RL78/G24

FAA LED 制御ライブラリ 導入ガイド

要旨

本アプリケーションノートでは Flexible Application Accelerator (FAA)を用いた LED 制御ライブラリ(以下 LED 制御ライブラリ)について記載しています。

FAA は RL78/G24 マイクロコントローラに内蔵されており、CPU とは独立したプロセッサとして動作します。

対象デバイス

RL78/G24

関連ドキュメント

・RL78/G24 ユーザーズマニュアル ハードウェア編 (R01UH0961)

・CS+ 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 (R20UT4691)

・RL78 スマート・コンフィグレータ ユーザーガイド：CS+編 (R20AN0580)

目次

1. LED 制御ライブラリの特徴 4
1.1 LED 定電流制御 4
1.2 Flexible Application Accelerator (FAA) 4

2. LED 制御ライブラリを使用可能なツール 5

3. LED 制御ライブラリのフォルダ構成 5

4. 使用リソース 6
4.1 周辺機能 6
4.2 ROM / RAM サイズ 6

5. LED 制御ライブラリの利用手順 7
5.1 CC-RL 環境 7
5.1.1 統合開発環境 CS+ (CS+ for CC) 7
5.1.2 スマート・コンフィグレータ 7
5.1.3 LED 制御ライブラリ 7
5.1.3.1 スマート・コンフィグレータの起動 7
5.1.3.2 クロック設定 8
5.1.3.3 FAA コンポーネントの追加 9
5.1.3.4 FAA モジュールのダウンロード 10
5.1.3.5 FAA モジュールのコンフィグレーション 11
5.1.3.6 コード生成 11
5.1.4 FAA を使用するためのプロジェクト設定(CS+) 12
5.1.4.1 FAA アセンブル・オプションの設定 12
5.1.4.2 セクション開始アドレスの設定 13
5.1.4.3 ROM から RAM へマップするセクションの設定 14

6. 制御仕様 15
6.1 PI（比例積分）制御 15
6.2 フィードバック制御周期 16
6.3 制御フロー 18
6.3.1.1 LED 制御開始処理(通常動作) 18
6.3.1.2 LED 制御フィードバック処理(通常動作) 19
6.3.1.3 LED 制御開始処理(高速動作) 20
6.3.1.4 LED 制御フィードバック処理(高速動作) 21

7. コンフィグレーション仕様 22

8. API 情報 25
8.1 API Typedef 定義 25
8.1.1 e_faa_ad_channel_t 25
8.1.2 e_faa_result_adc_t 25
8.2 API 関数仕様 26
8.2.1 R_{ConfigName}_LEDControl_Create 26

8.2.2 R_{ConfigName}_LEDControl_Start 27
8.2.3 R_{ConfigName}_LEDControl_Stop 28
8.2.4 R_{ConfigName}_LEDControl_SetTargetLevelCh{n} 29
8.2.5 R_{ConfigName}_LEDControl_GetCurrentLevelCh{n} 30
8.2.6 R_{ConfigName}_LEDControl_IsForceStopCh{n} 31
8.2.7 R_{ConfigName}_LEDControl_ClearForceStopCh{n} 32
8.2.8 R_{ConfigName}_LEDControl_RequestADC 33
8.2.9 R_{ConfigName}_LEDControl_GetAD 34

9. ホームページとサポート窓口 35

1. LED 制御ライブラリの特徴

本章では、LED 制御ライブラリの特徴について説明します。

1.1 LED 定電流制御

本ライブラリでは、タイマ KB PWM 出力機能を使用した最大 4 チャネルの LED 定電流制御を行います。LED 定電流制御は、PI（比例積分）制御に基づいたフィードバック処理にて実現されます。そのため、LED 定電流制御専用の外部 IC が不要になり、設計コストを削減することができます。

1.2 Flexible Application Accelerator (FAA)

LED 定電流制御を実現するためのフィードバック処理は FAA にて行われます。FAA は RL78/G24 マイクロコントローラに内蔵されており、CPU とは独立したプロセッサとして動作します。これにより、CPU を占有することなく高速で LED の調光制御処理を行うことができます。

FAA の詳細情報については RL78/G24 ユーザーズマニュアル ハードウェア編 (R01UH0961J)をご覧ください。

2. LED 制御ライブラリを使用可能なツール

本ライブラリは、スマート・コンフィグレータによるコード生成機能によりソースコードとして提供されます。生成されたソースコードは統合開発環境上でユーザープロジェクトに追加、ビルドする形で組み込みを行います。

表 2.1 に本ライブラリを使用するためのツール一覧を示します。利用手順については **5 LED 制御ライブラリの利用手順**を参照してください。

表 2.1 LED 制御ライブラリを使用可能なツール

項目	内容
統合開発環境(IDE)	CS+ for CC (Renesas IDE)
コンフィグレータ(SC)	Renesas Smart Configurator for RL78
コンパイラ	CC-RL

3. LED 制御ライブラリのフォルダ構成

以下に本ライブラリのフォルダ構成を示します。各ファイルはスマート・コンフィグレータによるコード生成機能により提供されます。

表 3.1 フォルダ構成

フォルダ、ファイル名	説明
smc_gen	SC 生成フォルダ
¥{ConfigName}	FAA コンポーネントフォルダ
{ConfigName}_common.c	FAA 共通機能のソースファイル
{ConfigName}_common.h	FAA 共通機能のヘッダファイル
{ConfigName}_common.inc	FAA 共通機能のインクルードファイル
{ConfigName}_LEDControl.c	LED 制御機能のソースファイル
{ConfigName}_LEDControl.h	LED 制御機能のヘッダファイル
{ConfigName}_src.dsp	FAA DSP アセンブラソースファイル

4. 使用リソース

4.1 周辺機能

本ライブラリは以下の周辺機能を使用します。

- フレキシブル・アプリケーション・アクセラレータ
- プログラマブル・ゲイン・アンプ
- A/D コンバータ
- D/A コンバータ
- コンパレータ
- 16 ビット・タイマ KB

4.2 ROM / RAM サイズ

以下のオプションを使用してビルドした際の ROM / RAM サイズを参考として記します。

コンパイラオプション

-cpu=S3 -memory_model=medium –Odefault

リンク・オプション

-NOOPtimize

表 4.1 ROM / RAM サイズ

LED チャネル数	ROM	RAM	FAACODE	FAADATA
1 (高速動作)	1,419 [byte]	0 [byte]	376 [byte]	172 [byte]
1	1,730 [byte]	0 [byte]	336 [byte]	172 [byte]
2	2,104 [byte]	0 [byte]	516 [byte]	204 [byte]
3	2,393 [byte]	0 [byte]	608 [byte]	236 [byte]
4	2,624 [byte]	0 [byte]	700 [byte]	268 [byte]

注. スマート・コンフィグレータの設定内容によってサイズが変化するため、上表では最大値を記載しています。

5. LED 制御ライブラリの利用手順

本ライブラリの利用手順を以下に示します。

5.1 CC-RL 環境

5.1.1 統合開発環境 CS+ (CS+ for CC)

CS+は Renesas ホームページからダウンロードしてください。

[Renesas ホームページ]

https://www.renesas.com/jp/ja/products/software-tools/tools.html

CS+の基本操作については以下ユーザーズマニュアルを参照してください。

- CS+ 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 (R20UT4691)

5.1.2 スマート・コンフィグレータ

下記 URL から「RL78 スマート・コンフィグレータ」および「CS+ RL78 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+ RL78 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/rl78-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。

5.1.3 LED 制御ライブラリ

スマート・コンフィグレータ上で、LED 制御ライブラリを利用する手順について説明します。

スマート・コンフィグレータの基本的な操作方法については下記ユーザーガイドを参照してください。

- RL78 スマート・コンフィグレータ ユーザーガイド：CS+編 (R20AN0580)

