RX ファミリ

FIR フィルタを使用した周波数帯域判定を行うサンプルプログラム

要旨

本書は、RX ファミリ用 DSP ライブラリの FIR フィルタ API の使用例を示すアプリケーションノートです。

本アプリケーションノートのサンプルプログラムは、図 1 のように構成されています。RX140 に入力されたアナログ信号は、A/D 変換、正規化処理を経て、FIR フィルタ処理を 3 チャンネル分行います。FIR フィルタ処理の結果から入力信号の周波数帯域を判定します。判定結果は LED で表示します。

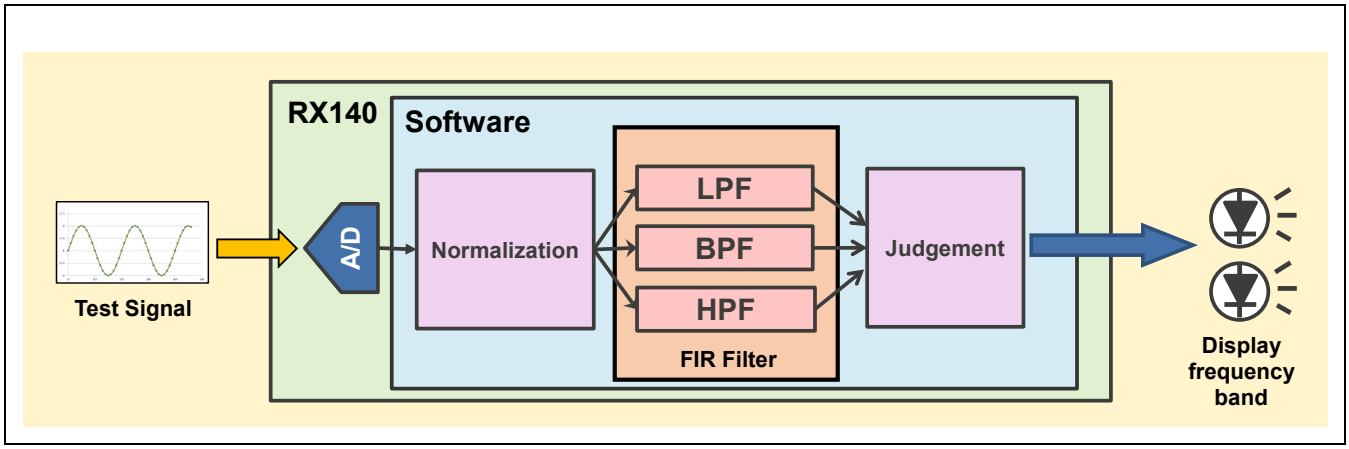


図 1　システム概略図

本サンプルプログラムは、16 ビット固定小数点、32 ビット固定小数点、単精度浮動小数点での FIR フィルタ処理を実装しています。本書はデフォルト設定である 16 ビット固定小数点での処理について説明します。32 ビット固定小数点、単精度浮動小数点でのサンプルプログラムを使用する場合、設定の変更や制約を伴います。

以降、本アプリケーションノートは、サンプルプログラムの環境と手順、サンプルプログラムについて説明します。

動作確認デバイス

RX140 グループ

動作確認ボード

Target Board for RX140

本アプリケーションノート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。

目次

1.　システムの概要

本アプリケーションノートのシステムの概要を図 1.1 に示します。

このシステムでは入力信号のサンプリングから判定結果の出力制御まで RX140 マイコンひとつで行っています。

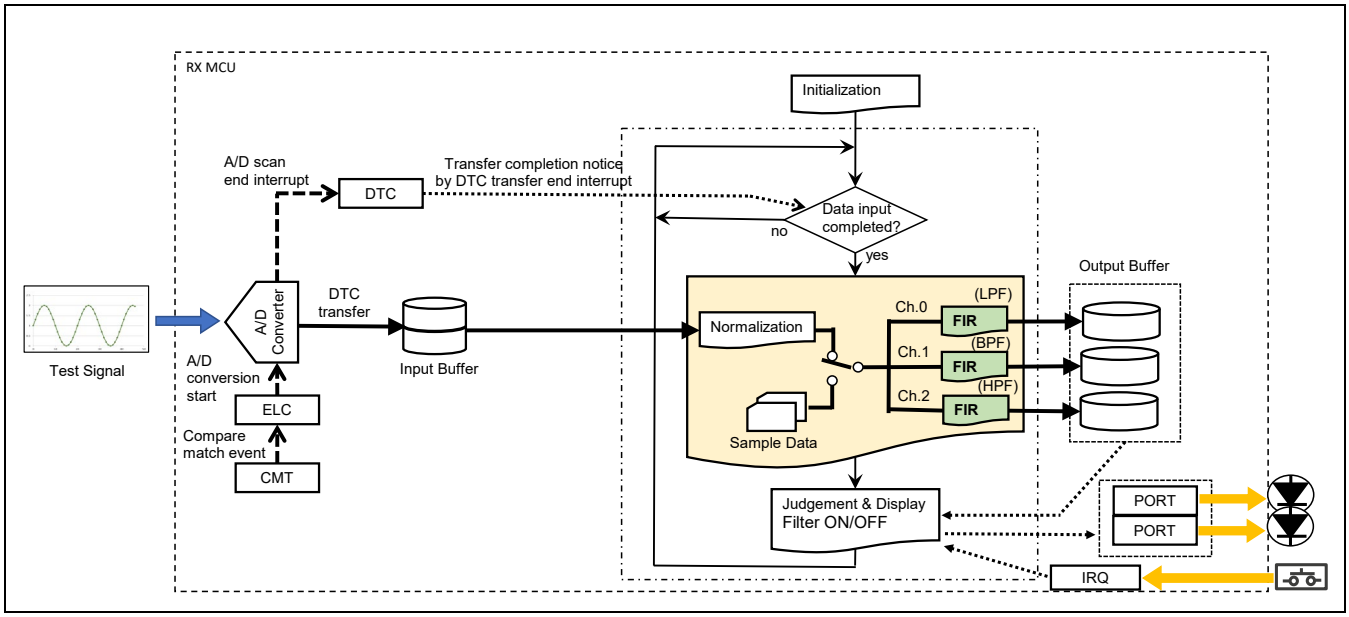


図 1.1　システムの概要

このシステムは次の処理を行います。

1. A/D 変換

　12 ビット A/D コンバータ（S12AD）、コンペアマッチタイマ（CMT）、イベントリンクコントローラ（ELC）を使って約 10kHz のサンプリング周波数で A/D 変換します。まず、CMT にて約 100μs の周期でコンペアマッチイベントを生成、このイベントを ELC にて S12AD の A/D 変換開始トリガとして与えます。変換後のデータは DTC コントローラ（DTC）により入力バッファに転送されます。256 回の DTC 転送が終わると、DTC 転送終了割り込みを発生させます。

2. Normalize

　S12AD で A/D 変換した入力信号は 12 ビット（符号なし）のデータとして入力バッファに格納されます。入力バッファに格納されている 12 ビットのデータを 15 ビット（符号付き）で正規化（バイアス処理とスケーリング）します。

3. FIR フィルタ

　正規化を行ったデータにローパスフィルタ（LPF）、バンドパスフィルタ（BPF）、ハイパスフィルタ（HPF）の FIR フィルタ処理を行い、結果を出力バッファに格納します。図中の Ch.0 が LPF、Ch.1 が BPF、Ch.2 が HPF となっています。フィルタはスイッチ入力により ON/OFF を切り替えることができます。

4. Judge

　入力された信号の周波数成分がどの帯域なのかの判定を行い、LED の点灯パターンを変えることで結果を表示します。

1.1　アプリケーションノートの構成

図 1.2 に本アプリケーションノートの構成を示します。本アプリケーションノート提供 ZIP ファイルを展開すると、ZIP ファイルと同名のフォルダが作成されます。その下の「workspace_fir_example」フォルダは e² studio 形式のプロジェクトを含む e² studio のワークスペースです。プロジェクトのフォルダには、図 1.3 に示すようにサンプルプログラムのソースコードのほか、e² studio の設定ファイル、本アプリケーションノートを含みます。

