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

R20AN0536JJ0104
Rev.1.04
2024.04.22

要旨

本ユーザーガイドは、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とスマート・コンフィグレータが生成したソースファイルを IAR 社製統合開発環境、GHS 社製統合環境にインポートするまでの手順について説明します。

スマート・コンフィグレータおよび統合開発環境の対象バージョンは以下の通りです。

- RH850 スマート・コンフィグレータ V1.11.0 以降
- IAR Embedded Workbench for RH850 V3.10 以降
- GHS MULTI V8.14 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

サンプル・プロジェクト

スマート・コンフィグレータのインストールフォルダには、RH850/F1KM 用の下記サンプル・プロジェクトが含まれています。

- IAREW, IAR ICC 用サンプル・プロジェクト
- MULTI, GHS CCRH850 用サンプル・プロジェクト

詳しくは、下記説明書をご参照ください。

スマート・コンフィグレータ RH850/F1KM 用サンプル・プロジェクト説明書（R01AN4422）

目次

1. 概要

1.1 目的

本ユーザーガイドは、スマート・コンフィグレータの基本操作と生成したソースファイルを IAR 社製統合開発環境、GHS 社製統合環境にインポートするまでの手順について説明します。

統合開発環境のインストール、使い方に関しては、各社統合開発環境のマニュアルをご参照ください。

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 2 つ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。

2. インストールとアインストール

インストールとアインストールについて説明します。

2.1 スマート・コンフィグレータのインストール

下記 URL から「RH850 スマート・コンフィグレータ」をダウンロードしてください。

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

インストーラ起動後、インストーラの手順に従ってインストールしてください。インストールは管理者権限で行ってください。

2.2 スマート・コンフィグレータのアンインストール

スマート・コンフィグレータのアンインストールを行う場合、コントロールパネルの［プログラムと機能］から、［Smart Configurator for RH850］を選択しアンインストールしてください。

2.3 サンプル・プロジェクトの準備

スマート・コンフィグレータは、main 関数とスマート・コンフィグレータのコンポーネントで設定した各周辺機能の初期化を行うソースファイルを出力します。マイクロコントローラをリセットしたあと、main 関数を実行する前に行う初期化処理，main 関数の起動などの処理を行うスタートアップ・ルーチンは出力しません。

そのため、スマート・コンフィグレータで設定した周辺機能とユーザー・アプリケーションをすぐにビルドできるよう、サンプルのスタートアップ等を含むサンプル・プロジェクトを用意しています。

以下のパスに格納されている説明書を参考に、サンプル・プロジェクトからプロジェクトを作成してください。

IAREW:

“C:¥Program Files (x86)¥Renesas Electronics¥SmartConfigurator¥RH850¥RH850F1KM_SampleProjects¥SC_IAREW”

GHS:

“C:¥Program Files (x86)¥Renesas Electronics¥SmartConfigurator¥RH850¥RH850F1KM_SampleProjects¥SC_MULTI”

3. スマート・コンフィグレータの操作方法

3.1 操作手順

スマート・コンフィグレータでソースファイルを生成し、IAR Embedded Workbench、GHS MULTI に取り込むまでの手順を**図 3-1 操作手順**に示します。サンプル・プロジェクトをご使用になる場合は、「スマート・コンフィグレータ RH850/F1KM 用サンプル・プロジェクト説明書（R01AN4422）」を参照してください。IAR Embedded Workbench、GHS MULTI の操作については、IAR 社、GHS 社の関連ドキュメントを参照してください。

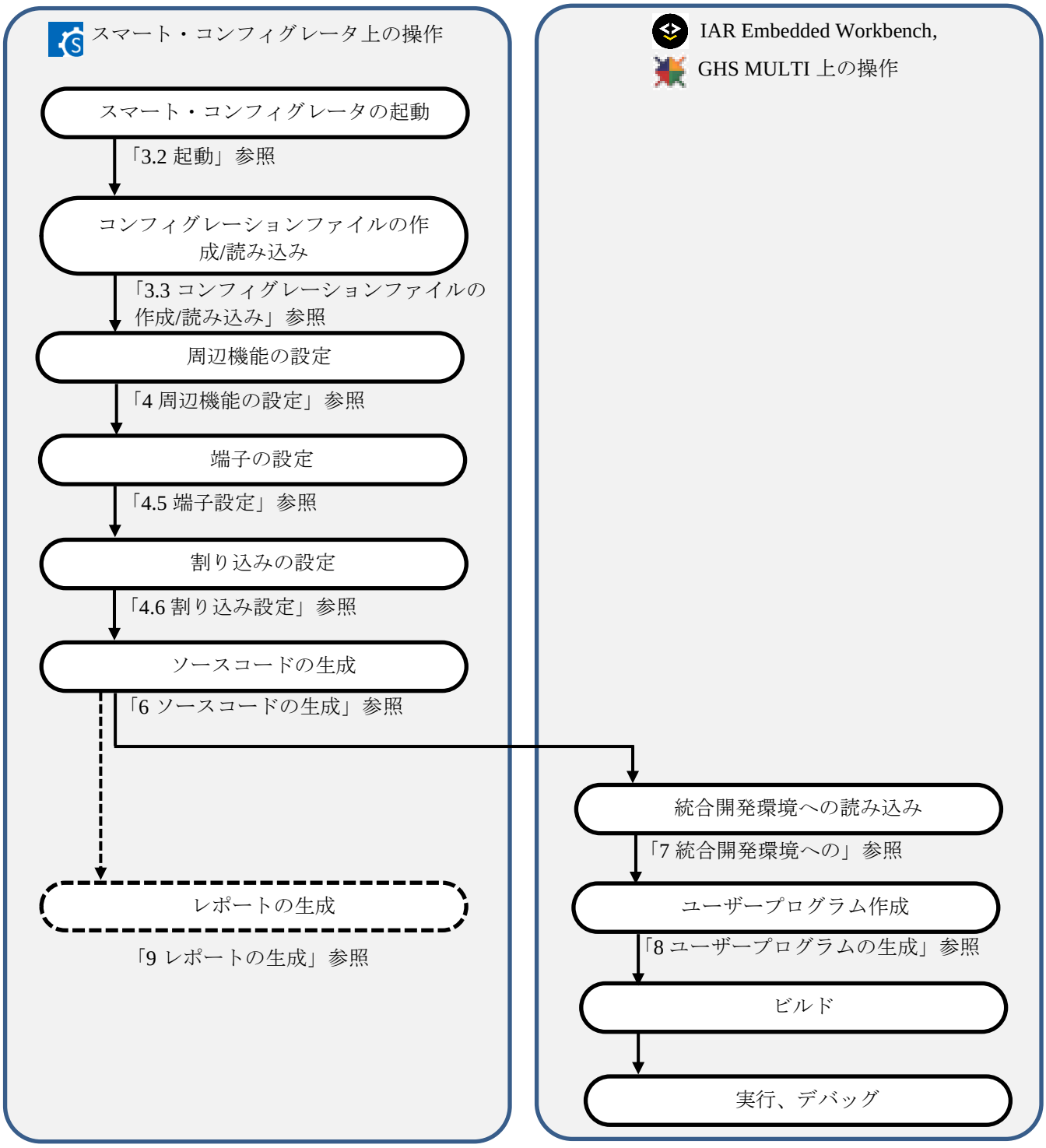


図 3-1 操作手順

3.2　起動

Windows スタートメニューから［Renesas Electronics Smart Configurator］→［Smart Configurator for RH850 Vx.x.x］を選択します。選択後、スマート・コンフィグレータのメインウィンドウが起動します。

【注】Vx.xx はご使用のバージョンに読み替えてください。

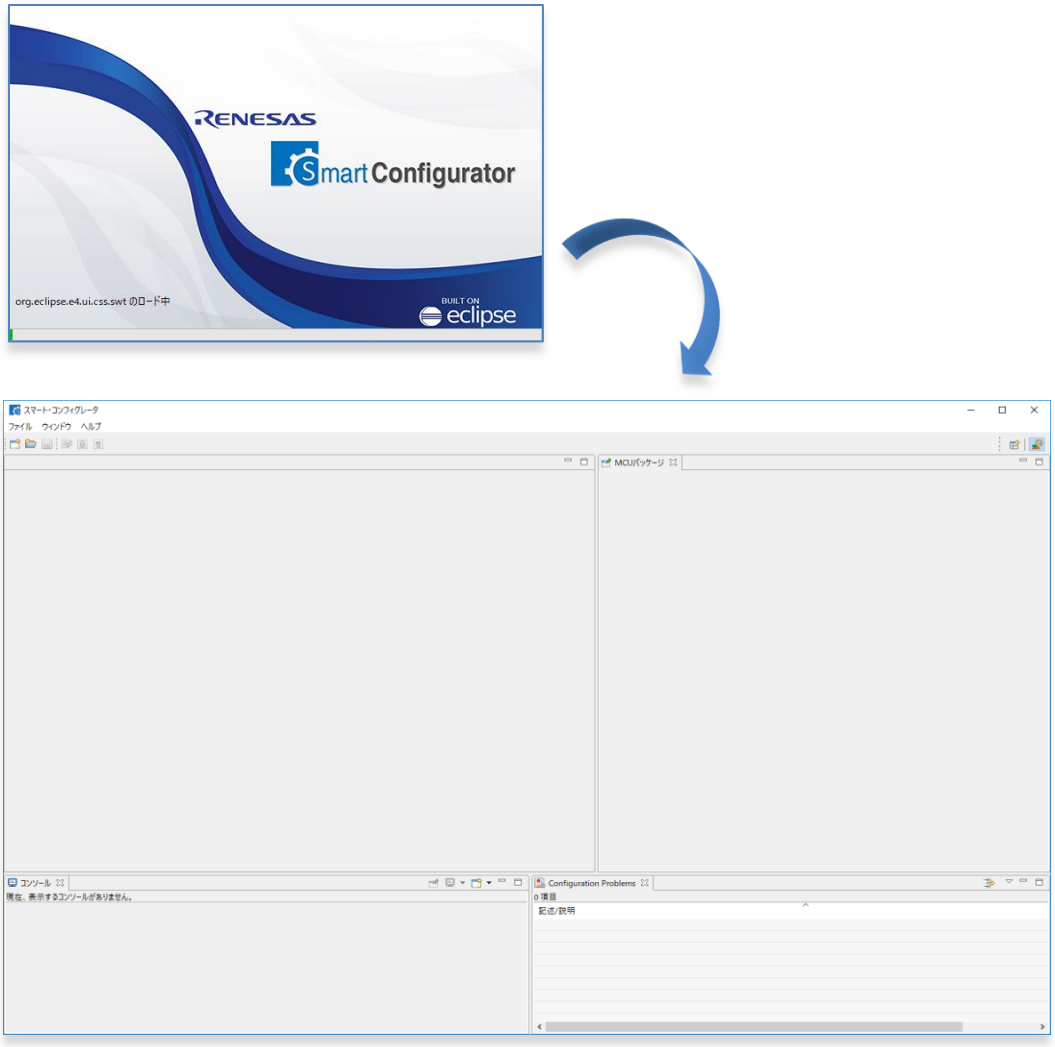


図 3-2 スマート・コンフィグレータの起動

3.3　コンフィグレーションファイルの作成/読み込み

スマート・コンフィグレータは、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コンフィグレーションファイル(*.scfg)に保存し、参照します。

3.3.1　新規作成

メインツールバーの  ［新規コンフィグレーションファイル］ボタンをクリックするとダイアログが表示されます。

(1) ［プラットフォーム:］で、デバイスを選択します。

(2) ［ツールチェーン:］で、ツールチェーンを選択します。

IAR 社製コンパイラをご使用の場合は、［IAR RH850 Toolchain］を選択してください。

GHS 社製コンパイラをご使用の場合は、［GHS RH850 Toolchain］を選択してください。

すべてのコンパイラを使用する場合は、「All Toolchain (CC-RH, GHS, IAR)」を選択してください。

「All Toolchain (CC-RH, GHS, IAR)」を選択時の使用方法は、“7.1.2「All Toolchain (CC-RH, GHS, IAR)」選択時”および“7.2.2「All Toolchain (CC-RH, GHS, IAR)」選択時”を参照してください。

(3) ［ファイル名:］に、ファイル名を入力します。

(4) ［ロケーション:］を確認します。変更したい場合は、［参照］をクリックして保存先を選択してください。

【注】 [コード生成] ボタンをクリックすると、IAR Embedded Workbench の buildinfo.ipcf ファイル、または MULTI の default.gpj、project.gpj、sc_file.gpj ファイルがここに生成されます。IAR Embedded Workbench の*.eww, *.ewp, *.ewd ファイルは初回のコード生成でのみ生成されます。

(5) RTOS を設定する場合は、［次へ］をクリックします。

［IAR RH850 Toolchain］選択時、RTOS は設定できません。

(6) ［終了］をクリックして、コンフィグレーションファイルを作成します。

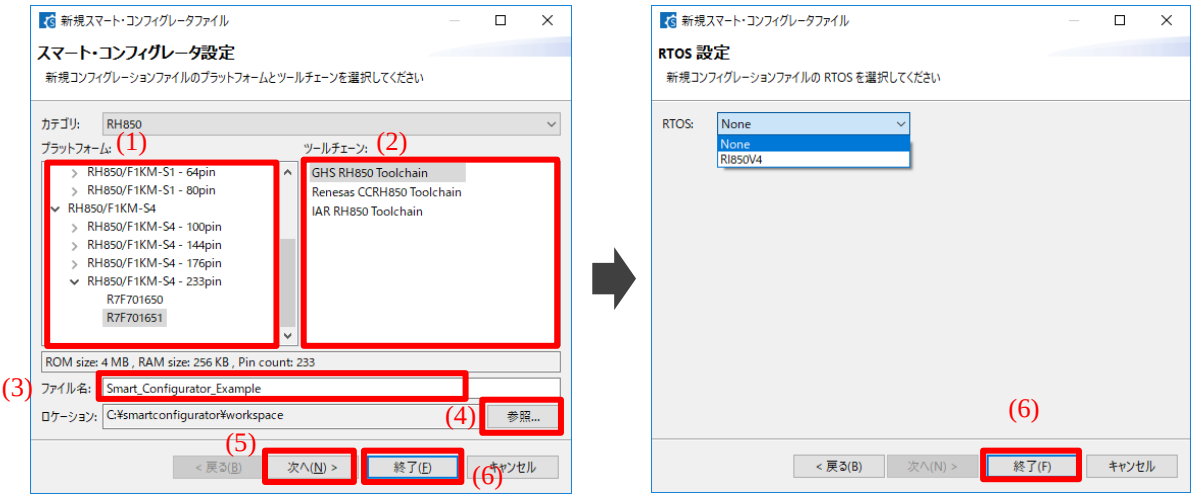


図 3-3 コンフィグレーションファイルの新規作成

(7) 任意のコンポーネントを追加し設定したあと、コードを生成し、プロジェクトを保存します。

【注】 *.eww、*.ewp、*.ewd ファイルは、初回のコード生成でのみ生成されますが、buildinfo.ipcf ファイルはコード生成のたびに生成されます。

RTOS 設定で［RIV850V4］を選択すると、コンポーネントの割り込みの非 OS 管理割り込み／OS 管理割り込みの選択が可能になります。詳細は、「4.6.1 割り込み優先レベルと OS 管理設定」を参照してください。

3.3.2　既存のコンフィグレーションファイルを開く

メインツールバーの［既存コンフィグレーションファイルを開く］ボタンをクリックすると、［ファイルを開く］ダイアログが表示されます。ファイルを選択して、［開く］ボタンをクリックしてください。

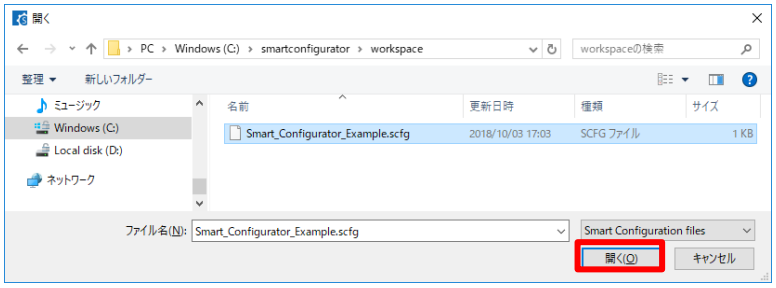


図 3-4 既存のコンフィグレーションファイルを開く

3.4　ウィンドウ

スマート・コンフィグレータを起動すると、メインウィンドウが表示されます。メインウィンドウの構成を

図 3-5 メインウィンドウに示します。

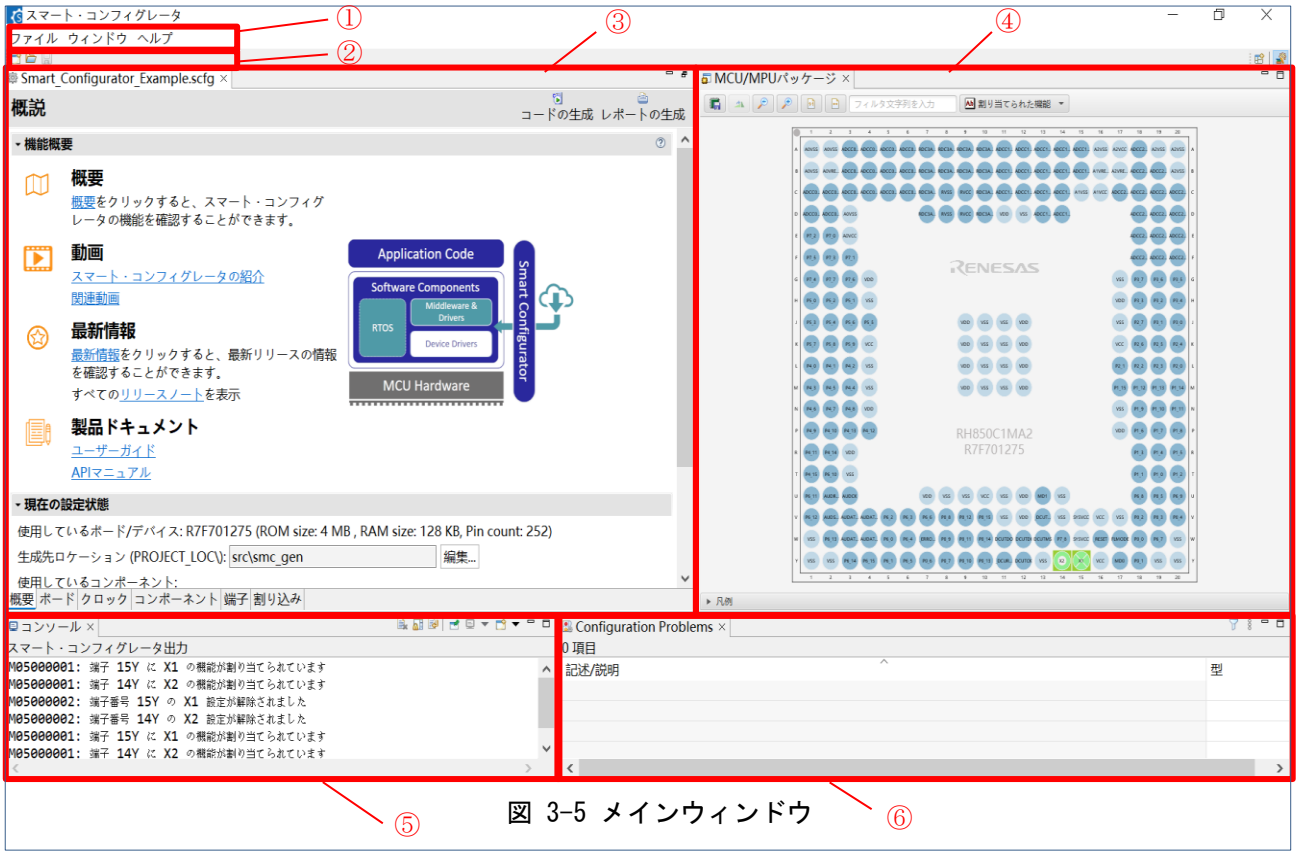


図 3-5 メインウィンドウ

① メニューバー

② メインツールバー

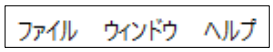
③ スマート・コンフィグレータビュー

④ MCU パッケージビュー

⑤ コンソールビュー

⑥ コンフィグレーションチェックビュー

3.4.1　メインメニュー

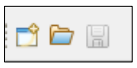


メインメニューの一覧を「**表 3-1 メニューー覧**」に示します。

表 3-1 メニューー覧

メニュー		内容
ファイル	新規	コンフィグレーションファイルを新規に作成するための［新規スマート・コンフィグレータファイル］ダイアログを表示します。
	開く	既存のコンフィグレーションファイルを開くための［開く］ダイアログを表示します。
	保存	コンフィグレーションファイルを同名で保存します。
	再開	スマート・コンフィグレータを再起動します。
	終了	スマート・コンフィグレータを終了します。
ウィンドウ	設定	コンフィグレーションファイルのプロパティを設定するための［設定］ダイアログを表示します。
	ビューの表示	ウィンドウの表示を設定するための［ビューの表示］ダイアログを表示します。
ヘルプ	ヘルプ目次	ヘルプを表示します。
	ホームページ	ルネサス WEB サイトのスマート・コンフィグレータページを表示します。
	リリースノート	リリースノートを表示します。
	ツールニュース	ツールニュースを表示します。
	API マニュアル	API マニュアルを表示します。
	説明	バージョン情報を表示します。

3.4.2　ツールバー



メインメニューの一部の機能は、ツールバーのボタンに割り当てられています。各ツールバーボタンに対応するメインメニューを「**表 3-2 ツールバーボタンとメインメニューの対応**」に示します。

表 3-2 ツールバーボタンとメインメニューの対応

ツールバーボタン	対応するメインメニュー
	［ファイル］ → ［新規］
	［ファイル］ → ［開く］
	［ファイル］ → ［保存］

3.4.3　スマート・コンフィグレータビュー

［概要］、［ボード］、［クロック］、［コンポーネント］、［端子］、［割り込み］の 6 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。