5.1.3.1 スマート・コンフィグレータの起動

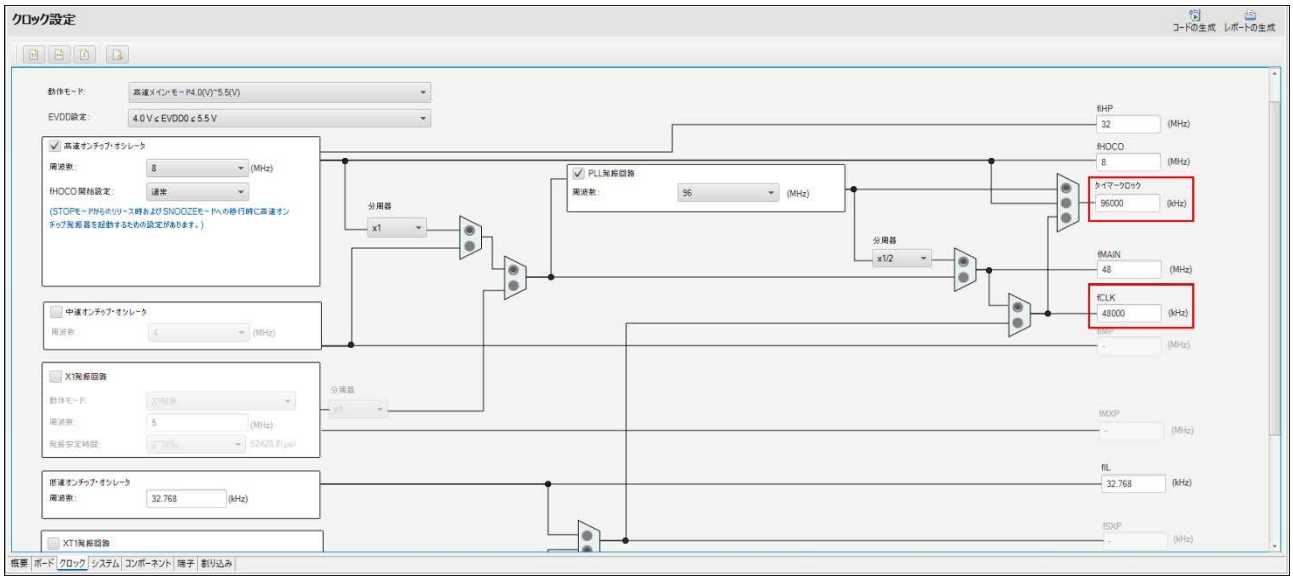
CS+にて新規プロジェクトの作成または既存プロジェクトの読み込みを行い、CS+画面よりスマート・コンフィグレータを起動してください。

5.1.3.2 クロック設定

「スマート・コンフィグレータビュー」の「クロック」ページを選択し、「タイマークロック」および「f_{CLK}」出力を下記のように設定してください。

- タイマークロック：96000[kHz]
- f_{CLK}：48000[kHz]

