

RH850 スマート・コンフィグレータ

R20AN0516JJ0130

Rev.1.30

ユーザーガイド: CS+編

2024.04.22

要旨

本アプリケーションノートでは、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・コンフィグレータの生成ファイルをCS+のプロジェクトに追加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CS+の対象バージョンは以下の通りです。

- ・ CS+ (CS+ for CC) V8.11.00以降
- ・ RH850 スマート・コンフィグレータ V1.11.0以降
- ・ CS+ RH850スマート・コンフィグレータ通信プラグイン V1.11.00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/smart-configurator>

目次

1. 概要.....	4
1.1 目的.....	4
1.2 特長.....	4
2. 使用前の準備.....	5
2.1 統合開発環境 CS+ (CS+ for CC)の準備.....	5
2.2 スマート・コンフィグレータのインストール.....	5
2.3 統合開発環境 CS+の設定.....	5
2.3.1 プラグイン設定状態の確認.....	5
2.3.2 実行パス設定状態の確認.....	6
2.4 スマート・コンフィグレータのアンインストール.....	7
2.5 サンプル・プロジェクトの準備.....	8
3. スマート・コンフィグレータの操作方法.....	9
3.1 操作手順.....	9
3.2 起動.....	10
3.3 プロジェクト情報の保存先.....	10
3.4 ウィンドウ.....	11
3.4.1 メインメニュー.....	12
3.4.2 ツールバー.....	12
3.4.3 スマート・コンフィグレータビュー.....	13
3.4.4 MCU/MPU パッケージビュー.....	14
3.4.5 コンソールビュー.....	15

3.4.6	コンフィグレーションチェックビュー	15
4.	周辺機能の設定	16
4.1	ボード設定	16
4.1.1	デバイス選択	16
4.1.2	ボード選択	17
4.1.3	ボード設定のエクスポート	18
4.1.4	ボード設定のインポート	18
4.2	クロック設定	19
4.3	システム設定(RH850 U2A のみ)	20
4.4	コンポーネント設定	21
4.4.1	コード生成コンポーネントの追加方法	21
4.4.2	コンポーネント一覧とハードウェア一覧との切り替え	23
4.4.3	ソフトウェアコンポーネントの削除	24
4.4.4	コンポーネントのコンフィグレーション設定	25
4.4.5	コンポーネントのリソース変更	26
4.4.6	コンポーネントの一般設定	28
4.5	端子設定	29
4.5.1	ソフトウェアコンポーネントの端子配置変更	30
4.5.2	MCU/MPU パッケージを使用した端子の設定	31
4.5.3	端子機能から端子番号の表示	32
4.5.4	端子設定のエクスポート	32
4.5.5	端子設定のインポート	33
4.5.6	ボードの端子情報を使用した端子設定	34
4.5.7	端子フィルタ機能	34
4.5.8	端子エラー/警告の設定	35
4.6	割り込み設定	36
4.6.1	割り込み優先レベルと OS 管理の設定	36
4.6.2	PE _n の設定の変更(RH850/U2A のみ)	37
4.6.3	割り込み設定の変更	38
5.	競合の管理	40
5.1	リソースの競合	40
5.2	端子競合の解消	41
6.	ソースの生成	43
6.1	生成ソースの CS+への登録	43
6.2	All Toolchain (CC-RH、GHS、IAR) で生成されたファイルの取り込み	44
6.3	生成ファイルの構成とファイル名	46
6.4	クロック設定	49
6.5	端子設定	50
6.6	割り込み設定	51
7.	ユーザープログラムの作成	52
7.1	コード生成の場合のカスタムコード追加方法	52
8.	生成ソースのバックアップ	54

9. レポートの生成	55
9.1 全設定内容レポート(PDF, txt 形式)	55
9.2 端子機能リスト、端子番号リスト設定内容(csv 形式).....	56
9.3 MCU/MPU パッケージ図(png 形式).....	56
10. ユーザーコード保護機能	57
10.1 ユーザーコード保護機能の指定タグ.....	57
10.2 ユーザーコード保護機能の使用例	57
10.3 競合発生時の対応方法	58
10.3.1 競合の発生条件	58
10.3.2 競合の解決方法	59
11. ヘルプ	61
11.1 ヘルプ.....	61
12. 参考ドキュメント.....	62
ホームページとサポート窓口	63

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CS+およびスマート・コンフィグレータを使用したプロジェクトの作成、基本的な使用方法とCS+のプロジェクトに追加するまでの手順について説明しています。

CS+の使い方は、CS+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. 使用前の準備

2.1 統合開発環境 CS+ (CS+ for CC)の準備

スマート・コンフィグレータで生成したソースを使用し、統合開発環境CS+上でプログラムの作成やビルドを行うには、CS+をインストールし、ターゲットデバイスをビルドできる状態にしてください。

2.2 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」および「CS+ RH850スマート・コンフィグレータ通信プラグイン」をダウンロードしてください。CS+ スマート・コンフィグレータ通信プラグインは、スマート・コンフィグレータで生成したソースをCS+に登録するために必要です。

<https://www.renesas.com/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。

【注】スマート・コンフィグレータでオールツールチェーン（CC-RH、GHS、IAR）で生成したソースコードは、CS+統合開発環境に追加してビルドすることもできますが、CS+ RH850スマート・コンフィグレータ通信プラグインのインストールやCS+の設定は必要ありません。統合開発環境。詳細な手順については、「6.2 All Toolchain (CC-RH、GHS、IAR) で生成されたファイルの取り込み」を参照してください。

2.3 統合開発環境 CS+の設定

スマート・コンフィグレータには、生成したソースファイルをCS+に登録し、それらをビルドするのに必要なCS+側の設定を自動で行う機能があります。スマート・コンフィグレータのインストール時に自動設定されますが、以下の方法で設定状態を確認し、必要に応じて設定を変更してください。

2.3.1 プラグイン設定状態の確認

CS+のメニューの [ツール] から [プラグインの管理] を選択し、「RH850用スマート・コンフィグレータ通信プラグイン」が有効(チェックが入っている)であることを確認してください。「RH850用スマート・コンフィグレータ通信プラグイン」が無効となっている場合は、有効に設定してください。

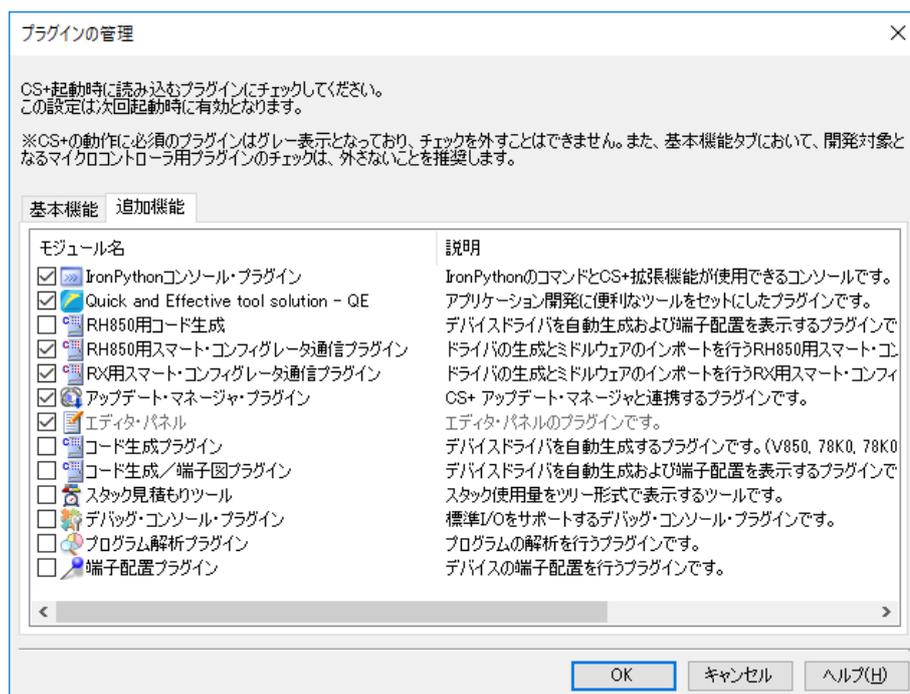


図 2-1 プラグインの管理

2.3.2 実行パス設定状態の確認

スマート・コンフィグレータ対象デバイスのCS+ プロジェクトを開くと、プロジェクト・ツリーに [Project name (プロジェクト)] → [スマート・コンフィグレータ(設計ツール)] が表示されます。 [スマート・コンフィグレータ(設計ツール)] をクリックすると、スマート・コンフィグレータのプロパティが表示されます。

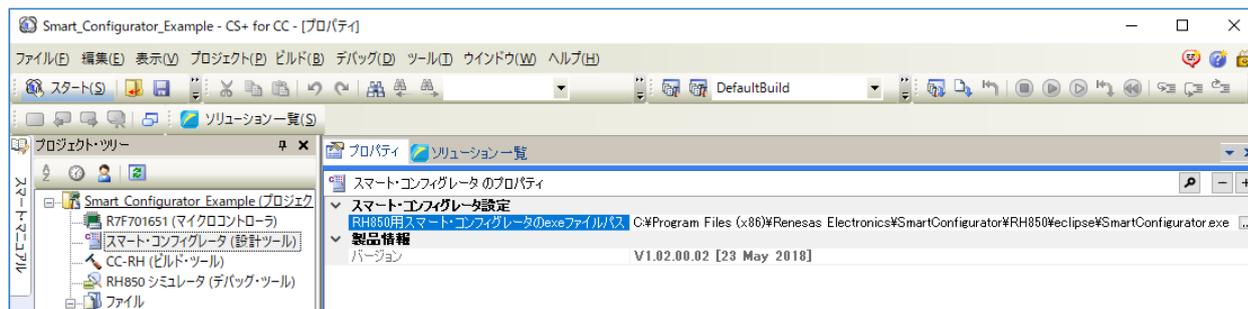


図 2-2 プロパティの表示

“スマート・コンフィグレータのexeファイルパス” がスマート・コンフィグレータの実行ファイルの設定です。初期状態のインストール先(“CS+” と “SmartConfigurator” が同階層)にスマート・コンフィグレータをインストールした場合は、以下のパスが設定されます。

“C:\Program Files (x86)\Renesas Electronics\SmartConfigurator\RH850\ eclipse\SmartConfigurator.exe”

手動で実行パスを設定する場合、“スマート・コンフィグレータのexeファイルパス”は相対パス、絶対パスのどちらでも設定できます。

2.4 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「アプリと機能」から、「Smart Configurator for RH850」と「CS+ SC Communication Plugins for RH850」を選択しアンインストールしてください。

2.5 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に、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CS+プロジェクトを作成してください。

“C:\Program Files (x86)\Renesas Electronics\SmartConfigurator\RH850\RH850C1M_SampleProjects”

“C:\Program Files (x86)\Renesas Electronics\SmartConfigurator\RH850\RH850F1KH_SampleProjects”

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

“C:\Program Files (x86)\Renesas Electronics\SmartConfigurator\RH850\RH850U2A_SampleProjects”

“C:\Program Files (x86)\Renesas Electronics\SmartConfigurator\RH850\RH850U2B_SampleProjects”

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周辺機能の設定し、CS+に登録してビルドするまでの手順を図 3-1 操作手順に示します。CS+の操作については、CS+の関連ドキュメントを参照してください。

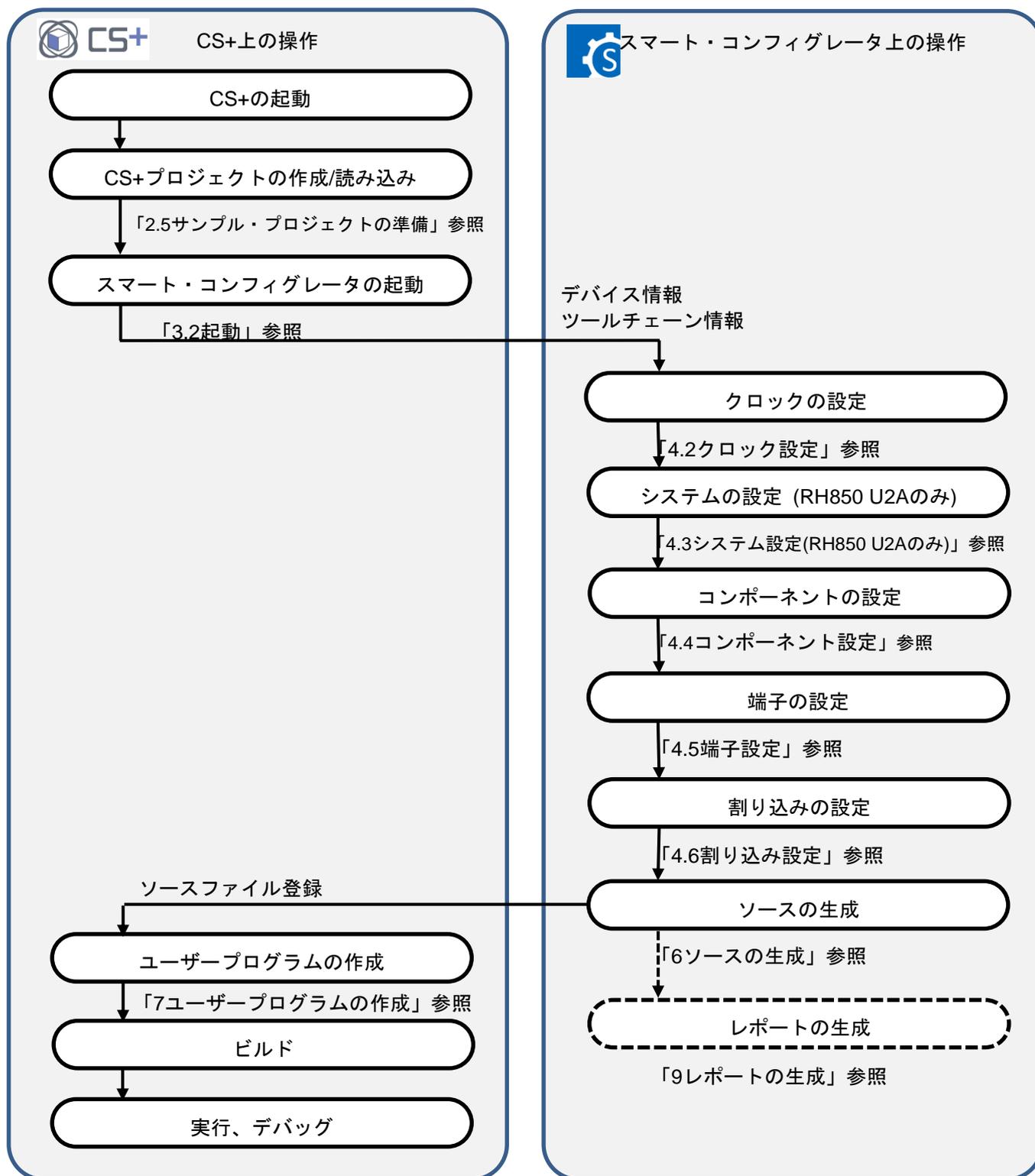


図 3-1 操作手順

3.2 起動

スマート・コンフィグレータの起動は、CS+のプロジェクト・ツリーの [Project name (プロジェクト)] → [スマート・コンフィグレータ(設計ツール)] をダブルクリックします。CS+のプロジェクト設定を引き継ぐため、スマート・コンフィグレータでは、デバイス選択やツールチェーンの選択は不要です。



図 3-2 スマート・コンフィグレータの起動

【注】 スマート・コンフィグレータの実行ファイルから起動した場合や、スマート・コンフィグレータのメニュー [ファイル] から新規プロジェクトを作成した場合や既存ファイルを開く場合は、CS+との連携はできません。

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。

CS+からスマート・コンフィグレータを起動する場合、スマート・コンフィグレータのプロジェクト・ファイルは、CS+のプロジェクト・ファイル(*.mtpj)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 3-3 メインウィンドウに示します。

