RH850 スマート・コンフィグレータ
ユーザーガイド: e² studio 編

R20AN0739JC0100
Rev.1.00
Jan.20.2024

要旨

本アプリケーションノートでは、e² studio のプラグインツールである RH850 スマート・コンフィグレータ（以下、スマート・コンフィグレータと略す）の基本的な使用方法について説明します。

統合開発環境 e² studio の対象バージョンは以下の通りです。

・e² studio 2024-01 以降

対象デバイス/対応コンパイラ

サポートしているデバイス及びコンパイラは、以下の URL をご参照ください。

https://www.renesas.com/rh850-smart-configurator

目次

1. 概要

1.1 目的

本アプリケーションノートは、統合開発環境 e² studio でスマート・コンフィグレータを使用したプロジェクトの作成、基本的な使用方法について説明しています。

e² studio の使い方は、e² studio のユーザーズマニュアルを参照してください。

1.2 特長

スマート・コンフィグレータは、「ソフトウェアを自由に組み合わせられる」をコンセプトとしたユーティリティです。ドライバコード生成、端子設定の機能でお客様のシステムへのルネサス製ドライバの組み込みを容易にします。タイミング波形などのスマート・コンフィグレータのグラフィカルな表示により、ドライバの設定が簡単になります。

2. プロジェクトの作成

スマート・コンフィグレータを使用して C/C++プロジェクトを生成する手順を、以下に説明します。

e² studio のプロジェクト作成ウィザードの詳細は、e² studio の関連ドキュメントを参照してください。

(1) e² studio を起動し、ワークスペースを指定します。起動後、［ファイル］→［新規］→［Renesas C/C++プロジェクト］→［Renesas RH850］の順に選択してプロジェクト作成ウィザードを開きます。

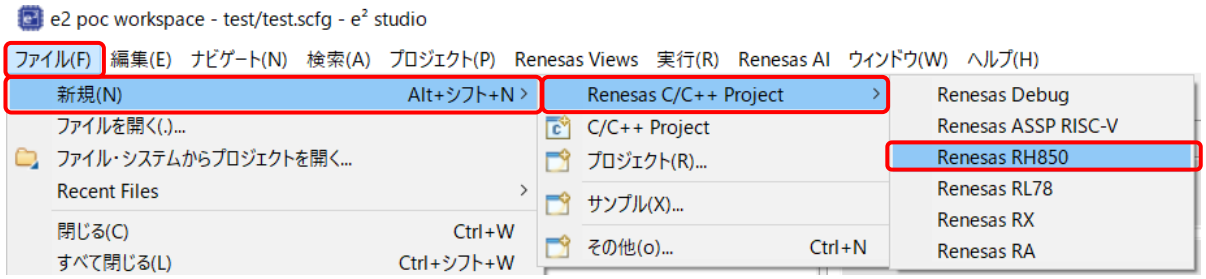


図 2-1 新規プロジェクトの作成

(2) プロジェクト作成ウィザードで、［Renesas RH850］→［Renesas CC-RH850 C/C++ Executable Project］を選択し、［次へ］ボタンをクリックします。

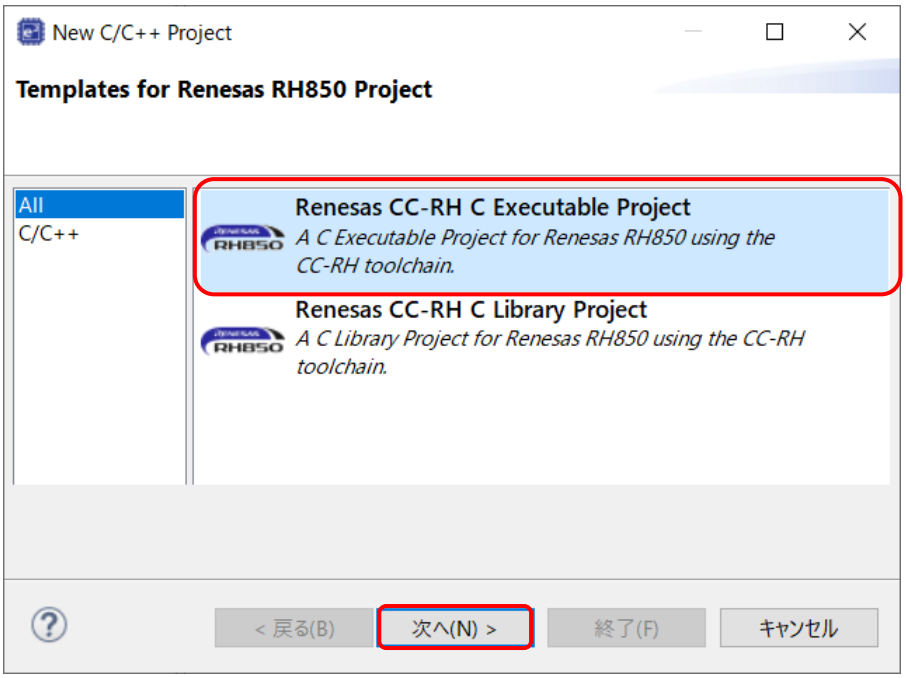


図 2-2 新規 C/C++プロジェクトのテンプレート

(3) プロジェクト名を入力し、［次へ］ボタンをクリックして次に進みます。

（例：CC-RH executable project，プロジェクト名：“Smart_Configurator_Example”）

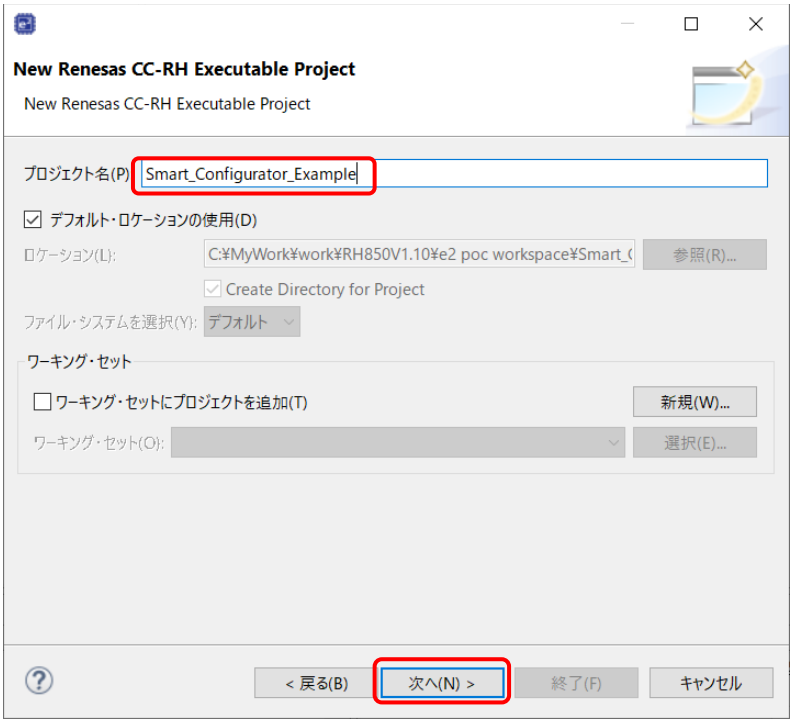


図 2-3 New Renesas CC-RH executable project の作成

(4) デバイスおよびデバッグ設定を選択し、[次へ] をクリックします。（例：Target Device: RH850F1KM – 272 pins）

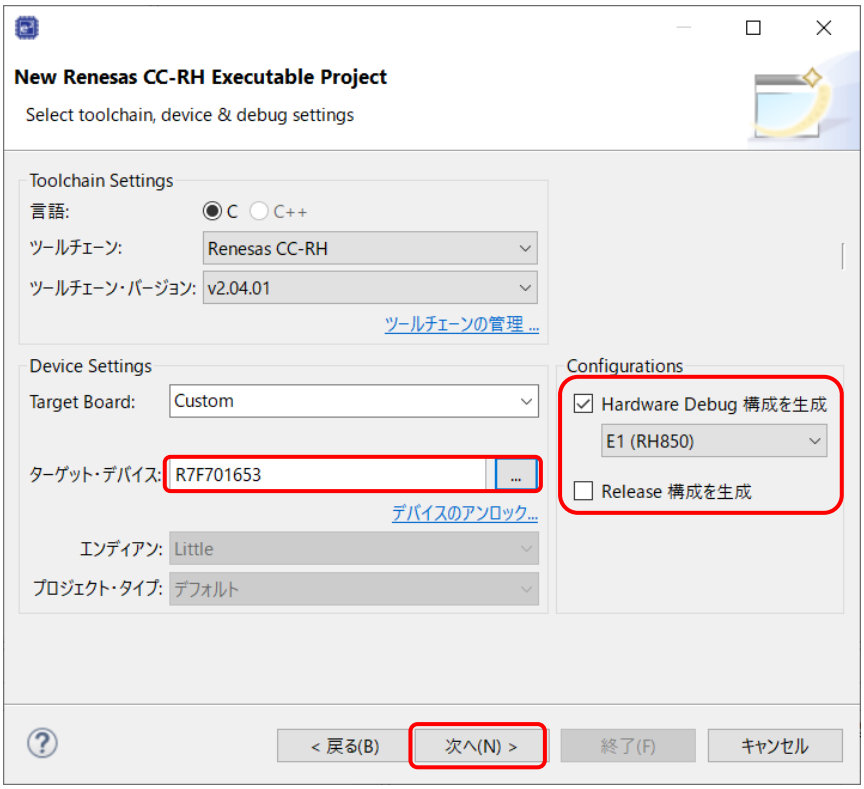


図 2-4 ツールチェーン、デバイス、デバッグ設定の選択

(5) ［コーディング・アシスタントツールの選択］ダイアログボックスで、［Use Smart Configurator］のチェックボックスを選び、［終了］をクリックします。

【注】 (4)で、スマート・コンフィグレータが対応しているデバイスを選択時のみ、［スマート・コンフィグレータを使用する］のチェックボックスが選択可能になります。

図 2-5 コーディング・アシストツールの選択

(6) プロジェクト作成の完了を待ちます。

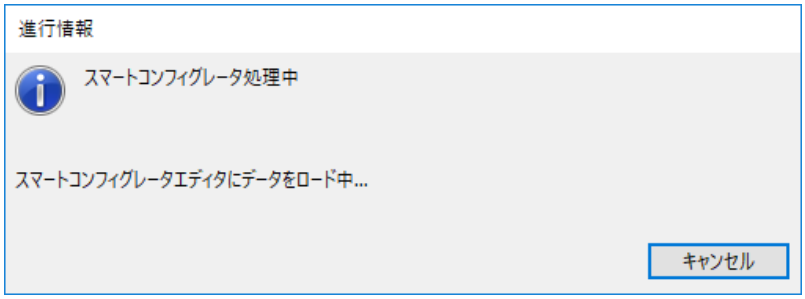


図 2-6 プロジェクト作成の処理

(7) 新規 C/C++プロジェクトの作成が成功すると、作成したプロジェクトがスマート・コンフィグレータ・パースペクティブ上で開きます。

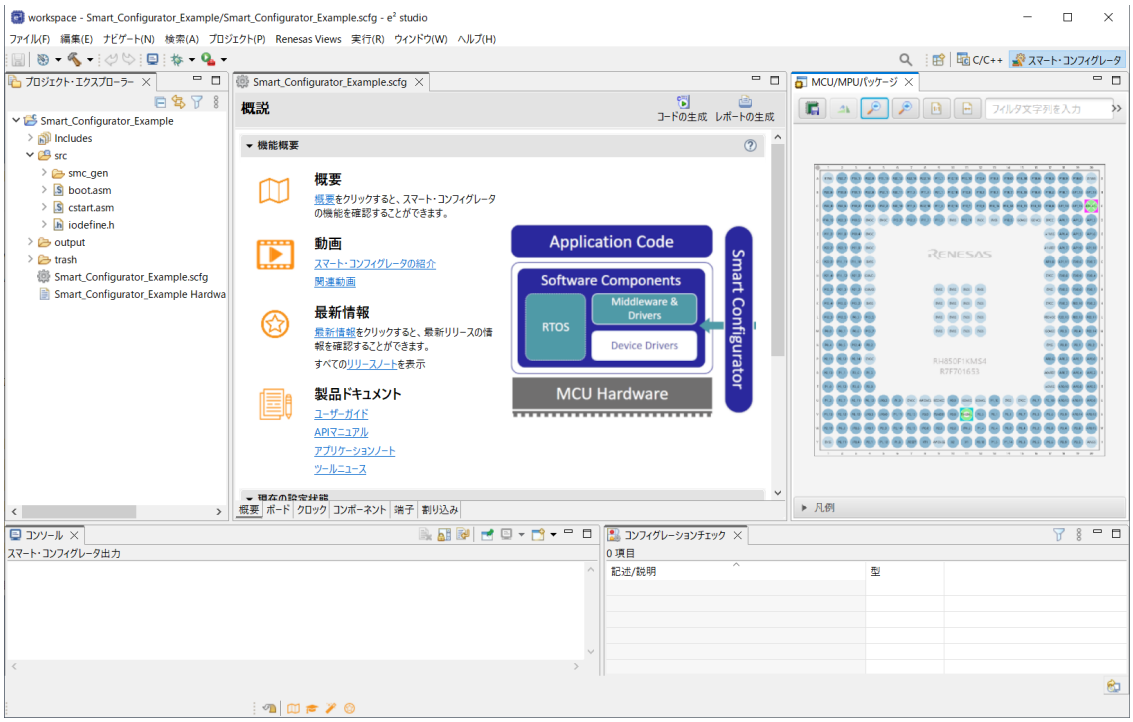


図 2-7 ［スマート・コンフィグレータ］パースペクティブ

3. スマート・コンフィグレータの操作方法

3.1 スマート・コンフィグレータの表示

スマート・コンフィグレータの機能を十分に活用するためには、スマート・コンフィグレータ・パースペクティブを確実に開いていることが必要です。開いていない場合は、e² studio ウィンドウ右上角のパースペクティブを選択してください。

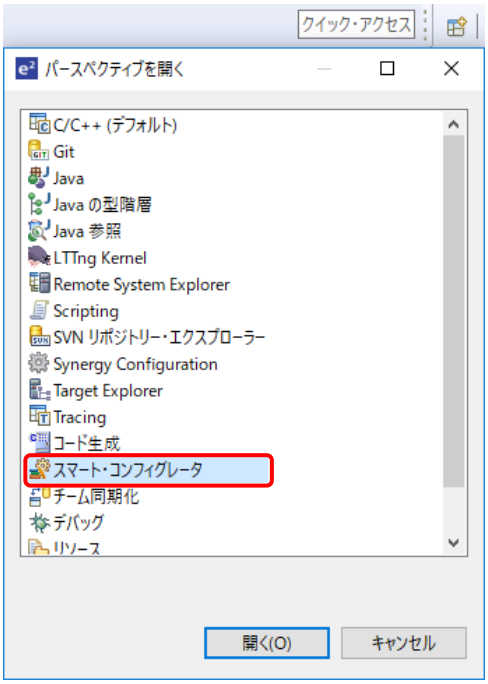


図 3-1 ［スマート・コンフィグレータ］パースペクティブを開く

3.2 操作手順

e² studio 上のスマート・コンフィグレータで周辺機能の設定し、ビルドするまでの手順を**図 3-2 操作手順**に示します。e² studio の操作については、e² studio の関連ドキュメントを参照してください。

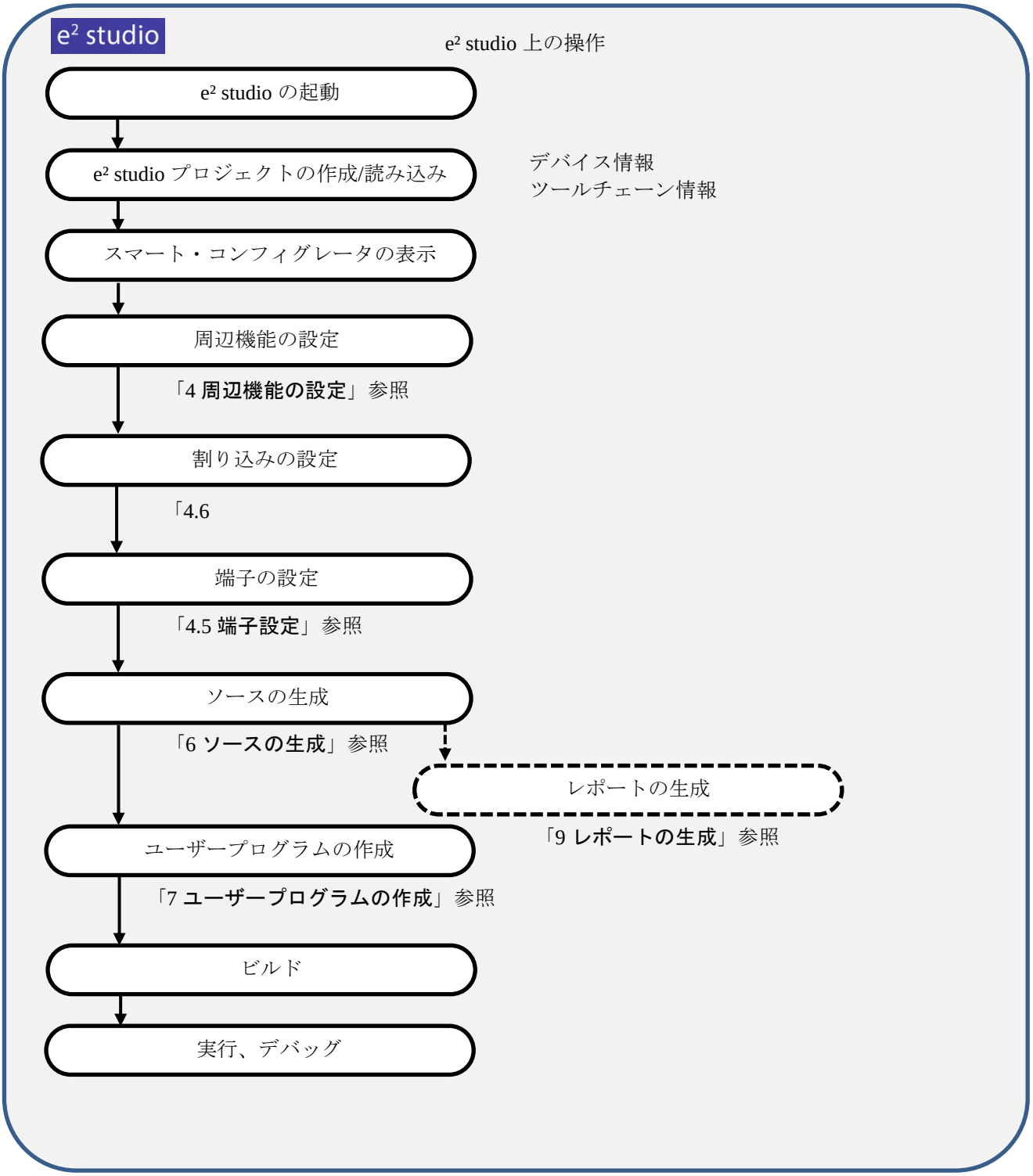


図 3-2 操作手順

3.3　プロジェクト情報の保存先

スマート・コンフィグレータは、プロジェクトで使用するマイクロコントローラ、ビルド・ツール、周辺機能、端子機能などの設定情報をプロジェクト・ファイル(*.scfg)に保存し、参照します。

スマート・コンフィグレータのプロジェクト・ファイルは、e² studio のプロジェクト・ファイル(.project)と同階層にある“プロジェクト名.scfg”に保存します。

3.4　ウィンドウ

［スマート・コンフィグレータ］パースペクティブの構成を図 3-3 ［スマート・コンフィグレータ］パースペクティブに示します。

図 3-3 ［スマート・コンフィグレータ］パースペクティブ

① プロジェクト・エクスプローラー

② スマート・コンフィグレータビュー

③ MCU/MPU パッケージビュー

④ コンソールビュー

⑤ コンフィグレーションチェックビュー

3.4.1　プロジェクト・エクスプローラー

プロジェクトのフォルダ構成をツリーで表示します。

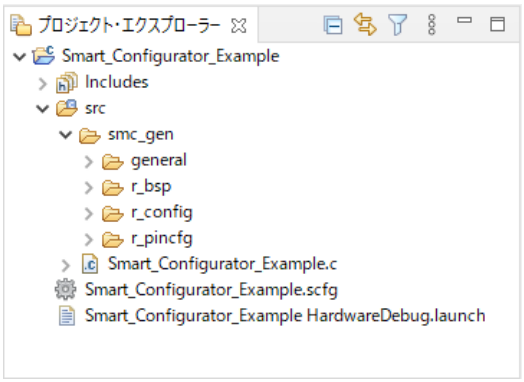


図 3-4 プロジェクト・エクスプローラー

ビューが開いていない場合は、e² studio メニュー上の［ウィンドウ］→［ビューの表示］→［その他］を選択し、開いた［ビューの表示］ダイアログボックスから［一般］→［プロジェクト・エクスプローラー］を選択してください。

3.4.2　スマート・コンフィグレータビュー

［概要］，［ボード］，［クロック］，［システム］，［コンポーネント］，［端子］，［割り込み］の 7 つのページから構成されます。タブをクリックして、ページを選択すると選択したタブに応じて内容が切り替わります。

図 3-5 スマート・コンフィグレータビュー

ビューが開いていない場合は、［プロジェクト・エクスプローラー］からプロジェクト・ファイル(*.scfg)を右クリックし、コンテキストメニューから［開く］を選択してください。

3.4.3　MCU/MPU パッケージビュー

MCU/MPU パッケージ図上に端子状態を表示します。端子設定を変更することもできます。

MCU/MPU パッケージビューは、[割り当てられた機能]、[ボード機能]、[シンボリック名]の 3 種類種類の切り替えが行えます。

・[割り当てられた機能]は、端子設定の割り当て状況を表示します。

・[ボード機能]は、ボードの初期端子設定情報を表示します。

・[シンボリック名]は、端子にユーザー定義したシンボリック名が表示されます。

シンボリック名のマクロ定義は、Pin.h ファイル内のポート読み取りまたは書き込み関数とともに生成されます。

ボードの初期端子設定情報は、［ボード］ページの［ボード:］で選択したボードの端子情報です（「4.1.2 ボード選択」、「4.5.6 ボード端子設定情報を使用した端子設定」参照）。

【注】シンボリック名の機能は、RH850/F1KM および RH850/F1KH には適用されません。

シンボリック名の機能は、APORT、JPORT、および IPORT には適用されません。

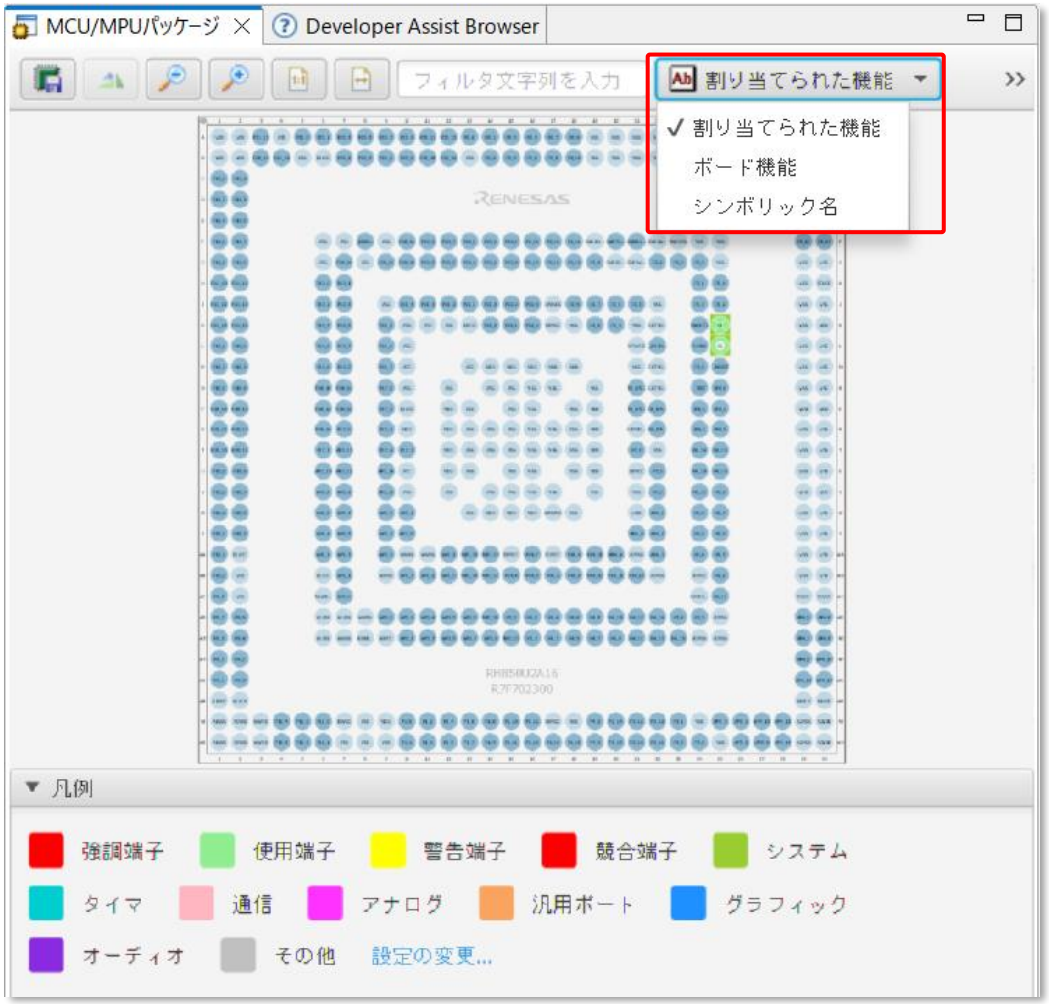


図 3-6 MCU/MPU パッケージビュー

ビューが開いていない場合は、e² studio メニュー上の［Renesas Views］→［スマート・コンフィグレータ］→［MCU/MPU パッケージ］を選択してください。

3.4.4 コンソールビュー

スマート・コンフィグレータビューまたは MCU/MPU パッケージビューでの設定変更内容が表示されます。

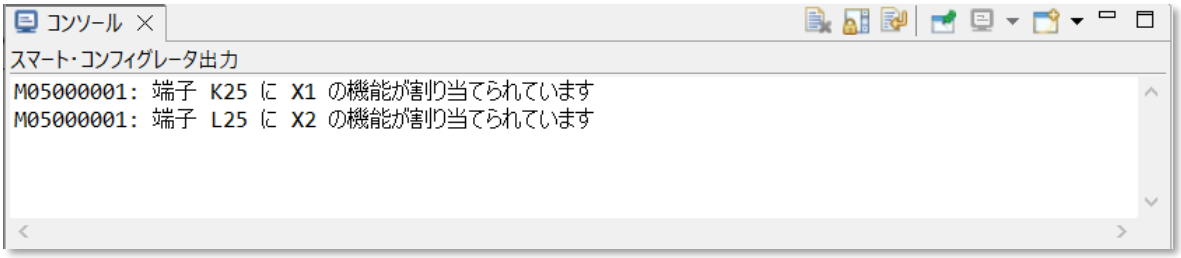


図 3-7 コンソールビュー

ビューが開いていない場合は、e² studio メニュー上の［ウィンドウ］→［ビューの表示］→［その他］を選択し、開いた［ビューの表示］ダイアログボックスから［一般］→［コンソール］を選択してください。