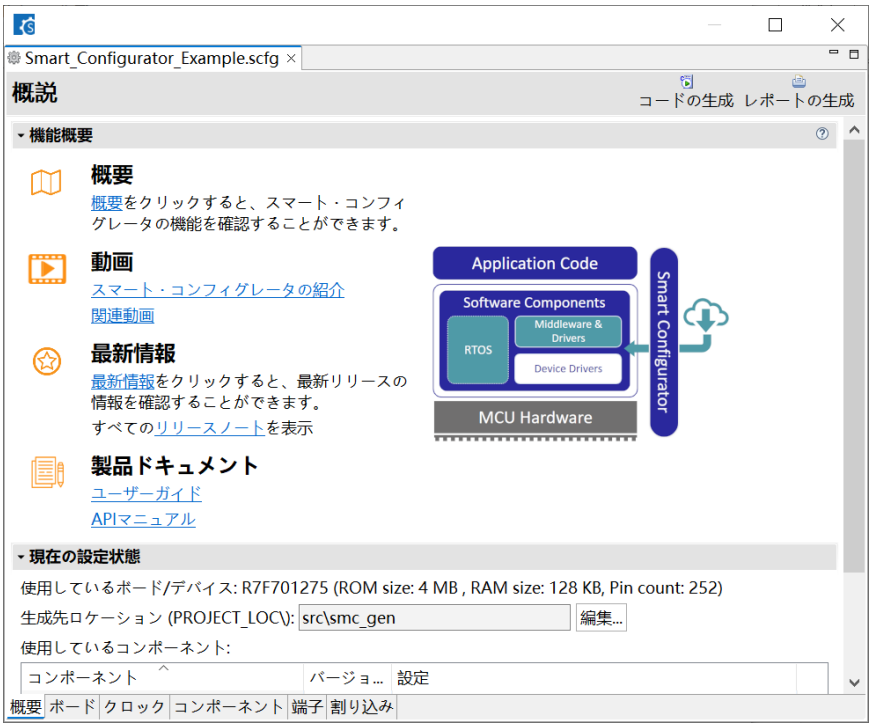


図 3-6 スマート・コンフィグレータビュー

3.4.4　MCU パッケージビュー

MCU パッケージを表示します。表示の回転や拡大、縮小、MCU パッケージビューをイメージファイルに保存できます。また、端子割り当ての状況および変更ができます。

[割り当てられた機能]と[ボード機能]、[シンボリック名]の 3 種類のパッケージビューを切り替えることができます。

・ [割り当てられた機能] には、端子設定の割り当て状況が表示されます。

・ [ボード機能]にはボードの初期端子設定情報が表示されます。

・ [シンボリック名]には、ユーザーが端子に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、ポートの read/write 関数とともに Pin.h ファイルに生成されます。

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となります(「4.1.2 ボード選択」および「4.5.6 章 ボード端子による端子設定」を参照) 構成情報」)。

注: シンボリック名は、RH850/F1KM, F1KH には適用されません。

シンボリック名は、APORT, JPORT, IPORT には適用されません。

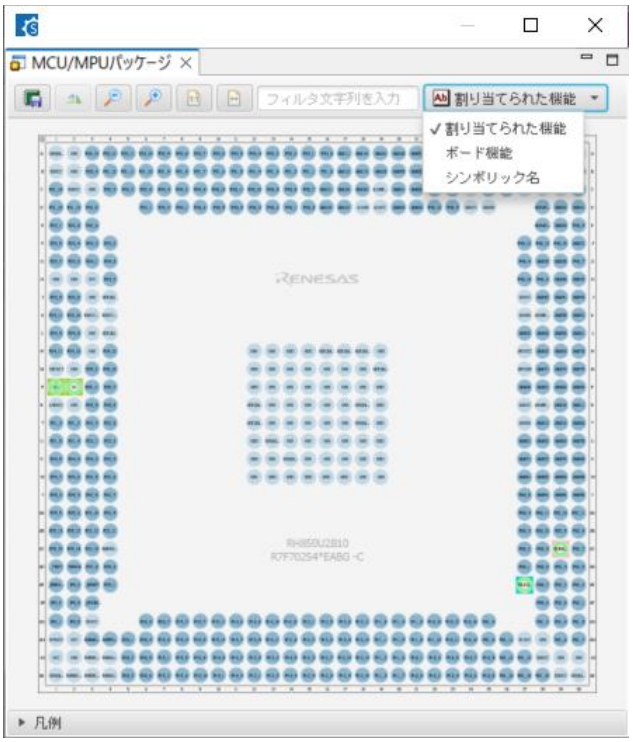


図 3-7 MCU パッケージビュー

3.4.5　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 パッケージビューでの設定変更内容が表示されます。

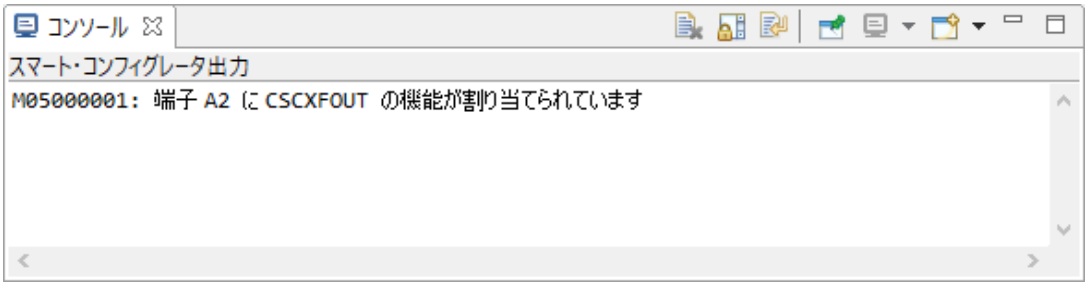


図 3-8 コンソールビュー

3.4.6　コンフィグレーションチェックビュー

周辺機能、割り込みおよび端子競合が発生した場合に、その内容を表示します。

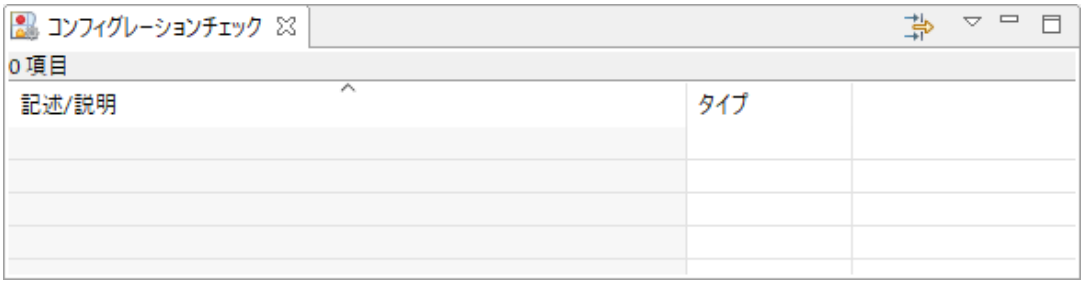


図 3-9 コンフィグレーションチェックビュー

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、ボードの選択および、デバイスの変更が可能です。

【注】デバイスの変更は、IAR および GHS のプロジェクトには反映されません。

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。

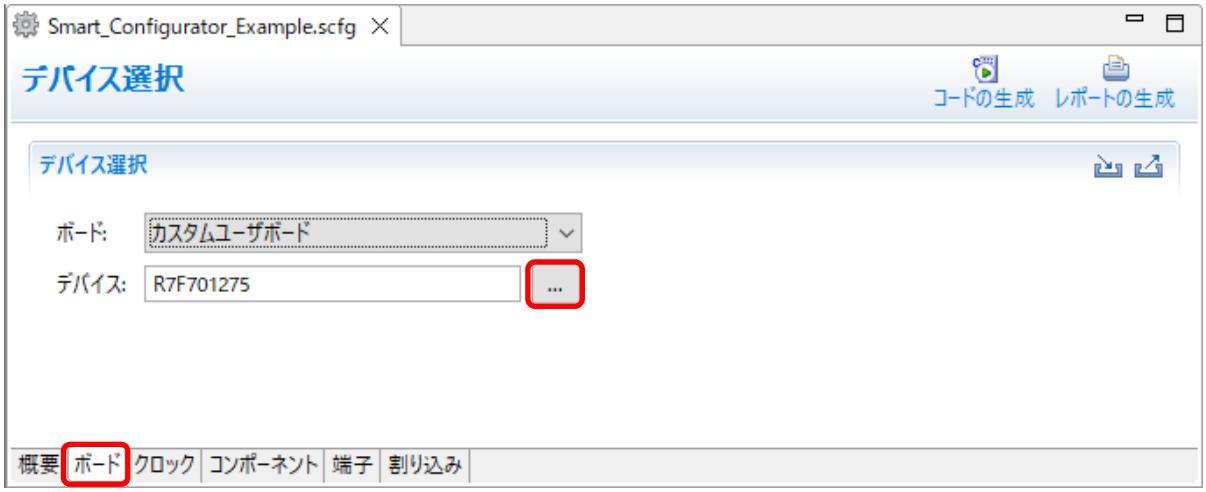


図 4-1 デバイス選択

デバイスを変更すると下記のメッセージが表示されます。各ボタン操作については、「**表 4-1 デバイス変更確認操作一覧**」を参照ください。

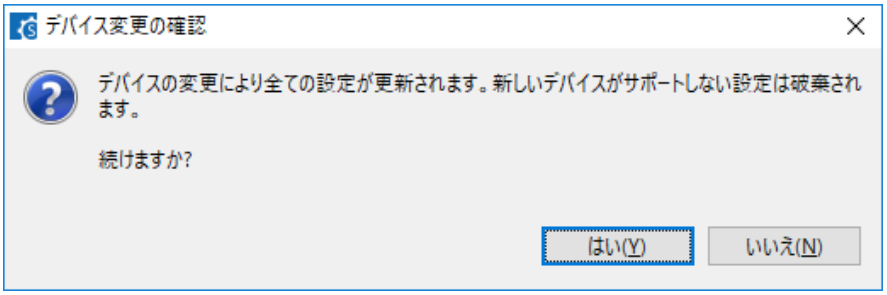


図 4-2 デバイス変更の確認

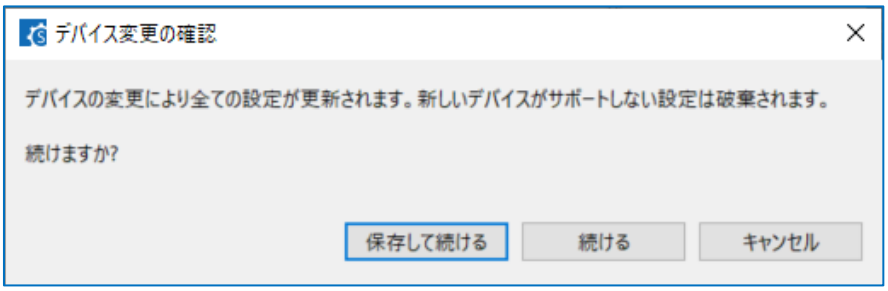


図 4-3 デバイス変更の確認(設定の変更をセーブをしていない場合)

表 4-1 デバイス変更確認操作一覧

ボタン	内容
はい	選択したデバイスに変更します。
いいえ	デバイスを変更しません。
保存して続ける	現在の設定内容をコンフィグレーションファイルに保存した後、選択したデバイスに変更します。
続ける	現在の設定内容をコンフィグレーションファイルに保存せず、選択したデバイスに変更します。
キャンセル	デバイスを変更しません。

4.1.2 ボード選択

［ ］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、周辺機能のコンフィグレーションを行うとボードの接続に合わせて端子が自動で設定されます。

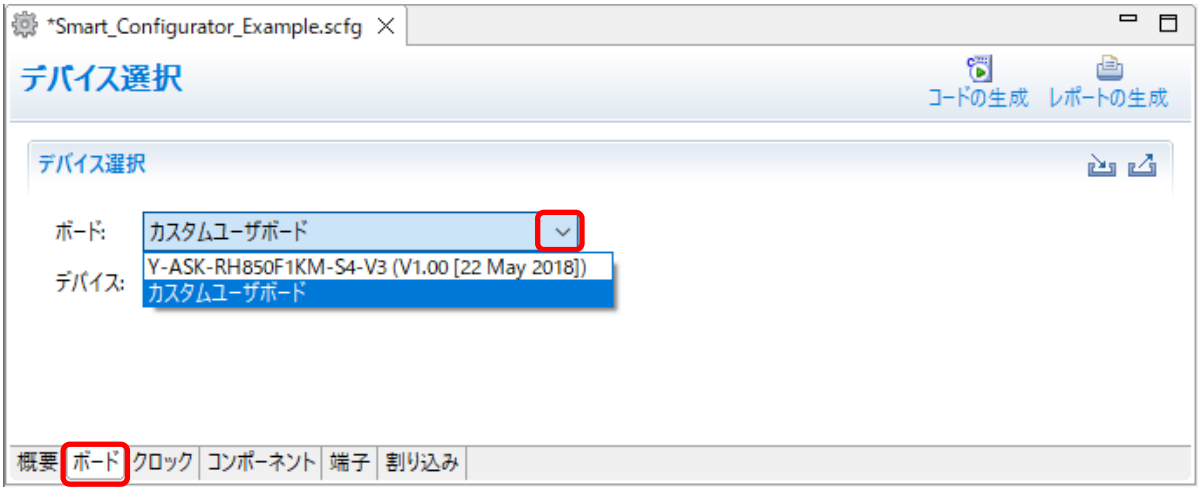


図 4-4 ボードの選択

選択したボードの設定に合わせて、以下の項目が変更されます。

- 端子割り当て
- メインクロック周波数
- サブクロック周波数
- デバイスの変更

ボードを変更すると**図 4-2** または下記のメッセージが表示されます。各ボタン操作については、「**表 4-2 ボード変更確認操作一覧**」を参照ください。

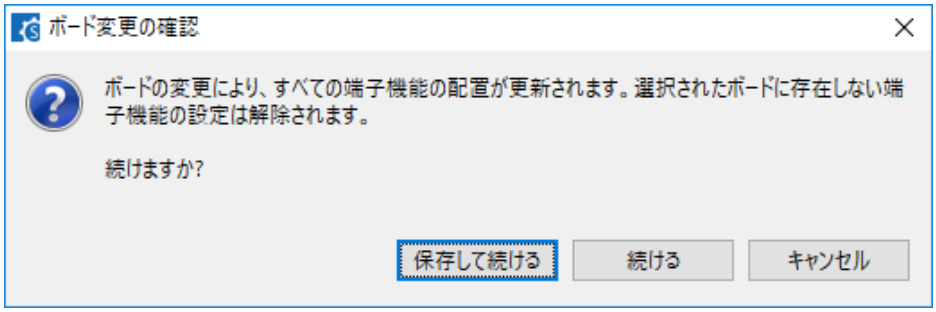


図 4-5 ボード変更の確認

表 4-2 ボード変更確認操作一覧

ボタン	内容
保存して続ける	現在の設定内容をコンフィグレーションファイルに保存した後、選択したデバイスに変更します。
続ける	現在の設定内容をコンフィグレーションファイルに保存せず、選択したデバイスに変更します。
キャンセル	ボードを変更しません。

【注】ボードの変更によりデバイスも変更されます。
　　デバイスの変更は、IAR および GHS のプロジェクトには反映されません。

4.1.3　ボード設定のエクスポート

現在のメインクロック周波数、サブクロック周波数および端子割り当ての設定を、ボード設定としてエクスポートすることができます。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を入力します。

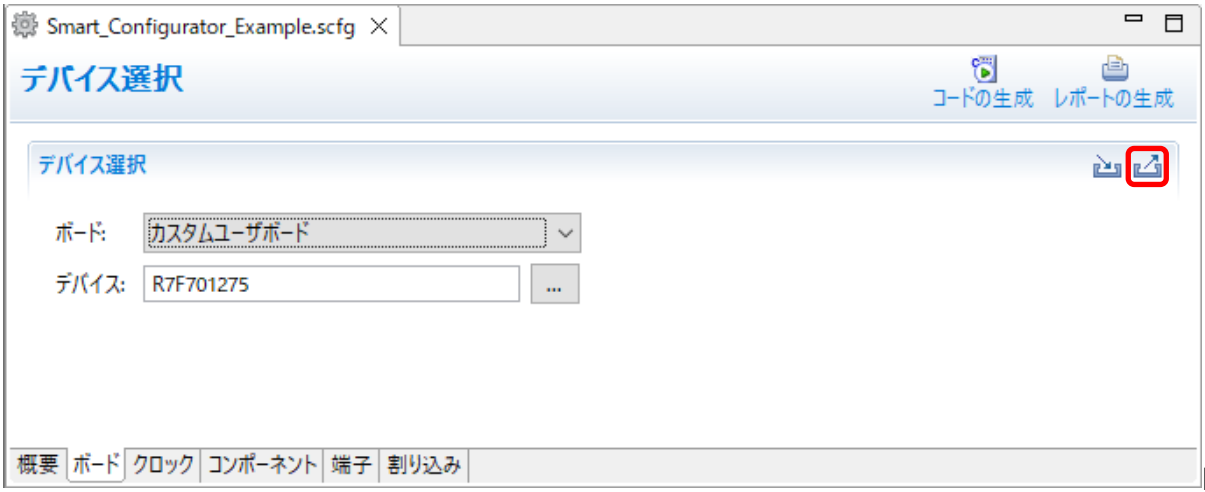


図 4-6 ボード設定のエクスポート(bdf 形式)

4.1.4　ボード設定のインポート

ボード設定は bdf (Board Description File) に定義されています。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 を選択してください。

(2) ボード選択の選択肢にインポートしたボード設定が追加されます。

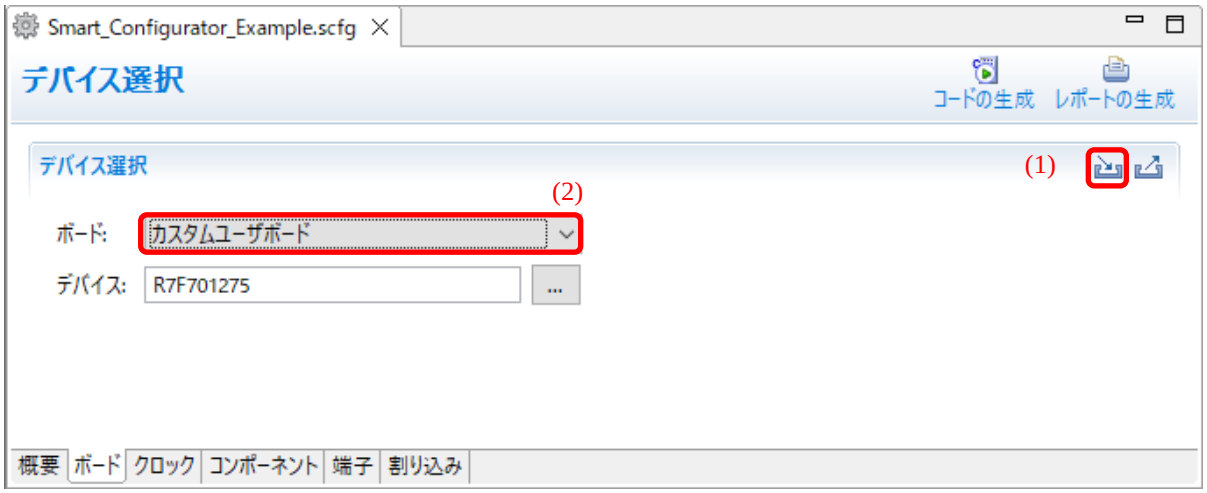


図 4-7 ボード設定のインポート(bdf 形式)

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、クロックを設定します。クロックページの設定は、各コンポーネントのクロックソースとして使用されます。コンポーネントの設定前にクロックを設定してください。

クロック設定は、以下の手順で行います。

(1) クロック発振回路を設定します。

(2) CPU、各周辺機能への供給クロックソースを設定します。

(a) 画面上でマウスを移動すると、クロック信号を青色で表示します。

(b) クロックセレクタは画面をクリックして選択します。

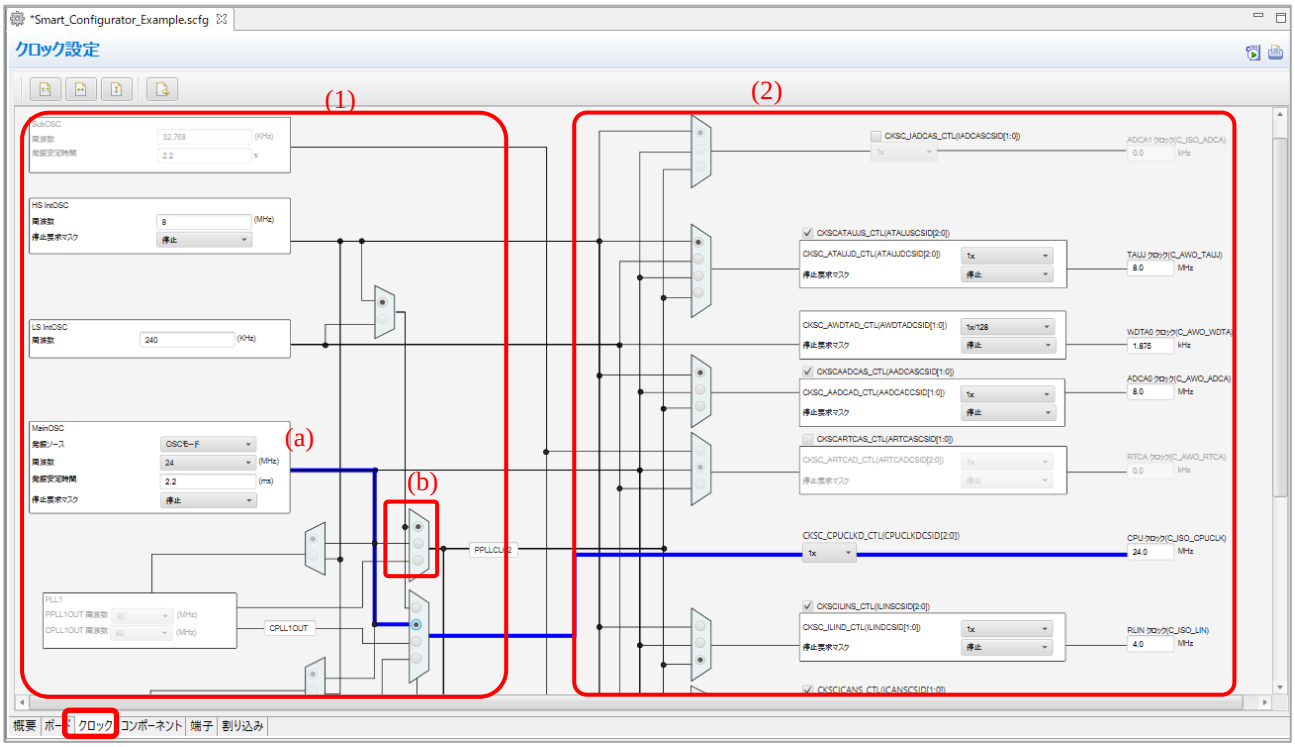


図 4-8 クロック設定

4.3 システム設定(RH850/U2A のみ)