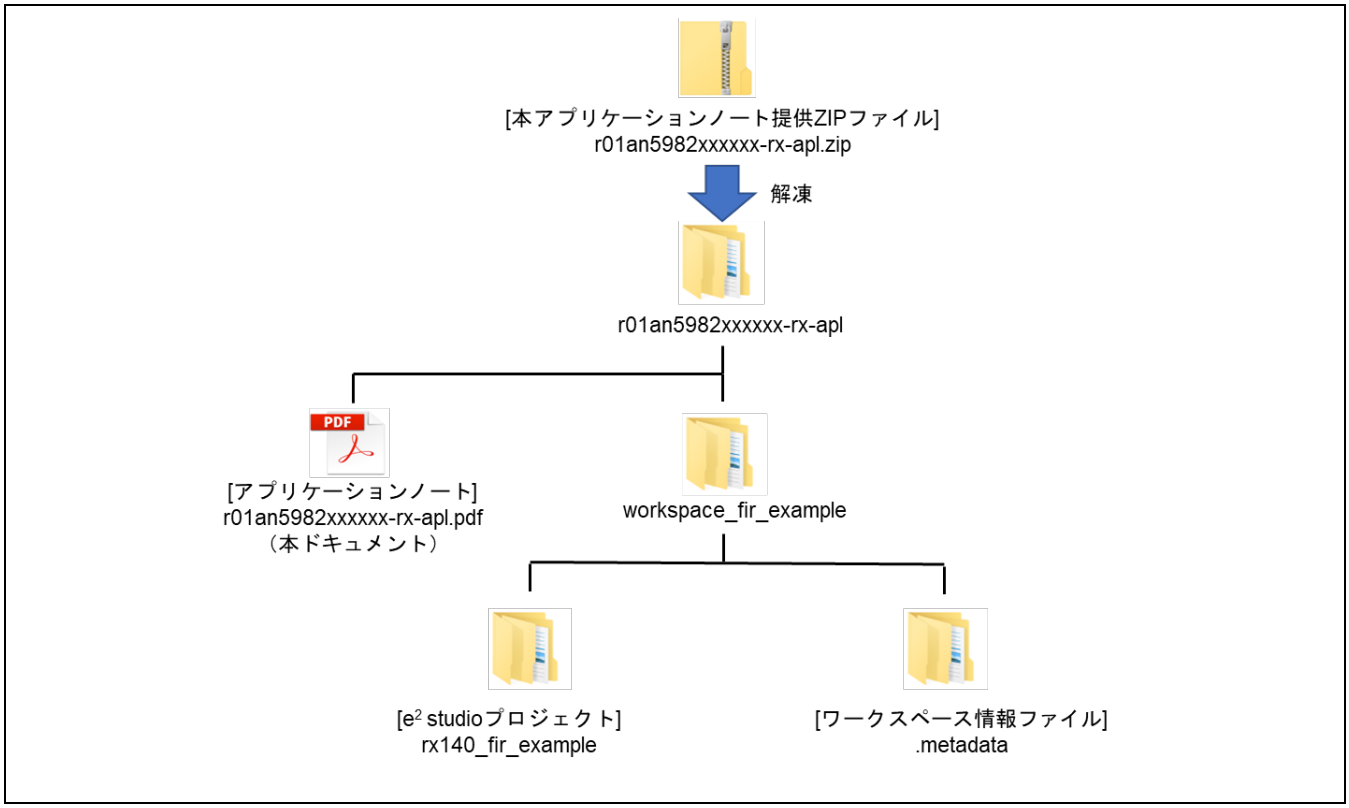


図 1.2　アプリケーションノートの構成

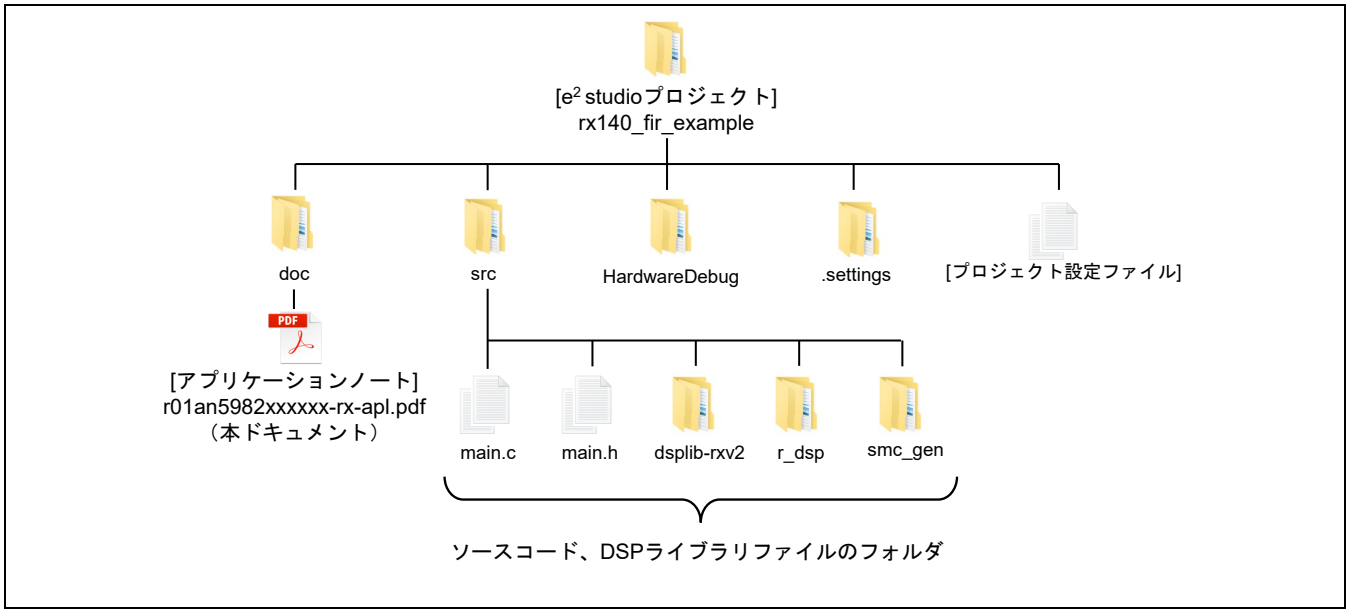


図 1.3 サンプルプロジェクトのフォルダ構成

1.2　サンプルプログラムの構成

サンプルプログラムの構成を図 1.4 に、使用しているソフトウェアモジュールを表 1.1 に示します。FIT モジュールと DSP ライブラリはルネサスの Web サイトから入手可能です。その他の周辺機能のドライバソフトウェアは e² studio のコード生成機能により生成しています。各ソフトウェアモジュールの詳細は、各アプリケーションノートや e² studio のヘルプを参照してください。