3.4.5 コンフィグレーションチェックビュー

端子競合が発生した場合に、その内容を表示します。

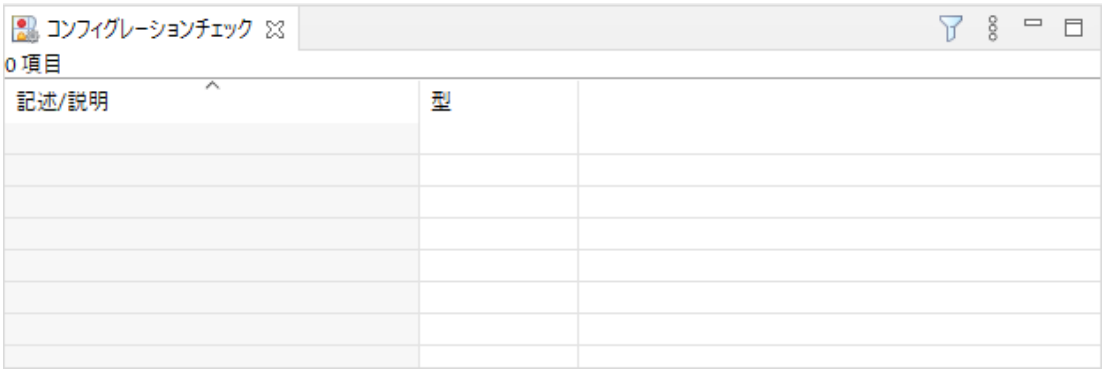


図 3-8 コンフィグレーションチェックビュー

ビューが開いていない場合は、e² studio メニュー上の［Renesas Views］→［スマート・コンフィグレータ］→［コンフィグレーションチェック］を選択してください。

4. 周辺機能の設定

周辺機能は、スマート・コンフィグレータビューから選択します。

4.1 ボード設定

ボードページでは、ボードおよび、デバイスの変更が可能です。

4.1.1 デバイス選択

[...] ボタンをクリックすると、デバイスが選択できます。

「4.7 MCU マイグレーション機能」の手順に従いデバイス変更を行ってください。

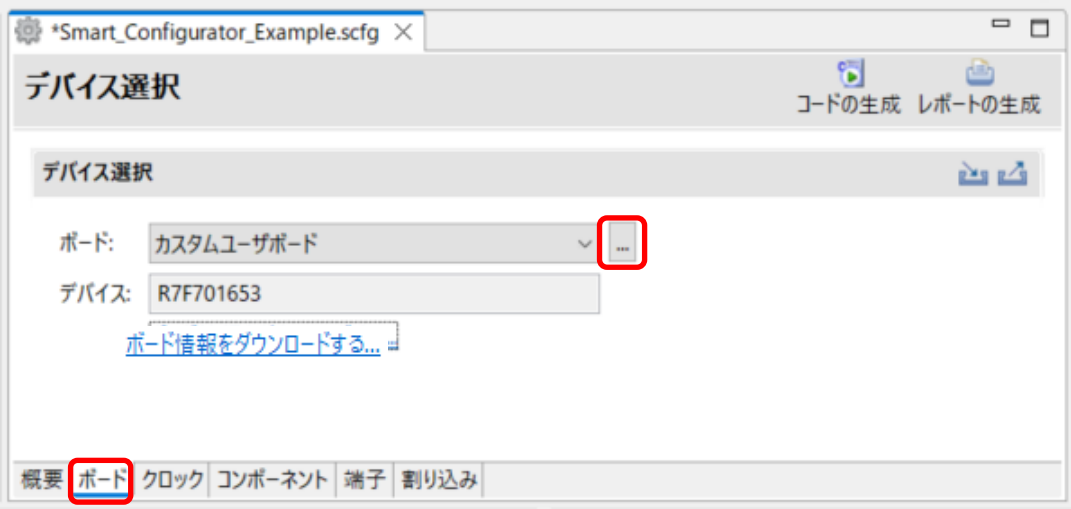


図 4-1 デバイス選択

4.1.2 ボード選択

[...] ボタンをクリックすると、ボードが選択できます。

ボード選択により、以下の一括変更が可能です。

- 端子割り当て（初期端子設定）
- メインクロック周波数
- サブクロック周波数
- デバイス
- オンチップ・デバッグ動作設定とエミュレータ設定

上記ボード設定情報は、Board Description File（.bdf）に定義されています。

Renesas 製ボード(Renesas Starter Kit 等)の.bdf ファイルを WEB からダウンロードし、インポートが可能です。

また、アライアンスパートナーが公開している.bdf ファイルを WEB からダウンロードし、インポートすることで、アライアンスパートナー製ボードの選択が可能となります。

選択したボードに応じて、デバイスが変更され、デバイスの変更は e² studio プロジェクトのターゲット・デバイスに反映されます。「4.7 MCU マイグレーション機能」の手順を参照してください。

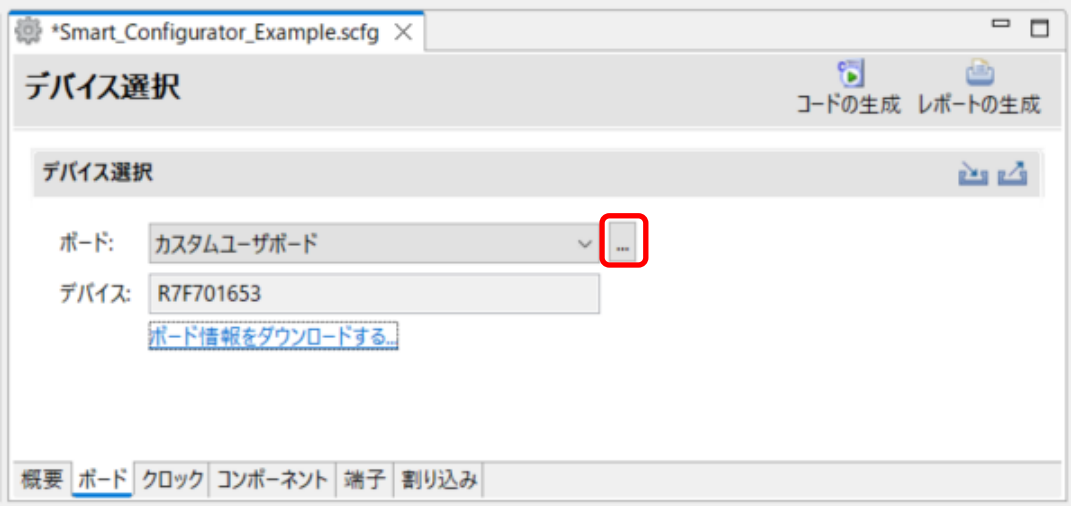


図 4-2 ボード選択

[検出された問題] に表示されたメッセージを確認して [次へ] をクリックします。

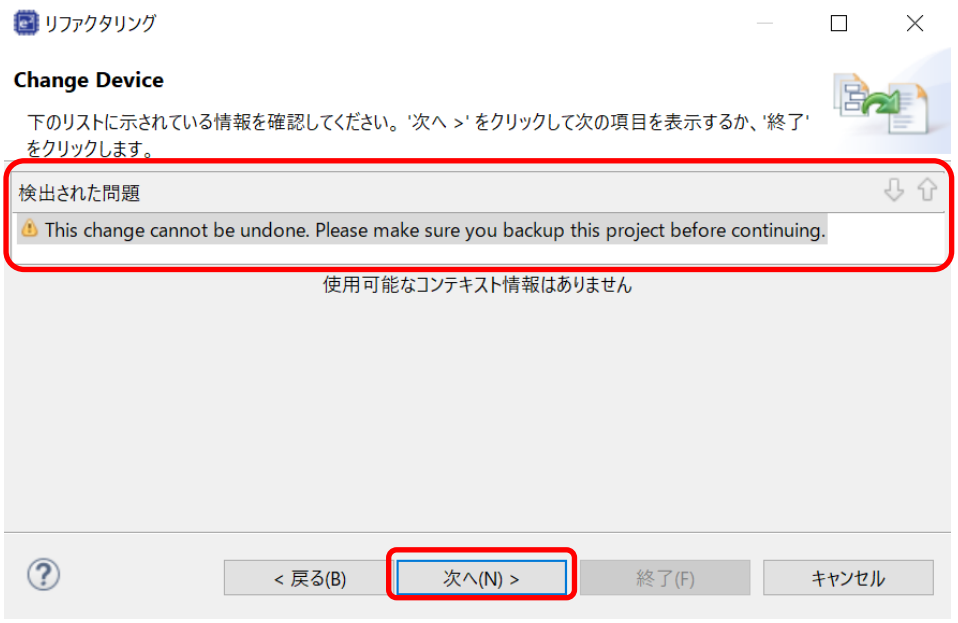


図 4-3 検出された問題

表 4-1 デバイス変更の[検出された問題]の表示一覧

メッセージ	説明
This change cannot be undone. Please make sure you backup this project before continuing.	デバイスを変更すると変更前に復元できませんので、プロジェクトのバックアップ後に実行してください。

[実行される変更] で、変更する項目を選択して [終了] をクリックします。

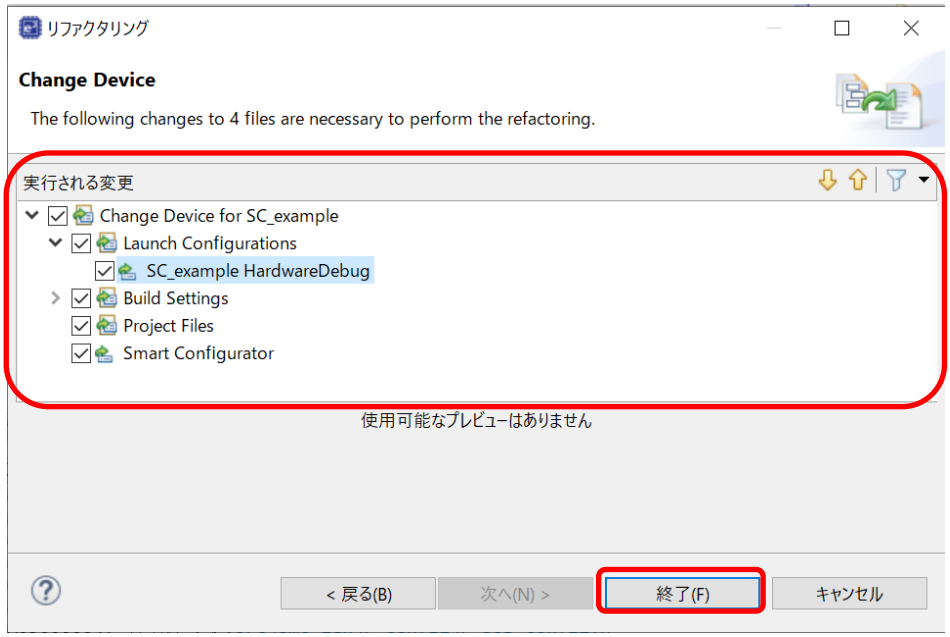


図 4-4 実行される変更項目確認

4.1.3 ボード設定のエクスポート

ボード設定のエクスポートは、以下の手順で行います。

(1) ボードページで、［ボードの設定をエクスポート］ ボタンをクリックします。

(2) 出力場所を選択し、エクスポートするファイル名(表示名)を入力します。

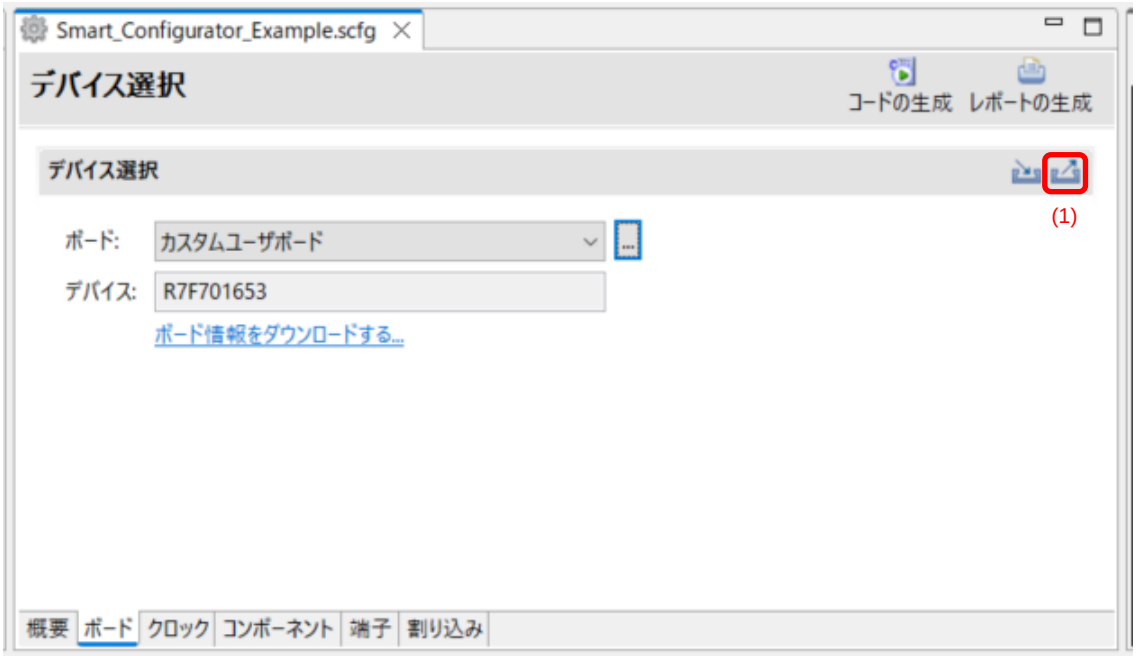


図 4-5 ボード設定のエクスポート(bdf 形式)

4.1.4　ボード設定のインポート

ボード設定のインポートは、以下の手順で行います。

(1) ［ボードの設定をインポート］ボタンをクリックし、bdf ファイルを選択してください。

(2) インポートしたボード設定がボード選択の選択肢に追加されます。

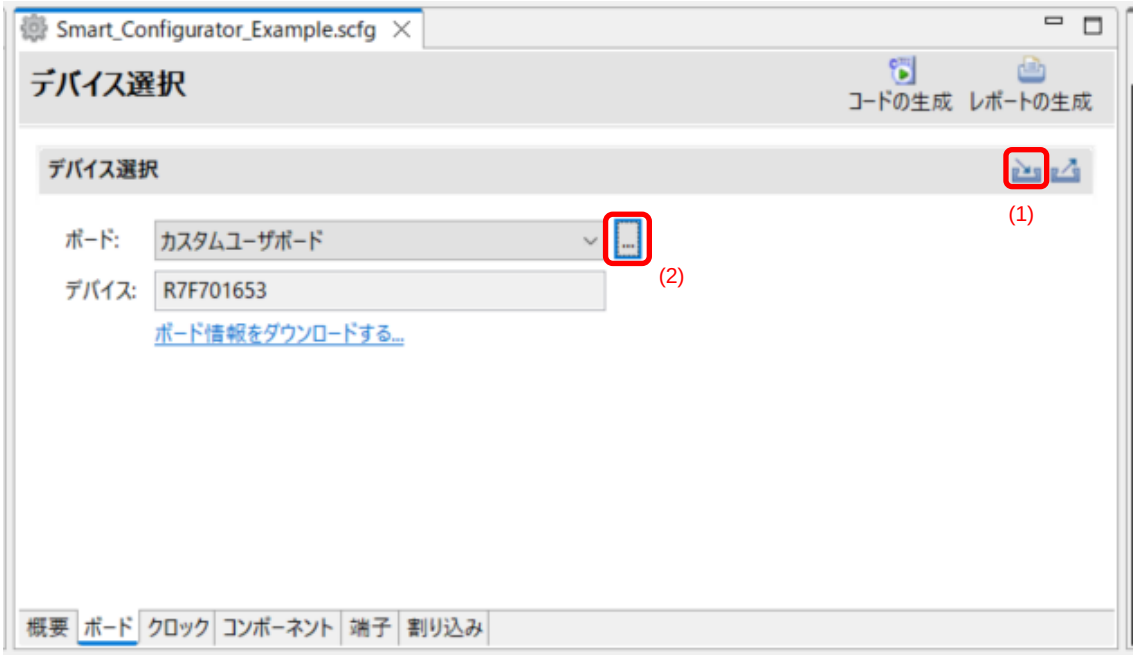


図 4-6 ボード設定のインポート(bdf 形式)

一度インポートしたボード設定は、同じデバイスグループのプロジェクトにおいてボード選択の選択肢に表示されます。

4.2　クロック設定

クロックページでは、システムクロックを設定することができます。クロックページで作成した設定は、全てのドライバおよびミドルウェアで使用されます。

クロック設定を更新するには、以下の手順で行います。

(1) ボード仕様に従って各クロックの周波数を指定します（ただし、一部の内部クロックは周波数が固定です）。

(2) PLL 回路を使用する場合、PLL のクロックソースを選択します。

(3) マルチプレクサの出力クロックのソースを選択します。

(4) ドロップダウンリストで分周比を設定します。

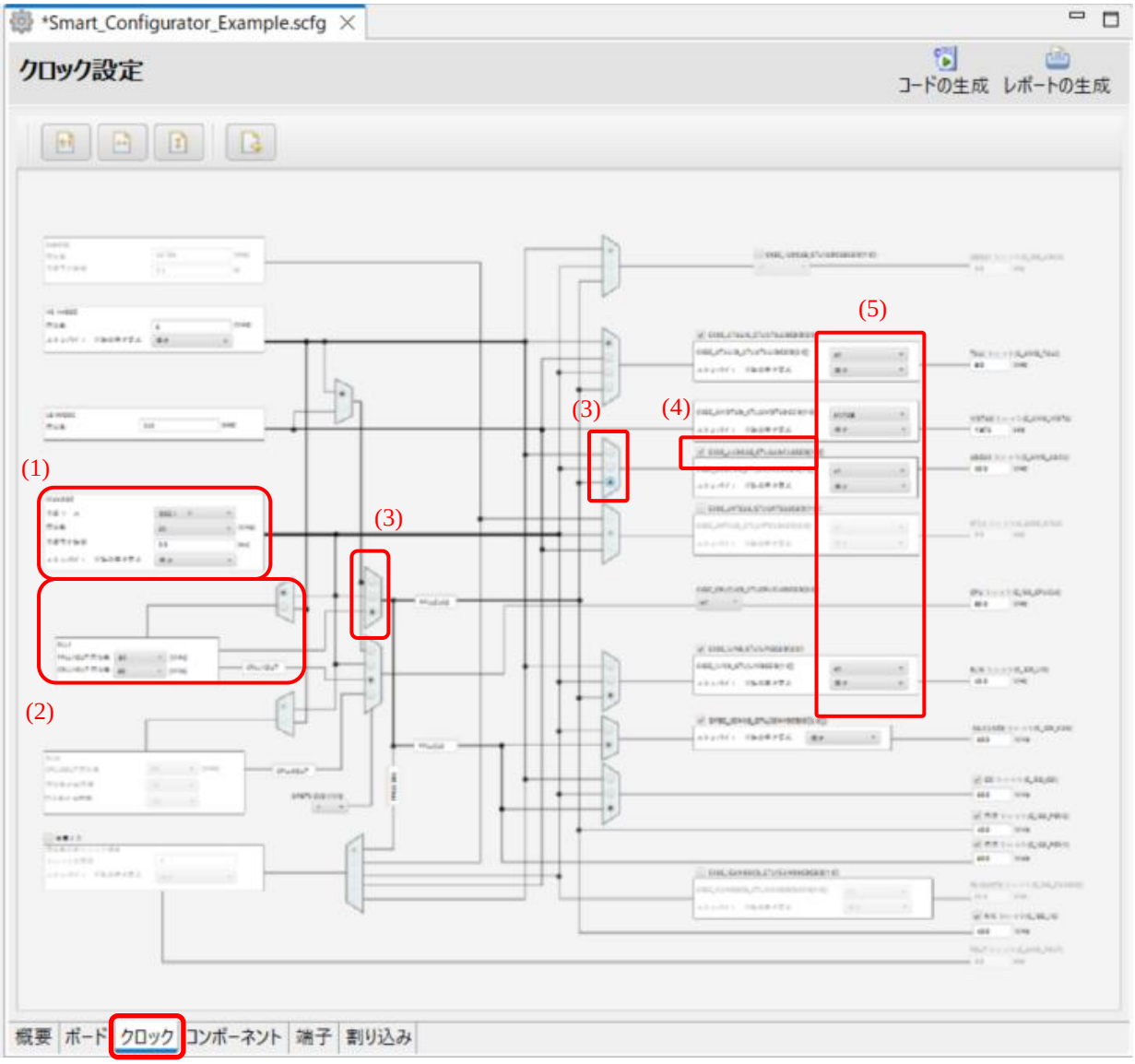


図 4-7 クロック設定

4.3 システム設定(RH850/U2A のみ)

