RL78/G24

RL78/G24 による Interleaved CrM PFC and LLC 制御 (FAA ソフトウェア編)

要旨

本アプリケーションノートでは Flexible Application Accelerator (FAA)を用いたデジタル電源コンバータ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

動作確認ボード

RL78/G24 Interleaved PFC + LLC Board 400W Kit (RTK0EL0006D00000BJ)

注意：サンプルアプリケーションを他のマイコンへ適用する場合、そのマイコンの仕様に合わせて変更し、十分評価してください。

関連ドキュメント

- RL78/G24 による Interleaved CrM PFC and LLC 制御 ハードウェア&ソフト基礎編 (R01AN8175)
- RL78/G24 による Interleaved CrM PFC and LLC 制御 CPU ソフトウェア編 (R01AN8177)
- 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)
- 統合開発環境 e2 studio ユーザーズマニュアルクイックスタートガイド(R20UT5293)
- RL78 スマート・コンフィグレータ ユーザーガイド：CS+編 (R20AN0580)
- RL78 スマート・コンフィグレータ ユーザーガイド：IAR 編 (R20AN0581)
- RL78 スマート・コンフィグレータ ユーザーガイド：e2 studio 編(R20AN0579)

目次

1.　デジタル電源コンバータ制御ライブラリの特徴 6
1.1　PFC + LLC コンバータ制御 6
1.2　Flexible Application Accelerator (FAA) 6

2.　デジタル電源コンバータ制御ライブラリを使用可能なツール 7

3.　使用リソース 8
3.1　周辺機能 8
3.2　周辺機能に関する制約事項 8
3.3　ROM / RAM サイズ 9

4.　デジタル電源コンバータ制御ライブラリの利用手順 10
4.1　開発環境 10
4.1.1　CS+ for CC 10
4.1.2　IAR Embedded Workbench for Renesas RL78 10
4.1.3　e2 studio 10
4.2　スマート・コンフィグレータ 11
4.3　デジタル電源コンバータ制御ライブラリ 11
4.3.1　スマート・コンフィグレータの起動 11
4.3.1.1　CS+ 11
4.3.1.2　IAR 11
4.3.1.3　e2 studio 11
4.3.2　クロック設定 12
4.3.3　FAA コンポーネントの追加 13
4.3.4　FAA モジュールのダウンロード 14
4.3.5　FAA モジュールのコンフィグレーション 15
4.3.6　コード生成 15

5.　ライブラリ構成 16
5.1　アーキテクチャ 16
5.2　フォルダ構成 17

6.　ライブラリ制御仕様 18
6.1　クロック動作モード 18
6.1.1　Low-Speed Mode 18
6.1.2　High-Speed Mode 18
6.2　PFC コンバータ制御 19
6.2.1　固定 On 幅出力制御 19
6.2.2　フィードバック制御 21
6.2.2.1　出力電圧のモニタリング 21
6.2.2.2　PI (比例積分) 制御 22
6.2.3　シングル、インターリーブモードの切り替え 23
6.2.3.1　フェード処理 23
6.2.3.2　出力エネルギー補正 24
6.2.4　最大周波数リミット制御 25

6.2.5　保護機能 25
6.2.5.1　D-OVP (ダイナミック過電圧保護) 25
6.2.5.2　OVP (過電圧保護) 25
6.2.5.3　OCP (過電流保護) 25
6.3　LLC コンバータ制御 26
6.3.1　フィードバック制御 26
6.3.1.1　出力電圧のモニタリング 26
6.3.1.2　ソフトスタート制御 28
6.3.1.3　PI (比例積分) 制御 29
6.3.2　保護機能 30
6.3.2.1　D-OVP (ダイナミック過電圧保護) 30
6.3.2.2　OCP (過電流保護) 30
6.4　ユーザーアプリケーション向け A/D 変換制御 31
6.4.1　Low-Speed Mode 時の A/D 変換制御 31
6.4.2　High-Speed Mode 時の A/D 変換制御 31
6.5　ユーザーアプリケーション向けデータ読み出し機能 31
6.6　制御フロー 32
6.6.1　フィードバック制御シーケンス 32
6.6.2　Task 処理フロー (FAA プログラム) 33
6.6.3　A/D 変換完了割り込み処理フロー (FAA プログラム) 34

7.　コンフィグレーション仕様 35
7.1　基本設定項目 35
7.2　PFC 設定項目 36
7.3　LLC 設定項目 38
7.4　ユーザーアプリケーション設定項目 40

8.　API 情報 41
8.1　API Typedef 定義 41
8.1.1　e_faa_peripheral_t 41
8.1.2　e_faa_clock_mode_t 41
8.1.3　e_faa_pfc_status_t 41
8.1.4　e_faa_pfc_output_t 42
8.1.5　e_faa_llc_status_t 42
8.1.6　e_faa_ad_channel_t 42
8.1.7　e_faa_result_adc_t 43
8.1.8　e_faa_request_id_t 43
8.1.9　faa_pfc_callback_t 43
8.1.10　faa_llc_callback_t 44
8.2　API 関数仕様 45
8.2.1　R_{ConfigName}_DPConv_Create 45
8.2.2　R_{ConfigName}_DPConv_PowerOn 46
8.2.3　R_{ConfigName}_DPConv_PowerOff 47
8.2.4　R_{ConfigName}_DPConv_ChangeClockMode 48
8.2.5　R_{ConfigName}_DPConv_PFC_Init 49
8.2.6　R_{ConfigName}_DPConv_PFC_StartFixedOnWidth 50
8.2.7　R_{ConfigName}_DPConv_PFC_Suspend 51

8.2.8　R_{ConfigName}_DPConv_PFC_Resume 52
8.2.9　R_{ConfigName}_DPConv_PFC_StartFeedbackControl 53
8.2.10　R_{ConfigName}_DPConv_PFC_Stop 54
8.2.11　R_{ConfigName}_DPConv_PFC_TaskForFeedbackControl 55
8.2.12　R_{ConfigName}_DPConv_PFC_GetStatus 56
8.2.13　R_{ConfigName}_DPConv_PFC_ChangeOutputMode 57
8.2.14　R_{ConfigName}_DPConv_PFC_SetMaxFreqLimit 58
8.2.15　R_{ConfigName}_DPConv_PFC_GetMaxFreqLimit 59
8.2.16　R_{ConfigName}_DPConv_PFC_SetFz 60
8.2.17　R_{ConfigName}_DPConv_PFC_SetKp 61
8.2.18　R_{ConfigName}_DPConv_PFC_SetAdjustParamForInterleaved 62
8.2.19　R_{ConfigName}_DPConv_PFC_GetOnWidthCount 63
8.2.20　R_{ConfigName}_DPConv_LLCn_Init 64
8.2.21　R_{ConfigName}_DPConv_LLCn_StartFeedbackControl 65
8.2.22　R_{ConfigName}_DPConv_LLCn_Stop 66
8.2.23　R_{ConfigName}_DPConv_LLCn_EnableOCP 67
8.2.24　R_{ConfigName}_DPConv_LLCn_DisableOCP 68
8.2.25　R_{ConfigName}_DPConv_LLCn_GetStatus 69
8.2.26　R_{ConfigName}_DPConv_GetAdValue 70
8.2.27　R_{ConfigName}_DPConv_RequestAdValue 71
8.2.28　R_{ConfigName}_DPConv_GetRequestedAdValue 72
8.2.29　R_{ConfigName}_DPConv_RequestData 73
8.2.30　R_{ConfigName}_DPConv_GetRequestedData 74

9.　サンプル・プログラム 75
9.1　ハードウェア構成 75
9.2　動作確認条件 75
9.3　ソフトウェア仕様 76
9.3.1　ソフトウェア制御 76
9.3.2　ソフトウェア構成 76
9.3.3　フォルダ構成 77
9.3.4　オプション・バイト設定一覧 78
9.3.5　コンフィグレーション設定 78
9.3.6　環境構築方法 84
9.3.6.1　CC-RL 版 (CS+ for CC) 84
9.3.6.2　CC-RL 版 (e2 studio) 85
9.3.6.3　IAR 版 86
9.3.7　API 関数 87

10.　QE for Lighting & Power を用いた動作 88
10.1　QE for Lighting & Power のダウンロード 88
10.2　ワークスペースの準備 88
10.2.1　ボードの選択 88
10.2.2　ワークスペースの選択 88
10.3　照明通信の設定 89
10.4　電源制御の設定 90
10.4.1　電源制御方式の設定 91

10.4.1.1 Power Configuration ビュー 91
10.4.2 電源制御シミュレータ 97
10.4.2.1 電源制御シミュレータダイアログ 98
10.4.3 電源制御の評価 102
10.4.3.1 ビルドの設定 103
10.4.3.2 評価プログラムの準備 103
10.4.3.3 ボードにダウンロード 103
10.4.3.4 ボード接続 104
10.4.3.5 電源制御の評価 104
10.5 コード生成とプログラム 107
10.5.1 ソースコードの生成 107
10.5.2 ボードにダウンロード 107
10.6 PFC 制御パラメータ A1、A2 に関する注意事項 108
10.6.1 Kp と fz の算出方法 108
10.6.2 Kp の fz の反映方法 108

11. 注意事項 109

12. 参考ドキュメント 109

1. デジタル電源コンバータ制御ライブラリの特徴

本章では、デジタル電源コンバータ制御ライブラリの特徴について説明します。

1.1 PFC + LLC コンバータ制御

本ライブラリでは、RL78/G24 マイクロコントローラのタイマ KB PWM 出力機能を使用した PFC(力率改善)コンバータ制御および LLC コンバータ制御を行います。各コンバータの定電圧制御は 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、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)

本ライブラリではデジタル電源コンバータ制御を実現するために、Flexible Application Accelerator（FAA）を活用しています。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。これにより、CPU リソースを占有することなく、高速で効率的な電圧制御処理を行うことが可能です。

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 (R01UH0961) をご覧ください。

2. デジタル電源コンバータ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 **4 デジタル電源コンバータ制御ライブラリの利用手順**を参照してください。

表 2.1　デジタル電源コンバータ制御ライブラリを使用可能なツール

環境	項目	内容
CS+ for CC	統合開発環境 (IDE)	ルネサス エレクトロニクス製 CS+ for CC V8.14.00 以上
	コンパイラ	CC-RL V1.15.01 以上
IAR	統合開発環境 (IDE)	IAR Systems 製 IAR Embedded Workbench for Renesas RL78 V5.10.3 以上
	コンパイラ	IAR C/C++ Compiler for Renesas RL78 V5.10.3.2716 以上
e2 studio	統合開発環境 (IDE)	ルネサス エレクトロニクス製 e2 studio 2025-10 以上
	コンパイラ	CC-RL V1.15.01 以上
共通	コンフィグレータ(SC)	Renesas Smart Configurator for RL78 V1.15.0 以上

3. 使用リソース

3.1 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
- A/D コンバータ
- D/A コンバータ
- コンパレータ
- タイマ・アレイ・ユニット
- 16 ビット・タイマ KB
- イベント・リンク・コントローラ

3.2 周辺機能に関する制約事項

本ライブラリは使用する際は、以下の点にご注意ください。

- イベント・リンク・コントローラにおける ELCAD3 イベントは本ライブラリで使用されます。
  イベント出力先選択レジスタ 33（ELSER33 レジスタ）の設定値を変更しないでください。
- タイマ・アレイ・ユニット 0 チャネル 1 は本ライブラリで使用されます。
  動作クロックとして CK00 を使用するため、CK00 の分周比を変更しないでください。
  （TPS0 レジスタの 3-0 ビットの設定値を変更しないでください）
- A/D コンバータを使用する場合は本ライブラリが提供する A/D 変換制御機能を使用してください。
  詳細は **6.4 ユーザーアプリケーション向け A/D 変換制御**をご覧ください。

3.3　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

CC-RL

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 3.1　ROM / RAM サイズ

項目	ROM	RAM	FAACODE	FAADATA
PFC	5,309 [byte]	12 [byte]	1,432 [byte]	556 [byte]
LLC1	3,630 [byte]	6 [byte]	1,080 [byte]	440 [byte]
LLC1 + LLC2	4,775 [byte]	11 [byte]	1,628 [byte]	616 [byte]
PFC + LLC1	5,856 [byte]	17 [byte]	2,016 [byte]	768 [byte]
PFC + LLC1 + LLC2	6,997 [byte]	22 [byte]	2,564 [byte]	944 [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増減するため、上表では最大値を記載しています。

ROM サイズには FAACODER セクションおよび FAADATAR セクションを含みます。

4.　デジタル電源コンバータ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。

4.1　開発環境

IDE 別に開発環境の構築方法を記載します。

4.1.1　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)

4.1.2　IAR Embedded Workbench for Renesas RL78

インストールについては下記動画を参照ください。

IAR Embedded Workbench for RL78 でのスマート・コンフィグレータの使い方(1/2) - インストール

4.1.3　e2 studio

e2 studio 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

e2 studio 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。

統合開発環境 e2 studio ユーザーズマニュアルクイックスタートガイド(R20UT5293)

4.2　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「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。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開発環境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。

4.3　デジタル電源コンバータ制御ライブラリ

スマート・コンフィグレータ上で、デジタル電源コンバータ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)
- RL78 スマート・コンフィグレータ ユーザーガイド：IAR 編 (R20AN0581)
- RL78 スマート・コンフィグレータ ユーザーガイド：e2 studio 編(R20AN0579)

4.3.1　スマート・コンフィグレータの起動

4.3.1.1　CS+

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+ のプロジェクトツリー内にある「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て起動してください。

4.3.1.2　IAR

スマート・コンフィグレータを起動し、新規プロジェクトの作成または既存プロジェクトの .scfg ファイルを開いてください。

4.3.1.3　e2 studio

新規プロジェクトの作成または既存プロジェクトのインポートを行い、e2 studio のプロジェクトワークスペース内にある .scfg ファイルをダブルクリックして起動してください。

4.3.2　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。このクロック設定以外を設定した場合、ユーザー側で設定した周辺機能が正常に動作しなくなる可能性があります。

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

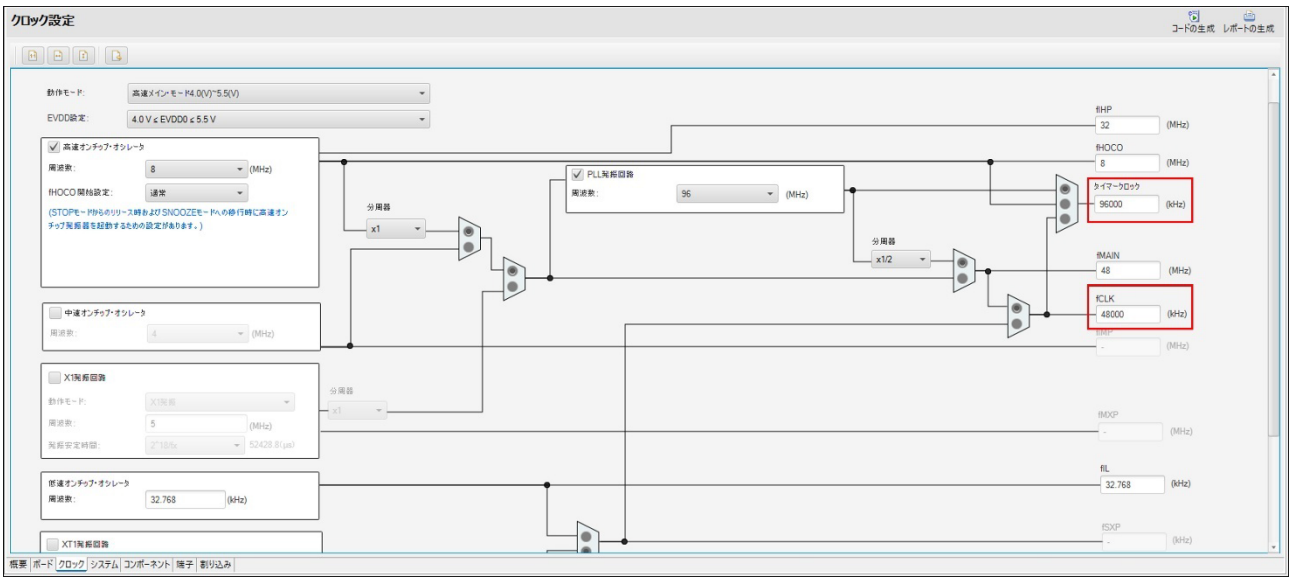

図 4.1　クロック設定

4.3.3　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。

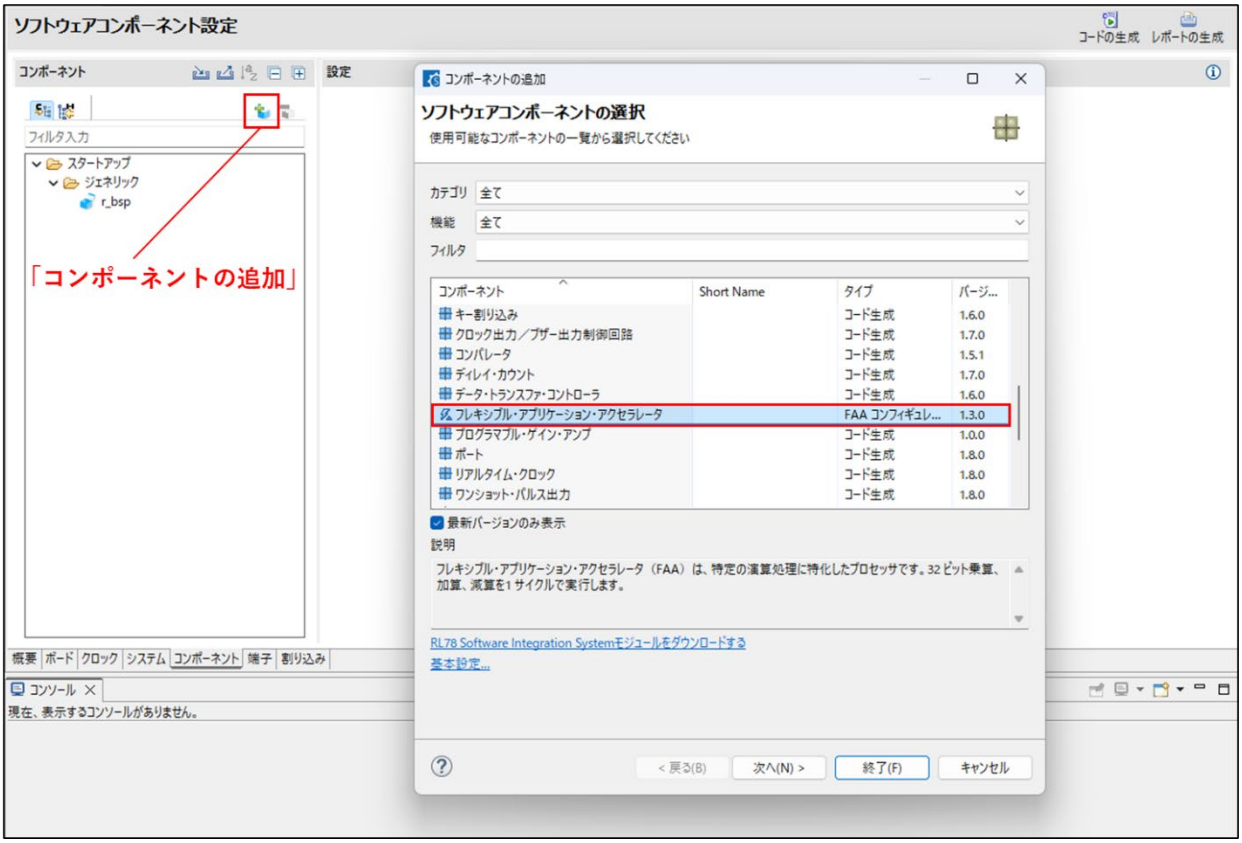

図 4.2　FAA コンポーネントの追加

4.3.4　FAA モジュールのダウンロード

画面上に表示されている「 」アイコン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。「Digital Power Converter」を選択してダウンロードしてください。

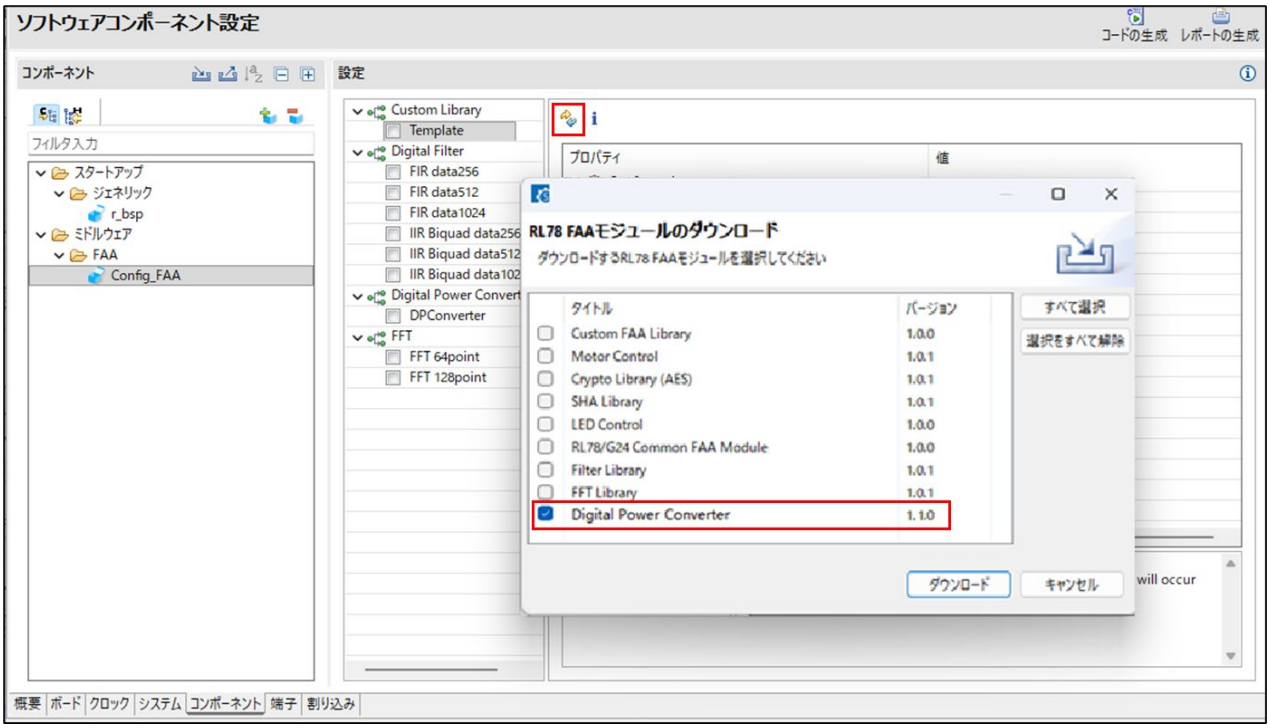

図 4.3　FAA モジュールのダウンロード

4.3.5　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「Digital Power Converter」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。