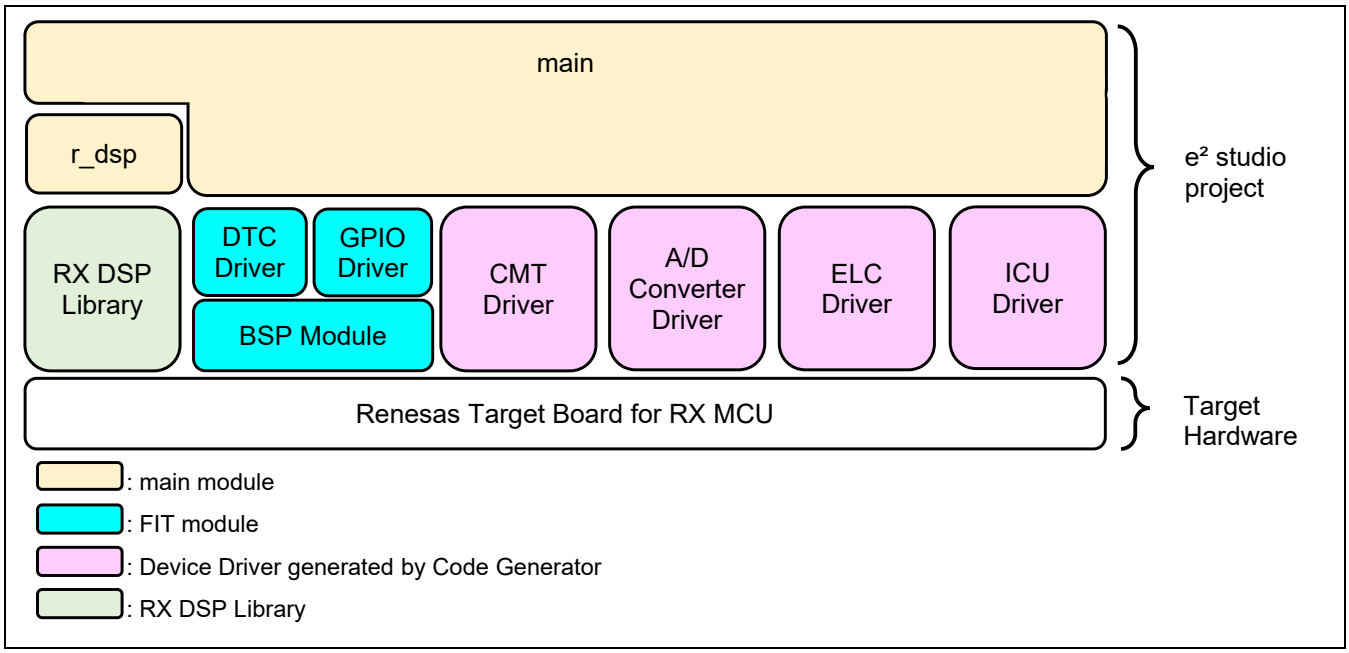


図 1.4 サンプルプログラムの構成

表 1.1 使用ソフトウェアモジュール一覧

モジュール	ドキュメントタイトル	ドキュメント番号	種類
main	-	-	本アプリケーションノートで開発した main 関数を含むモジュール
r_dsp	-	-	本アプリケーションノートで開発した DSP ライブラリの操作モジュール
BSP	RX ファミリ ボードサポートパッケージモジュール Firmware Integration Technology	R01AN1685	FIT モジュール
DTC	RX ファミリ DTC モジュール Firmware Integration Technology	R01AN1819	FIT モジュール
GPIO	RX ファミリ GPIO モジュール Firmware Integration Technology	R01AN1721	FIT モジュール
S12AD	-	-	コード生成機能で生成したドライバ関数
CMT	-	-	
ELC	-	-	
ICU	-	-	
RX DSP Library	RX ファミリ DSP ライブラリ Version 5.0	R01AN4359	DSP ライブラリ

1.3　動作環境

本アプリケーションノートのサンプルプログラムは、表 1.2 の条件で動作を確認しています。

表 1.2 動作確認条件

項目	説明
MCU	R5F51403ADFM（RX140 グループ）
動作周波数	・　HOCO クロック: 48 MHz ・　システムクロック（ICLK）: 48MHz（HOCO クロックを 1 分周する） ・　FlashIF クロック（FCLK）: 48 MHz（HOCO クロックを 1 分周する） ・　周辺モジュールクロック（PCLKB）: 24 MHz（HOCO クロックを 2 分周する） ・　周辺モジュールクロック（PCLKD）: 48 MHz（HOCO クロックを 1 分周する）
動作電圧	3.3V
動作モード	シングルチップモード
プロセッサモード	スーパバイザモード
統合開発環境	ルネサス エレクトロニクス e² studio 2022-01
C コンパイラ	ルネサス エレクトロニクス RX Compiler CC-RX V3.04.00
	コンパイルオプション ・　-lang = c99 ・　-save_acc
エンディアン	・　データ・エンディアン：リトル・エンディアン ・　デバッグツール設定：リトル・エンディアン
iodefine.h	Version 1.00A
サンプルプログラム	Version 1.01
評価ボード	ルネサス エレクトロニクス Target Board for RX140 (RTK5RX1400C00000BJ) ・　搭載 MCU：上記 ・　電源：USB から電源を供給する
ファンクション ジェネレータ	正弦波を出力できるアナログ信号出力端子を備えた信号発生器。 出力信号は GND に対しバイアス 1.65V、振幅 3.0Vpp に設定。 ・　アナログ信号出力(+)を Target board の CN3.60 端子に接続します。 　CN3.60 端子に印可された信号は S12AD の AN000 の入力となります。 アナログ信号出力(GND)を Target board の CN3.40 端子に接続します。 CN3.40 端子は Target board の GND に接続されています。

2.　サンプルプログラムの実行方法

本アプリケーションノートの実行方法を以降に示します。

2.1　　ワークスペースの起動

アプリケーションノートの zip ファイルをパスに日本語が含まれない場所に展開してください。次に e² studio を起動し、ワークスペースの選択画面が表示されたら本アプリケーションノートに同梱されているワークスペース（workspace_fir_example）を選択してください。