[システム]ページで使用する 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。

RH850/U2A のみがシステム設定をサポートしています。

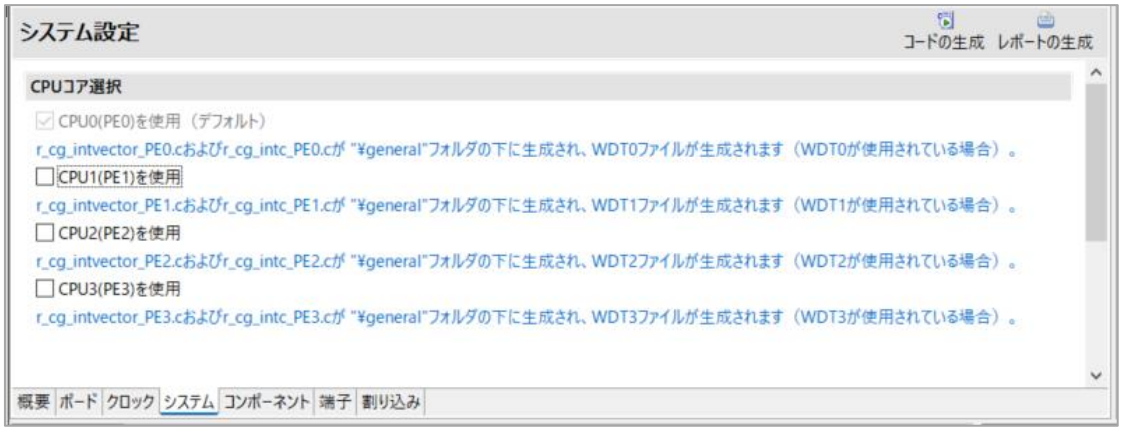


図 4-9 [System]ページ

4.4　コンポーネント設定

コンポーネントページで、周辺機能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。

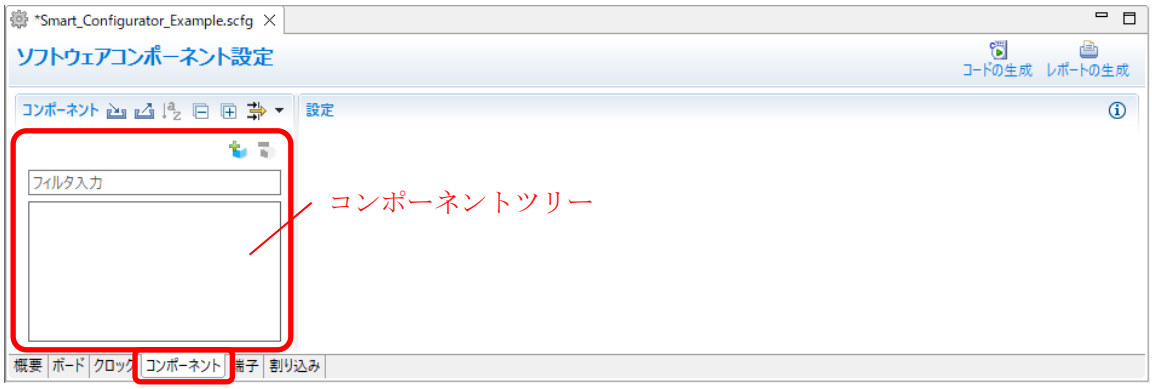


図 4-10 コンポーネントページ

4.4.1　コンポーネント一覧とハードウェア一覧との切り替え

コンポーネントツリーは、コンポーネント一覧とハードウェア一覧の表示切り替えができます。ハードウェア一覧表示は、コンポーネントツリーのノードを直接クリックすることで、新しいコンポーネントを追加できます。ハードウェア一覧表示からコンポーネントを追加するには、以下の手順で行います。

(1) ［表示メニュー］ アイコンをクリックし、［ハードウェア一覧表示］を選択します。コンポーネントツリーに、ハードウェアリソースが表示されます。

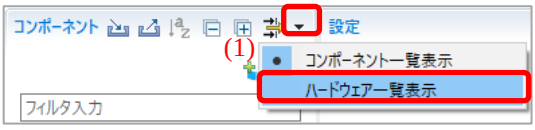


図 4-11 ハードウェア一覧への切り替え

(2) ハードウェアリソースのノードをダブルクリックし、［コンポーネントの追加］ダイアログボックスを開きます。 (例：タイマ・アレイ・ユニット B の TAUB0 をダブルクリック)

(3) ［ソフトウェアコンポーネントの選択］ページからコンポーネントを選択します。(例：PWM 出力)

(4) ［次へ］をクリックします。

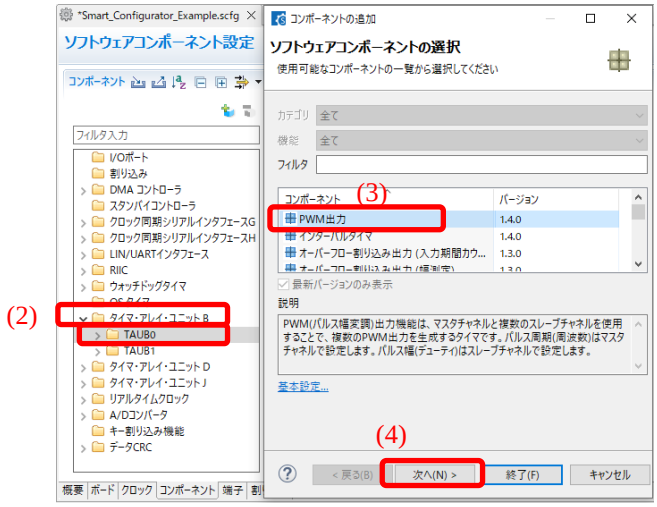


図 4-12 ハードウェア一覧からのコンポーネントの追加

(5) ［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、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_TAUB1）

(6) リソースを選択、またはデフォルトのリソースを使用します。（例：TAUB1）

(7) ［終了］をクリックします。コンポーネントがコンポーネントツリーに追加されます。

図 4-13 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（例：Config_TAUB0）

4.4.2 コンポーネントの追加

(1) ［コンポーネントの追加］ アイコンをクリックします。

図 4-14 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］ページで、リストからコンポーネントを選択します。（例：PWM 出力）

(3) ［次へ］をクリックします。

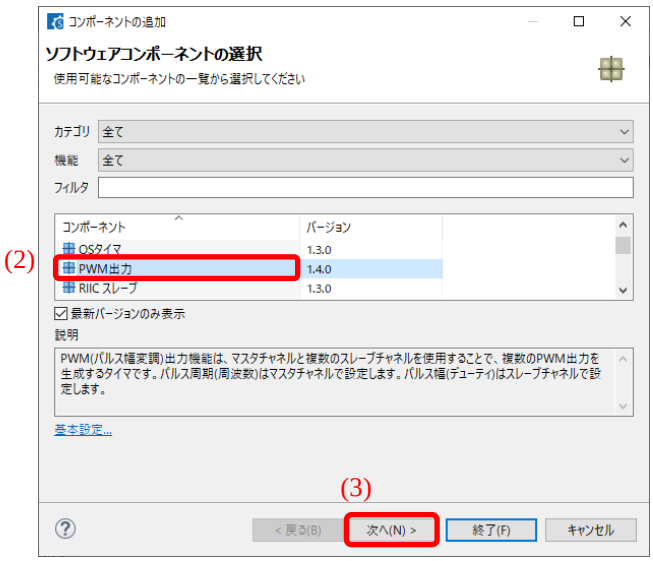


図 4-15 ソフトウェアコンポーネントの選択

(4) ［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、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_TAUB0）

(5) リソースを選択、またはデフォルトのリソースを使用します。（例：TAUB0）

(6) ［終了］をクリックします。コンポーネントがコンポーネントツリーに追加されます。

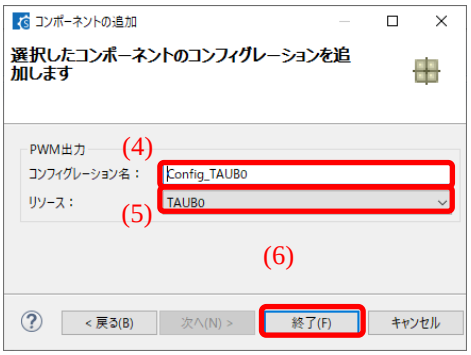


図 4-16 選択したコンポーネントのコンフィグレーションを追加します（例：Config_TAUB0）

4.4.3 コンポーネントの削除

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コンポーネントを選択します。 (Ctrl キーを押しながら複数選択可)

(2) ［コンポーネントの削除］ アイコンをクリックします。

コンポーネントツリーから、選択したコンポーネントが削除されます。

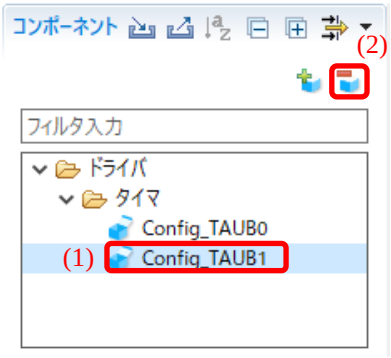


図 4-17 コンポーネントの削除

4.4.4 コンポーネントのコンフィグレーション設定

コンポーネントのコンフィグレーション設定は、以下の手順で行います。

(1) コンポーネントツリーにある コンポーネントをクリックします。（例：Config_TAUB0）

(2) 右側の設定パネルでコンフィグレーションを設定します。図 4-18 は例です。

a. ［クロックソース］の項目で［PCLK/2］を選択します。

b. チャネル 1 スレーブ、チャネル 2 スレーブ、チャネル 3 スレーブを選択します。

c. ［マスター0］タブ上の［パルス周期］を設定します。

d. ［スレーブ 1］、［スレーブ 2］、［スレーブ 3］タブ上それぞれの［デューティ］を設定します。

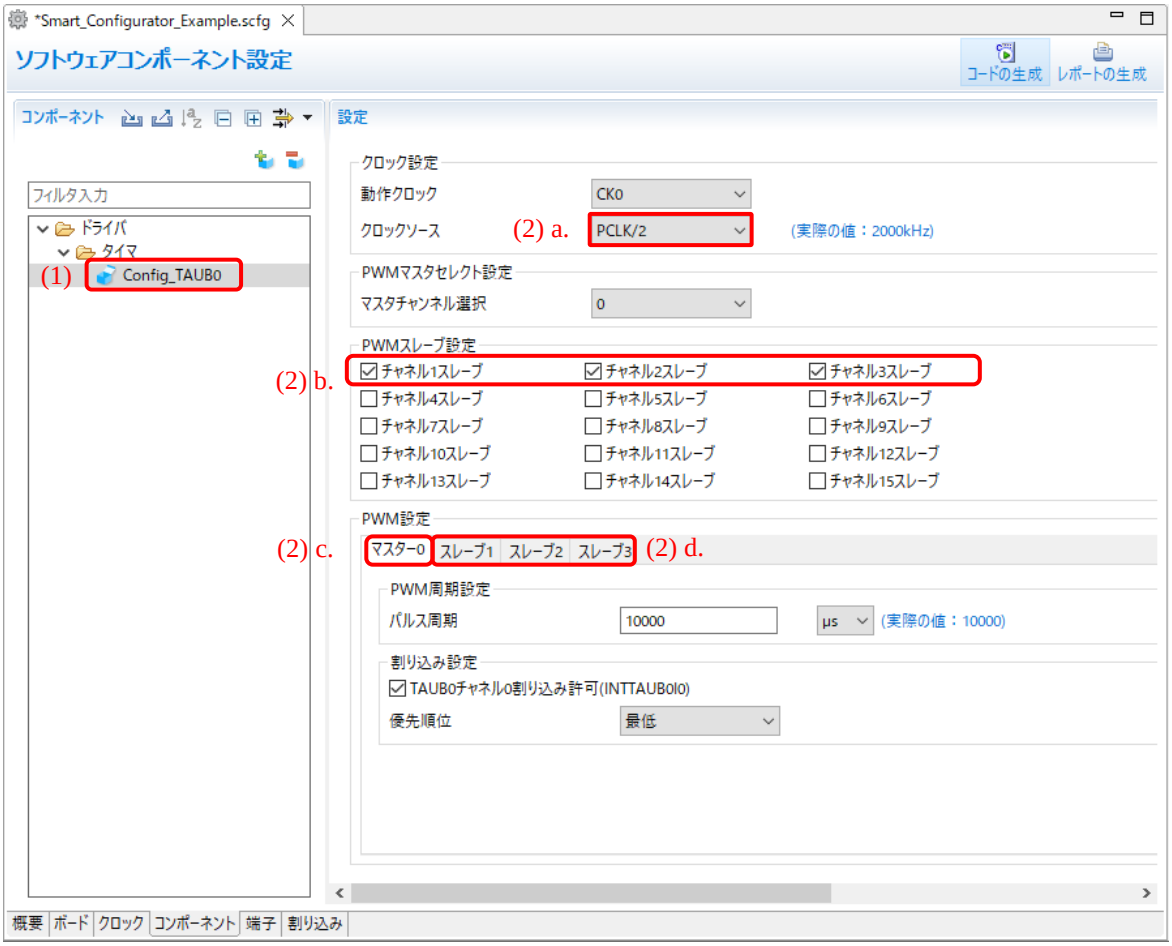


図 4-18 コンポーネントのコンフィグレーション設定

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。

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　コンポーネントのリソース変更

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：TAUB0 から TAUB1 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。リソースを変更するには、以下の手順で行います。

(1)　コンポーネントを右クリックします。（例：Config_TAUB0）

(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。

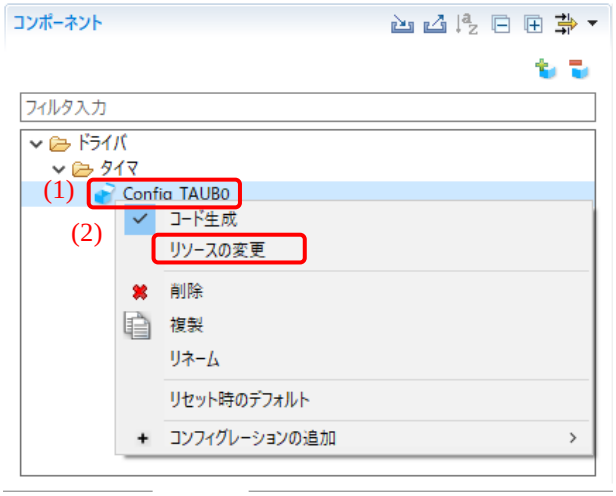


図 4-19 リソースの変更

(3) ［リソースの選択］ダイアログボックスでリソースを選択します。（例：TAUB1）

(4) ［次へ］ボタンをクリックします。

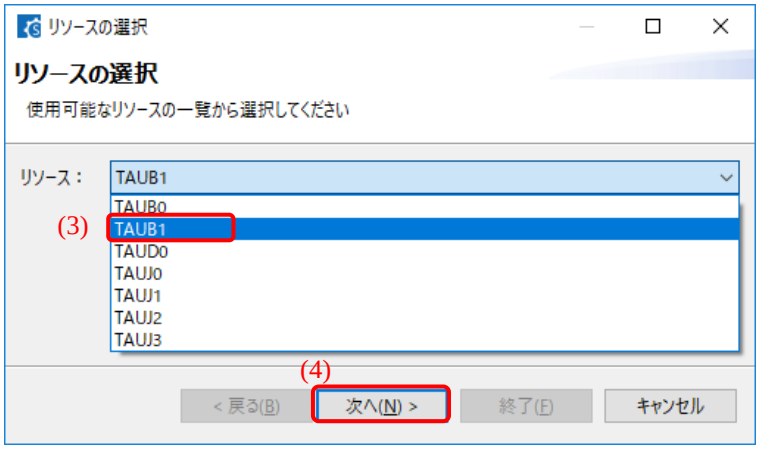


図 4-20 新しいリソースの選択

(5) ［リソースの選択］ダイアログの［コンフィグレーション設定の選択］ページに、設定項目の内容が表示されます。

(6) 設定が変更可能であるかを確認します。

(7) 表示されている設定を使用するか、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。

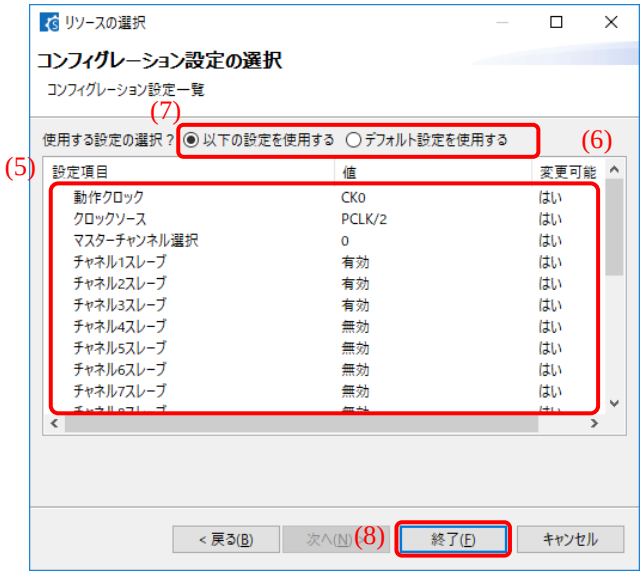


図 4-21 新しいリソース設定の確認

リソースは、自動的に更新されます。（例：INTTAUB0I0 から INTTAUB1I0）

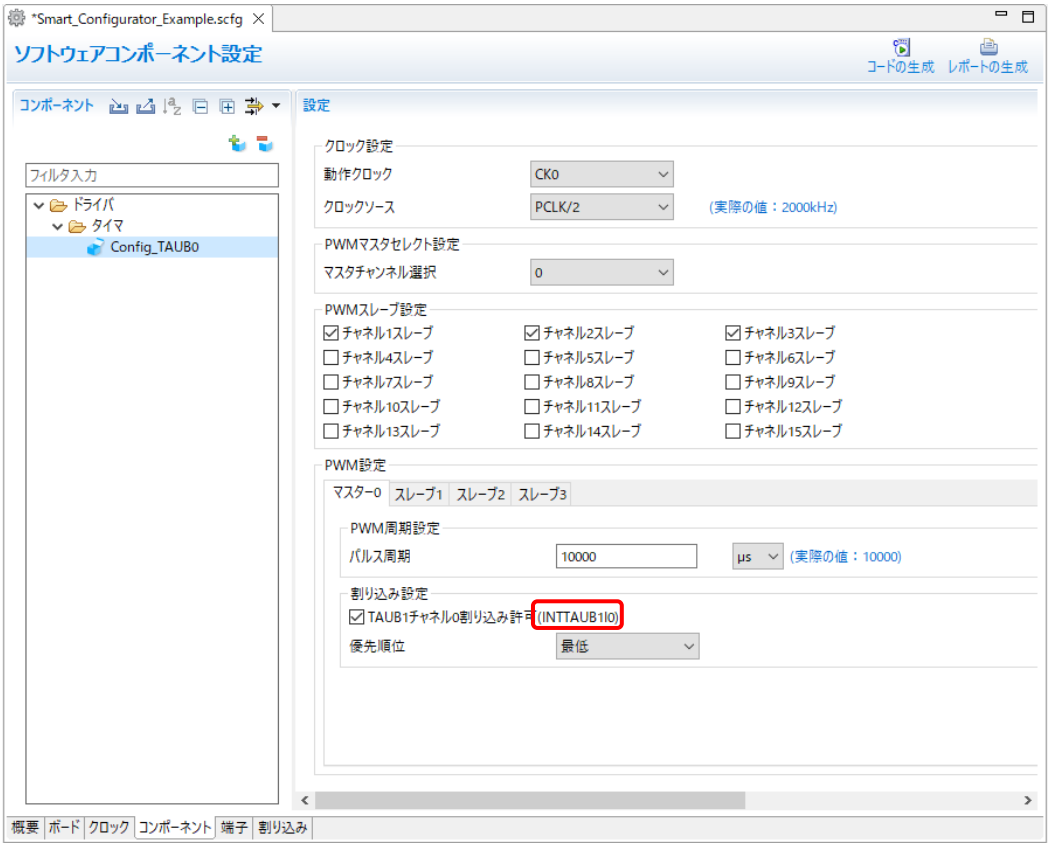


図 4-22　自動的に更新されるリソース

コンフィグレーション名の変更が必要な場合、(9)、(10)を行ってください。

(9)　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、コンフィグレーション名を変更します。（例：Config_TAUB0 を Config_TAUB1）。

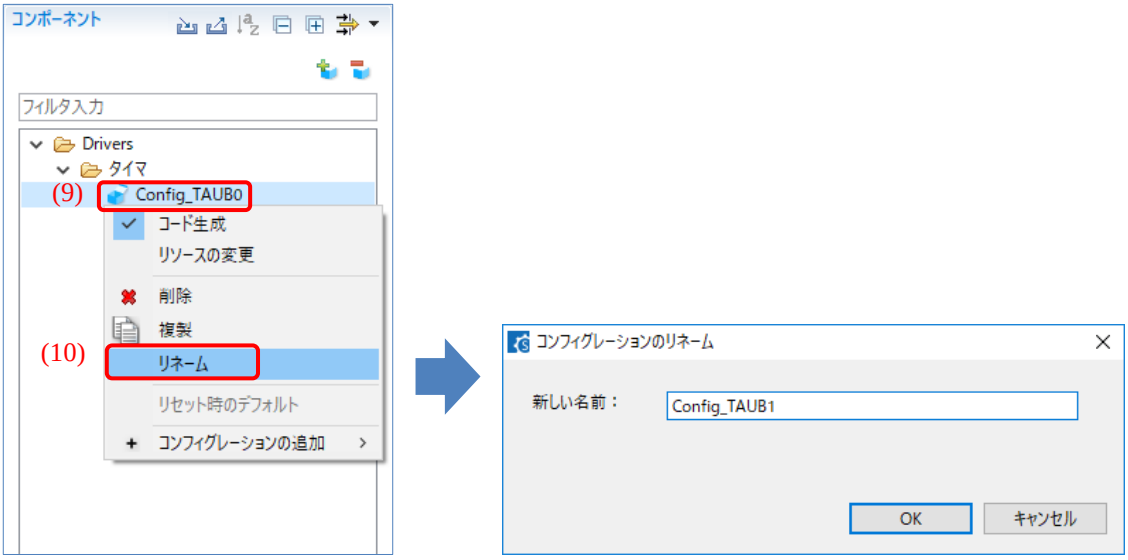


図 4-23　コンフィグレーション名の変更

4.4.6　コンポーネント構成のエクスポート

[コンポーネント] ページの [(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。

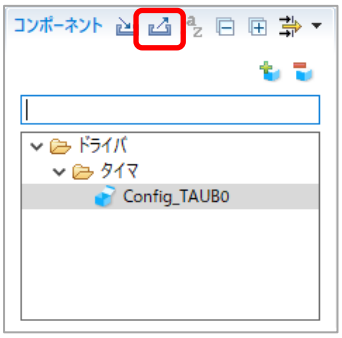


図 4-24　コンフィグレーションのエクスポート

4.4.7　コンポーネント構成のインポート

[(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、エクスポートされた*.xml ファイルを選択すると、コンポーネントの構成がインポートできます。