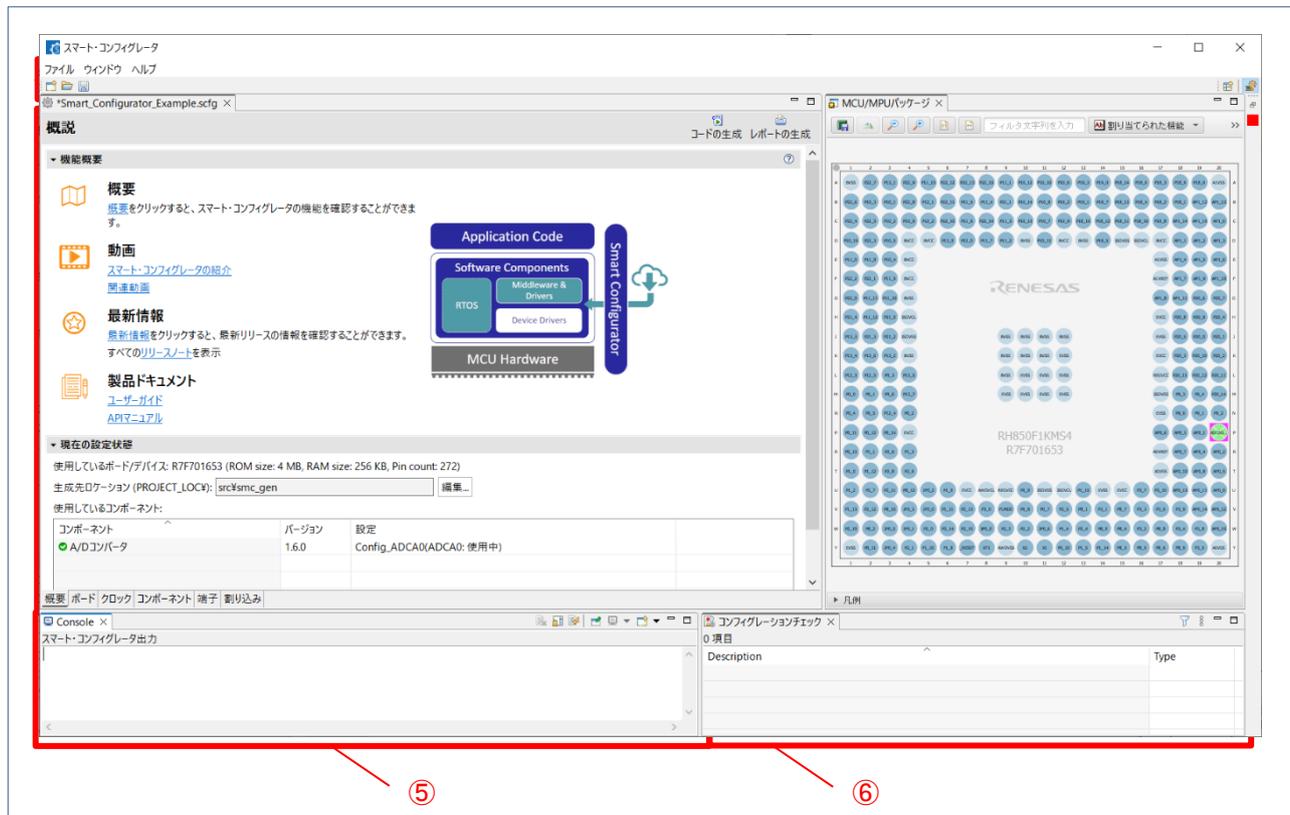


図 3-3 メインウィンドウ

- 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

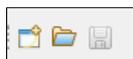
ファイル ウィンドウ ヘルプ

メインメニューの一覧を表 3-1 メニュー一覧に示します。

表 3-1 メニュー一覧

メニュー		内容
ファイル	新規	プロジェクト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	既存のプロジェクトを開くための [開く] ダイアログを表示します。
	保存	プロジェクト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。 CS+から引き継いだプロジェクト設定が消失してしまいますので、使用しないでください。
	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	プロジェクト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。
	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]、[システム]、[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。

【注】 [システム]ページはRH850/U2Aのみサポート

The screenshot displays the 'Smart Configurator' interface. At the top right, there are buttons for 'Code Generation' and 'Report Generation'. The main content is divided into two sections: 'Function Overview' and 'Current Configuration Status'.

機能概要 (Function Overview):

- 概要 (Summary):** Clicking here confirms the Smart Configurator's functionality.
- 動画 (Videos):** Includes an introduction and related videos for the Smart Configurator.
- 最新情報 (Latest Information):** Clicking here shows the latest release information and all release notes.
- 製品ドキュメント (Product Documents):** Links to the user guide and API manual.

現在の設定状態 (Current Configuration Status):

Using board/device: R7F701653 (ROM size: 4 MB, RAM size: 256 KB, Pin count: 272)
Generation location (PROJECT_LOC%):

Using components:

コンポーネント	バージョン	設定
● A/Dコンバータ	1.6.0	Config_ADCA0(ADCA0: 使用中)

At the bottom, there is a tabbed navigation bar with options: 概要 | ボード | クロック | コンポーネント | 端子 | 割り込み.

図 3-4 スマート・コンフィグレータビュー

3.4.4 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可能です。

パッケージビューは[割り当て機能]、[ボード機能]、[シンボル名]の3種類に切り替え可能です。

- ・ [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。
- ・ [シンボル名]には、シンボリック端子の設定情報が表示されます。

ボードの初期ピン設定情報は、[ボード]ページの[ボード:]で選択したボードのピン情報です（「4.1ボード設定」「4.5.6ボードの端子情報を使用した端子設定」参照）。

【注】 RH850/F1KM用SCおよびRH850/F1KH用SCIにはシンボリックネーム機能は適用されません。

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

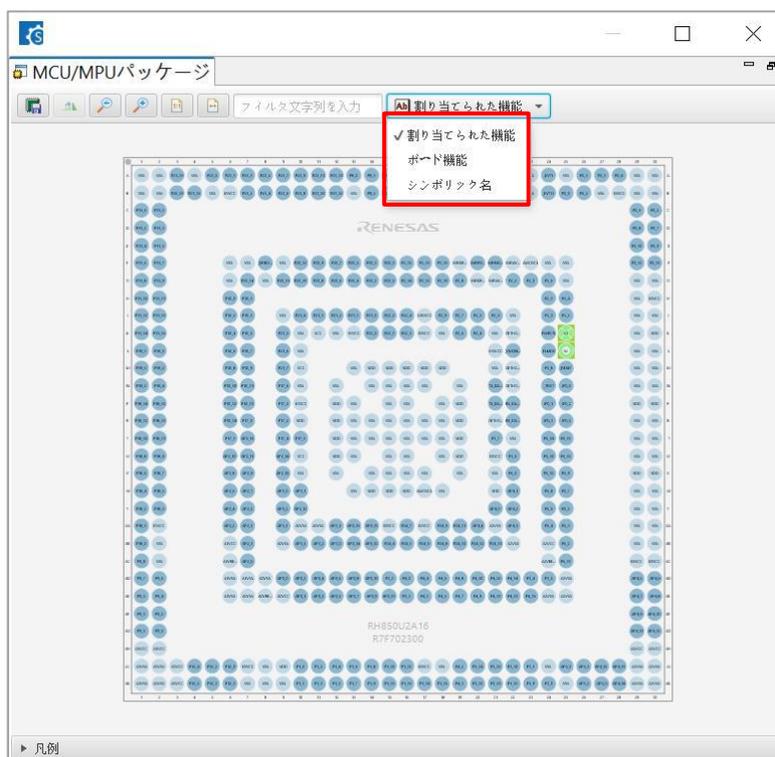


図 3-5 MCU/MPUパッケージビュー

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/MPUパッケージビューでの設定変更内容が表示されます。

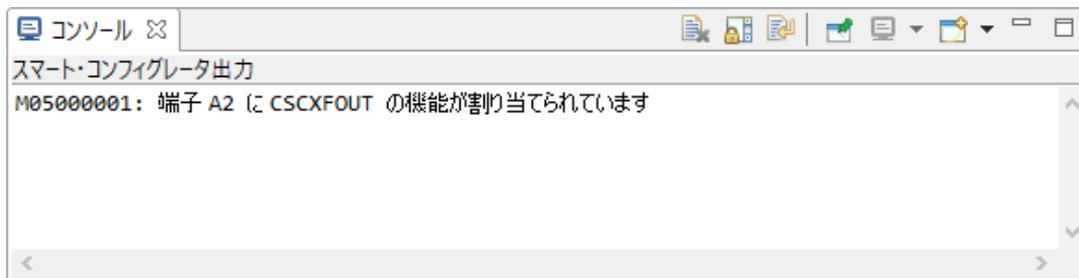


図 3-6 コンソールビュー

3.4.6 コンフィグレーションチェックビュー

端子競合が発生した場合に、その内容を表示します。



図 3-7 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。CS+からスマート・コンフィグレータを起動する場合、設定しないでください。

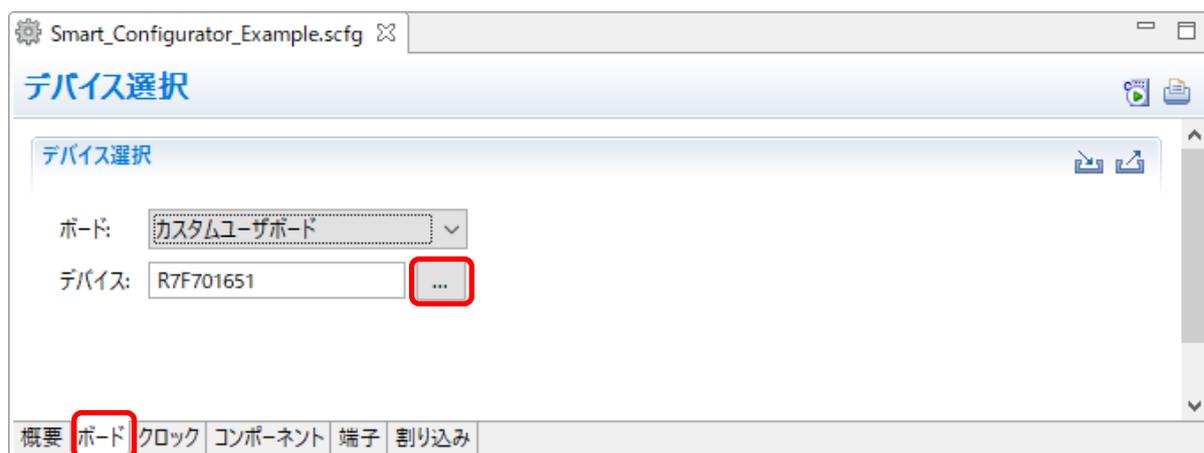


図 4-1 デバイス選択

4.1.2 ボード選択

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
- 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

ボード設定は.bdf ファイル(Board Description File)に定義されています。

ルネサス製のボード(スターターキットなど)の.bdfファイルは、ルネサスのWEBページから入手可能です。

また、アライアンスパートナーが公開している.bdf ファイルをインポートすれば、アライアンスパートナー製ボードの選択が可能になります。

[デバイス]に表示されているデバイス名が.bdfファイルと同じ場合、「ボード変更の確認」ダイアログが表示されます。

[デバイス]に表示されているデバイス名が.bdfファイルと異なる場合、「デバイス変更の確認」ダイアログが表示されます。

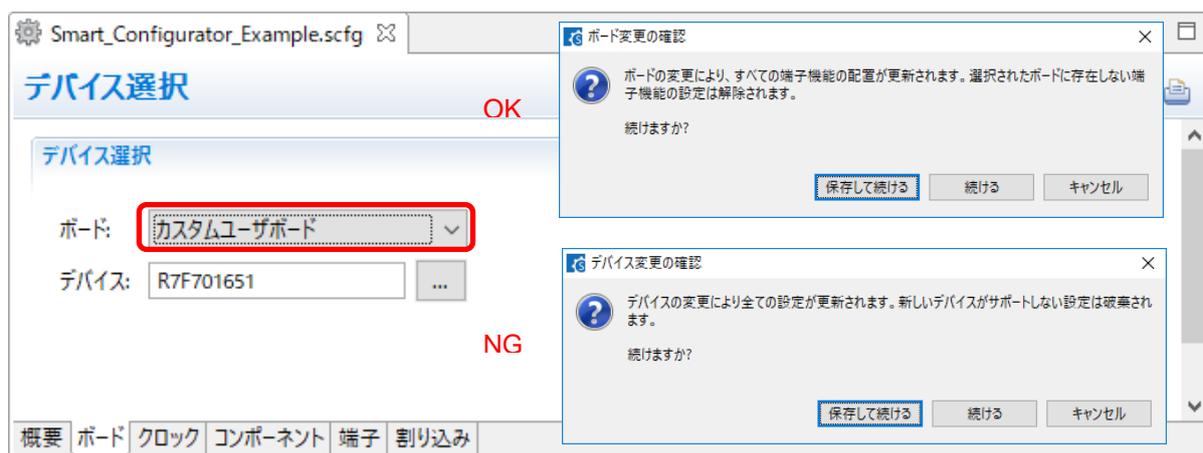


図 4-2 ボード選択

【注】 選択したボードに応じてデバイスが変わりますが、デバイスの変更はCS+プロジェクトのデバイス(マイクロコントローラ)には反映されません。

4.1.3 ボード設定のエクスポート

ボード設定をエクスポートして、参照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。

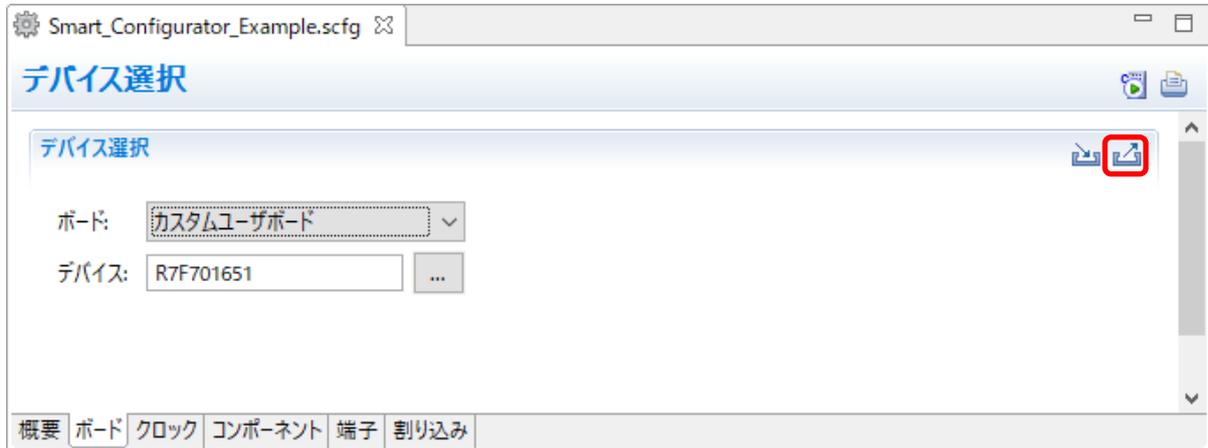


図 4-3 ボード設定のエクスポート (bdf形式)

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) ボード選択の選択肢が追加されます。

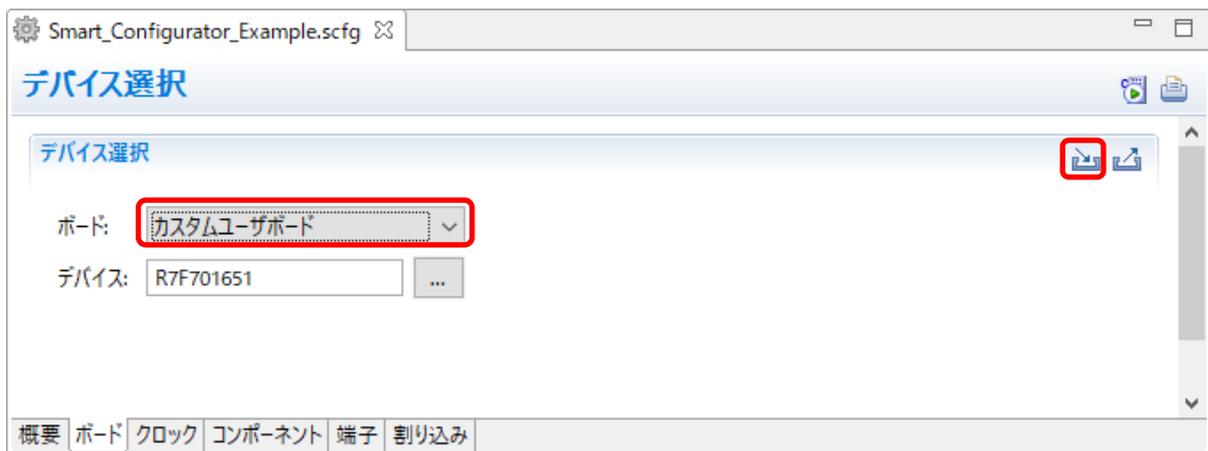


図 4-4 ボード設定のインポート (bdf形式)

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) ボード上で動作するために必要なクロックを選択します。(内部クロックの周波数は固定されているものもあります)
- (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) マルチプレクサの図形では、出力クロックのクロックソースを選択します。
- (4) 指定したクロックを有効にします (RH850/F1KM と RH850/F1KH のみ必要)
- (5) 期待した出力クロック周波数を得るために、ドロップダウンリストで分周比を設定します。