図 5-1 クロック設定

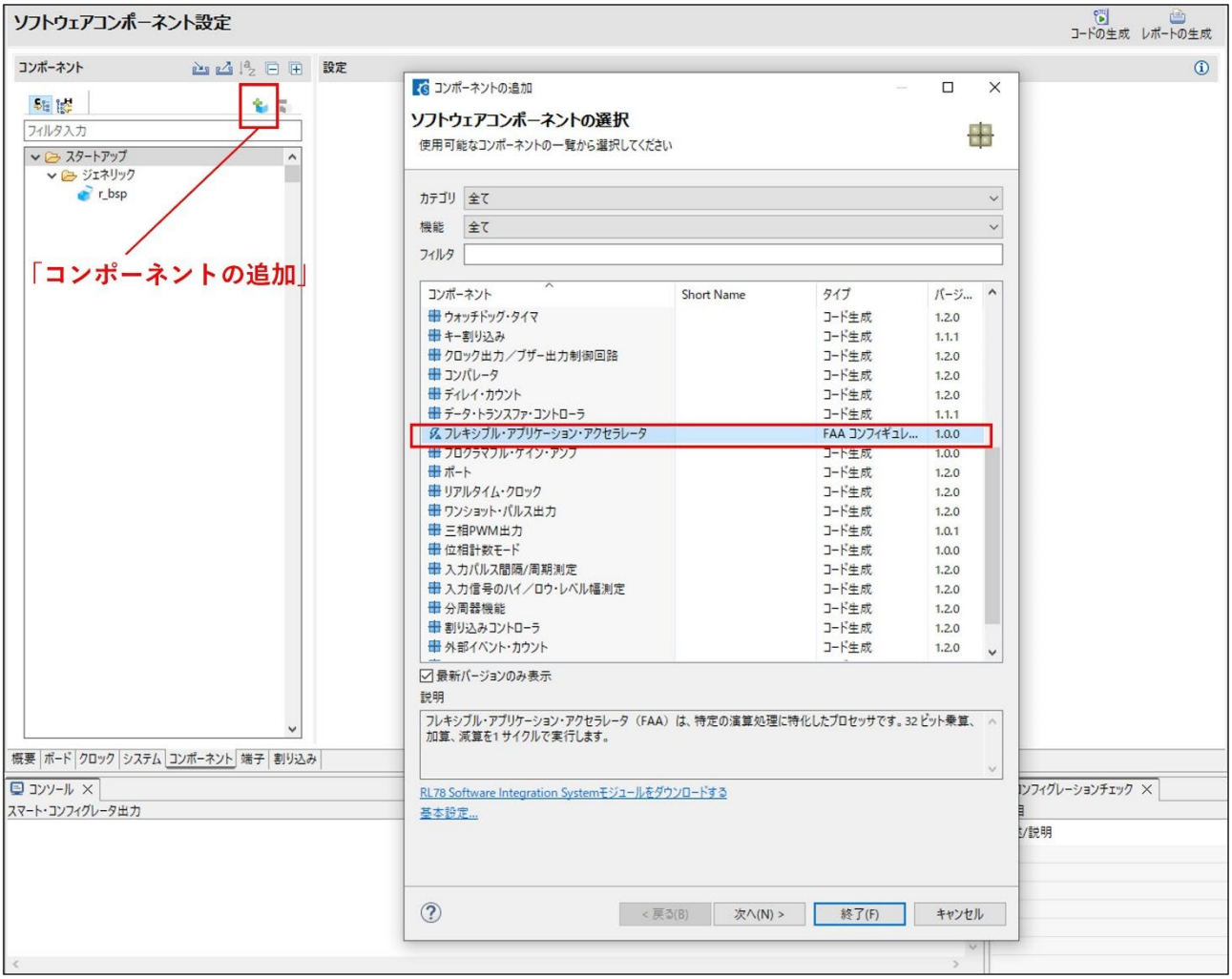


5.1.3.3 FAA コンポーネントの追加

「スマート・コンフィグレータビュー」の「コンポーネント」ページを選択し、「コンポーネントの追加」ボタンを押下してください。

次に「ソフトウェアコンポーネントの選択」画面より「フレキシブル・アプリケーション・アクセラレータ」コンポーネントを追加してください。

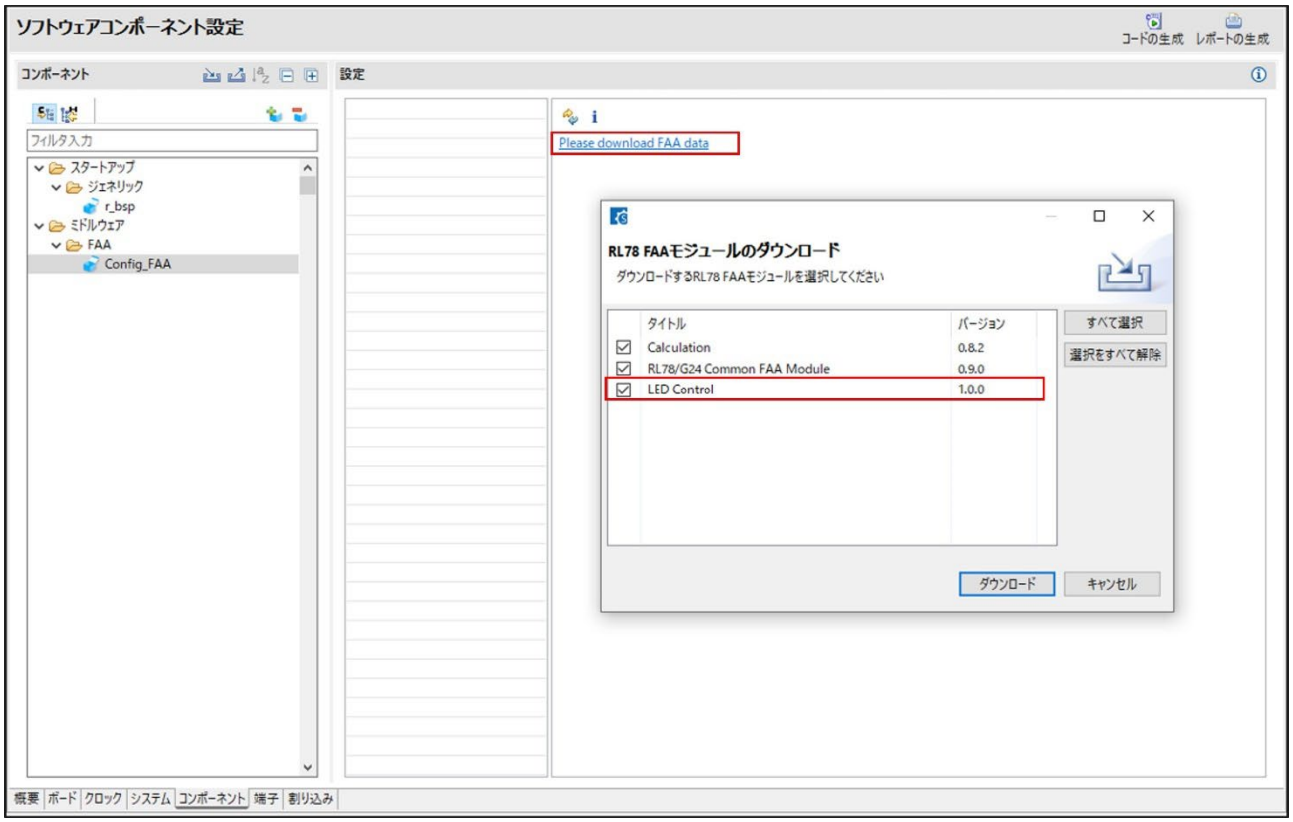
図 5-2 FAA コンポーネントの追加



5.1.3.4 FAA モジュールのダウンロード

画面上に表示されている「Please download FAA data」をクリックするとダウンロード可能な FAA モジュールが表示されます。「LED Control」を選択してダウンロードしてください。

図 5-3 FAA モジュールのダウンロード

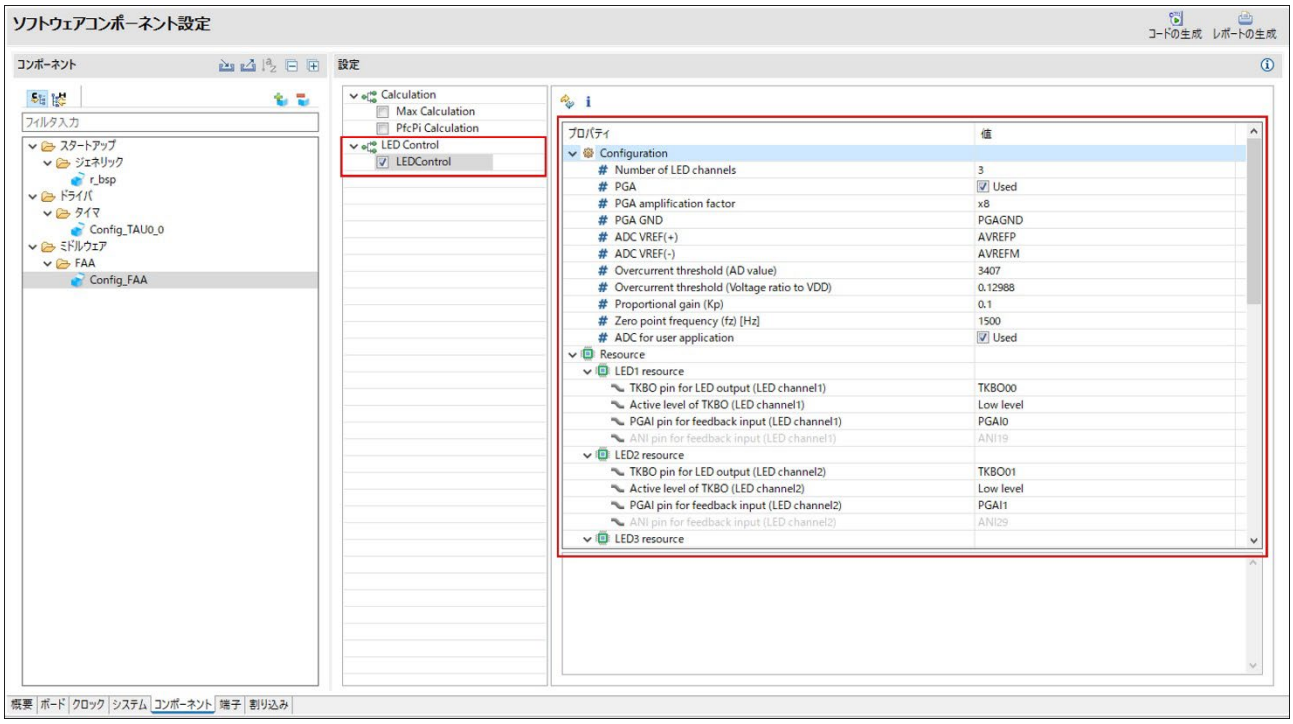


5.1.3.5 FAA モジュールのコンフィグレーション

ダウンロードされた FAA モジュールの一覧より「LEDControl」モジュールを選択すると、コンフィグレーション画面が表示されます。

ユーザー環境に応じてコンフィグレーション設定を行ってください。各コンフィグレーション項目の詳細については **7 コンフィグレーション仕様**をご覧ください。

図 5-4 FAA モジュールのコンフィグレーション



5.1.3.6 コード生成

コンフィグレーション設定後、「コード生成」ボタンを押下することで、LED 制御ライブラリのソースコードが生成されます。生成されたソースコードは CS+上のユーザープロジェクトに自動登録されます。

ユーザーアプリケーションにて、本ライブラリより提供される API 関数のコール処理を実装してください。各種 API 関数の詳細については **8.2 API 関数仕様**をご覧ください。

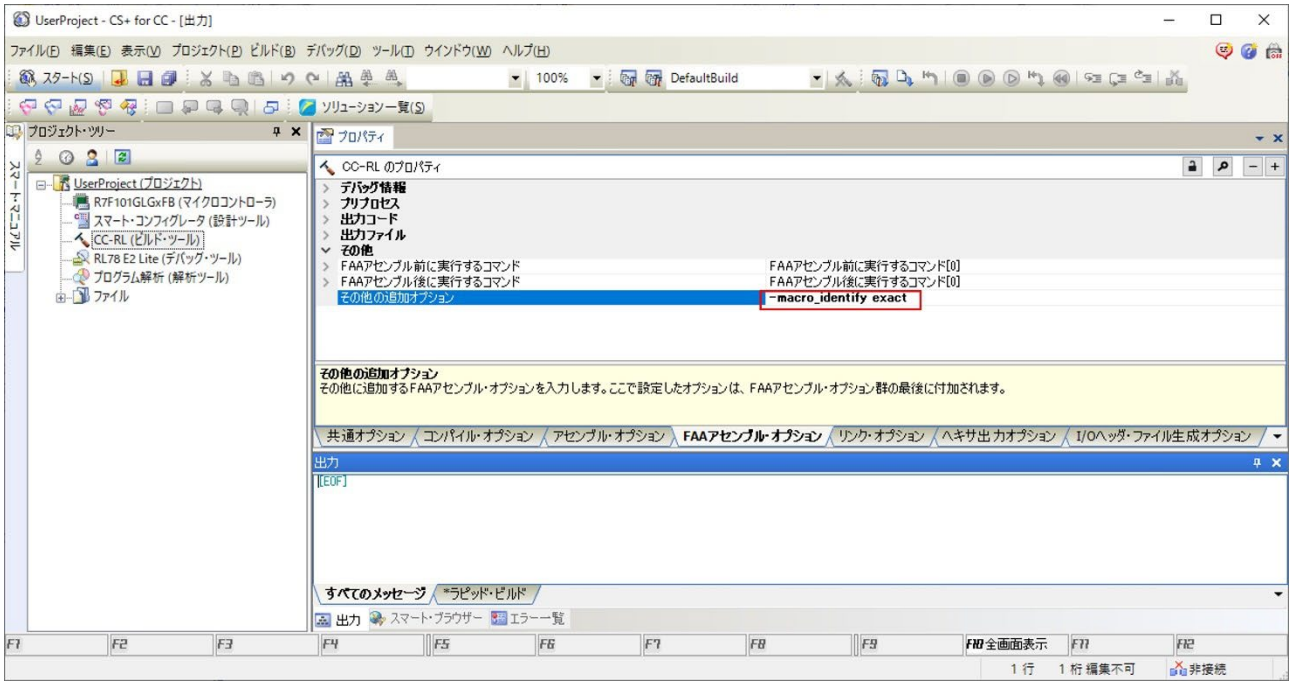
5.1.4 FAA を使用するためのプロジェクト設定(CS+)