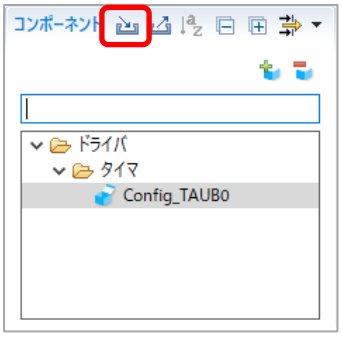


図 4-25　コンフィグレーションのインポート

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 4-12）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。

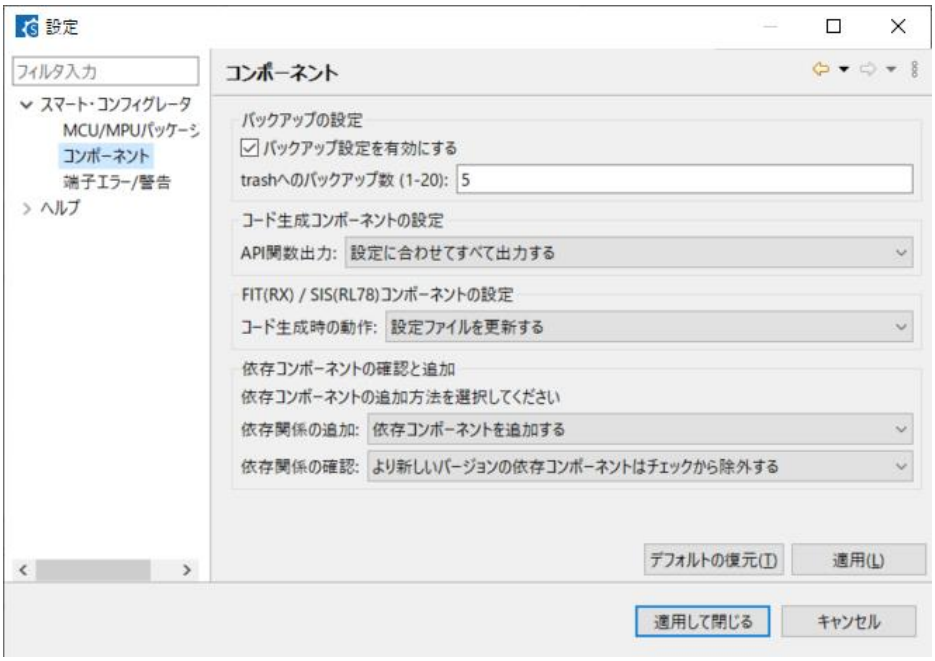


図 4-26 コンポーネントの基本設定

【注】

1. [バックアップ設定を有効にする] を選択し、下図の [trash へのバックアップ数 (1-20)]を設定することで、trash フォルダに作成されるバックアップのフォルダの数を制限できます。

   設定値を超えると、最も古いフォルダが置き換わります。

   バックアップ数を 0 に設定すると、バックアップ機能が無効になります。

図 4-27　バックアップ数の設定

2. 初期化関数のみを生成したい場合は、下図の [A'I 関数出力]を[初期化関数のみ出力する]に変更します。 *.h *、*c * 内の void R_ConfigXXX_Create (void)、void R_ConfigXXX_Create_UserInit (void) のみが生成されるようにします。(XXX はコンフィグレーション名)

   設定を「設定に合わせてすべて出力する」に戻すと、すべての API 関数が再生成されます。

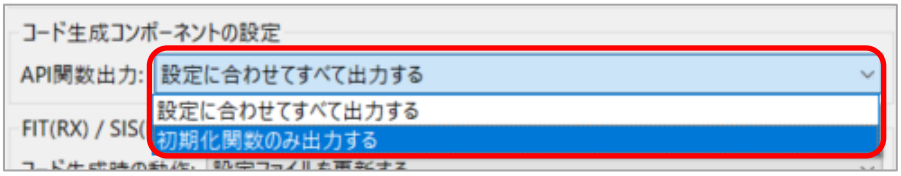


図 4-28 API 関数出力設定

4.5　端子設定

端子ページで、端子機能の割り当てをします。 ［端子機能］リストでは、リソース別に端子機能を表示します。 ［端子番号］リストでは、端子番号順に全ての端子を表示します。

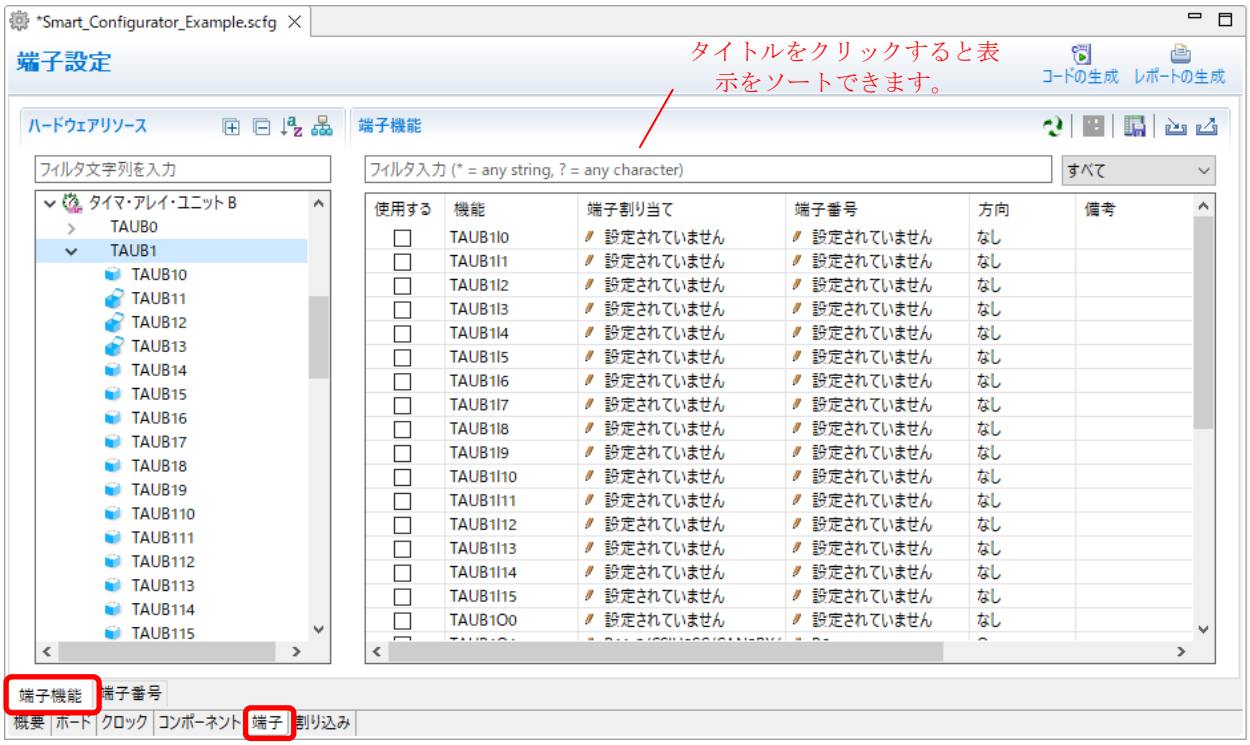


図 4-29 端子ページ（端子機能）

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

また、[機能]の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。

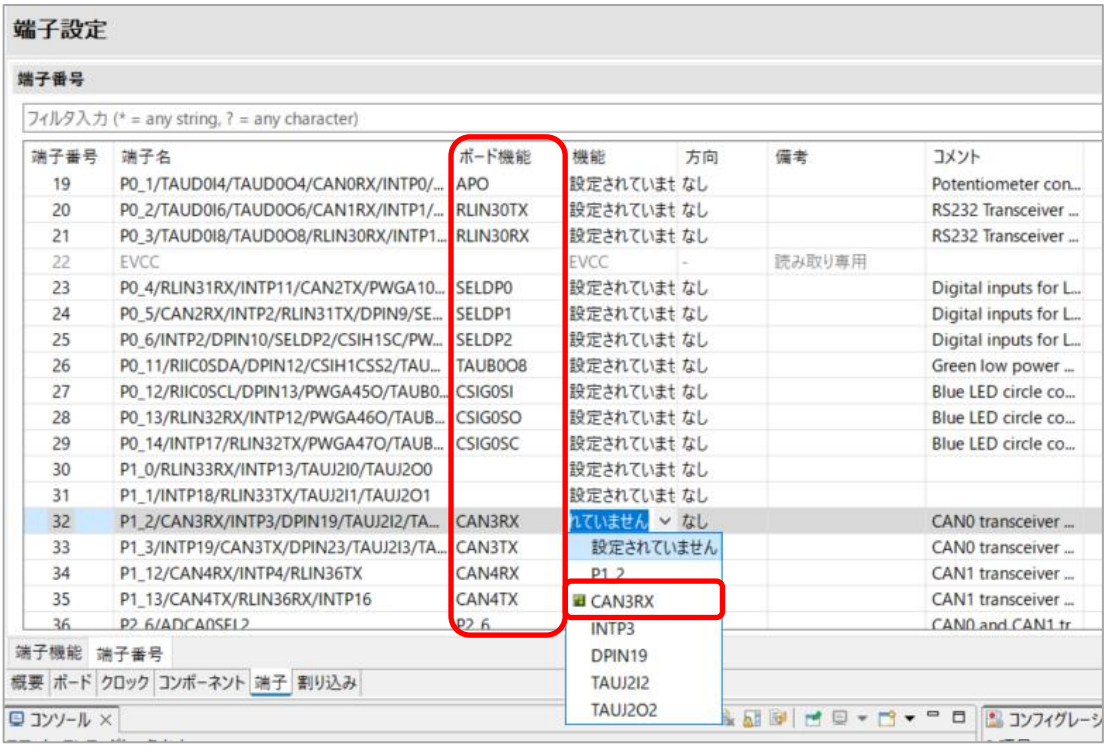


図 4-30 端子ページ（端子番号）

4.5.1　リソースの端子割り当て

端子ページで、コンポーネントのリソースで使用する端子割り当てを行います。

端子割り当ては、［端子機能］リストと［端子番号］リストのどちらでも行えます。

［端子機能］リストで端子割り当てを行う手順を説明します。端子割り当ては、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、ハードウェアリソース表示に切り替えます。

(2) ハードウェアリソースを選択します。（例：割り込み）

(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) ［端子割り当て］、［端子番号］欄、または［選択されたリソースの次の端子割り当て先］ ボタンで端子割り当てを行います。

(a) ［端子割り当て］または［端子番号］をクリックし、リストから端子を割り当てます。（例：［端子割り当て］で P10_0 → P0_1 ）

(b) ［選択されたリソースの次の端子割り当て先］ ボタンをクリックし、端子配置を変更します。クリックするごとに、機能を持つ端子が表示されます。

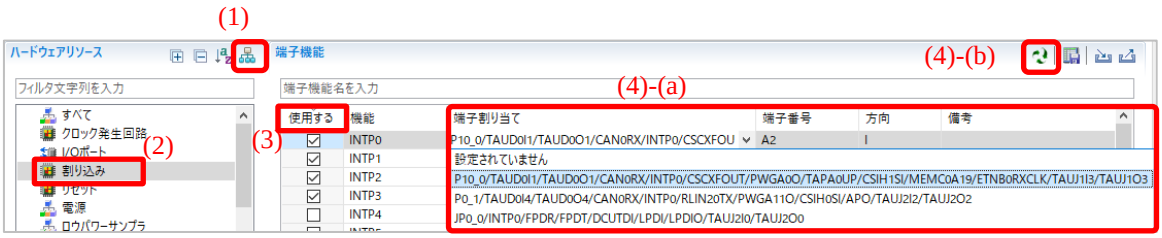


図 4-31　［端子機能］リストでの端子割り当て

コンポーネントが設定されている場合、［使用する］欄のチェックボックスにチェックが入っています。コンポーネントが設定されていない状態でも端子の割り当てが可能です。コンポーネントが設定されていない状態で端子割り当てを行うと、備考欄に“この端子はコンポーネントにより使用されていません”と表示します。

4.5.2　MCU パッケージでの端子割り当て

MCU パッケージビューで端子割り当てを設定するには、以下の手順で行います。

(1) ［拡大］ ボタンをクリックするか、マウスホイールをスクロールして、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可能です。

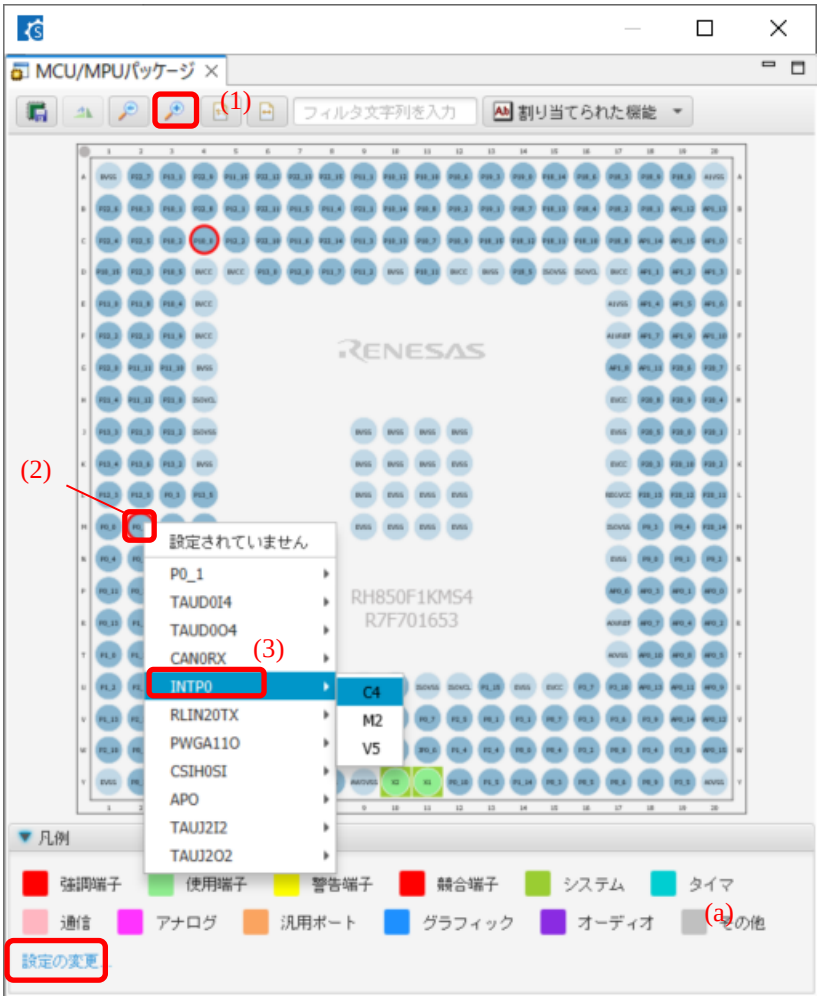


図 4-32 MCU パッケージでの端子割り当て

4.5.3　端子機能から端子番号を表示する

端子機能に関連付けられた端子番号にジャンプすることができます。

以下の手順で行います。

(1)　[端子機能] タブで端子機能を右クリックし、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。

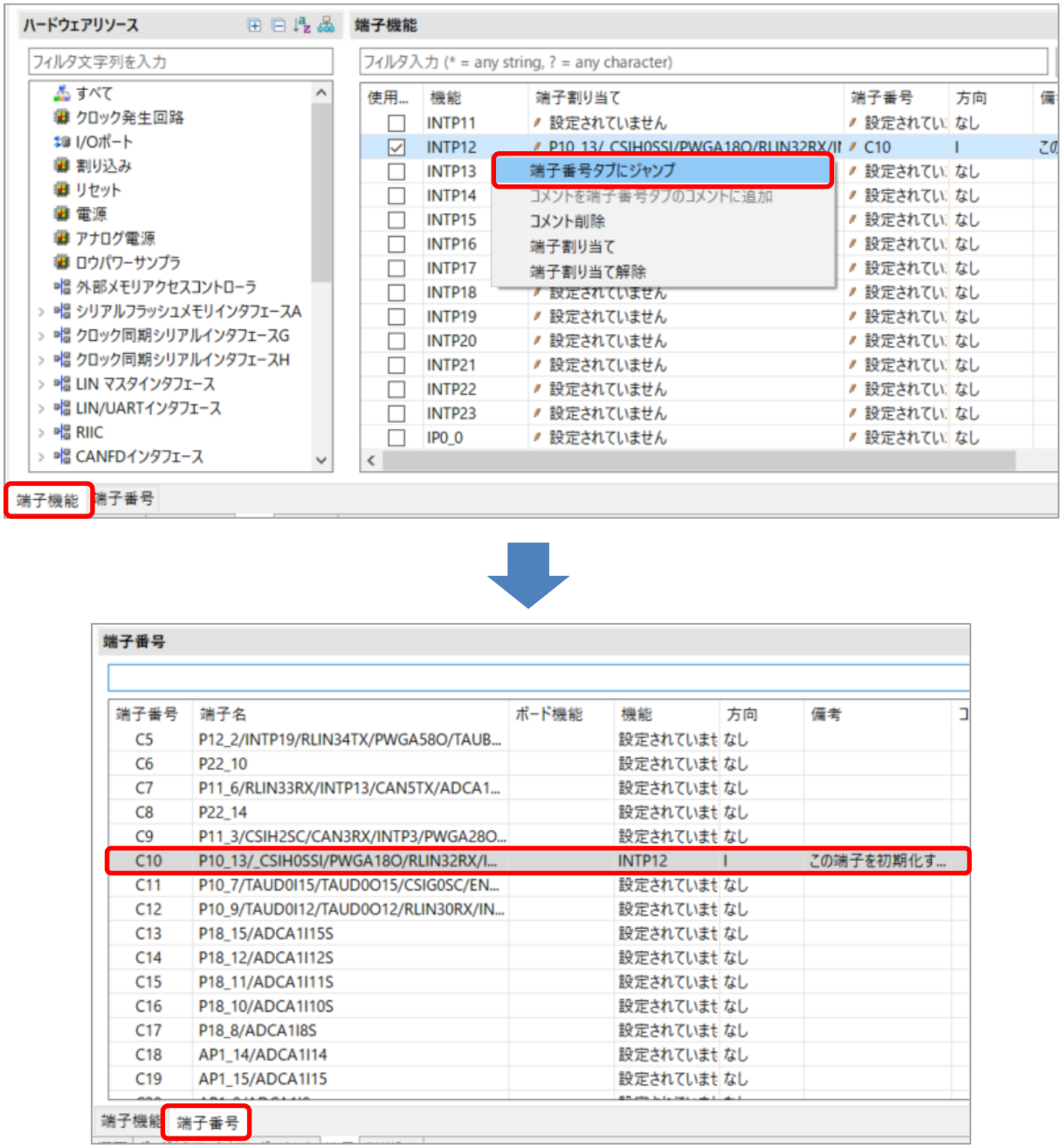


図 4-33　端子番号タブにジャンプ

4.5.4　端子設定のエクスポート

端子割り当ての設定を XML 形式でエクスポートできます。エクスポートしたファイルは、同じデバイスファミリのプロジェクトにインポート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)　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。

(2)　［エクスポート］ダイアログで、エクスポートするファイル名を入力します。

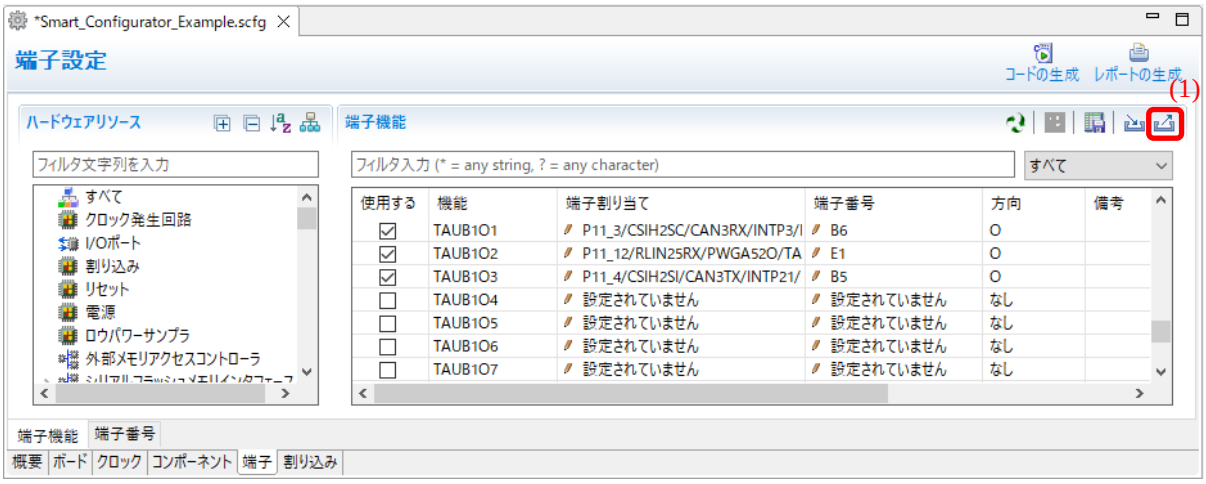


図 4-34 端子設定のエクスポート（XML 形式）

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と、端子割り当ての設定情報を CSV 形式でファイルに保存します。

4.5.5　端子設定のインポート

端子割り当て設定を含む XML 形式のファイルをインポートすることができます。ファイルをインポートすると、端子の割り当てが反映されます。端子設定のインポートは、以下の手順で行います。