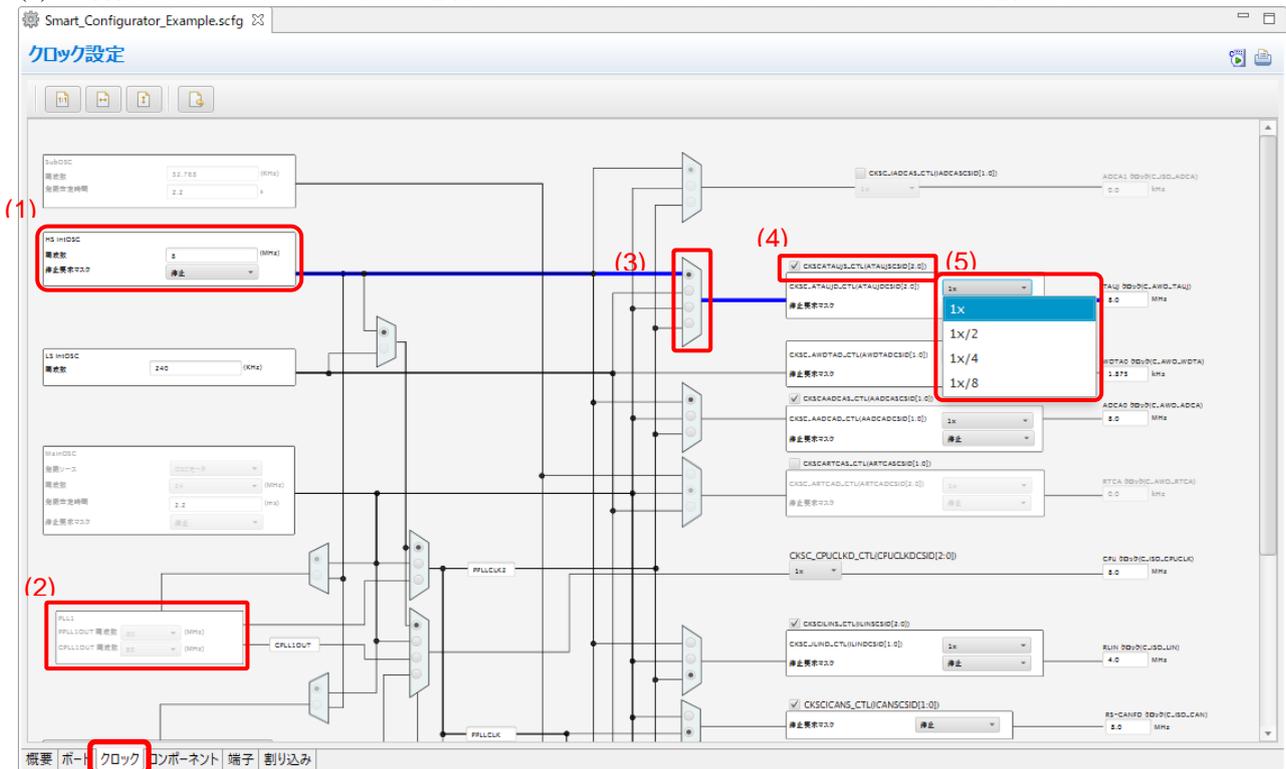


図 4-5 クロック設定

4.3 システム設定(RH850 U2A のみ)

[システム]タブで使用するCPU_n(PE_n)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 4-6 システムページ

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。

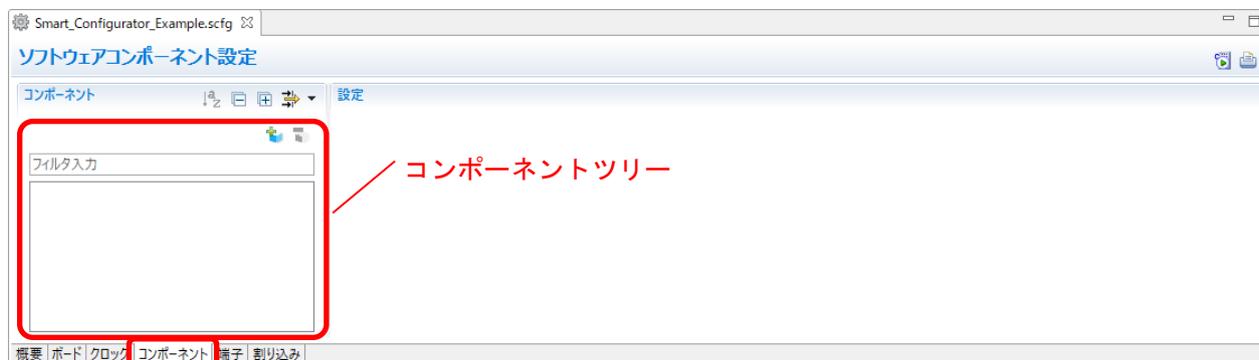


図 4-7 コンポーネントページ

4.4.1 コード生成コンポーネントの追加方法

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) [コンポーネントの追加] アイコンをクリックします。



図 4-8 コンポーネントの追加

- (2) [ソフトウェアコンポーネントの選択] ダイアログボックスの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。

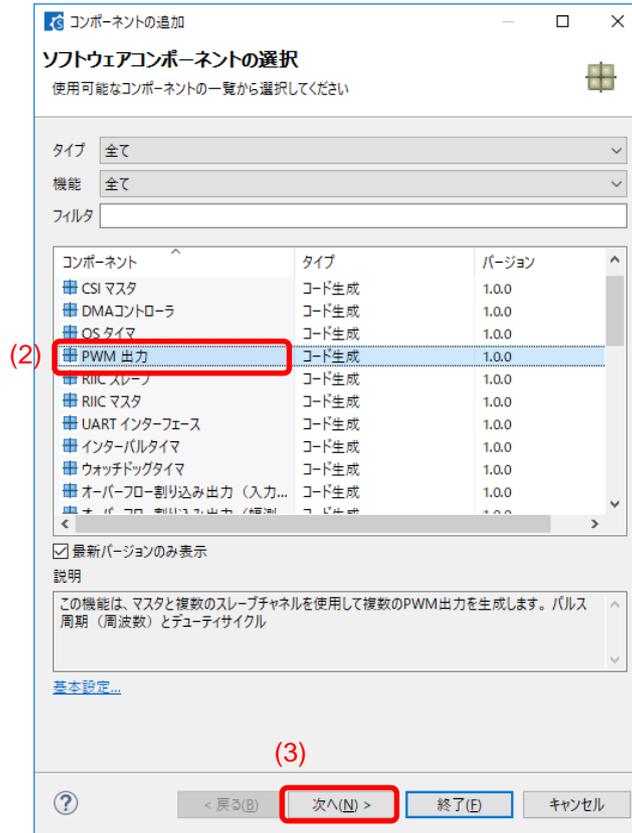


図 4-9 コード生成コンポーネントの追加

- (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] ダイアログボックスで、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_TAU0)
- (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: TAU0)
- (6) [終了] をクリックします。

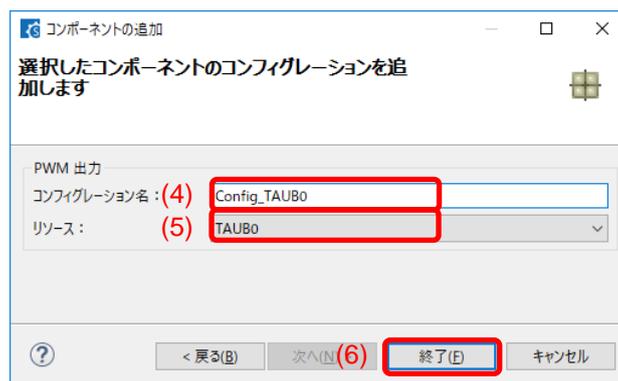


図 4-10 コンポーネントの追加

4.4.2 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の追加をスマート・コンフィグレータがサポートできるようになります。この機能を使用するには、ユーザーはコンポーネント一覧から、ハードウェア一覧表示へコンポーネントツリーを変更する必要があります。

- (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は、ハードウェアリソース階層の中にコンポーネントを表示します。

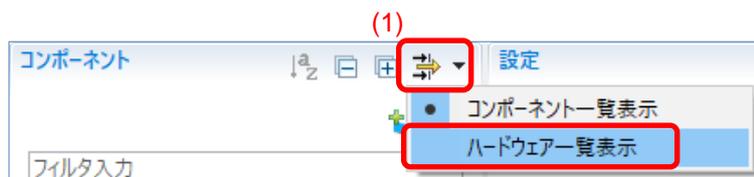


図 4-11 ハードウェア一覧への切り替え

- (2) ハードウェアリソースノード(例：タイマ・アレイ・ユニット B1→ TAUB10)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。
- (3) リストからコンポーネントを選択し(例：PWM 出力)、4.4.1 章に記載されている手順で新しい構成を開きます。

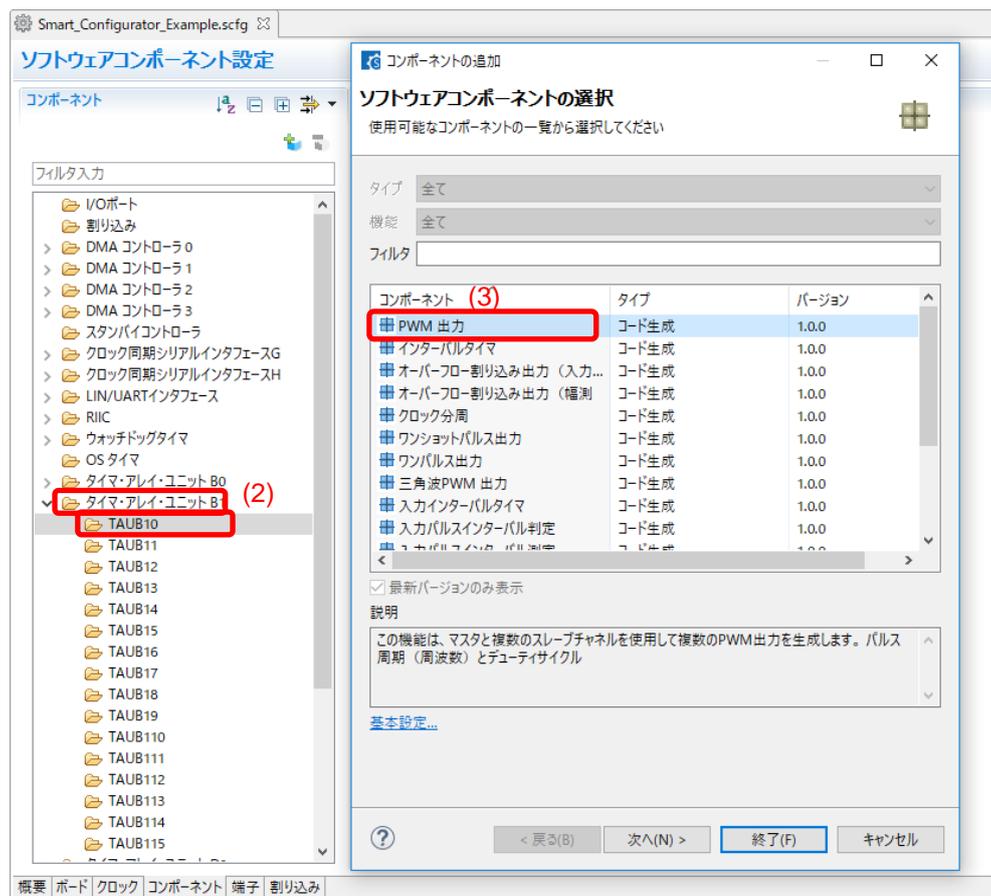


図 4-12 ハードウェア一覧へのコンポーネントの追加

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。(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。

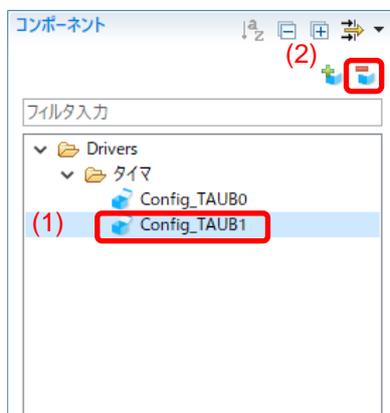


図 4-13 ソフトウェアコンポーネントの削除

コンポーネントツリーから、選択したソフトウェアが削除されます。

再コード生成を行うと、CS+のプロジェクト・ツリーにあるソースファイルも削除されます。

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コンフィグレーションをクリックし、選択します。(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. チャンネル1スレブ、チャンネル2スレブ、チャンネル3スレブを選択します。
 - c. [マスター0]タブ上の[パルス周期]を設定します。
 - d. [スレブ1]、[スレブ2]、[スレブ3]タブ上それぞれの[デューティ]を設定します。

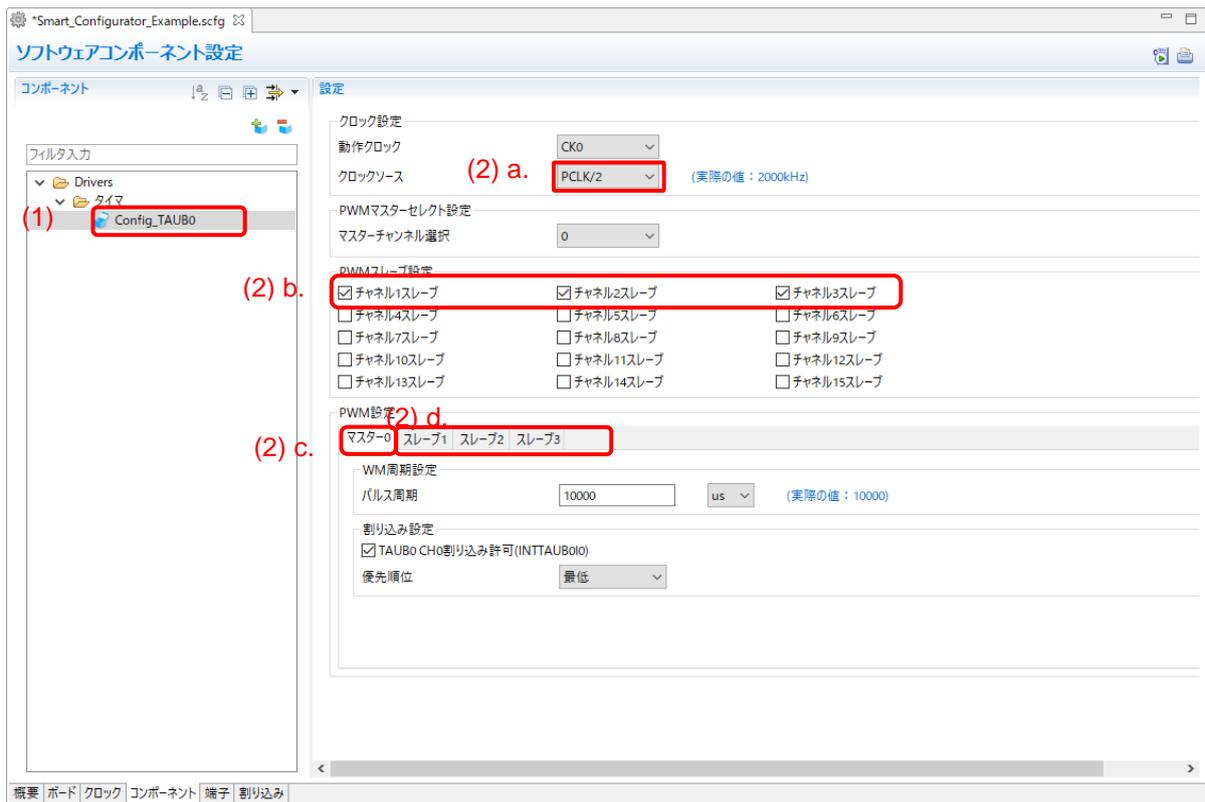


図 4-14 コンフィグレーション設定

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

スマート・コンフィグレータでは、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。

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) コンフィグレーション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。

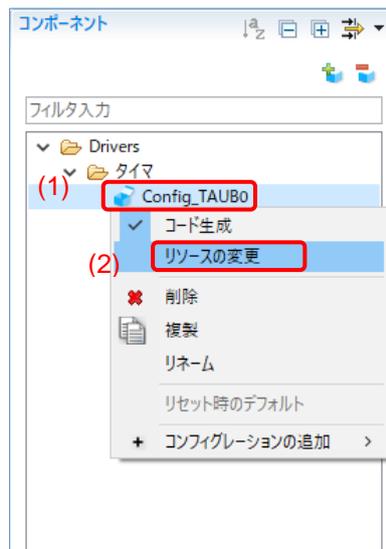


図 4-15 リソースの変更

- (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：TAUB1）。
- (4) 「次へ」ボタンが有効になるので、クリックします。