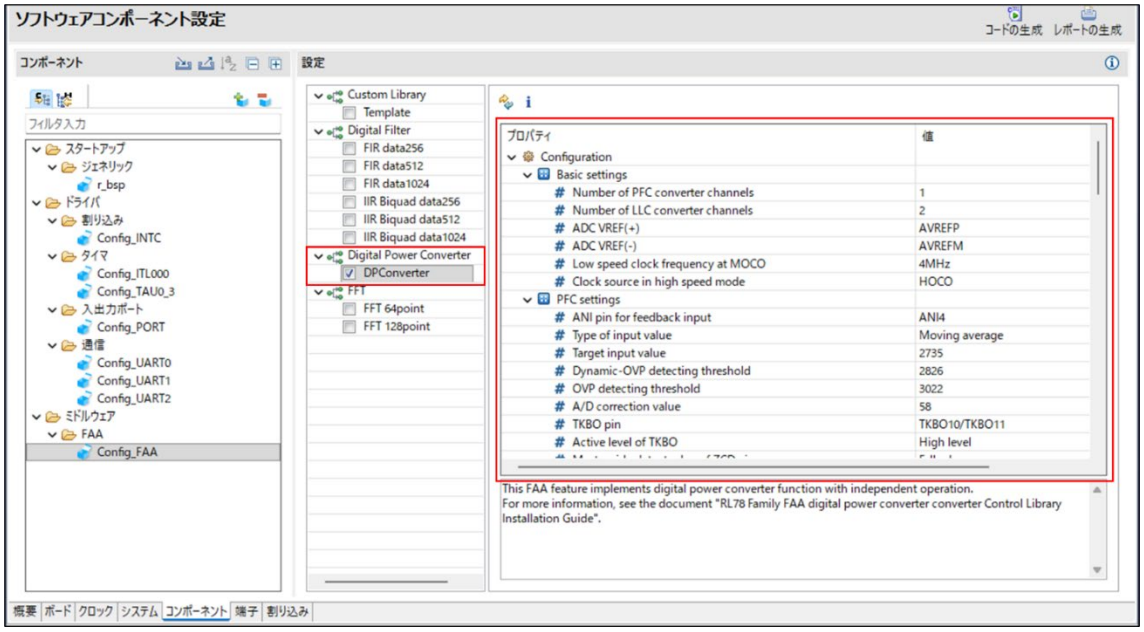

図 4.4　FAA モジュールのコンフィグレーション

4.3.6　コード生成

コンフィグレーション後、「コード生成」ボタンを押下することで、デジタル電源コンバータ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開発環境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. ライブラリ構成

本ライブラリの構成について説明します。

5.1 アーキテクチャ

本ライブラリの基本的なアーキテクチャに関して説明します。

本ライブラリは CPU プログラム部と FAA プログラム部にて構成され、それぞれのプログラム部からアクセス可能な共有メモリがあります。CPU プログラムと FAA プログラムは共有メモリを介して協調制御を行います。

ユーザーアプリケーションから API 関数を呼び出すことにより、本ライブラリの CPU プログラムは共有メモリに FAA プログラムに対するリクエストを設定します。FAA プログラムは定期的に共有メモリをポーリングし、取得したリクエストに対する処理を実行します。その後、FAA プログラムよりステータス情報が共有メモリに設定されます。このような処理の流れにより、API 関数のコールから 1～12us の範囲でリクエストされた処理が完了します。

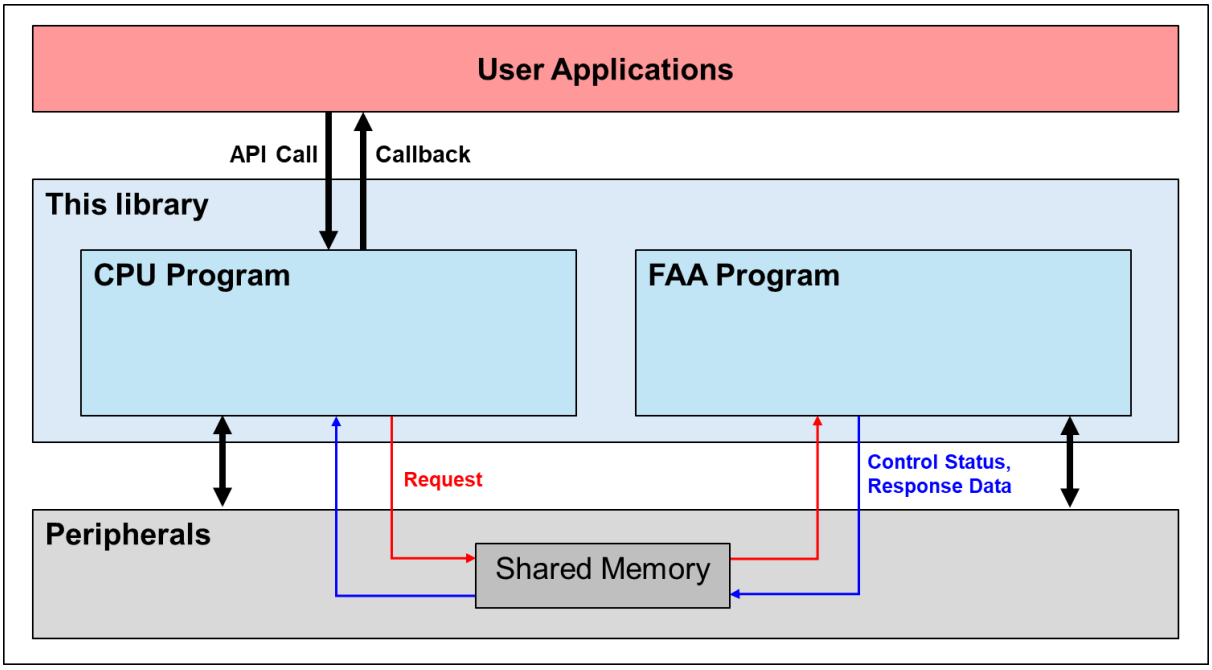

図 5.1　ライブラリアーキテクチャ

5.2　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 5.1　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}_DPConv.c	デジタル電源コンバータ制御機能のソースファイル
{ConfigName}_DPConv.h	デジタル電源コンバータ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

6. ライブラリ制御仕様

ライブラリの制御仕様について説明します。

6.1 クロック動作モード

本ライブラリには 2 種類のクロック動作モードがあります。動作モードは下記 API 関数にて切り替えが可能です。

API 関数： R_{ConfigName}_DPConv_ChangeClockMode

6.1.1 Low-Speed Mode

このモードでは、低消費電力での動作を実現するため、中速オンチップ・オシレータ(MOCO)を使用した低速動作を行います。MOCO クロック周波数は 1 / 2 / 4MHz より選択可能です。また、FAA プログラム動作を停止し、CPU プログラムによる最小限の動作機能を提供します。

このモード中にユーザーアプリケーションで周辺機能を使用する場合は、ベースクロックが MOCO となることを考慮し、適切な設定を行ってください。

6.1.2 High-Speed Mode

このモードでは、各コンバータのフィードバック制御を行なうため、システムの動作速度が最大化されます。8MHz の高速オンチップ・オシレータ(HOCO)と PLL 回路により、f_{CLK}=48MHz にて CPU や FAA プログラムが動作します。また、コンバータ制御を実現するために使用される 16 ビット・タイマ KB は 96MHz で動作します。

6.2　PFC コンバータ制御

PFC コンバータ制御は、臨界導通モード(CRM)に対応しており、シングル出力およびインターリーブ出力をサポートします。以下に、制御機能の詳細について記載します。

6.2.1　固定 On 幅出力制御

この機能は Low-Speed Mode 時のみ有効で、On 幅を固定した状態で PWM 出力を行います。このとき臨界導通モードとして動作するため、ZCD 検出で自動的にパルス出力をリスタートします。後述するフィードバック制御とは異なり、短周期での演算処理を必要としないため、低速クロック動作で低消費電力を実現する動作となります。PWM 出力周期と On 幅は下記のように決まります。

周期：　コンフィグレーションパラメータ ”Longest period [us]”
(表 7.3　スマート・コンフィグレータ設定項目一覧(PFC settings) (2/2) 参照)

On 幅：　API 関数 ”R_{ConfigName}_DPConv_PFC_StartFixedOnWidth” の引数値

例えばシステムが省電力状態を必要とする場面において、PFC の出力電圧レベルが一定の閾値まで低下した際に本制御による出力を開始し、出力電圧レベルが再び一定の閾値まで上昇した際に本制御を停止させるような間欠的な動作を行うことにより、省消費電力化を実現することができます。

この動作例のフローチャートを図 6.1 に記載します。

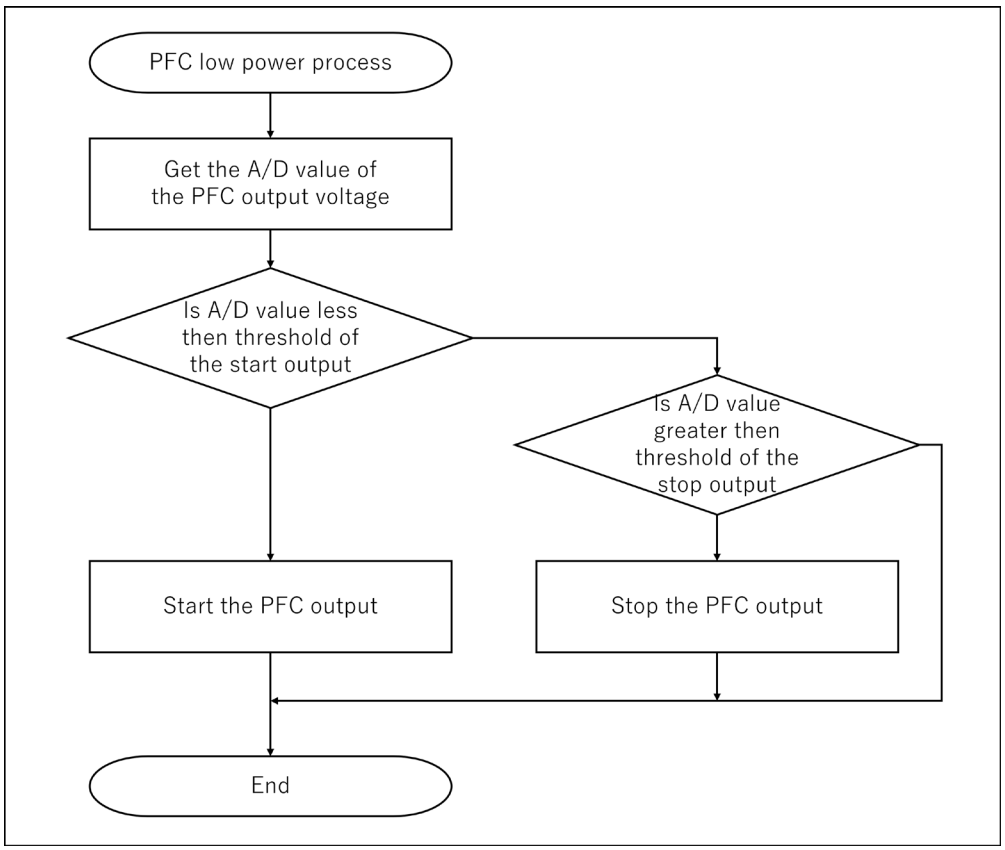

図 6.1 PFC 省電力制御例フローチャート

また、このフローチャートに基づいて動作させる際の動作例を図 6.2 に示します。

定期的に PFC 出力電圧の A/D 値を取得し、出力開始閾値(TH_{start})を下回ると API 関数

”R_{ConfigName}_DPConv_PFC_StartFixedOnWidth”を呼び出し、引数に指定した On 幅の PWM 出力を開始させます。PWM 出力の Off 毎にインダクタに蓄積されたエネルギーによる電流が流れるため、PFC 出力電圧が昇圧されます。このとき PWM 出力は臨界導通モード(CRM)で動作するため、インダクタ電流が 0A になったことで ZCD(Zero Current Detection)波形の立下りで PWM 出力がリスタートします。

PWM 出力による昇圧した結果、PFC 出力電圧が出力停止閾値(TH_{stop})を上回ると API 関数

“R_{ConfigName}_DPConv_PFC_Stop”を呼び出し、PFC の PWM 出力を停止させます。

この一連の動作を省電力制御が必要な期間の間継続します。

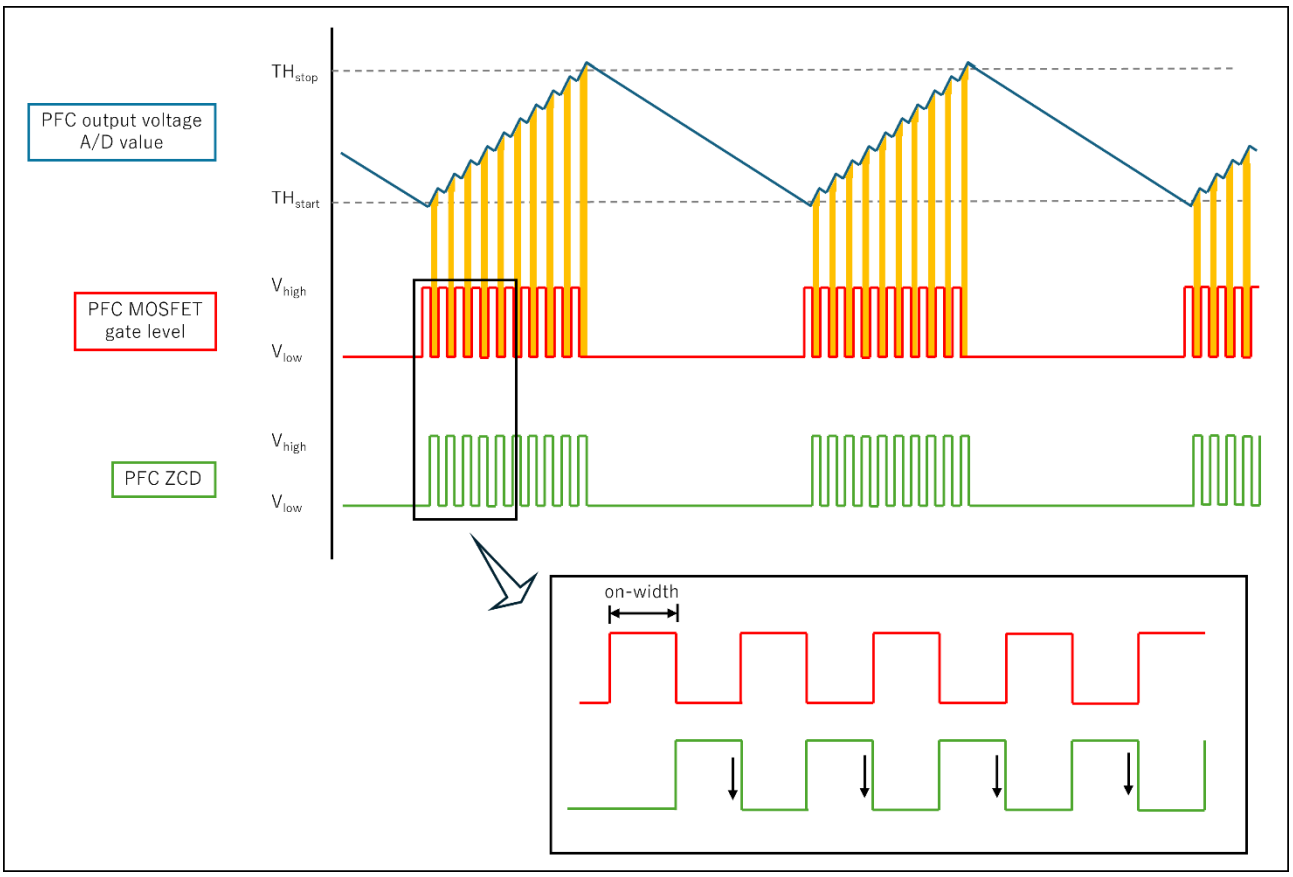

図 6.2 PFC 省電力制御動作例

6.2.2　フィードバック制御

この機能は High-Speed Mode 時のみ有効です。PFC の出力電圧をモニタリングし、定電圧を維持するためにフィードバック制御 (PI 制御) を行います。

6.2.2.1　出力電圧のモニタリング

フィードバック制御において、PFC 出力電圧は、下記のように A/D 値の移動平均により算出されます。

移動平均法：

12.5us 間隔で A/D 変換値をサンプリングし、直近 N 回分のデータの平均値を計算します。

(N = PFC フィードバック制御周期 ÷ 12.5us)

6.2.2.2　PI (比例積分) 制御

PFC コンバータにおける定電圧制御は 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 On 幅

D（n－1)：　前回の PWM 出力 On 幅

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

係数 A1 および A2 は、ゼロ・ポイント周波数(f_Z) 、比例定数(K_P) 、フィードバック制御周期(T)を基に計算されます。各パラメータの設定については**表 7.3　スマート・コンフィグレータ設定項目一覧(PFC settings) (2/2)**を参照してください。

また、ゼロ・ポイント周波数(f_Z) 、比例定数(K_P) は下記 API 関数を用いて動的に変更することが可能です。

API 関数：　R_{ConfigName}_DPConv_PFC_SetFz
R_{ConfigName}_DPConv_PFC_SetKp

6.2.3　シングル、インターリーブモードの切り替え

API 関数 ”R_{ConfigName}_DPConv_PFC_ChangeOutputMode” にて動的に PFC 出力モードを切り替えることができます。さらに、オプションにより、モード切り替え時のフェード処理の有無や、切り替え時に出力エネルギーを一定に保つための補正処理を行うことも可能です。

6.2.3.1　フェード処理

API 関数 ” R_{ConfigName}_DPConv_PFC_ChangeOutputMode” コール時の引数にて、フェード処理の有無を指定することができます。

フェード処理なしで出力モードを切り替える場合、次のフィードバックタイミングでシングル(マスタ 100%)とインターリーブ(マスタ 51%、スレーブ 49%)[注]の出力モードが切り替わります。

フェード処理ありで出力モードを切り替える場合、フィードバック制御周期毎に総 On 幅の割合を 1%ずつ段階的に変更していきます。下図はシングルからインターリーブへ切り替える際のフェード処理を図示したものです。総 On 幅は変わらずにスレーブの On 幅割合を上昇させ、マスタの On 幅割合を減少させるようなフェード処理が適用されます。

また、下図ではフェード時の On 幅の割合を説明するため便宜的に総 On 幅が変わらない表現としていますが、実際はフィードバック制御周期毎に総 On 幅が再計算され、計算された結果を適切な割合でマスタとスレーブで分配する形となります。

注：マスタ On 幅<=スレーブ On 幅になると回路、周辺素子、入出力負荷等の条件によって ZCD 信号が規定時間に入らず、タイマ KB インターリーブ条件 No.8 が発生(パルス抜け)する可能性があります。

これを避けるため、本プログラムではスレーブ側の ZCD 信号入力がやや早く入るよう、マスタの ON 幅とスレーブの ON 幅に差をつけています。

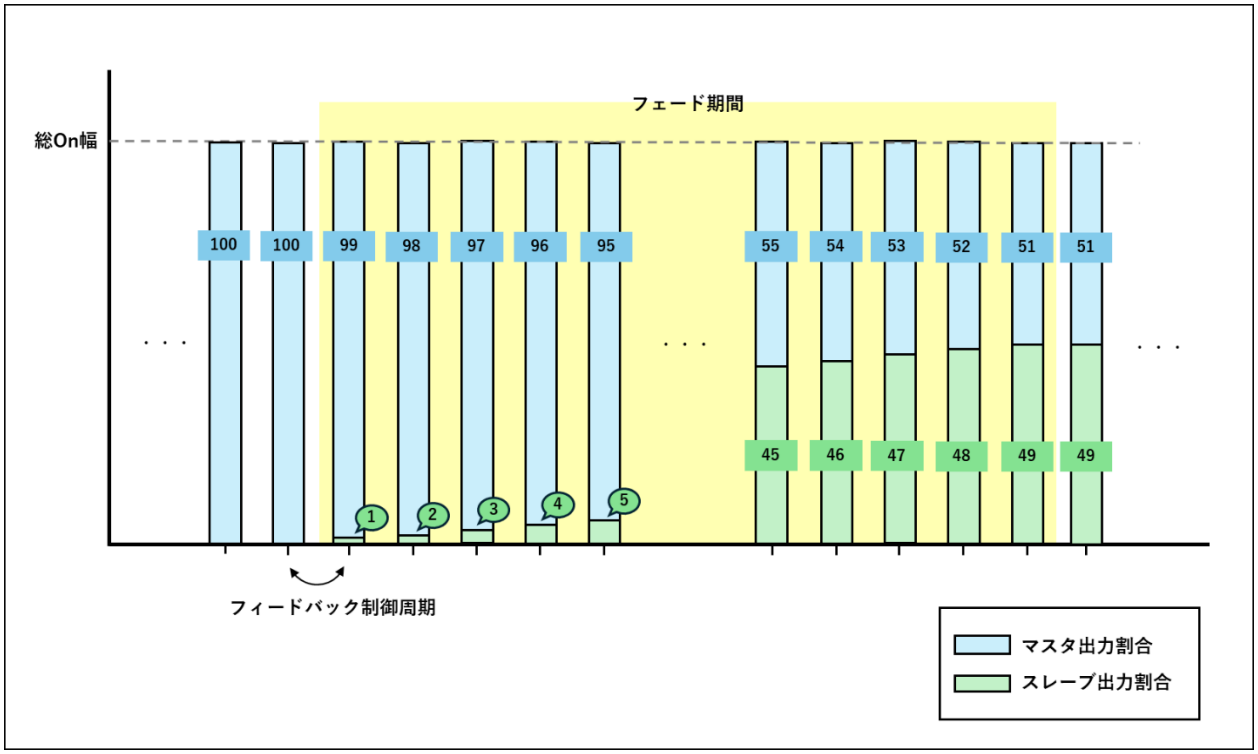

図 6.3　フェード処理

6.2.3.2　出力エネルギー補正

出力モード切り替えを行う際、マスタおよびスレーブの On 幅合計値が同一となるように調整が行われます。例えば、シングルモードからインターリーブモードへの切り替え時には、シングルモードでのマスタの On 幅を基準とし、この合計値が変更されないようにマスタおよびスレーブ間で On 幅が適切に分配されます。インターリーブモードからシングルモードへの切り替え時には、マスタおよびスレーブの On 幅合計値が、マスタ On 幅値に設定されます。