[システム]タブで使用する CPUn(PEn)を選択します。

CPU0(PE0)は常にデフォルト設定として選択されます。

システム設定は、RH850/U2A のみサポートします。

図 4-8 システム設定

たとえば、CPU0(PE0)と CPU1(PE1)を使用する場合、以下の設定が必要です。

(1) スマート・コンフィグレータの[システム]ページをクリックし、CPU0(PE0)と CPU1(PE1)を選択します。 コードを生成します。

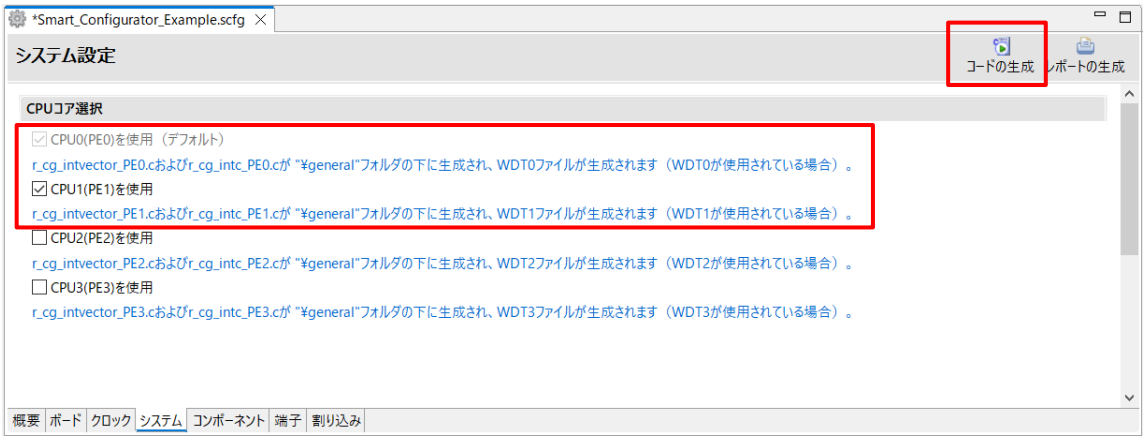


図 4-9 CPU コア選択

(2) 「cstart_pm0.asm」ファイルを新しいファイル「cstart_pm1.asm」としてコピーします。

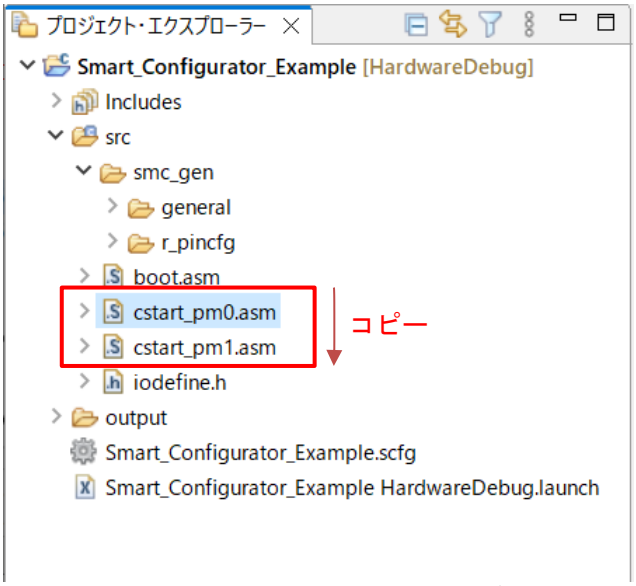


図 4-10 cstart_pm0.asm のコピー

(3) 「cstart_pm1.asm」ファイルの内容を以下のように変更します。

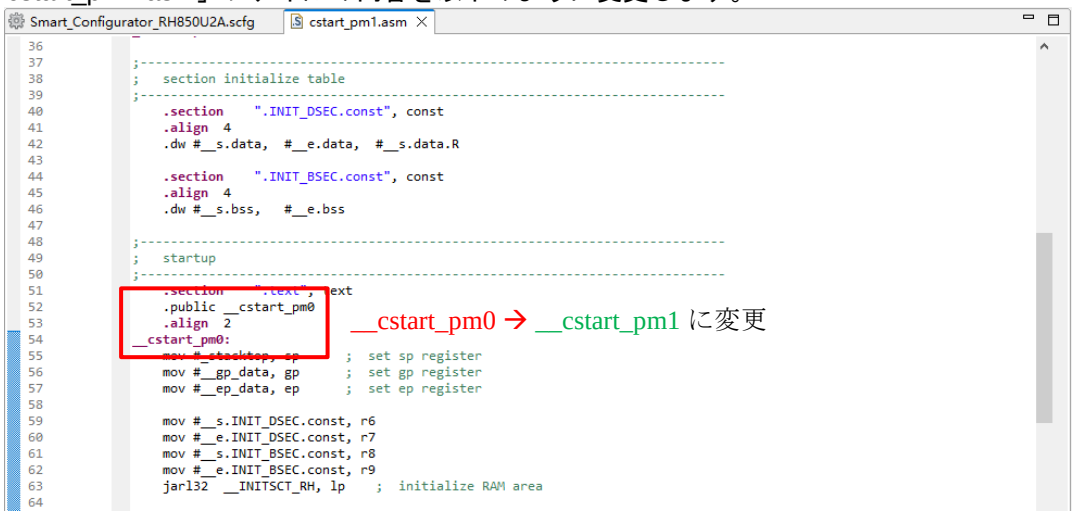


図 4-11 cstart_pm1.asm の内容変更

(4) 「boot.asm」ファイルの内容を以下のように変更します。

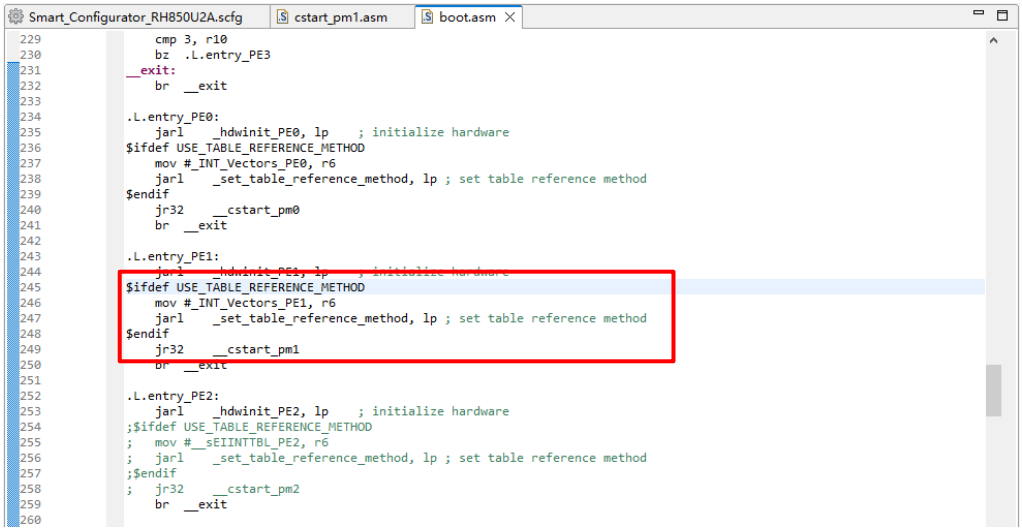


図 4-12 boot.asm の内容変更

(5) PE1 のセクションを設定します。

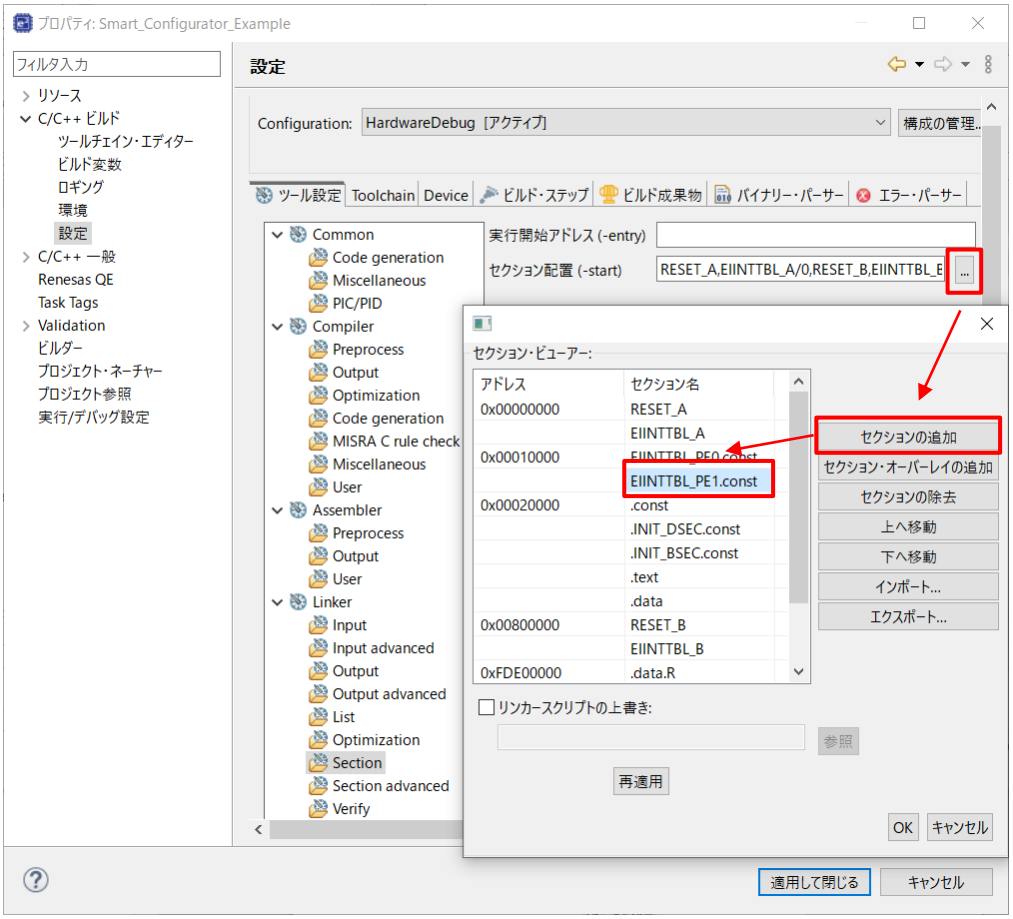


図 4-13 セクションの設定

(6) 「r_cg_intc_PE1.c」ファイルにある API 名「R_Interrupt_Initialize_ForPE」を「R_Interrupt_Initialize_ForPE1」に変更します。

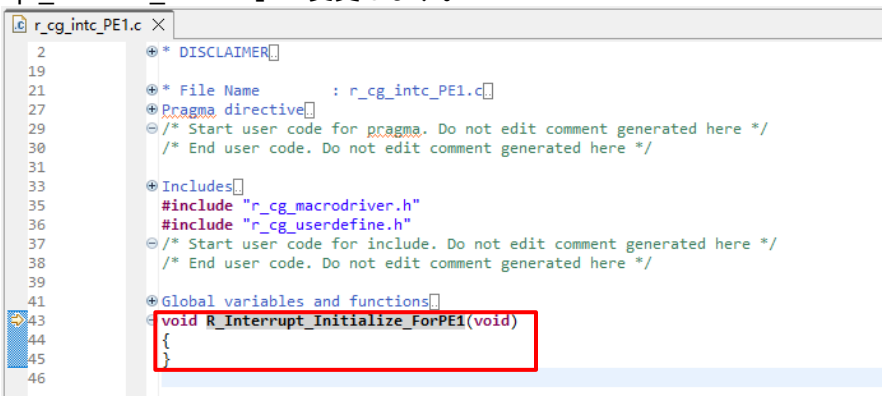


図 4-14 API 名変更

(7) API「R_Interrupt_Initialize_ForPE1」の関数宣言を追加します。

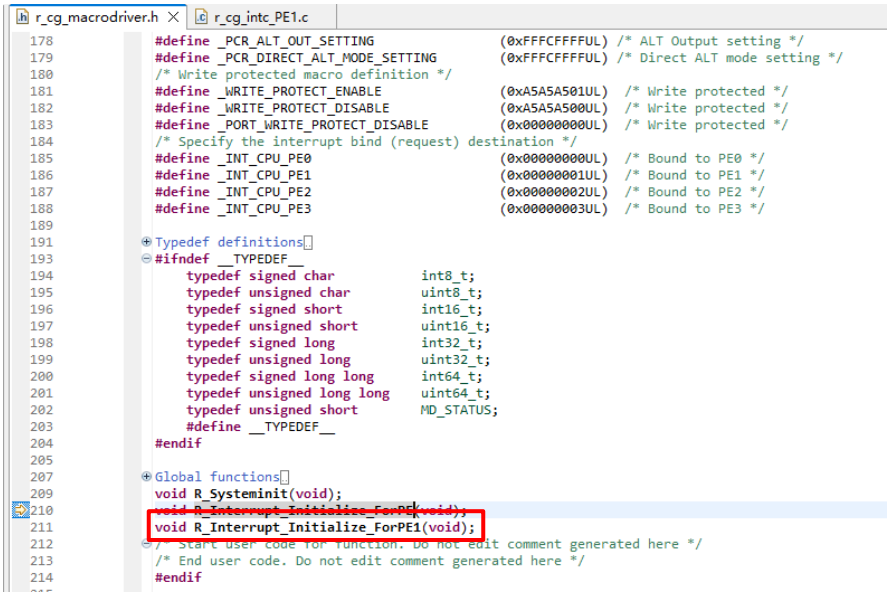


図 4-15 関数宣言の追加

(8) API「R_Interrupt_Initialize_ForPE1」の呼び出し記述を追加します。

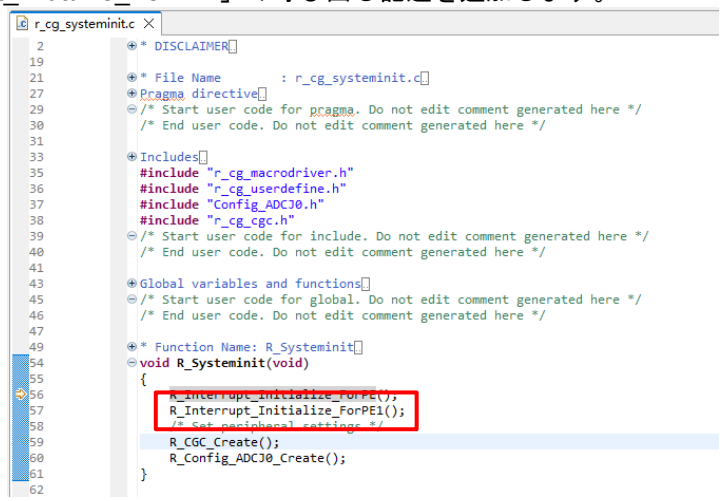


図 4-16 関数呼び出し記述の追加

4.4　コンポーネント設定

コンポーネントページは、ドライバやミドルウェアをソフトウェアコンポーネントとして組み合わせます。追加したコンポーネントは、左側のコンポーネントツリーに表示されます。

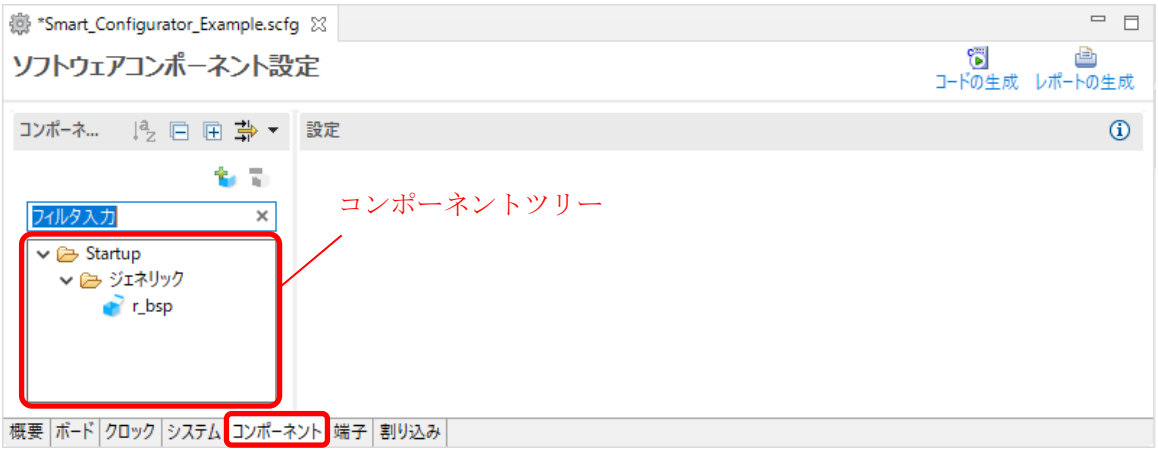


図 4-17 コンポーネントページ

4.4.1　コンポーネント一覧表示とハードウェア一覧表示の切り替え

スマート コンフィグレータには、コンポーネント一覧表示とハードウェア一覧表示の 2 つのツリー ビューが用意されています。 ユーザーは、次のアイコンをクリックして 2 つのビューを切り替えることができます。

・コンポーネント一覧表示

ツリー ビューには、コンポーネントのカテゴリごとにコンポーネントが表示されます。

・ハードウェアビューで表示

ツリー ビューには、ハードウェアリソース階層内のコンポーネントが表示されます。

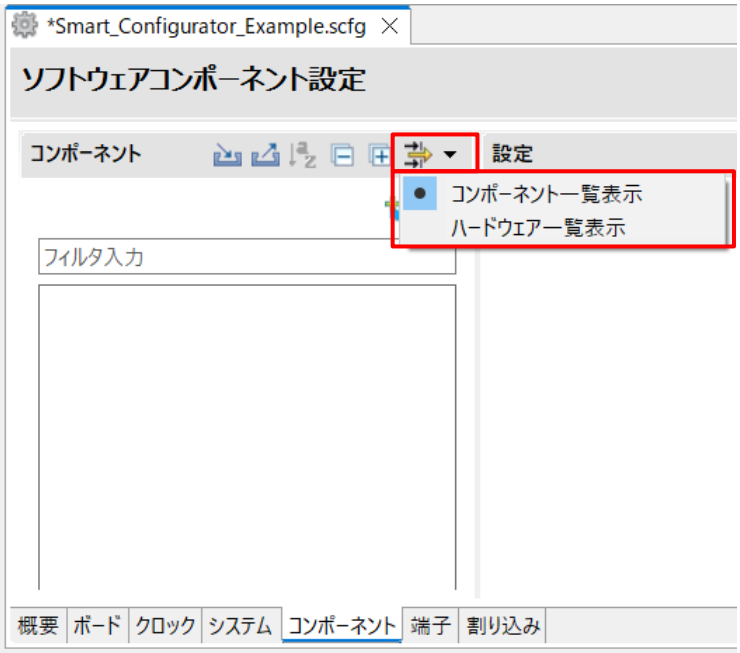


図 4-18 ビューの切り替え

4.4.2　コード生成コンポーネントの追加方法

コポーネントの追加は、以下の手順で行います。

(1) ［コンポーネントの追加］ アイコンをクリックします。

図 4-19 コンポーネントの追加

(2) ［コンポーネントの追加］ダイアログボックスの［ソフトウェアコンポーネントの選択］のリストからコンポーネントを選択し、［次へ］をクリックします。（例：A/D コンバータ）

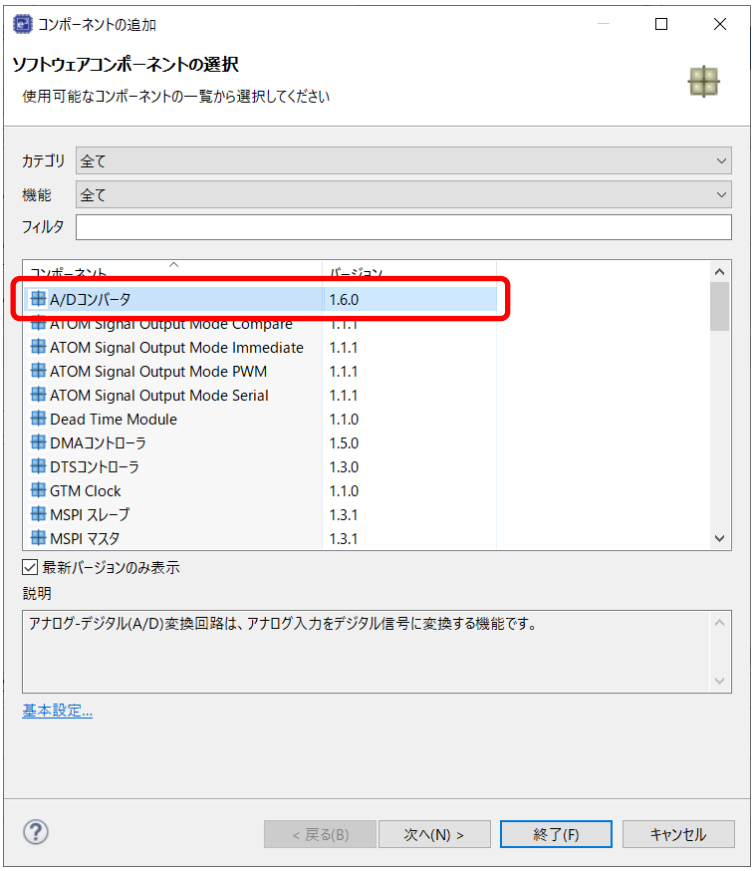


図 4-20 コード生成コンポーネントの追加

(3) ［コンポーネントの追加］ダイアログボックスの［選択したコンポーネントのコンフィグレーションを追加します］ページで、適切なコンフィグレーション名を入力、またはデフォルト名を使用します。（例：Config_ADCJ0）

リソースを選択し、[終了] をクリックします。（例：ADCJ0）

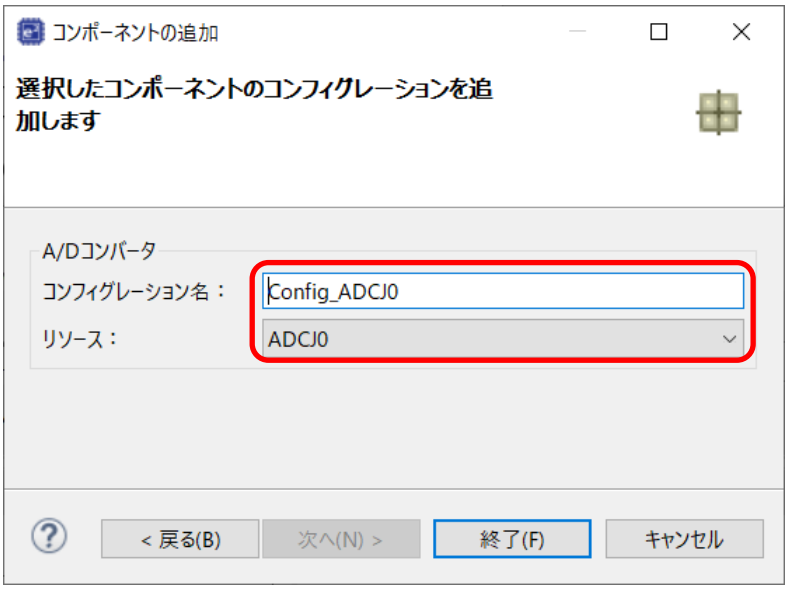


図 4-21 コンポーネントのコンフィグレーション追加

4.4.3　ソフトウェアコンポーネントの削除

プロジェクトからソフトウェアコンポーネントを削除するには、以下の手順で行います。

(1) コンポーネントツリーからソフトウェアコンポーネントを選択します。

(2) ［コンポーネントの削除］ アイコンをクリックします。

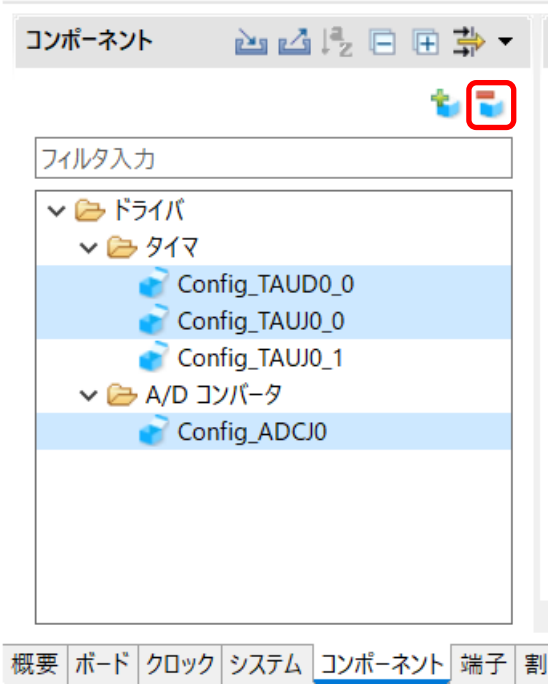


図 4-22 ソフトウェアコンポーネントの削除

コンポーネントツリーから、選択したソフトウェアコンポーネントが削除されます。

この操作では、プロジェクトに登録されているソースファイルは削除されません。[コード生成] ボタンでソースファイル出力を行うと、削除したソフトウェアコンポーネントのソースファイルが削除されます。

4.4.4　ソフトウェアコンポーネントの設定

ソフトウェアコンポーネントを設定するには、以下の手順で行います。

(1) コンポーネントツリーにある ソフトウェアコンポーネントを選択します。（例：A/D コンバータ）

(2) 右側の設定パネルでドライバを設定します。図は例です。

a. ［サンプリング制御設定］で［24 サイクル］を選択します。

b. ［スキャングループ選択］で［スキャングループ 2 を使用する］を選択します。

c. ［仮想チャンネルのエンドポイント］に［3］を入力します。

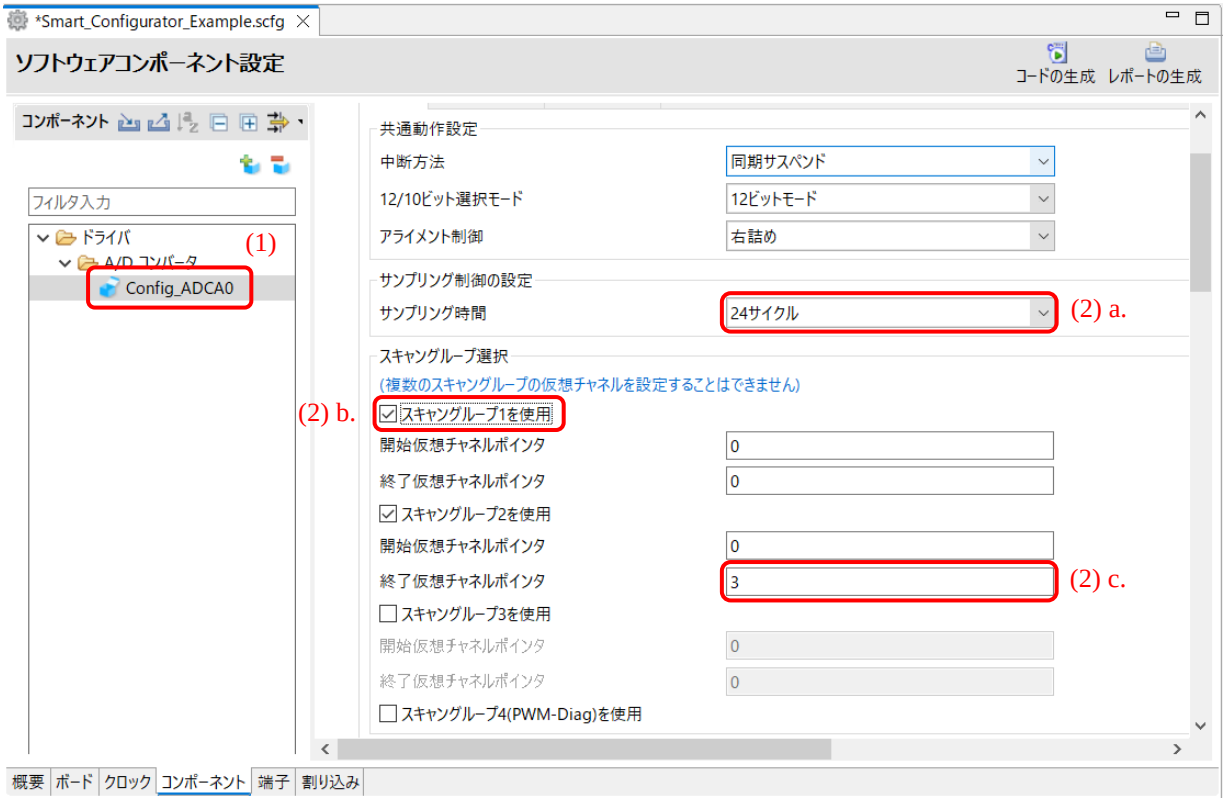


図 4-23 ソフトウェアコンポーネントの設定

ソフトウェアコンポーネントのコード生成は、デフォルトで生成する設定になっています。

ソフトウェアコンポーネントを右クリックし、［ コード生成 ］をクリックすると、［ コード生成 ］に変わりコードを生成しません。

［ コード生成 ］をクリックすると、［ コード生成 ］に変わりコードを生成します。

4.4.5　ソフトウェアコンポーネントのリソース変更

スマート・コンフィグレータでは、ユーザーは ソフトウェアコンポーネントのリソースを変更することができます（例：ADCA0 から ADCA1 に変更）。互換性のある設定は、現在のリソースから新しく選択したリソースへ移行することができます。