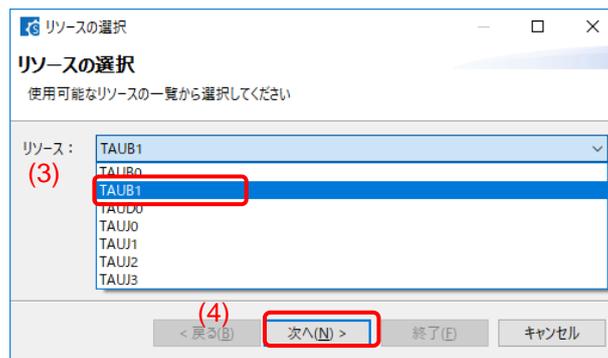


図 4-16 コンポーネントページ-新しいリソースの選択

- (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を選択します。
- (8) 「終了」をクリックします。

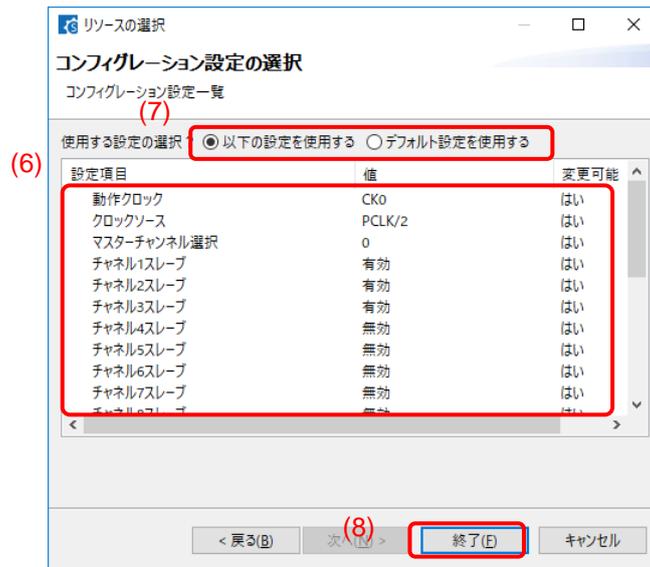


図 4-17 新しいリソース設定の確認

リソースは、自動的に更新されます。(例：INTTAUB010 から INTTAUB110)

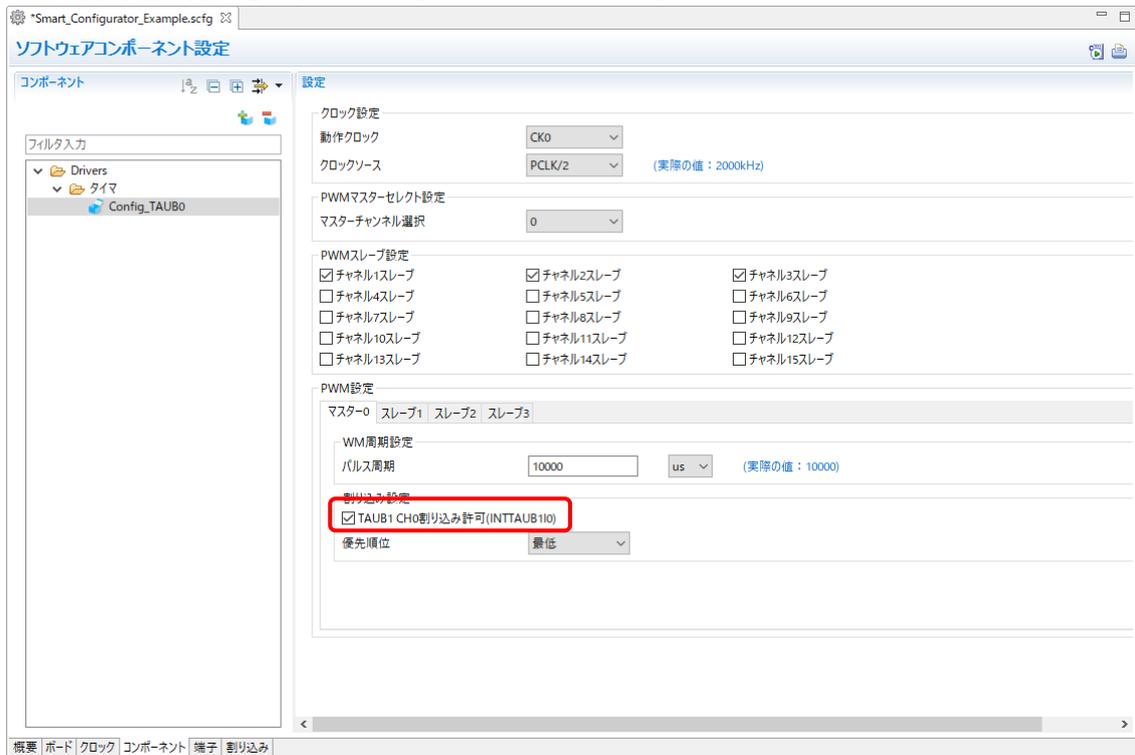


図 4-18 自動的に更新されるリソース

(9) コンフィグレーション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に再度名前をつけます (例：Config_TAUB0 を Config_TAUB1)。

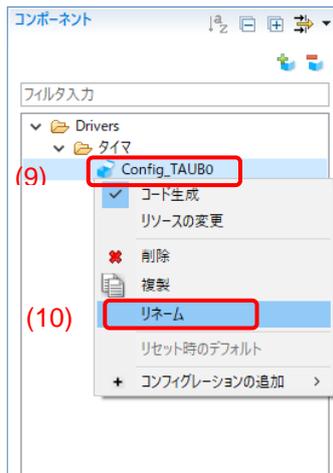


図 4-19 コンフィグレーションに再度名前をつける

4.4.6 コンポーネントの一般設定

バックアップ設定やAPI機能の出力設定など、コンポーネントの全般的な設定を変更することができます。設定の変更は、[設定] ダイアログの[コンポーネント] ページで行います。

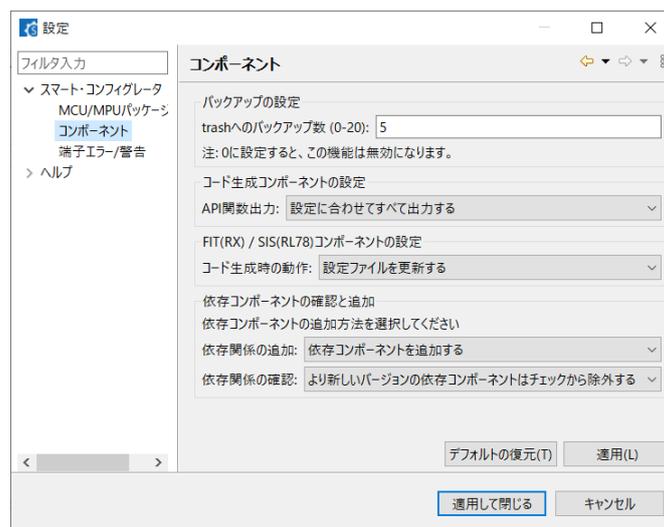


図 4-20 コンポーネントの一般設定

【注】

1. 生成されたコードをバックアップするtrashに作成するバックアップフォルダの最大数を変更したい場合は、[trashへのバックアップ数 (0-20)] の数を変更します。上限を超えると、古いフォルダから削除されます。
2. 初期化API関数のみを生成したい場合は、[API関数出力]を[初期化関数のみを出力する]に設定します。この場合、

```
void R_{ConfigurationName}_Create (void), void R_{ConfigurationName}_Create_UserInit (void)
```

 関数のみが生成されます。
3. [FIT(RX)/SIS(RL78)コンポーネントの設定]と[依存コンポーネントの確認と追加]は、Smart Configurator for RH850ではサポートしていません。

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。

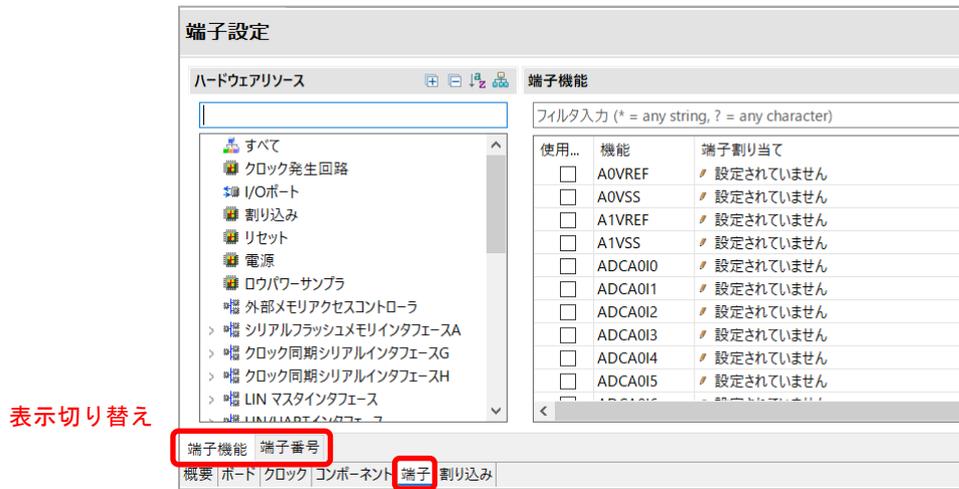


図 4-21 端子ページ (端子機能)

[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。

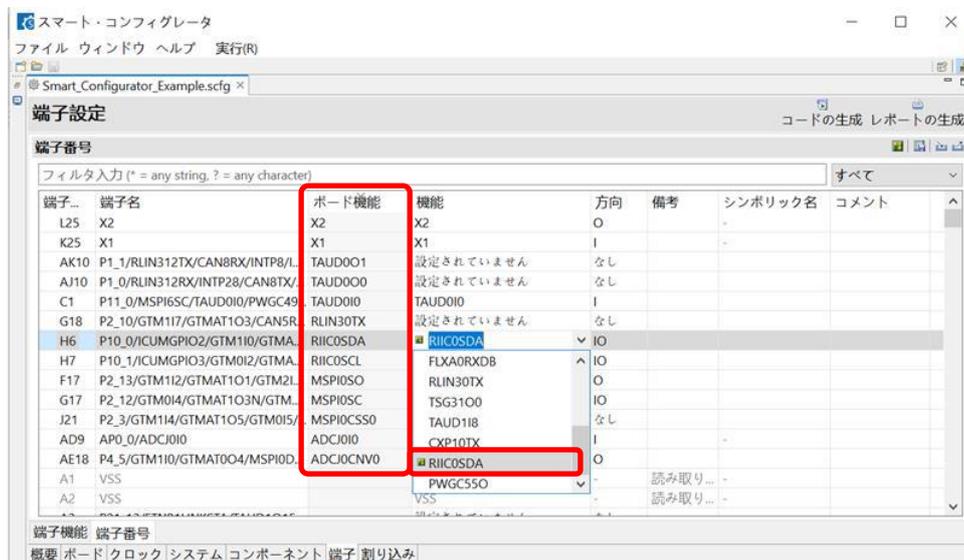


図 4-22 端子ページ (端子番号)

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) 同じ周辺チャンネルに属する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。

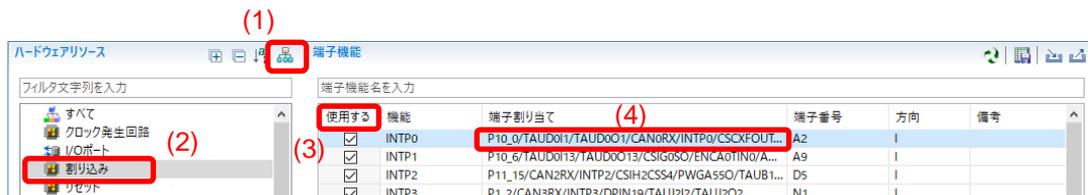


図 4-23 端子設定- [端子機能] リストの端子配置設定

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 パッケージを使用した端子の設定

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。

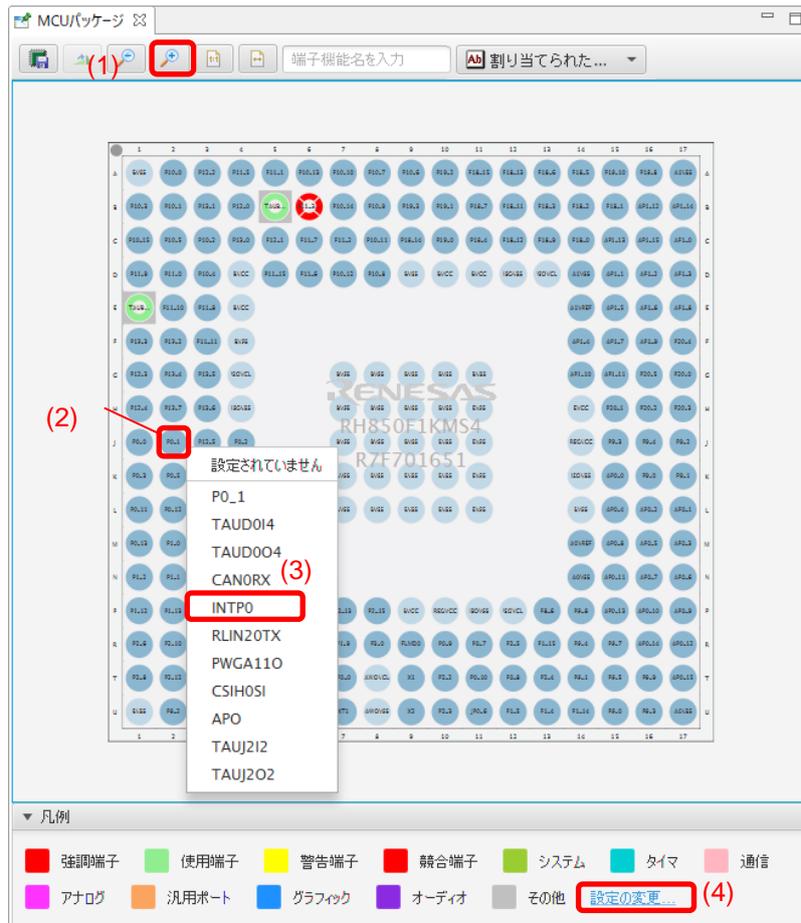


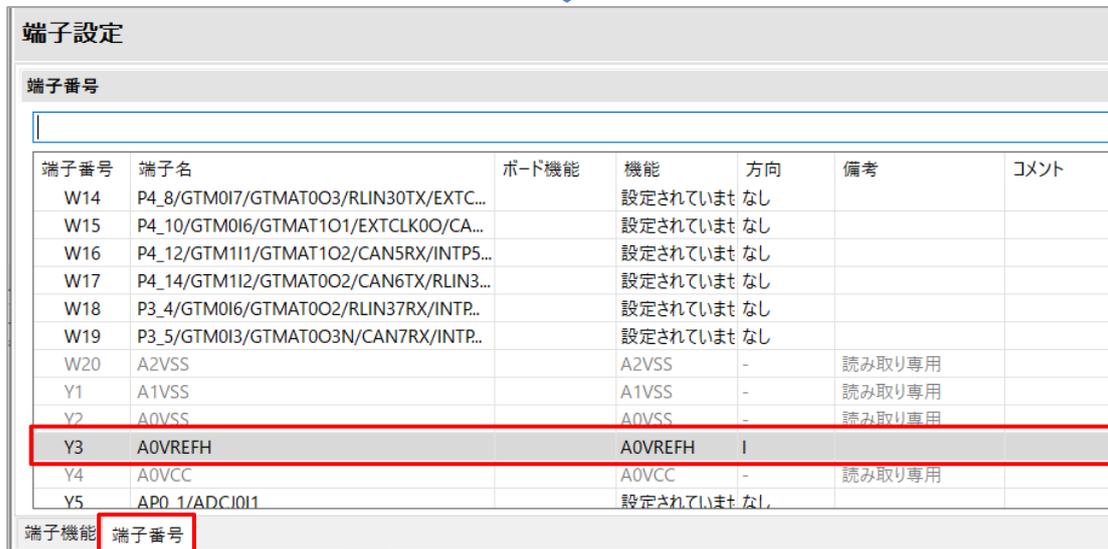
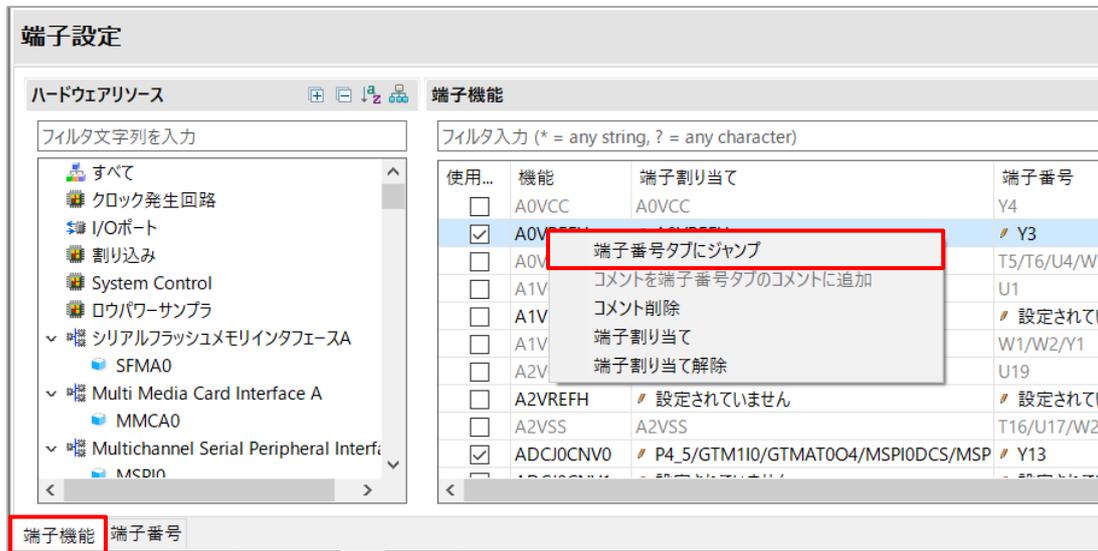
図 4-24 MCU パッケージを使用した端子設定