理論上、On 幅の総和が同一であれば出力エネルギーを一定に保つことができますが、実際にはスイッチングの立ち上がり／立ち下がりの差やマスタとスレーブの回路バラつきなどに起因する要素により、シングルモードとインターリーブモード間で出力エネルギーに差異が生じることがあります。

前述のフェード処理により、シングルモードとインターリーブモード間で生じる差異に対する調整が可能ですが、より安定的な切り替えを実現するために、補正処理を適用することができます。

補正処理は、以下の線形関数を用いて実施されます。この方法により、モード切り替え時に発生する出力エネルギーの変動が線形補正によって調整され、安定したモード切り替えが実現されます。

$$D_{interleave} = slope \cdot D_{single} + intercept$$

$D_{interleave}$：インターリーブモードにおける On 幅

D_{single}：シングルモードにおける On 幅

slope：補正式の傾き

intercept：補正式の切片

上記の線形補正式における slope、intercept は下記 API 関数にて任意の値に設定することが可能です。

デフォルトでは、slope = 1 および intercept = 0 が適応されており、この設定では実質的に補正が無効となっています。

API 関数： R_{ConfigName}_DPConv_PFC_SetAdjustParamForInterleaved

6.2.4　最大周波数リミット制御

RL78/G24 の 16 ビット・タイマ KB には、最大周波数リミット機能が搭載されています。この機能は、臨界導通モード (CRM) の PFC 制御において、過度に高い周波数でのスイッチングを抑制することを目的としています。設定した最大周波数より早いタイミングで ZCD（Zero Crossing Detection）検出トリガが発生した場合、タイマは設定した最大周波数までリスタートを保留することが可能です。この保留中、システムは電流不連続モード (DCM) で動作します。高い周波数でのスイッチングが抑制されるため、ノイズ発生とスイッチングロス軽減が期待できます。

本ライブラリでは、下記 API 関数を介して最大周波数リミット設定値を変更することが可能であり、システムの状況に応じて動的に設定を変更することができます。

API 関数： R_{ConfigName}_DPConv_PFC_SetMaxFreqLimit

6.2.5　保護機能

PFC コンバータ制御における保護機能について説明します。

6.2.5.1　D-OVP (ダイナミック過電圧保護)

PFC フィードバック制御中に、PFC 出力電圧が D-OVP 閾値を超過した場合、ソフトウェア制御により PFC 出力が一時的に停止されます。この動作は、出力電圧が再び D-OVP 閾値を下回るまで継続され、過電圧状態を防止します。

6.2.5.2　OVP (過電圧保護)

PFC フィードバック制御中に、PFC 出力電圧が OVP 閾値を超過した場合、ソフトウェア制御により PFC 出力が強制的に停止されます。この機能により、過電圧状態からシステムを保護し、回路および負荷への損傷を防止することができます。

また、OVP 検出時にコールバック関数を通じてユーザーアプリケーションに通知が行われます。

なお、本ライブラリで提供される LLC 機能とは独立しているため、PFC の OVP 検出に伴って LLC の動作を停止させたい場合はユーザー側で停止していただく必要があります。

6.2.5.3　OCP (過電流保護)

PFC フィードバック制御および固定 On 幅出力制御において、マスタおよびスレーブの MOSFET ソース電流値が電圧換算で OCP（過電流保護）閾値を超過した場合、RL78/G24 マイクロコントローラに内蔵された 16 ビット・タイマ KB の強制停止機能が作動し、PFC 出力が即座に強制停止します。この機能により、過電流状態からシステムを保護し、回路および負荷への損傷を防止することができます。

また、OCP 検出時にコールバック関数を通じてユーザーアプリケーションに通知が行われます。

なお、本ライブラリで提供される LLC 機能とは独立しているため、PFC の OCP 検出に伴って LLC の動作を停止させたい場合はユーザー側で停止していただく必要があります。

6.3　LLC コンバータ制御

本ライブラリでは最大 2 チャネルの LLC コンバータ制御機能を提供します。以下に、制御機能の詳細について記載します。

6.3.1　フィードバック制御

この機能は High-Speed Mode 時のみ有効です。LLC の出力電圧をモニタリングし、定電圧を維持するためにフィードバック制御 (PI 制御) を行います。

6.3.1.1　出力電圧のモニタリング - 特許出願中

フィードバック制御における LLC 出力電圧のモニタリングには、移動平均、2 値判定の 2 種類の方法があります。

1. 移動平均法

   12.5us 間隔で A/D 値のサンプリングを行い、直近 N 回分のデータの平均値を算出します。

   (N = LLC フィードバック制御周期 ÷ 12.5us)

   このモニタリング方法は、LLC 出力電圧を絶対的な電圧レベルで測定するような回路において有効です。

   例えば、LLC 回路における 2 次側の出力電圧を分圧し、2 次側に配置されたマイコンの ANI 端子に入力することで測定するような回路が該当します。

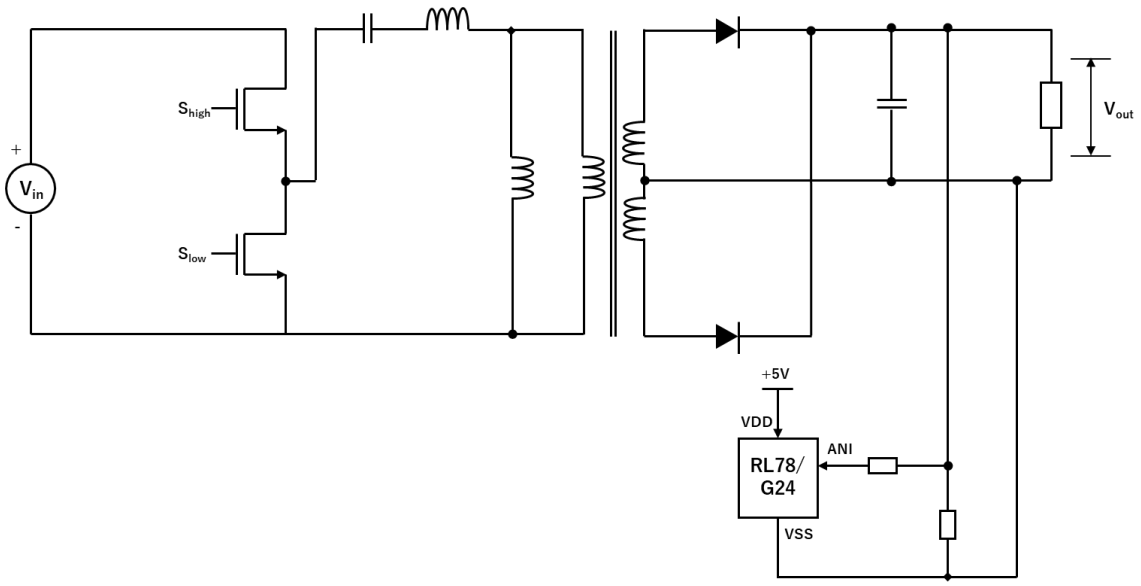

図 6.4　2 次側での LLC 出力電圧計測回路例

2.　2 値判定法

12.5us 間隔で A/D 値のサンプリングを行い、直近 N 回分のデータの内、目標電圧を上回った回数をカウントします。(N = LLC フィードバック制御周期 ÷ 12.5us)

このモニタリング方法は、LLC 出力レベルを相対的な電圧レベル(目標電圧レベルより上か下か)で判定するような回路において有効です。

例えば、LLC 回路における 2 次側の出力電圧レベルをフォトカプラを介して 1 次側に配置されたマイコンで測定するような回路構成です。

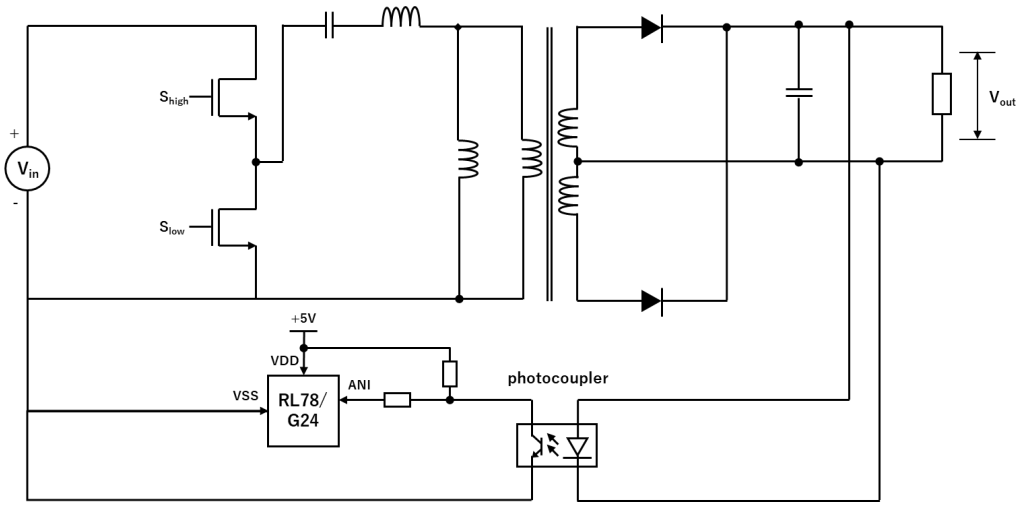

図 6.5　フォトカプラを使用した 1 次側での LLC 出力電圧簡易計測回路例

このような回路の場合、フォトカプラの出力電圧は下図のように基準電圧(目標電圧)付近で急激なレベル変化が発生します。そのため、通常では二次側の出力状態を A/D コンバータで測定するには工夫が必要です。

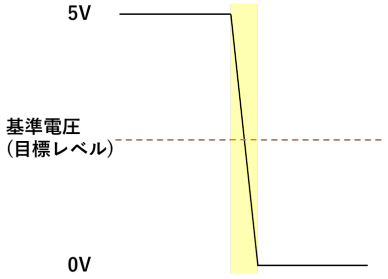

図 6.6　フォトカプラによる電圧測定

本モニタリング方法では一定の時間内で出力側の電圧レベルが基準電圧を上回った回数をカウントし、この結果をフィードバック制御における入力値として使用します。フィードバック制御では、基準電圧を上回った回数と下回った回数が均衡するように出力値を調整することにより、LLC 出力電圧を基準電圧に収束させます。

例えば 16 回の A/D 値のサンプリングを行う場合、フィードバック制御における目標値は 8 カウントとなります。

6.3.1.2　ソフトスタート制御

ソフトスタート制御は、LLC 出力周波数を段階的に低下させながらコンバータを起動することで、起動時の突入電流を抑制します。また、この制御により、スイッチング周波数の急激な変動が緩和され、可聴ノイズの発生を抑えることができます。

本制御は、API 関数 “R_{ConfigName}_DPConv_LLCn_StartFeedbackControl” を介してフィードバック制御リクエストが発生した時点で開始され、LLC 出力電圧が終了目標値に到達するか、または最大継続時間が経過した時点で終了します。制御期間中、LLC 出力周波数は 25us 間隔で更新されます。各パラメータは下記に基づいて決定されます。

開始時周波数：コンフィグレーションパラメータ ” Frequency at the start [kHz]”

終了時周波数：コンフィグレーションパラメータ ” Frequency at the end [kHz]”

最大継続時間：コンフィグレーションパラメータ ” Max duration [ms]”

終了目標値：　コンフィグレーションパラメータ ” Analog input value for terminate”

ステップ幅：　(開始時周波数 － 終了時周波数) ÷ (最大継続時間　÷ 25us)

各コンフィグレーションパラメータの詳細は**表 7.4　スマート・コンフィグレータ設定項目一覧 (LLCn setting) (1/2)**を参照してください。

6.3.1.3　PI (比例積分) 制御

LLC コンバータにおける定電圧制御は 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。

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

P（n)：　最新の PFM 出力周期

P（n−1)：　前回の PFM 出力周期

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

係数 A1 および A2 は、ゼロ・ポイント周波数(f_Z) 、比例定数(K_P) 、フィードバック制御周期(T)を基に計算されます。各パラメータの設定については**表 7.5　スマート・コンフィグレータ設定項目一覧(LLCn setting) (2/2)**を参照してください。

6.3.2　保護機能

LLC コンバータ制御における保護機能について説明します。

6.3.2.1　D-OVP (ダイナミック過電圧保護)

LLC フィードバック制御中に、LLC 出力周波数が D-OVP 閾値を超過した場合、ソフトウェア制御により LLC 出力が一時的に停止されます。この動作は、フィードバック処理において計算された LLC 出力周波数が D-OVP 閾値を下回るまで継続され、過電圧状態を防止します。

6.3.2.2　OCP (過電流保護)

LLC 共振電流値が電圧換算で OCP（過電流保護）閾値を超過した場合、RL78/G24 マイクロコントローラに内蔵された 16 ビット・タイマ KB の強制停止機能が作動し、LLC 出力が即座に強制停止します。この機能により、過電流状態からシステムを保護し、回路および負荷への損傷を防止することができます。

OCP による監視は、フィードバック制御開始時に有効化され、制御停止時に無効化されます。さらに、下記 API 関数を通じて任意のタイミングで有効化/無効化が可能です。

API 関数：　R_{ConfigName}_DPConv_LLCn_EnableOCP

　　R_{ConfigName}_DPConv_LLCn_DisableOCP

また、OCP 検出時にコールバック関数を通じてユーザーアプリケーションに通知が行われます。

なお、本ライブラリで提供される PFC 機能とは独立しているため、LLC の OCP 検出に伴って PFC の動作を停止させたい場合はユーザー側で停止していただく必要があります。

6.4　ユーザーアプリケーション向け A/D 変換制御

本ライブラリでは、以下の A/D 変換機能を提供します。CPU 側ユーザーアプリケーションにおいて A/D 変換を必要とする場合は、こちらの変換機能を使用してください。

6.4.1　Low-Speed Mode 時の A/D 変換制御

Low-Speed Mode 時には、下記 API 関数を通じて A/D 変換処理を要求してください。この関数では、引数として指定された回数分の A/D 値を取得し、その平均値を取得することが可能です。

API 関数：　R_{ConfigName}_DPConv_GetAdValue

6.4.2　High-Speed Mode 時の A/D 変換制御

High-Speed Mode 時は下記 API 関数にて A/D 変換リクエストの発行および変換結果の取得を行ってください。

API 関数：　R_{ConfigName}_DPConv_RequestAdValue

R_{ConfigName}_DPConv_GetRequestedAdValue

6.5　ユーザーアプリケーション向けデータ読み出し機能

本ライブラリでは、ライブラリ内部のデータ読み出し機能を提供します。CPU 側ユーザーアプリケーションにおいて FAA データメモリの値を必要とする場合は、下記 API 関数にてデータ読み出しリクエストの発行および読み出し結果の取得を行うことができます。

API 関数：　R_{ConfigName}_DPConv_RequestData

R_{ConfigName}_DPConv_GetRequestedData

6.6　制御フロー

6.6.1　フィードバック制御シーケンス

本ライブラリは、API 関数を通じてユーザーアプリケーションからフィードバック制御の開始要求やステータス情報の取得要求を受け付けます。ライブラリ内部では、CPU プログラムと FAA プログラムが SHDMEM（データ共有メモリ）を介して協調して制御を行います。

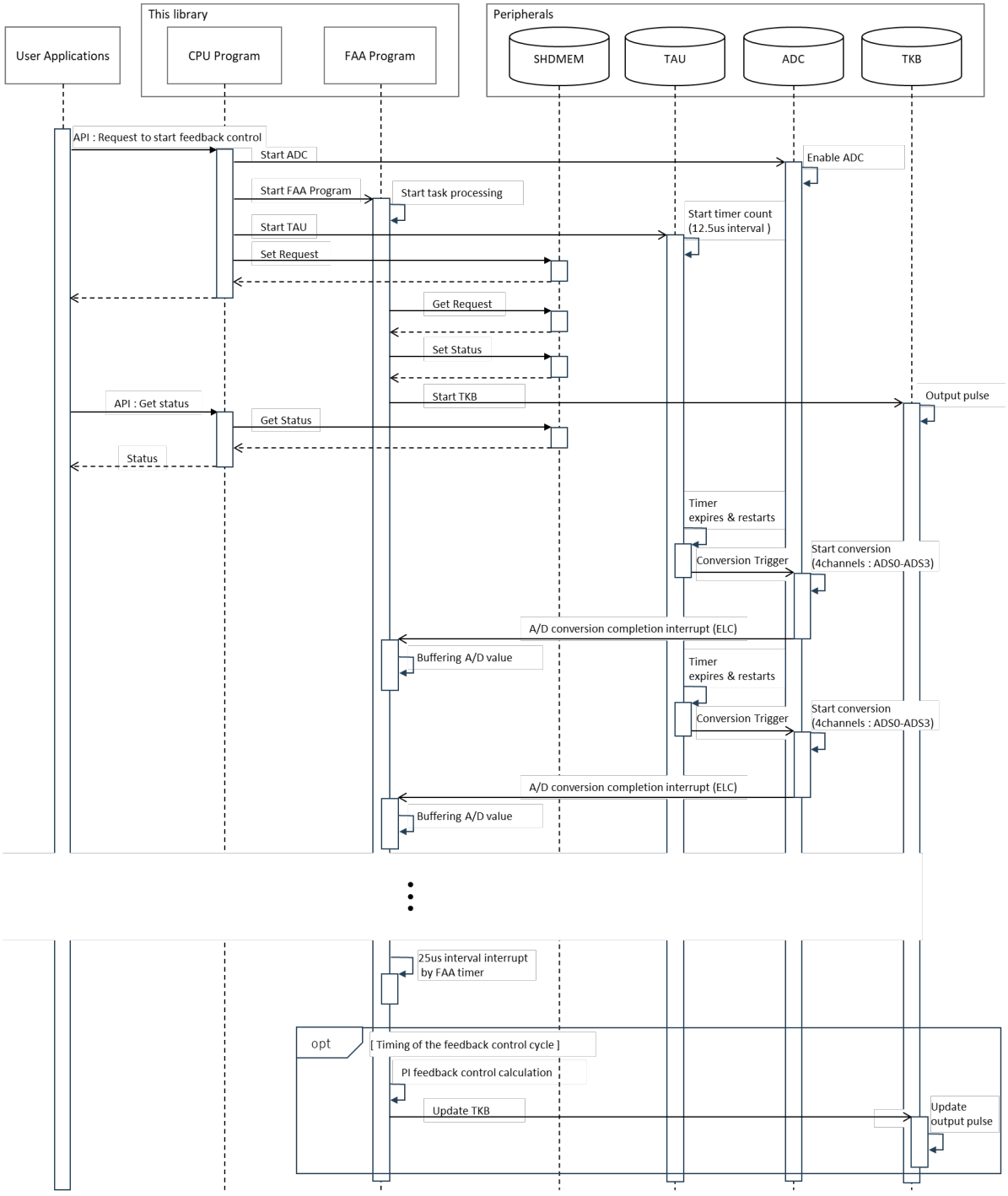

図 6.7　フィードバック制御シーケンス

6.6.2　Task 処理フロー (FAA プログラム)

FAA プログラムにおける Task 処理について説明します。FAA プログラムは起動後に初期化処理を行い、ループ内で以下の処理を行います。

- API によるリクエストの取得および実行処理
- PFC フィードバック制御処理 (PFC フィードバック制御周期毎に実行)
- LLC ソフトスタート制御処理 (25us 毎に実行)
- LLC フィードバック制御処理 (LLC フィードバック制御周期毎に実行)