e² studio の起動時にワークスペースの選択画面が出ない場合は、e² studio の起動後に

「ファイル」>>「ワークスペースの切り替え(W)」>>「その他(O)」

からワークスペースを選択しなおしてください。

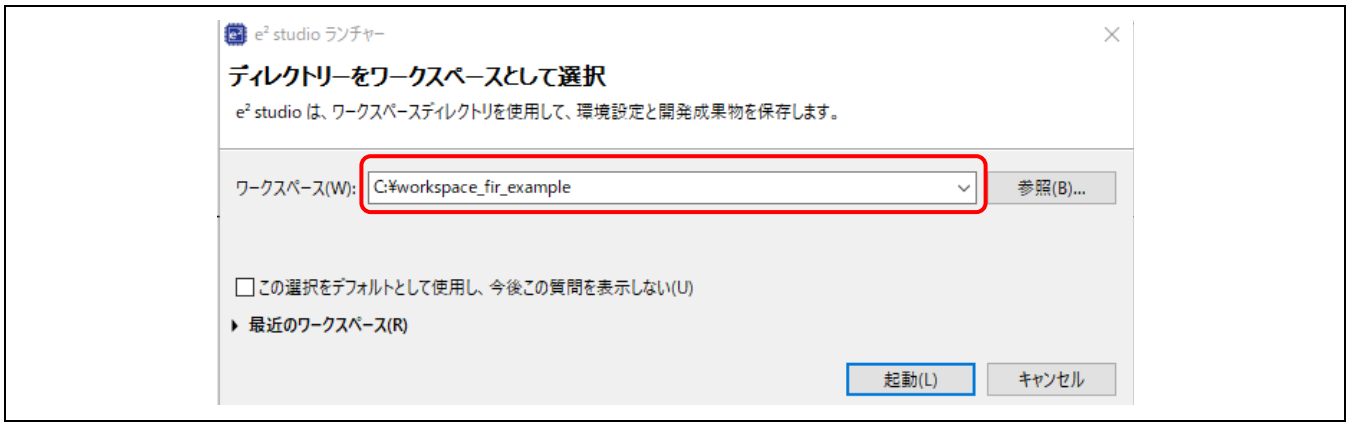


図 2.1　Eclipse ランチャー

2.2　機器の接続

必要な機器を接続します。図 2.2 のように各機器を接続してください。

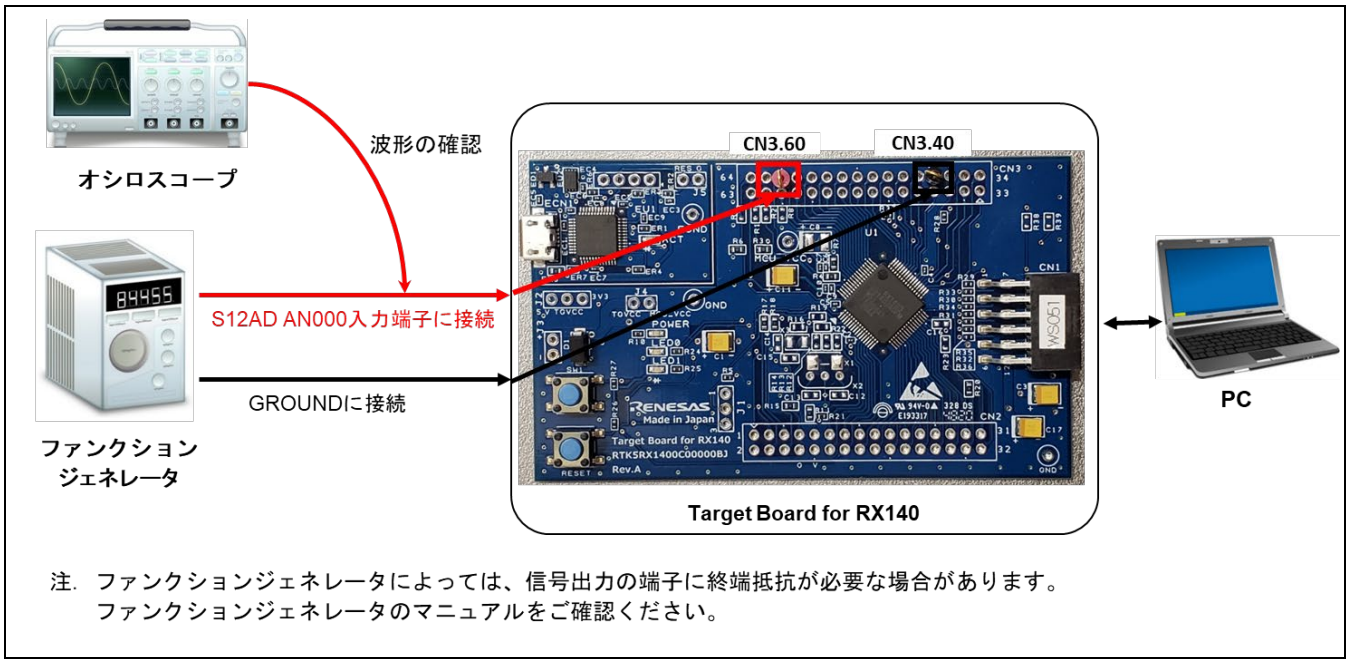


図 2.2 機器の接続

2.3　サンプルプログラムの実行と動作確認

e² studio を使用し、デバッグ接続、サンプルプログラムを実行する手順を以下に示します。

1)  （ビルド）ボタンをクリックしサンプルプログラムをビルド。

2)  （デバッグ）ボタンをクリックしサンプルプログラムをダウンロード。

3)  （リセット）ボタンをクリックし MCU をリセット。

4)  （再開）ボタンでサンプルプログラムを実行。

5) main 関数の先頭で停止するので、再度  ボタンをクリック。サンプルプログラムが実行されます。

2.3.1　LED による周波数帯域表示

サンプルプログラムは、3 つの FIR フィルタの出力をもとに入力信号の周波数帯域を判定します。判定結果を図 2.3 に示すターゲットボード上の LED0 と LED1 の点灯パターンで示します。点灯パターンは約 1 秒周期で更新されます。LED の点灯パターンについては 3.4.5 FIR フィルタ処理結果による周波数帯域判定を参照してください。

また、プログラム実行中に Target Board のスイッチ(SW1)を押下すると FIR フィルタの ON/OFF を切り替えることができます。

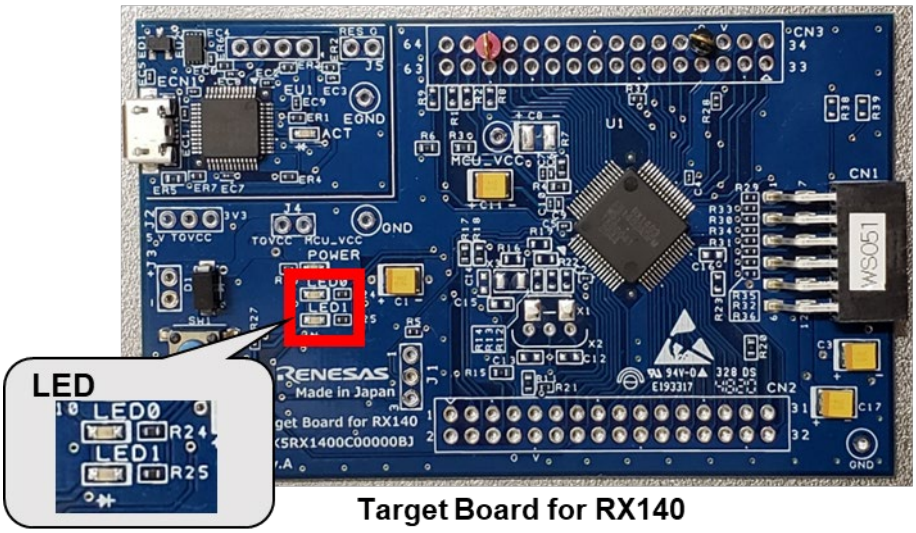


図 2.3 Target Board for RX140 の LED

2.3.2　e² studio の機能を使った FIR フィルタ動作のモニタ

e² studio には多くのデバッグ機能が搭載されています。本アプリケーションノートでは、Waveform レンダリング機能を使用して FIR フィルタの出力信号などをモニタするパースペクティブ（e² studio の画面構成）を用意しています。

デバッグ接続後、図 2.4 に示す「FIR_Filter」ボタンをクリックしてください。パースペクティブを切り替えることができます。

なお、このパースペクティブはワークスペースの情報に含まれておりますので、2.1 ワークスペースの起動に記載のとおり、本アプリケーションノートに同梱しているワークスペースをご使用ください。

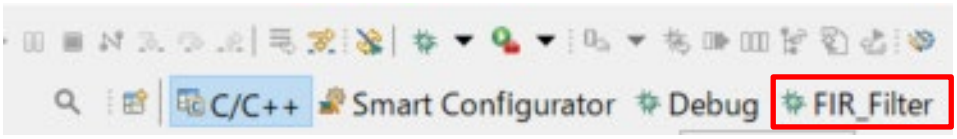


図 2.4　パースペクティブの切り替え

FIR_Filter パースペクティブ（e² studio の画面構成）は図 2.5 のとおりです。

図 2.5　FIR_Filter パースペクティブ

「メモリー」ビューは「リアルタイムリフレッシュ」を有効にすることで、周期的に更新されます。（図 2.6）

図 2.6　リアルタイムリフレッシュの有効

FIR_Filter パースペクティブで使用している「メモリー」ビューの「Waveform レンダリング機能の使用方法については 5.1 e² studio による信号処理のモニタ、および e² studio のヘルプを参照してください。

ヘルプ(H) →　「e² studio ユーザーガイド」→「デバッグに関する機能」→「ビュー」

- 「メモリー」→「Waveform メモリー・レンダリング」

2.4　定義変更可能な設定

本システムでユーザが変更可能な設定を表 2.1、表 2.2 に示します。

表 2.1 変更可能な設定（main.h）

機能/定義名	説明	デフォルト値
入力データの切り換え		
SAMPLE_DATA_MODE	FIR フィルタ処理を行う信号ソースを選択します。 0: 外部入力信号 1: サンプルデータ	0
SELECT_SAMPLE_DATA	使用するサンプルデータを選択します。 設定値: 1: 156.25Hz が支配的な 3 つの正弦波をミックスした信号 2: 1250Hz が支配的な 3 つの正弦波をミックスした信号 3: 4000Hz が支配的な 3 つの正弦波をミックスした信号	1
スリープモードの実行		
SLEEP_MODE	スリープモードを実行するかの選択をします。 0: スリープモードを実行しない 1: スリープモードを実行する	0

表 2.2 変更可能な設定（r_dsp_fir_config.h）

機能/定義名	説明	デフォルト値
FIR フィルタの演算語長の選択		
FIR_OPERATION*	FIR フィルタの演算語長を選択します。 設定値: 0: FIR_I16I16 1: FIR_I32I32 2: FIR_F32F32	0
FIR フィルタのチャンネル数		
NUM_FIR_PROC	FIR フィルタのチャンネル数を設定します。 設定値: 1, 2, 3	3
FIR フィルタの設定値		
FIR_UNIT_INPUT*	FIR フィルタの処理単位サンプル数を設定します。 設定範囲: 16 - 1024 (2 のべき乗)	256
FIR_TAPS*	FIR フィルタのタップ数を設定します。 設定範囲: 64 - 256 (2 のべき乗)	64