(1)　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。

(2)　［インポート］ダイアログで、インポートするファイル名を選択します。

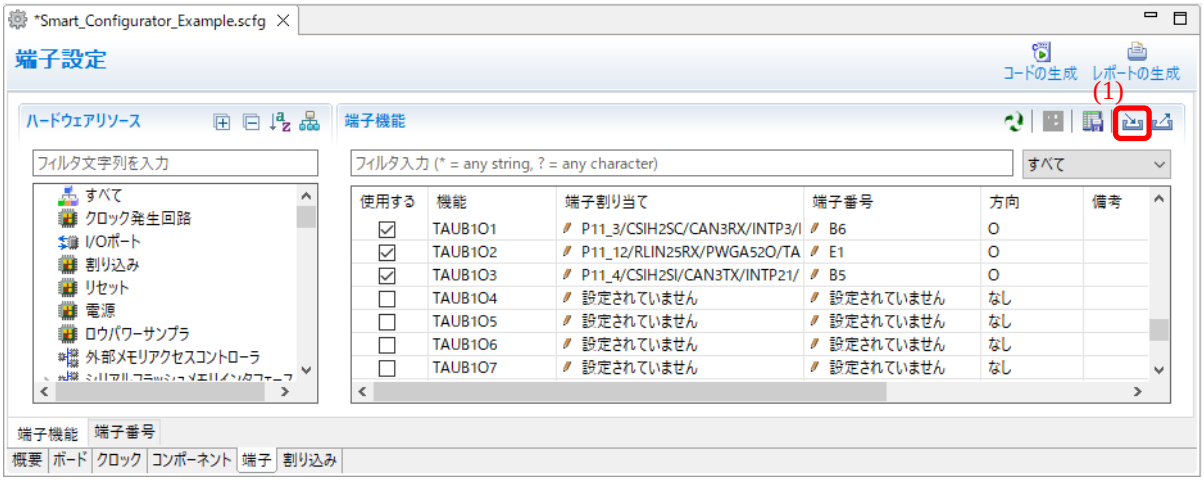


図 4-35 端子設定のインポート（XML 形式）

4.5.6　ボードの端子情報を使用した端子設定

使用するルネサス製ボードに対応した初期端子構成を設定できます。
選択したボードは、[ボード]ページで確認できます。
ボード設定のインポートについては 4.1.4 を参照してください。
以下に、ボードの端子情報を使用して端子設定をする手順を説明します。

(1)　[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。
(2)　[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。
(3)　[OK]をクリックします。

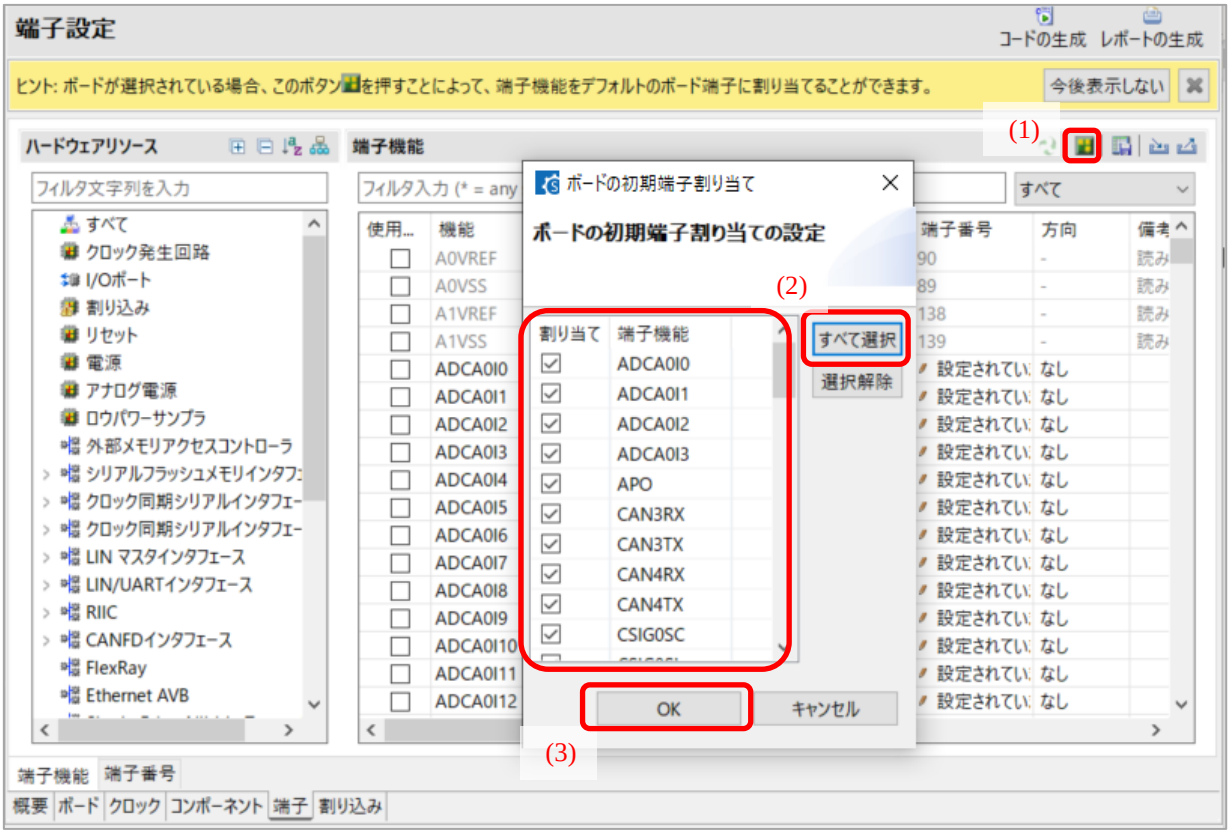


図 4-36　端子の初期設定

4.5.7　端子フィルタ機能

[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。

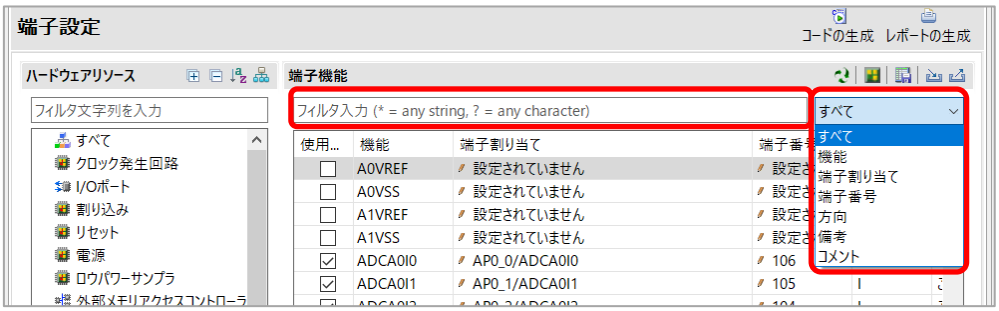


図 4-37　端子表示のフィルタリング

4.5.8　端子エラー/警告

[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。

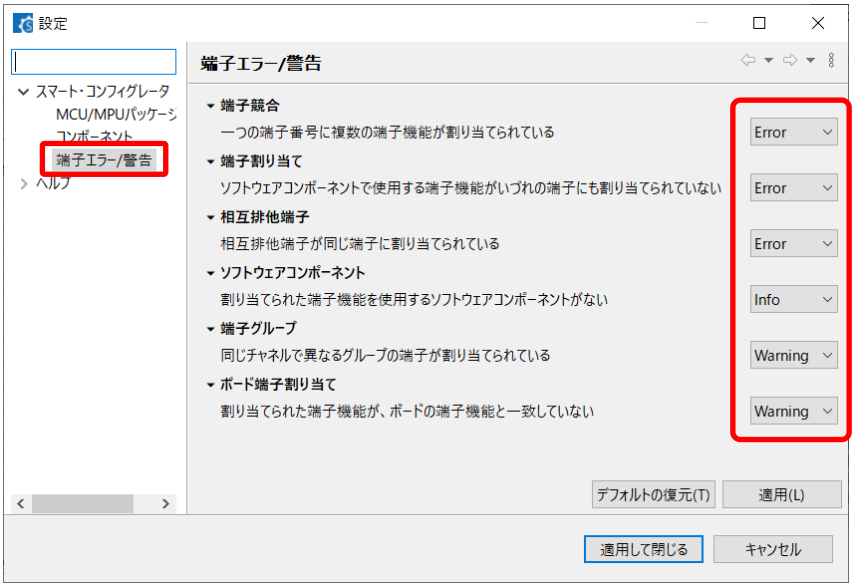


図 4-38 端子エラー/警告の設定

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 **4-39** のようになります。

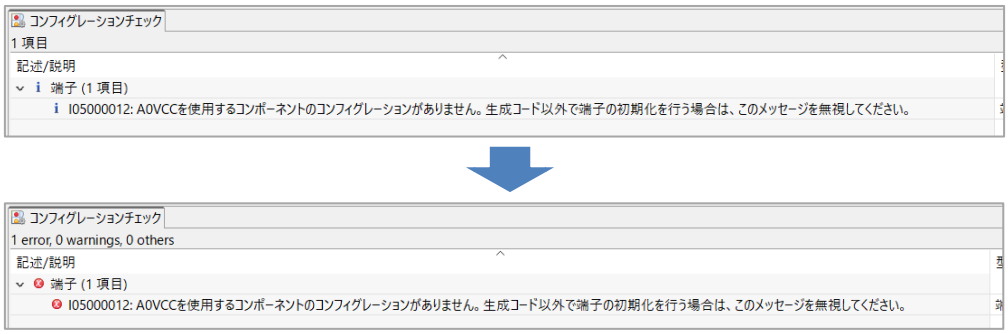


図 4-39 端子エラー/警告の設定の変更例

4.6　割り込み設定

［割り込み］ページでは、［コンポーネント］ページで設定したコンポーネントの割り込みの確認および優先レベルの変更が行えます。

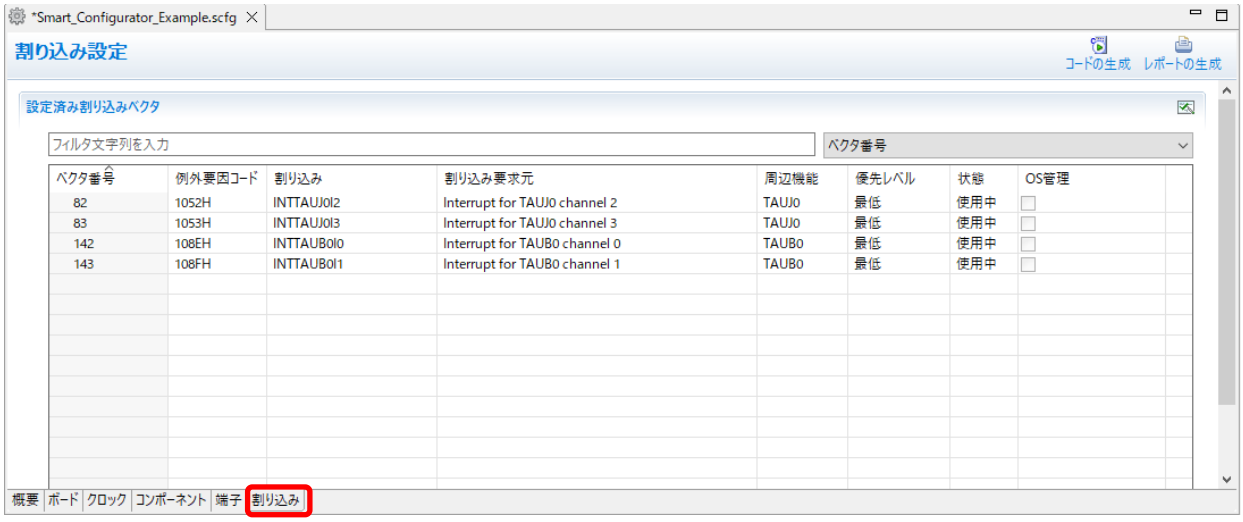


図 4-40 割り込みページ

4.6.1　割り込み優先レベルと OS 管理設定の変更

コンポーネントのコンフィグレーション設定で割り込みを使用している場合、［状態］欄に“使用中”と表示されています。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) ［優先レベル］で、割り込み優先レベルを変更できます。優先レベルの設定は、コンポーネントのコンフィグレーション設定に反映されます。

(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(*1)で出力されます。

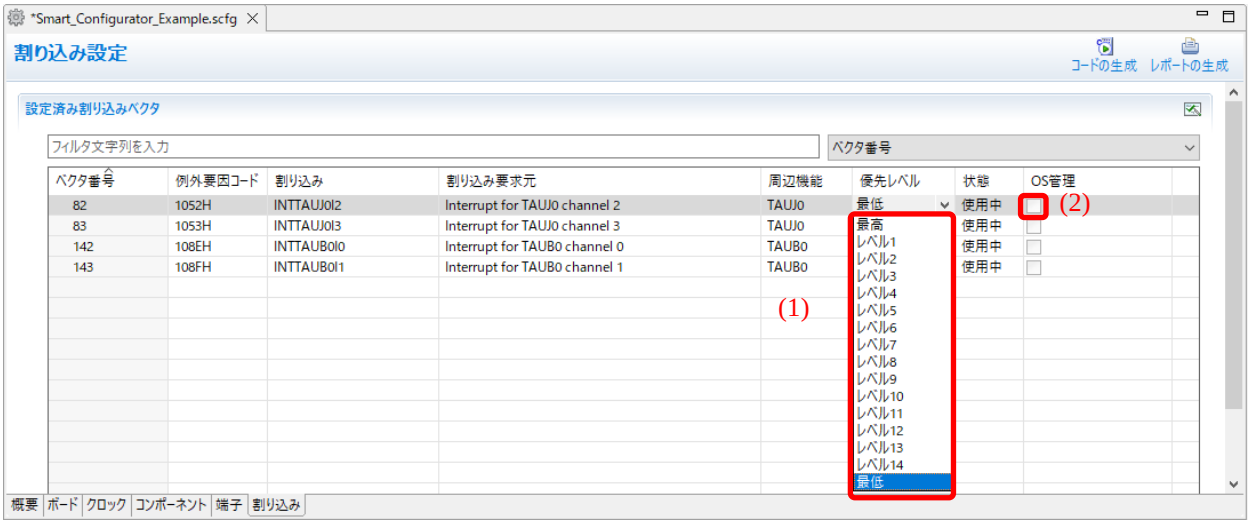


図 4-41 割り込み設定

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 PEn を選択します(4.3 システム設定(RH850/U2A のみ)を参照)。

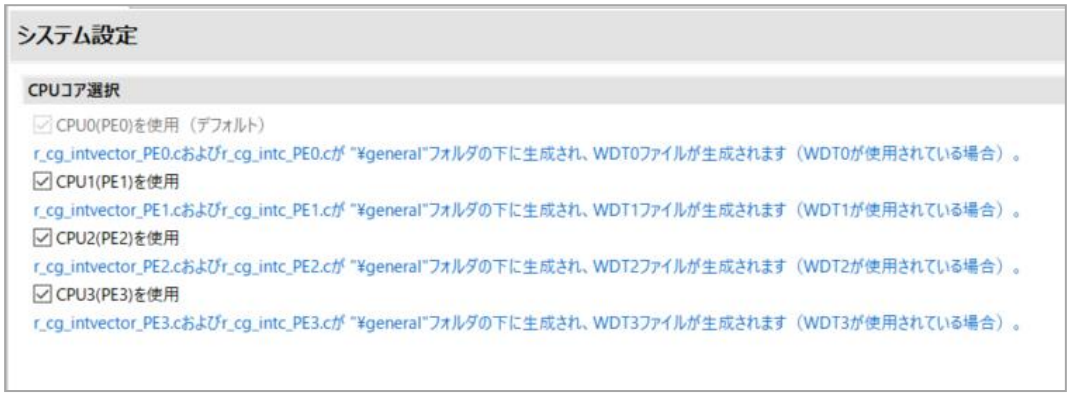


図 4-42 [システム]ページの PE 選択

(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。

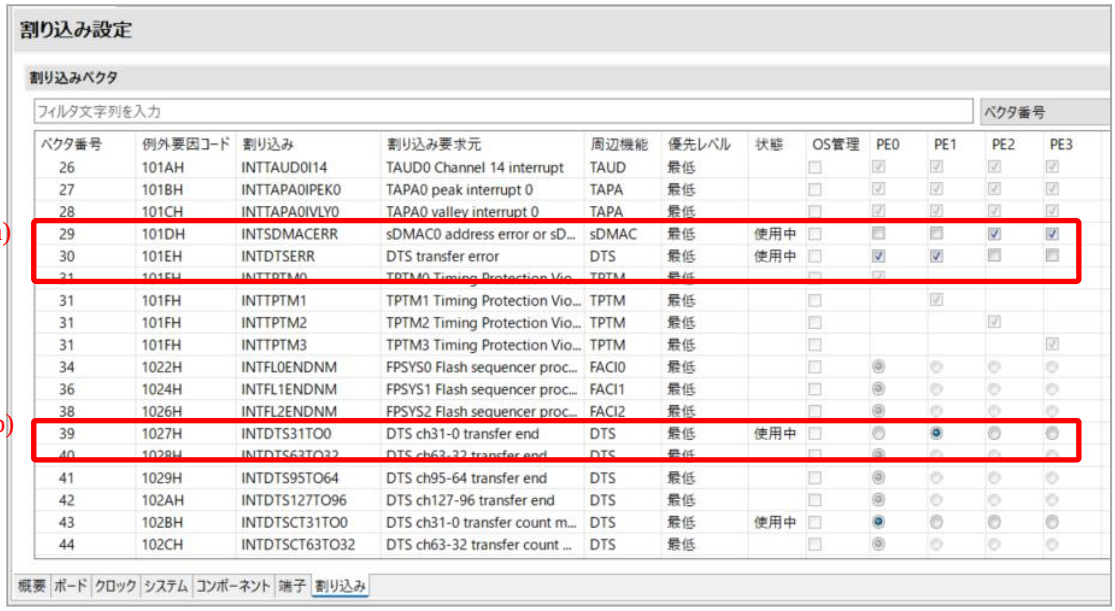


図 4-43　[割り込み]ページの PE 設定

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。

[割り込みハンドラ]欄にはデフォルトの割り込みハンドラとして「eiintn」が表示されます。 ユーザーはこの列を編集し、以下の基本ルールに従ってユーザー定義のハンドラ名 (デフォルト名「eiintn」を除く) を入力できfます。

・入力できる文字は、「a」～「z」、「A」～「Z」、「0」～「9」、「_」のみです。
・数字で始まる割り込みハンドラ名は入力できません。
・割り込みハンドラ名を空にすることはできません
・eiintn(n=現在の割り込み番号)以外の予約割り込みハンドラ名「eiintn」は入力できません。
・同じ割り込みハンドラ名を2つ入力することはできません。

【注】コンポーネントによって使用される割り込みハンドラは編集できません。

(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 4-45 割り込み有効/無効関数のコード例」を参照してください。

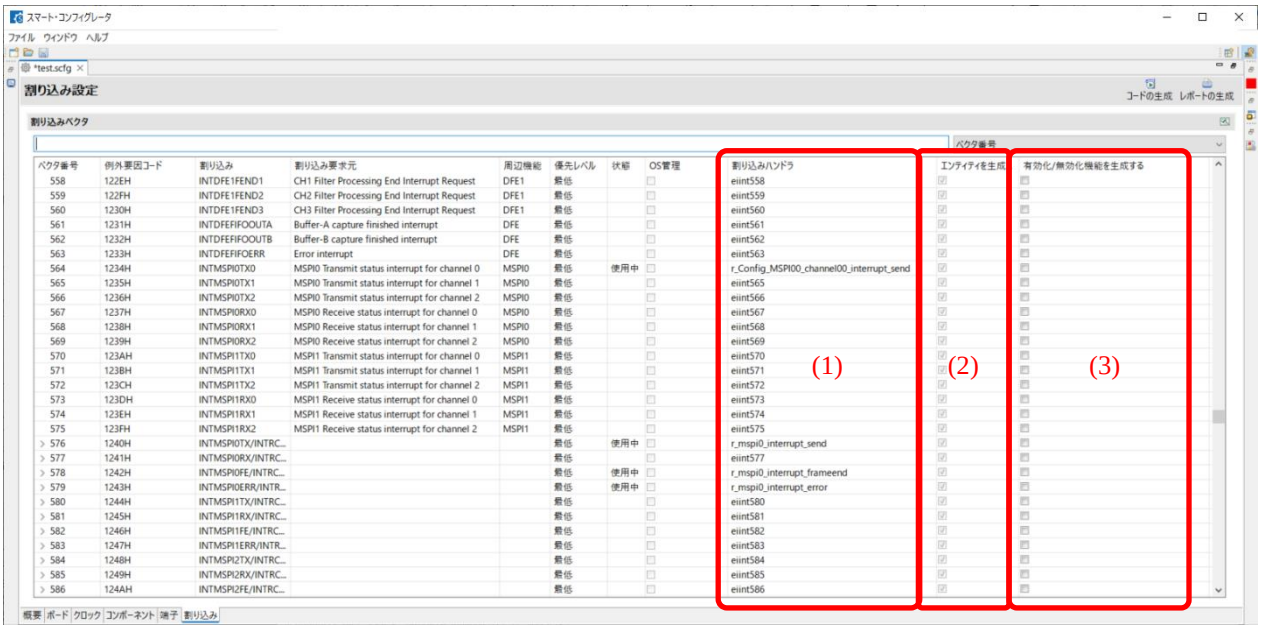


図 4-44　割り込みハンドラとエンティティの設定

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 4-45　割り込み有効/無効関数のコード例

5. 競合の管理

コンポーネントの追加、リソース変更、端子割り当ておよび割り込みの設定をすると、リソースの不一致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

リソースの競合がある場合、［コンポーネント］ページの［コンポーネントツリー］にエラーマーク が表示されます。（例：TAUB1）

［コンフィグレーションチェックビュー］にリソースの競合に関するメッセージが表示され、ユーザーにリソースの競合が発生しているコンポーネントのコンフィグレーション設定を知らせます。