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。

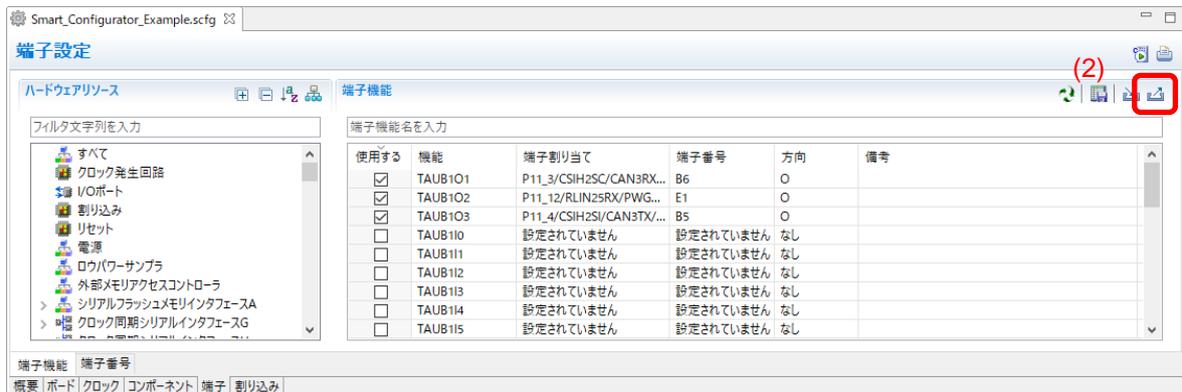


図 4-25 端子設定を XML ファイルへエクスポートする

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。

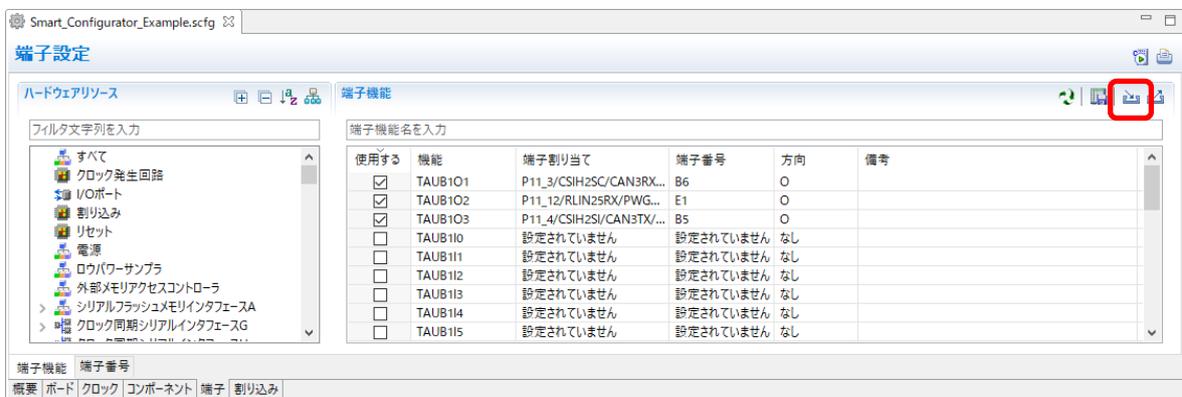


図 4-26 端子設定を XML ファイルからインポートする

【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 ボードの端子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。

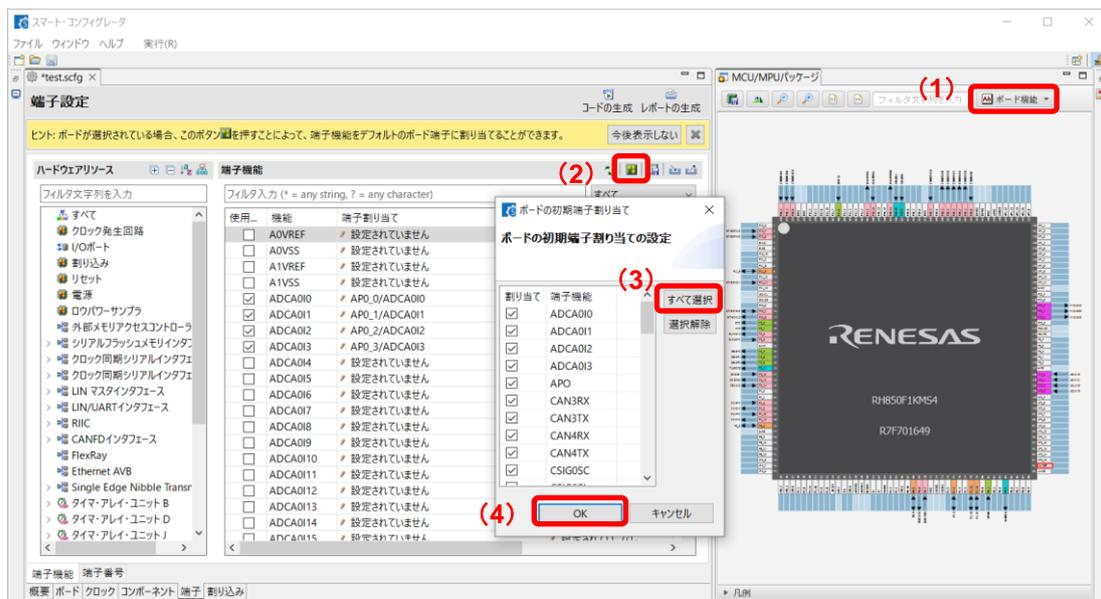


図 4-27 端子の初期設定

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 4-28 端子設定のフィルタリング

4.5.8 端子エラー/警告の設定

[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。

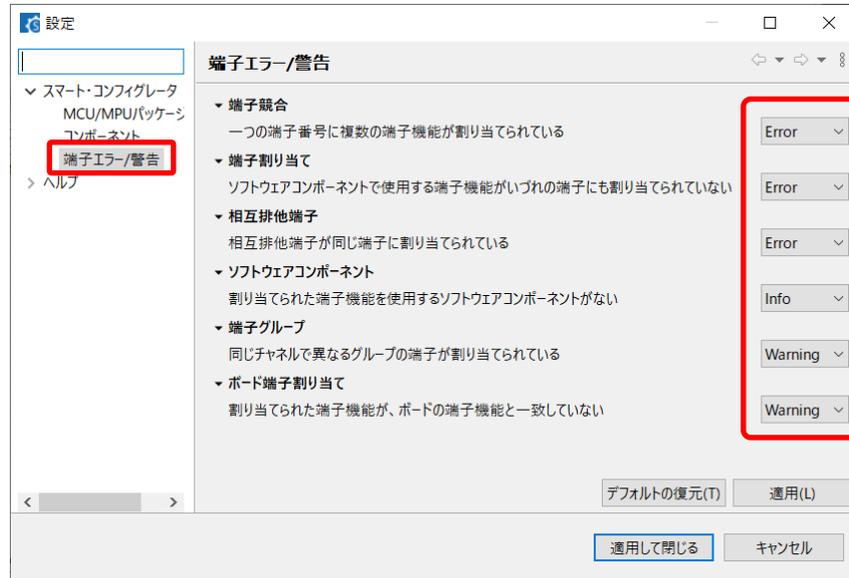


図 4-29 端子エラー/警告の設定

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 4-30 のようになります。

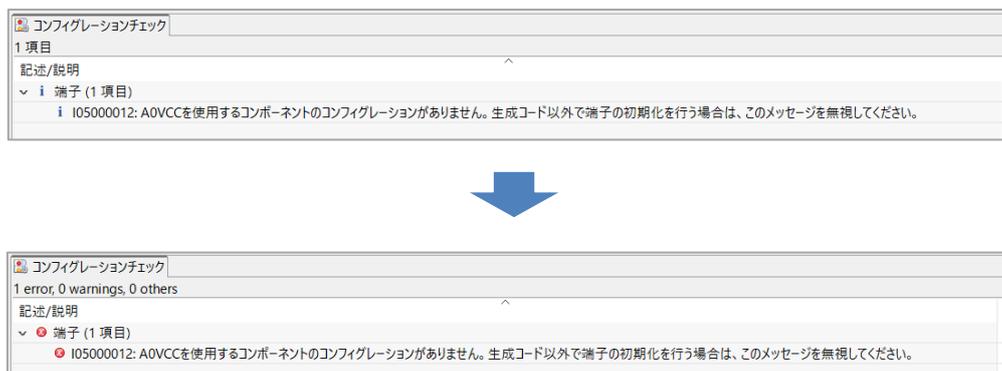


図 4-30 端子エラー/警告の設定の変更例

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。ペ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。。

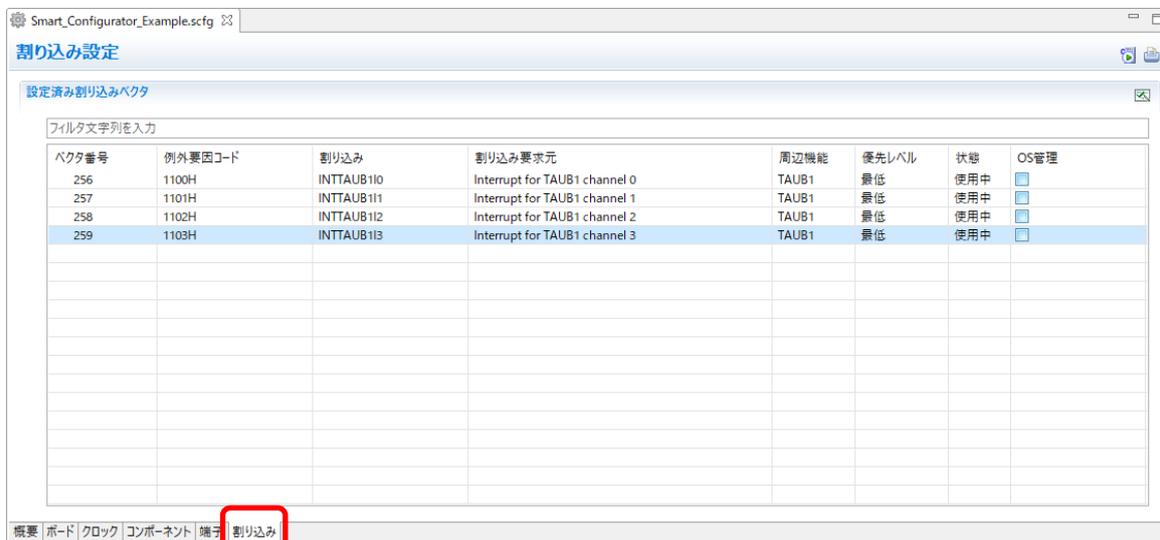


図 4-31 割り込みページ

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 (R1850V4) 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。

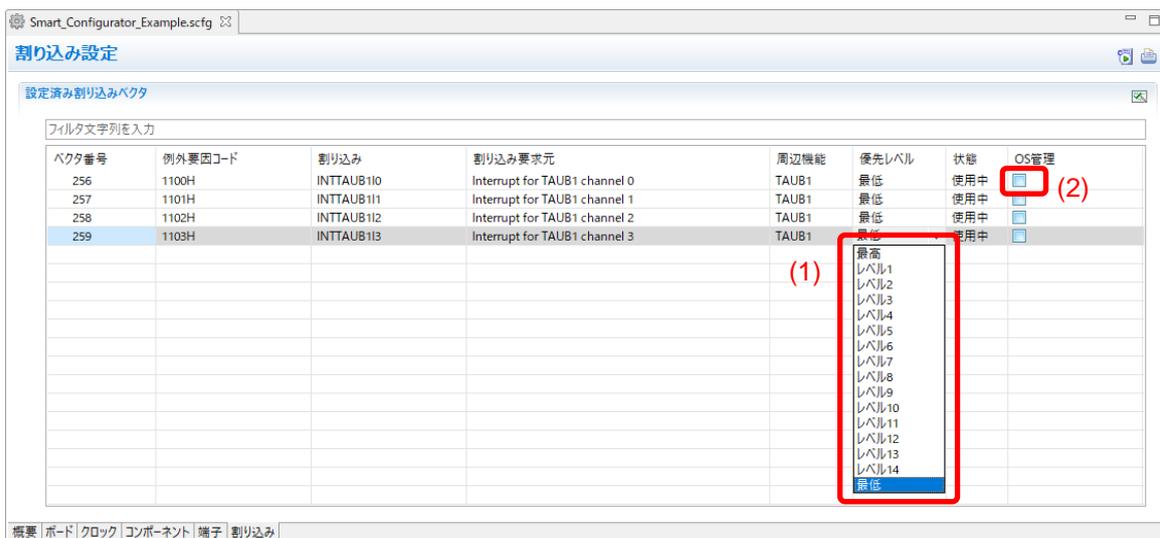


図 4-32 割り込み設定

4.6.2 PEn 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 PEn を選択することができます。 PEn 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 4-33 [システム]ページのPE選択

- (2) [割り込み] ページの PEn 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、PEn 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、PEn 列の1だけが応答可能。



図 4-34 [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 4-36割り込み有効/無効関数のコード例」を参照してください。

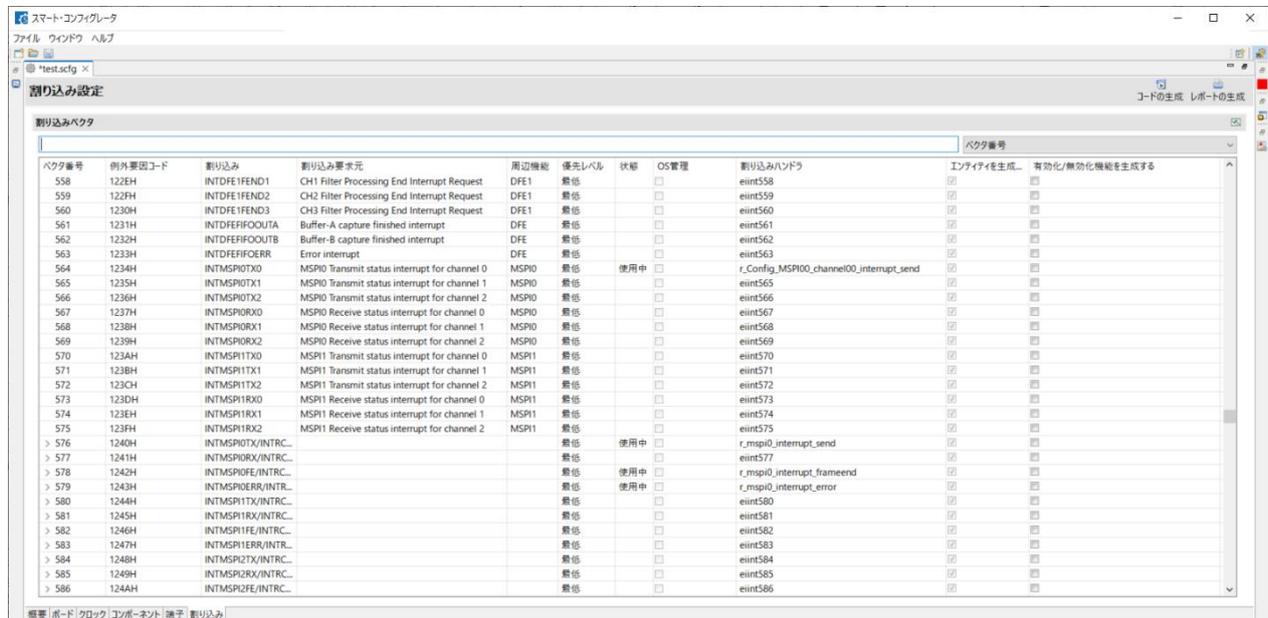


図 4-35 割り込みハンドラとエンティティの設定

```
r_smc_interrupt.c
19
20
21  * File Name      : r_smc_interrupt.c
22  * Version       : 1.3.0
23  * Device(s)    : R7F702301BEBBA
24  * Description   : None
25
26  /* Start user code for pragma. Do not edit comment generated here */
29  /* End user code. Do not edit comment generated here */
30
31
32  #include "r_cg_macrodriver.h"
35  #include "r_cg_userdefine.h"
36  #include "r_smc_interrupt.h"
37
38
39
42  void R_Interrupt_Create(void)
43  {
44  }
45
46
47  void r_Config_MSPI00_channel100_interrupt_send_enable_interrupt(void)
48  {
49      /* Clear INTMSPI0TX0 request and enable operation */
50      INTC2.EIC244.BIT.EIRF244 = _INT_REQUEST_NOT_OCCUR;
51      INTC2.EIC244.BIT.EIMK244 = _INT_PROCESSING_ENABLED;
52  }
53
54  void r_Config_MSPI00_channel100_interrupt_send_disable_interrupt(void)
55  {
56      /* Disable INTMSPI0TX0 operation and clear request */
57      INTC2.EIC244.BIT.EIMK244 = _INT_PROCESSING_DISABLED;
58      INTC2.EIC244.BIT.EIRF244 = _INT_REQUEST_NOT_OCCUR;
59  }
60
61  void r_Config_MSPI00_channel100_interrupt_receive_enable_interrupt(void)
62  {
63      /* Clear INTMSPI0RX0 request and enable operation */
64      INTC2.EIC245.BIT.EIRF245 = _INT_REQUEST_NOT_OCCUR;
65      INTC2.EIC245.BIT.EIMK245 = _INT_PROCESSING_ENABLED;
66  }
67
68  void r_Config_MSPI00_channel100_interrupt_receive_disable_interrupt(void)
69  {
70      /* Disable INTMSPI0RX0 operation and clear request */
71      INTC2.EIC245.BIT.EIMK245 = _INT_PROCESSING_DISABLED;
72      INTC2.EIC245.BIT.EIRF245 = _INT_REQUEST_NOT_OCCUR;
73  }
74
```