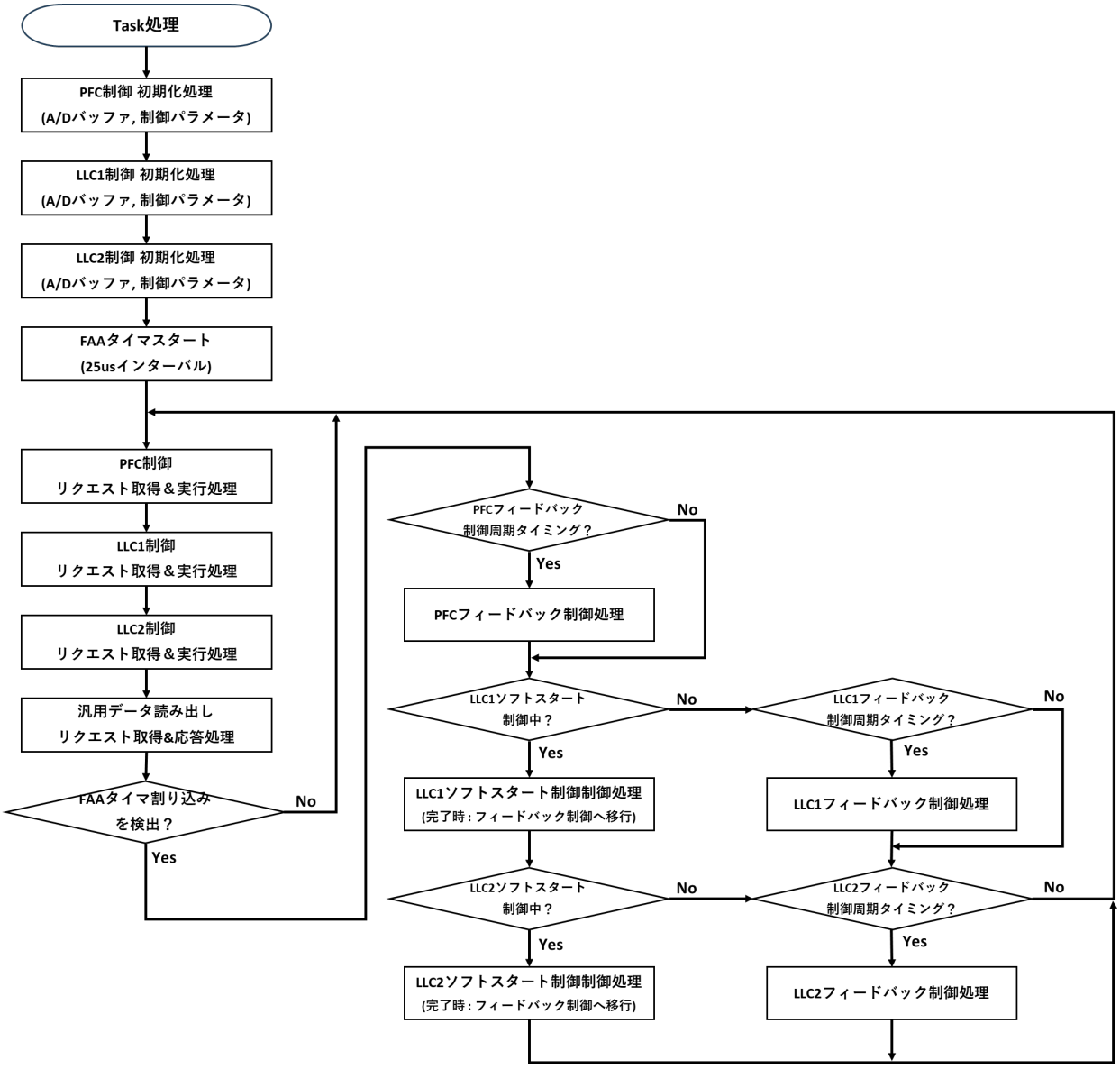

図 6.8　Task 処理フロー (FAA プログラム)

6.6.3　A/D 変換完了割り込み処理フロー (FAA プログラム)

FAA プログラムにおける A/D 変換完了割り込み処理フローについて説明します。

まず、12.5us のインターバルで TAU から割り込みが発生します。この割り込みが A/D 変換処理の開始トリガとなります。RL78/G24 マイコンでは ADC のアドバンスド・モードを使用することで、4 チャネルの A/D 変換を連続的に実施することができます。4 チャネルすべての A/D 変換が完了すると、イベント・リンク・コントローラ(ELC)を経由して、FAA プログラム内に設定された割り込み関数が呼び出されます。

FAA プログラム内の割り込み関数では、変換完了した A/D 値を取得し、各フィードバック制御向けにバッファ値や合計値の更新を行います。また、API を介したユーザーからの汎用 A/D 変換リクエストがあれば、対象となるチャネルの更新と、変換結果の取得が行われます。

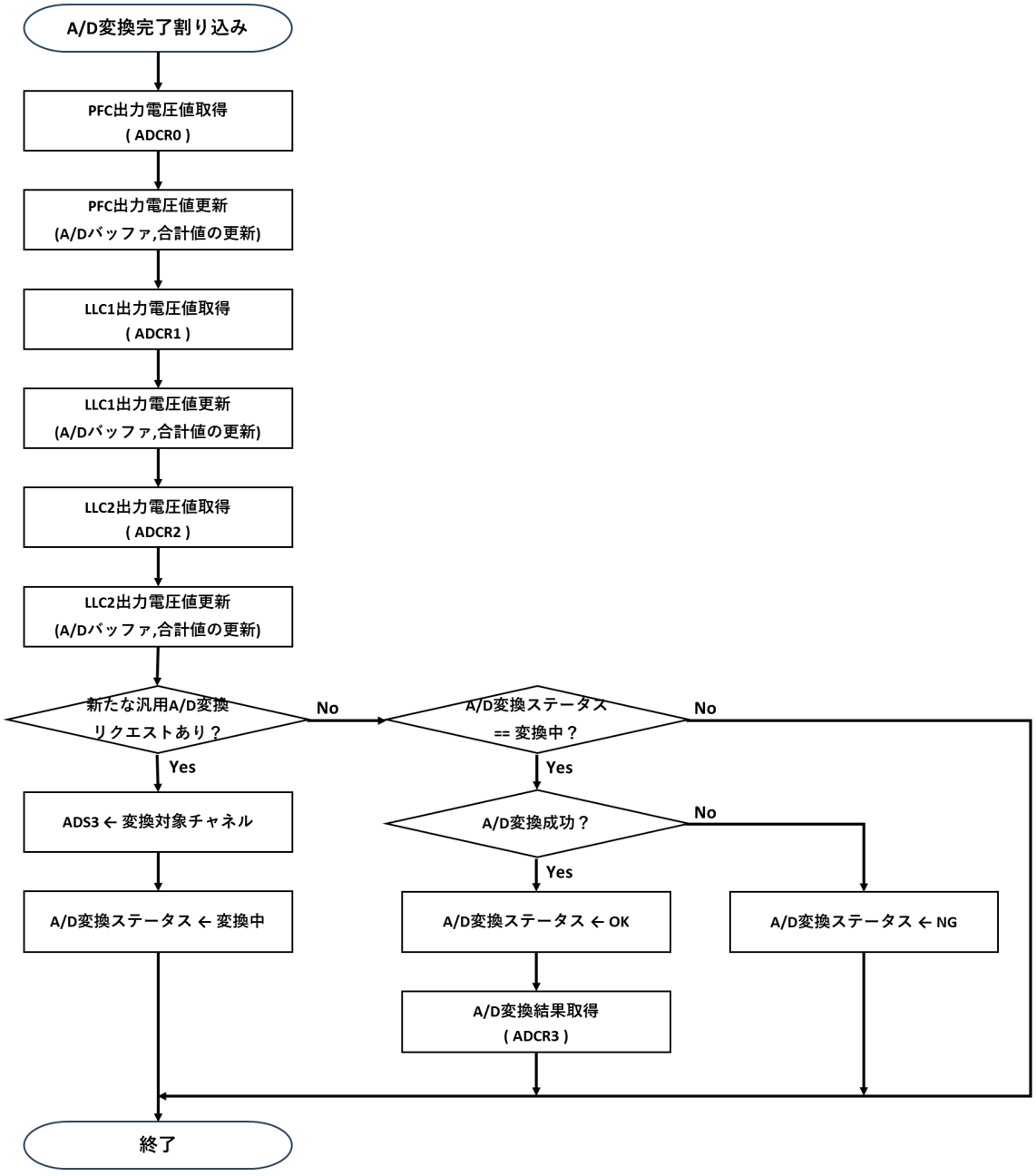

図 6.9　A/D 変換完了割り込み処理フロー (FAA プログラム)

7. コンフィグレーション仕様

本ライブラリにおけるコンフィグレーション仕様について説明します。

以下にスマート・コンフィグレータ (SC) 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。

7.1 基本設定項目

SC コンフィグレーションにおける基本設定 (Basic settings) 項目について説明します。

表 7.1　スマート・コンフィグレータ設定項目一覧(Basic settings)

property	value	description
Number of PFC converter channels	0～1	PFC コンバータの制御チャンネル数を選択します。
Number of LLC converter channels	0～2	LLC コンバータの制御チャンネル数を選択します。
ADC VREF(+)	VDD, AVREFP	A/D コンバータの＋側基準電圧を選択します。
ADC VREF(-)	VSS, AVREFM	A/D コンバータの-側基準電圧を選択します。
Low speed clock frequency at MOCO	1MHz, 2MHz, 4MHz	Low-Speed Mode での動作クロックを選択します。
Clock source in high speed mode	HOCO, X1	High-Speed Mode のクロックソースを選択します。

注：PFC コンバータと LLC コンバータの数の合計が 0 になる設定はできません。

7.2　PFC 設定項目

SC コンフィグレーションにおける PFC 設定 (PFC settings) 項目について説明します。

表 7.2　スマート・コンフィグレータ設定項目一覧(PFC settings) (1/2)

property	value	description
ANI pin for feedback input	ANI0, ANI1, ANI2, ANI3, ANI4, ANI5, ANI6, ANI7, ANI16, ANI17, ANI18, ANI19, ANI20, ANI21, ANI22, ANI23, ANI24, ANI25, ANI26, ANI27, ANI28, ANI29, ANI30	フィードバック入力に使用する ANI ピンを選択します。
Type of input value	Moving average	入力値のタイプを選択します。 詳細は 6.2.2.1 を参照してください。
Target input value	0 <= value <= 4095	PI フィードバック制御における目標値を指定します。
Dynamic-OVP detecting threshold	“Target input value” < value <= 4095	Dynamic-OVP 閾値を指定します。(出力電圧が閾値を超えている間、出力を一時的に停止する保護機能)
OVP detecting threshold	“Dynamic-OVP detecting threshold” < value <= 4095	OVP を検出するための入力閾値を指定します。(出力電圧が閾値を超えると出力を継続的に停止する保護機能)
A/D correction value	0 < value <= 4095	A/D 取得値に対する補正値を指定します。対象アナログ端子に入力する信号の出力インピーダンスが高い場合補正が必要になります。
TKBO pin	TKBO00/TKBO01, TKBO10/TKBO11, TKBO20/TKBO21	出力対象の TKBO 端子を選択します。
Active level of TKBO	Low level, High level	TKBO のアクティブレベルを選択します。
Master side detect edge of ZCD pin	Rise edge, Fall edge	マスタ側 ZCD 端子の検出エッジ(極性)を選択します。
Slave side detect edge of ZCD pin	Rise edge, Fall edge	スレーブ側 ZCD 端子の検出エッジ(極性)を選択します。

表 7.3　スマート・コンフィグレータ設定項目一覧(PFC settings) (2/2)

property		value	description
Longest period [us]		10 <= value <= 50	出力パルスの最長周期を指定します。 (ZCD が検出されなかった場合の出力周期設定)
Minimum on-time [us]		0.5 <= value <= 20	出力パルスの最小 On 時間を指定します。
Maximum on-time of each output pulse [us]		10 <= value <= 50	出力パルスの最大 On 時間を指定します。
PI parameters	Zero point frequency [Hz]	1 <= value <= 65535	PI フィードバック制御におけるゼロ・ポイント周波数を指定します。
	Feedback period [us]	100, 200, 300, 400	PI フィードバック制御周期を選択します。
	Proportional Gain (Kp)	0 < value < 16	PI フィードバック制御における比例ゲイン(Kp)を指定します。
Comparator combination of PFC OCP (Over Current Protection)		Disable, IVCMP0(M)+IVCMP1(S), IVCMP1(M)+IVCMP0(S), IVCMP1(M)+IVCMP3(S), IVCMP3(M)+IVCMP1(S), IVCMP0(M)+IVCMP3(S), IVCMP3(M)+IVCMP0(S)	OCP(過電流保護)にて使用するコンパレータ入力端子を選択します。OCP を適用しない場合は Disable を選択してください。
OCP threshold voltage ratio to VDD		0 < value < 16	OCP 閾値(電圧換算値)を VDD 電圧値に対する比率で指定します。 e.g. VDD = 5V、OCP 閾値 = 2.4V の場合、のパラメータ値は以下のように計算されます。 2.4V / 5V = 0.48

7.3　LLC 設定項目

SC コンフィグレーションにおける LLCn 設定 (LLCn settings) 項目について説明します。(n = 1, 2)

表 7.4　スマート・コンフィグレータ設定項目一覧(LLCn setting) (1/2)

property		value	description
ANI pin for feedback input		ANI0, ANI1, ANI2, ANI3, ANI4, ANI5, ANI6, ANI7, ANI16, ANI17, ANI18, ANI19, ANI20, ANI21, ANI22, ANI23, ANI24, ANI25, ANI26, ANI27, ANI28, ANI29, ANI30	フィードバック入力に使用する ANI ピンを選択します。
Type of input value		Moving average, Binary total	入力値の種類を選択します。 詳細は 6.3.1.1 を参照してください。
Target input value		0 <= value <= 4095	PI フィードバック制御における目標値を指定します。
Active level at over target voltage		Low level, High level	入力値が目標値を上回った際のフォトカプラの出力レベルを選択します。※入力タイプが Binary total 時のみ有効
TKBO pin		TKBO00/TKBO01, TKBO10/TKBO11, TKBO20/TKBO21	出力対象の TKBO 端子を選択します。
Active level of TKBO		Low level, High level	TKBO のアクティブレベルを選択します。
Deadtime [ns]		50 <= value <= 500	デッドタイム[ns]を指定します。
Soft start parameters	Frequency at the start [kHz]	20 <= value <= 500	ソフトスタート開始時の周波数[kHz]を指定します。
	Frequency at the end [kHz]	20 < value <= “Frequency at the start”	ソフトスタート終了時の周波数[kHz]を指定します。
	Max duration [ms]	0 < value <= 500	ソフトスタートの最大継続時間[ms]を指定します。
	Analog input value for terminate	0 <= value <= 4095	ソフトスタート終了判定閾値を指定します。(アナログ値)

表 7.5　スマート・コンフィグレータ設定項目一覧(LLCn setting) (2/2)

property		value	description
PI parameters	Frequency at the start [kHz]	20 <= value <= 500	PI フィードバックの開始周波数を指定します。
	Lowest frequency [kHz]	20 <= value <= 100	最低周波数[kHz]を指定します。
	Highest frequency [kHz]	Lowest frequency <= value <= 500	最高周波数[kHz]を指定します。
	Zero point frequency [Hz]	0 < value <= 10000	PI フィードバック制御におけるゼロ・ポイント周波数を指定します。
	Feedback period [us]	100, 200, 300, 400	PI フィードバック制御周期を選択します。
	Proportional Gain (Kp)	0 < value < 10	PI フィードバック制御における比例ゲイン(Kp)を指定します。
OCP(Over Current Protection)		Disable, IVCMP0, IVCMP1, IVCMP2, IVCMP3	OCP(過電流保護)にて使用するコンパレータ入力端子を選択します。OCP を適用しない場合は Disable を選択してください。
OCP threshold voltage ratio to VDD		0 < value < 1	OCP 閾値(電圧換算値)を VDD 電圧値に対する比率で指定します。 e.g. VDD = 5V、OCP 閾値 = 4V の場合、のパラメータ値は以下のように計算されます。 4V / 5V = 0.8

7.4　ユーザーアプリケーション設定項目

SC コンフィグレーションにおけるユーザーアプリケーション設定 (User settings) 項目について説明します。

表 7.6　スマート・コンフィグレータ設定項目一覧(User setting)

property	Value	description
ANI0	Unused/Used	ユーザーアプリケーションにてアナログ入力(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	
ANI5	Unused/Used	
ANI6	Unused/Used	
ANI7	Unused/Used	
ANI16	Unused/Used	
ANI17	Unused/Used	
ANI18	Unused/Used	
ANI19	Unused/Used	
ANI20	Unused/Used	
ANI21	Unused/Used	
ANI22	Unused/Used	
ANI23	Unused/Used	
ANI24	Unused/Used	
ANI25	Unused/Used	
ANI26	Unused/Used	
ANI27	Unused/Used	
ANI28	Unused/Used	
ANI29	Unused/Used	
ANI30	Unused/Used	

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_peripheral_t

この Typedef は本ライブラリにて使用する周辺機能を定義します。

```
typedef enum
{
    FAA_PERIPHERAL_ADC,
    FAA_PERIPHERAL_TKB,
    FAA_PERIPHERAL_TAU0,
    FAA_PERIPHERAL_PGACOMP,
    FAA_PERIPHERAL_DAC,
} e_faa_peripheral_t;
```

8.1.2 e_faa_clock_mode_t

この Typedef はクロック動作モードを定義します。

```
typedef enum
{
   CLOCK_MODE_HIGH_SPEED = 0,
   CLOCK_MODE_LOW_SPEED = 1,
} e_faa_clock_mode_t;
```

8.1.3 e_faa_pfc_status_t

この Typedef は PFC コンバータ制御におけるステータス情報を定義します。

```
typedef enum
{
    FAA_PFC_STATUS_STOP = 0x00,
    FAA_PFC_STATUS_FIXED_WIDTH = 0x01,
    FAA_PFC_STATUS_FEEDBACK_SINGLE = 0x02,
    FAA_PFC_STATUS_FEEDBACK_INTERLEAVED = 0x04,
    FAA_PFC_STATUS_DETECT_OVP = 0x10,
    FAA_PFC_STATUS_DETECT_OCP = 0x20,
} e_faa_pfc_status_t;
```

8.1.4　e_faa_pfc_output_t

この Typedef は PFC コンバータ制御おける出力モードを定義します。

```
typedef enum
{
    FAA_PFC_OUTPUT_SINGLE = 0,
    FAA_PFC_OUTPUT_INTERLEAVED = 1,
} e_faa_pfc_output_t;
```

8.1.5　e_faa_llc_status_t

この Typedef は LLC コンバータ制御におけるステータス情報を定義します。

```
typedef enum
{
    FAA_LLC_STATUS_STOP = 0,
    FAA_LLC_STATUS_FEEDBACK = 1,
    FAA_LLC_STATUS_DETECT_OCP = 2,
    FAA_LLC_STATUS_DETECT_ERROR = 255,
} e_faa_llc_status_t;
```

8.1.6　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汎用 A/D 変換制御において、変換対象のチャネルを指定するために使用されます。

表 7.6　スマート・コンフィグレータ設定項目一覧(User setting)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,
    . . .
    ADTEMPERSENSOR0 = 32,
    ADINTERREFVOLT = 33,
} e_faa_ad_channel_t;
```

8.1.7　e_faa_result_adc_t

この Typedef はユーザーアプリケーション向けユーザーアプリケーション向けに提供する汎用 A/D 変換制御における変換結果を定義します。

```
typedef enum
{
    FAA_ADC_NO_REQUEST = 0,
    FAA_ADC_CONVERTING = 1,
    FAA_ADC_CONVERT_OK = 2,
    FAA_ADC_CONVERT_NG = 3,
} e_faa_result_adc_t;
```

8.1.8　e_faa_request_id_t

この Typedef はデータ読み出し機能における Request ID を定義します。

```
typedef enum
{
    FAA_REQUEST_ID_NONE          = 0x0000,
    FAA_REQUEST_ID_PFC_V_FB      = 0x0001,
    FAA_REQUEST_ID_LLC1_V_FB     = 0x0002,
    FAA_REQUEST_ID_LLC2_V_FB     = 0x0003,
    FAA_REQUEST_ID_SPECIFIED_ADDR = 0xFFFF,
} e_faa_request_id_t;
```

8.1.9　faa_pfc_callback_t

この Typedef は PFC コンバータ制御における異常検出時のコールバック型を定義します。

```
typedef struct
{
    void (*p_detect_ovp)(void);            /* callback for detecting PFC OVP */
    void (*p_detect_ocp)(void);            /* callback for detecting PFC OCP */
} faa_pfc_callback_t;
```

8.1.10 faa_llc_callback_t

この Typedef は LLC コンバータ制御における異常検出時のコールバック型を定義します。

```
typedef struct
{
    void (*p_detect_ocp)(void);            /* callback for detecting LLC OCP */
} faa_llc_callback_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。

8.2.1　R_{ConfigName}_DPConv_Create

この関数は、デジタル電源コンバータ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}_DPConv_Create (void)
```

Parameters

なし

Return Values

なし

Properties

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

Description

この関数は、デジタル電源コンバータ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。

・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

・タイマ・アレイ・ユニット 0

・外部割り込み

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}_DPConv_PowerOn

この関数は、デジタル電源コンバータ制御で必要な周辺機能へのクロック供給を許可します。

Format

```
void R_{ConfigName}_DPConv_PowerOn (e_faa_peripheral_t peripheral)
```

Parameters

peripheral：対象とする周辺機能

FAA_PERIPHERAL_ADC：A/D コンバータ

FAA_PERIPHERAL_TKB：16 ビット・タイマ TB

FAA_PERIPHERAL_TAU0：タイマ・アレイ・ユニット 0

FAA_PERIPHERAL_PGACOMP：PGA/コンパレータ

FAA_PERIPHERAL_DAC：D/A コンバータ

Return Values

なし

Properties

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

Description

この関数は、引数で指定された周辺機能へのクロック供給を許可します。（周辺イネーブル・レジスタ 0-2 の対応 bit をセットします。）

後述の R_{ConfigName}_DPConv_PowerOff と組み合わせて使用することで、本ライブラリ内で使用される各周辺機能に対し、クロック供給を周辺機能単位で許可または禁止することができます。システムの状況に応じてクロック供給を最小化することで、消費電力の低減が可能となります。

Special Notes:

High-Speed Mode では e_faa_peripheral_t で定義される全ての周辺機能に対してクロック供給を許可してください。Low-Speed Mode ではユーザーアプリケーション要件に基づき、最適な設定を行ってください。

8.2.3　R_{ConfigName}_DPConv_PowerOff

この関数は、デジタル電源コンバータ制御で必要な周辺機能へのクロック供給を禁止します。

Format