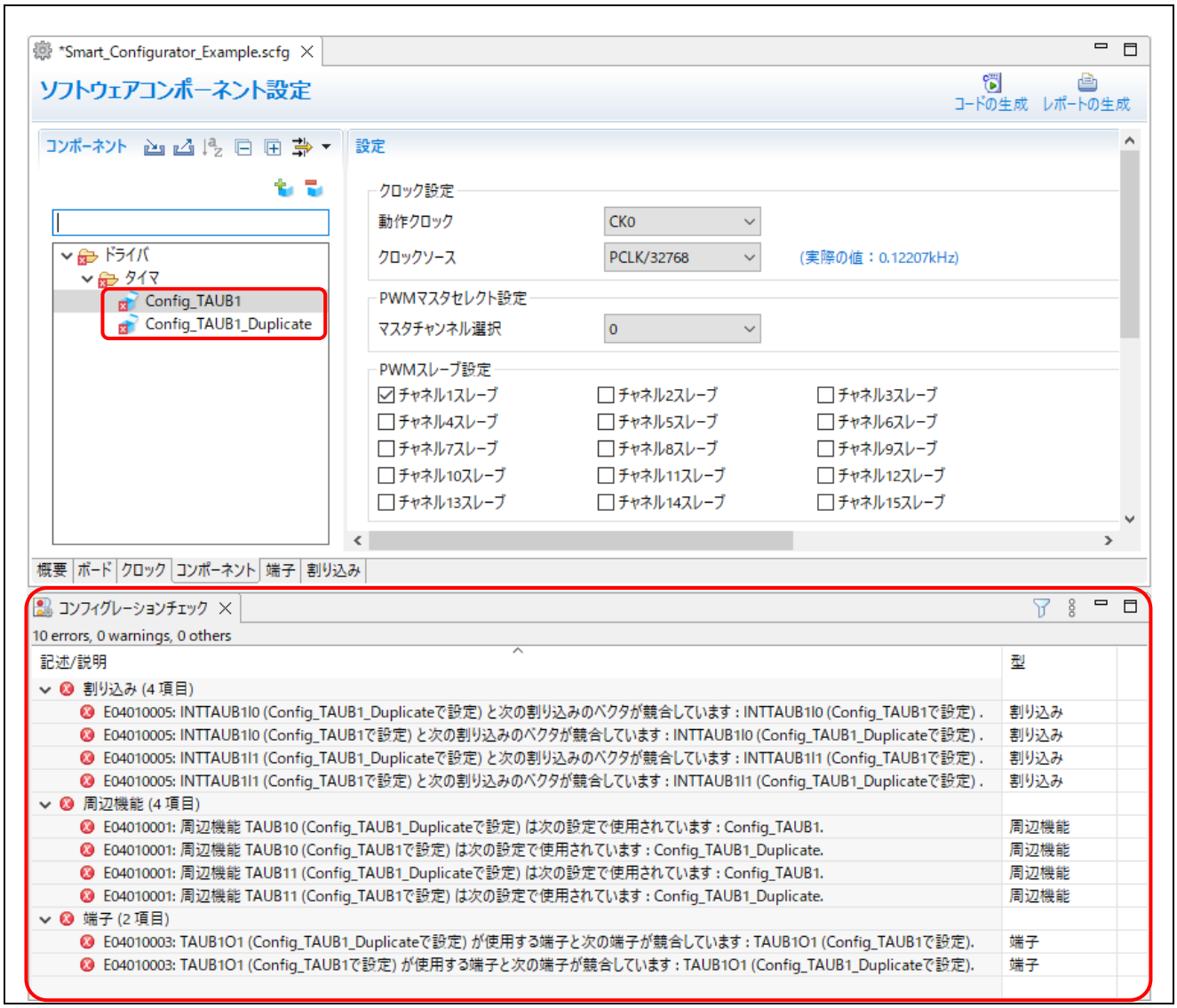


図 5-1 リソースの競合

5.2　端子の競合

複数の端子機能が同一端子に割り当てられると、端子ページでエラーマーク がツリーと端子機能リストに表示されます。

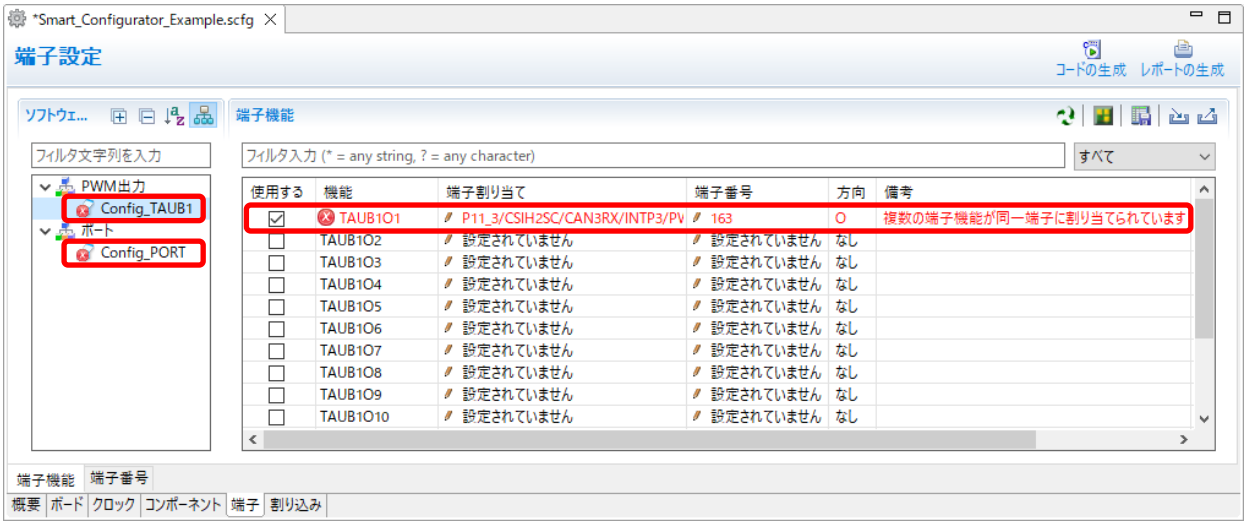


図 5-2 端子の競合（端子機能）

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。

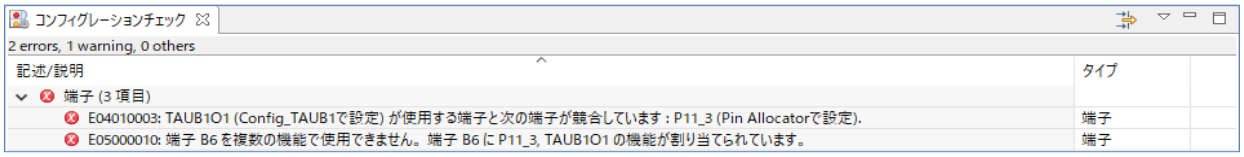


図 5-3 端子競合のメッセージ

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。

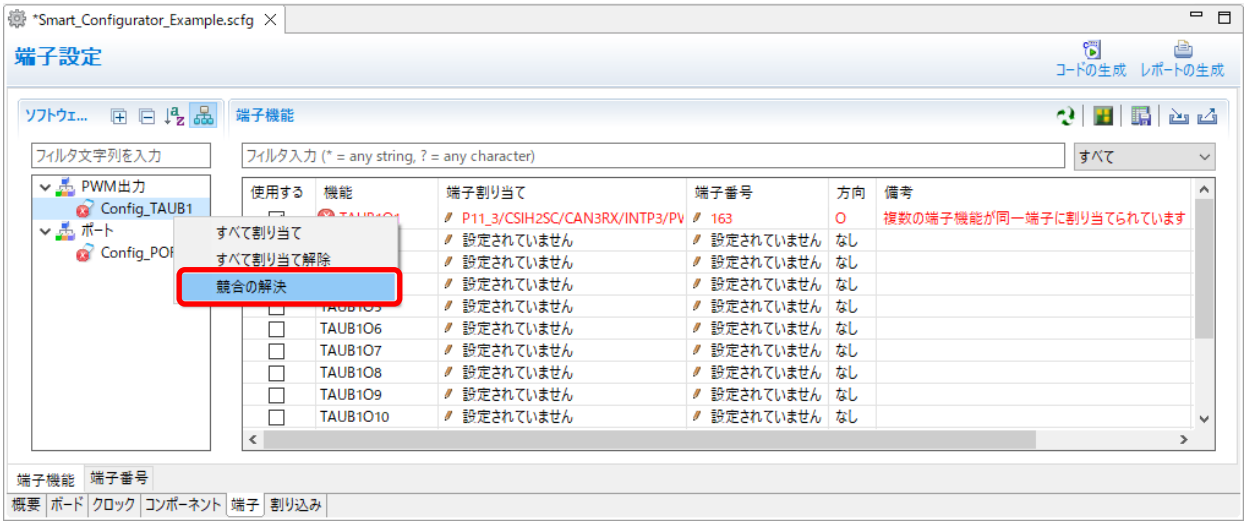


図 5-4 端子競合の解消

選択したノードの端子が、他のピンに再割り当てされます。

6. ソースコードの生成

6.1 コードの生成

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、コンポーネントのコンフィグレーション設定に応じたソースファイルを出力します。

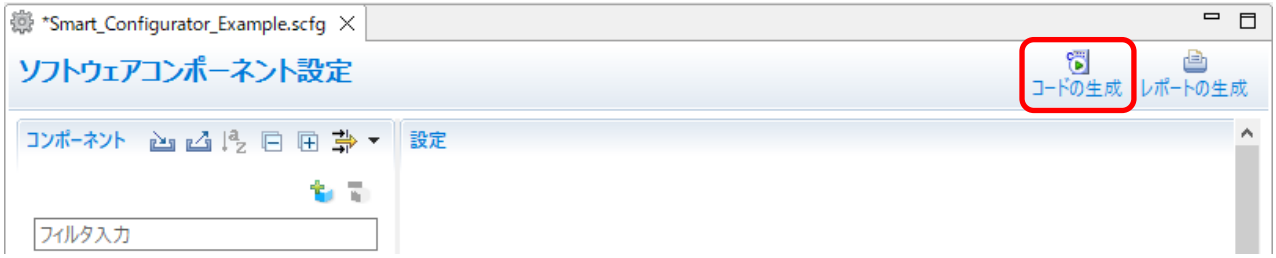


図 6-1 ソースファイルの生成

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、バックアップを生成します。詳細は、「6.6 生成ソースのバックアップ」を参照ください。

6.2　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「**図 6-2 IAR 向け生成ファイルの構成とファイル名**」に示します。「*ConfigName*」は、コンポーネントに設定されている構成名を示します。「*ProjectName*」はプロジェクト名を示します。

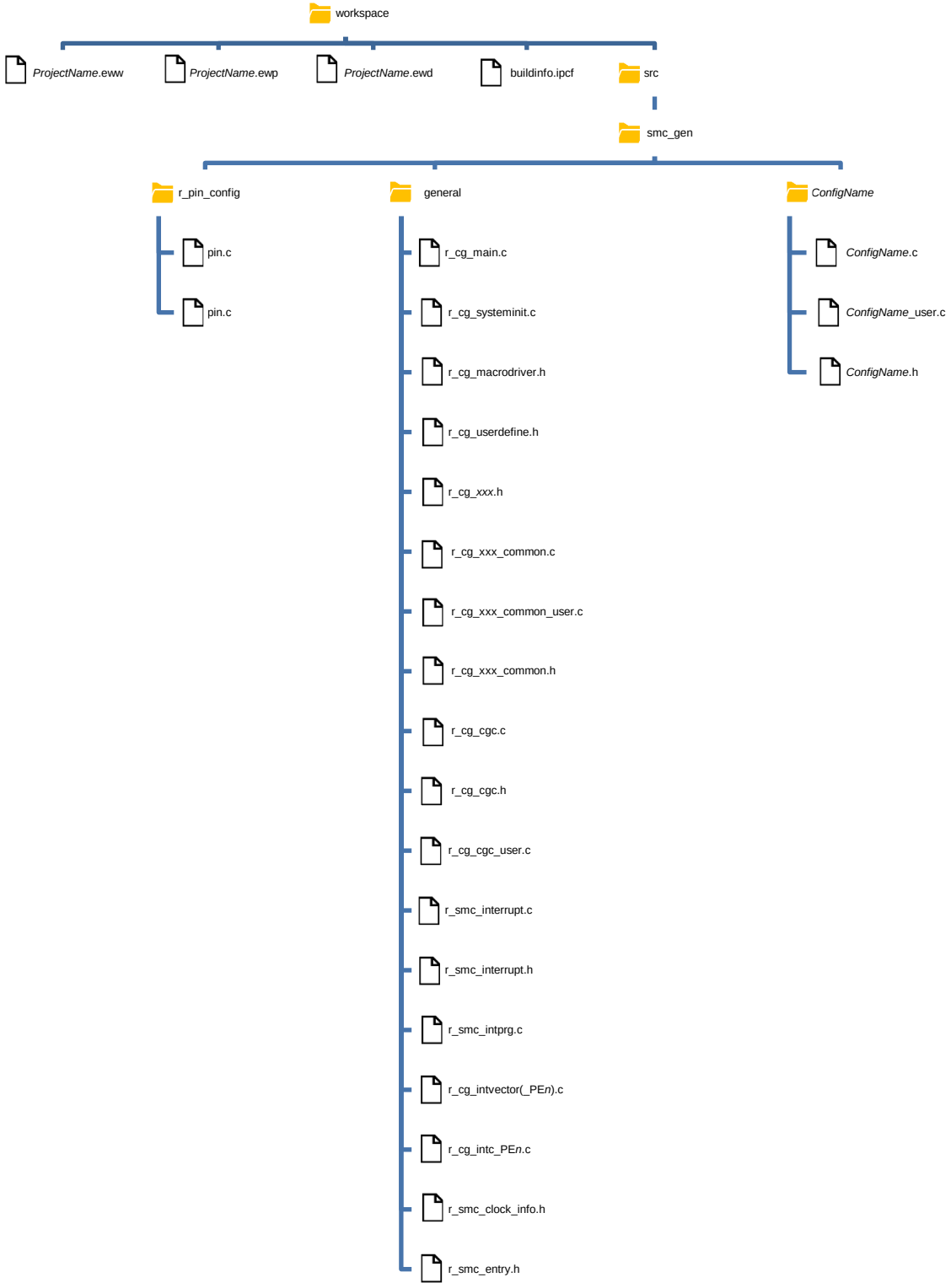


図 6-2 IAR 向け生成ファイルの構成とファイル名

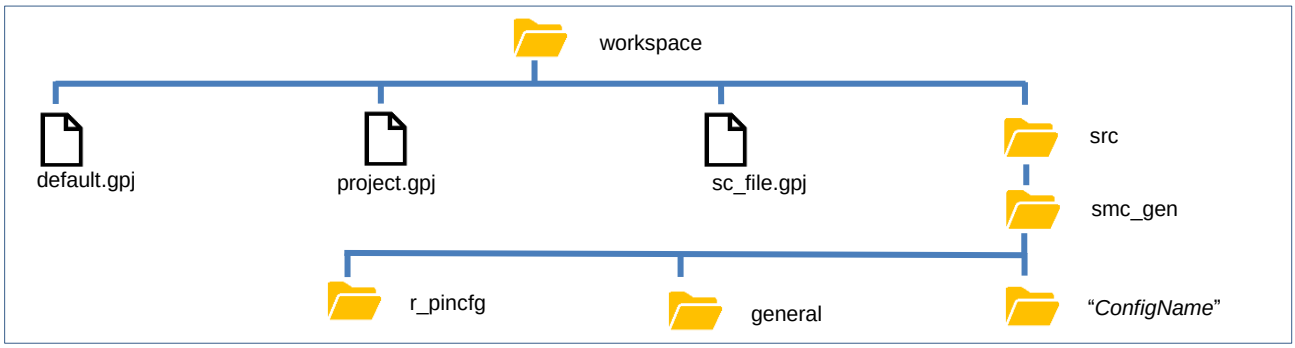


図 6-3 GHS 向け生成ファイルの構成とファイル名

【注】 smc_gen フォルダ下の生成ファイルとファイル名は、IAR EW と GHS MULTI で共通です。

表 6-1 生成ファイルの説明

フォルダ	ファイル	説明
Workspace	-	このフォルダはスマート コンフィグレータによって生成されるのではなく、スマート コンフィグレータのプロジェクトを作成するためにユーザーが指定するフォルダです。 スマート・コンフィグレータは、すべてのフォルダおよびファイルをこのフォルダ内に生成します。
	ProjectName.eww	IAREW 用のワークスペースファイルです。 スマート・コンフィグレータのプロジェクトを初めて作成するときに生成されます。
	ProjectName.ewp	IAREW 用のプロジェクト・ファイルです。 スマート・コンフィグレータのプロジェクトを初めて作成するときに生成されます。
	ProjectName.ewd	IAREW のデバッグファイルです。 スマート・コンフィグレータのプロジェクトを初めて作成するときに生成されます。
	buildinfo.ipcf	IAREW 用の接続ファイルです。 各コンポーネントのすべての生成ファイルのパスが含まれます。 スマート・コンフィグレータでコード生成を行うたびに生成されます。
	default.gpj	GHS MULTI の Top プロジェクト・ファイルです。 各コンポーネントのすべての生成ファイルのパスが含まれます。 スマート・コンフィグレータでコード生成を行うたびに生成されます。
	project.gpj	GHS MULTI の Program プロジェクト・ファイルです。 *sc_file.gpj* をインクルードします。 スマート・コンフィグレータでコード生成を行うたびに生成されます。
	sc_file.gpj	GHS MULTI の Sub プロジェクト・ファイルです。 各コンポーネントのすべての生成ファイルのパスが含まれます。 スマート・コンフィグレータでコード生成を行うたびに生成されます。

フォルダ	ファイル	説明
{*ConfigName*}	-	このフォルダは、プロジェクトに追加される CG ドライバ用に生成されます。このフォルダ内の API 関数には、*ConfigName*（設定名）を含んだ名称がつけられています。
	{*ConfigName*}.c	このファイルは、ドライバを初期化する関数 (R_*ConfigName*_Create) 、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。
r_pincfg	Pin.c	このファイルは常時生成されます。 [端子] ページで設定される全リソース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。 Pin.c での端子設定の関数プロトタイプを含みます。
general	-	このフォルダは常時生成されます。同じ周辺機能の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。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、クロック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(_PE0).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。
	r_smc_intprg.c	このファイルは、RH850/U2B グループのデバイスを選択した場合に生成されます。 [Interrupts] ページの設定に従って、すべての割り込みハンドラ エンティティが含まれます。

フォルダ	ファイル	説明
	r_smc_clock_info.h	このファイルは、RH850/U2B グループのデバイスを選択した場合に生成されます。 クロック設定マクロ定義が含まれています。
	*r_cg_xxx_common_user.c(Note*1)*	
	r_smc_entry.h	

＊1：*xxx* はコンポーネント名を意味します。

6.3　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。

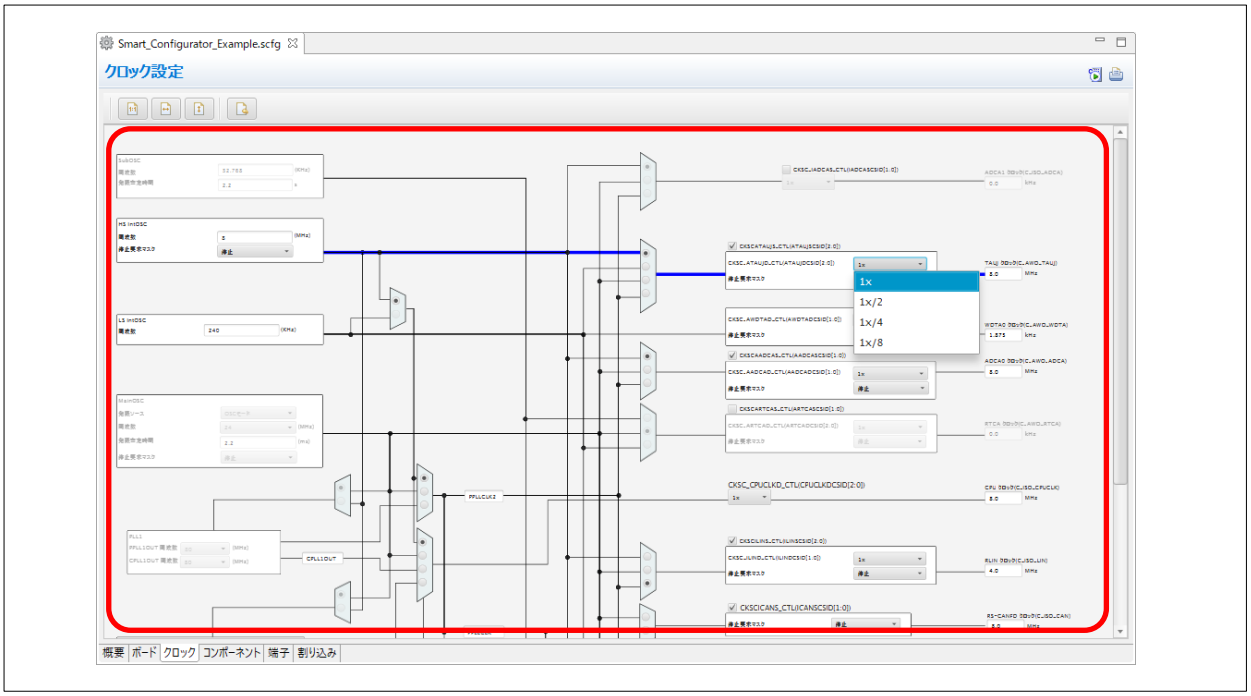


図 6-4 クロックソースの設定

表 6-2 クロックに関する生成ファイルの説明

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	クロック初期化後に、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。
	r_smc_clock_info.h	クロックに関連するマクロ	これらのマクロは、[クロック] ページのクロック設定用です。

6.4　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) コンポーネントが使用する端子の初期化

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

表 6-3 端子の初期化を行うファイル

フォルダ	ファイル	関数	説明
{ConfigName}	*{ConfigName}.c*	*R_ConfigName_Create*	このコンポーネントが使用する端子を 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

［端子］ページで設定したすべての端子機能の初期化コードについては、¥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）

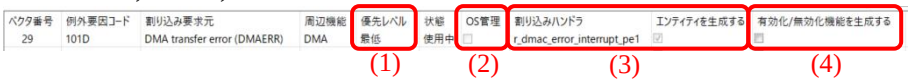
表 6-4 全端子の初期化の参照ファイル

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。

6.5　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 6-5 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。

6.6　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、スマート・コンフィグレータは <ProjectDir>¥trash¥<Date-and-Time>に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。

7. 統合開発環境への取り込み

スマート・コンフィグレータで出力したソースコードを統合開発環境プラットフォームに読み込みます。

7.1 IAR Embedded Workbench への取り込み

7.1.1 「IAR RH850 Toolchain」選択時

スマート・コンフィグレータは、使用するコンパイラに IAR 環境を選択したとき、ソースファイルとともに関連ファイル(.eww/.ewp/.ewd/r_cg_main.c)を生成します。 ユーザーが IAR Embedded Workbench でプロジェクト・ファイルを作成する必要はありません。

下記の手順で使用してください。

(1) IAR Embedded Workbench の [ファイル] メニューから [ワークスペースを開く] を選択します。

(2) [ワークスペースを開く] ダイアログボックスで、プロジェクト・ファイルが保存されているフォルダを参照し、プロジェクト・ファイル (*.eww) を選択して [開く] ボタンをクリックします。