現在のソフトウェアコンポーネント用にリソースを変更するには、以下の手順で行います。

(1)　ソフトウェアコンポーネントを右クリックします（例：Config_ADCA0）。

(2)　コンテキストメニューから［リソースの変更］を選択します。

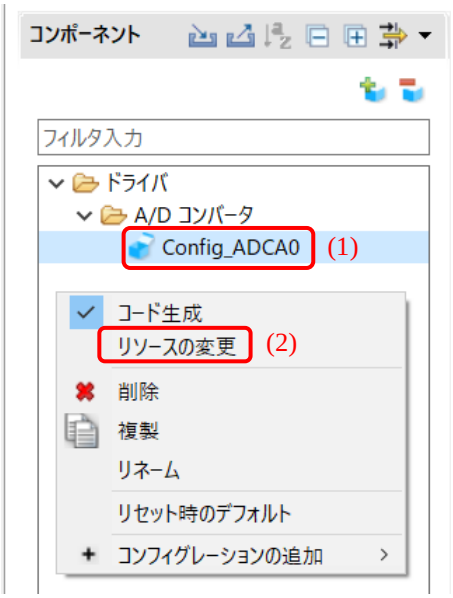


図 4-24 リソースの変更

(3)　［リソースの選択］ダイアログボックスにある新しいリソースを選択します（例：ADCA1）。

(4)　［次へ］ボタンが有効になるので、クリックします。

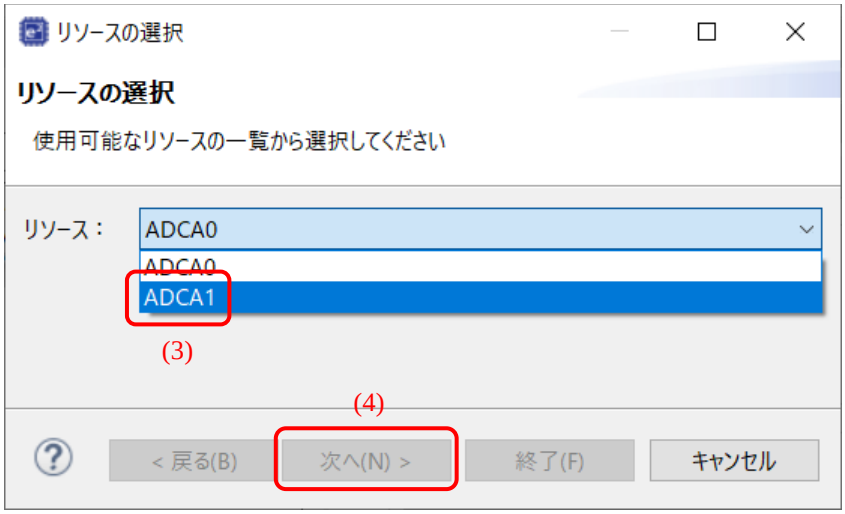


図 4-25 コンポーネントページ-新しいリソースの選択

(5) コンフィグレーション設定は、［コンフィグレーション設定の選択］ダイアログボックスに表示されます。

(6) 設定が変更可能であるかを確認します。

(7) テーブル内の設定を使用するか、デフォルト設定を使用するか選択します。

(8) ［終了］をクリックします。

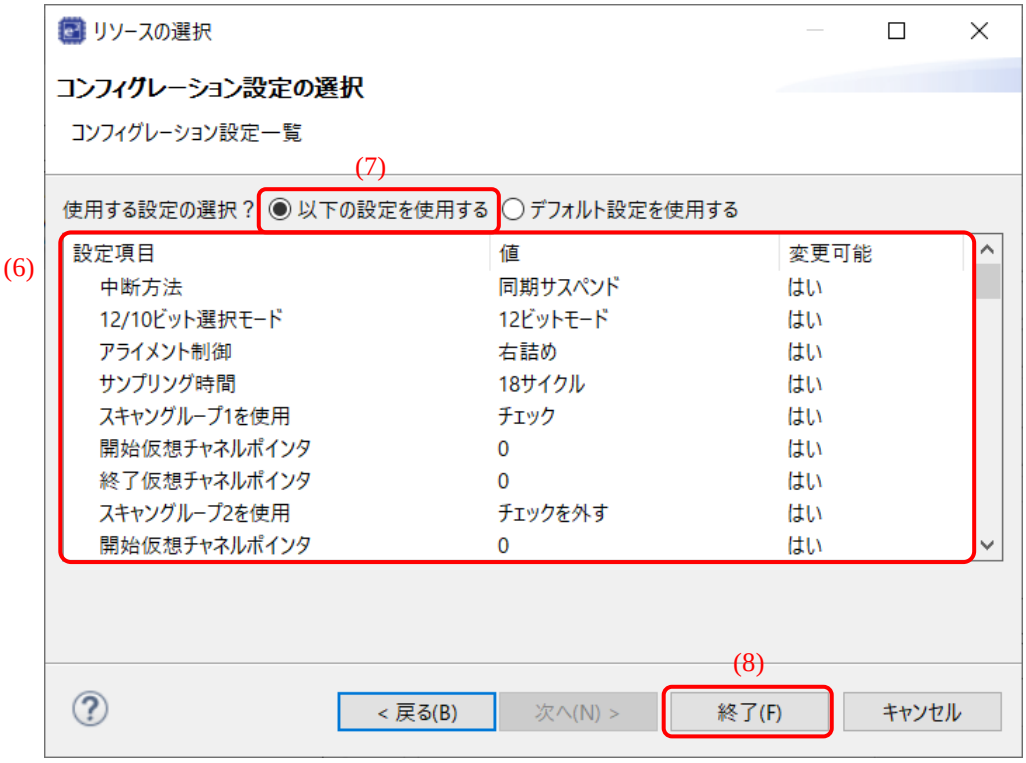


図 4-26 新しいリソース設定の確認

リソースは、自動的に更新されます。（例：ADCA0 から ADCA1）

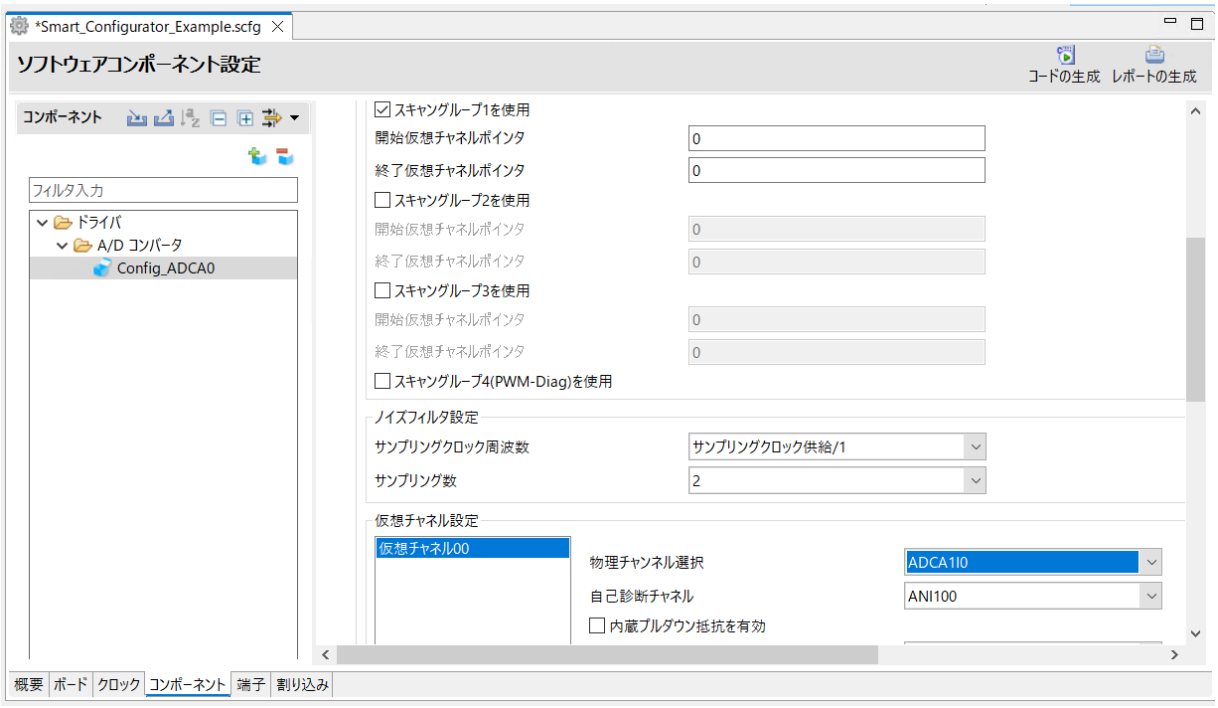


図 4-27 自動的に更新されるリソース

コンポーネント名を変更する場合は、以下の手順で行います。

(9) コード ジェネレーター構成を右クリックします。

(10) [名前の変更] を選択して、構成の名前を変更します (例: Config_ADCA0 を Config_ADCA1 に変更します)。

コンフィグレーション名を変更する場合は、以下の手順で行います。

(9) ソフトウェアコンポーネントを右クリックします。

(10) ［リネーム］を選択して、コンポーネントに再度名前をつけます（例：Config_ADCA0 を Config_ADCA1）。

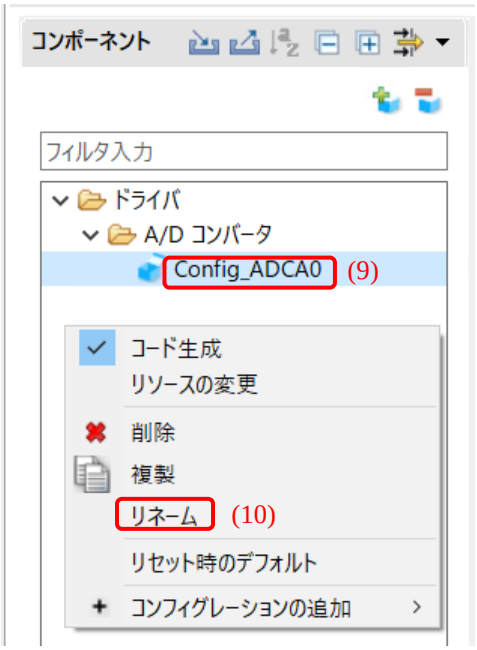


図 4-28 コンフィグレーションに再度名前をつける

4.4.6 コンポーネントの設定のエクスポート

[コンポーネント]ページに[コンフィグレーションのエクスポート] ボタンをクリックすると、現在の設定を*.xml ファイルとしてエクスポートできます。

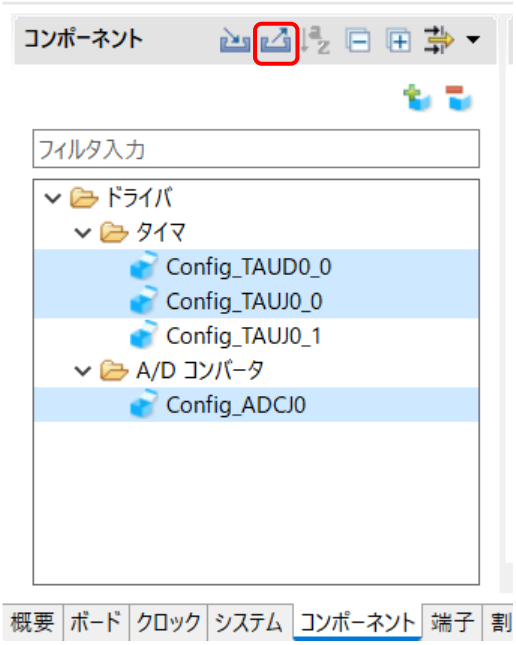


図 4-29 コンフィグレーションのエクスポート

4.4.7　コンポーネントの設定のインポート

[コンフィグレーションのインポート] ボタンをクリックし、エクスポートした*.xml ファイルを選択すると、*.xml ファイルの内容をインポートします。

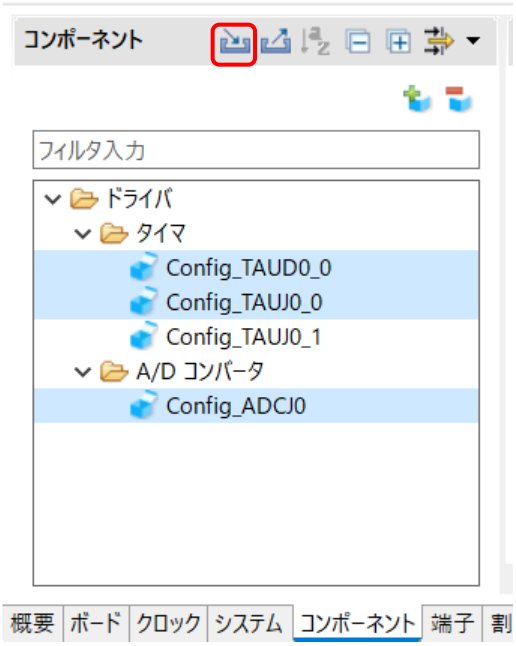


図 4-30 コンフィグレーションのインポート

4.4.8　コンポーネントの基本設定

モジュールの保存先、依存関係などのコンポーネントの基本設定を変更できます。変更するには、［コンポーネントの追加］ダイアログ（**図 4-20 コード生成コンポーネントの追加**）に表示される［ソフトウェアコンポーネントの選択］ページの［基本設定］リンクをクリックし、［設定］ダイアログを表示させます。

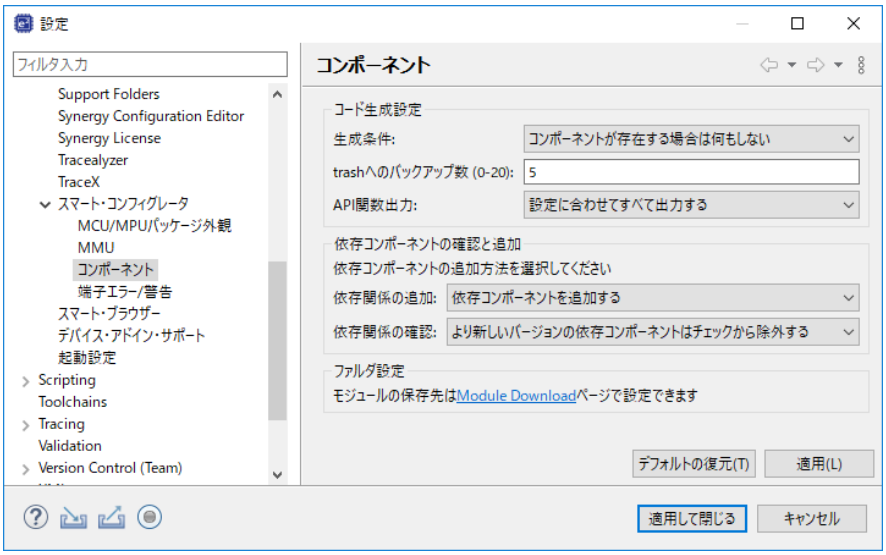


図 4-31 コポーネント基本設定

【注】1. ユーザーは、[trash へのバックアップ数(1~20)] オプション（図 4-32 に示す）を設定することで、バックアップのためにトラッシュフォルダに生成されるフォルダの数を制限できます。 制限を超えると、最も古いフォルダが新しいフォルダに置き換わります。

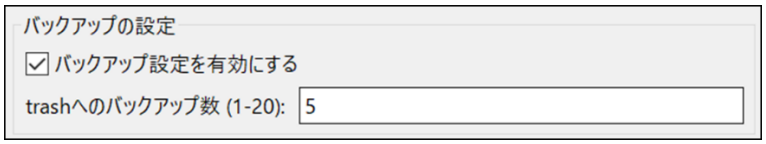


図 4-32 バックアップ数の設定

2. 初期化 API 関数のみを生成したい場合は、[API 関数出力:] リストボックスで「初期化関数のみ出力する」に変更してください。".h "、". c "ファイルの voidR_ {ConfigurationName} _Create（void）、void R_ {ConfigurationName} _Create_UserInit（void）のみが生成されます。 デフォルトのオプション設定「設定に合わせてすべて出力する」に変更するとすべての API 関数が再度生成されます。

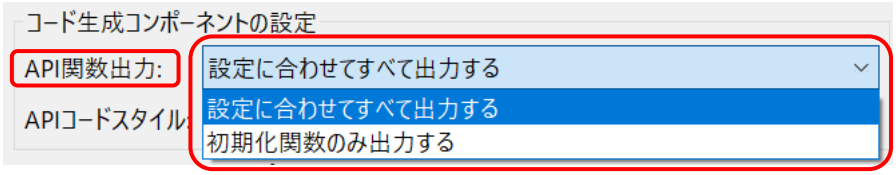


図 4-33 [API 関数出力:] の変更

4.5　端子設定

端子ページは、端子機能の割り当てに使用します。周辺機能別に端子機能を表示する［端子機能］リストと、端子番号順に全ての端子を表示する［端子番号］リストの 2 つの表示があり、タブを切り替えることで切り替えることができます。

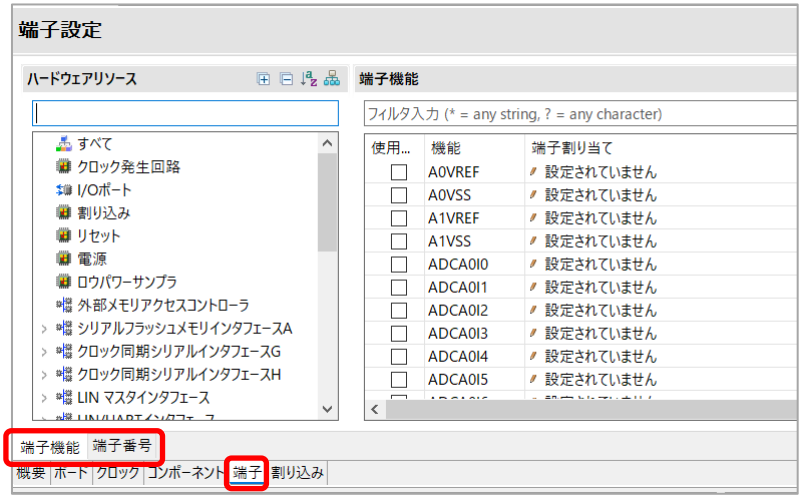


図 4-34 端子ページ（端子機能）

[ボード]ページでボードを選択すると、[ボード機能]に初期端子設定情報が表示されます。

また、[機能]の選択リストに表示される[ ]アイコンは、ボードの初期端子機能を示します。

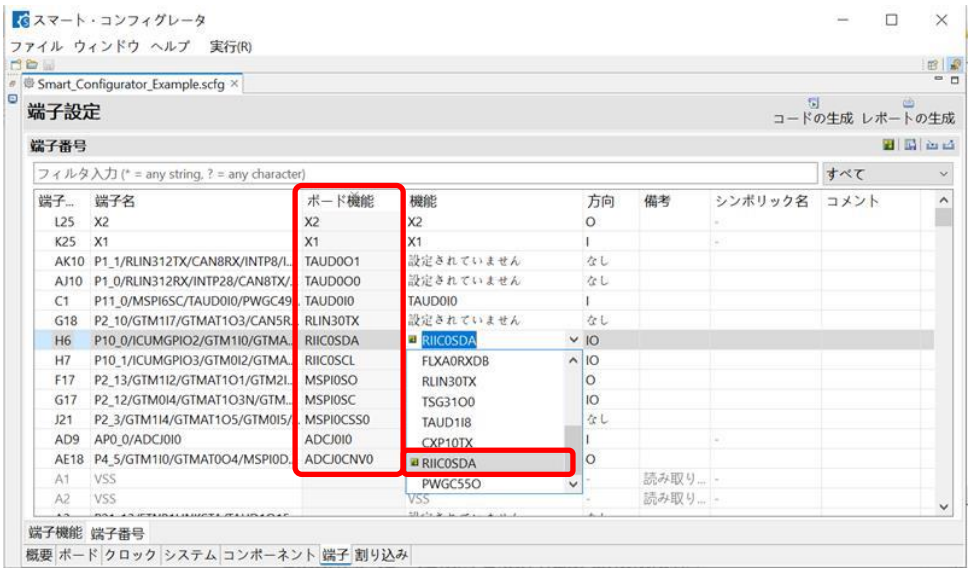


図 4-35 端子ページ（端子番号）

4.5.1　ソフトウェアコンポーネントの端子配置変更

スマート・コンフィグレータは、プロジェクトに追加されるソフトウェアコンポーネントに端子を配置します。端子の配置は端子ページで変更可能です。

このページでは、端子機能と端子番号のリストを表示します。

端子機能リストにあるソフトウェアコンポーネントの端子配置を変更するには、以下の手順で行います。

(1) ［ハードウェアリソース表示とソフトウェアコンポーネント表示の切り替え］ をクリックして、ソフトウェアコンポーネントによって表示するように変更します。
(2) ソフトウェアコンポーネントを選択します。（例：Config_ICU）
(3) ［使用する］タブをクリックし、使用した端子でソートします。
(4) 端子機能リストの端子割り当て欄で、端子配置を変更します。（例：P10_0 から P0_1）
(5) 同じ周辺チャネルに属する1つの端子または複数の端子の配置は、一度［選択されたリソースの次の端子割り当て先］ ボタンをクリックするだけで変更することができます。

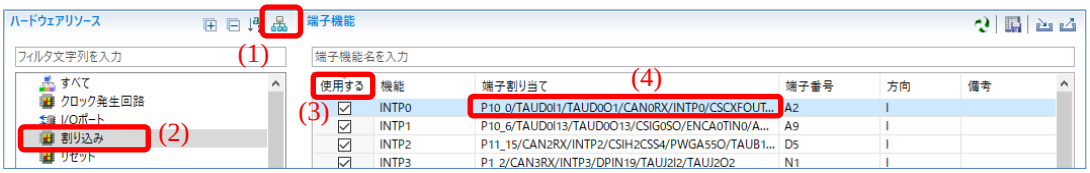


図 4-36 端子設定-［端子機能］リストの端子配置設定

スマート・コンフィグレータでは、ユーザーは他のソフトウェアコンポーネントにリンクすることなく、端子ページで端子機能を有効にすることができます。それらの端子をソフトウェアコンポーネントが使用する他の端子と区別するため、表の中に“この端子を使用するコンポーネントはない”という注意書きがつけられます。

【注】 現在、ソフトウェアコンポーネント表示は未サポートです。端子配置変更はハードウェアリソース表示から行ってください。

4.5.2　MCU/MPU パッケージの端子割り当て

スマート・コンフィグレータでは、 MCU パッケージビューでの端子設定を視覚化します。ユーザーは MCU/MPU パッケージビューをイメージファイルにキャプチャーでき、回転や拡大、縮小ができます。

MCU/MPU パッケージビューで端子を設定するには、以下の手順で行います。

(1)　［拡大］ボタンをクリックするか、マウスをスクロールして、ビュー内を拡大します。

(2)　端子の上で右クリックします。

(3)　割り当てを選択します。

(4)　［設定の変更...］で、端子の色をカスタマイズすることができます。

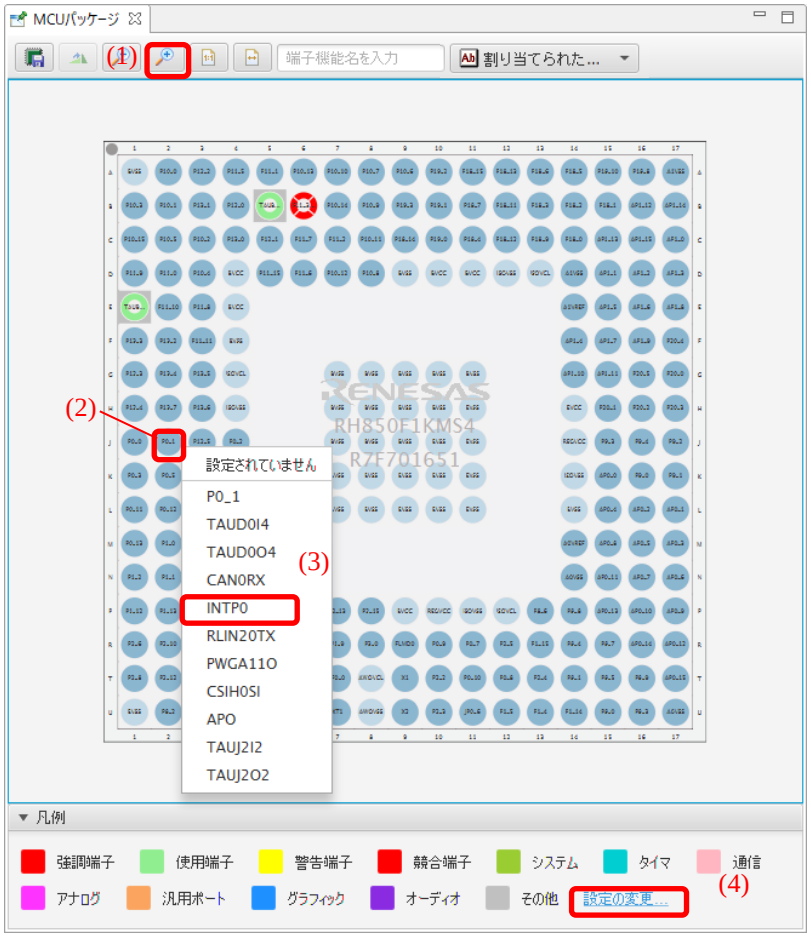


図 4-37 MCU パッケージを使用した端子設定

4.5.3 端子機能から端子番号の表示

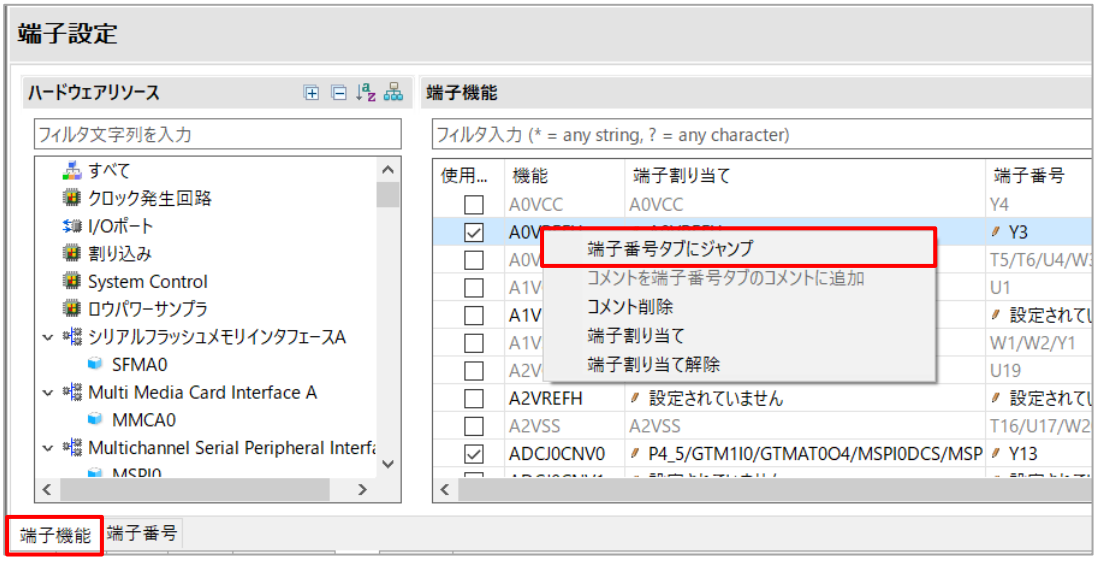
端子機能に関連付けられた端子番号を表示できます。

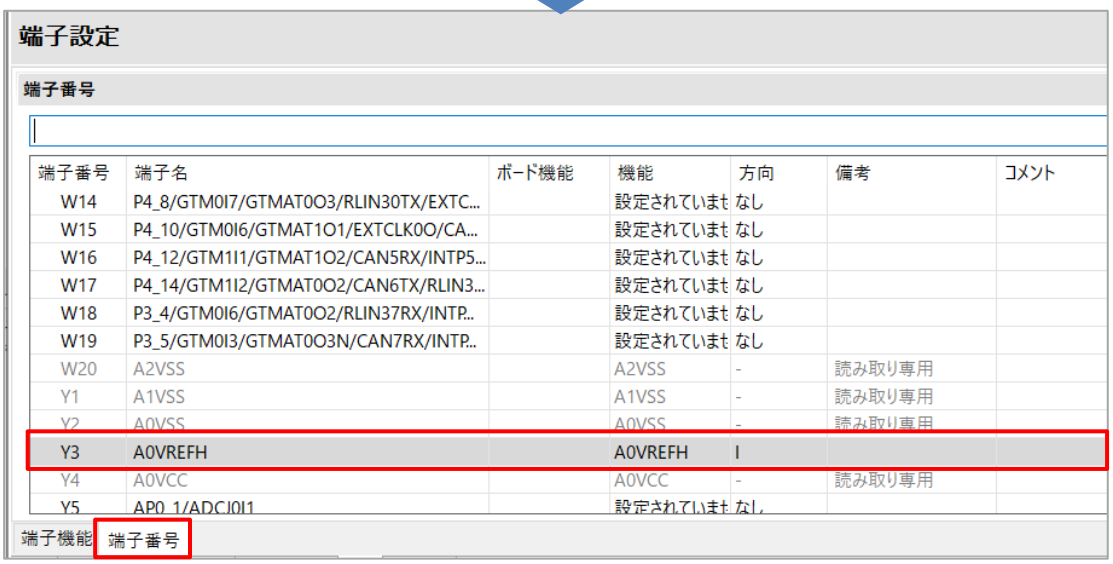
端子機能から端子番号にジャンプするには以下の手順で行います。

(1) [端子機能]タブで対象を右クリックし、ポップアップメニューを表示します。

(2) [端子番号タブへジャンプ]を選択します。

(3) [端子番号]タブが開き、端子番号が選択された状態になります。





4.5.4　端子設定のエクスポート

端子設定をエクスポートして、参照することができます。端子設定のエクスポートは、以下の手順で行います。

(1)　{ProjName}.scfg ファイルをセーブします。

(2)　端子ページで、［ボードの設定をエクスポート］ボタンをクリックします。

(3)　出力場所を選択し、エクスポートするファイル名を入力します。

XML フォーマットでエクスポートしたファイルは、同じデバイスの型名がある他のプロジェクトにインポートすることができます。

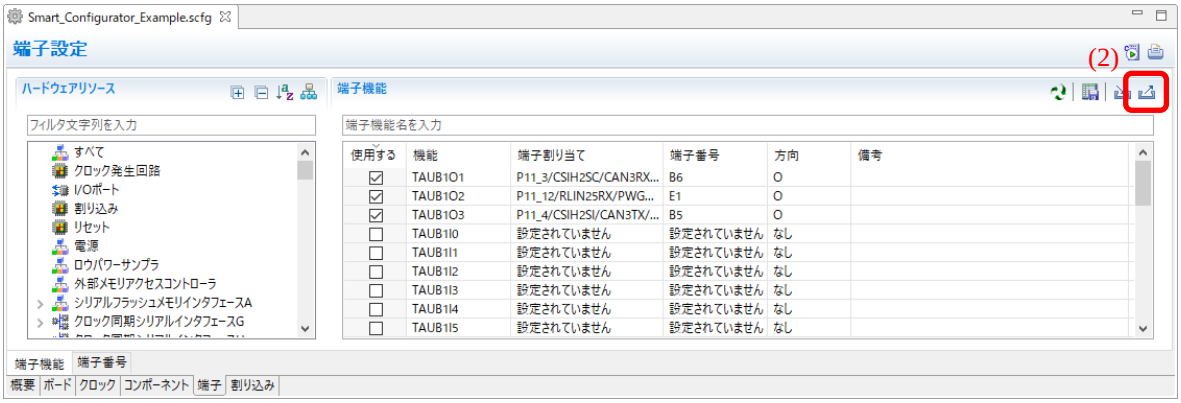


図 4-38 端子設定を XML ファイルへエクスポートする

端子ページの［.csv ファイルにリストを保存］ボタンをクリックすることで、スマート・コンフィグレータは CSV エクスポートをサポートすることができます。

4.5.5　端子設定のインポート

現在のプロジェクトに端子設定をインポートするには、［ボードの設定をインポート］ボタンをクリックし、端子設定を含む XML ファイルを選択してください。設定がプロジェクトにインポートされると、このファイルに指定された設定は、端子設定ページに反映されます。