* サンプルプログラムは RX140 をターゲットとしてデフォルト設定を決めています。これらの設定値を変更した場合、RAM が不足したり、CPU 処理が過負荷となり動作しなくなる場合があります。必要な資源の見積もりを十分に行い、設定の変更を行ってください。

3. サンプルプログラムの説明

3.1 サンプルプログラムの概要

本サンプルプログラムは、表 3.1 に示す処理から構成されています。

表 3.1 処理の役割

処理	役割
main 処理	・ 周辺機能と FIR フィルタ処理の初期化 ・ 最初の DTC 転送の開始 ・ DTC 転送終了割り込み処理での DTC 転送先の通知 ・ 入力データの正規化処理 ・ FIR フィルタ処理 ・ FIR フィルタ処理結果の判定と LED 点灯パターンの表示処理
DTC 転送終了割り込み処理	・ 2 回目以降の DTC 転送の設定と転送の開始

3.2　処理シーケンス

サンプルプログラムの処理シーケンスは主に main 処理、DTC 転送終了割り込み処理があります。main 処理、DTC 転送終了割り込み処理のシーケンスを図 3.1 に示します。

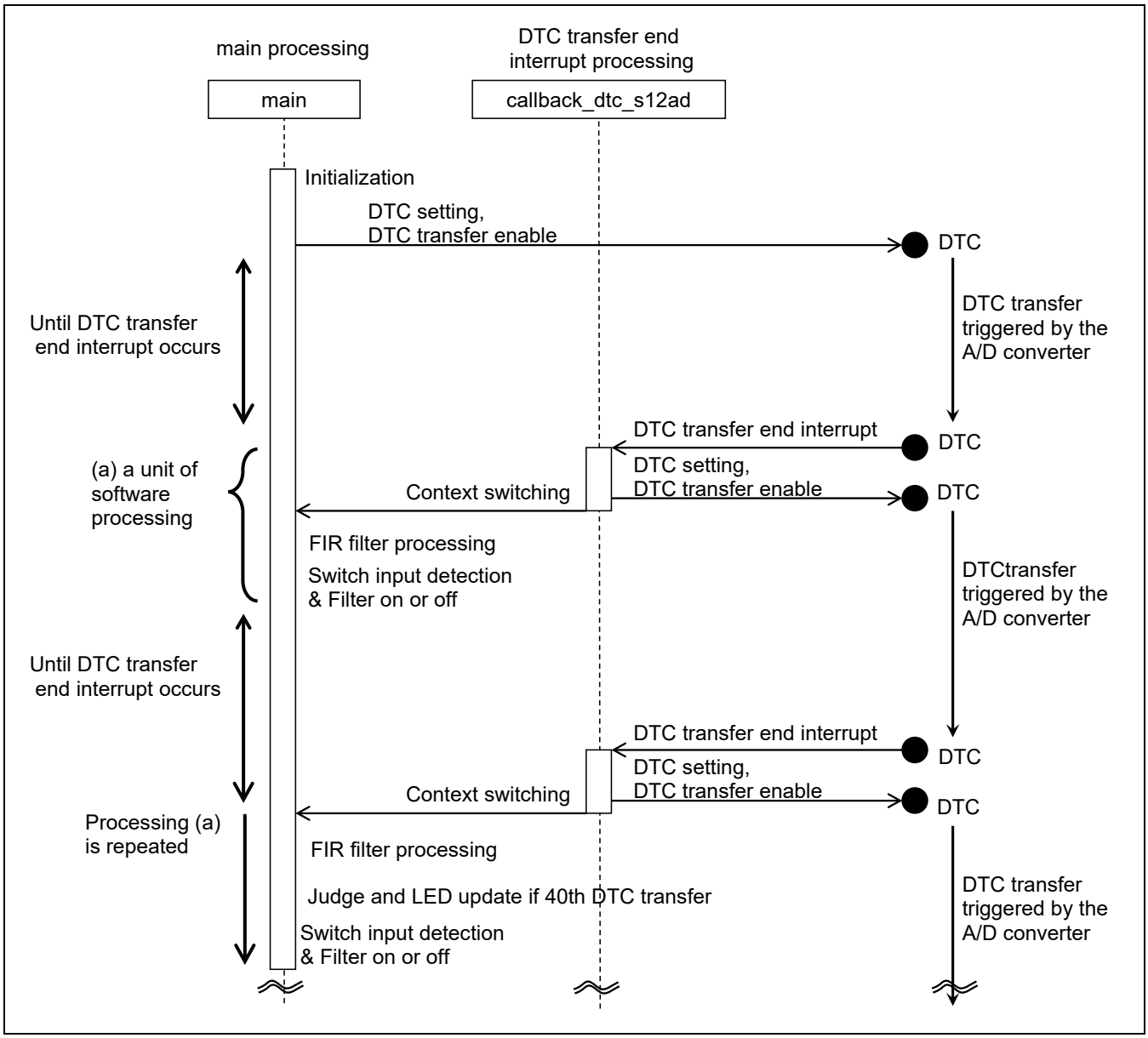


図 3.1 サンプルプログラムの処理シーケンス

（main 処理、DTC 転送終了割り込み処理）

図中、最初の DTC 転送は main 処理で A/D 変換完了による転送を許可します。指定サンプル数の DTC 転送の終了による DTC 転送終了割り込み要求が発生します。これをトリガとして DTC 転送終了割り込み処理と main 処理を順次実行します。以後同じ処理を繰り返し実行します。

