .;
    } > BTCM

    .irq_stack 0x807C00 : AT (0x807C00)
    {
        irq_stack_start = .;
        .   = ALIGN(0x8);
        *(.irq_stack)
        . += IRQ_STACK_SIZE;
        irq_stack_end = .;
        _irq_stack = .;
    } > BTCM

    .fiq_stack 0x807D00 : AT (0x807D00)
    {
        fiq_stack_start = .;
        .   = ALIGN(0x8);
        *(.fiq_stack)
        . += FIQ_STACK_SIZE;
        fiq_stack_end = .;
        _fiq_stack = .;
    } > BTCM

    .und_stack 0x807E00 : AT (0x807E00)
    {
        und_stack_start = .;
        .   = ALIGN(0x8);
        *(.und_stack)
        . += UND_STACK_SIZE;
        und_stack_end = .;
        _und_stack = .;
    } > BTCM

    .abt_stack 0x807F00 : AT (0x807F00)
    {
        abt_stack_start = .;
        .   = ALIGN(0x8);
        *(.abt_stack)
        . += ABT_STACK_SIZE;
        abt_stack_end = .;
        _abt_stack = .;
    } > BTCM
}
```

/********** ここまでをコピー **********/

ファイル＞保存の順にクリックします。

- プロジェクトのプロパティ＞ C/C++ ビルド＞設定＞ Linker ＞その他の順に開く。
- オプション「コマンドファイルの上書き」で「External Linker script(-T)」を選択する。
- 「ファイル」エントリに以下を追加する（ダブルクォーテーションマークを含む)。

"${workspace_loc:/${ProjName}/src/linker_file.ld}"

- 「適用」をクリックする。
- 以下の順に操作する：プロジェクトのプロパティ＞ C/C++ ビルド＞設定＞ Linker ＞その他＞その他
- 「未使用の入力セクションのガベージコレクションを有効にする (-gc-sections)」にチェックマークが入っている場合は、かならずチェックマークを外す。
- 「OK」をクリックする。

上記 7 つの手順は、「HardwareDebug」構成と「Release」構成の両方に必要です。

6.2 プロジェクトの構成

ここまでに、コード生成支援ツールで作成されたプロジェクトテンプレートをビルドするための準備が整いました。「プロジェクト・エクスプローラー」ウィンドウで、src フォルダを展開してください。

「プロジェクト」メニューから「プロジェクトのビルド」を選択または  ボタンを使用して CG_Tutorial プロジェクトをビルドしてください。

7. プロジェクトの実行

「プロジェクト・エクスプローラー」ウィンドウで、「CG_Tutorial」プロジェクトが選択されていることを確認します。 ボタンをクリックしてプロジェクトをデバッグしてください。図 7.1 に示すダイアログが表示されます。

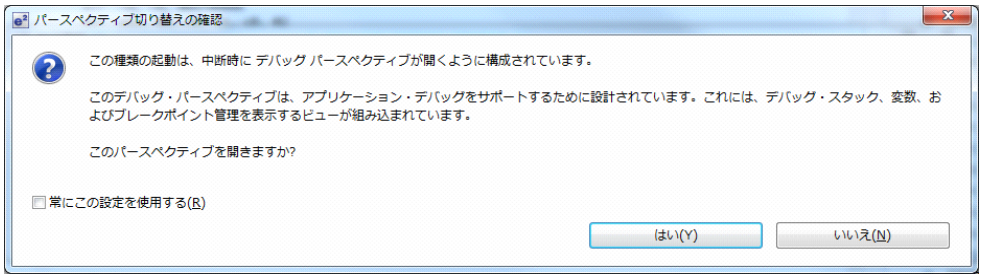


図 7.1 パースペクティブ切り替えダイアログ

「OK」をクリックし、デバッグウィンドウのパースペクティブが使用されることを確認してください。

デバッガが起動し、e2 studio はコード生成の PowerOn_Reset() 関数を表示します。

「再開」ボタン をクリックします。main() 関数の先頭でデバッガが再び停止します。

を再度クリックし、コードを実行してください。

プログラムを実行すると、LED0 が RV1 の回転位置により設定された時間間隔で点滅します。RV1 を時計回りにゆっくりと全回転させてから半時計回りに全回転させ、LED 点滅間隔が変化することを確認します。SW3 を押下すると、その時点の RV1 回転位置での LED 点滅間隔が保持されます。この状態で RV1 を回しても、LED 点滅間隔は変化しません。再度 SW3 を押下すると、RV1 回転位置による LED 点滅間隔の変化が有効になります。

e2 studio デバッガの詳細は、チュートリアルマニュアルを参照してください。

8. ご使用上の注意

8.1 iodefine.h ファイル

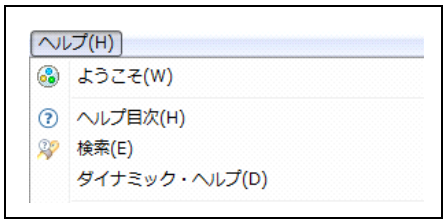
iodefine.h ファイルの場所

デフォルトでは、ヘッダファイル r_cg_macrodriver.h は、インクルードする iodefine.h ファイルを src フォルダから参照します。

9. 追加情報

技術サポート

e2 studio の使用方法の詳細は、e2 studio のメニューバーから「ヘルプ＞ヘルプ目次」の順に選択し、ヘルプファイルを参照してください。



RZ/T1 グループの詳細については、RZ/T1 グループ ユーザーズマニュアル ハードウェア編（R01UH0483JJ）を参照してください。

技術サポート窓口

『クイックスタートガイド』の 11 章に記載の窓口情報をご覧ください。

ルネサス製品に関する総合情報は以下の Web サイトよりご覧ください。
http://japan.renesas.com/

商標

本マニュアルで使用するすべての商標名および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	RZ/T1 グループ Renesas Starter Kit+ コード生成支援ツールチュートリアルマニュアル e^2 studio 版

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2016.09.26	—	初版発行

RZ/T1 グループ Renesas Starter Kit+
コード生成支援ツールチュートリアルマニュアル e2 studio 版

発行年月日　2016 年 09 月 26 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）

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 〒135-0061 東京都江東区豊洲3-2-24（豊洲フォレシア）

■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/

Colophon 3.0

RZ/T1 グループ

ルネサス エレクトロニクス株式会社

R20UT3281JG0100