図 4-36 割り込み有効/無効関数のコード例

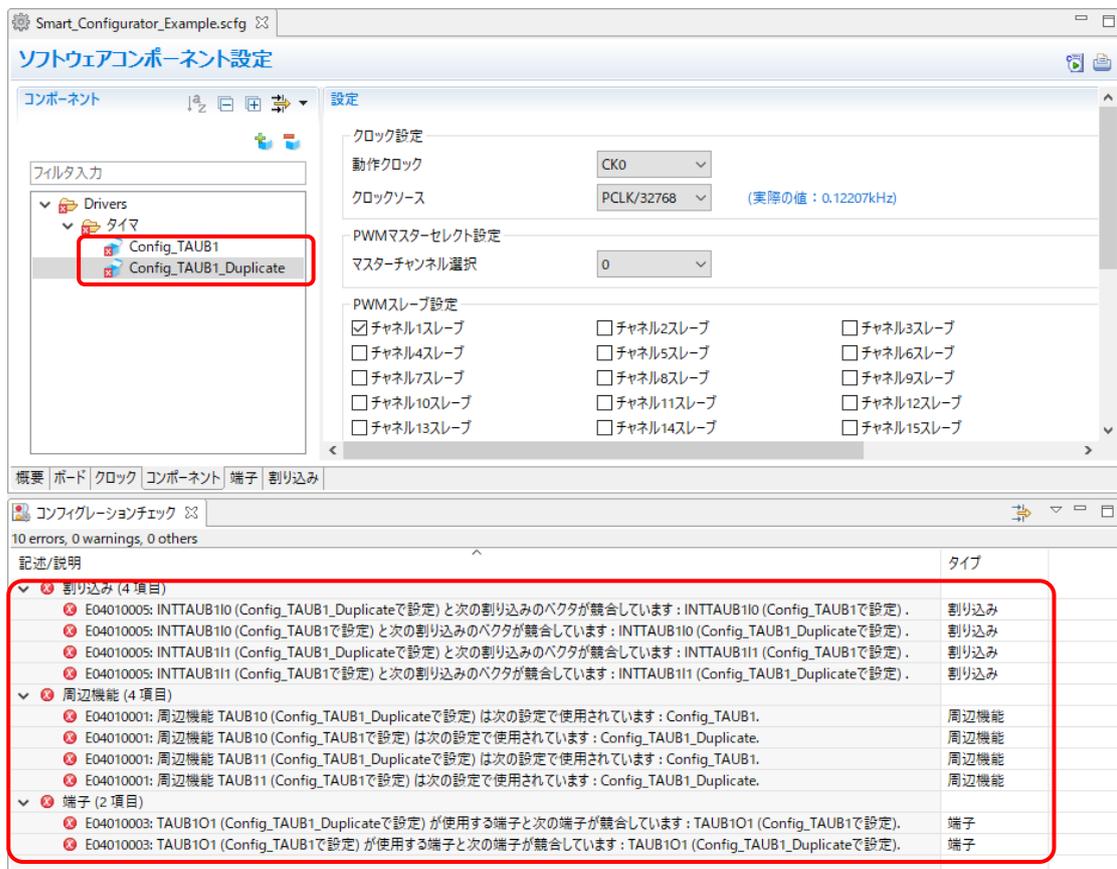
5. 競合の管理

コンポーネントの追加、端子や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

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。



The screenshot shows the Smart Configurator interface. In the component tree, 'Config_TAUB1' and 'Config_TAUB1_Duplicate' are highlighted with a red box. Below, the configuration check results are displayed in a table:

記述/説明	タイプ
<ul style="list-style-type: none"> ✖ 割り込み (4項目) <ul style="list-style-type: none"> ✖ E04010005: INTTAUB1I0 (Config_TAUB1_Duplicateで設定) と次の割り込みのバグが競合しています: INTTAUB1I0 (Config_TAUB1で設定)。 ✖ E04010005: INTTAUB1I0 (Config_TAUB1で設定) と次の割り込みのバグが競合しています: INTTAUB1I0 (Config_TAUB1_Duplicateで設定)。 ✖ E04010005: INTTAUB1I1 (Config_TAUB1_Duplicateで設定) と次の割り込みのバグが競合しています: INTTAUB1I1 (Config_TAUB1で設定)。 ✖ E04010005: INTTAUB1I1 (Config_TAUB1で設定) と次の割り込みのバグが競合しています: INTTAUB1I1 (Config_TAUB1_Duplicateで設定)。 ✖ 周辺機能 (4項目) <ul style="list-style-type: none"> ✖ E04010001: 周辺機能 TAUB10 (Config_TAUB1_Duplicateで設定) は次の設定で使用されています: Config_TAUB1。 ✖ E04010001: 周辺機能 TAUB10 (Config_TAUB1で設定) は次の設定で使用されています: Config_TAUB1_Duplicate。 ✖ E04010001: 周辺機能 TAUB11 (Config_TAUB1_Duplicateで設定) は次の設定で使用されています: Config_TAUB1。 ✖ E04010001: 周辺機能 TAUB11 (Config_TAUB1で設定) は次の設定で使用されています: Config_TAUB1_Duplicate。 ✖ 端子 (2項目) <ul style="list-style-type: none"> ✖ E04010003: TAUB1O1 (Config_TAUB1_Duplicateで設定) が使用する端子と次の端子が競合しています: TAUB1O1 (Config_TAUB1で設定)。 ✖ E04010003: TAUB1O1 (Config_TAUB1で設定) が使用する端子と次の端子が競合しています: TAUB1O1 (Config_TAUB1_Duplicateで設定)。 	

図 5-1 リソースの競合

5.2 端子競合の解消

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。

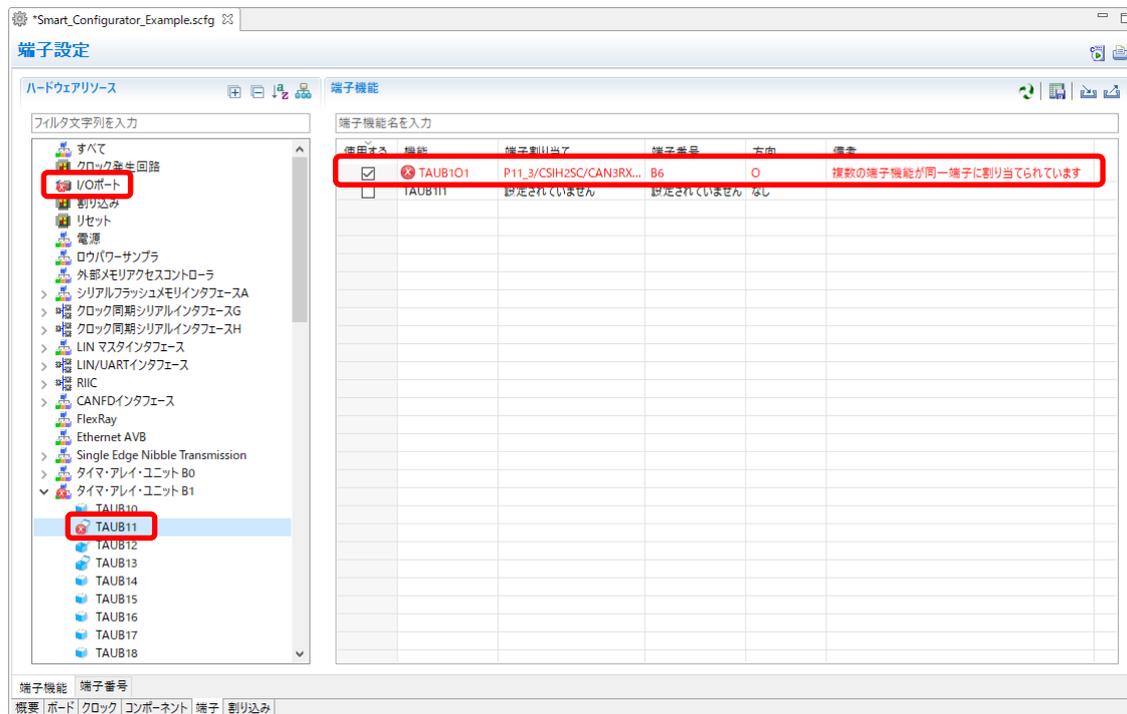


図 5-2 端子の競合

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。

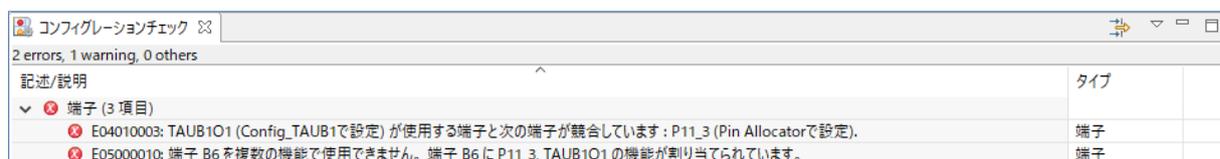


図 5-3 端子競合のメッセージ

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。

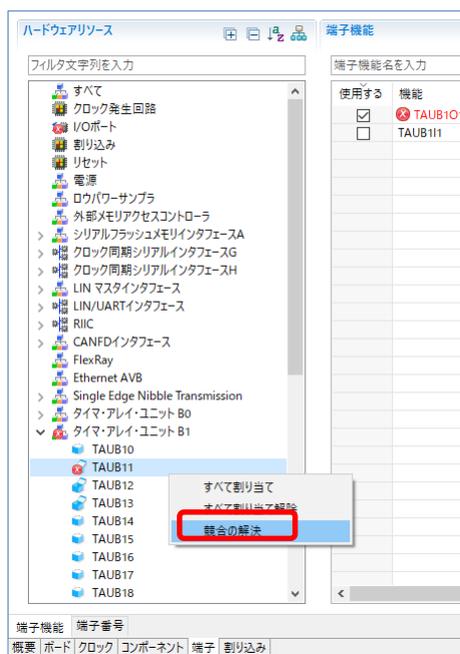


図 5-4 端子競合の解決

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CS+への登録

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、<ProjectDir>%src%smc_genにファイルを生成し、CS+のプロジェクトにファイルを登録します。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。）

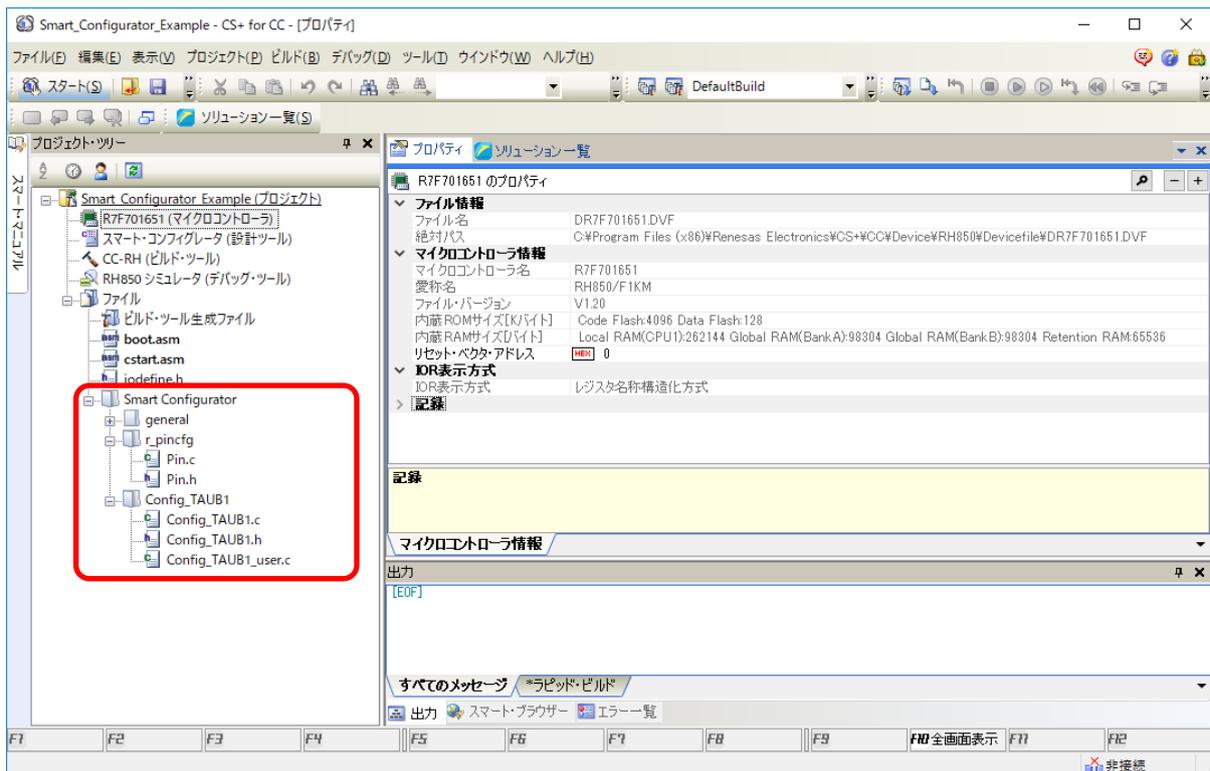


図 6-2 CS+プロジェクトへのソースファイル登録

6.2 All Toolchain (CC-RH、GHS、IAR) で生成されたファイルの取り込み

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3つのツールチェーンすべてに対応したソースファイルを出力します。これらの生成されたファイルをCS+プロジェクトに取り込む場合は、以下の手順に従ってください。

- (1) CS+プロジェクトのプロジェクトウィザードに従って新規プロジェクトを作成します。
- (2) CS+で作成されたデフォルトファイル「main0.c」「main.c」をCS+プロジェクトから削除します。

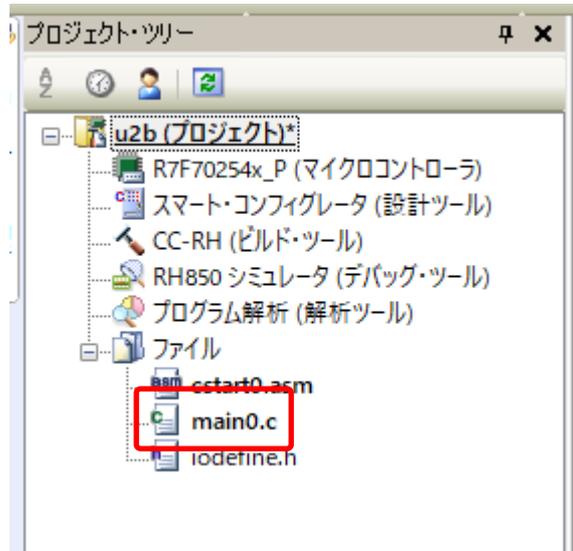


図 6-3 main0.cの削除

【注】 エントリー関数として、スマート・コンフィグレータが生成した「r_cg_main.c」を使用します。

- (3) 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手動でCS+のプロジェクトに追加します。
CS+で作成された「iodefine.h」をCS+のプロジェクトに追加します。