40 回目(約 1 秒)の DTC の転送終了割り込み処理が発生すると周波数帯域の判定を行い、入力されている信号の周波数帯域を 2 つの LED で表示します。

3.3　処理フロー

サンプルプログラムの処理フローを図 3.2 に示します。

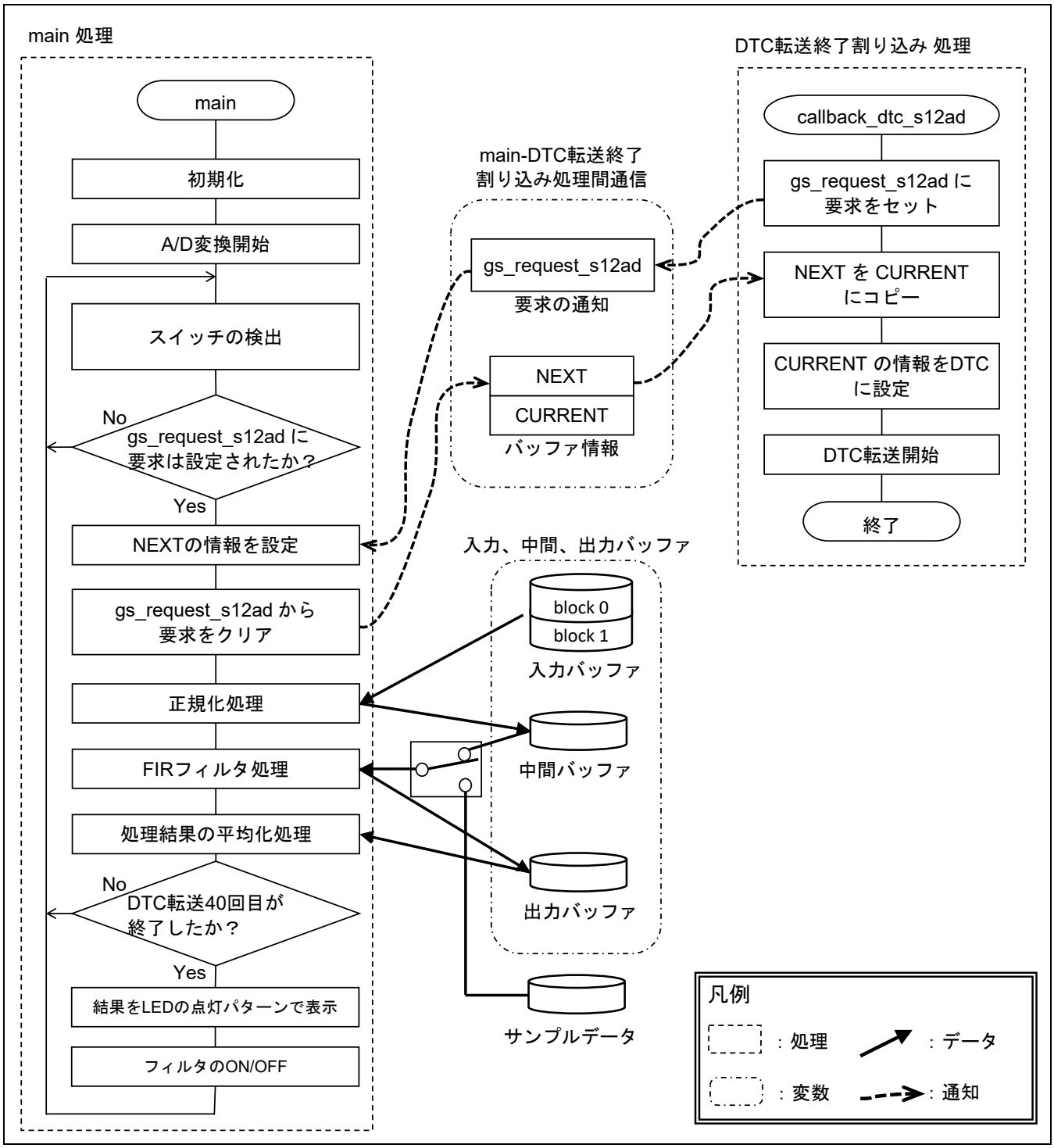


図 3.2 処理フロー

図 3.2 の要素について以下に説明します。

- main 処理
  最初に初期化を行います。
  初期化後、以下の処理を繰り返します。
  ・　スイッチ（SW1）操作の検出
  ・　DTC により A/D 変換結果がバッファに格納されたか判定
  　A/D 変換結果が格納されていた場合、以下を実行します。
  　・次の DTC 転送先の設定と FIR フィルタ処理を行います。
  　・スイッチ操作が検出されていた場合、FIR フィルタの ON/OFF の切り換えを行います。
- DTC 転送終了割り込み処理
  次の DTC 転送先の更新を main 処理に要求し、今回の DTC 転送を開始させます。
- main-DTC 転送終了割り込み処理間通信
  main 処理から DTC 転送終了割り込み処理への次の DTC 転送先の通知、DTC 転送終了割り込み処理から main 処理への次の DTC 転送先の更新の要求は、本処理間通信の変数を介して行います。**表** 3.2 に各変数について説明します。
- 入力バッファ
  DTC が A/D 変換結果を格納し、FIR フィルタ処理が入力データとして読み出しを行います。
  CPU（FIR フィルタ処理）と DTC のアクセス競合を避けるため 2 ブロック構成とします。CPU、DTC それぞれがアクセスするバッファは、DTC 転送終了割り込みをトリガとして切り替えます。図 3.3 に示すように DTC 転送終了割り込みをトリガとして切り替えます。図 3.4 に入力バッファの構成を示します。
- 中間バッファ
  正規化処理の結果を格納します。図 3.4 に中間バッファの構成を示します
- 出力バッファ
  FIR フィルタ処理の処理結果を格納します。FIR フィルタ処理は 3 チャンネル分行います。図 3.4 に出力バッファの構成を示します
- サンプルデータ
  マクロ定義 SAMPLE_DATA_MODE が“1”の場合に使用します。外部信号入力のデータの代わりにサンプルデータを FIR フィルタ処理の入力にします。

表 3.2 main-DTC 転送終了割り込み処理間通信の変数

種類		説明
(a)　要求の通知		DTC 転送終了割り込み処理 が、(b)のバッファの情報の更新を main 処理に要求するための変数。DTC 転送終了割り込み処理が要求をセットする。セットされている場合 main 処理は、(b)　の NEXT 面の情報を更新し、要求をクリアする。 初期化時、main 処理がクリアする。
(b)　バッファ情報		DTC 転送先のバッファの先頭アドレスとデータ数を格納する変数。 処理間のアクセス競合を避けるため NEXT と CURRENT の 2 面で構成する。初期値は 2 面共 main 処理が設定する。
	NEXT	main 処理が、(a)に応じ先頭アドレスとデータ数を格納する面。 以下に初期値を示す。 ・　先頭アドレス：　Input Buffer の block 1 の Data 0 ・　データ数：256
	CURRENT	DTC 転送終了割り込み処理が、DTC に設定する先頭アドレスとデータ数を参照する面。DTC 転送終了割り込み処理自身が NEXT 面の情報を CURRENT 面にコピーする。 以下に初期値を示す。 ・　先頭アドレス：　Input Buffer の block 0 の Data 0 ・　データ数：256