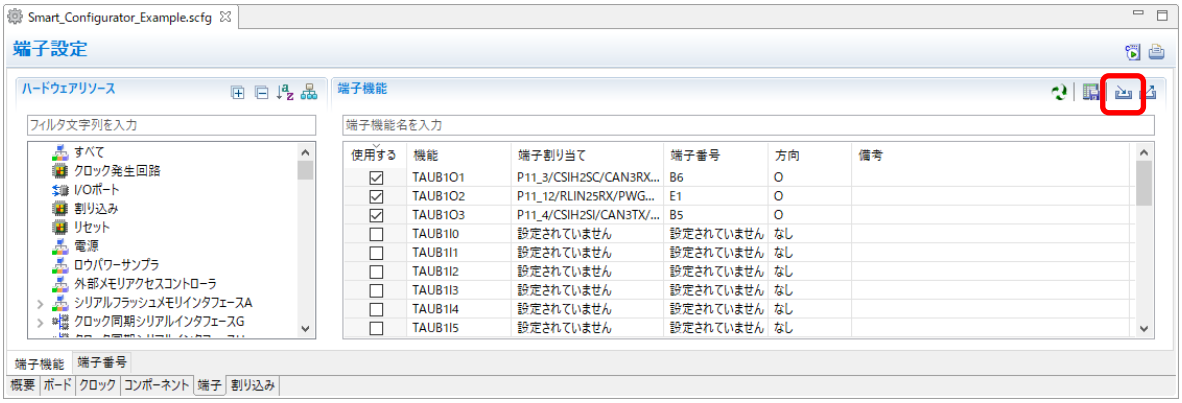


図 4-39 端子設定を XML ファイルからインポートする

【注】 端子設定は反映されますが、コンポーネント設定には反映されません。

4.5.6　ボード端子設定情報を使用した端子設定

ルネサス製ボードの初期端子構成を設定できます。選択したボードは[ボード]タブで確認できます。

以下に端子を一括設定する手順を説明します。

(1)　[MCU/MPU パッケージ]の[ボード機能]を選択します。

(2)　[端子設定]ページを開き、[ボードの初期端子割り当ての設定]ボタンをクリックします。

(3)　[ボードの初期端子割り当て]ダイアログが開くので、[すべて選択]をクリックします。

(4)　[OK]をクリックします。

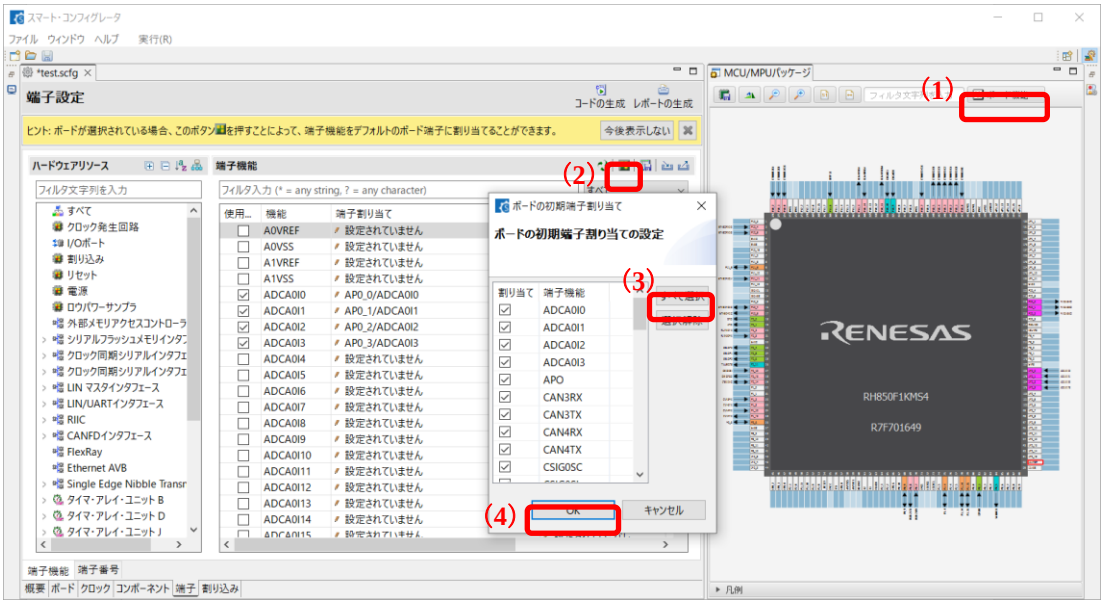


図 4-40 端子の初期設定

4.5.7　端子のフィルタ機能

[端子]ページの[端子設定]画面で、端子機能及び端子番号をフィルタリングすることが出来ます。

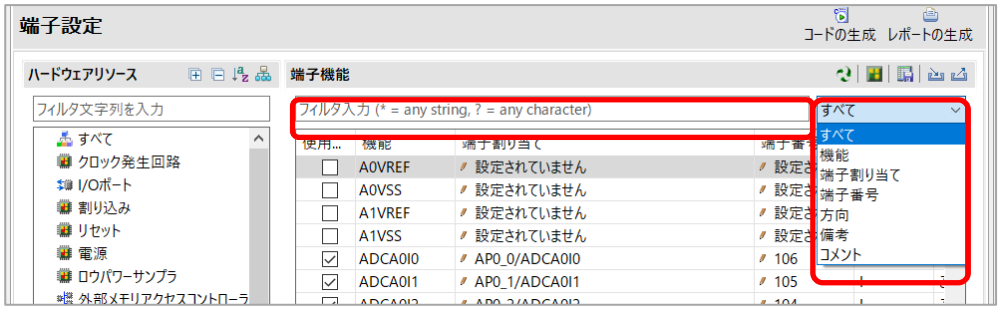


図 4-41 端子設定のフィルタリング

4.5.8　端子エラー／警告設定

[端子エラー/警告] 設定を使用して、[コンフィグレーションチェック] ビューのメッセージのエラーレベルを制御できます。

エラーレベルを変更は、[設定]ダイアログを開き、[スマート・コンフィグレータ] > [端子エラー/警告] ページで行います。

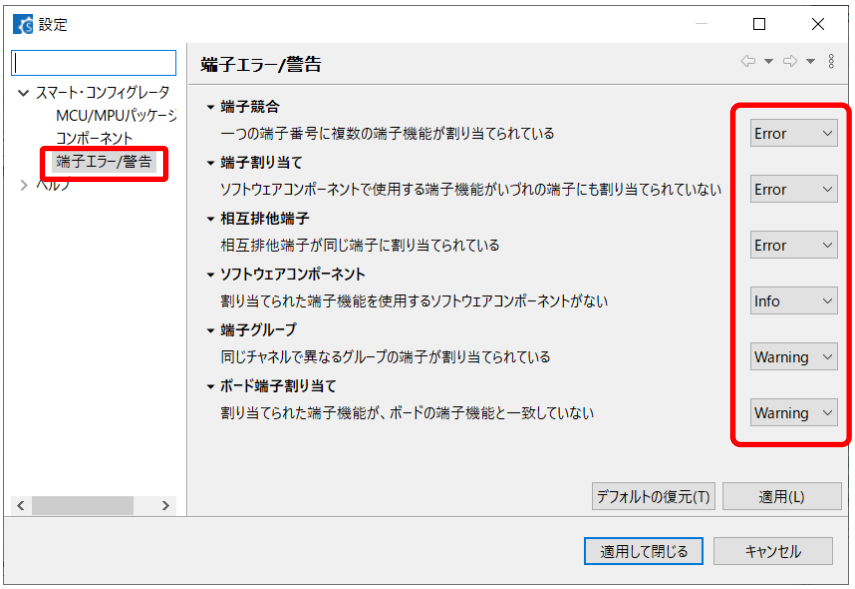


図 4-42 端子エラー/警告の設定

例えば、「ソフトウェアコンポーネント」の「割り当てられた端子機能を使用するソフトウェアコンポーネントがない」の項目を「Info」から「Error」に変更すると、メッセージが図 **4-43** のようになります。

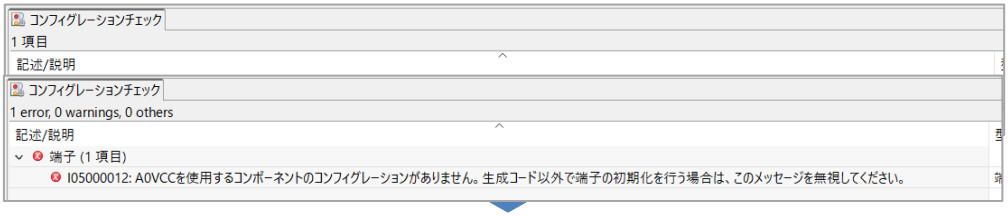


図 4-43 端子エラー/警告の設定の変更例

4.6　割り込み設定

割り込みページは、［コンポーネント］ページで設定した周辺機能の割り込みの確認および設定を行います。ベクタ番号別に割り込みが表示されます。割り込み優先レベル、OS 管理、割り込みハンドラ、エンティティの生成、関数の生成の有効化/無効化などの共通設定を設定できます。 RH850/U2A の場合、PEn を設定して、PEn に割り込みを適用するかどうかを決定できます。。

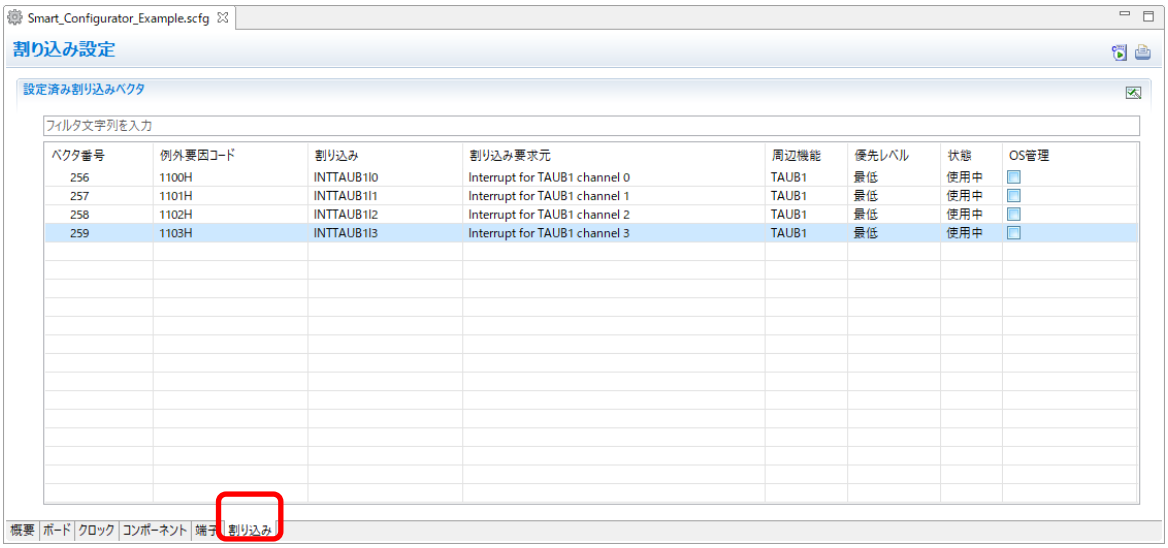


図 4-44 割り込みページ

4.6.1　割り込み優先レベルと OS 管理の設定

コンポーネントページでのコンフィグレーションに割り込みを使用する場合、割り込みの状態は“使用中”に変わります。使用中の割り込みだけを表示する場合は、［設定した割り込みのみ表示］ ボタンをクリックしてください。

(1) 割り込みページで、割り込み優先レベルを変えることができます。

(2) RTOS（RI850V4）を使用するプロジェクト時、OS管理カラムが有効になります。チェックをすると、割り込み関数がOS管理の割り込み書式で出力されます。

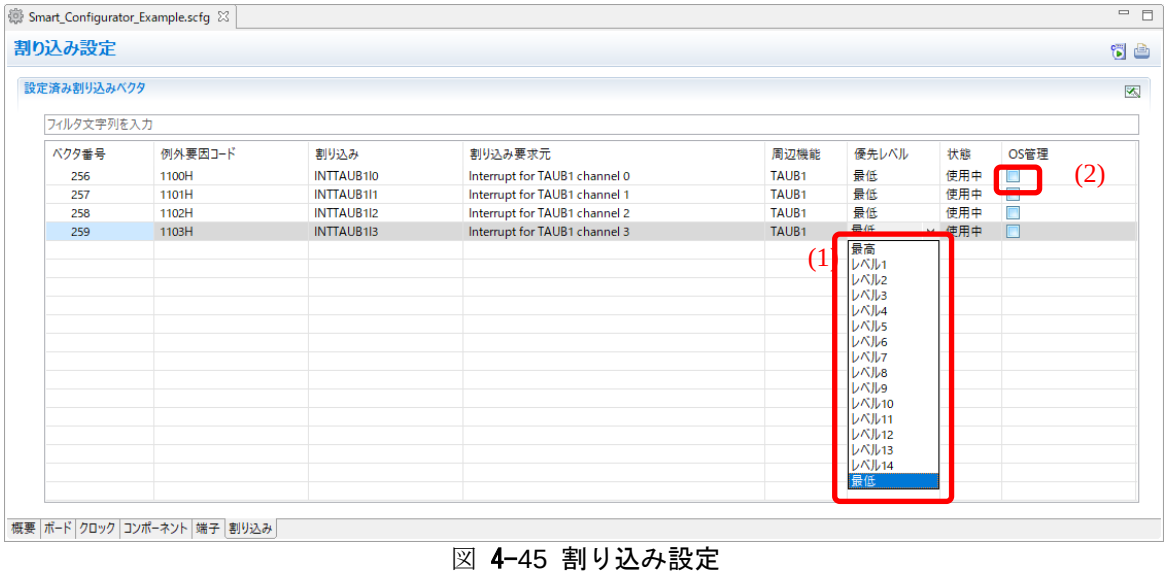


図 4-45 割り込み設定

4.6.2　PE*n* の設定の変更(RH850/U2A のみ)

RH850/U2A デバイスでは、[割り込み] ページで各割り込みに応答する PE*n* を選択することができます。

PE*n* は以下の手順で設定できます。

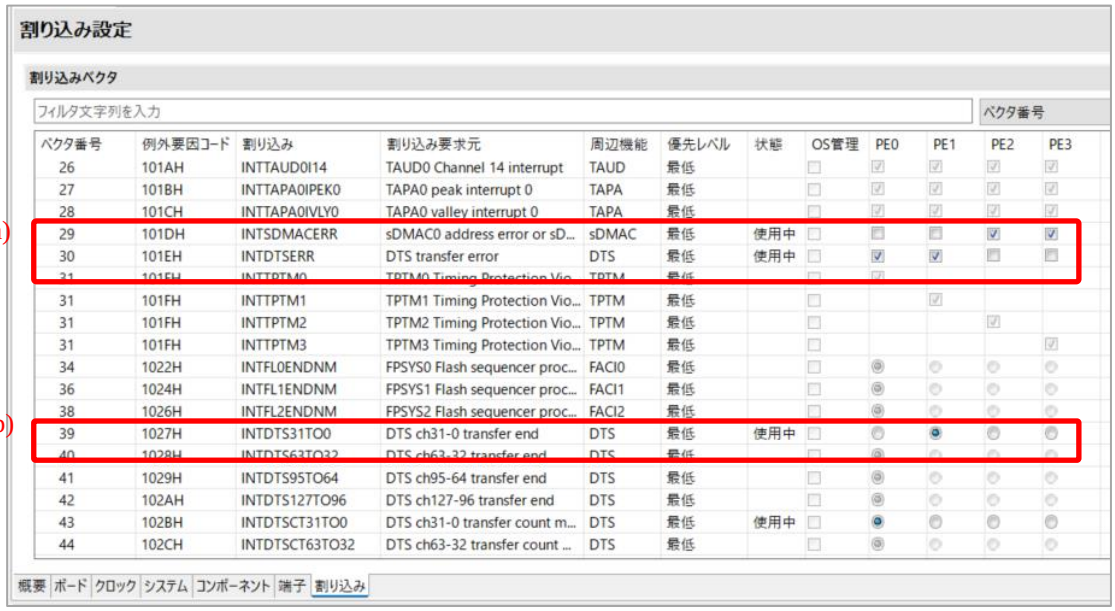
(1)　[システム]ページで使用する PE*n* を選択します(4.3 **システム設定(RH850/U2A のみ)**を参照)。

図 4-46　[システム]ページの PE 選択

(2)　[割り込み] ページの PE*n* 列のチェックボックスをオンまたはオフにして、割り込みに応答する PE を決定します。 割り込みには次の 2 種類があります。

(a)　各 PE の INTC1 に接続され、PE*n* 列で選択された各 PE が応答可能。

(b)　複数の PE で共有される INTC2 に接続され、PE*n* 列の 1 だけが応答可能。



ベクタ番号	例外要因コード	割り込み	割り込み要求元	周辺機能	優先レベル	状態	OS管理	PE0	PE1	PE2	PE3
26	101AH	INTTAUD0I14	TAUD0 Channel 14 interrupt	TAUD	最低			☑	☑	☑	☑
27	101BH	INTTAPA0IPEK0	TAPA0 peak interrupt 0	TAPA	最低			☑	☑	☑	☑
28	101CH	INTTAPA0IVLY0	TAPA0 valley interrupt 0	TAPA	最低			☑	☑	☑	☑
29	101DH	INTSDMACERR	sDMAC0 address error or sD...	sDMAC	最低	使用中		☐	☐	☑	☑
30	101EH	INTDTSERR	DTS transfer error	DTS	最低	使用中		☑	☑	☐	☐
31	101FH	INTTPTM0	TPTM0 Timing Protection Vio...	TPTM	最低			☑			
31	101FH	INTTPTM1	TPTM1 Timing Protection Vio...	TPTM	最低			☐	☑		
31	101FH	INTTPTM2	TPTM2 Timing Protection Vio...	TPTM	最低			☐		☑	
31	101FH	INTTPTM3	TPTM3 Timing Protection Vio...	TPTM	最低			☐			☑
34	1022H	INTFL0ENDNM	FPSYS0 Flash sequencer proc...	FACI0	最低			◎	○	○	○
36	1024H	INTFL1ENDNM	FPSYS1 Flash sequencer proc...	FACI1	最低			◎	○	○	○
38	1026H	INTFL2ENDNM	FPSYS2 Flash sequencer proc...	FACI2	最低			◎	○	○	○
39	1027H	INTDTS31TO0	DTS ch31-0 transfer end	DTS	最低	使用中		○	◉	○	○
40	1028H	INTDTS63TO32	DTS ch63-32 transfer end	DTS	最低			◎	○	○	○
41	1029H	INTDTS95TO64	DTS ch95-64 transfer end	DTS	最低			◎	○	○	○
42	102AH	INTDTS127TO96	DTS ch127-96 transfer end	DTS	最低			◎	○	○	○
43	102BH	INTDTSCT31TO0	DTS ch31-0 transfer count m...	DTS	最低	使用中		◉	○	○	○
44	102CH	INTDTSCT63TO32	DTS ch63-32 transfer count ...	DTS	最低			◎	○	○	○

図 4-47　[割り込み]ページの PE 設定

4.6.3　割り込み設定の変更

[割り込み]ページで、各割り込みハンドラの編集および、割り込みハンドラのエンティティを生成するかどうかを設定できます。

(1)　ユーザーは、コンポーネントで使用されていない割り込みハンドラ名を手動で入力できます。

(2)　[エンティティを生成する]にチェックを入れると、割り込みハンドラのエンティティを生成します。

デフォルトではチェックが入っています。 チェックを外すと割り込みハンドラコードが生成されなくなり、ユーザーが独自のハンドラコードを使用できるようになります。

(3) [有効化/無効化関数を生成する]にチェックを入れると、割り込みを有効/無効にする関数を生成します。デフォルトではチェックは入っていません。

チェックを入れると、「r_smc_interrupt.c」ファイルに割り込み有効/無効の関数が生成されます。

ユーザーはこれらの API を直接呼び出すことで簡単に割り込みを使用できます。

割り込み有効/無効の関数のコードの例については、「図 4-49 割り込み有効/無効関数のコード例」を参照してください。

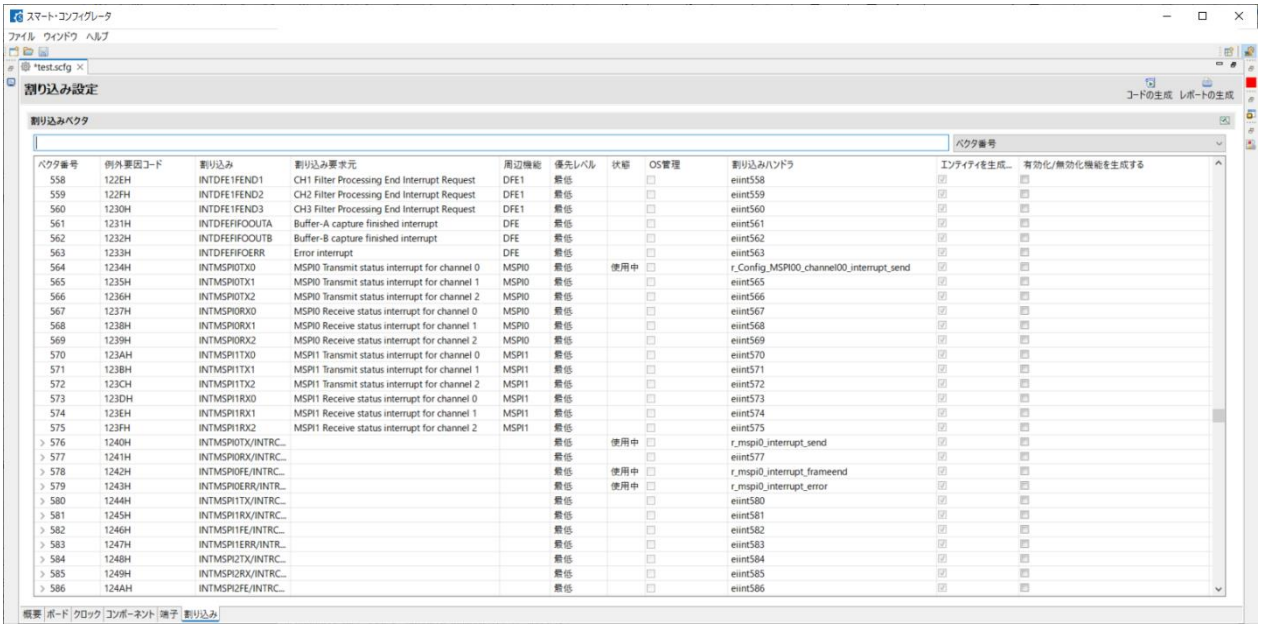


図 4-48　割り込みハンドラとエンティティの設定

r_smc_interrupt.c

```
/***********************************************************************************************************************
* File Name        : r_smc_interrupt.c
* Version          : 1.3.0
* Device(s)        : R7F702301BEBBA
* Description      : None
***********************************************************************************************************************/
/***********************************************************************************************************************
/* Start user code for pragma. Do not edit comment generated here */
/* End user code. Do not edit comment generated here */

/***********************************************************************************************************************
#include "r_cg_macrodriver.h"
#include "r_cg_userdefine.h"
#include "r_smc_interrupt.h"

/***********************************************************************************************************************
void R_Interrupt_Create(void)
{

}

void r_Config_MSPI00_channel00_interrupt_send_enable_interrupt(void)
{
    /* Clear INTMSPI0TX0 request and enable operation */
    INTC2.EIC244.BIT.EIRF244 = _INT_REQUEST_NOT_OCCUR;
    INTC2.EIC244.BIT.EIMK244 = _INT_PROCESSING_ENABLED;
}

void r_Config_MSPI00_channel00_interrupt_send_disable_interrupt(void)
{
    /* Disable INTMSPI0TX0 operation and clear request */
    INTC2.EIC244.BIT.EIMK244 = _INT_PROCESSING_DISABLED;
    INTC2.EIC244.BIT.EIRF244 = _INT_REQUEST_NOT_OCCUR;
}

void r_Config_MSPI00_channel00_interrupt_receive_enable_interrupt(void)
{
    /* Clear INTMSPI0RX0 request and enable operation */
    INTC2.EIC245.BIT.EIRF245 = _INT_REQUEST_NOT_OCCUR;
    INTC2.EIC245.BIT.EIMK245 = _INT_PROCESSING_ENABLED;
}

void r_Config_MSPI00_channel00_interrupt_receive_disable_interrupt(void)
{
    /* Disable INTMSPI0RX0 operation and clear request */
    INTC2.EIC245.BIT.EIMK245 = _INT_PROCESSING_DISABLED;
    INTC2.EIC245.BIT.EIRF245 = _INT_REQUEST_NOT_OCCUR;
}
```