CS+にて FAA を使用するためには以下のプロジェクト設定が必要となります。

5.1.4.1 FAA アセンブル・オプションの設定

「CC-RL(ビルドツール)」->「FAA アセンブル・オプション」->「その他の追加オプション」に追加オプション” -macro_identify exact”を設定してください。

図 5-5 FAA アセンブル・オプションの設定



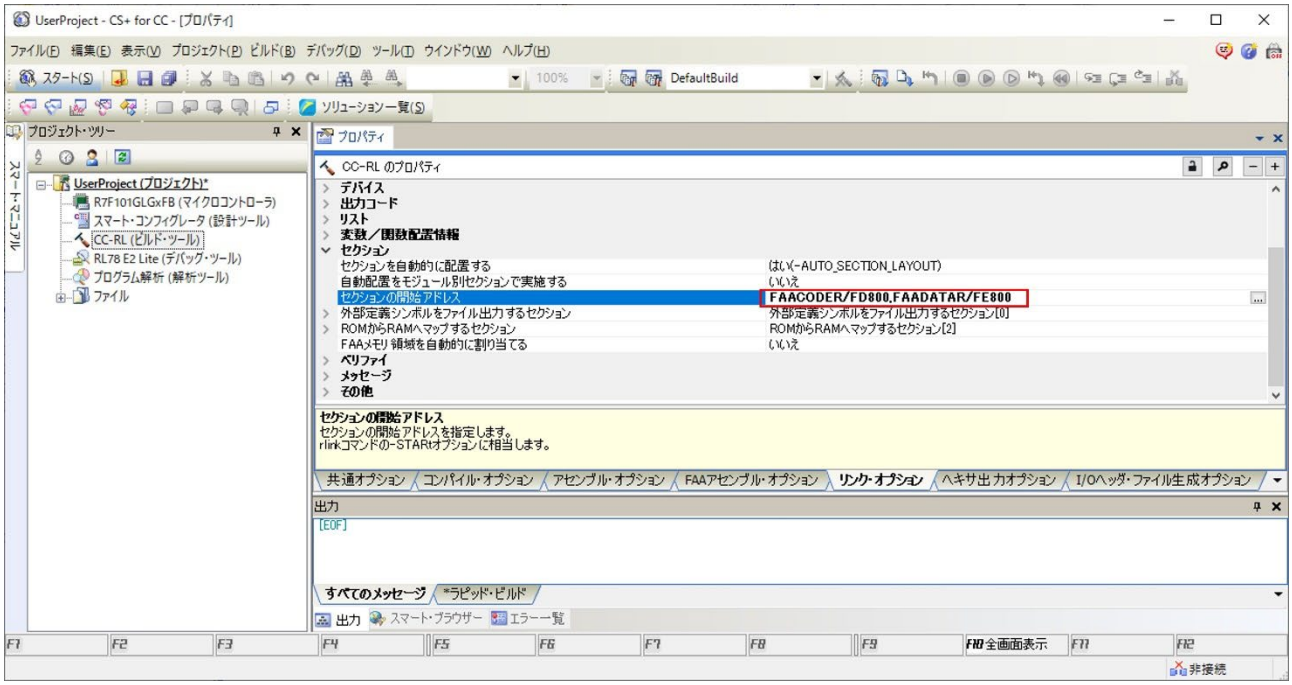
5.1.4.2 セクション開始アドレスの設定

「CC-RL(ビルドツール)」->「リンク・オプション」->「セクションの開始アドレス」を以下のように設定してください。

表 5.1 セクション開始アドレスの設定

アドレス	セクション
0xFD800	FAACODER
0xFE800	FAADATAR

図 5-6 セクション開始アドレスの設定

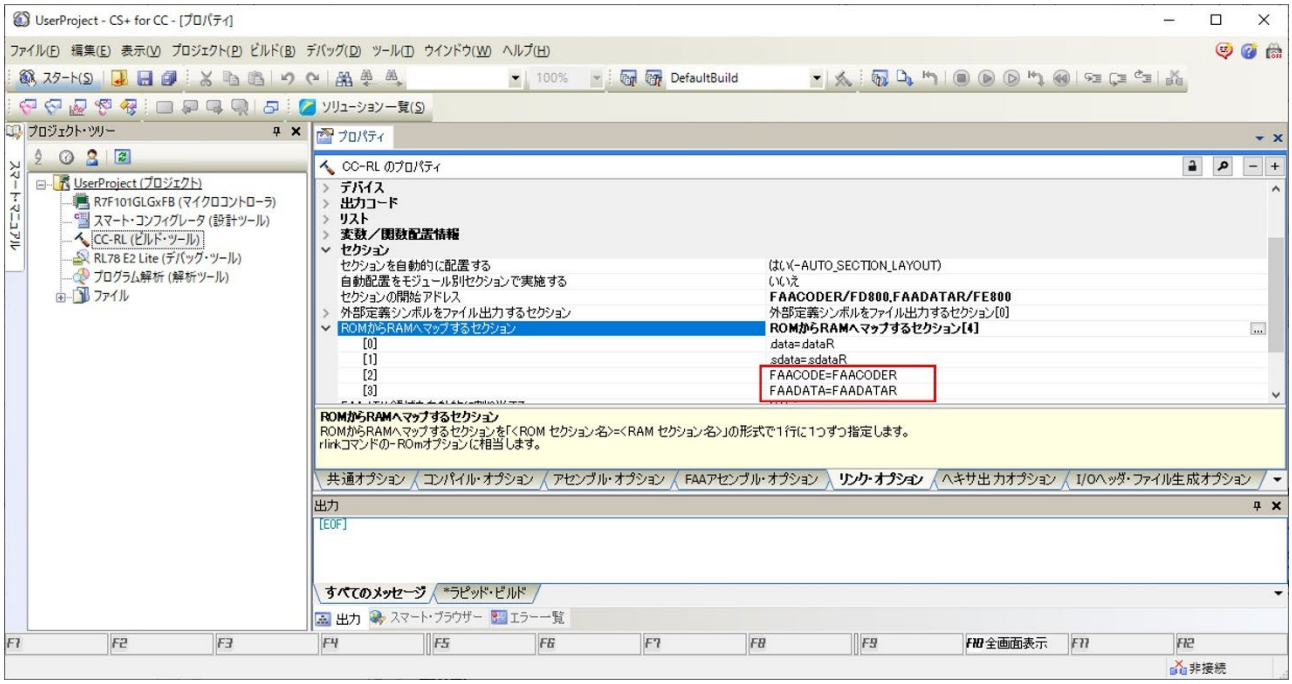


5.1.4.3 ROM から RAM へマップするセクションの設定

「CC-RL(ビルドツール)」->「リンク・オプション」->「ROM から RAM へマップするセクション」に以下の設定を追加してください。

- “FAACODE=FAACODER”
- “FAADATA=FAADATAR”