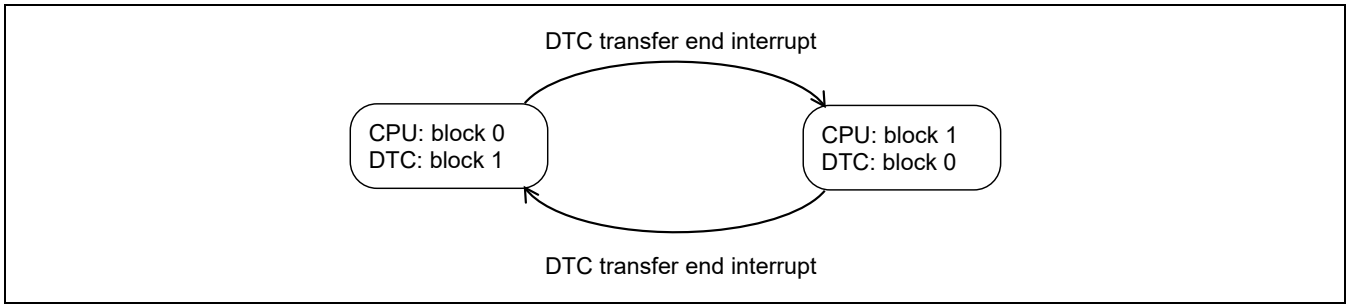


図 3.3　CPU と DTC がアクセスする入力バッファの切り替え

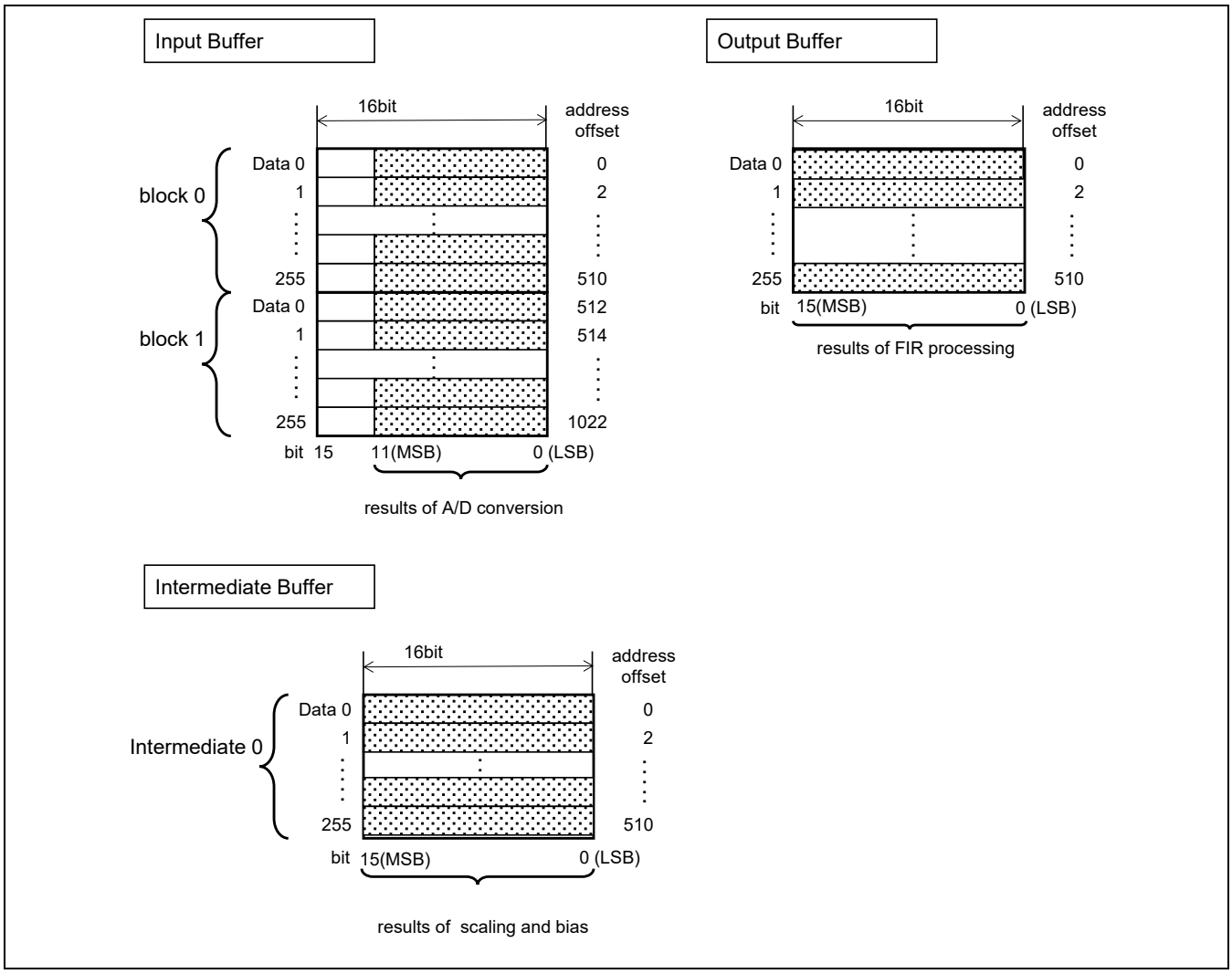


図 3.4　入力、中間、出力バッファ

3.4　詳細

3.4.1　初期化

サンプルプログラムは、下記の順序で初期化を行います。

1. 周辺機能の初期化
   CMT、S12AD、ELC、を初期化する。
2. FIR フィルタ処理の初期化
   FIR フィルタ処理を初期化し、各チャンネルに係数を設定する。
3. main-DTC 転送終了割り込み処理間通信の変数の初期化

   表 3.2 の main-DTC 転送終了割り込み処理間通信の変数を初期化する。
4. DTC 転送を許可する

   起動要因、転送モード、転送元アドレス、転送先アドレス、転送データ数等を設定し、DTC 転送を許可する。

この後、A/D 変換動作を開始させると、DTC 転送がはじまります。

3.4.2　正規化処理

サンプルプログラムでは、図 3.5 に示す正規化処理（バイアス処理とスケーリング）を行っています。

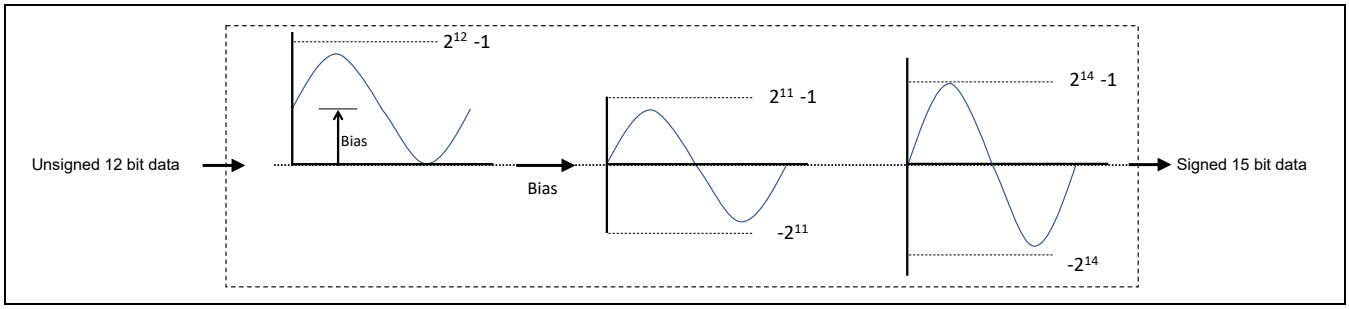


図 3.5 正規化処理

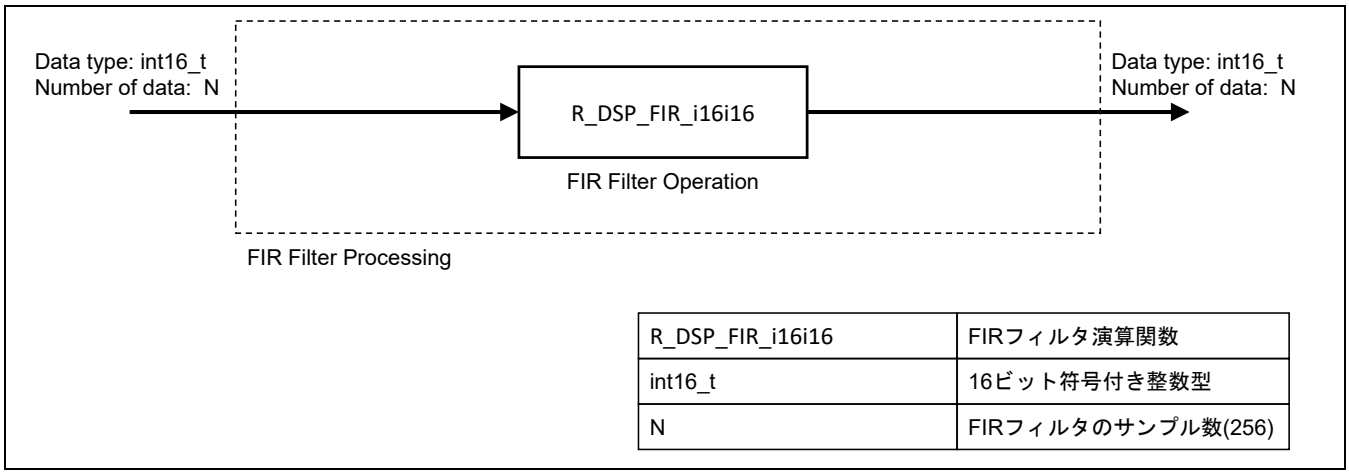
- 正規化処理の構成
  S12AD から入力されるデータは 12 ビット（符号なし）のため、FIR フィルタ処理において十分な演算結果から得られるよう 15 ビット（符号付き）に正規化します。正規化処理の内容はバイアス処理とスケーリングです。
- 32 ビット整数型、単精度浮動小数点での正規化処理について

  32 ビット整数型の場合、正規化処理により A/D 変換出力した 12 ビット（符号なし）整数を 31 ビット（符号付き）に変換します。値域は -2^{30} ～ $2^{30}-1$ となります。

  単精度浮動小数点の場合、正規化処理により A/D 変換出力した 12 ビット（符号なし）整数を単精度浮動小数点に変換します。値域は-1.0~1.0 となります。

3.4.3　FIR フィルタ処理

サンプルプログラムでは、**図** 3.6 に示す DSP ライブラリの API 関数を使って入力信号にフィルタ処理を行っています。



R_DSP_FIR_i16i16	FIRフィルタ演算関数
int16_t	16ビット符号付き整数型
N	FIRフィルタのサンプル数(256)

図 3.6 FIR フィルタ処理のデータフロー

- FIR フィルタ処理を構成する API
  FIR フィルタ処理は、FIR フィルタ演算関数 R_DSP_FIR_i16i16 で行います。
  R_DSP_FIR_i16i16 は、関数に与えるフィルタ係数に応じた演算結果を出力します。サンプルプログラムでは、LPF、BPF、HPF の 3 つのフィルタ特性を Ch.0, Ch.1, Ch.2 に割り当てています。
- 入力、出力データについて