図 4-49　割り込み有効/無効関数のコード例

4.7　MCU マイグレーション機能

MCU マイグレーション機能は、異なるデバイス間でプロジェクト設定の移行を行います。プロジェクト設定の変換は、同一ファミリ内で可能で、e² studio の［プロジェクト］メニューから以下の手順で行います。

【注】　デバイスの変更により、プロジェクトの設定が変わる場合があります。
　　デバイス変更を実行する前にプロジェクトのバックアップを行ってください。

(1)　プロジェクトを選択し、［プロジェクト］メニューから［Change Device］を選択します。

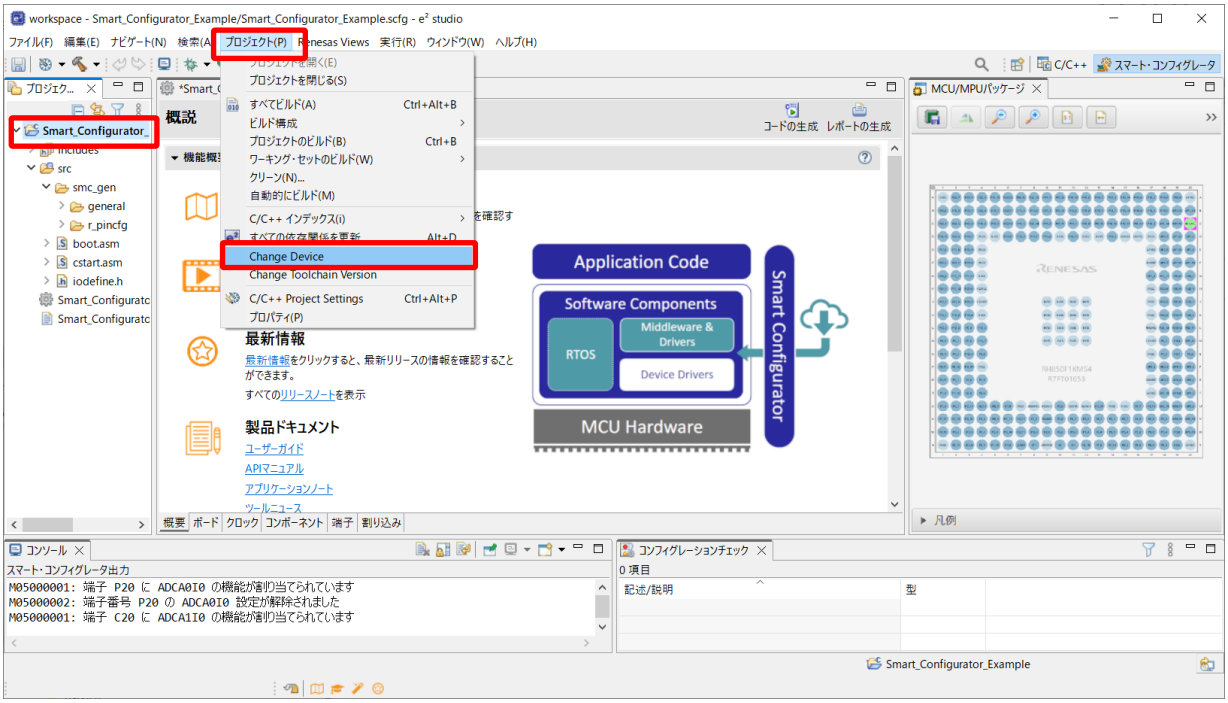


図 4-50 ［Change Device］選択

(2) デバイス選択リストから対象のデバイスを手動で選択し、「OK」をクリックします。 (ワイルドカード検索対応) (例: RH850 - C1M 252pin 部品番号: R7F701275 に変更)。

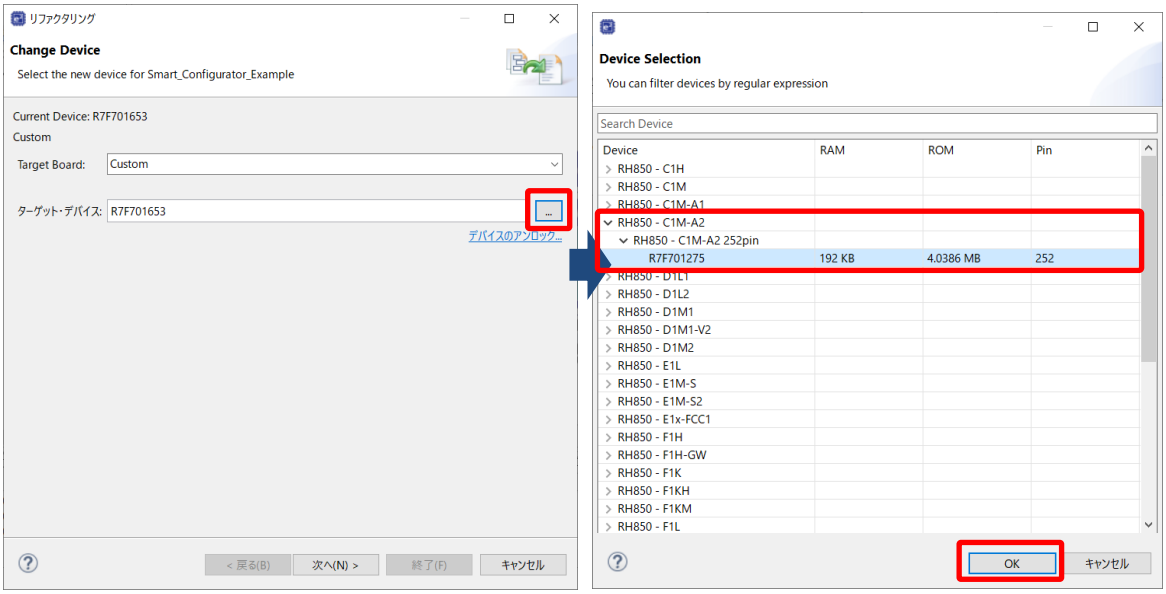


図 4-51 ターゲット・デバイス選択

(3) ［検出された問題］に表示されたメッセージを確認して［次へ］をクリックします。

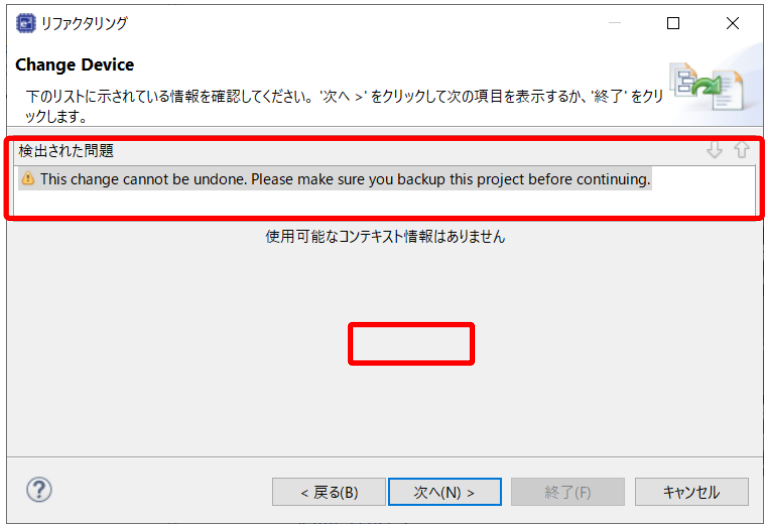


図 4-52 検出された問題

メッセージ	説明
ターゲット・デバイスはスマート・コンフィグレータでサポートされていません	スマート・コンフィグレータがサポートしていないデバイスへの変更時に表示されます。スマート・コンフィグレータの変換は実行できませんが、プロジェクト、ビルダー、リンカー、デバッカーは変換できます。
This change cannot be undone. Please make sure you backup this project before continuing.	デバイスを変更すると変更前に戻すことができませんので、プロジェクトのバックアップ後に実行してください。

(4) ［実行される変更］で、変更する項目を選択してマイグレーションを実行します。

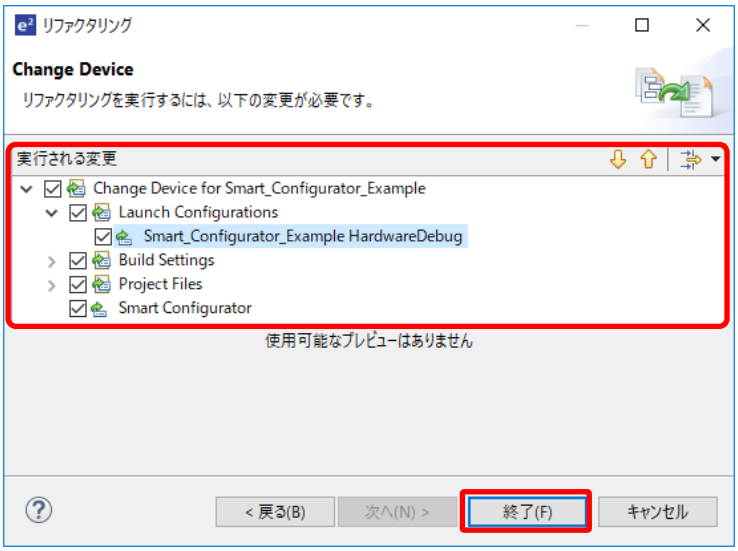


図 4-53 実行される変更

(5)　デバイスの変更が完了すると、「概要」ページのデバイス名が更新されます。

図 4-54 デバイス更新確認

(6)　コンソールにデバイス変更結果レポートが出力されます。

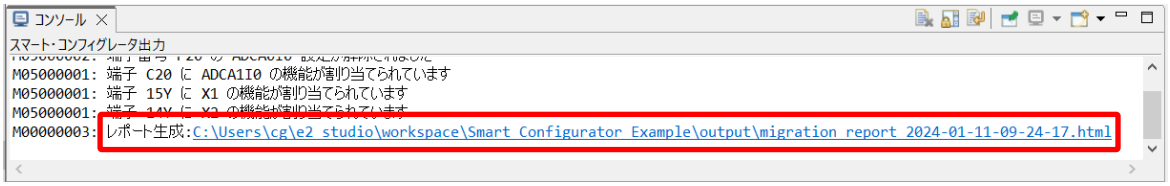


図 4-55 設定変換ステータスレポート

【注】 デバイス変更結果レポートが正しく表示されない場合は、右メニューより「エンコード」の設定を“Unicode(UTF-8)”に変更してください。

5. 競合の管理

コンポーネントの追加、端子や割り込みの設定をすると、リソースの不一致や失われた依存関係にあるモジュールに関連する問題が起こる可能性があります。この情報はコンフィグレーションチェックビューに表示されます。表示された情報を参照して、競合問題を解決してください。

5.1 リソースの競合

同じリソース（例：TAUB1）を使うために、二つのソフトウェアのコンフィグレーションを設定した場合、コンポーネントツリーにエラーマーク が表示されます。

コンフィグレーションチェックビューに周辺機能の競合に関するメッセージが表示され、ユーザーに周辺機能に競合が見つかったソフトウェア設定を知らせます。

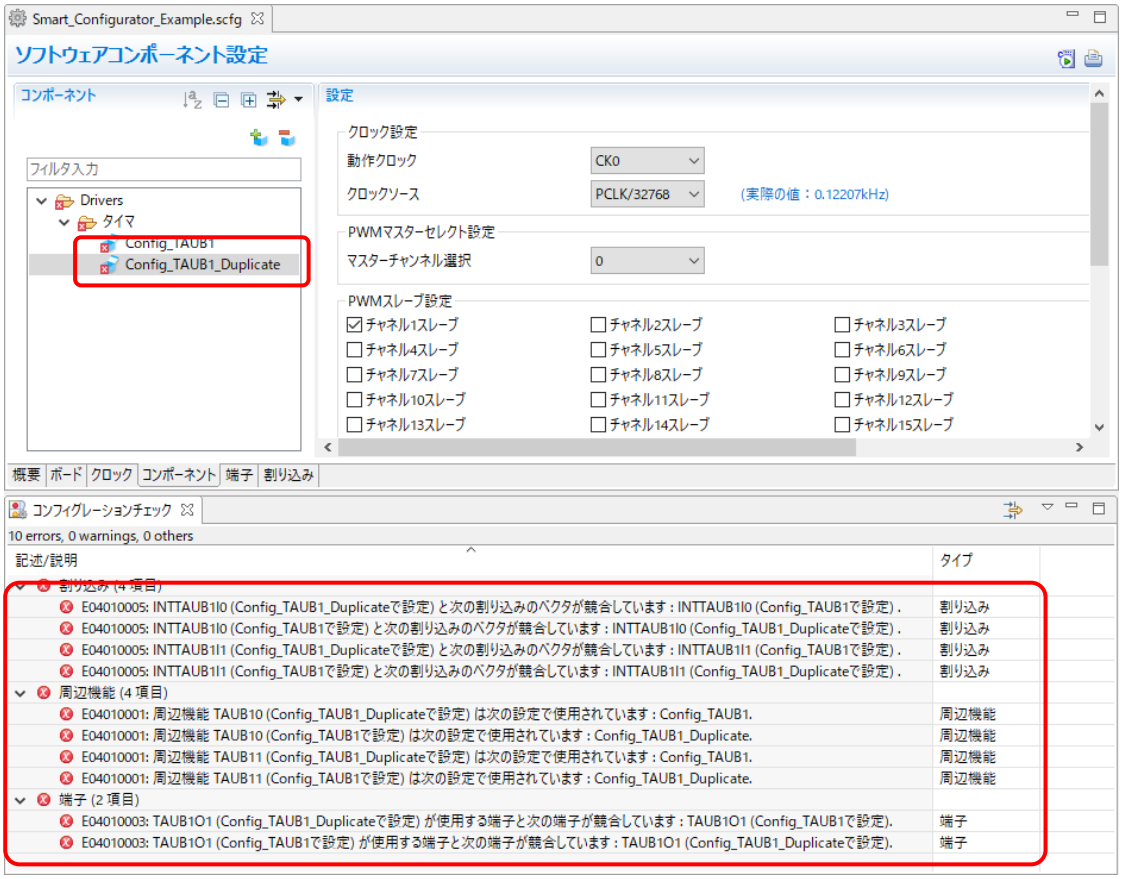


図 5-1 リソースの競合

5.2 端子の競合

端子の競合がある場合、エラーマーク がツリーと端子機能リストに表示されます。

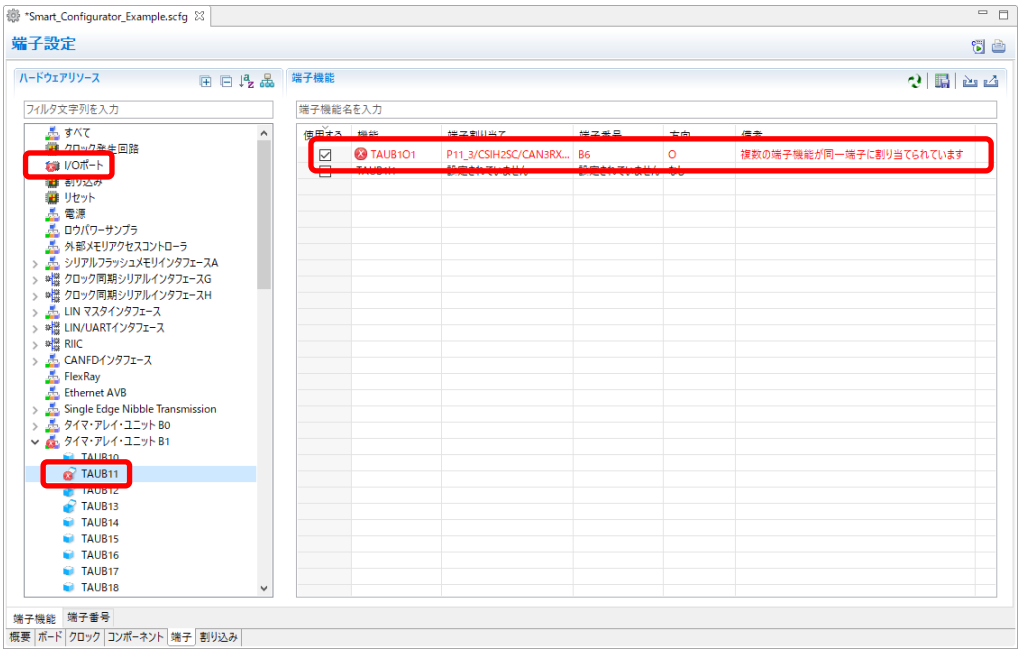


図 5-2 端子の競合

競合情報の詳細は、コンフィグレーションチェックビューに表示されます。

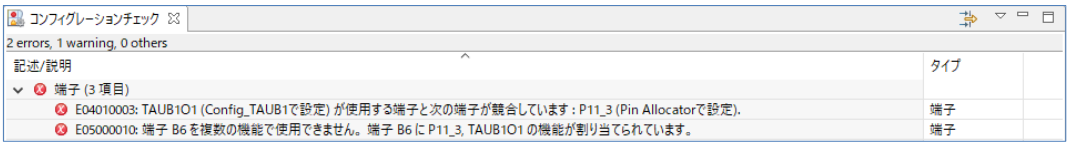


図 5-3 端子競合のメッセージ

エラーマークのあるツリーノードを右クリックし、［競合の解決］を選択して競合を解決してください。

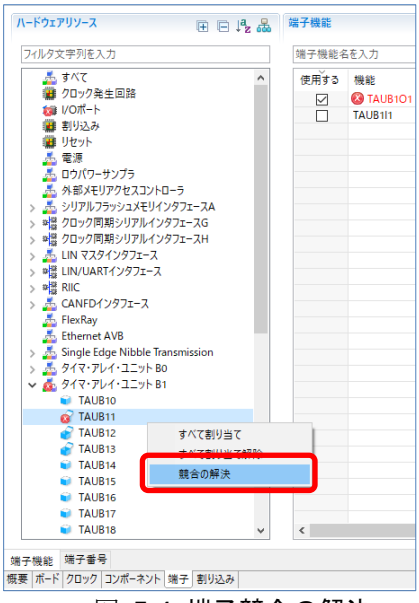


図 5-4 端子競合の解決

選択されたノードの端子は、他の端子に再度割り当てられます。

6. ソースの生成

6.1 生成ソースの出力

スマート・コンフィグレータビューの［コードの生成］ボタンをクリックすると、設定した内容に応じたソースファイルを出力します。

図 6-1 ソースファイルの生成

スマート・コンフィグレータは、<PeojectDir>¥src¥smc_gen にファイルを生成し、プロジェクト・エクスプローラー内のソースファイルを更新します。すでにスマート・コンフィグレータでファイルを生成している場合、バックアップも生成します。（「8 生成ソースのバックアップ」を参照ください。）

【注】ユーザー作成のソースファイルを sms_gen フォルダに置くと、ソースコード生成時に消去されます。

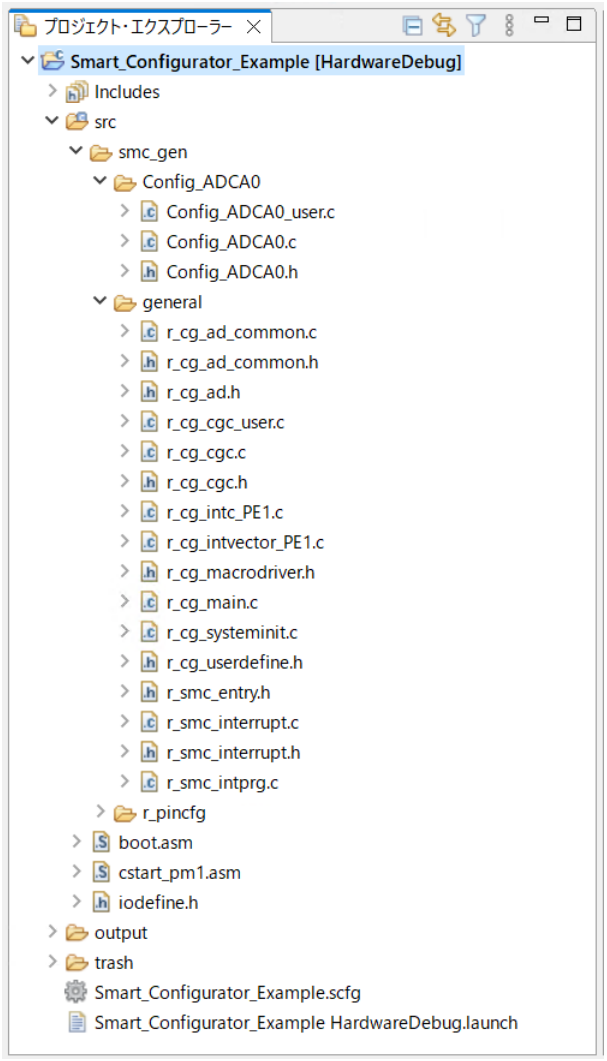


図 6-2 プロジェクト・エクスプローラー内のソースファイル

6.2　ソース生成先の設定

ソースの生成先は、スマート・コンフィグレータの［概要］ページから設定できます。

(1) ［概要］ページの［現在の設定状態］の下にある［編集］ボタンをクリックします。

図 6-3 ソース生成先の変更

(2) ［フォルダの選択］ダイアログで、生成先のフォルダを選択し、［OK］ボタンをクリックします。

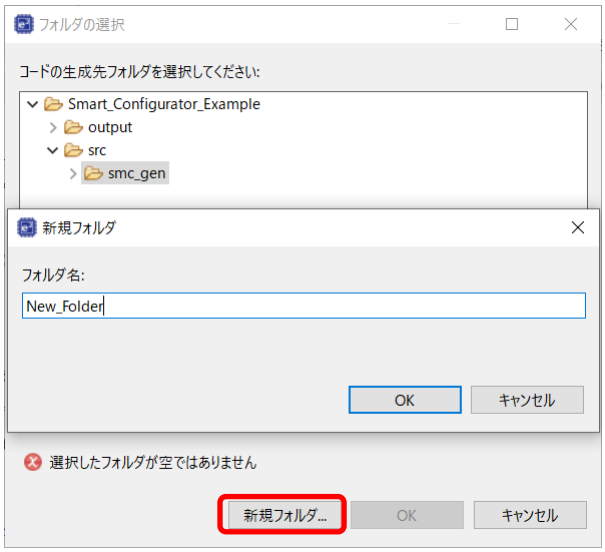


図 6-4 フォルダ選択

(3) ［コード生成  ］ボタンをクリックすると、(2)で選択したフォルダにソースファイルが生成されます。［概要］ページで現在の生成コードの場所を確認することもできます。

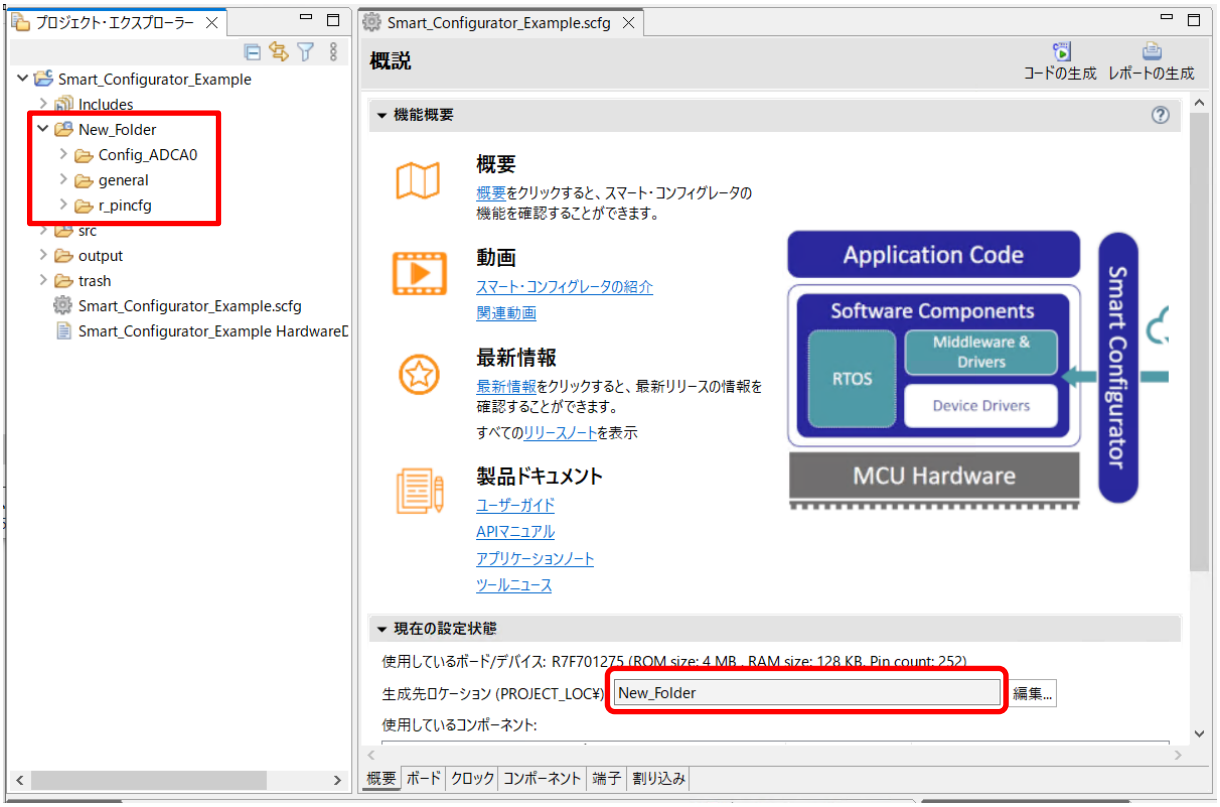


図 6-5 プロジェクト・エクスプローラー内のソースファイル

6.3　生成ファイルの構成とファイル名

スマート・コンフィグレータが出力するフォルダとファイルを下図に示します。なお、*main()*関数は e² studio でプロジェクト作成時に生成する*{Project name}.c* に含まれます。