```
void R_{ConfigName}_DPConv_PowerOff (e_faa_peripheral_t peripheral)
```

Parameters

peripheral：対象とする周辺機能

FAA_PERIPHERAL_ADC：A/D コンバータ

FAA_PERIPHERAL_TKB：16 ビット・タイマ TB

FAA_PERIPHERAL_TAU0：タイマ・アレイ・ユニット 0

FAA_PERIPHERAL_PGACOMP：PGA/コンパレータ

FAA_PERIPHERAL_DAC：D/A コンバータ

Return Values

なし

Properties

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

Description

この関数は、引数で指定された周辺機能へのクロック供給を禁止します。（周辺イネーブル・レジスタ 0-2 の対応 bit をクリアします。）

前述の R_{ConfigName}_DPConv_PowerOn と組み合わせて使用することで、本ライブラリ内で使用される各周辺機能に対し、クロック供給を周辺機能単位で許可または禁止することができます。システムの状況に応じてクロック供給を最小化することで、消費電力の低減が可能となります。

Special Notes:

High-Speed Mode では e_faa_peripheral_t で定義される全ての周辺機能に対してクロック供給を許可してください。Low-Speed Mode ではユーザーアプリケーション要件に基づき、最適な設定を行ってください。

8.2.4　R_{ConfigName}_DPConv_ChangeClockMode

この関数は、クロック動作モードを変更します。

Format

```
void R_{ConfigName}_DPConv_ChangeClockMode (e_faa_clock_mode_t clock)
```

Parameters

clock：クロック動作モード

CLOCK_MODE_HIGH_SPEED：High speed mode (f_{CLK}=48MHz, f_{TIMER}=96MHz)

CLOCK_MODE_LOW_SPEED：Low speed mode (f_{CLK}=1or2or4MHz) ※SC で設定したクロック

Return Values

なし

Properties

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

Description

この関数は、クロック動作モードを引数で指定されたモードに切り替えます。

指定可能なクロック動作モードの詳細は **6.1 クロック動作モード**を参照してください。

Special Notes:

なし

8.2.5　R_{ConfigName}_DPConv_PFC_Init

この関数は、PFC コンバータ制御機能を初期化します。

Format

```
void R_{ConfigName}_DPConv_PFC_Init (faa_pfc_callback_t * p_callback)
```

Parameters

p_callback：コールバック関数のポインタ

p_detect_ovp：PFC 過電圧検出時のコールバック関数登録用関数ポインタ

p_detect_ocp：PFC 過電流検出時のコールバック関数登録用関数ポインタ

Return Values

なし

Properties

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

Description

この関数は、PFC コンバータ制御機能において、パラメータの初期化およびユーザーコールバック関数の登録を行います。

Special Notes:

OVP および OCP の検出時にユーザーコールバックを必要としない場合は、引数として渡される faa_pfc_callback_t 型変数のメンバ変数を NULL に設定してください。

8.2.6　R_{ConfigName}_DPConv_PFC_StartFixedOnWidth

この関数は、固定 On 幅出力制御を開始します。

Format

```
void R_{ConfigName}_DPConv_PFC_StartFixedOnWidth
                                (uint16_t on_width_time_ns)
```

Parameters

on_width_time_ns：パルスの On 幅時間[ns]

Return Values

なし

Properties

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

Description

この関数は、引数で指定されたパルス On 幅時間に基づいて、固定 On 幅の出力制御を開始します。

Special Notes:

この関数は、Low-Speed Mode でのみ使用可能です。

8.2.7　R_{ConfigName}_DPConv_PFC_Suspend

この関数は、PFC 出力を一時中断します。

Format

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

Parameters

なし

Return Values

なし

Properties

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

Description

この関数は、PFC 出力を一時中断します。

Special Notes:

なし

8.2.8　R_{ConfigName}_DPConv_PFC_Resume

この関数は、PFC 出力を再開します。

Format

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

Parameters

なし

Return Values

なし

Properties

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

Description

この関数は、PFC 出力を再開します。

Special Notes:

なし

8.2.9　R_{ConfigName}_DPConv_PFC_StartFeedbackControl

この関数は、PFC フィードバック制御を開始します。

Format

```
void R_{ConfigName}_DPConv_PFC_StartFeedbackControl
                              (uint16_t on_width_time_ns)
```

Parameters

on_width_time_ns：パルスの On 幅時間[ns]

Return Values

なし

Properties

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

Description

この関数は、PFC フィードバック制御を開始します。開始時の出力 On 幅は、引数で指定されたパルス On 幅時間に基づいて算出されます。また、OCP 検出は有効となります。

Special Notes:

この関数は、High-Speed Mode でのみ使用可能です。

8.2.10 R_{ConfigName}_DPConv_PFC_Stop

この関数は、PFC コンバータ制御を停止します。

Format

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

Parameters

なし

Return Values

なし

Properties

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

Description

この関数は、PFC コンバータ制御を停止します。

Special Notes:

なし

8.2.11 R_{ConfigName}_DPConv_PFC_TaskForFeedbackControl

この関数は、PFC フィードバック制御におけるタスク処理を実行します。

Format

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

Parameters

なし

Return Values

なし

Properties

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

Description

この関数は、PFC フィードバック制御におけるタスク処理を実行します。

Special Notes:

この関数は、High-Speed Mode でのみ定期的に呼び出してください。

8.2.12 R_{ConfigName}_DPConv_PFC_GetStatus

この関数は、PFC コンバータ制御におけるステータス情報を取得します。

Format

```
e_faa_pfc_status_t R_{ConfigName}_DPConv_PFC_GetStatus (void)
```

Parameters

なし

Return Values

PFC ステータス情報

FAA_PFC_STATUS_STOP：停止中

FAA_PFC_STATUS_FIXED_WIDTH：固定 On 幅出力動作中

FAA_PFC_STATUS_FEEDBACK_SINGLE：フィードバック制御(シングル出力)動作中

FAA_PFC_STATUS_FEEDBACK_INTERLEAVED：フィードバック制御(インターリーブ出力)動作中

FAA_PFC_STATUS_DETECT_OVP：過電圧検出

FAA_PFC_STATUS_DETECT_OCP：過電圧検出

Properties

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

Description

この関数は、PFC コンバータ制御におけるステータス情報を取得します。

Special Notes:

なし

8.2.13 R_{ConfigName}_DPConv_PFC_ChangeOutputMode

この関数は、PFC 出力モードを変更します。

Format

```
void R_{ConfigName}_DPConv_PFC_ChangeOutputMode
                                (e_faa_pfc_output_t mode, bool is_fade)
```

Parameters

mode：PFC 出力モード

FAA_PFC_OUTPUT_SINGLE：シングル出力モード

FAA_PFC_OUTPUT_INTERLEAVED：インターリーブ出力モード

is_fade：フェード処理の有無

true：フェードあり

false：フェードなし

Return Values

なし

Properties

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

Description

この関数は、PFC 出力モードを引数で指定されたモードに切り替えます。フェード処理が要求される場合、スレーブ出力の開始および停止時の出力レベルを段階的に調整しスムーズな切り替えを実現します。

フェード処理についての詳細は **6.2.3.1 フェード処理**を参照してください。

Special Notes:

なし

8.2.14 R_{ConfigName}_DPConv_PFC_SetMaxFreqLimit

この関数は、PFC 出力における最大周波数を設定します。

Format

```
void R_{ConfigName}_DPConv_PFC_SetMaxFreqLimit (uint8_t freq_10khz)
```

Parameters

freq_10khz：最大周波数[10kHz]

Return Values

なし

Properties

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

Description

この関数は、PFC 出力における最大周波数を設定します。引数には、設定したい最大周波数を 10kHz 単位で指定してください。(例．200kHz→設定値 20)

Special Notes:

この関数はオプションです。ユーザーアプリケーション要件に応じて、呼び出してください。

8.2.15 R_{ConfigName}_DPConv_PFC_GetMaxFreqLimit

この関数は、PFC 出力における最大周波数設定値を取得します。

Format

```
uint8_t R_{ConfigName}_DPConv_PFC_GetMaxFreqLimit (void)
```

Parameters

なし

Return Values

最大周波数[10kHz]

Properties

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

Description

この関数は、PFC 出力における最大周波数設定値を取得します。取得値は、最大周波数(単位:10kHz)となります。

Special Notes:

この関数はオプションです。ユーザーアプリケーション要件に応じて、呼び出してください。

8.2.16 R_{ConfigName}_DPConv_PFC_SetFz

この関数は、PFC フィードバック制御におけるゼロ・ポイント周波数(f_Z)パラメータを設定します。

Format

```
void R_{ConfigName}_DPConv_PFC_SetFz (uint16_t fz_hz)
```

Parameters

fz_hz：ゼロ・ポイント周波数[Hz]

Return Values

なし

Properties

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

Description

この関数は、PFC フィードバック制御におけるゼロ・ポイント周波数(f_Z)パラメータを動的に設定します。設定されたパラメータは次回の PFC フィードバック制御処理時に反映されます。

Special Notes:

この関数はオプションです。ユーザーアプリケーション要件に応じて、呼び出してください。

8.2.17 R_{ConfigName}_DPConv_PFC_SetKp

この関数は、PFC フィードバック制御における比例定数(K_P)パラメータを設定します。

Format

```
void R_{ConfigName}_DPConv_PFC_SetKp (uint16_t kp)
```

Parameters

kp：比例定数(K_P)

Return Values

なし

Properties

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

Description

この関数は、PFC フィードバック制御における比例定数(K_P)パラメータを動的に設定します。引数として指定する値は、設定したいパラメータ値を 4096 倍した値を入力してください。設定されたパラメータは次回の PFC フィードバック制御処理時に反映されます。

Special Notes:

この関数はオプションです。ユーザーアプリケーション要件に応じて、呼び出してください。

8.2.18 R_{ConfigName}_DPConv_PFC_SetAdjustParamForInterleaved

この関数は、出力エネルギー補正式におけるパラメータ値を設定します。

Format

```
void R_{ConfigName}_DPConv_PFC_SetAdjustParamForInterleaved
                                (double slope, double intercept)
```

Parameters

slope：線形補正式における傾き

intercept：線形補正式における切片

Return Values

なし

Properties

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

Description

この関数は、シングルモードとインターリーブモードの切り替え時に発生する出力エネルギーの変動を補正するためのパラメータ値を設定します。仕様の詳細は 6.2.3.2 出力エネルギー補正を参照してください。

Special Notes:

この関数はオプションです。ユーザーアプリケーション要件に応じて、呼び出してください。

また、この関数を使用する場合は PFC フィードバック制御を開始する前に呼び出してください。

8.2.19 R_{ConfigName}_DPConv_PFC_GetOnWidthCount

この関数は、PFC マスタ出力におけるパルス On 幅を示すカウント値を取得します。

Format

```
uint16_t R_{ConfigName}_DPConv_PFC_ GetOnWidthCount (void)
```

Parameters

なし

Return Values

パルス On 幅カウント値

Properties

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

Description

この関数は、PFC マスタ出力におけるパルスの On 幅カウント値を取得します。

なお、PFC 動作を実現している 16 ビット・タイマ KB の供給クロックは 96MHz であるため、1 カウント当たりの時間は 10.416[ns]です。

Special Notes:

この関数はオプションです。ユーザーアプリケーション要件に応じて、呼び出してください。

8.2.20 R_{ConfigName}_DPConv_LLCn_Init

この関数は、LLCn コンバータ制御機能を初期化します。(n = 1, 2)

Format

```
void R_{ConfigName}_DPConv_LLCn_Init (faa_llc_callback_t * p_callback)
```

Parameters

p_callback：コールバック関数のポインタ

p_detect_ocp：LLCn の過電流検出時のコールバック関数登録用関数ポインタ

Return Values

なし

Properties

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

Description

この関数は、LLCn コンバータ制御機能において、パラメータの初期化およびユーザーコールバック関数の登録を行います。

Special Notes:

OCP の検出時にユーザーコールバックを必要としない場合は、引数として渡される faa_llc_callback_t 型変数のメンバ変数 p_detect_ocp を NULL に設定してください。

8.2.21 R_{ConfigName}_DPConv_LLCn_StartFeedbackControl

この関数は、LLCn フィードバック制御を開始します。(n = 1, 2)

Format

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

Parameters

なし

Return Values

なし

Properties

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

Description

この関数は、LLCn フィードバック制御を開始します。このとき OCP 検出は有効になります。

Special Notes:

この関数は、High-Speed Mode でのみ使用可能です。

8.2.22 R_{ConfigName}_DPConv_LLCn_Stop

この関数は、LLCn コンバータ制御を停止します。(n = 1, 2)

Format

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

Parameters

なし

Return Values

なし

Properties

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

Description

この関数は、LLCn コンバータ制御を停止します。

Special Notes:

なし

8.2.23 R_{ConfigName}_DPConv_LLCn_EnableOCP

この関数は、LLCn コンバータの過電流保護機能を有効化します。(n = 1, 2)

Format

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

Parameters

なし

Return Values

なし

Properties

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

Description

この関数は、LLCn コンバータの過電流保護機能を有効化します。

Special Notes:

この関数はオプションです。ユーザーが実現したい動作において必要な場合、呼び出してください。

ユーザーアプリケーション要件に応じて、Low-Speed Mode 中の任意のタイミングで OCP の検出を開始することができます。

8.2.24 R_{ConfigName}_DPConv_LLCn_DisableOCP

この関数は、LLCn コンバータの過電流保護機能を無効化します。(n = 1, 2)

Format

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

Parameters

なし

Return Values

なし

Properties

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

Description

この関数は、LLCn コンバータの過電流保護機能を無効化します。

Special Notes:

なし

8.2.25 R_{ConfigName}_DPConv_LLCn_GetStatus

この関数は、LLCn コンバータ制御におけるステータス情報を取得します。(n = 1, 2)

Format

```
e_faa_llc_status_t R_{ConfigName}_DPConv_LLCn_GetStatus (void)
```

Parameters

なし

Return Values

LLCn ステータス情報

FAA_LLC_STATUS_STOP：停止中

FAA_LLC_STATUS_FEEDBACK：フィードバック制御動作中

FAA_LLC_STATUS_DETECT_OCP：過電流検出

FAA_LLC_STATUS_DETECT_ERROR：未定義エラー検出

Properties

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

Description

この関数は、LLCn コンバータ制御におけるステータス情報を取得します。

Special Notes:

なし

8.2.26 R_{ConfigName}_DPConv_GetAdValue

この関数は、A/D 変換を実行し、変換結果の平均値を取得します。

Format

```
uint16_t R_{ConfigName}_DPConv_GetAdValue
                        (e_faa_ad_channel_t channel, uint8_t number_of_times)
```

Parameters

channel：ANI チャネル

number_of_times：変換回数

Return Values

A/D 平均値

Properties

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

Description

この関数は、引数で指定された ANI チャネルに対して A/D 変換を指定された回数実行し、その平均値を取得します。

Special Notes:

この関数は、Low-Speed Mode でのみ使用可能です。

この関数を使用した際の A/D 変換設定は以下です。

- 入力クロック周波数(ADM1 レジスタ ADLSP ビット)：4Mhz 以下
- ワンショット・セレクトモード
- ソフトウェアトリガ(本関数内でトリガ)
- アドバンスドモード(標準 1)で指定の 1ch のみ変換
- 12bit 分解能
- A/D 変換時間 43/f_{CLK} (High speed mode の fCLK が 4MHz の場合、10.75us/ch)
- サンプリング・クロック数 27f_{AD} (f_{AD}=f_{CLK})

8.2.27 R_{ConfigName}_DPConv_RequestAdValue

この関数は、指定されたアナログチャネルに対して A/D 変換をリクエストします。

Format

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

Parameters

channel：ANI チャネル

Return Values

なし

Properties

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

Description

この関数は、引数にて指定されたアナログチャネルに対して A/D 変換をリクエストします。

Special Notes:

この関数は、High-Speed Mode でのみ使用可能です。

この関数を使用した際の A/D 変換設定は以下です。

- 入力クロック周波数(ADM1 レジスタ ADLSP ビット)：4Mhz 超
- ワンショット・セレクトモード
- ハードウェアトリガ(本ライブラリ CPU プログラムの TAU01 で 12.5us 間隔でトリガ)
- アドバンスドモード(標準 1)で 4ch 変換
- 12bit 分解能
- A/D 変換時間 $43/f_{CLK}$ (High speed mode の fCLK が 48MHz なので、0.8958us/ch)
- サンプリング・クロック数 $27f_{AD}$ ($f_{AD}=f_{CLK}$)

8.2.28 R_{ConfigName}_DPConv_GetRequestedAdValue

この関数は、A/D 変換結果を取得します。

Format

```
e_faa_result_adc_t R_{ConfigName}_DPConv_GetRequestAdValue
                                    (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_CONVERT_OK の場合にのみ、有効値となります。

Return Values

A/D 変換ステータス

Properties

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

Description

この関数は、リクエストされた A/D 変換の結果取得します。

Special Notes:

この関数は、High-Speed Mode でのみ使用可能です。

この関数を呼び出す前に、R_{ConfigName}_DPConv_RequestAdValue 関数にて A/D 変換のリクエストを行ってください。

8.2.29 R_{ConfigName}_DPConv_RequestData

この関数は、指定されたデータの読み出しをリクエストします。

Format

```
void R_{ConfigName}_DPConv_RequestData (e_faa_request_id_t id, uint16_t addr)
```

Parameters

id：リクエスト ID

FAA_REQUEST_ID_NONE：リクエストなし

FAA_REQUEST_ID_PFC_V_FB：PFC のフィードバック入力に使用する ANI の A/D 値[注 1]

FAA_REQUEST_ID_LLC1_V_FB：LLC1 のフィードバック入力に使用する ANI の A/D 値[注 1]

FAA_REQUEST_ID_LLC2_V_FB：LLC2 のフィードバック入力に使用する ANI の A/D 値[注 1]

FAA_REQUEST_ID_SPECIFIED_ADDR：特定の FAA データ・メモリアドレス指定

addr：リクエスト FAA データ・メモリアドレス[注 2]

FAA アドレス・マップのデータ・メモリ(000H～7FFH)の範囲で特定データのアドレス

注 1 フィードバックに使用する入力値の種別に応じた値が取得できます。

注 2 リクエスト ID が FAA_REQUEST_SPECIFIED_ADDR の場合のみ、設定が有効となります。

Return Values

なし

Properties

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

Description

この関数は、引数にて指定されたデータの読み出しをリクエストします。

Special Notes:

この関数は、High-Speed Mode でのみ使用可能です。

8.2.30 R_{ConfigName}_DPConv_GetRequestedData

この関数は、リクエストされた読み出しデータを取得します。

Format

```
bool R_{ConfigName}_DPConv_GetRequestedData (uint32_t * p_data)
```

Parameters

p_data：読み出しデータの格納先

※本関数の戻り値が true (読み出し完了) の場合にのみ、有効値となります。

Return Values

取得結果

true： 読み出し完了

false： 読み出し中

Properties

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

Description

この関数は、リクエストされた読み出しデータを取得します。

Special Notes:

この関数は、High-Speed Mode でのみ使用可能です。

9. サンプル・プログラム

本章ではデジタル電源コンバータ制御ライブラリを用いたサンプル・プログラムについて説明します。

9.1 ハードウェア構成

ターゲットボードおよびハードウェアの詳細情報については、下記ドキュメントの 2 章を参照してください。

RL78/G24 による Interleaved CrM PFC and LLC 制御 (ハードウェア&ソフト基礎編) (R01AN8175)

9.2 動作確認条件

サンプル・プログラムは以下の環境で動作確認をしております。

表 9.1　動作確認環境

項目		内容
使用マイコン		RL78/G24 (R7F101GFGxFP)
使用ボード		RL78/G24 Interleaved PFC + LLC Board 400W Kit (RTK0EL0006D00000BJ)
動作周波数		高速オンチップ・オシレータ・クロック(fHOCO)：8MHz PLL クロック(fPLL)：96MHz CPU／周辺ハードウェア・クロック(fCLK)：48MHz
動作電圧		AC 100V, 230V
CS+ for CC	統合開発環境 (IDE)	ルネサス エレクトロニクス製 CS+ for CC V8.14.00
	コンパイラ	CC-RL V1.15.01
IAR	統合開発環境 (IDE)	IAR Systems 製 IAR Embedded Workbench for Renesas RL78 V5.10.3
	コンパイラ	IAR C/C++ Compiler for Renesas RL78 V5.10.3.2716
e2 studio	統合開発環境 (IDE)	ルネサス エレクトロニクス製 e2 studio 2025-10
	コンパイラ	CC-RL V1.15.01
スマート・コンフィグレータ (SC)		V1.15.0 [使用コンポーネント] Board Support Packages V1.91 UART 通信 V1.10.0 インターバル・タイマ V1.8.0 フレキシブル・アプリケーション・アクセラレータ V1.3.0 ポート V1.8.0 入力信号のハイ／ロウ幅測定 V1.7.0 割り込みコントローラ V1.7.0 ウォッチドッグ・タイマ V1.7.0 電圧検出回路 V1.6.1

9.3　ソフトウェア仕様

9.3.1　ソフトウェア制御

サンプル・プログラムのソフトウェア制御概要については下記ドキュメントの 2 章を参照してください。

RL78/G24 による Interleaved CrM PFC and LLC 制御 (ハードウェア&ソフト基礎編) (R01AN8175)