図 5-7 ROM から RAM へマップするセクションの設定



6. 制御仕様

本ライブラリの LED 定電流制御について説明します。

6.1 PI（比例積分）制御

LED 定電流制御は PI（比例積分）制御に基づいたフィードバック処理にて実現されます。

PI 制御式を以下に示します。

$$D(n) = D(n-1) + A_1 \cdot E(n) + A_2 \cdot E(n-1)$$

D（n)： 最新の PWM 出力デューティ

D（n−1)： 前回の PWM 出力デューティ

E（n)： 最新の誤差値＝（A/D 変換目標値）−（最新の A/D 変換測定値）

E（n−1)： 前回の誤差値＝（A/D 変換目標値）−（前回の A/D 変換測定値）

A1、A2： 係数

係数 A1 および A2 は次式から得られます。

$$A1 = (\pi \times f_Z \times T + 1) \times K_P$$
$$A2 = (\pi \times f_Z \times T - 1) \times K_P$$

π： パイ（円周率）
f_Z： ゼロ・ポイント周波数
T： フィードバック制御周期
K_P： 比例定数

上記のように係数 A1 および A2 は、3 つのパラメータ f_Z、T、K_P を基に計算されます。

ゼロ・ポイント周波数 f_Z、比例定数 K_P については**表 7.1 スマート・コンフィグレータ設定項目一覧(1/3)** に記載のとおり、ユーザー環境に応じた任意値の設定が可能となっております。

フィードバック制御周期 T についてはスマート・コンフィグレータ設定内容に応じて、自動計算されます。詳細については **6.2 フィードバック制御周期**をご覧ください。

6.2 フィードバック制御周期

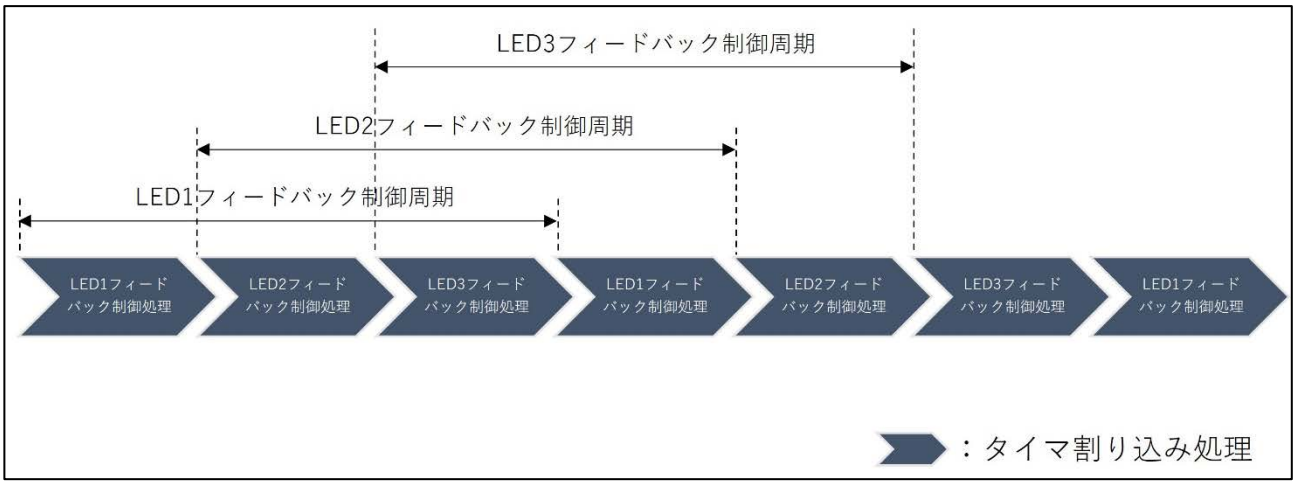
フィードバック制御処理はインターバルな FAA タイマ割り込みタイミングで行われます。制御対象の LED チャネル数が 2 以上の場合は、タイマ割り込み毎に 1 チャネルずつフィードバック制御処理が実施されます。

従って、フィードバック制御周期 T は

T = タイマ割り込み周期 × LED チャネル数

となります。

図 6-1 フィードバック制御周期 (例:LED チャネル数=3)



タイマ割り込み周期はスマート・コンフィグレータの設定内容に応じて、最速動作となるよう以下の設定が適用されます。スマート・コンフィグの設定項目については **7 コンフィグレーション仕様**をご覧ください。

① Number of LED channels = 1 (LED チャネル数 = 1)の場合

ADC for user application = Unused 設定時：2.5 μs

ADC for user application = Used 設定時：6 μs

② Number of LED channels = 2～4 (LED チャネル数 = 2～4)の場合

PGA = Unused 設定時：6.5 μs

PGA = Unused, PGA amplification factor = x4/x8 設定時：7 μs

PGA = Unused, PGA amplification factor = x16/x32 設定時：12 μs

表 6.1 タイマ割り込み周期

LED 数	割り込み周期				
	ADC for user application=Unused	ADC for user application=Used	PGA=Unused	PGA=Used	
				PGA amplification factor = x4,x8	PGA amplification factor = x16,x32
1	2.5μs	6μs			
2 - 4			6.5μs	7μs	12μs

このように、タイマ割り込み周期は条件別に最小値が適用されるような仕様となります。

割り込み周期を任意の値に変更したい場合は、{ConfigName}_LEDControl.h ファイル内の定数値(R_{ConfigName}_LEDControl_TMCP0)を手動で修正してください。

”{ConfigName}”はスマート・コンフィグレータにて設定した FAA コンポーネントのコンフィグレーション名を示します。

Example

```
/* Set 100μs @48MHz */
#define R_Config_FAA_LEDControl_TMCP0                    (0x000012C0)
```

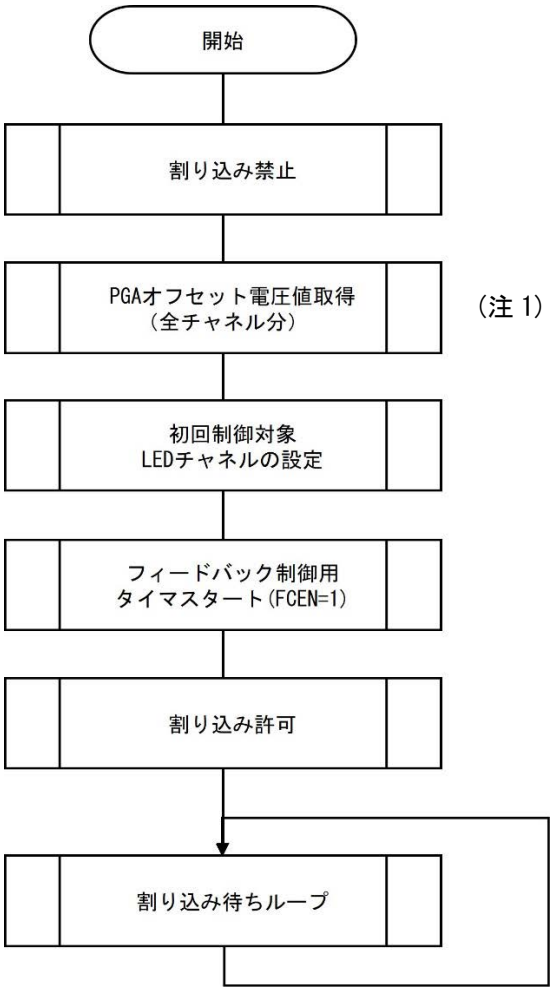
6.3 制御フロー

LED 定電流制御のフローを以下に示します。制御処理は” {ConfigName}_src.dsp”ファイルに実装されており、FAA によって実行されます。

6.3.1.1 LED 制御開始処理(通常動作)

API 関数「R_{ConfigName}_LEDControl_Start」コールにより、LED 制御開始処理が実行されます。

図 6-2 LED 制御開始処理(通常動作)