“ConfigName”はコンポーネント設定で設定したコンフィグレーション名を示します。

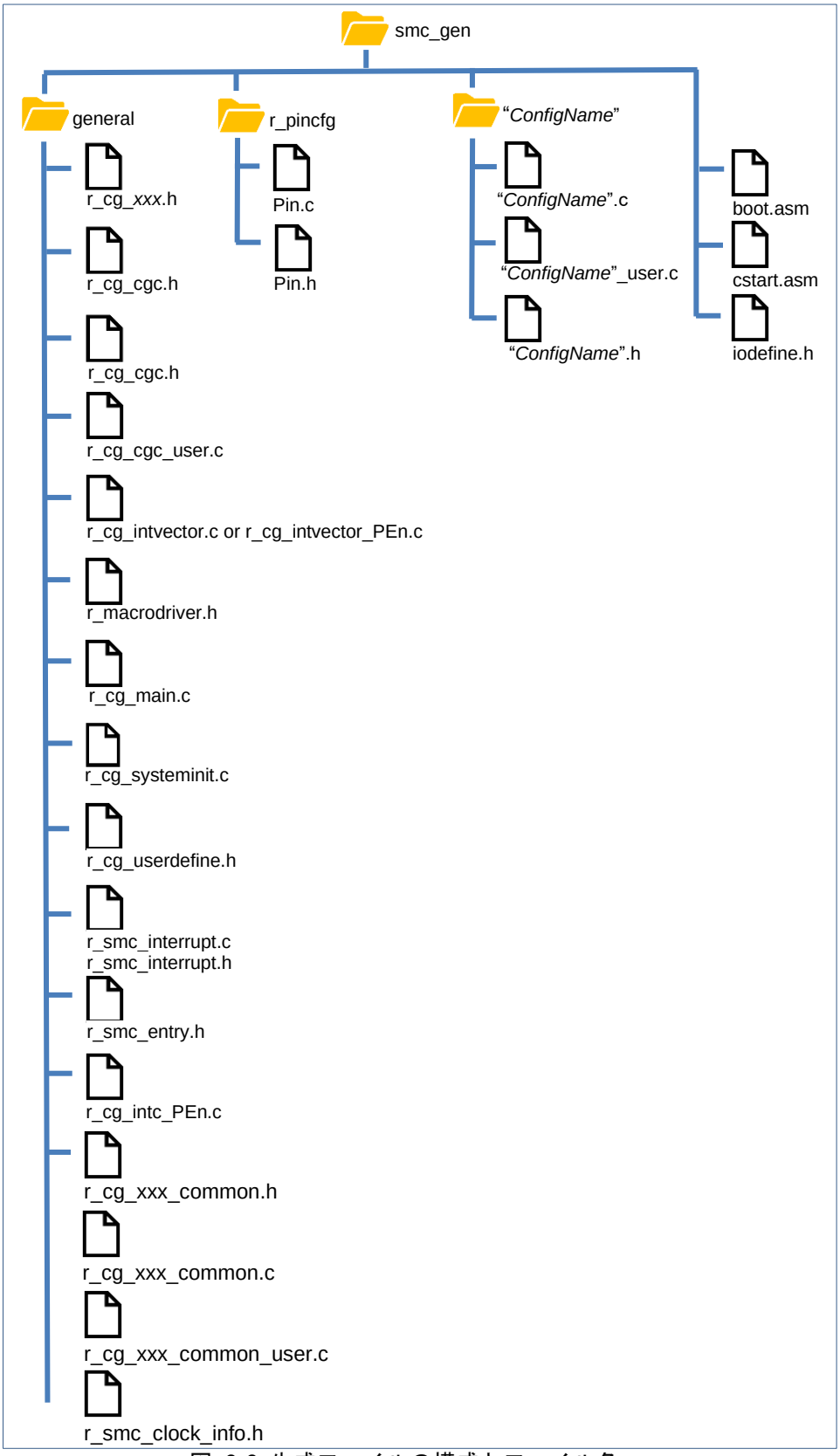


図 6-6 生成ファイルの構成とファイル名

フォルダ	ファイル	説明
general		このフォルダは常時生成されます。同じ周辺機能の CG ドライバで共通に使用される、ヘッダファイルとソースファイルを含みます。
	r_cg_xxx.h(*1)	これらのファイルは常時生成されます。SFR レジスタを設定するためのマクロ定義を含みます。
	r_cg_cgc.c	このファイルは常時生成されます。 クロックページの設定を基にしたクロックソースの初期化を含みます。
	r_cs_cgc.h	このファイルは常時生成されます。 このヘッダファイルは、クロックを初期化するマクロ定義を含みます。
	r_cs_cgc_user.c	このファイルは、CGC 初期化後にユーザーがコードを *R_CGC_Create* に追加する関数を含みます。 ユーザーは、コードと関数を専用のユーザーコード領域に追加することができます。
	r_cg_intvector.c *r_cg_intvector_PEn.c*	このファイルは常時生成されます。割り込みベクタ・テーブルの定義を含みます。
	r_cg_macrodriver.h	このファイルは常時生成されます。 このヘッダファイルは、ドライバで使用される共通のマクロ定義を含みます。
	r_cg_main.c	このファイルは常時生成されます。*main()*関数を定義します。
	*r_cg_* systeminit.*c*	このファイルは常時生成されます。 *R_ConfigName_Create* の名前を持つ全てのドライバの初期化関数を呼び出す *R_Systeminit* を含みます。*R_ Systeminit* は、クロックの初期化も呼び出します。
	r_cg_userdefine.h	このファイルは常時生成されます。 ユーザーは、専用のユーザーコード領域にマクロ定義を追加することができます。
	r_smc_interrupt.c	このファイルは常時生成されます。
	r_smc_interrupt.h	このファイルは常時生成されます。 [割り込み] ページで設定されたすべての割り込みの優先レベルの定義が含まれます。 ユーザーはこれらのマクロ定義をアプリケーション・コードで使用できます。
	r_smc_entry.h	このファイルは常時生成されます。 このファイルは以下のファイルをインクルードします。 r_cg_xxx_common.h r_cg_macrodriver.h r_cg_userdefine.h r_cg_cgc.h {ConfigName}.h このファイルは、r_cg_main.c にインクルードされます。

フォルダ	ファイル	説明
	r_cg_intc_PEn.c	このファイルは以下に対してのみ生成されます。 RH850/U2A： PE PEn(n=0~3) の使用が選択された場合のみ r_cg_intc_PEn.c(n=0~3)が生成されます。 RH850/C1M： r_cg_intc_PE1.c が生成されます。 RH850/U2B： r_cg_intc_PE0.c が生成されます。 このファイルは割り込み初期化 API 定義を含みます。
	r_cg_xxx_common.c(*1)	このファイルは、すべてのコンポーネントで共有される共通設定を持つコンポーネントに対してのみ生成されます。 通常、複数の共有 API が含まれており、ユーザーによって呼び出されます。
	r_cg_xxx_common.h(*1)	r_cg_xxx_common.c および r_cg_xxx_common_user.c のヘッダファイルです。 このファイルは、すべてのコンポーネントで共有される共通設定を持つコンポーネントに対してのみ生成されます。 通常、複数の共有 API の宣言が含まれます。
	r_cg_xxx_common_user.c(*1)	このファイルは、すべてのコンポーネントで共有される共通設定を持つコンポーネントに対してのみ生成されます。 通常、複数の共有割り込みサービスルーチンが含まれています。 ユーザーは、専用のユーザーコード領域にコードと関数を追加できます。
	r_smc_clock_info.h	RH850/U2B 専用に生成されます。 クロックソースのマクロ定義と、[Clock]ページのモジュールクロック設定が含まれています。 このファイルをインクルードすることで、クロック設定マクロを使用することができます。
r_pincfg	*Pin.c*	このファイルは常時生成されます。 ［端子］タブで設定される全周辺機能に使用する端子機能初期化のリファレンスです（I/O ポート以外）。
	Pin.h	このファイルは常時生成されます。 このファイルは、 *Pin.c* での端子設定の関数プロトタイプ シンボル名の定義 シンボル名ユーザーガイド シンボル名 API を含みます。
{ConfigName}		このフォルダは、プロジェクトに追加されるコンポーネント用に生成されます。 このフォルダの API 関数は、*ConfigName*（設定名）の後に命名します。
	{ConfigName}.c	このファイルは、ドライバ(*R_ConfigName_Create*)を初期化する関数、ドライバに特有な操作、例えばスタート(*R_ConfigName_Start*)やストップ(*R_ConfigName_Stop*)を実行する関数を含みます。
	{ConfigName}_user.c	ドライバの初期化(*R_ConfigName_Create*)の後に追加することができる割り込みサービスルーチンと関数を含みます。 ユーザーは、専用のユーザーコード領域にコードと関数を追加することができます。
	{ConfigName}.h	*{ConfigName}.c* と *{ConfigName}_user.c* のヘッダファイルです。

＊1：*xxx* は周辺機能名を意味します。

6.4 クロック設定

クロックページにあるクロックソースの設定は、¥src¥smc_gen¥general フォルダに生成されます。

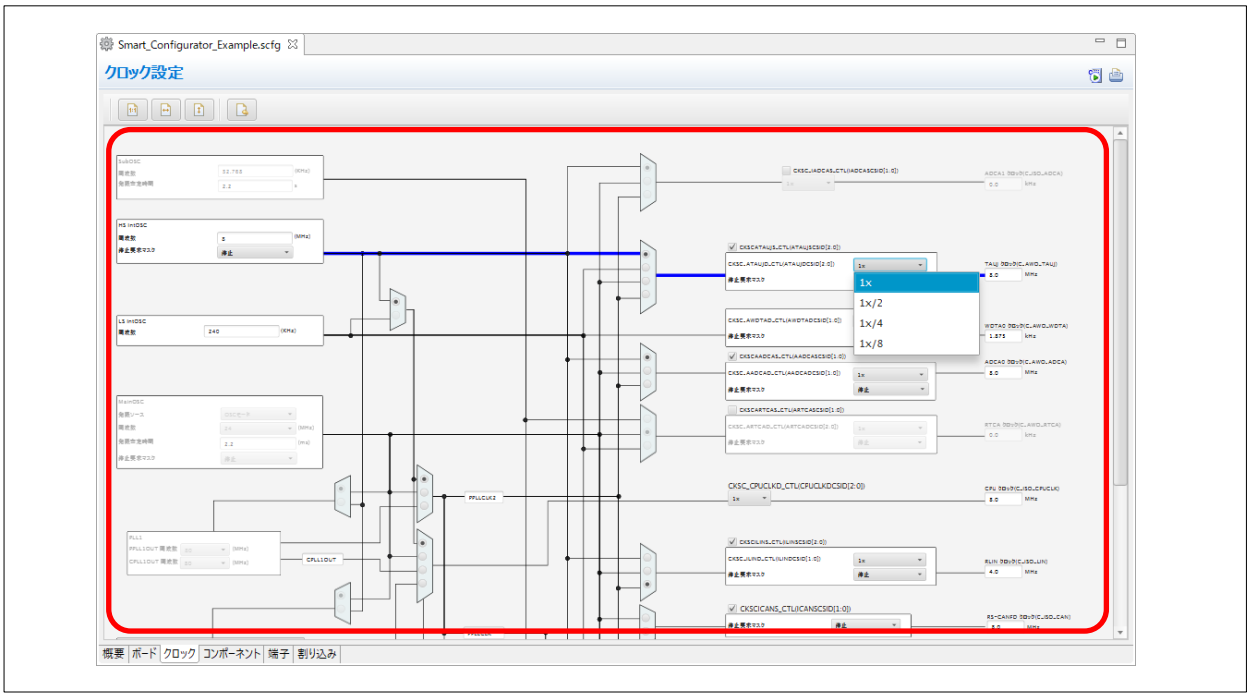


図 6-7 メインクロックをクロックソースとして選択した場合のクロック設定

No	フォルダ	ファイル	マクロ/関数	説明
general		*r_cg_cgc.c*	*R_CGC_Create*	この API 関数は、クロックを初期化します。*r_cg_systeminit.c* の *R_Systeminit* は、*main()*関数から、この関数を呼び出します。
		r_cg_cgc.h	クロックに関連するマクロ	これらのマクロは、*R_CGC_Create* のクロックの初期化に使用されます。
		r_cg_cgc_user.c	*R_CGC_Create_UserInit*	CGC 初期化後に、ユーザーが *R_CGC_Create* にコードを追加する際にこの API 関数を使用します。

6.5　端子設定

端子設定は、コンポーネントにより下記(1)から(2)に示すようなソースファイルに生成されます。

(1) {ConfigName}を使用したドライバの端子初期化

端子機能は¥src¥smc_gen¥{ConfigName}¥{ConfigName}.c の R_ConfigName_Create で初期化されます。

端子初期化コードは、main()で処理されます。

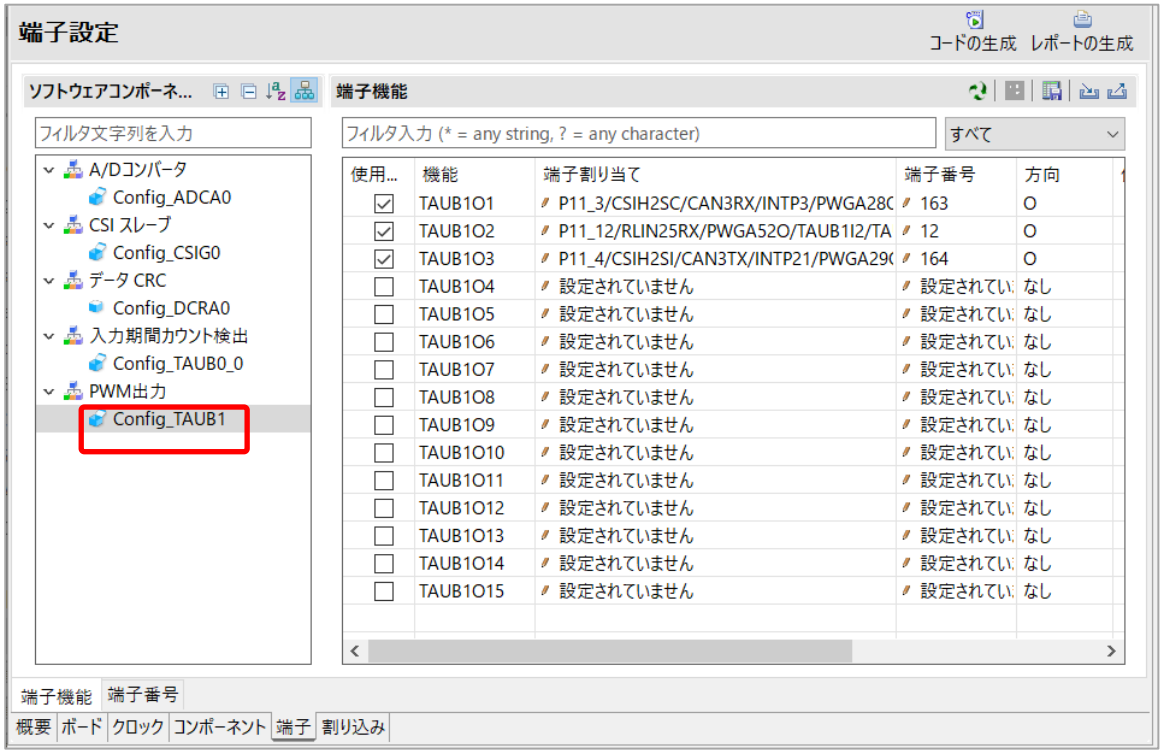


図 6-8 Config_TAUB1 の端子設定

フォルダ	ファイル	関数	説明
{ConfigName}	*{ConfigName}.c*	*R_ConfigName_Create*	このドライバが使用した端子を API 関数が初期化します。*main()*関数から、*r_cg_systeminit.c* の *R_Systeminit* はこの関数を呼び出します。

(2) 端子初期化コードの参照

プロジェクトで使用されるすべての周辺端子機能については、¥src¥smc_gen¥r_pincfg フォルダにある Pin.c を参照してください（I/O ポート以外）。

フォルダ	ファイル	関数	説明
r_pincfg	*Pin.c*	*R_Pins_Create*	［端子］ページで設定された、I/O ポート以外の全端子機能の初期化コードを含みます。

6.6 割り込み設定

割り込みページの設定は、いくつかのソースファイルに生成されます。

RH850/C1M, F1KM, F1KH and U2B:

RH850/U2A:

図 6-9 割り込み設定

RH850/F1KM, F1KH and U2B:

No	項目	フォルダ	ファイル	説明
(1)	優先レベル	**{ConfigName}**	*{ConfigName}.c*	割り込み優先レベル設定は、このファイルの *R_ConfigName_Create* で初期化されます。この関数は、*main()*関数の中から *r_cg_systeminit.c* の *R_Systeminit* によって呼ばれます。
(2)	OS 管理	**{ConfigName}** **or** **general**	*{ConfigName}_user.c* *or* *r_cg_xxx_common_user.c*	このファイルで定義されている割り込み関数を OS 管理の割り込み書式で出力します。
(3)	割り込みハンドラ エンティティを生成する	**general**	*r_smc_intprg.c* *r_cg_intvector_PE0.c*	[エンティティを生成する]にチェックを入れると、[割り込みハンドラ]に表示される割り込みハンドラのエンティティが「r_smc_intprg.c」に生成されます。
(4)	有効化/無効化関数を生成する	**general**	*r_smc_interrupt.c* *r_smc_interrupt.h*	[有効化/無効化関数を生成する]にチェックを入れると、有効化/無効化関数が r_smc_interrupt.c に生成されます。

RH850/C1M, U2A:

No	項目	フォルダ	ファイル	説明
(1)	優先レベル	**general**	*r_cg_intc_PEn.c*	割り込み優先レベル設定は、このファイルの *R_Interrupt_Initialize_ForPE*()関数で設定されます。この関数は、*main()*関数の中から *r_cg_systeminit.c* の *R_Systeminit()*関数によって呼ばれます。
(2)	OS 管理	**{ConfigName}** **または** **general**	*{ConfigName}_user.c* または *r_cg_xxx_common_user.c*	このファイルで定義されている割り込み関数を OS 管理の割り込み書式で出力します。
(3)	割り込みハンドラ エンティティを生成する	**general**	*r_smc_intprg.c* *r_cg_intvector_PE0.c*	[エンティティを生成する]にチェックを入れると、[割り込みハンドラ]に表示される割り込みハンドラのエンティティが「r_smc_intprg.c」に生成されます。
(4)	有効化/無効化関数を生成する	**general**	*r_smc_interrupt.c* *r_smc_interrupt.h*	[有効化/無効化関数を生成する]にチェックを入れると、有効化/無効化関数が *r_smc_interrupt.c* に生成されます。
(5)	PE*n* (RH850/U2A のみ)	**general**	*r_cg_intc_PEn.c*	PE 選択の設定は、このファイルの *R_Interrupt_Initialize_ForPE* で初期化されます。この関数は、*main()*関数の中から *r_cg_systeminit.c* の *R_Systeminit*()関数によって呼ばれます。

7. ユーザープログラムの作成

スマート・コンフィグレータのコンポーネントタイプには［コード生成］があります。ここでは、［コード生成］のカスタムコード追加方法について説明します。

7.1 コード生成の場合のカスタムコード追加方法

コンポーネントのタイプで［コード生成］を選択した場合、ソースコード出力の際、同一ファイルが存在する場合には、以下のコメントで囲まれた部分に限り、該当ファイルをマージします。

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


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

図 7-1 コード追加用のコメント

［コード生成］の場合、特定の周辺機能ごとに 3 つのファイルを生成します。デフォルトのファイル名は、「Config_xxx.h」、「Config_xxx.c」、「Config_xxx_user.c」となり、xxx は周辺機能を表します。（例えば、A/D コンバータ(リソース ADCA0)の場合、xxx は“ADCA0”になります。）カスタムコードを追加するためのコメントは、3 つのファイルそれぞれの先頭と最後に設けられる他、「Config_xxx_user.c」にある周辺機能の割り込み関数内にも追加されます。以下に ADCA0 の例（Config_ADCA0_user.c）を示します。

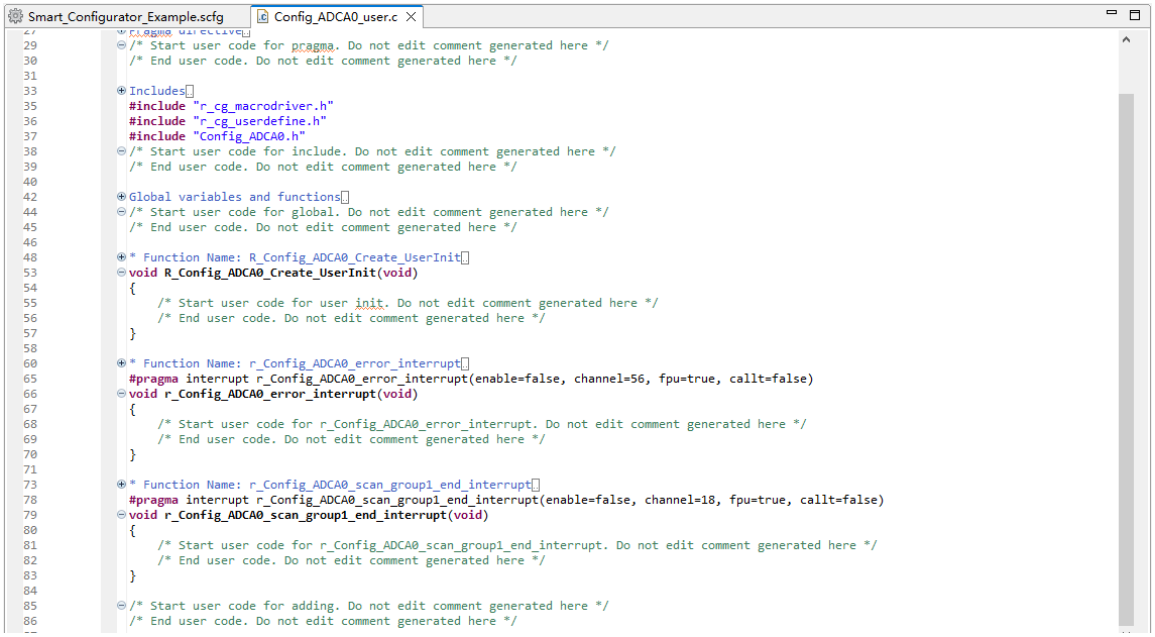


図 7-2 生成コードの例

7.2　ユーザーアプリケーションコードの使用

生成されたコードを使用するには、以下の手順で行います。

r_cg_main.c ファイルを開き、main 関数で生成された関数を呼び出します。

コード生成の場合、端子初期化を含むドライバ初期化関数（R_ConfigName_Create）は、デフォルトで r_cg_hardware_setup.c の R_Systeminit 関数で呼び出されます。

ドライバ固有の処理を実行するには、アプリケーションコードを追加する必要があります。 例えば、開始（R_*ConfigName*_Start）および停止（R_*ConfigName*_Stop）。

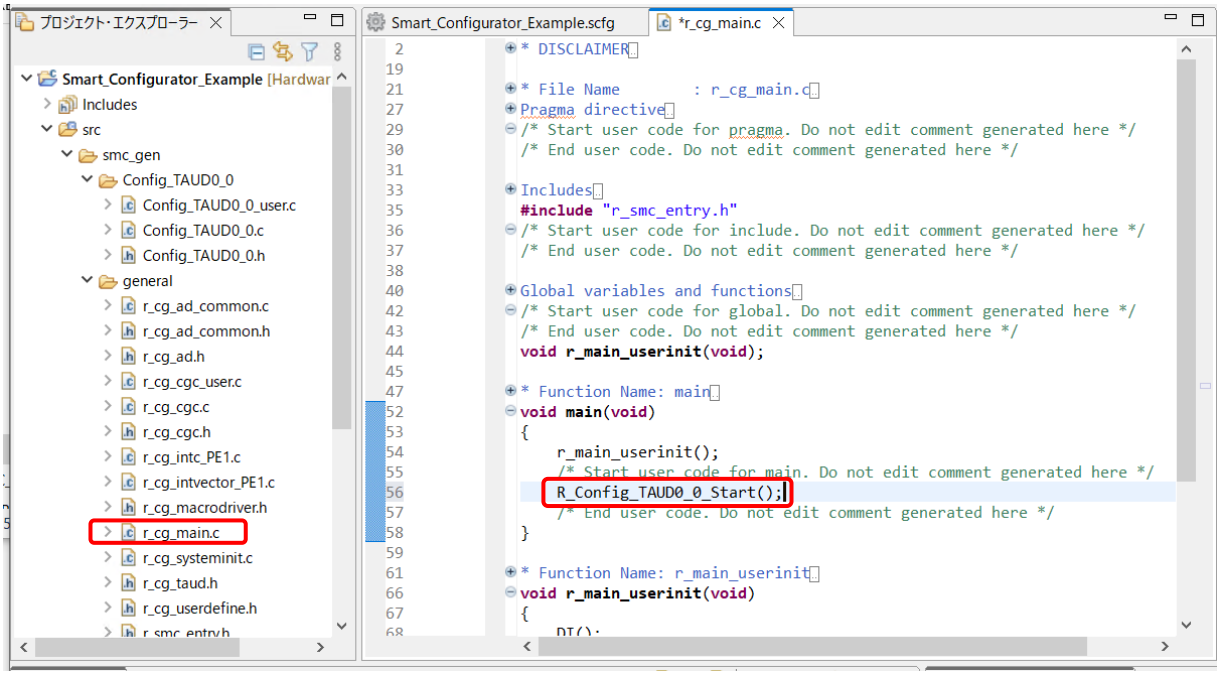


図 7-3 コード生成関数コール

8.　生成ソースのバックアップ

スマート・コンフィグレータには、ソースコードのバックアップ機能があります。

［コードの生成］ ボタンをクリックしてコードの生成を行うとき、スマート・コンフィグレータは生成ソースのバックアップを生成します。<Date-and-Time>は、コード生成を実行しバックアップフォルダを作成した日時です。