9.3.2　ソフトウェア構成

サンプル・プログラムのソフトウェア構成を記載します。

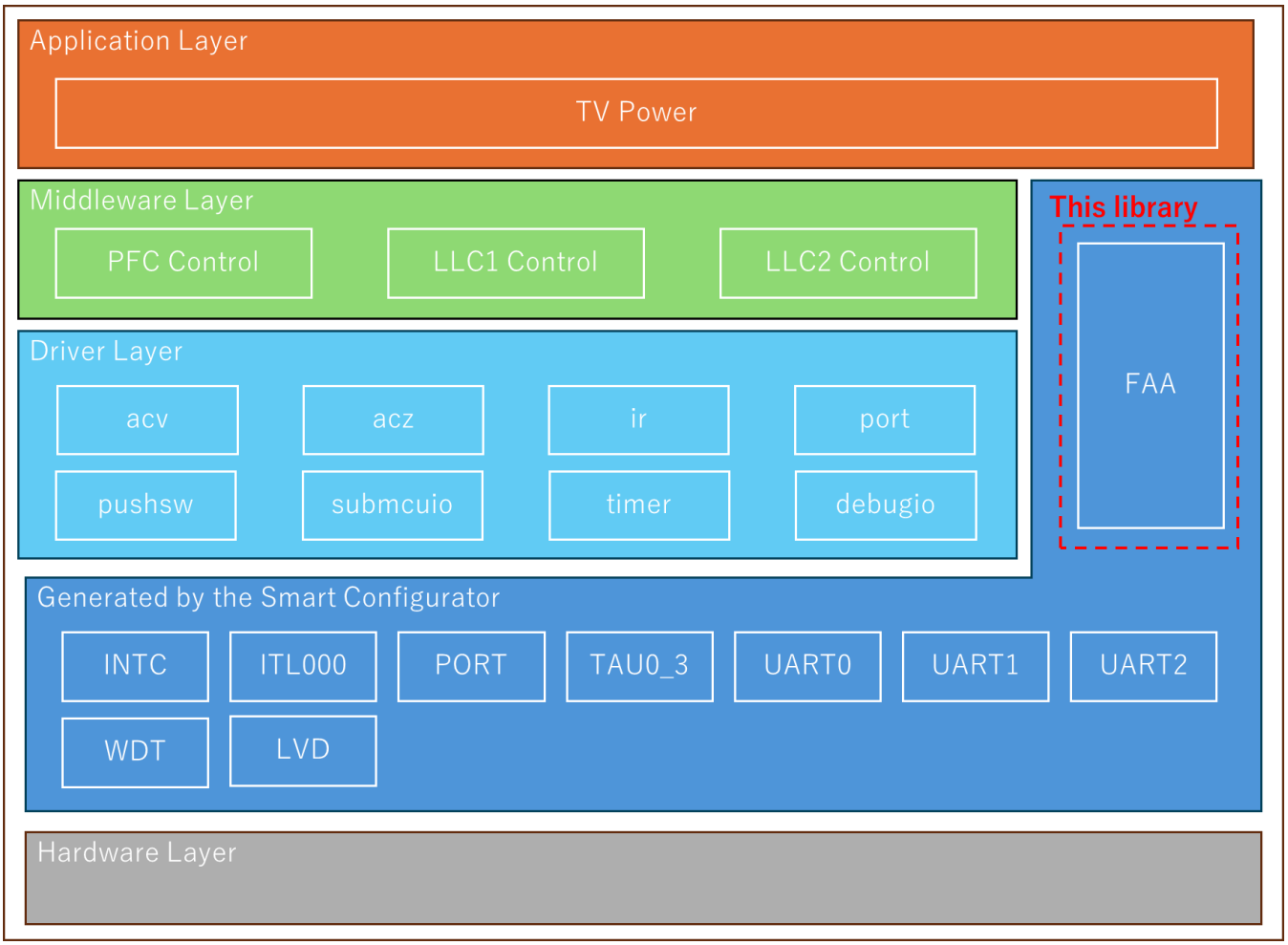

図 9.1　ソフトウェア構成図

9.3.3　フォルダ構成

サンプル・プログラムのフォルダ構成を記載します。

表 9.2　フォルダ構成

フォルダ、ファイル名	説明
TvPower_FAA	サンプル・プログラムのフォルダ
main.c	メインファイル
src	ソースコード格納フォルダ
r_common.h	共通定義ファイル
application	アプリケーションプログラム格納フォルダ
r_tvpower.c	TV 用電源アプリケーションのソースファイル
r_tvpower.h	TV 用電源アプリケーションのヘッダファイル
middleware	ミドルウェアプログラム格納フォルダ
r_pfc.c	PFC 制御ミドルウェアのソースファイル
r_pfc.h	PFC 制御ミドルウェアのヘッダファイル
r_llc1.c	LLC1 制御ミドルウェアのソースファイル
r_llc1.h	LLC1 制御ミドルウェアのヘッダファイル
r_llc2.c	LLC2 制御ミドルウェアのソースファイル
r_llc2.h	LLC2 制御ミドルウェアのヘッダファイル
driver	ドライバプログラム格納フォルダ
r_acv.c	AC 電圧判定ドライバのソースファイル
r_acv.h	AC 電圧判定ドライバのヘッダファイル
r_acz.c	AC ゼロクロス検出ドライバのソースファイル
r_acz.h	AC ゼロクロス検出ドライバのヘッダファイル
r_debugio.c	デバッグ用 IO ドライバのソースファイル
r_debugio.h	デバッグ用 IO ドライバのヘッダファイル
r_ir.c	IR ドライバのソースファイル
r_ir.h	IR ドライバのヘッダファイル
r_port.c	Port ドライバのソースファイル
r_port.h	Port ドライバのヘッダファイル
r_pushsw.c	PushSW ドライバのソースファイル
r_pushsw.h	PushSW ドライバのヘッダファイル
r_submcuio.c	Sub MCU IO ドライバのソースファイル
r_submcuio.h	Sub MCU IO ドライバのヘッダファイル
r_timer.c	Timer ドライバのソースファイル
r_timer.h	Timer ドライバのヘッダファイル
smc_gen	スマート・コンフィグレータ生成フォルダ (以下省略) ※配下に本ライブラリのコードが生成されます

9.3.4　オプション・バイト設定一覧

表 9-3 にオプション・バイト設定を示します。

表 9-3 オプション・バイト設定

アドレス	設定値	内容
000C0H/040C0H	1111 0010 (F2H)	ウォッチドッグ・タイマ動作許可 ・　オーバーフロー時間：2^8/fIL (7.8125 ms) ・　リセット解除後、カウント開始 ・　HALT/STOP モード時、カウンタ動作停止
000C1H/040C1H	1111 1011 (FBH)	LVD リセット・モード (立ち下がり：2.91V 立ち上がり：2.97V)
000C2H/040C2H	1110 1010 (EAH)	フラッシュ動作モード：高速メインモード 高速オンチップ・オシレータの周波数：8MHz
000C3H/040C3H	1000 0100 (84H)	オンチップ・デバッグ動作許可

9.3.5　コンフィグレーション設定

本サンプル・プログラムにおけるデジタル電源コンバータ制御ライブラリのコンフィグレーション設定内容は、以下の通りです。

表 9.4　スマート・コンフィグレータ設定値 (Basic settings)

property	value	description
Number of PFC converter channels	1	PFC コンバータの制御チャンネル数を選択します。
Number of LLC converter channels	2	LLC コンバータの制御チャンネル数を選択します。
ADC VREF(+)	AVREFP	A/D コンバータの＋側基準電圧を選択します。
ADC VREF(-)	AVREFM	A/D コンバータの-側基準電圧を選択します。
Low speed clock frequency at MOCO	4MHz	Low-Speed Mode での動作クロックを選択します。
Clock source in high speed mode	HOCO	High-Speed Mode のクロックソースを選択します。

表 9.5　スマート・コンフィグレータ設定値(PFC settings) (1/2)

property		value	description
ANI pin for feedback input		ANI4	フィードバック入力に使用する ANI ピンを選択します。
Type of input value		Moving average	入力値のタイプを選択します。
Target input value		2735	PI フィードバック制御における目標値を指定します。
Dynamic-OVP detecting threshold		2826	Dynamic-OVP 閾値を指定します。(出力電圧が閾値を超えている間、出力を一時的に停止する保護機能)
OVP detecting threshold		3022	OVP を検出するための入力閾値を指定します。(出力電圧が閾値を超えると出力を継続的に停止する保護機能)
A/D correction value		58	A/D 取得値に対する補正値を指定します。 対象アナログ端子に入力する信号の出力インピーダンスが高い場合補正が必要になります。
TKBO pin		TKBO10/TKBO11	出力対象の TKBO 端子を選択します。
Active level of TKBO		High level	TKBO のアクティブレベルを選択します。
Master side detect edge of ZCD pin		Fall edge	マスタ側 ZCD 端子の検出エッジ(極性)を選択します。
Slave side detect edge of ZCD pin		Fall edge	スレーブ側 ZCD 端子の検出エッジ(極性)を選択します。
Longest period [us]		20	出力パルスの最長周期を指定します。(ZCD が検出されなかった場合の出力周期設定)
Minimum on-time [us]		0.5	出力パルスの最小 On 時間を指定します。
Maximum on-time of each output pulse [us]		15.625	出力パルスの最大 On 時間を指定します。
PI parameters	Zero point frequency [Hz]	2	PI フィードバック制御におけるゼロ・ポイント周波数を指定します。
	Feedback period [us]	400	PI フィードバック制御周期を選択します。
	Proportional Gain (Kp)	0.25	PI フィードバック制御における比例ゲイン(Kp)を指定します。

表 9.6　スマート・コンフィグレータ設定値(PFC settings) (2/2)

property	value	description
Comparator combination of PFC OCP (Over Current Protection)	IVCMP1(M)+IVCMP3(S)	OCP(過電流保護)にて使用するコンパレータ入力端子を選択します。OCP を適用しない場合は Disable を選択してください。
OCP threshold voltage ratio to VDD	0.48	OCP 閾値(電圧換算値)を VDD 電圧値に対する比率で指定します。 e.g. VDD = 5V、OCP 閾値 = 2.4V の場合、のパラメータ値は以下のように計算されます。 2.4V / 5V = 0.48

表 9.7　スマート・コンフィグレータ設定値(LLC1 setting)

property		Value	description
ANI pin for feedback input		ANI5	フィードバック入力に使用する ANI ピンを選択します。
Type of input value		Binary total	入力値の種類を選択します。
Target input value		-	PI フィードバック制御における目標値を指定します。
Active level at over target voltage		Low level	入力値が目標値を上回った際のフォトカプラの出力レベルを選択します。※入力タイプが Binary total 時のみ有効
TKBO pin		TKBO00/TKBO01	出力対象の TKBO 端子を選択します。
Active level of TKBO		High level	TKBO のアクティブレベルを選択します。
Deadtime [ns]		300	デッドタイム[ns]を指定します。
Soft start parameters	Frequency at the start [kHz]	500	ソフトスタート開始時の周波数[kHz]を指定します。
	Frequency at the end [kHz]	120	ソフトスタート終了時の周波数[kHz]を指定します。
	Max duration [ms]	100	ソフトスタートの最大継続時間[ms]を指定します。
	Analog input value for terminate	9	ソフトスタート終了判定閾値を指定します。(アナログ値)
PI parameters	Frequency at the start [kHz]	100	PI フィードバックの開始周波数を指定します。
	Lowest frequency [kHz]	65	最低周波数[kHz]を指定します。
	Highest frequency [kHz]	300	最高周波数[kHz]を指定します。
	Zero point frequency [Hz]	1500	PI フィードバック制御におけるゼロ・ポイント周波数を指定します。
	Feedback period [us]	200	PI フィードバック制御周期を選択します。
	Proportional Gain (Kp)	0.015625	PI フィードバック制御における比例ゲイン(Kp)を指定します。
OCP(Over Current Protection)		IVCMP0	OCP(過電流保護)にて使用するコンパレータ入力端子を選択します。OCP を適用しない場合は Disable を選択してください。
OCP threshold voltage ratio to VDD		0.84	OCP 閾値(電圧換算値)を VDD 電圧値に対する比率で指定します。 e.g. VDD = 5V、OCP 閾値 = 4V の場合、のパラメータ値は以下のように計算されます。 4V / 5V = 0.8

表 9.8　スマート・コンフィグレータ設定値(LLC2 setting)

property		Value	description
ANI pin for feedback input		ANI6	フィードバック入力に使用する ANI ピンを選択します。
Type of input value		Binary total	入力値の種類を選択します。
Target input value		-	PI フィードバック制御における目標値を指定します。
Active level at over target voltage		Low level	入力値が目標値を上回った際のフォトカプラの出力レベルを選択します。※入力タイプが Binary total 時のみ有効
TKBO pin		TKBO20/TKBO21	出力対象の TKBO 端子を選択します。
Active level of TKBO		High level	TKBO のアクティブレベルを選択します。
Deadtime [ns]		300	デッドタイム[ns]を指定します。
Soft start parameters	Frequency at the start [kHz]	500	ソフトスタート開始時の周波数[kHz]を指定します。
	Frequency at the end [kHz]	120	ソフトスタート終了時の周波数[kHz]を指定します。
	Max duration [ms]	100	ソフトスタートの最大継続時間[ms]を指定します。
	Analog input value for terminate	9	ソフトスタート終了判定閾値を指定します。(アナログ値)
PI parameters	Frequency at the start [kHz]	100	PI フィードバックの開始周波数を指定します。
	Lowest frequency [kHz]	65	最低周波数[kHz]を指定します。
	Highest frequency [kHz]	300	最高周波数[kHz]を指定します。
	Zero point frequency [Hz]	1250	PI フィードバック制御におけるゼロ・ポイント周波数を指定します。
	Feedback period [us]	200	PI フィードバック制御周期を選択します。
	Proportional Gain (Kp)	0.059375	PI フィードバック制御における比例ゲイン(Kp)を指定します。
OCP(Over Current Protection)		IVCMP2	OCP(過電流保護)にて使用するコンパレータ入力端子を選択します。OCP を適用しない場合は Disable を選択してください。
OCP threshold voltage ratio to VDD		0.72	OCP 閾値(電圧換算値)を VDD 電圧値に対する比率で指定します。 e.g. VDD = 5V、OCP 閾値 = 4V の場合、のパラメータ値は以下のように計算されます。 4V / 5V = 0.8

表 9.9　スマート・コンフィグレータ設定値(User setting)

property	Value	description
ANI0	-	ユーザーアプリケーションにてアナログ入力(ANI)を使用するかどうかを選択します。
ANI1	-	
ANI2	Used	
ANI3	Used	
ANI4	-	
ANI5	-	
ANI6	-	
ANI7	Used	
ANI16	Unused	
ANI17	Unused	
ANI18	-	
ANI19	-	
ANI20	Unused	
ANI21	Unused	
ANI22	-	
ANI23	-	
ANI24	-	
ANI25	-	
ANI26	-	
ANI27	-	
ANI28	Unused	
ANI29	-	
ANI30	-	

9.3.6　環境構築方法

本サンプル・プログラムは CC-RL 版と IAR 版の 2 種のコンパイラにて動作するプロジェクトを提供しております。コンパイラ別の環境構築方法を記載します。

9.3.6.1　CC-RL 版 (CS+ for CC)

CC-RL 版 (CS+ for CC)の環境構築方法を記載します。

(1)　CS+ for CC のダウンロードおよびインストール

サンプルアプリケーションを動作させるにあたって、統合開発環境 CS+ for CC が必要になります。

ルネサスエレクトロニクスのホームページからダウンロード及びインストールしてください。

(2)　サンプルアプリケーションの起動

1. 提供プロジェクトファイルを解凍、任意の場所（フォルダ）へ配置します。
2. 解凍したプロジェクトフォルダ内の TvPower_FAA.mtpj をダブルクリックします。

(3)　サンプルアプリケーションのビルド方法

1. CS+ for CC のメニューバーから「ビルド」→「ビルド・プロジェクト」をクリックします。
2. ビルドが開始され、「すべてのメッセージ」にビルドの状況が表示されるので“ビルド終了”というメッセージが表示されたらビルド完了です。

(4)　サンプルアプリケーションのデバッグ方法

1. ターゲットボードにマウントしたデバッグボードの E2 Lite コネクタに E2 Lite を接続します。
2. デバッグボードの JP400 をオープンにします。
3. E2 Lite からデバッグボードへ 3.3V 電源を供給する設定にします。
4. ターゲットボードの CN1-CN2 間に AC 入力電圧を印加します。
5. CS+ for CC のメニューバーから「デバッグ」→「デバッグ・ツールへダウンロード」をクリックします。
6. マイクロコントローラへプログラムをダウンロードします。

   「デバッグ」→「実行」をクリックし、サンプルアプリケーションのデバッグを実行します。

注：ブレークポイント使用時は、安全のため「RL78 E2 Lite プロパティ」→「停止時にタイマ周辺のエミュレーションを停止する」を「はい」に設定してください。

また、プログラムがブレークポイントで停止した場合は、CPU をリセットしてから再開してください。

9.3.6.2　CC-RL 版 (e2 studio)

CC-RL 版 (e2studio)の環境構築方法を記載します。

(1)　e2 studio のダウンロードおよびインストール

サンプルアプリケーションを動作させるにあたって、統合開発環境 e2 studio が必要になります。

ルネサスエレクトロニクスのホームページからダウンロード及びインストールしてください。

(2)　サンプルアプリケーションのインポート

1. 提供プロジェクトファイルを解凍、任意の場所（フォルダ）へ配置します。
2. e2 studio を起動します。
3. e2 studio のメニューバーから「ファイル」→「インポート」を選択します。
4. 「既存プロジェクトをワークスペースへ」を選択し、「次へ」を選択します。
5. ルート・ディレクトリーの選択の「参照」を選択し、1.で配置したプロジェクトのルート・ディレクトリーを選択します。
6. プロジェクトのリストに表示される TvPower_FAA にチェックを入れて「終了」を選択します。

(3)　サンプルアプリケーションのビルド方法

1. プロジェクト・エクスプローラからインポートしたサンプル・プロジェクトを右クリックし、「プロジェクトをビルド」をクリックします。
2. ビルドが開始され、「コンソール」にビルドの状況が表示されるので“Build Finished.”というメッセージが表示されたらビルド完了です。

(4)　サンプルアプリケーションのデバッグ方法

1. ターゲットボードにマウントしたデバッグボードの E2 Lite コネクタに E2 Lite を接続します。
2. デバッグボードの JP400 をオープンにします。
3. E2 Lite からデバッグボードへ 3.3V 電源を供給する設定にします。
4. ターゲットボードの CN1-CN2 間に AC 入力電圧を印加します。
5. e2 studio の ボタンをクリックします。
6. マイクロコントローラへプログラムをダウンロードし、デバッグを行います。

注：ブレークポイント使用時は、安全のため「デバッグ構成」→「デバッグツール設定」→「停止中はタイマー・グループのエミュレーションを停止する」を「はい」に設定してブレーク中にタイマ出力が停止(Hi-z)するようにしてください。

また、プログラムがブレークポイントで停止した場合は、そこから再開させず、CPU をリセットしてプログラムの最初から実行するようにしてください。

9.3.6.3　IAR 版

IAR 版の環境構築方法を記載します。

(1)　IAR Embedded Workbench のダウンロードおよびインストール

サンプルアプリケーションを動作させるにあたって、統合開発環境 IAR Embedded Workbench for Renesas RL78 が必要になります。

IAR Systems のホームページからダウンロード及びインストールしてください。

(2)　サンプルアプリケーションの起動

1. IAR Embedded Workbench for Renesas RL78 を起動します。
2. 「ファイル」タブ→「新規ワークスペース」を選択します。
3. 「ファイル」から「名前を付けてワークスペース保存」を選択し任意の名前、場所(フォルダ)にワークスペースを配置します。
4. 提供プロジェクトファイルを解凍、作成したワークスペース(フォルダ)を配置します。
5. 「プロジェクト」から「既存プロジェクトの追加」を選択し、3.で配置したプロジェクトファイル(EWP ファイル)を選択します。
6. 「ファイル」から「終了」をクリックします。

(3)　サンプルアプリケーションのビルド方法

1. ワークスペースでプロジェクトを右クリックし、「メイク」を選択します。
2. ビルドが開始され、「コンソール」にビルドの状況が表示されるので“エラー合計数: 0”というメッセージが表示されたらビルド完了です。

(4)　サンプルアプリケーションのデバッグ方法

1. ターゲットボードにマウントしたデバッグボードの E2 Lite コネクタに E2 Lite を接続します。
2. デバッグボードの JP400 をオープンにします。
3. E2 Lite からデバッグボードへ 3.3V 電源を供給する設定にします。
4. ターゲットボードの CN1-CN2 間に AC 入力電圧を印加します。
5. IAR Embedded Workbench for Renesas RL78 の ボタンをクリックします。
6. マイクロコントローラへプログラムをダウンロードし、デバッグを行います。

注：ブレークポイント使用時は、安全のため、デバッガの「ハードウェア設定」ウィンドウ中の「周辺ブレーク」にある「A(タイマ)」にチェックを入れ、ブレーク中にタイマ出力が停止(Hi-z)するようにしてください。

また、プログラムがブレークポイントで停止した場合は、そこから再開させず、CPU をリセットしてプログラムの最初から実行するようにしてください。

9.3.7　API 関数

本サンプル・プログラムにおけるミドルウェア層の API 関数を参考情報として記載します。

表 9.10　PFC 制御 (ミドルウェア層) API 関数一覧

関数名	説明
R_PFC_Init	PFC 制御機能を初期化します。
R_PFC_StartBoost	PFC 昇圧制御を開始します。
R_PFC_StartBurst	PFC バースト制御を開始します。
R_PFC_StartNormalSingle	PFC 通常制御を開始します。(シングル出力動作)
R_PFC_SwitchToInterleaved	PFC 出力をインターリーブモードへ切り替えます。
R_PFC_SwitchToSingle	PFC 出力をシングルモードへ切り替えます。
R_PFC_Stop	PFC 制御を停止します。
R_PFC_TickMs	PFC 制御における Tick タイミングを通知します。
R_PFC_TaskNormal	PFC 通常制御における Task 処理を実行します。
R_PFC_GetStatus	PFC 制御状態を取得します。
R_PFC_UpdateMaxFreqLimit	PFC 最大周波数リミット設定を更新します。
R_PFC_GetLoadPower100V	負荷電力を推計し、取得します。(AC100V)
R_PFC_GetLoadPower200V	負荷電力を推計し、取得します。(AC200V)

表 9.11　LLCn 制御 (ミドルウェア層) API 関数一覧 (n = 1, 2)

関数名	説明
R_LLCn_Init	LLCn 制御機能を初期化します。
R_LLCn_StartBurst	LLCn バースト制御を開始します。
R_LLCn_StartNormal	LLCn 通常制御を開始します。
R_LLCn_Stop	LLCn 制御を通知します。
R_LLCn_TickMs	LLCn 制御における Tick タイミングを通知します。
R_LLCn_GetStatus	LLCn 制御状態を取得します。