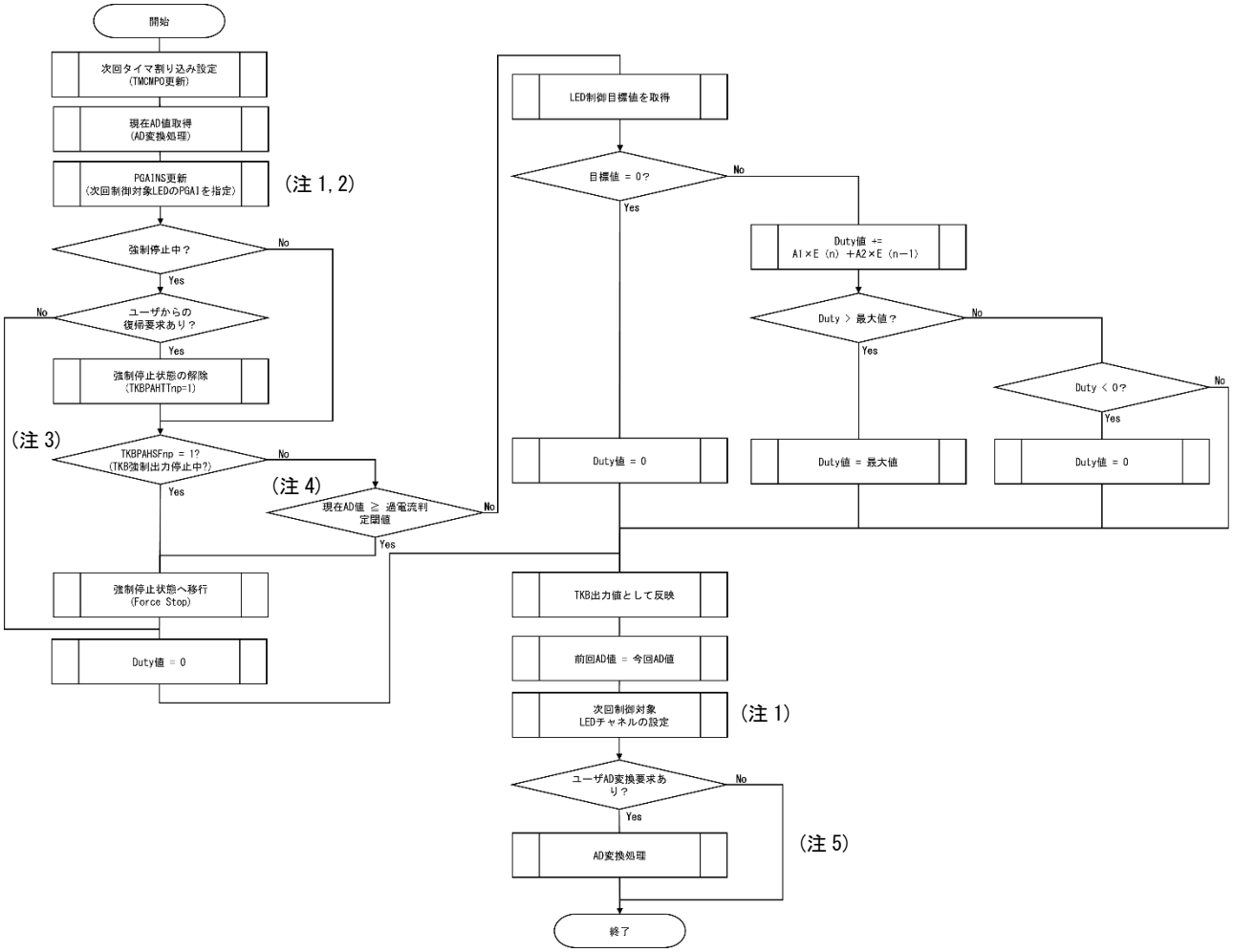


注1. 「PGA」=「Unused」設定時は無効となります。

6.3.1.2 LED 制御フィードバック処理(通常動作)

LED 制御開始処理後、インターバルなタイマ割り込みにより LED 制御フィードバック処理が実行されます。

図 6-3 LED 制御フィードバック処理(通常動作)



注1. 「Number of LED channels」=「1」設定時は無効となります。

注2. 「PGA」=「Unused」設定時は無効となります。

注3. 「Overcurrent threshold (Voltage ratio to VDD)」=「0」設定時は無効となります。

注4. 「Overcurrent threshold (AD value)」=「0」設定時は無効となります。

注5. 「ADC for user application」=「Unused」設定時は無効となります。

6.3.1.3 LED 制御開始処理(高速動作)

スマート・コンフィグレータにて下記の設定が指定された場合、より高速なフィードバック処理が適用されます。

スマート・コンフィグレータ設定：

「Number of LED channels」=「1」且つ「ADC for user application」= 「Unused」

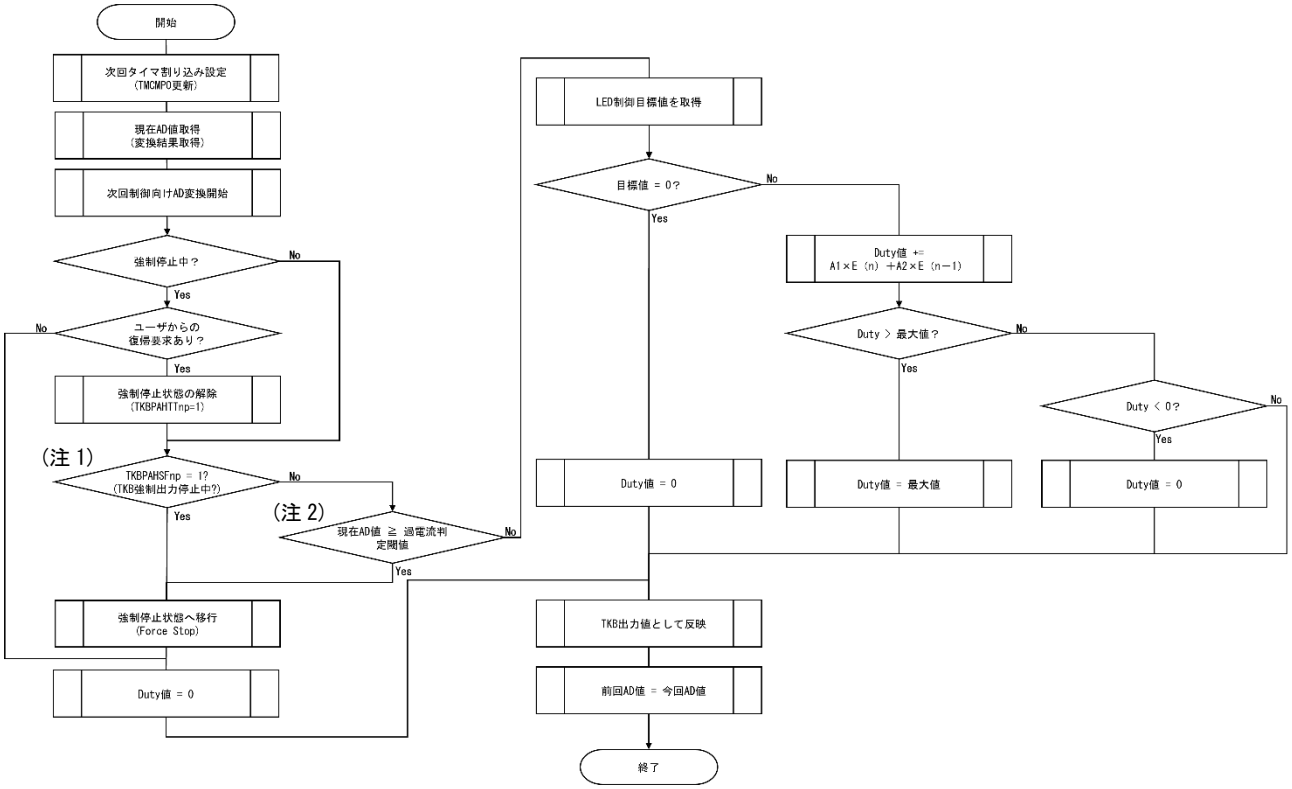
図 6-4 LED 制御開始処理(高速動作)

注1. 「PGA」=「Unused」設定時は無効となります。

6.3.1.4 LED 制御フィードバック処理(高速動作)

高速動作適用時、LED 制御フィードバック処理は 2.5μs 間隔で実行されます。

図 6-5 LED 制御フィードバック処理(高速動作)



注1. 「Overcurrent threshold (Voltage ratio to VDD)」=「0」設定時は無効となります。

注2. 「Overcurrent threshold (AD value)」=「0」設定時は無効となります。

7. コンフィグレーション仕様

以下にスマート・コンフィグレータにて設定可能なコンフィグレーション項目一覧を示します。

表 7.1 スマート・コンフィグレータ設定項目一覧(1/3)