<ProjectDir>¥trash¥<Date-and-Time>

9. レポートの生成

スマート・コンフィグレータは、設定情報をレポートで提供します。レポートを生成するには、以下の手順で行います。

9.1 全設定内容レポート

スマート・コンフィグレータビューで［レポートの生成］ボタンをクリックし、レポートを出力します。出力形式は 、PDF とテキストの 2 種類が選択可能です。

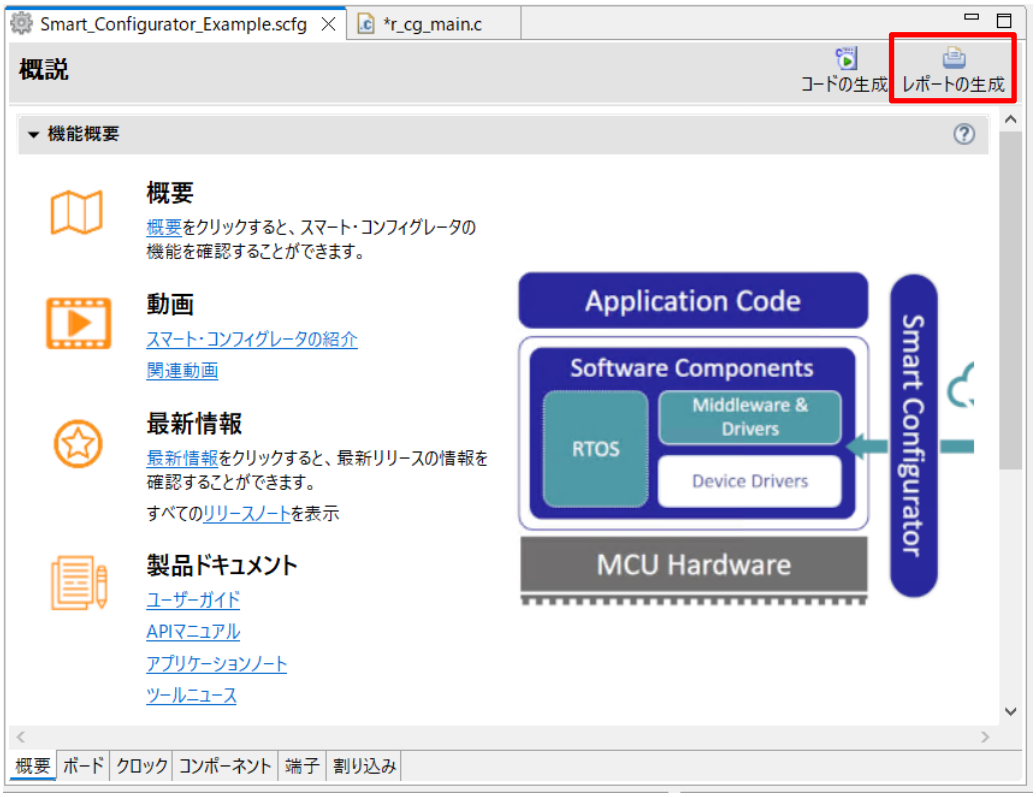


図 9-1 全設定内容レポート出力(txt 形式)

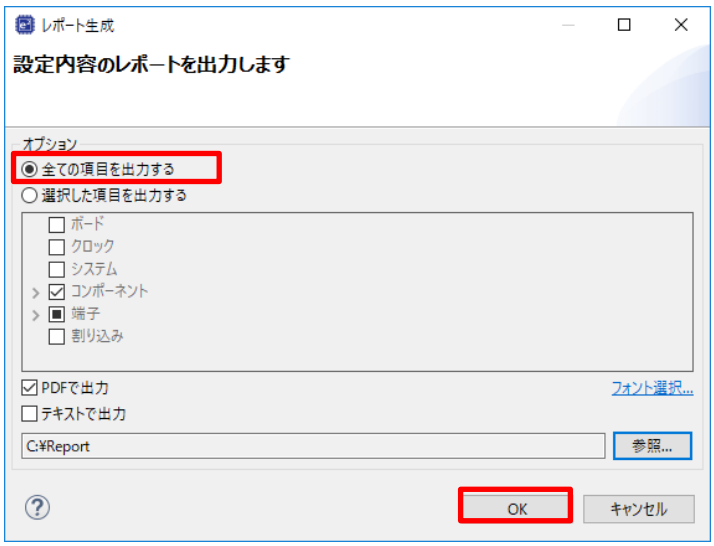


図 9-2 レポート出力ダイアログ

9.2　端子機能リスト、端子番号リスト設定内容(csv 形式)

スマート・コンフィグレータビューの［端子］ページで［.csv ファイルにリストを保存］ボタンをクリックし、端子機能リスト、端子番号リスト設定内容を出力します。

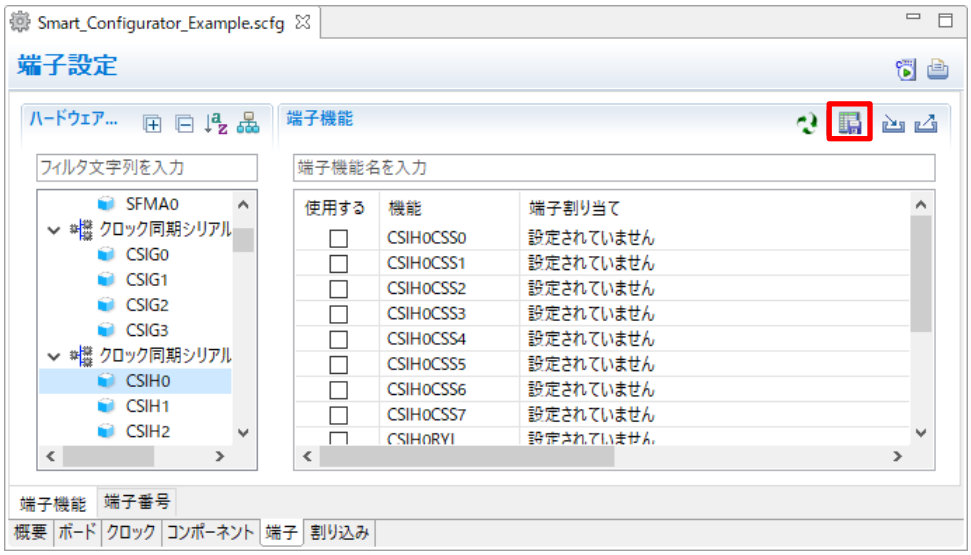


図 9-3 端子機能リスト、端子番号リスト出力(csv 形式)

9.3　MCU/MPU パッケージ図(png 形式)

MCU/MPU パッケージビューの［端子配置図を保存］ボタンをクリックし、MCU/MPU パッケージ図を出力します。

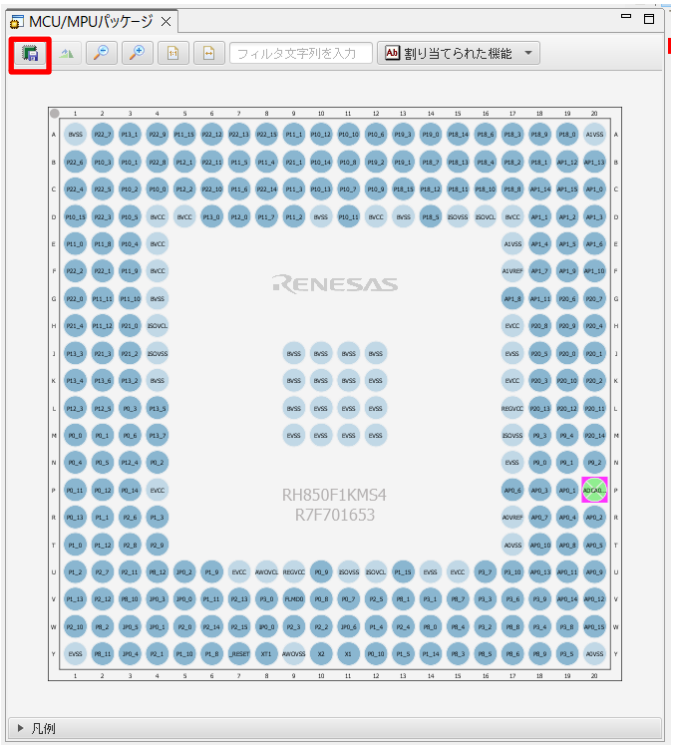


図 9-4 MCU/MPU パッケージ図出力(png 形式)

10. ユーザーコード保護機能

図 10-1 の指定タグを追加することで、任意の位置にユーザーコードを追加できます。追加されたユーザーコードはコード生成時に保護されます。

ユーザーコード保護機能は「コード生成コンポーネント」が生成したファイル、クロック生成ファイル、割り込み生成ファイルのみサポート対象となります。

10.1 ユーザーコード保護機能の指定タグ

ユーザーコード保護機能を使用する場合、図 10-1 のように、/* Start user code */ と /* End user code */ を挿入し、このタグの間にユーザーコードを追加してください。指定タグが完全に一致しない場合は、保護されません。

```
/* Start user code */
コメントの間にユーザーコードを追加
/* End user code */
```

図 10-1 ユーザーコード保護機能の指定タグ

10.2 ユーザーコード保護機能の使用例

図 10-2 に示すように、図 10-1 の指定タグを使用し、PWM 出力モジュールの Create 関数の中に新しいユーザーコードを挿入します。その後、PWM 出力の GUI での設定を更新し、再びコード生成すると、挿入されたユーザーコードが新たに生成されたファイルに自動的にマージされます。

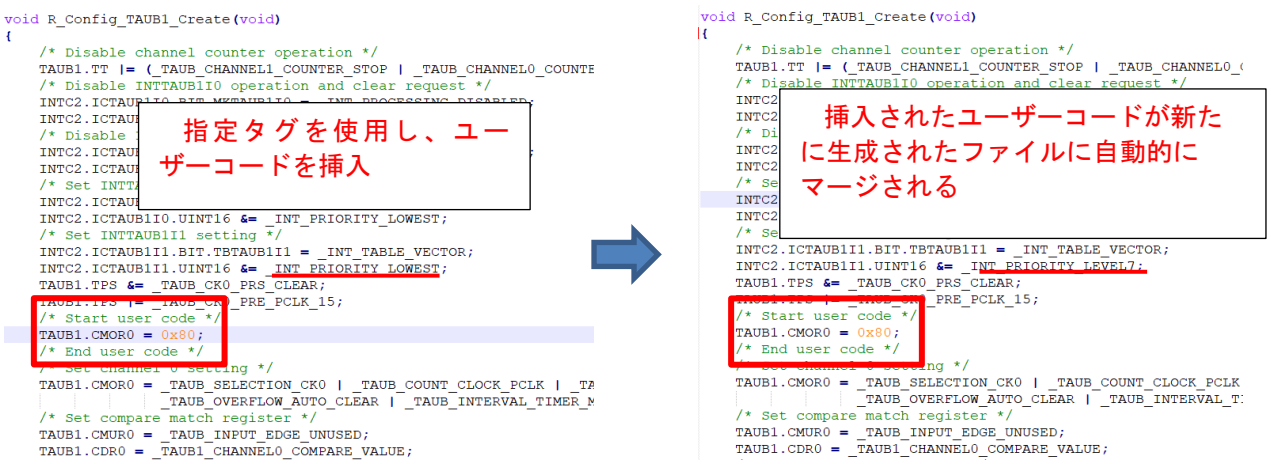


図 10-2 ユーザーコードの保護機能

10.3　競合発生時の対応方法

10.3.1　競合の発生条件

挿入したユーザーコードの前後にある生成コードに変更がある場合(GUI の設定変更、スマート・コンフィグレータのバージョンアップなど)、図 10-3 のように生成コードに競合が発生します。

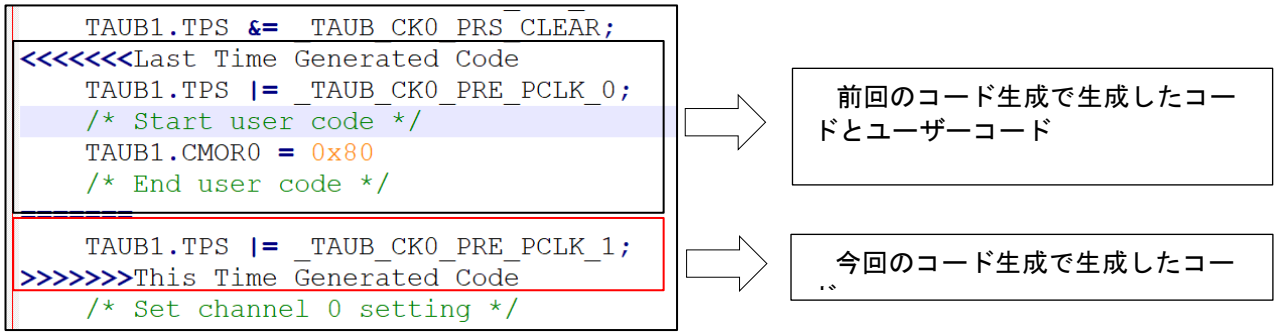


図 10-3 生成された競合コード

競合が発生した場合、コンソールに図 10-4 のようなメッセージが表示されます。

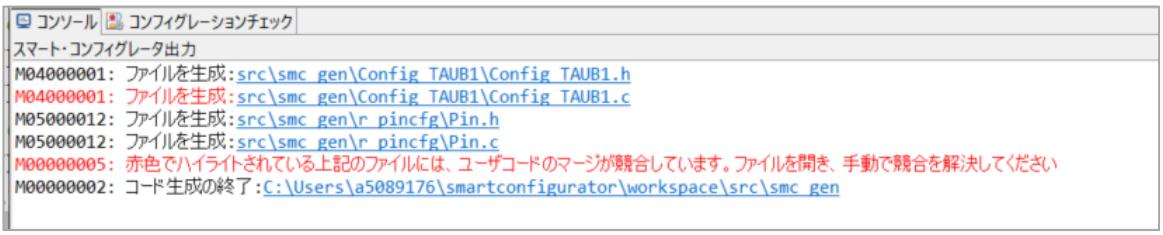


図 10-4 競合のメッセージ

10.3.2　競合の解決方法

競合を解決するには、競合が発生したファイルを開いて、下記の手順に従って手動でコードを修正してください。

1) コンソールで競合しているファイルをクリックし、[File Compare]ビューを開きます。(図 10-5)
2) 「左から右へ現在の変更をコピー」をクリックします。(図 10-5)
3) 未使用のコードを削除します。(図 **10-6**)
4) 変更後のコードを保存します。(図 **10**-7)

【注】左側パネルのコードを右側パネルにコピーするか、右側パネルのコードを直接編集することで、競合を手動で解決することもできます
競合が解決された後も、競合メッセージをクリックすると、[File Compare]ビューを開くことができます。

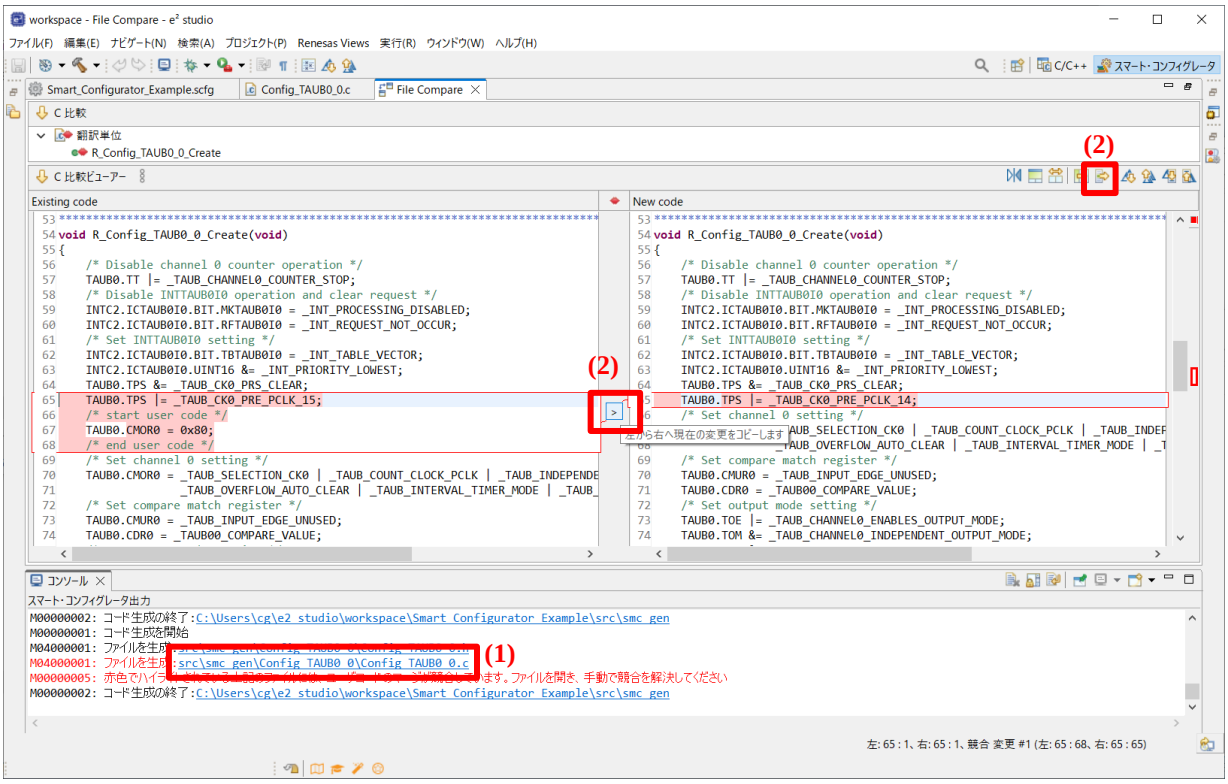


図 10-5　生成された競合コード

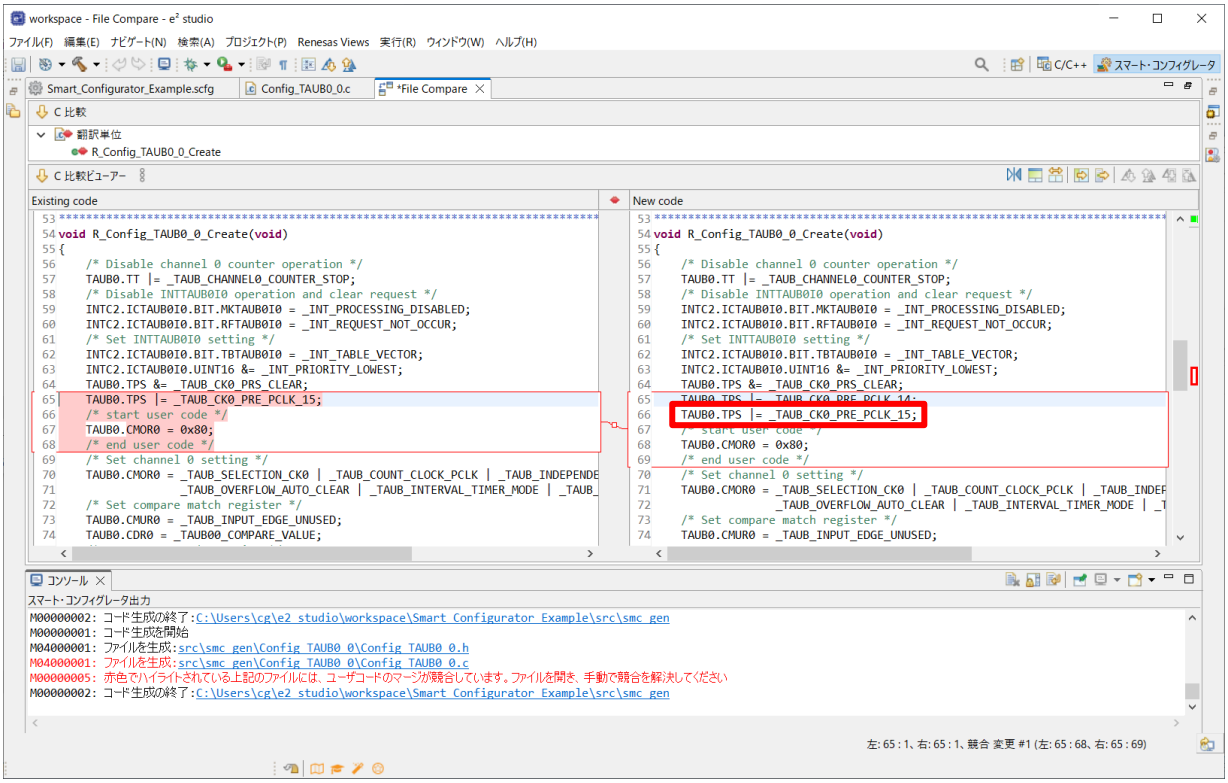


図 **10-6**　「左から右へ現在の変更をコピー」後のコード

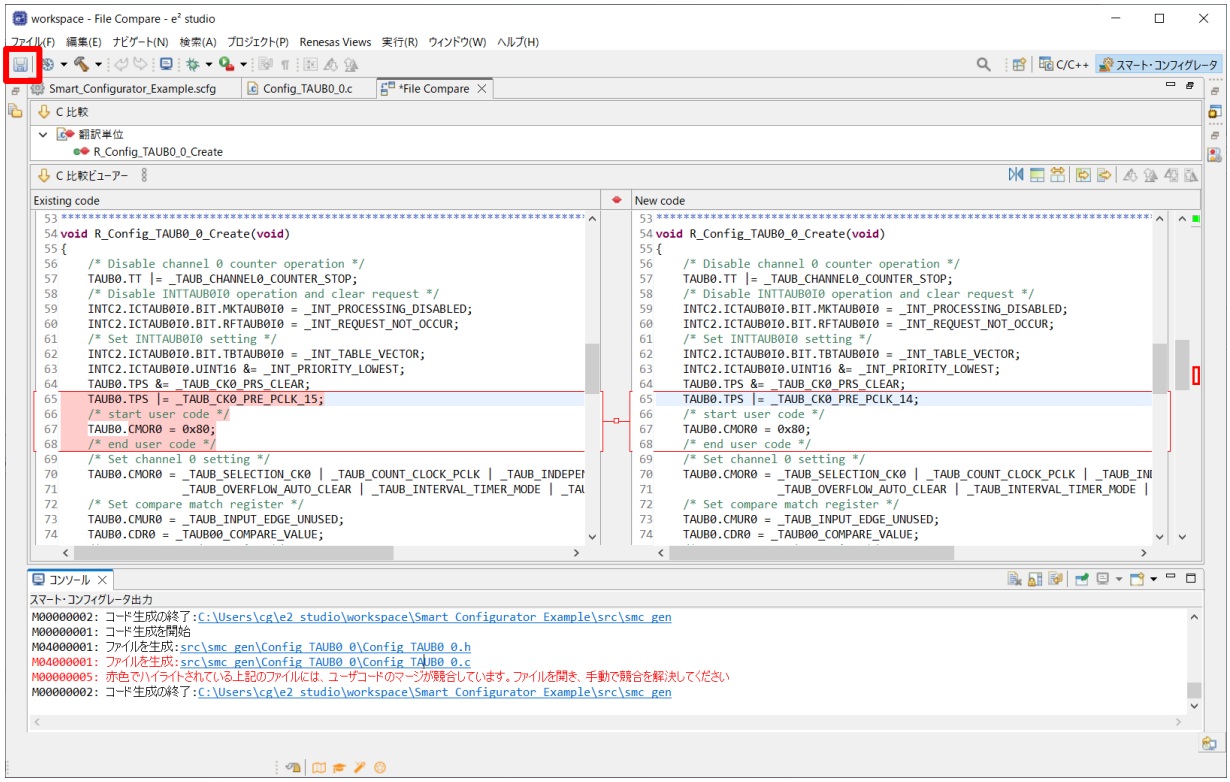


図 **10-7**　競合を解決した後のコード

11. ヘルプ

11.1 ヘルプ

スマート・コンフィグレータの詳細情報は、e² studio メニュー上のヘルプを参照ください。

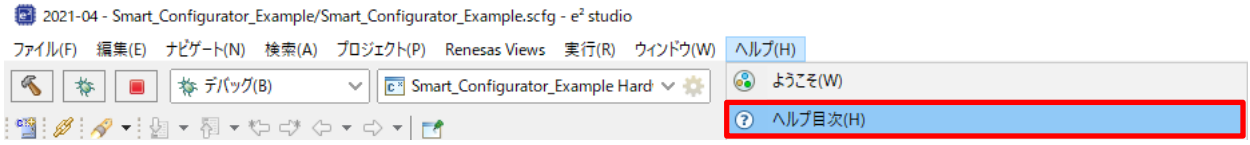


図 11-1 ヘルプ表示

「概要」ページの アイコンクリックにより、ヘルプを参照できます。

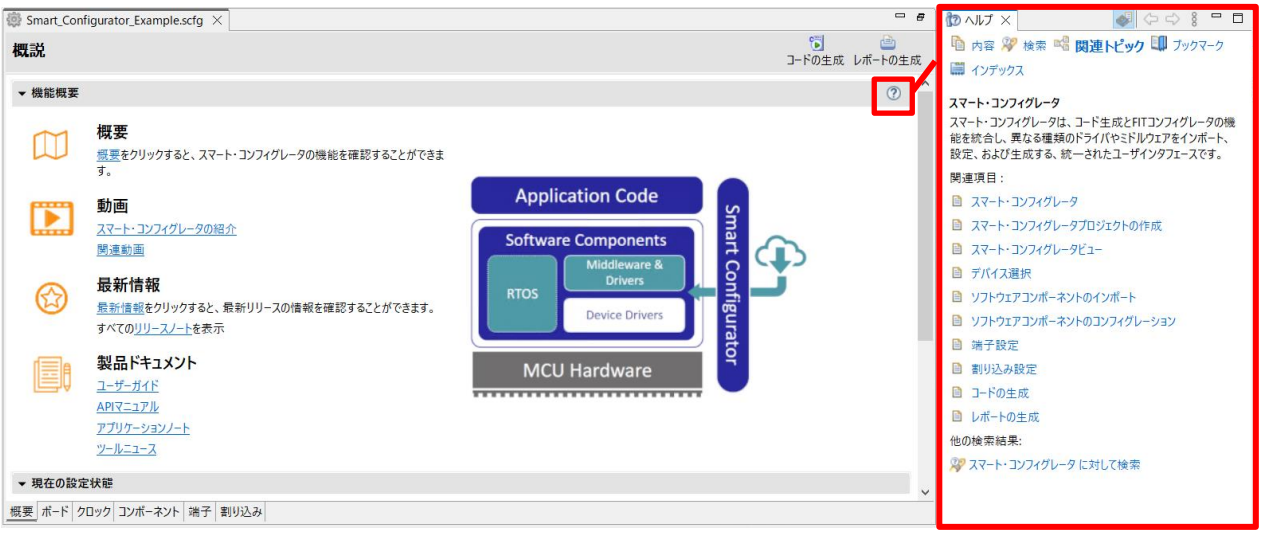


図 11-2 クイックスタート

12. 参考ドキュメント

ユーザーズマニュアル：ハードウェア

最新版をルネサスエレクトロニクスホームページから入手してください。

テクニカルアップデート／テクニカルニュース

最新の情報をルネサスエレクトロニクスホームページから入手してください。

ユーザーズマニュアル：開発環境

統合開発環境 e2 studio ユーザーズマニュアル 入門ガイド (R20UT2858)

CC-RH コンパイラ ユーザーズマニュアル (R20UT3516)

スマート・コンフィグレータ ユーザーズマニュアル RH850 API リファレンス編 (R20UT4361)

（最新版をルネサスエレクトロニクスホームページから入手してください。）

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
http://www.renesas.com

お問合せ先
http://www.renesas.com/contact/

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2024.01.20	ー	新規作成

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。

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。