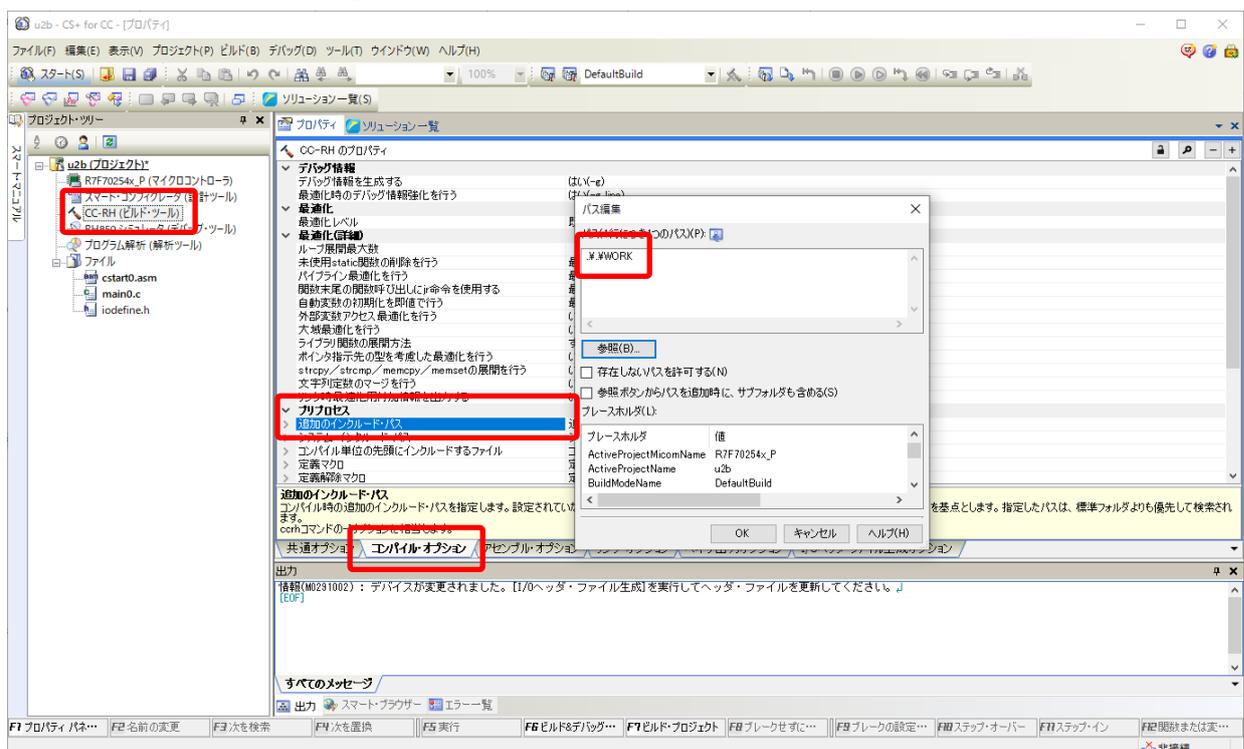


図 6-4 「iodefine.h」の追加

- (4) スマート・コンフィグレータが出力したソースファイルを、下図のように手動でプロジェクトに追加します。

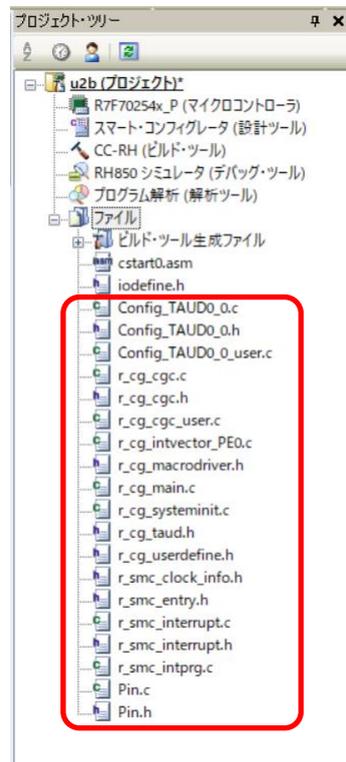


図 6-5 CS+のプロジェクトにファイルを追加

【注】CS+スマート・コンフィグレータ通信プラグインは「All Toolchain (CC-RH、GHS、IAR)」をサポートしていないため、スマート・コンフィグレータでの構成変更後にソースファイルの追加/削除をユーザーが行う必要があります。

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図 6-6 生成ファイルの構成とファイル名に示します。なお、*main()*関数はCS+でプロジェクト作成時に生成するmain.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。

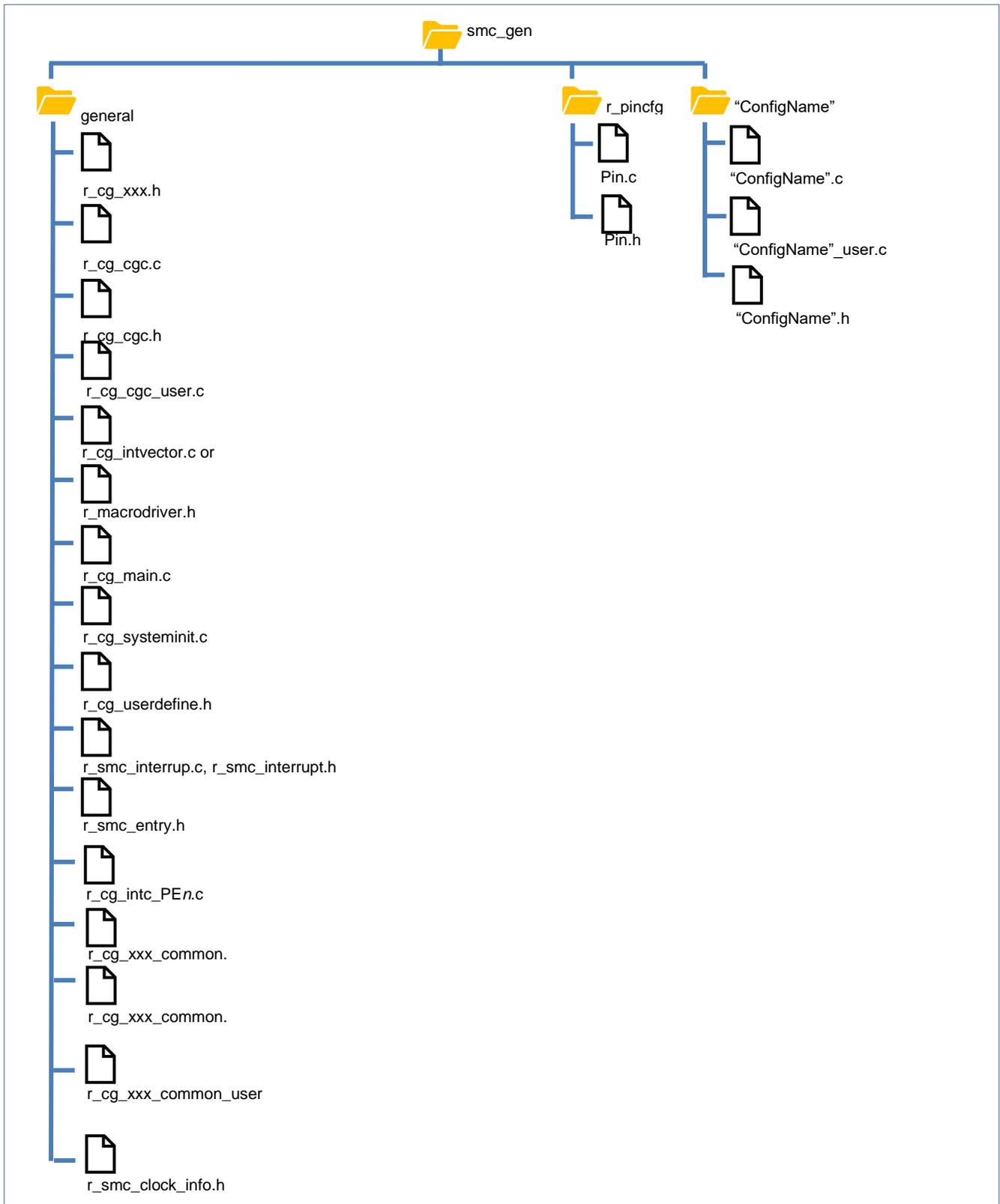


図 6-6 生成ファイルの構成とファイル名

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。
	<i>r_cg_XXX.h^(*)</i>	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。
	<i>r_cg_cgc.c</i>	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。
	<i>r_cs_cgc.h</i>	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。
	<i>r_cs_cgc_user.c</i>	このファイルは、CGC 初期化後にユーザーがコードを <i>R_CGC_Create</i> 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。
	<i>r_cg_intvector.c</i> <i>r_cg_intvector_PEn.c</i>	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。
	<i>r_cg_macrodriver.h</i>	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。
	<i>r_cg_main.c</i>	このファイルは常時生成されます。 <i>main()</i> 関数を定義します。
	<i>r_cg_systeminit.c</i>	このファイルは常時生成されます。 <i>R_ConfigName_Create</i> の名前を持つ全てのドライバの初期化関数を呼び出す <i>R_Systeminit</i> を含みます。 <i>R_Systeminit</i> は、クロックの初期化も呼び出します。
	<i>r_cg_userdefine.h</i>	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。
	<i>r_smc_interrupt.c</i>	このファイルは常時生成されます。
	<i>r_smc_interrupt.h</i>	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。
	<i>r_smc_entry.h</i>	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。 <i>r_cg_XXX_common.h</i> <i>r_cg_macrodriver.h</i> <i>r_cg_userdefine.h</i> <i>r_cg_cgc.h</i> {ConfigName}.h このファイルは、 <i>r_cg_main.c</i> にインクルードされます。

フォルダ	ファイル	説明
	<i>r_cg_intc_PEn.c</i>	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 <i>r_cg_intc_PEn.c</i> (n=0~3)が生成されます。 RH850/C1M : <i>r_cg_intc_PE1.c</i> が生成されます。 RH850/U2B : <i>r_cg_intc_PE0.c</i> 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。
	<i>r_cg_XXX_common.c</i> ^(*)	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。
	<i>r_cg_XXX_common.h</i> ^(*)	<i>r_cg_XXX_common.c</i> および <i>r_cg_XXX_common_user.c</i> 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。
	<i>r_cg_XXX_common_user.c</i> ^(*)	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。
	<i>r_smc_clock_info.h</i>	RH850/U2B 専用で生成されます。 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	<i>Pin.c</i>	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)。
	<i>Pin.h</i>	このファイルは常時生成されます。 このファイルは、 <i>Pin.c</i> 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、 <i>ConfigName</i> (設定名) の後に命名します。
	<i>{ConfigName}.c</i>	このファイルは、ドライバ(<i>R_ConfigName_Create</i>)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(<i>R_ConfigName_Start</i>)やストップ(<i>R_ConfigName_Stop</i>)を実行する関数を含みます。
	<i>{ConfigName}_user.c</i>	ドライバの初期化(<i>R_ConfigName_Create</i>)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。
	<i>{ConfigName}.h</i>	<i>{ConfigName}.c</i> と <i>{ConfigName}_user.c</i> 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。

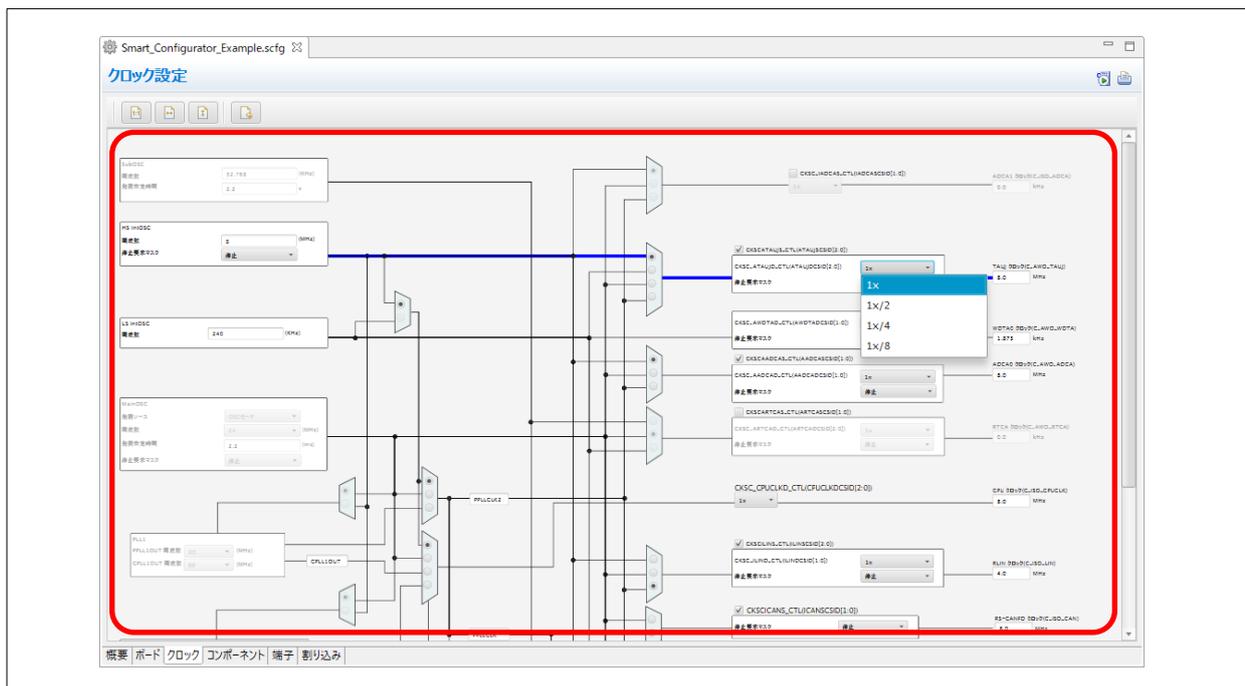


図 6-7 メインクロックをクロックソースとして選択した場合のクロック設定

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

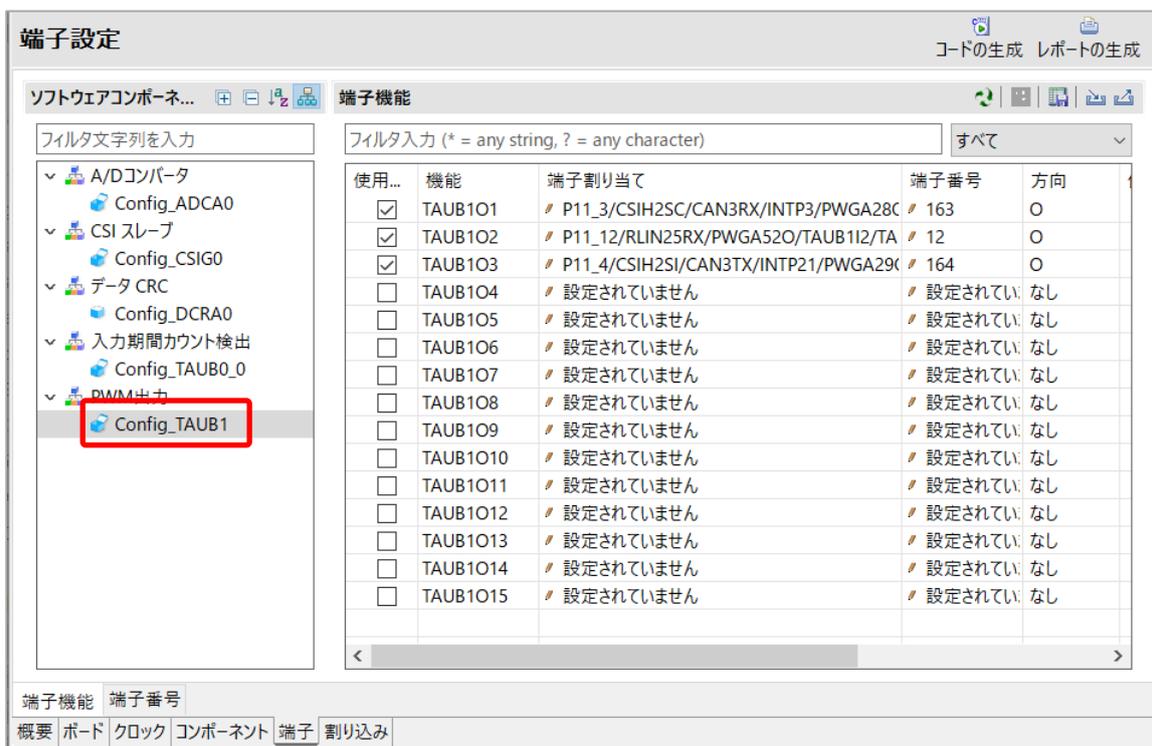


図 6-8 Config_TAUB1の端子設定

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:														
バグ番号	例外要因コード	割り込み要求元	周辺機能	優先レベル	状態	OS管理	割り込みハンドラ	エンティティを生成する	有効化/無効化機能を生成する					
29	101D	DMA transfer error (DMAERR)	DMA	最低	使用中	<input type="checkbox"/>	r_dmac_error_interrupt_pe1	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
				(1)	(2)	(3)	(4)							
RH850/U2A:														
バグ番号	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	割り込みハンドラ	エンティティを生成する	有効化/無効化機能を生成する	PE0	PE1	PE2	PE3
29	101DH	INTSDMACERR	sDMAC0 address error or sDMAC1 address erro...	sDMAC	最低	使用中	<input type="checkbox"/>	r_sdmac_address_error_interrupt	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
					(1)	(2)	(3)	(4)			(5)			