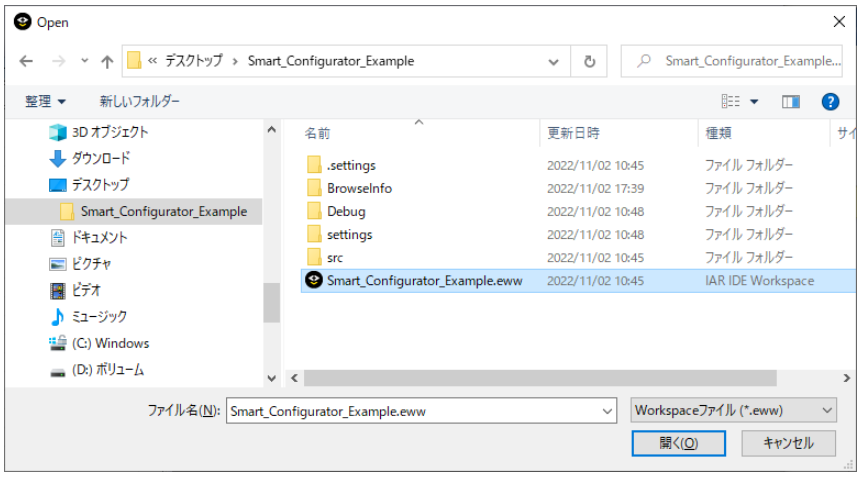


図 7-1 *.eww ファイルの読み込み

(3) スマート・コンフィグレータの生成ファイルが、ワークスペースに追加されます。

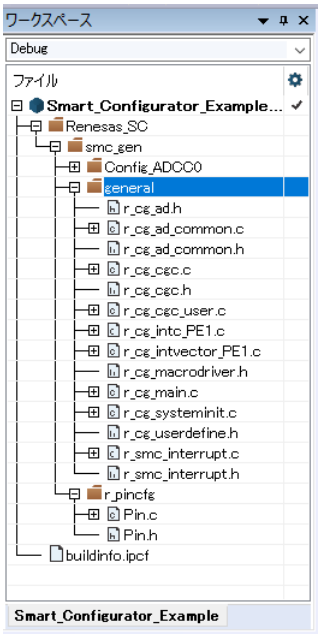


図 7-2 IAR ワークスペース／プロジェクトへのソースファイルの追加

(4) IAR Embedded Workbench の [プロジェクト] メニューから [オプション] を選択します。

(5) [ノード “*ProjectName*” のオプション] ダイアログボックスで、[ターゲット]タブのデバイスを対象デバイスに変更します。

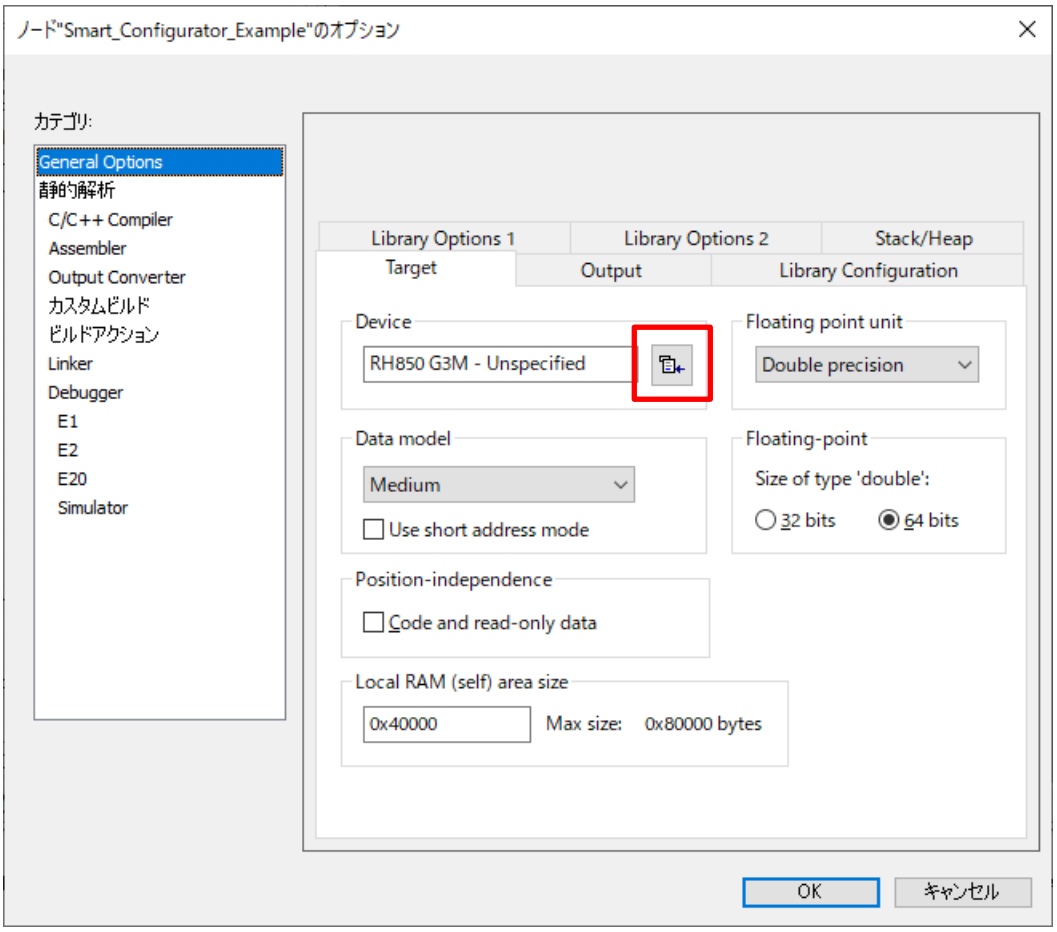


図 7-3　ターゲット・デバイスの変更

7.1.2　「All Toolchain (CC-RH, GHS, IAR)」選択時

スマート・コンフィグレータを単体で起動し、ツールチェーンで「All Toolchain（CC-RH、GHS、IAR）」を選択した場合、スマート・コンフィグレータは、CC-RH、IAR、GHS のすべてに対応したソースファイルを生成します。 生成されたファイルを IAR Embedded Workbench に取り込む場合は、以下の手順に従ってください。

(1) IAR Embedded Workbench で新しいプロジェクトを作成します。
(2) 以下に示すように、[Project] -> [Add Project Connection] を選択します。

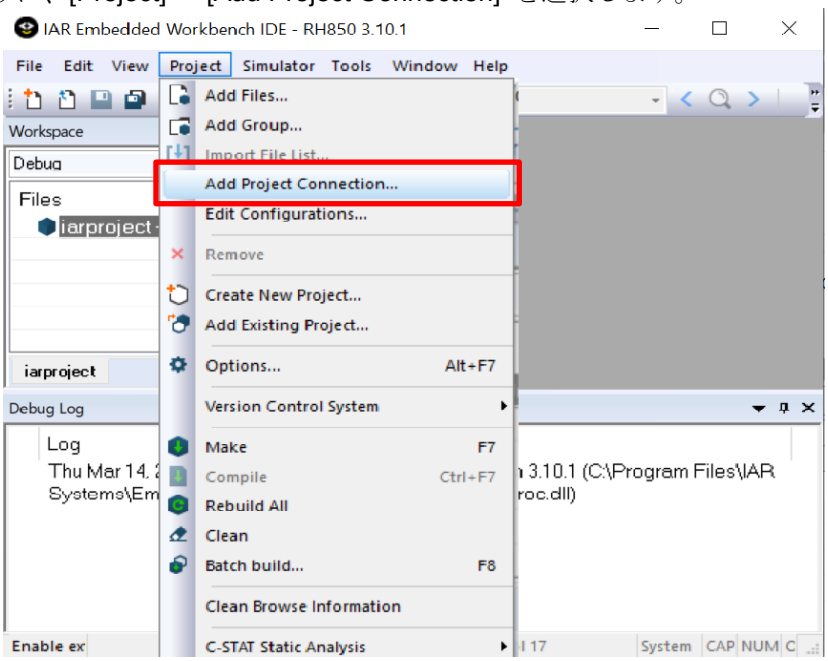


図 7-4 Add Project Connection

(3) [IAR Project Connection]を選択します。

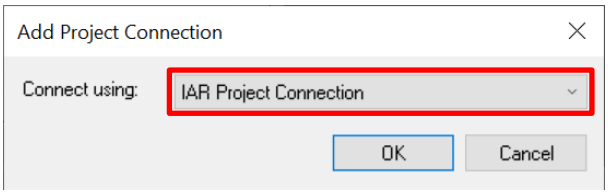


図 7-5 Connect using の選択

(4) [Select Project Connection File]ビューで、下図に示すようにスマート・コンフィグレータのプロジェクトフォルダに移動し、buildinfo.ipcf を選択します。

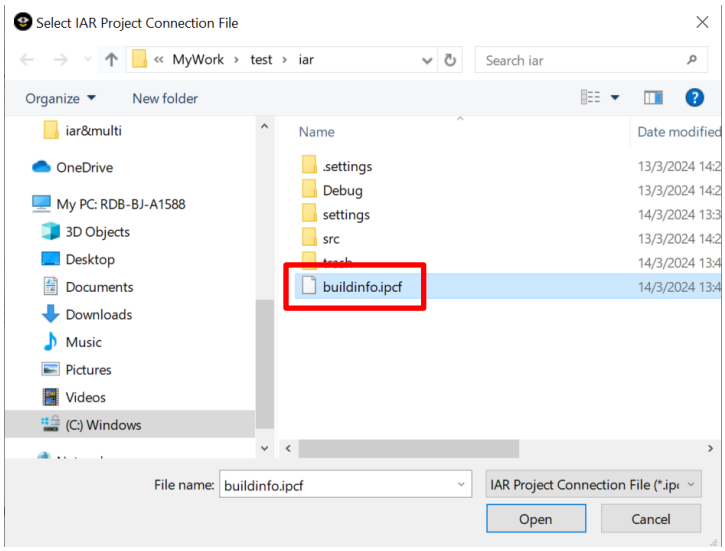


図 7-6 buildinfo.ipcf の選択

(5) IAR Embedded Workbench の[Project]メニューから[Options…]を選択します。

(6) [Options for node “ProjectName] ビューで [[C/C++ Compiler] を選択し、[…] をクリックして手動で "iodefine.h" のインクルードパスを追加します。

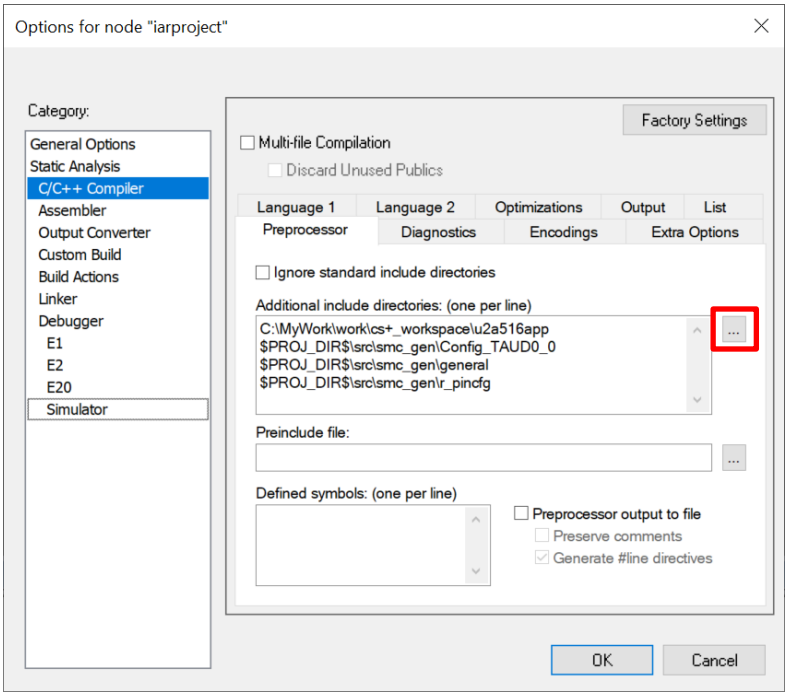


図 7-7 インクルードパスの追加

(7) [Options for node “ProjectName”]ビューで[General Options]を選択し、[Target]タブでスマート・コンフィグレータの設定ファイル作成時に選択したデバイスに合わせて[Device]を変更します。

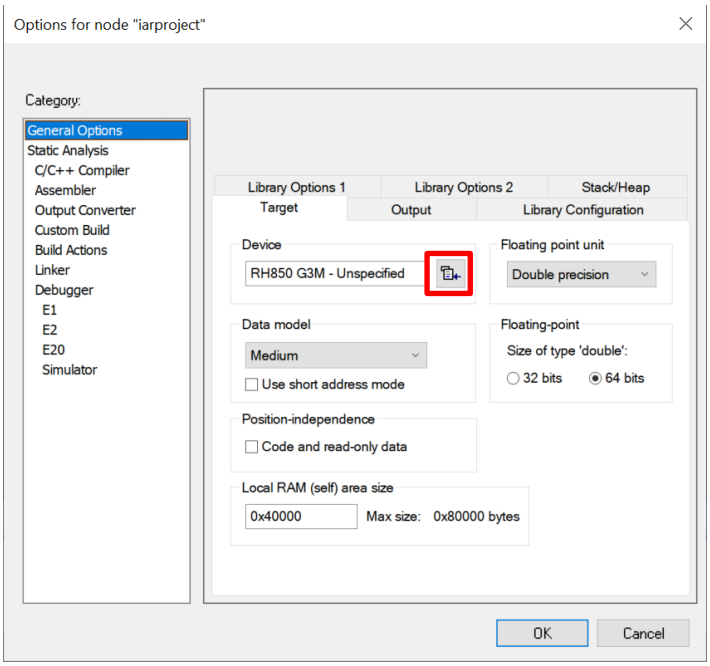


図 7-8 デバイスの設定

7.2　GHS MULTI への取り込み

7.2.1　「GHS RH850 Toolchain」選択時

スマート・コンフィグレータは、使用するコンパイラに GHS 環境を選択したとき、ソースファイルと共に GHS MULTI のプロジェクト・ファイル（.gpj）も生成します。プロジェクト・ファイルには、ソースファイルの登録情報が含まれています。再コード生成時、プロジェクト・ファイルを出力するため、スマート・コンフィグレータで設定変更をした場合、ソースファイルの追加や削除をユーザーが行う必要がありません

下記の手順で使用してください。

(1) MULTI の［Components］メニューから［Open Project Manager…］を選択します。

(2) ［Select a project to open］ダイアログが表示されますので、コンフィグレーションファイルが保存されているフォルダを参照し、default.gpj を選択して［Open］ボタンをクリックします。

(3) プロジェクトに、スマート・コンフィグレータが出力したソースファイルが追加されます。

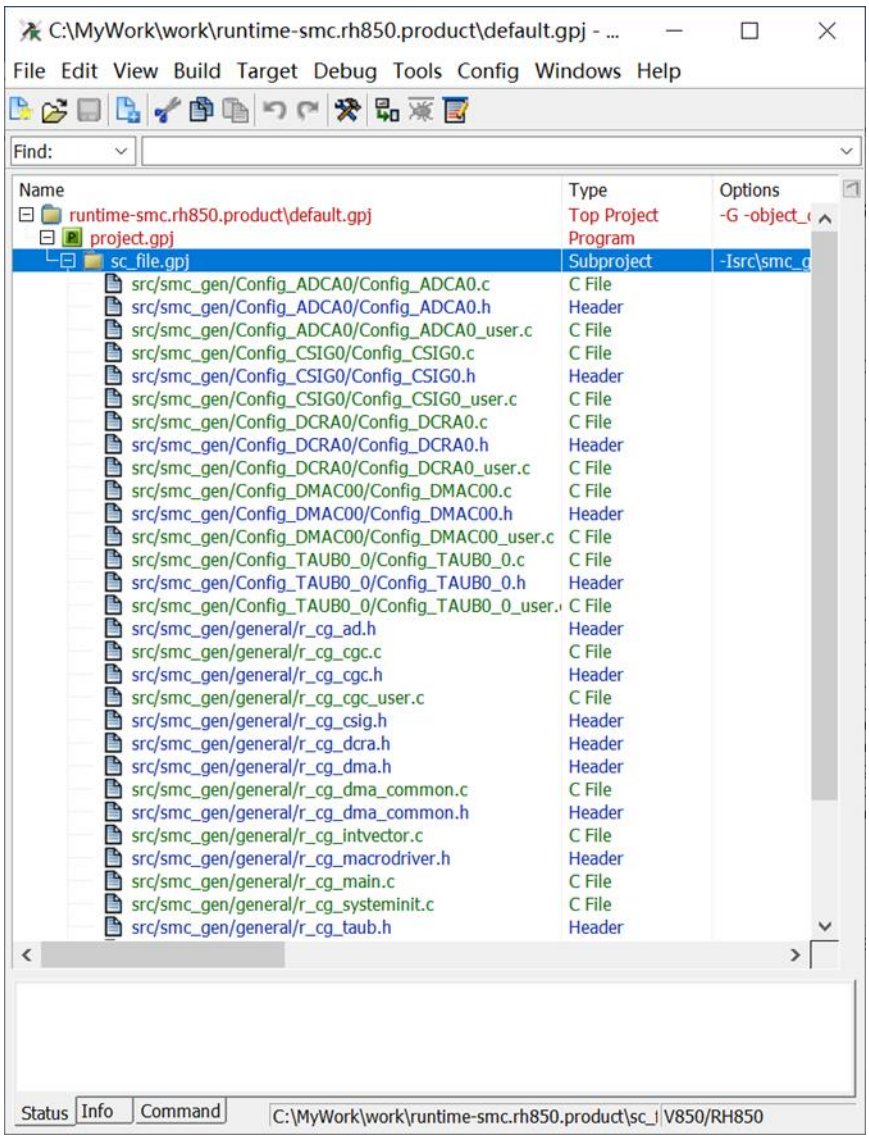


図 7-9　GHS MULTI のワークスペースへのファイルの追加

7.2.2　「All Toolchain (CC-RH, GHS, IAR)」選択時

スマート・コンフィグレータを単体で起動し、ツールチェーンで「All Toolchain（CC-RH、GHS、IAR）」を選択した場合、スマート・コンフィグレータは、CC-RH、IAR、GHS のすべてに対応したソースファイルとともに GHS MULTI 用のプロジェクトファイル（sc_file.gpj）を生成します。 生成されたファイルを GHS MULTI のワークスペースに取り込む場合は、以下の手順に従ってください。

(1) GHS MULTI のウィザードに従って新しいプロジェクトを作成します。

(2) GUI に表示されているプロジェクト・ファイル(.gpj)を右クリックし、[Add File into “projectname”.gpj]を選択します。

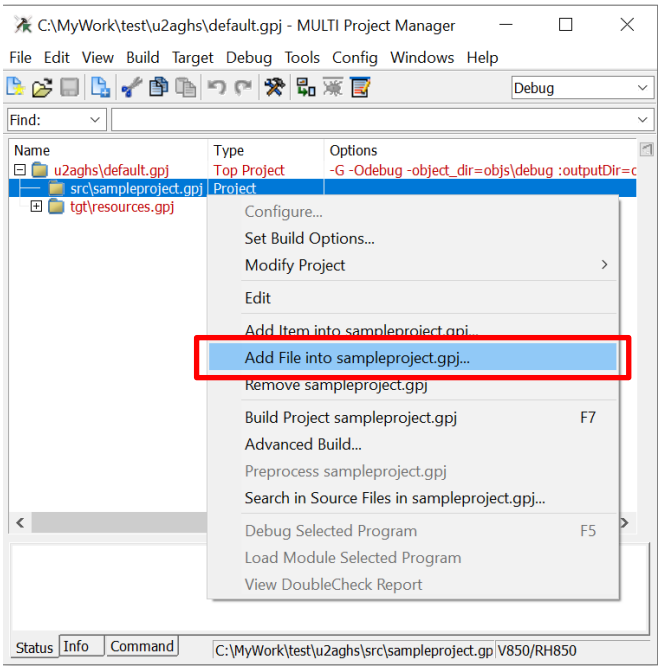


図 7-10 プロジェクト・ファイルの追加

(3) [Choose file(s) to add:]ビューで、スマート・コンフィグレータのプロジェクト・ファイルが存在するフォルダに移動し、ファイル「sc_file.gpj」を選択して [追加] をクリックすると、スマート・コンフィグレータで生成されたすべてのファイルが追加されます。

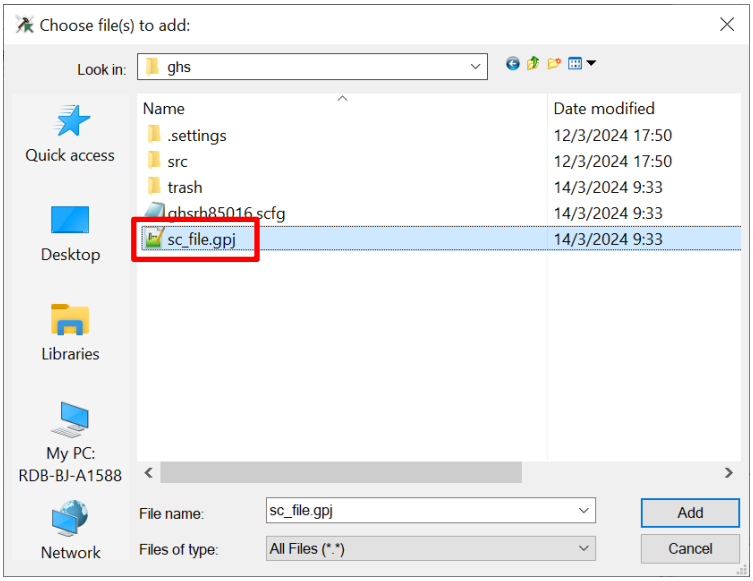


図 7-11 「sc_file.gpj」の選択

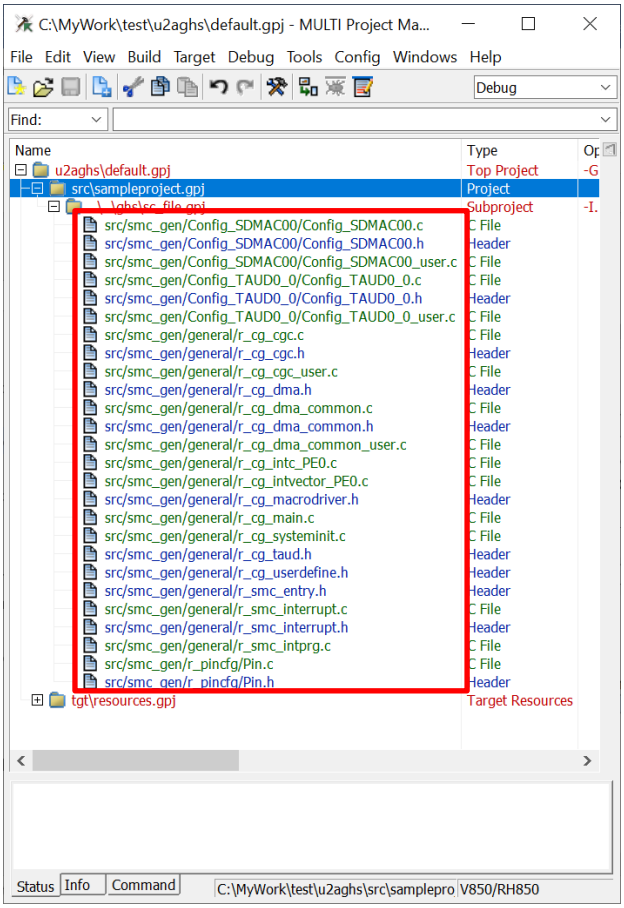


図 7-12 生成コードの取り込み

(4) GUI に表示されているプロジェクト・ファイル(.gpj)を右クリックし、[Set Build Options…]を選択し、ポップアップウィンドウの[Include Directories]に“iodefine.h”へのパスを設定します。