10. QE for Lighting & Power を用いた動作

QE for Lighting & Power (V2.0.0 以降)では、電源制御パラメータの設定、シミュレーション、評価、コード生成までを一貫して行うことが可能です

本章では QE for Lighting & Power を用いてターゲットボードの電源制御機能を確認・評価する際の一連の動作について説明します。

10.1 QE for Lighting & Power のダウンロード

ルネサスエレクトロニクスのホームページからダウンロード及びインストールしてください。

10.2 ワークスペースの準備

ターゲットボードの選択やアプリケーションプログラムを格納するフォルダとなるワークスペースを設定します。

10.2.1 ボードの選択

1. 「2. ワークスペースの準備」を開き、「ルネサスボード」を選択します。
2. 対象プロジェクトの選択から「[R01AN8175JJ0100] Interleaved CrM PFC and LLC Control with RL78/G24 (Hardware & Software Basics)」を選択します。

10.2.2 ワークスペースの選択

「ワークスペースの選択」からアプリケーションプログラムを格納する任意のフォルダを選択します。

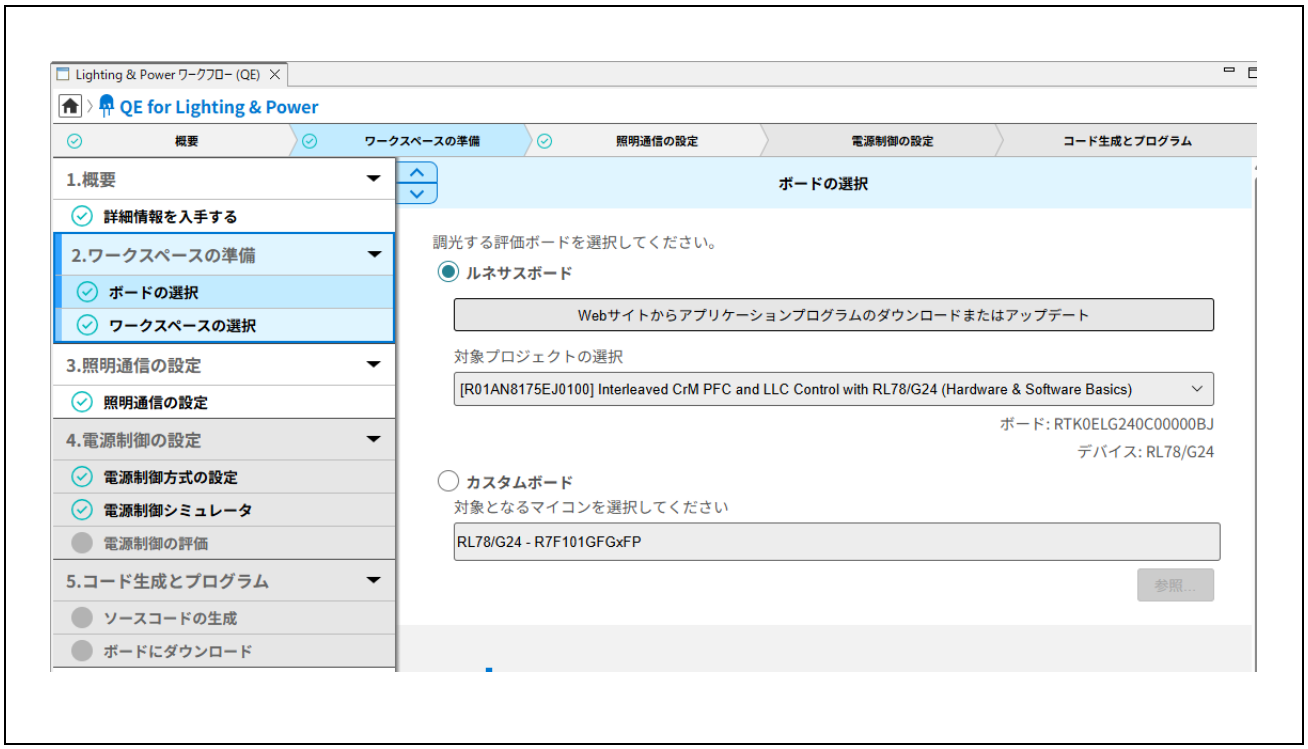

図 10.1 ワークスペースの準備

10.3 照明通信の設定

本対象プロジェクトでは照明通信に対応していないため「3.照明通信の設定」は使用できません。

図 10.2 照明通信の設定

10.4 電源制御の設定

「4.電源制御の設定」では、Power Configuration ビューを用いて回路特性値および電源制御パラメータの変更が可能です。

図 10.3 電源制御の設定

10.4.1 電源制御方式の設定

「電源制御方式を有効」を有効にすることで、電源制御パラメータの設定、シミュレーション、コード生成などの電源制御機能が使用可能になります。

10.4.1.1 Power Configuration ビュー

「Power Configuration ビューを開く」ボタンを押下することで、回路図の参照と、回路特性値および電源制御パラメータの変更が可能です。

Power Configuration ビューはターゲットボードの回路図（上部）とコンフィグ可能なパラメータ（下部）で構成されます。

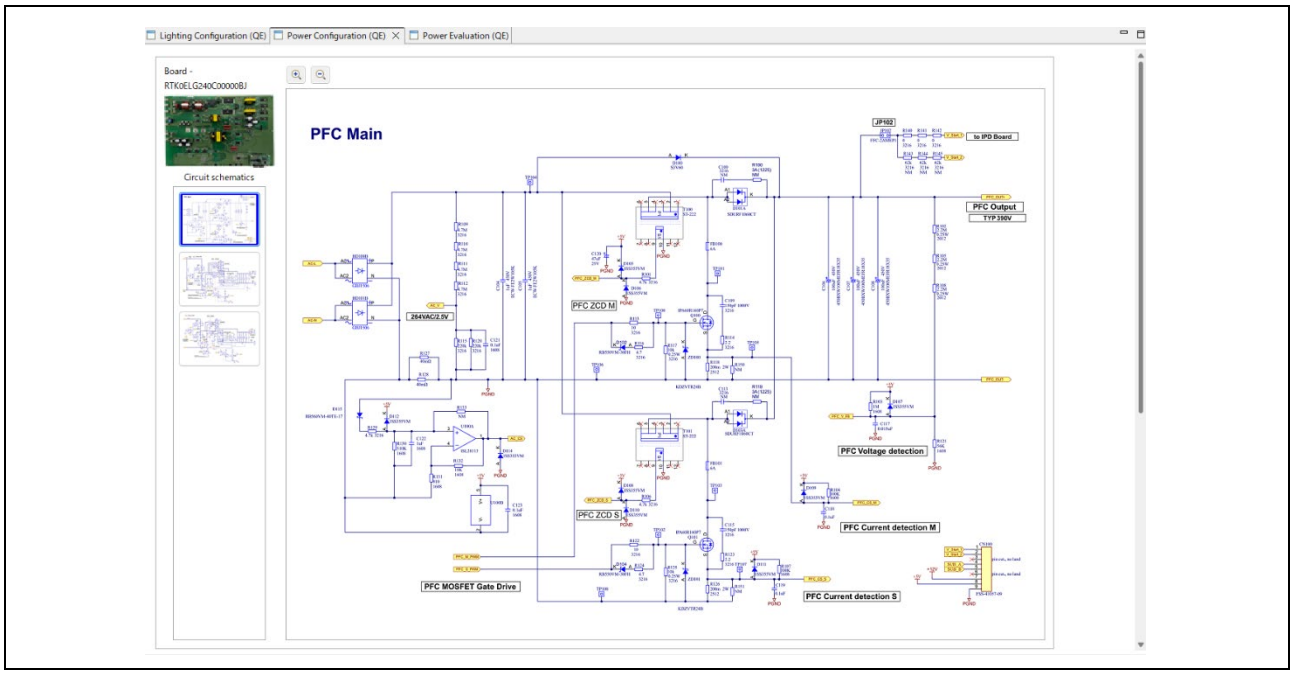

図 10.4 Power Configuration (ターゲットボードの回路図)

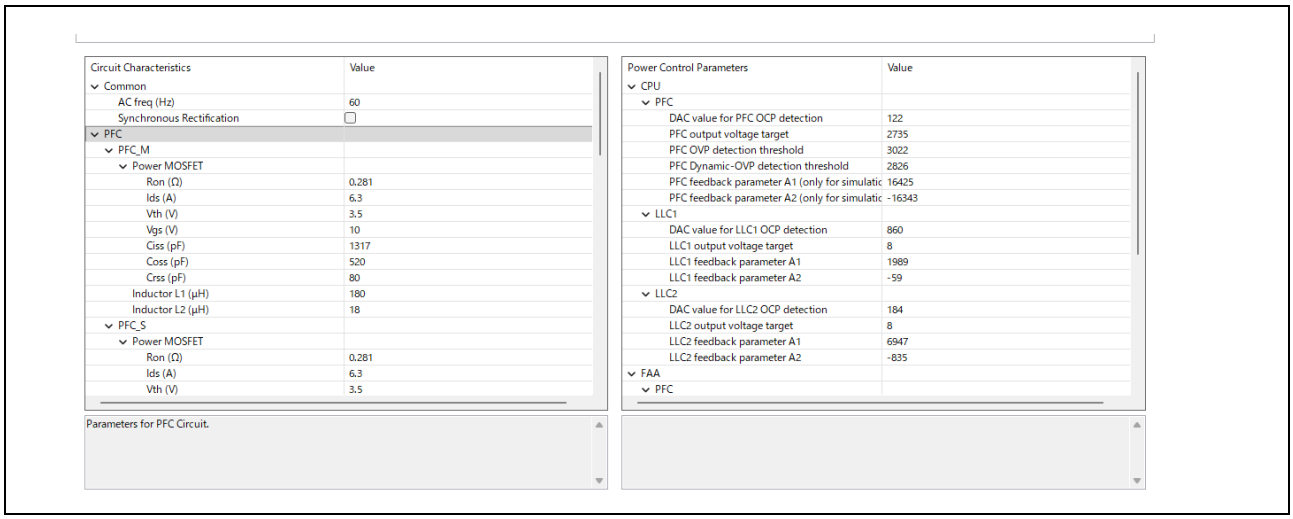

図 10.5 Power Configuration (コンフィグ可能なパラメータ)

コンフィグ可能なパラメータは「Circuit Characteristics」と「Power Control Parameters」から構成されます。

Circuit Characteristics は回路特性のパラメータを設定することができます。ここで設定したパラメータは後述の電源制御シミュレータに反映されます。

表 10.1　Circuit Characteristics 一覧（1/3）

項目			デフォルト値	説明
Common	AC freq (Hz)		60	AC 電圧の周波数
	Synchronous Rectification		チェックなし	LLC の同期／非同期整流回路を切り替えます。 チェックすると LLC 同期整流回路が有効になります。
PFC_M	Power MOSFET	Ron (Ω)	0.281	PFC マスタ側 MOSFET Ron [Ω]
		Ids (A)	6.3	PFC マスタ側 MOSFET Ids [A]
		Vth (V)	3.5	PFC マスタ側 MOSFET Vth [V]
		Vgs (V)	10	PFC マスタ側 MOSFET Vgs [V]
		Ciss (pF)	1317	PFC マスタ側 MOSFET Ciss [pF]
		Coss (pF)	520	PFC マスタ側 MOSFET Coss [pF]
		Crss (pF)	80	PFC マスタ側 MOSFET Crss [pF]
	Inductor L1 (μH)		180	PFC マスタ側トランスの一次側インダクタンス L1 [μH]
	Inductor L2 (μH)		18	PFC マスタ側トランスの二次側インダクタンス L2 [μH]
PFC_S	Power MOSFET	Ron (Ω)	0.5	PFC スレーブ側 MOSFET Ron [Ω]
		Ids (A)	6.3	PFC スレーブ側 MOSFET Ids [A]
		Vth (V)	3.5	PFC スレーブ側 MOSFET Vth [V]
		Vgs (V)	1317	PFC スレーブ側 MOSFET Vgs [V]
		Ciss (pF)	520	PFC スレーブ側 MOSFET Ciss [pF]
		Coss (pF)	520	PFC スレーブ側 MOSFET Coss [pF]
		Crss (pF)	80	PFC スレーブ側 MOSFET Crss [pF]
	Inductor L1 (μH)		180	PFC スレーブ側トランスの一次側インダクタンス L1 [μH]
	Inductor L2 (μH)		18	PFC スレーブ側トランスの二次側インダクタンス L2 [μH]

表 10.2　Circuit Characteristics 一覧（2/3）

項目			デフォルト値	説明
LLC1	Power MOSFET High Side	Ron (Ω)	0.237	LLC1 high side MOSFET Ron [Ω]
		Ids (A)	3.6	LLC1 high side MOSFET Ids [A]
		Vth (V)	4	LLC1 high side MOSFET Vth [V]
		Vgs (V)	10	LLC1 high side MOSFET Vgs [V]
		Ciss (pF)	807	LLC1 high side MOSFET Ciss [pF]
		Coss (pF)	249	LLC1 high side MOSFET Coss [pF]
		Crss (pF)	80	LLC1 high side MOSFET Crss [pF]
	Power MOSFET Low Side	Ron (Ω)	0.237	LLC1 low side MOSFET Ron [Ω]
		Ids (A)	3.6	LLC1 low side MOSFET Ids [A]
		Vth (V)	4	LLC1 low side MOSFET Vth [V]
		Vgs (V)	10	LLC1 low side MOSFET Vgs [V]
		Ciss (pF)	807	LLC1 low side MOSFET Ciss [pF]
		Coss (pF)	249	LLC1 low side MOSFET Coss [pF]
		Crss (pF)	80	LLC1 low side MOSFET Crss [pF]
	Inductor L1 (μH)		525	LLC1 トランスの一次側インダクタンス L1 [μH]
	Inductor L2 (μH)		2.5	LLC1 トランスの二次側インダクタンス L2 [μH]
	Inductor L3 (μH)		2.5	LLC1 L3 [μH]
	Inductor LR (μH)		75	LLC1 の負荷抵抗[μH]
	Capacitor CR (μF)		0.044	LLC1 の共振コンデンサ容量 [μF]

表 10.3　Circuit Characteristics 一覧（3/3）

項目			デフォルト値	説明
LLC2	Power MOSFET High Side	Ron (Ω)	0.237	LLC2 high side MOSFET Ron [Ω]
		Ids (A)	3.6	LLC2 high side MOSFET Ids [A]
		Vth (V)	4	LLC2 high side MOSFET Vth [V]
		Vgs (V)	10	LLC2 high side MOSFET Vgs [V]
		Ciss (pF)	807	LLC2 high side MOSFET Ciss [pF]
		Coss (pF)	249	LLC2 high side MOSFET Coss [pF]
		Crss (pF)	80	LLC2 high side MOSFET Crss [pF]
	Power MOSFET Low Side	Ron (Ω)	0.237	LLC2 low side MOSFET Ron [Ω]
		Ids (A)	3.6	LLC2 low side MOSFET Ids [A]
		Vth (V)	4	LLC2 low side MOSFET Vth [V]
		Vgs (V)	10	LLC2 low side MOSFET Vgs [V]
		Ciss (pF)	807	LLC2 low side MOSFET Ciss [pF]
		Coss (pF)	249	LLC2 low side MOSFET Coss [pF]
		Crss (pF)	80	LLC2 low side MOSFET Crss [pF]
	Inductor L1 (μH)		425	LLC2 トランスの一次側インダクタンス L1 [μH]
	Inductor L2 (μH)		28	LLC2 トランスの二次側インダクタンス L2[μH]
	Inductor L3 (μH)		28	LLC2 L3 [μH]
	Inductor LR (μH)		75	LLC2 の負荷抵抗[μH]
	Capacitor CR (μF)		0.044	LLC2 の共振コンデンサ容量 [μH]

Power Control Parameters は電源制御に関するパラメータを設定することができます。ここで設定したパラメータは後述の電源制御シミュレータと生成プログラムに反映されます。本書では FAA 版評価プログラムの設定項目を以下に示します。

表 10.4 電源制御パラメータ値一覧(1/2)

項目		デフォルト値	説明
PFC	PFC output voltage target	2735	PFC_V_FB ピンの目標 A/D 値を設定します。 実測値に基づき、以下の暫定計算式を使用します。 _N_DPConv_PFC_TARGET_VALUE = 6.5 × Voltage + 207.5
	PFC OVP detection threshold	3022	PFC の OVP 検出用しきい値を設定します （検出対象：PFC_V_FB ピン）。 実測値に基づき、以下の暫定計算式を使用します。 _N_DPConv_PFC_OVP_THRESHOLD = 6.5 × Voltage + 207.5
	PFC Dynamic-OVP detection threshold	2826	PFC の Dynamic-OVP 検出用しきい値を設定します （検出対象：PFC_V_FB ピン）。 実測値に基づき、以下の暫定計算式を使用します。 _N_DPConv_PFC_DOVP_THRESHOLD = 6.5 × Voltage + 207.5
	PFC output voltage target correction	58	PFC 出力電圧目標における A/D 値の補正値を設定します。
	PFC feedback parameter A1 (only for simulation)	16425	PI 制御の係数 A1 を設定します。 A1 = (π × fz × T + 1) × Kp 計算を容易にし、小数部の精度を向上させるため A1 には 65536 を乗算する必要があります。 例：fz = 2 Hz、T = 400 μs、Kp = 0.25 A1 = (π × 2 × 400 × 10^{-6} + 1) × 0.25 × 65536 = 16425
	PFC feedback parameter A2 (only for simulation)	-16433	PI 制御の係数 A2 A2 = (π × fz × T − 1) × Kp 計算を容易にし、小数部の精度を向上させるため、A2 には 65536 を乗算する必要があります。 例：fz = 2 Hz、T = 400 μs、Kp = 0.25 A2 = (π × 2 × 400 × 10^{-6} − 1) × 0.25 × 65536 = −16343

表 10.5 電源制御パラメータ値一覧(2/2)

項目		デフォルト値	説明
LLC1	LLC1 feedback parameter A1	1989	PI 制御の係数 A1 A1 = (π × fz × T + 1) × Kp 計算を容易にし、小数部の精度を向上させるため、A1 には 65536 を乗算する必要があります。 例：fz = 1500 Hz、T = 200 μs、Kp = 0.0156 _N_DPConv_LLC1_A1_PARAM = (π × 1500 × 200 × 10^{-6} + 1) × 0.0156 × 65536
	LLC1 feedback parameter A2	-59	PI 制御の係数 A2 A2 = (π × fz × T − 1) × Kp 計算を容易にし、小数部の精度を向上させるため、A2 には 65536 を乗算する必要があります。 例：fz = 1500 Hz、T = 200 μs、Kp = 0.0156 _N_DPConv_LLC1_A2_PARAM = (π × 1500 × 200 × 10^{-6} − 1) × 0.0156 × 65536 = −59
LLC2	LLC2 feedback parameter A1	6947	PI 制御の係数 A1 A1 = (π × fz × T + 1) × Kp 計算を容易にし、小数部の精度を向上させるため、A1 には 65536 を乗算する必要があります。
LLC2	LLC2 feedback parameter A2	-835	PI 制御の係数 A2 A2 = (π × fz × T − 1) × Kp 計算を容易にし、小数部の精度を向上させるため、A2 には 65536 を乗算する必要があります。

10.4.2 電源制御シミュレータ

電源制御シミュレータは、Power Configuration ビューで設定したパラメータを用いてターゲットボードの動作をソフトウェア上でシミュレーションする機能です。

本シミュレータを使用することで、実機で確認する前に、PFC や LLC の電圧・電流の時間変化や制御動作の傾向をグラフ等で事前に確認できます。

10.4.2.1 電源制御シミュレータダイアログ

「電源制御シミュレーションダイアログを開く」ボタンを押下することで電源制御シミュレータを起動することができます。電源制御シミュレータでは、以下パラメータを設定したうえで「Start Simulation」ボタンを押下することでシミュレーション開始できます。シミュレーション中の各パラメータ値の推移をグラフで可視化されます。

表 10.6 設定可能パラメーター覧

項目	時間[ms] (デフォルト値)	デフォルト値		説明
AC Voltage Gain	0.000	282.000 [V]		入力電圧 設定した時間［ms］が経過すると、設定値が反映されます。
LLC1 Load Resistance	0.000	100 [Ω]		LLC1 負荷抵抗 設定した時間［ms］が経過すると、設定値が反映されます
LLC2 Load Resistance	0.000	100 [Ω]		LLC2 負荷抵抗 設定した時間［ms］が経過すると、設定値が反映されます
Operating modes	0.000	SW1 チェックなし	SW2 チェックなし	動作モード切り替え SW **基本動作**： 設定した時間［ms］が経過すると、設定した SW 状態が反映されます。 • チェックあり：SW 押下 • チェックなし：SW 未押下 SW の 押下時間 により、短押し/長押しを判定します。 **判定条件**： 短押し：1999 ms 以下 長押し：2000 ms 以上 **動作**： SW1 短押し：LLC2 起動／停止 SW1 長押し：最大周波数リミット機能 ON/OFF SW2 短押し：Standby/Normal モード切り替え SW2 長押し：Standby/Normal モード切り替え
Simulation time [ms]	-	5		シミュレーション時間

表 10.7 出力波形一覧

項目		説明
Common	Input AC Power	AC 電力[W]
PFC	PFC Output Current	PFC 出力電流[A]
	PFC Output Voltage	PFC 出力電圧[V]
	PFC M PWM On Duty	PFC マスタ側 PWM デューティ比[%]
	PFC S PWM On Duty	PFC スレーブ側 PWM デューティ比[%]
	PFC PWM Frequency	PFC PWM 動作周波数[Hz]
LLC1	LLC1 Output Current	LLC1 出力電流[A]
	LLC1 Output Voltage	LLC1 出力電圧[V]
	LLC1 LC Resonant Current	LLC1 LC 共振電流[A]
	LLC1 Secondary Current	LLC1　二次側電流[A]
	LLC1 PWM Frequency	LLC1 PWM 動作周波数[Hz]
LLC2	LLC2 Output Current	LLC1 出力電流[A]
	LLC2Output Voltage	LLC1 出力電圧[V]
	LLC2 LC Resonant Current	LLC1 LC 共振電流[A]
	LLC2 Secondary Current	LLC1　二次側電流[A]
	LLC2 PWM Frequency	LLC1 PWM 動作周波数[Hz]