項目	取りうる値	説明
Number of LED channels	1～4 (注 1)	制御対象の LED チャネル数を選択します。
PGA	Unused/Used	LED 電流値の検出に PGA を使用するかどうかを選択します。
PGA amplification factor	x4/x8/x16/x32	PGA によるフィードバック電流の増幅率を選択します。
PGA GND	VSS/PGAGND	PGA のフィードバック抵抗の GND を選択します。
ADC VREF(+)	VDD/AVREFP	A/D コンバータのプラス側の基準電圧を選択します。
ADC VREF(-)	VSS/AVREFM	A/D コンバータのマイナス側の基準電圧を選択します。
Overcurrent threshold (AD value)	0～4095	LED の過電流判定閾値(AD 値)を設定します。 PGA を使用する場合は、増幅率を考慮した AD 値を設定します。 0 に設定すると、この機能は無効になります。
Overcurrent threshold (Voltage ratio to VDD)	0～0.99999…	TKB による強制出力停止をトリガするための LED 電流値(電圧値換算値)を VDD 電圧値との比率で設定します。 0 に設定すると、この機能は無効になります。 例：VDD=5V、過電流判定閾値が 0.6V の場合 0.6V / 5V = 0.12
Proportional gain (Kp)	0.0001～1000	PI フィードバック制御における比例ゲイン値を設定します。
Zero point frequency (fz) [Hz]	1～100000	PI フィードバック制御におけるゼロ点周波数を設定します。
ADC for user application	Unused/Used	ユーザーアプリケーションにて ADC を使用するかどうかを選択します。

注1. 20 ピン製品では「1～3」となります。

表 7.2 スマート・コンフィグレータ設定項目一覧(2/3)

項目	取りうる値	説明
TKBO pin for LED output (LED chanel1)	TKBO00/TKBO01/TKBO10 /TKOB11/TKBO20/TKBO21 (注 1)	LED チャネル 1 への出力に使用する TKBO ピンを選択します。
TKBO pin for LED output (LED chanel2)	TKBO00/TKBO01/TKBO10 /TKOB11/TKBO20/TKBO21 (注 1)	LED チャネル 2 への出力に使用する TKBO ピンを選択します。
TKBO pin for LED output (LED chanel3)	TKBO00/TKBO01/TKBO10 /TKOB11/TKBO20/TKBO21 (注 1)	LED チャネル 3 への出力に使用する TKBO ピンを選択します。
TKBO pin for LED output (LED chanel4)	TKBO00/TKBO01/TKBO10 /TKOB11/TKBO20/TKBO21 (注 1)	LED チャネル 4 への出力に使用する TKBO ピンを選択します。
Active level of TKBO (LED channel1)	Low level/High level	LED チャネル 1 への出力に使用する TKBO のアクティブレベルを選択します。
Active level of TKBO (LED channel2)	Low level/High level	LED チャネル 2 への出力に使用する TKBO のアクティブレベルを選択します。
Active level of TKBO (LED channel3)	Low level/High level	LED チャネル 3 への出力に使用する TKBO のアクティブレベルを選択します。
Active level of TKBO (LED channel4)	Low level/High level	LED チャネル 4 への出力に使用する TKBO のアクティブレベルを選択します。
PGAI pin for feedback input (LED chanel1)	PGAI0/ PGAI1/PGAI2/PGAI3 (注 2)	LED チャネル 1 のフィードバック入力に使用する PGAI ピンを選択します。
PGAI pin for feedback input (LED chanel2)	PGAI0/ PGAI1/PGAI2/PGAI3 (注 2)	LED チャネル 2 のフィードバック入力に使用する PGAI ピンを選択します。
PGAI pin for feedback input (LED chanel3)	PGAI0/ PGAI1/PGAI2/PGAI3 (注 2)	LED チャネル 3 のフィードバック入力に使用する PGAI ピンを選択します。
PGAI pin for feedback input (LED chanel4)	PGAI0/ PGAI1/PGAI2/PGAI3 (注 2)	LED チャネル 4 のフィードバック入力に使用する PGAI ピンを選択します。
ANI pin for feedback input (LED chanel1)	ANI18/ANI19/ANI29/ANI30 (注 3)	LED チャネル 1 のフィードバック入力に使用する ANI ピンを選択します。
ANI pin for feedback input (LED chanel2)	ANI18/ANI19/ANI29/ANI30 (注 3)	LED チャネル 2 のフィードバック入力に使用する ANI ピンを選択します。
ANI pin for feedback input (LED chanel3)	ANI18/ANI19/ANI29/ANI30 (注 3)	LED チャネル 3 のフィードバック入力に使用する ANI ピンを選択します。
ANI pin for feedback input (LED chanel4)	ANI18/ANI19/ANI29/ANI30 (注 3)	LED チャネル 4 のフィードバック入力に使用する ANI ピンを選択します。

注1. 20 ピン製品では「TKBO20」「TKBO21」はサポート外となります。

注2. 20 ピン製品では「PGAI3」はサポート外となります。

注3. 20 ピン製品では「ANI18」はサポート外となります。

表 7.3 スマート・コンフィグレータ設定項目一覧(3/3)

項目	取りうる値	説明
ANI0	Unused/Used	ユーザーアプリケーションでアナログ入力(ANI)を使用するかどうかを選択します。
ANI1	Unused/Used	
ANI2	Unused/Used	
ANI3	Unused/Used	
ANI4	Unused/Used (注 1,2,3)	
ANI5	Unused/Used (注 1,2,3)	
ANI6	Unused/Used (注 1,2,3)	
ANI7	Unused/Used (注 1,2,3,4)	
ANI16	Unused/Used (注 1,2,3,4,5)	
ANI17	Unused/Used (注 1,2,3,4,5)	
ANI18	Unused/Used (注 1)	
ANI19	Unused/Used	
ANI20	Unused/Used	
ANI21	Unused/Used	
ANI22	Unused/Used	
ANI23	Unused/Used	
ANI24	Unused/Used (注 2)	
ANI25	Unused/Used (注 1,2)	
ANI26	Unused/Used (注 1)	
ANI27	Unused/Used (注 1,2)	
ANI28	Unused/Used (注 1,2,3,4)	
ANI29	Unused/Used	
ANI30	Unused/Used	

注1. 20 ピン製品ではサポート外となり、「Used」を選択することができません。

注2. 24 ピン製品ではサポート外となり、「Used」を選択することができません。

注3. 25-32 ピン製品ではサポート外となり、「Used」を選択することができません。

注4. 40 ピン製品ではサポート外となり、「Used」を選択することができません。

注5. 44-48 ピン製品ではサポート外となり、「Used」を選択することができません。

8. API 情報

8.1 API Typedef 定義

本ライブラリが提供する Typedef 定義について説明します。

8.1.1 e_faa_ad_channel_t

この Typedef はアナログ入力チャネルを定義します。ユーザーアプリケーション向けに提供する ADC 機能において、変換対象のチャネルを指定するために使用されます。

表 7.3 スマート・コンフィグレータ設定項目一覧(3/3)の設定に応じて対象となるアナログチャネル定義が生成されます。

```
typedef enum
{
    ADCHANNEL0 = 0,
    ADCHANNEL1 = 1,
    ADCHANNEL2 = 2,
    ADCHANNEL2 = 3,
    . . .

} e_faa_ad_channel_t;
```

8.1.2 e_faa_result_adc_t

この Typedef はユーザーアプリケーション向け ADC 機能における変換結果を定義します。

```
typedef enum
{
    FAA_ADC_NOTCOMPLETED,
    FAA_ADC_COMPLETED,
    FAA_ADC_FAILED
} e_faa_result_adc_t;
```

8.2 API 関数仕様

本ライブラリが提供する API 関数について説明します。

API 関数名に含まれる”{ConfigName}”はスマート・コンフィグレータにて設定した FAA コンポーネントのコンフィグレーション名を示します。また、”{n}”は LED チャネルの番号を示します。