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 <i>R_ConfigName_Create</i> で初期化されます。この関数は、 <i>main()</i> 関数の中から <i>r_cg_systeminit.c</i> の <i>R_Systeminit</i> 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 <i>R_Interrupt_Initialize_ForPE()</i> 関数で設定されます。この関数は、 <i>main()</i> 関数の中から <i>r_cg_systeminit.c</i> の <i>R_Systeminit()</i> 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)	PEn (RH850/U2A のみ)	general	r_cg_intc_PEn.c	PE 選択の設定は、このファイルの <i>R_Interrupt_Initialize_ForPE</i> で初期化されます。この関数は、 <i>main()</i> 関数の中から <i>r_cg_systeminit.c</i> の <i>R_Systeminit()</i> 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 */
```

[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、PWM 出力(リソースTAUB1)の場合、xxxは“TAUB1”と名付けられます。）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 */
```

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、ユーザーが行っている設定のレポートを提供します。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(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(txt形式)

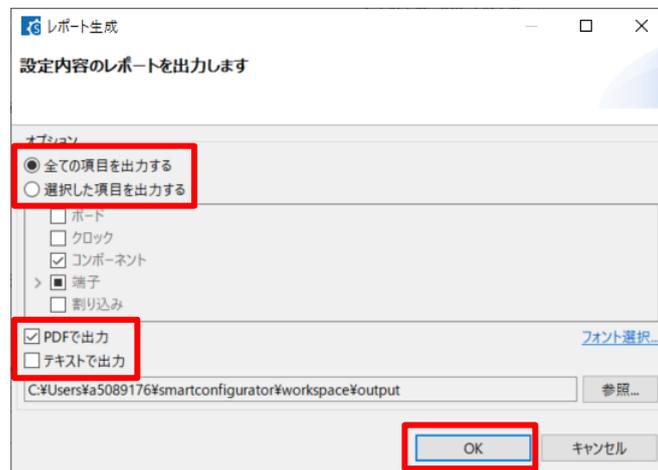


図 9-2 レポート出力ダイアログ

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。

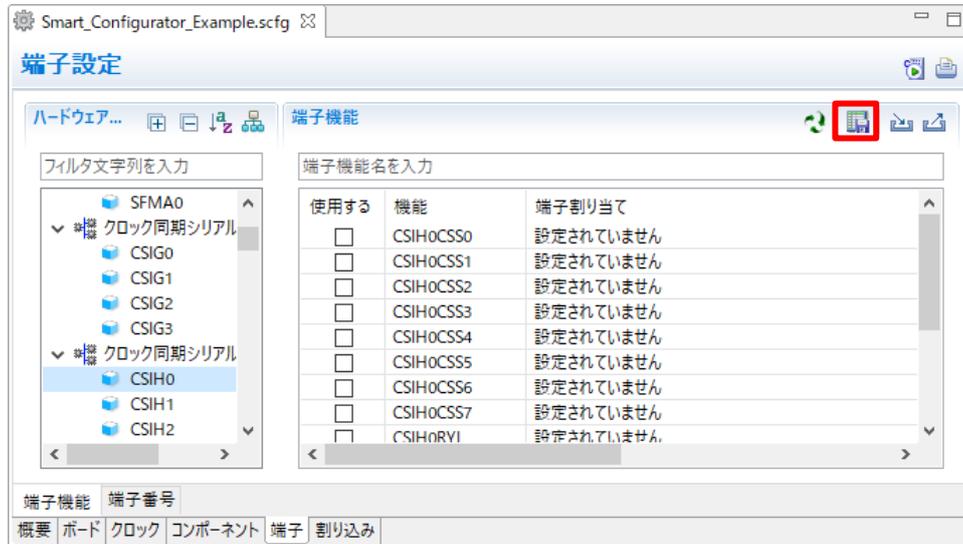


図 9-3 端子機能リスト、端子番号リスト出力 (csv形式)

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。

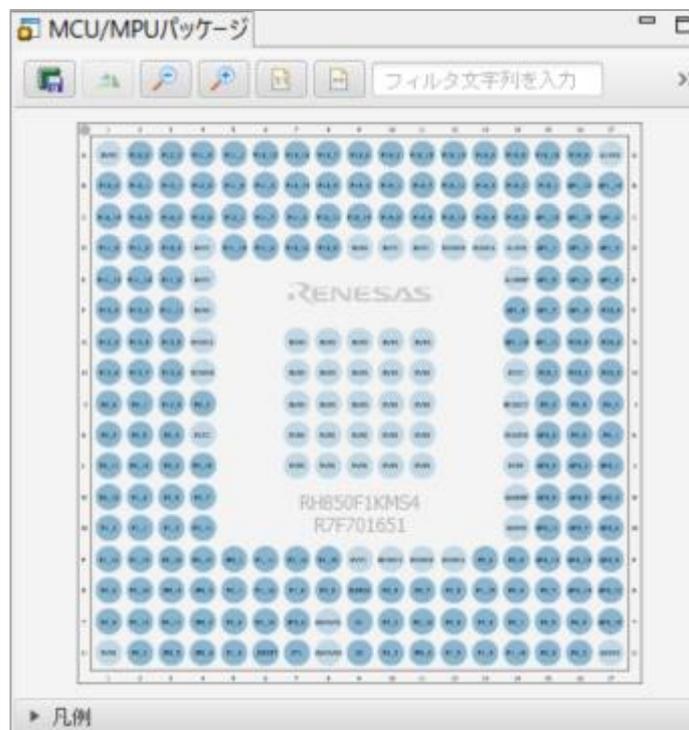


図 9-4 MCUパッケージ図出力 (png形式)

10.3 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。

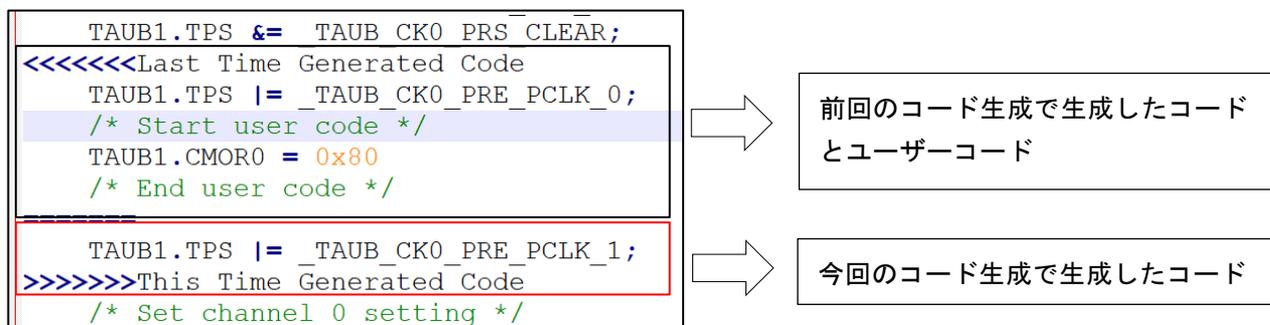


図10-3生成された競合コード

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。

```

コンソール | コンフィグレーションチェック
スマート・コンフィグレータ出力
M04000001: ファイルを生成: src\smc_gen\Config TAUB1\Config TAUB1.h
M04000001: ファイルを生成: src\smc_gen\Config TAUB1\Config TAUB1.c
M05000012: ファイルを生成: src\smc_gen\r_pincfg\Pin.h
M05000012: ファイルを生成: src\smc_gen\r_pincfg\Pin.c
M00000005: 赤色でハイライトされている上記のファイルには、ユーザーコードのマージが競合しています。ファイルを開き、手動で競合を解決してください
M00000002: コード生成の終了: C:\Users\A5089176\smartconfigurator\workspace\src\smc_gen

```

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)

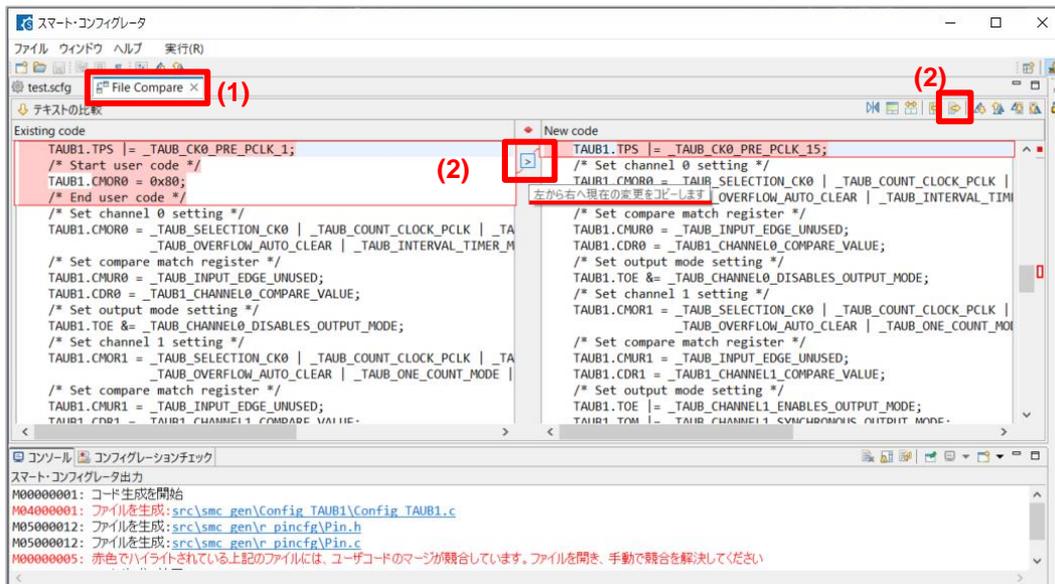


図 10-5 生成された競合コード

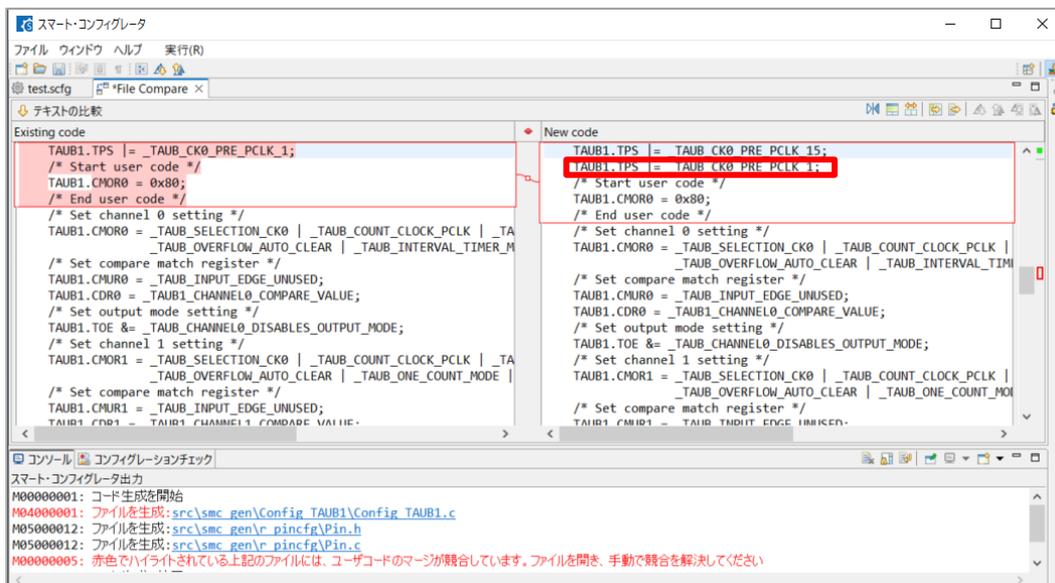


図 10-6 「左から右へ現在の変更をコピー」後のコード

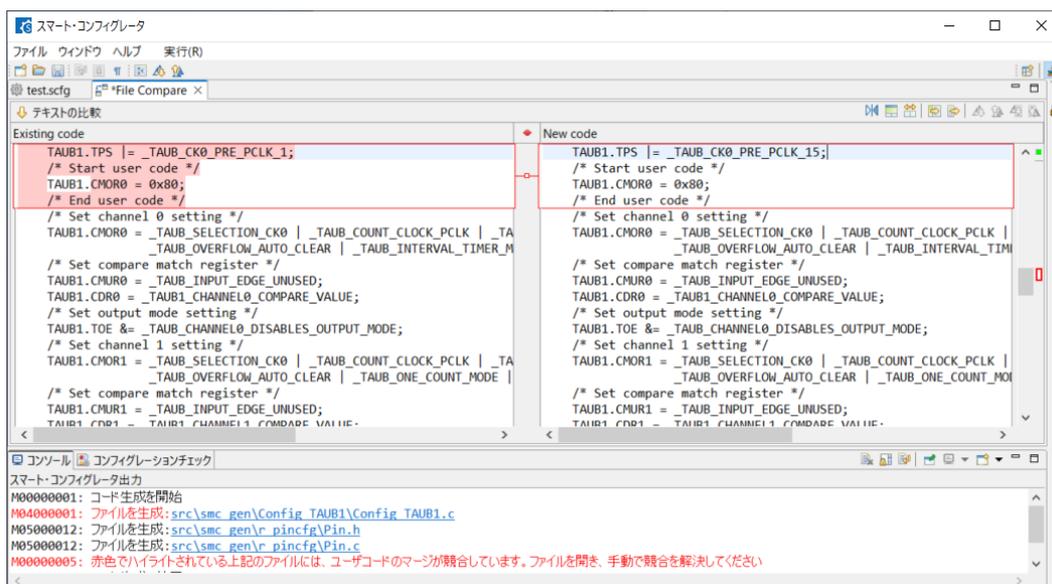


図 10-7 競合を解決した後のコード

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。

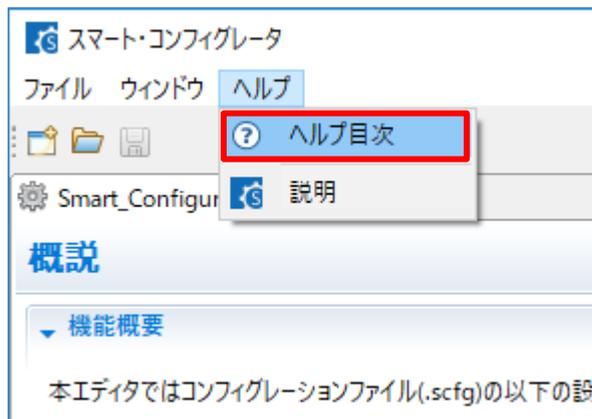


図 11-1 ヘルプ表示

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

(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

CS+ V8.11.00 統合開発環境 ユーザーズマニュアル プロジェクト操作編

CS+ V8.11.00 統合開発環境 ユーザーズマニュアル RH850デバッグ・ツール編

CS+ V8.11.00 統合開発環境 ユーザーズマニュアル メッセージ編

CC-RHコンパイラ ユーザーズマニュアル

(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>

お問い合わせ先

<http://www.renesas.com/contact/>

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。

改訂記録

改訂内容 ポイント	章	内容
1.00	—	新規作成
1.10	要旨	CS+ (CS+ for CC) V8.10.00
	全体	図を更新
	2.5	32bit 環境を削除、4 サンプルを追加。
	3.4.4 4.1.2 4.4.3 6.2 6.5	内容と図の更新
	4.3 4.4.6 4.5.3 4.5.6 ~ 4.5.8 4.6.2、4.6.3 10	追加
	5.2	端子競合の解消 を 5.2 に移動
	9.1 12	内容の更新
	1.20	要旨
3.4		図 3-3 メインウィンドウ、図 3-4 スマート・コンフィグレータビューの更新
4.6.3		割り込みハンドラ編集機能、エンティティ生成機能の更新 割り込み有効化/無効化関数生成機能の追加
6.5		割り込みハンドラ編集機能、割り込み有効化/無効化関数生成機能の追加
10		ユーザー コード保護機能のクロック生成ファイル、割り込み生成ファイルのサポートを追加
1.30	2.2	“All Toolchain (CC-RH, GHS, IAR)”選択時の注意事項を追加
	4.6.3	割り込みハンドラ名のルールを追加
	6.2	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. 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しないでください。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、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えば、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、ノイズ輻射量などが異なる場合があります。型名が違えば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したします。
 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

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。

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/