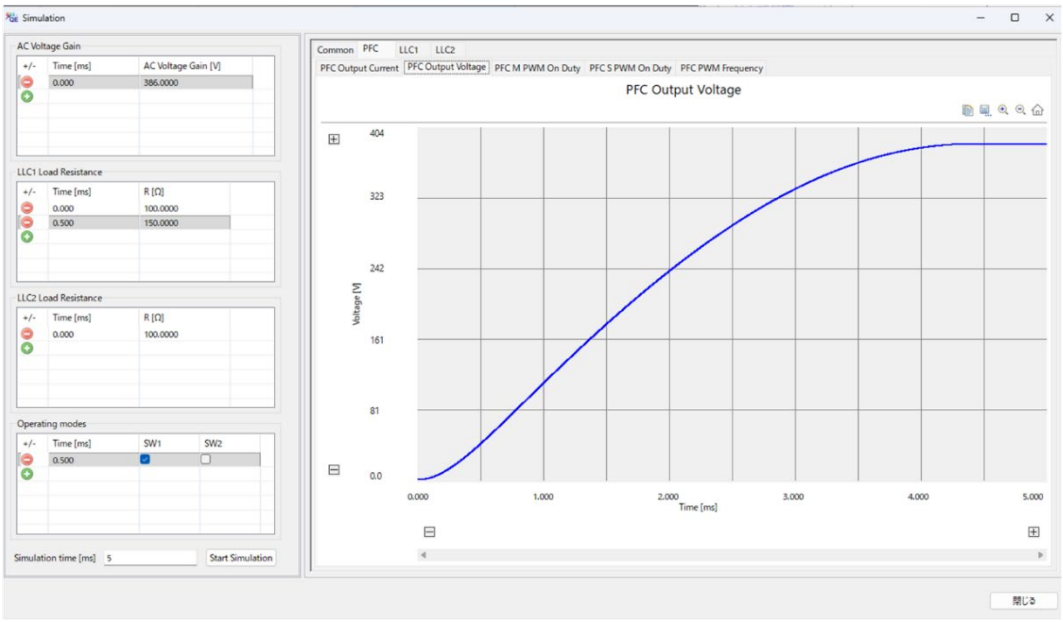

図 10.6 電源制御シミュレータダイアログ

操作手順(基本操作)

1. シミュレーション条件の設定

   AC Voltage Gain、負荷抵抗値、動作モード切り替え（SW 動作）など、

   シミュレーション条件を必要に応じて設定します。

   各設定項目には複数の条件を設定でき、指定した「時間［ms］」が経過すると、シミュレーション中に設定値が動的に切り替わります。

2. シミュレーション時間の設定

   「Simulation time [ms]」でシミュレーション時間を入力します。

3. シミュレーションの実行/停止

   「Start Simulation」を押下するとシミュレーションが開始します。

   シミュレーション中は、ボタンが「Stop Simulation」に切り変わり、そのボタンを押下してシミュレーションを停止できます。

   シミュレーション停止後、再度「Start Simulation」を押下することで、シミュレーションを再開できます。

4. 出力波形の確認

   シミュレーション実行中は、PFC や LLC の電圧・電流の時間変化、PWM デューティ比など、電源制御動作の傾向を波形として確認できます。

操作手順(Normal モード移行方法)

Standby モードから Normal モードへの遷移をシミュレートする場合は、「Simulation time [ms]」を 600ms 以上に設定し「Operating modes」の SW2 を 600ms 以降に短押しとなるように設定し、シミュレーションを開始してください。

シミュレーション開始から 600 ms 経過後に SW2 の短押し操作が反映され、Standby モードから Normal モードへ切り替わります。

なお、上記の操作は、デフォルト条件での一例です。

Power Configuration ビューで変更した回路特性値や電源制御パラメータの内容に応じて、「Simulation time [ms]」 および 「Operating modes」のタイミングは適切な値に変更してください。

シミュレーション時間の目安

Simulation time［ms］で指定したシミュレーション時間に応じて、PC 上での演算処理が行われます。

シミュレーションの実行時間は、PC の性能に依存しますが、目安として以下のようになります。

表 10.8 シミュレーション時間の目安

Simulation time[ms] (設定値)	シミュレーション実行時間	実行環境
1000	約 90 分	CPU：Intel Core i7-13700 メモリ：32GB OS：Windows11

注意事項

Simulation time に大きな値を設定すると実行時間やメモリ使用量に影響を与え、エラーが発生する可能性があります。

詳細はリリースノートを参照してください。

10.4.3 電源制御の評価

Power Configuration ビューで設定したパラメータ等を反映した評価プログラムの生成することができます。また、「Variable Read/Write」や「Variable Monitoring」を用いて評価プログラムが動作しているターゲットボード上で様々なパラメータ値の変化をリアルタイムに確認することが可能です。

図 10.7 電源制御の評価

10.4.3.1 ビルドの設定

Power Configuration ビューで設定したパラメータ等を反映した、評価用のプロジェクトのビルド設定を行います。

(1) コンパイラの選択

生成するプロジェクトは以下コンパイラに対応しています。

- CC-RL コンパイラ（e² studio）
- CC-RL コンパイラ（CS+）
- IAR RL78 コンパイラ

(2) プログラムの種類の選択

また、電源制御処理を実行するコアの種類として CPU と FAA を選択できます。

本書では FAA を選択する前提で以下を説明します。

10.4.3.2 評価プログラムの準備

選択したコンパイラとプログラムの種類に合わせた評価プログラムを生成します。

(1) 設定に合わせたプロジェクトの生成

「設定に合わせたプロジェクトの生成」を押下すると評価用プロジェクトが以下フォルダに生成されます。

<workspace>¥qeLighting_gen¥tuning program

(2) 通信プログラムをプロジェクトに追加

「通信プログラムをプロジェクトに追加」ボタンを押下すると Variable Read/Write や Variable Monitoring で使用する通信プログラムがプロジェクトに追加されます。

※通信プログラムを追加した後は、ヘルプの記載内容(「設定変更と評価プログラムの作成(ヘルプ参照)」を押下)に従い、生成されたアプリケーションプログラム統合してください。

10.4.3.3 ボードにダウンロード

生成したプロジェクトをターゲットボードに書き込みます。

ターゲットボードにマウントしたデバッグボードの E2 Lite コネクタに E2 Lite を接続し、「評価用プログラムを評価ボードに書き込む」ボタンを押下してください。

10.4.3.4 ボード接続

通信には USB ケーブルを使用し、ターゲットボードとホスト PC を接続します

本接続は、Variable Read/Write および Variable Monitoring による電源制御評価時の通信に使用します。

注意：通信を開始する前に、SW2 を手動で押下し、ターゲットボードを Normal モードに設定してください。

1. デバッグボードに USB ケーブルを接続します。
2. 「ビットレート」に 500000 を設定します。
3. 「COM Port」に PC と接続した COM ポートを指定します。
4. 「接続」ボタンを押下します。

10.4.3.5 電源制御の評価

ターゲットボード上の電源制御パラメータをリアルタイムで評価します。

2 つの評価機能を保有しています。

(1) Variable Read/Write

Variable Read/Write では、アプリケーションプログラムの内部変数やレジスタの値をリアルタイムに読み出し、モニタリングすることが可能です。

本アプリケーションノートでモニタリングできる内部変数はすべて読み取り専用であり、値の書き込みはできません。

なお、共有メモリレジスタ値の書き込みは可能ですが、意図しない動作を招く可能性があるため、書き込みは行わないでください。

「[Variable Read/Write]タブを開く」ボタンを押下すると[Power Evaluation (QE)]ビューの[Variable Read/Write]タブが開きます。

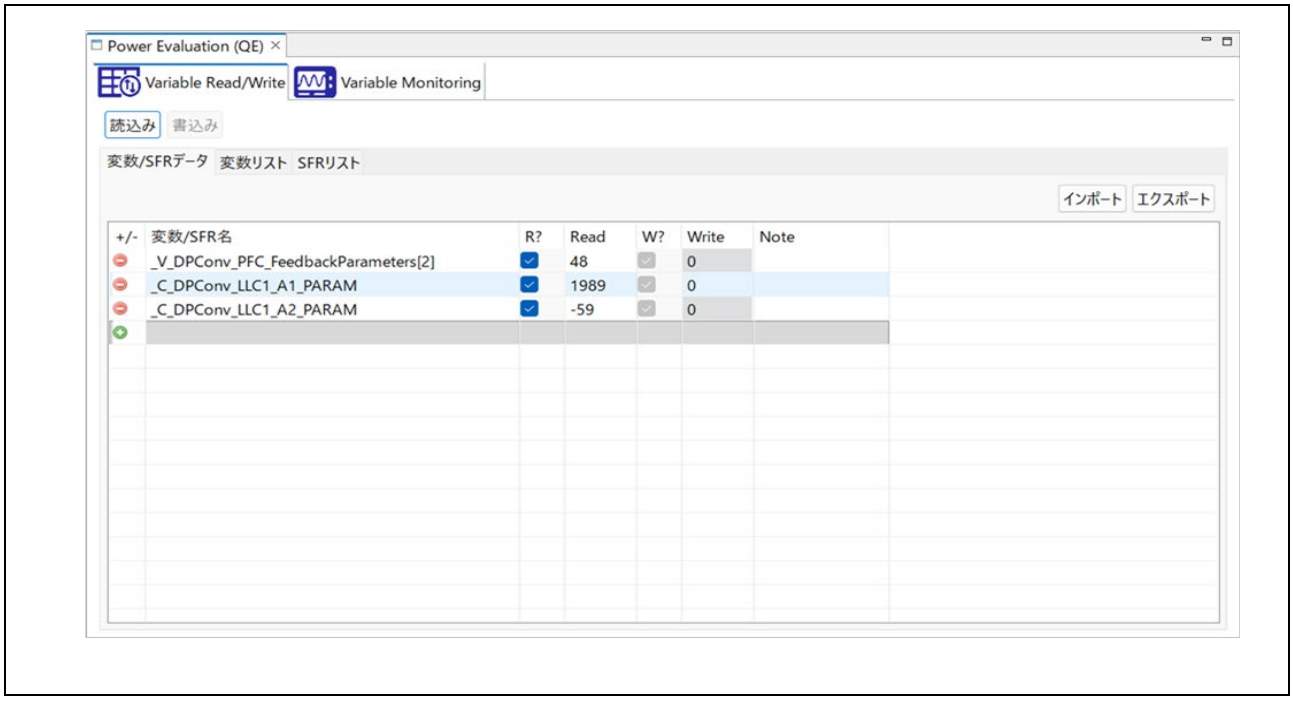

図 10.8 Variable Read/Write

操作手順

1. 評価対象変数の選択

   タブ内の変数選択リストから、評価対象となる電源制御変数を選択します

2. 変数値の読み出し

   「読み込み」ボタンを押下すると現在の変数値が表示されます。

(2)　Variable Monitoring

Variable Monitoring では、評価プログラムの内部変数やレジスタの値一定周期で連続的に取得し、時間変化をグラフとして表示することが可能です。

「[Variable Monitoring]タブを開く」ボタンを押下すると[Power Evaluation (QE)]ビューの[Variable Monitoring]タブが開きます。

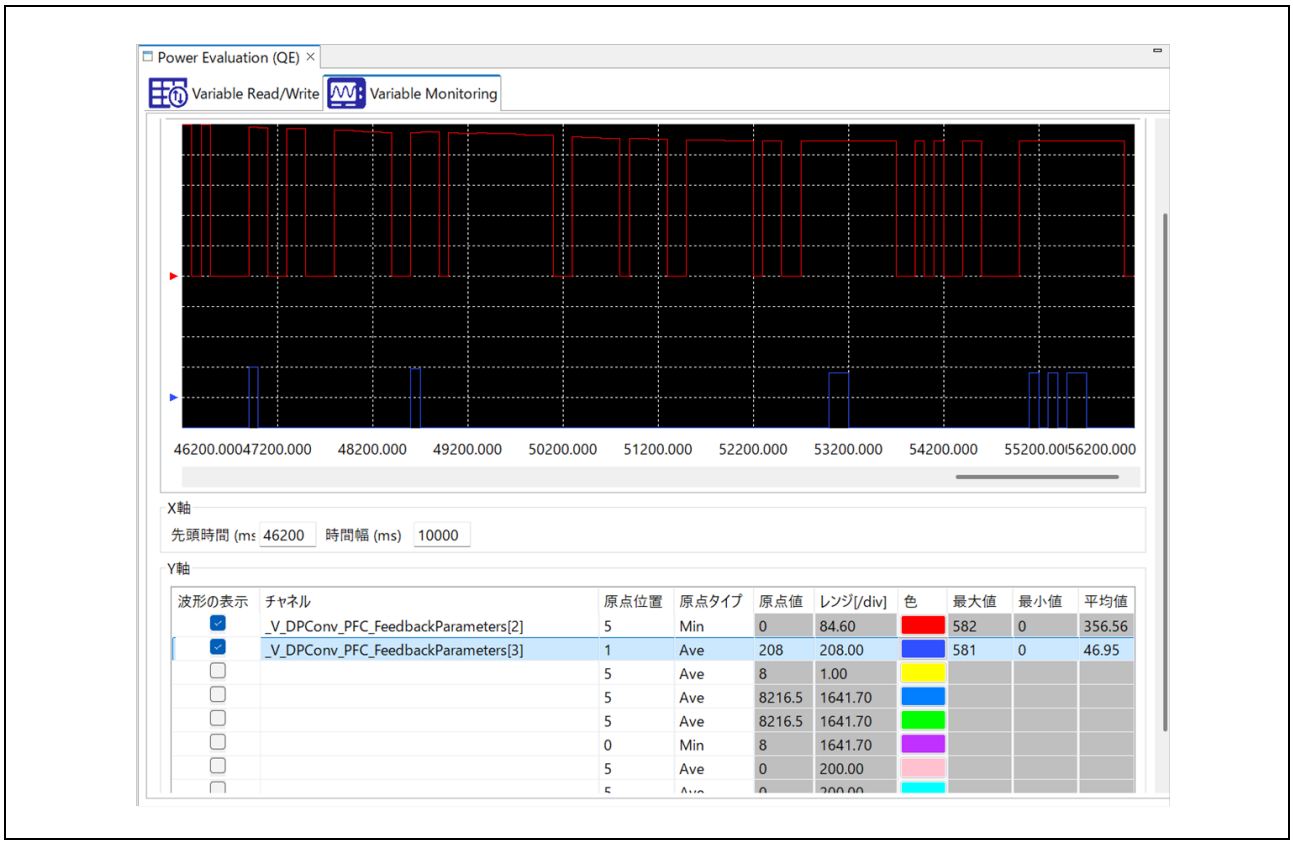

図 10.9 Variable Monitoring

操作手順

1. モニタリング対象変数の選択

タブ内のドロップダウンリスト(チャンネル)から波形表示したい変数を選択します。

複数の変数を同時に選択することも可能です。

2. 取得条件の設定

モニタリング時の時間の設定や出力波形のレンジの設定が可能です。

3. モニタリングの開始

「モニタリングを開始します」を押下すると、変数値の取得と波形表示が開始されます。

10.5 コード生成とプログラム

電源制御の評価時に設定したパラメータを反映したプロジェクトを生成し、ターゲットボードへ書き込むことができます。

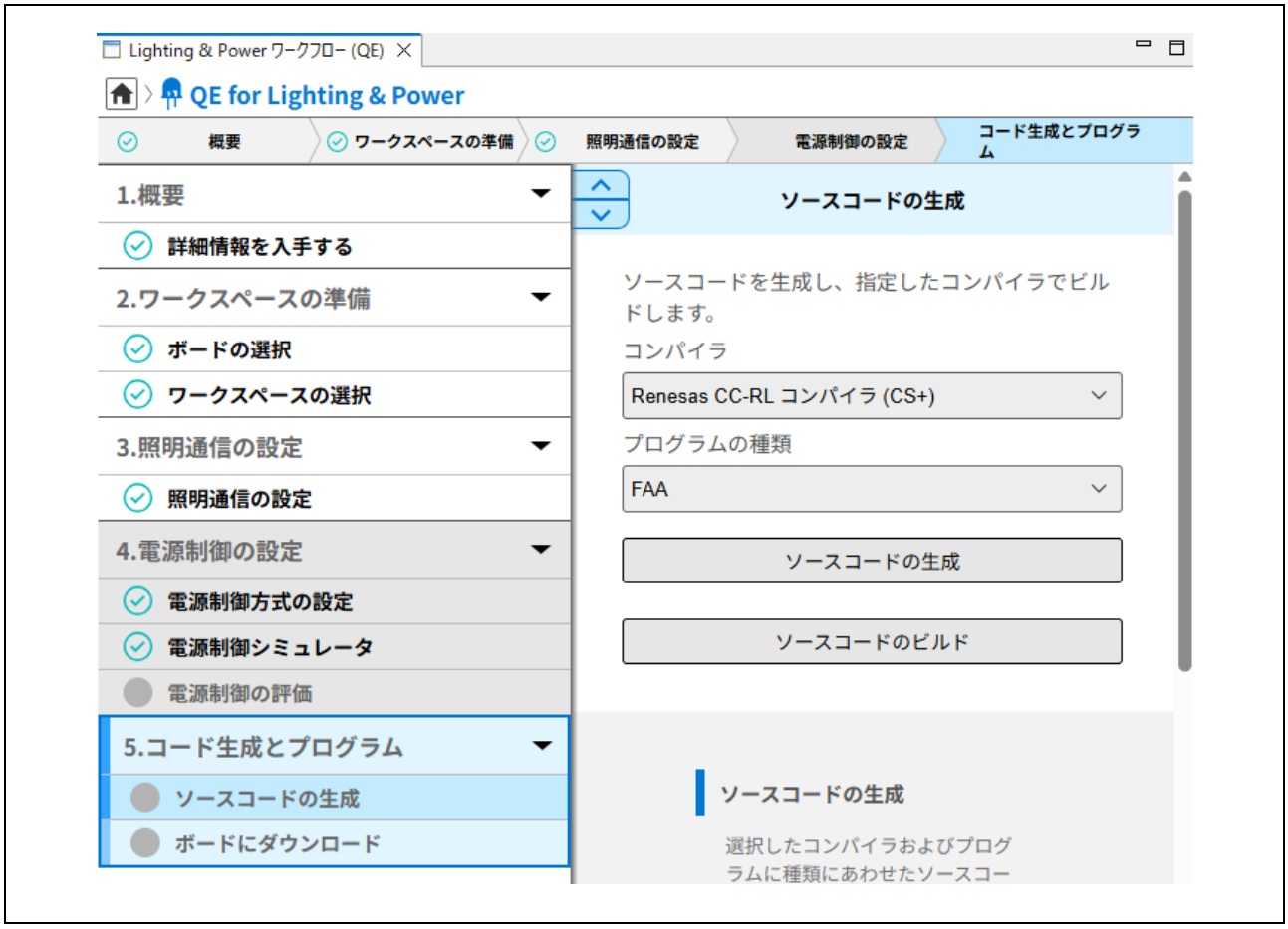

図 10.10 コード生成とプログラム

10.5.1 ソースコードの生成

「ソースコードの生成」ボタンを押下すると以下フォルダにプロジェクトが生成されます。

“<workspace>¥qeLighting_gen¥solution”

10.5.2 ボードにダウンロード

「ボードにダウンロード」ボタンを押下すると E2 Lite 経由でプロジェクトをターゲットボードに書き込みます。

10.6 PFC 制御パラメータ A1、A2 に関する注意事項

PFC PI 制御の係数 A1 及び A2 は、評価プログラム内部でゼロ・ポイント周波数(f_Z)や比例ゲイン(Kp)などから演算される仕様となっています。そのため、Power Configuration ビューやシミュレータで算出・設定した A1 及び A2 の値は、生成されるアプリケーションプログラムには反映されません。

本パラメータを変更する場合は、アプリケーションプログラムに定義されたゼロ・ポイント周波数(fz)、比例ゲイン(Kp)の値を直接変更する必要があります。

10.6.1 Kp と fz の算出方法

Power Configuration ビューやシミュレータで算出・設定した A1、A2 の値から次式でゼロ・ポイント周波数(f_Z)、比例ゲイン(Kp)を算出することができます。

ただし、A1 と A2 の値はアプリケーションプログラムの都合上 65536 で乗算された値となっているため、計算を行う際には 65536 で除算した値で代入してください。

また、対象プログラムにおける PFC のフィードバック周期 T は 400us 固定となります。

$$Kp = \frac{A1 - A2}{2}$$

$$fz = \frac{A1 + A2}{\pi T(A1 - A2)}$$

10.6.2 Kp の fz の反映方法

計算式より求めたゼロ・ポイント周波数(f_Z)と比例ゲイン(Kp) の値は、アプリケーションプログラム内の以下のマクロ値を直接編集することで変更できます。それぞれ整数値で設定してください。

表 10.9　fz、Kp プログラム修正箇所

対象ファイル	マクロ名	デフォルト値	備考
Config_FAA_DPConv.c	PFC_DEFAULT_FZ	2	単位は Hz となります。
	PFC_DEFAULT_KP	1024	アプリケーションプログラムの都合上、4096 で乗算した値を設定してください。 例). Kp が 0.25 の場合 0.25 * 4096 = 1024

11. 注意事項

本書が提供するサンプル・プログラム動作はあくまでもサンプルです。アプリケーション層は、使用目的に合わせた設計変更・評価を行ってください。

12. 参考ドキュメント

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

RL78 ファミリ ユーザーズマニュアル ソフトウェア編 (R01US0015)

(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

(最新版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。)

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2026.3.19	—	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。

© 2026 Renesas Electronics Corporation. All rights reserved.