8.2.1 R_{ConfigName}_LEDControl_Create

この関数は、LED 制御で必要な周辺機能の初期化処理を実施します。

Format

```
void R_{ConfigName}_LEDControl_Create (void)
```

Parameters

なし

Return Values

なし

Properties

{ConfigName}_LEDControl.h にプロトタイプ宣言されています。

Description

この関数は、LED 制御で使用する下記周辺機能の初期化処理を実施します。

・プログラマブル・ゲイン・アンプ

・A/D コンバータ

・D/A コンバータ

・コンパレータ

・16 ビット・タイマ KB

Example

```
/* Set FAA settings */
R_Config_FAA_Create();
R_Config_FAA_LEDControl_Create();
```

Special Notes:

この関数のコール処理はスマート・コンフィグレータより生成されるソースコードに含まれるため、ユーザープログラムからの関数コール処理は不要となります。

8.2.2 R_{ConfigName}_LEDControl_Start

この関数は、LED 制御で必要な周辺機能および FAA 処理を開始します。

Format

```
FAA_Status_t R_{ConfigName}_LEDControl_Start (void)
```

Parameters

なし

Return Values

FAA_SUCCESS	FAA 処理成功
FAA_ALREADY_RUNNING	FAA 実行中

Properties

{ConfigName}_LEDControl.h にプロトタイプ宣言されています。

Description

この関数は、LED 制御で使用する下記周辺機能および FAA 処理を開始します。

・プログラマブル・ゲイン・アンプ

・A/D コンバータ

・D/A コンバータ

・コンパレータ

・16 ビット・タイマ KB

Example

```
/* Start LED Control */
FAA_Status_t status;
status = R_Config_FAA_Start();

if (status != FAA_SUCCESS)
{
  . . .
}
```

Special Notes:

この関数をコールする前に必ず R_{ConfigName}_LEDControl_Create()をコールしてください。

8.2.3 R_{ConfigName}_LEDControl_Stop

この関数は、LED 制御で必要な周辺機能および FAA 処理を停止します。

Format

```
void R_{ConfigName}_LEDControl_Stop (void)
```

Parameters

なし

Return Values

なし

Properties

{ConfigName}_LEDControl.h にプロトタイプ宣言されています。

Description

この関数は、LED 制御で使用する下記周辺機能および FAA 処理を停止します。

・プログラマブル・ゲイン・アンプ

・A/D コンバータ

・D/A コンバータ

・コンパレータ

・16 ビット・タイマ KB

Example

```
/* Stop LED Control */
R_Config_FAA_Stop();
```

8.2.4 R_{ConfigName}_LEDControl_SetTargetLevelCh{n}

この関数は、LED 定電流制御における目標値を設定します。

Format

```
void R_{ConfigName}_LEDControl_ SetTargetLevelCh{n} (uint16_t target_level)
```

Parameters

target_level

LED 定電流制御 目標値 (有効範囲 : 0 - 4095)

Return Values

なし

Properties

{ConfigName}_LEDControl.h にプロトタイプ宣言されています。

Description

この関数は、LED 定電流制御における目標値(12bit)を設定します。

Example

```
/* Set LED target level */
uint16_t level = 150;
R_Config_FAA_LEDControl_SetTargetLevelCh1(level);
```

8.2.5 R_{ConfigName}_LEDControl_GetCurrentLevelCh{n}

この関数は、LED 定電流制御における現在値を取得します。

Format

```
uint16_t R_{ConfigName}_LEDControl_GetCurrentLevelCh{n} (void)
```

Parameters

なし

Return Values

LED 定電流制御 現在値 (有効範囲 : 0 - 4095)

Properties

{ConfigName}_LEDControl.h にプロトタイプ宣言されています。

Description

この関数は、LED 定電流制御における現在値(12bit)を取得します。

Example

```
/* Get LED current level */
uint16_t level;
level = R_Config_FAA_LEDControl_GetCurrentLevelCh1();

if (level >= 20)
{
  . . .
}
```

8.2.6 R_{ConfigName}_LEDControl_IsForceStopCh{n}

この関数は、LED 強制停止の発生有無を取得します。

Format

```
bool R_{ConfigName}_LEDControl_IsForceStopCh{n} (void)
```

Parameters

なし

Return Values

強制停止状態 (true : 発生有 / false : 発生無)

Properties

{ConfigName}_LEDControl.h にプロトタイプ宣言されています。

Description

この関数は、LED 過電流検出による強制停止の発生有無を取得します。

Example

```
/* Get LED force stop status */
bool is_force_stop;
is_force_stop = R_Config_FAA_LEDControl_IsForceStopCh1();

if (is_force_stop == true)
{
  . . .
}
```

8.2.7 R_{ConfigName}_LEDControl_ClearForceStopCh{n}

この関数は、LED 強制停止状態をクリアします。

Format

```
void R_{ConfigName}_LEDControl_ClearForceStopCh{n} (void)
```

Parameters

なし

Return Values

なし

Properties

{ConfigName}_LEDControl.h にプロトタイプ宣言されています。

Description

この関数は、LED 過電流検出による強制停止状態をクリアします。

Example

```
/* Get LED current level */
uint16_t level;
level = R_Config_FAA_LEDControl_GetCurrentLevelCh1();

if (level <= 15)
{
  R_Config_FAA_LEDControl_ClearForceStopCh1();
}
```

8.2.8 R_{ConfigName}_LEDControl_RequestADC

この関数は、指定されたアナログチャネルの AD 変換をリクエストします。

Format

```
void R_{ConfigName}_LEDControl_RequestADC (e_faa_ad_channel_t channel)
```

Parameters

channel

変換対象となるアナログチャネル

Return Values

なし

Properties

{ConfigName}_LEDControl.h にプロトタイプ宣言されています。

Description

この関数は、指定されたアナログチャネルの AD 変換を FAA 機能にリクエストします。

Example

```
/* Request AD conversion */
R_Config_FAA_LEDControl_RequestADC(ADCHANNEL2);
```

Special Notes:

スマート・コンフィグレータにて「ADC for user application」= 「Unused」を設定した場合、本関数は提供されません。

8.2.9 R_{ConfigName}_LEDControl_GetAD

この関数は、AD 変換結果を取得します。

Format

```
e_faa_result_adc_t R_{ConfigName}_LEDControl_GetAD (uint16_t * const buffer)
```

Parameters

buffer

AD 変換結果格納バッファへのポインタ

※本関数の戻り値が FAA_ADC_COMPLETED(AD 変換完了)の時のみ有効

Return Values

FAA_ADC_COMPLETED	AD 変換完了
FAA_ADC_FAILED	AD 変換失敗
FAA_ADC_NOTCOMPLETED	AD 変換未完了

Properties

{ConfigName}_LEDControl.h にプロトタイプ宣言されています。

Description

この関数は、R_{ConfigName}_LEDControl_RequestADC()によってリクエストされた AD 変換の結果を取得します。

Example

```
uint16_t buff;
e_faa_result_adc_t adc_result;

/* Request AD conversion */
R_Config_FAA_LEDControl_RequestADC(ADCHANNEL2);
/* Wait for completion of AD conversion by FAA */
do
{
  adc_result = R_Config_FAA_LEDControl_GetAD(&buff);
}
while (adc_result == FAA_ADC_NOTCOMPLETED);
```

Special Notes:

スマート・コンフィグレータにて「ADC for user application」= 「Unused」を設定した場合、本関数は提供されません。

9. 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://japan.renesas.com/

お問合せ先
http://japan.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

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Sep. 1st, 23	—	初版発行

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策
   CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。
2. 電源投入時の処置
   電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. 電源オフ時における入力信号
   当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。
4. 未使用端子の処理
   未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。
5. クロックについて
   リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
6. 入力端子の印加波形
   入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、V_{IL}（Max.）から V_{IH}（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、V_{IL}（Max.）から V_{IH}（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。
7. リザーブアドレス（予約領域）のアクセス禁止
   リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
8. 製品間の相違について
   型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。回路、ソフトウェアおよびこれらに関連する情報を使用する場合、お客様の責任において、お客様の機器・システムを設計ください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品または本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
6. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準：　コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等
   当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因しまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
10. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
12. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
13. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
14. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev.5.0-1　2020.10)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。

© 2023 Renesas Electronics Corporation. All rights reserved.