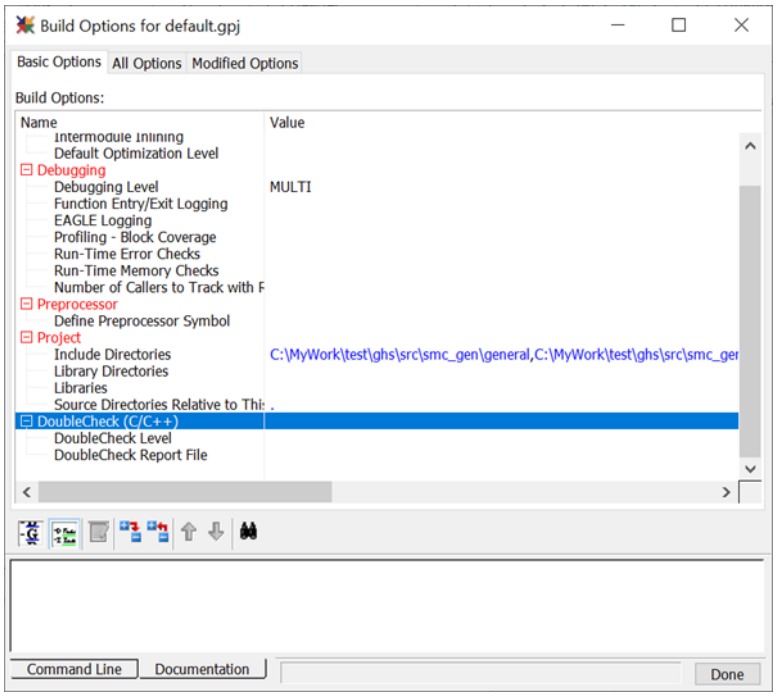


図 7-13 インクルードパスの設定

8. ユーザープログラムの生成

統合開発環境でユーザープログラムを作成します。ここでは、スマート・コンフィグレータが生成したソースファイルにユーザーコードを追加する方法について説明します。

8.1 ユーザーコードの追加方法

ソースファイル内の以下のコメントで囲まれた部分にユーザーコードを追加します。以下のコメントで囲まれた部分は、再コード生成時に再生成されたソースファイルにマージされます。

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

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

コンポーネントごとに 3 つのファイルを生成します。ファイル名は、「< ConfigName >.h」、「< ConfigName >.c」、「< ConfigName >_user.c」となります。ユーザー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と、「＜ConfigName >_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 TAUB1 の例（Config_TAUB1_user.c）を示します。

```
/***********************************************************************************
Pragma directive
***********************************************************************************/
/* Start user code for pragma. Do not edit comment generated here */
/* End user code. Do not edit comment generated here */

/***********************************************************************************
Includes
***********************************************************************************/
#include "r_cg_macrodriver.h"
#include "r_cg_userdefine.h"
#include "Config_TAUB1.h"
/* Start user code for include. Do not edit comment generated here */
/* End user code. Do not edit comment generated here */

/***********************************************************************************
Global variables and functions
***********************************************************************************/
/* Start user code for global. Do not edit comment generated here */
/* End user code. Do not edit comment generated here */

/***********************************************************************************
* Function Name: R_Config_TAUB1_Create_UserInit
* Description  : This function adds user code after initializing the TAUB1 channel
* Arguments    : None
* Return Value : None
***********************************************************************************/

void R_Config_TAUB1_Create_UserInit(void)
{
    /* Start user code for user init. Do not edit comment generated here */
    /* End user code. Do not edit comment generated here */
}
```

```
/***********************************************************************************
* Function Name: r_Config_TAUB1_channel0_interrupt
* Description  : This function is TAUB10 interrupt service routine
* Arguments    : None
* Return Value : None
***********************************************************************************/
#pragma interrupt r_Config_TAUB1_channel0_interrupt(enable=false, channel=256, fpu=true,
callt=false)
void r_Config_TAUB1_channel0_interrupt(void)
{
    /* Start user code for r_Config_TAUB1_channel0_interrupt. Do not edit comment generated
here */
    /* End user code. Do not edit comment generated here */
}

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

9. レポートの生成

スマート・コンフィグレータは、プロジェクトの設定情報をレポートに出力できます。レポートを出力するには、以下の手順で行います。

9.1 全設定内容レポート(PDF/txt 形式)

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。

図 9-1 全設定内容レポート出力

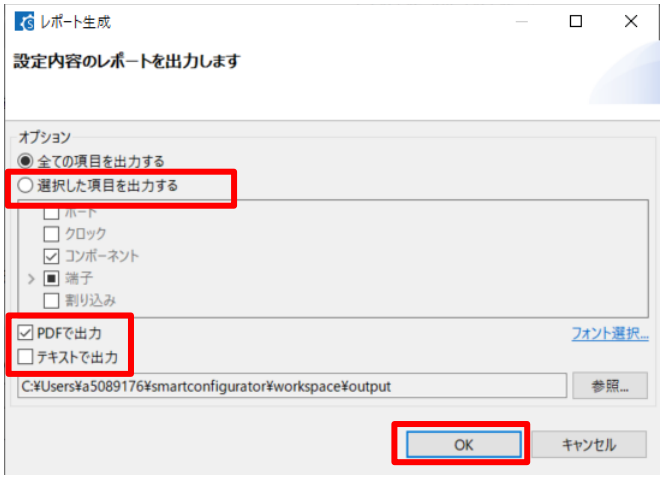


図 9-2 レポート出力ダイアログ

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。

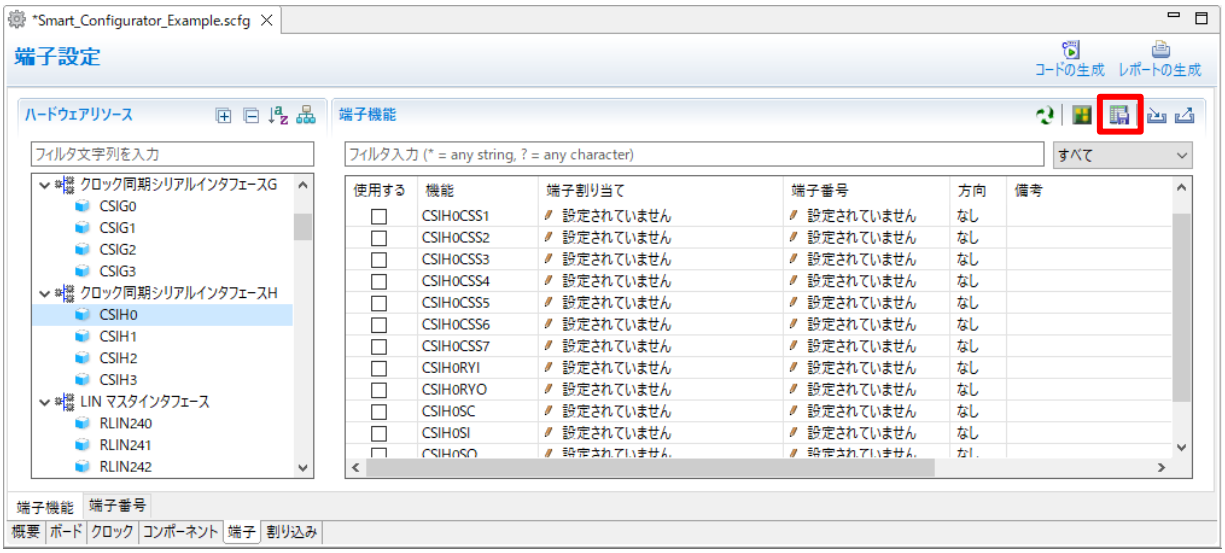


図 9-3 端子機能リスト、端子番号リスト出力(csv 形式)

9.3　MCU パッケージ図(png 形式)

MCU 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 パッケージ図を出力します。

図 9-4 MCU 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。

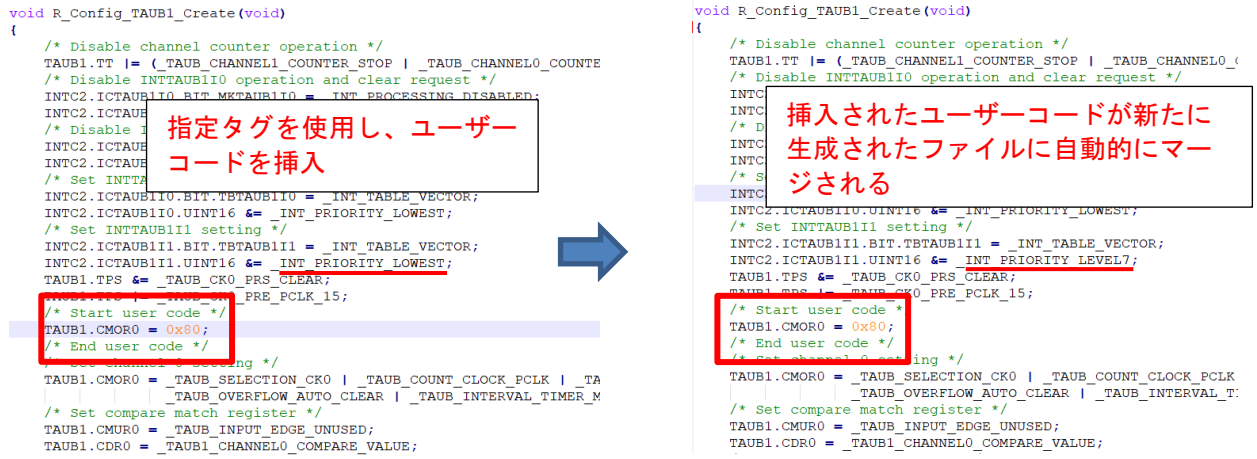


図 10-2 ユーザーコードの保護機能

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。

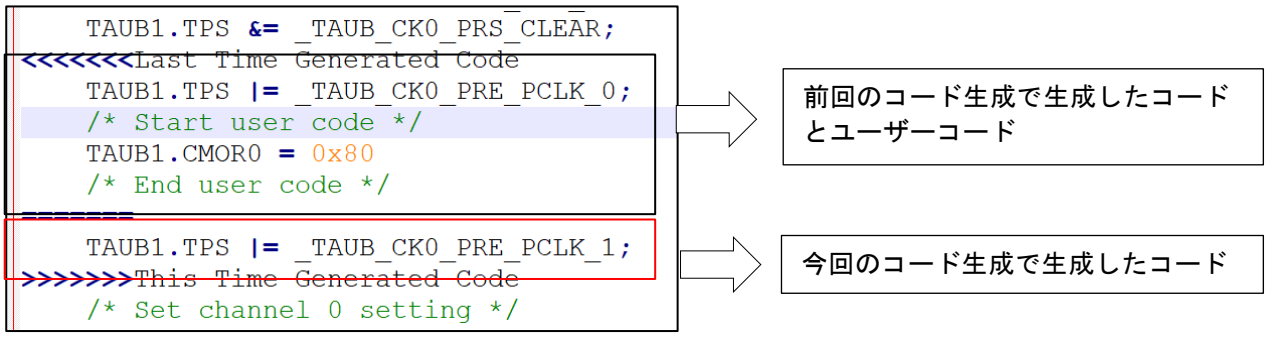


図 10-3 生成された競合コード

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。

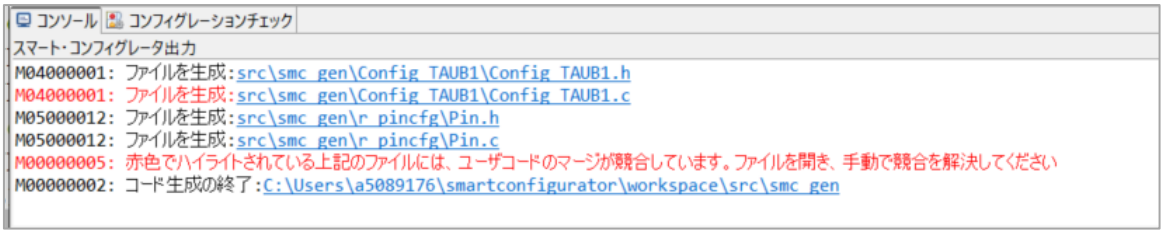


図 10-4 競合のメッセージ

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**)

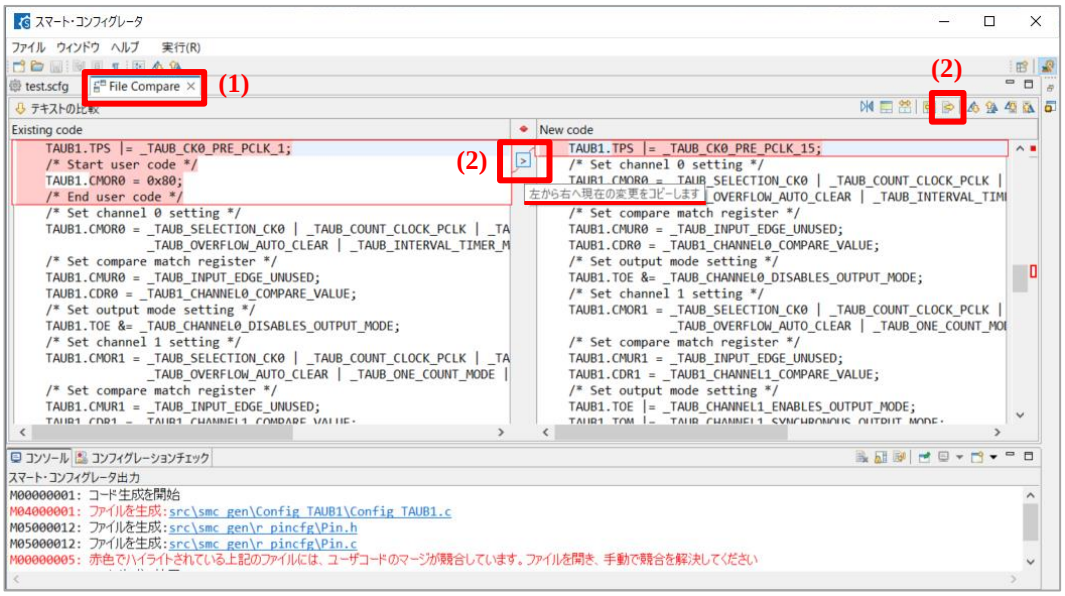


図 10-5　生成された競合コード

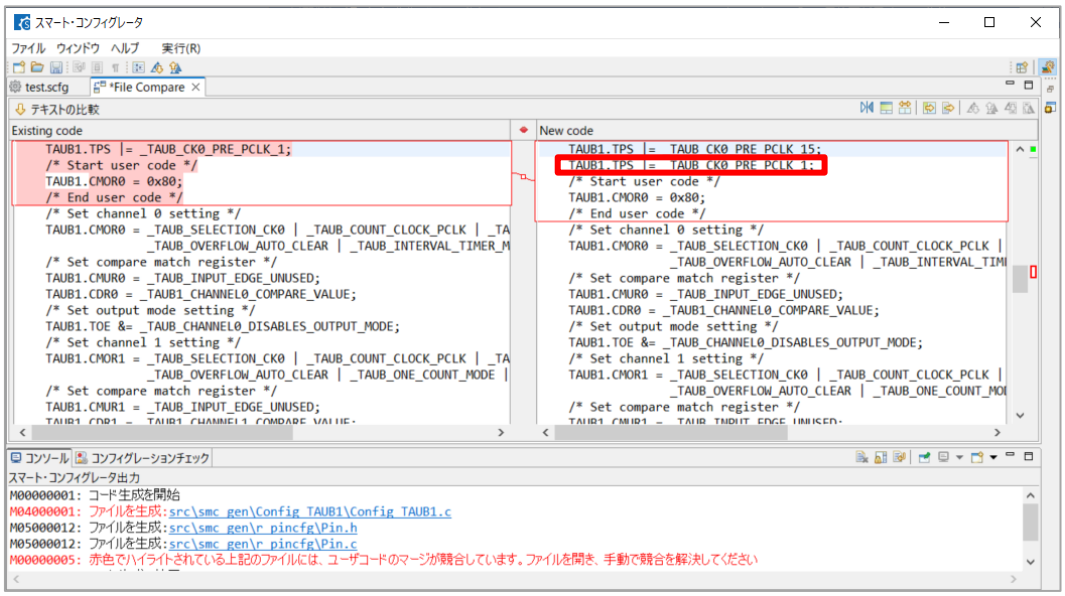


図 10-6　「左から右へ現在の変更をコピー」後のコード

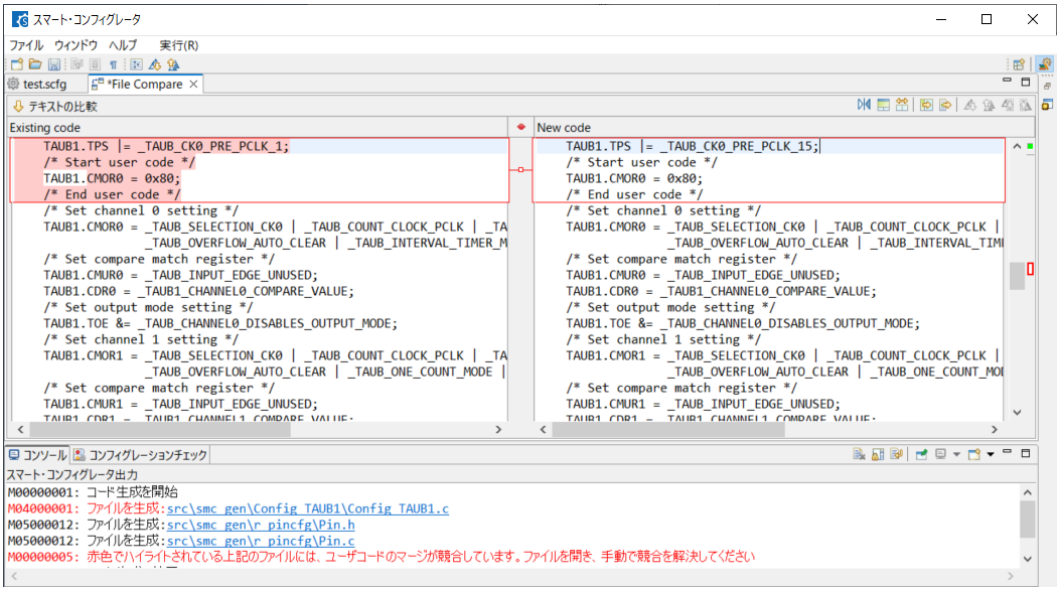


図 10-7 競合を解決した後のコード

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、ヘルプを参照ください。

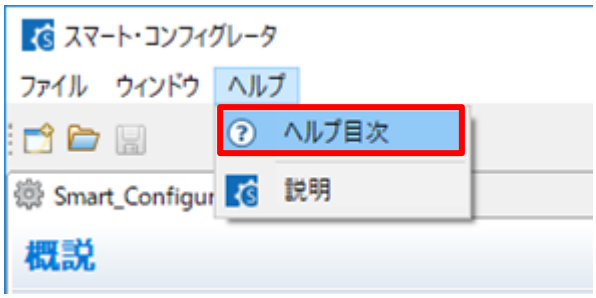


図 11-1 ヘルプ表示

概要ページからも参照できます。

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

スマート・コンフィグレータ　RH850/F1KM 用サンプル・プロジェクト説明書（R01AN4422）

ユーザーズマニュアル：開発環境

最新版を各社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.	章	改訂内容
1.00		新規作成
1.01		4.3.6 コンポーネント構成のエクスポート追加 4.3.7 コンポーネント構成のインポート追加
		4.3.8 コンポーネントの基本設定 追加
		4.5.2 割り込みハンドラと生成エンティティ設定の変更 追加
		6.2 ソースファイルの生成 更新
		7.1 IAR Embedded Workbench への読み込み
1.02	3.1	図を更新
	3.3.1	図を更新
	3.4.4	図と説明を更新
	4.1.1	図を追加
	4.1.2	【注】を追加
	4.3	新規追加
	4.4.3	複数選択の記述を追加
	4.4.8	図と説明を更新
	4.5	図を更新
	4.5.2	図と説明を更新
	4.5.3	新規追加
	4.5.6	新規追加
	4.5.7	新規追加
	4.5.8	新規追加
	4.6.3	新規追加
	6.2	図と表に下記のファイルを追加。 R_cg_xxx_common_user.c R_smc_entry.h r_cg_intc_PEn.c
	6.5	図と表を追加
	7.2	図を追加
	9.1	図と説明を更新
	9.3	図を更新
	10	新規追加
1.03	要旨	RH850 スマート・コンフィグレータ V1.10.0 に更新
	3.4	図 3-5 メインウィンドウ、図 3-6 スマート・コンフィグレータビューの更新
	4.6.2、4.6.3、6.5	割り込み機能の記述を更新
	10	ユーザー コード保護機能の記述を更新
1.04	3.3.1	「All Toolchain (CC-RH, GHS, IAR)」選択時の説明を追加。
	4.6.3	割り込みハンドラ名のルールを追加
	7.1.2	新規追加
	7.2.2	新規追加

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. 未使用端子の処理
   【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。
   CMOS製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI周辺のノイズが印加され、LSI内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。
2. 電源投入時の処置
   【注意】電源投入時は，製品の状態は不定です。
   電源投入時には、LSIの内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。
   外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。
   同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. リザーブアドレス（予約領域）のアクセス禁止
   【注意】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、将来の機能拡張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。
4. クロックについて
   【注意】リセット時は、クロックが安定した後、リセットを解除してください。
   プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。
   リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子
   （または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
5. 製品間の相違について
   【注意】型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。
   同じグループのマイコンでも型名が違うと、内部ROM、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。