  サンプルプログラムでは、FIR フィルタ処理において 256 サンプルのデータを int16_t 型で入力し、同じく int16_t 型で出力します。

API 仕様の詳細は「RX DSP ライブラリ API Version 5.0 ユーザーズマニュアル ソフトウェア編」を参照してください。

3.4.4　FIR フィルタ処理の結果の平均化処理

出力バッファ（gs_output_buffer）に格納された FIR フィルタ処理の結果に対して、フィルタごとの平均値を取得します。256 サンプル毎に出力バッファに格納されたデータに対して以下のように平均化処理を行います。出力バッファに格納されたデータは符号付きデータのため、絶対値で計算を行います。この処理は main.c の get_average_value 関数で実行されます。

この処理は 3 つのチャンネルの全ての FIR フィルタの処理結果に対して行います。

・　平均化 $= \sum_{i=0}^{255} |gs\_output\_buffer[i]| / 256$

3.4.5　FIR フィルタ処理結果による周波数帯域判定

周波数帯域判定のイメージを図 3.7 に示します。

入力信号の周波数帯域の判定は、3 チャンネル分の FIR フィルタ処理と平均化処理を使って 1 秒ごとに行います。1 秒間の平均化処理の回数は 40 回です。（サンプリング周波数が 10kHz、処理単位が 256 サンプルであるため）

40 回分の平均化処理の結果を全て加算し、Ch.0、Ch.1、Ch.2 のそれぞれで値を比較して最も値が大きいフィルタの周波数帯域を結果として LED の点灯パターンで表示します。比較処理は main.c の evaluate_max_values で行い、LED 点灯処理は main.c の display_led で行います。

点灯パターンはフィルタの特性によって表 3.3 のようになります。

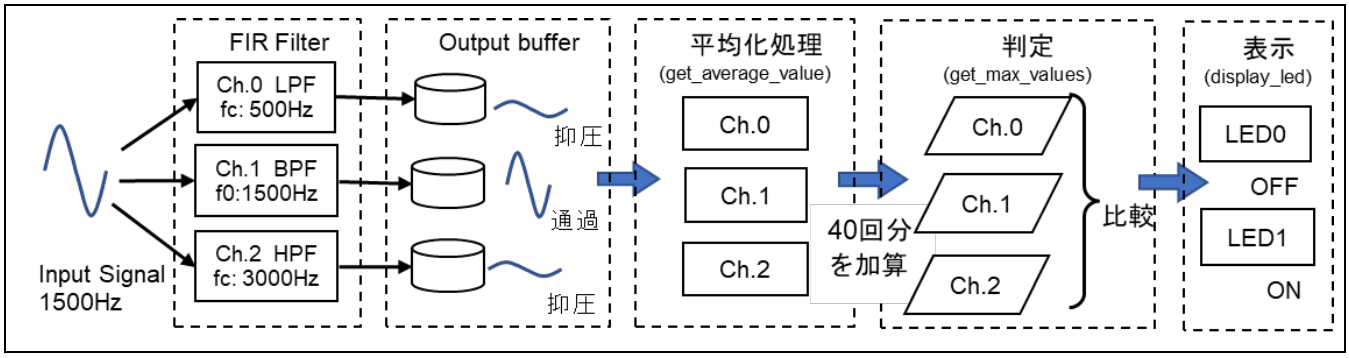


図 3.7 周波数帯域判定のイメージ

表 3.3 FIR フィルタ処理による LED の点灯パターンの結果表示

	LED0	LED1
Ch0(LPF)	ON	OFF
Ch1(BPF)	OFF	ON
Ch2(HPF)	ON	ON
判定不能	OFF	OFF

3.5　ファイル構成

サンプルプログラムに実装したソフトウェアモジュールのファイル構成を表 3.4 に示します。また、各ソースファイルの関数を表 3.5～表 3.9 に示します。その他のモジュールの詳細は各アプリケーションノートを参照してください。

表 3.4 ソフトウェアモジュールのファイル構成

モジュール/ファイル		説明
Main		サンプルプログラムのメイン処理
	main.c	main 処理と DTC 転送終了割り込み処理など
	main.h	main.c のヘッダファイル
r_dsp		FIR フィルタ関連処理
	r_normalize.c	正規化処理
	r_normalize.h	r_normalize.c のヘッダファイル
	r_dsp_fir_i16i16.c	16 ビット符号付き整数での FIR フィルタ処理の初期化とフィルタ処理
	r_dsp_fir_i32i32.c	32 ビット符号付き整数での FIR フィルタ処理の初期化とフィルタ処理
	r_dsp_fir_f32f32.c	32 ビット浮動小数点での FIR フィルタ処理の初期化とフィルタ処理
	r_dsp_fir.h	r_dsp_fir_i16i16.c、 r_dsp_fir_i132i32.c、r_dsp_fir_f32f32.c のヘッダファイル
	r_dsp_fir_config.h	FIR フィルタの設定ファイル
	r_coef_lpf_*.c	LPF の係数ファイル カットオフ周波数:500 [Hz] *:演算語長によって i16、i32、f32 に読み替えてください
	r_coef_bpf_*.c	BPF の係数ファイル 中心周波数:1500 [Hz] *:演算語長によって i16、i32、f32 に読み替えてください
	r_coef_hpf_*.c	HPF の係数ファイル カットオフ周波数:3000 [Hz] *:演算語長によって i16、i32、f32 に読み替えてください
	r_coef_flat_*.c	フラットな特性の係数ファイル *:演算語長によって i16、i32、f32 に読み替えてください
	r_wave_sample_*_lpf.c	LPF チェック用サンプル波形のファイル 3 つの正弦波をミックスした信号（156.25Hz が支配的） *:演算語長によって i16、i32、f32 に読み替えてください
	r_wave_sample_*_bpf.c	BPF チェック用サンプル波形のファイル 3 つの正弦波をミックスした信号（1250Hz が支配的） *:演算語長によって i16、i32、f32 に読み替えてください
	r_wave_sample_*_hpf.c	HPF チェック用サンプル波形のファイル 3 つの正弦波をミックスした信号（4000Hz が支配的） *:演算語長によって i16、i32、f32 に読み替えてください

表 3.5 main.c ファイルの関数一覧

関数名	説明
main	・ 周辺デバイスを初期化 ・ 最初の DTC 転送の開始 ・ DTC 転送終了割り込み処理 に次の DTC 転送先を通知 ・ FIR フィルタ処理の実行 ・ 処理結果の判定と LED への表示
set_buf_info	main-DTC 転送終了割り込み処理間通信の変数に次の DTC 転送先アドレスとデータ数を設定
dtc_init	S12AD チャネル 0 を転送元として DTC を初期化
dtc_start	DTC に転送情報を格納して、DTC を開始
callback_dtc_s12ad	main 処理により DTC モジュールに登録されるコールバック関数。 ・ DTC の DTC 転送終了割り込み処理 ・ 次の転送先を DTC に設定し、DTC 転送を許可 ・ 次の転送先の情報を main 処理に要求
get_average_value	出力バッファに格納されたサンプルの平均値を取得
evaluate_max_values	get_average_value が出力した 3 つの平均値を比較し最大値を判定
display_led	evaluate_max_values の結果に基づき LED の点灯パターンを更新

表 3.6 r_normalize.c ファイルの関数一覧

関数名	説明
R_Normalize_Operation	・ 正規化処理

表 3.7 r_dsp_fir_i16i16.c ファイルの関数一覧

関数名	説明
R_DSP_FIR_Init	・ FIR フィルタ処理の初期化
R_DSP_FIR_Change_Coef	・ FIR フィルタ係数の設定
R_DSP_FIR_Operation	・ FIR フィルタ処理の実行

表 3.8 r_dsp_fir_i32i32.c ファイルの関数一覧

関数名	説明
R_DSP_FIR_Init	・ FIR フィルタ処理の初期化
R_DSP_FIR_Change_Coef	・ FIR フィルタ係数の設定
R_DSP_FIR_Operation	・ FIR フィルタ処理の実行

表 3.9 r_dsp_fir_f32f32.c ファイルの関数一覧

関数名	説明
R_DSP_FIR_Init	・ FIR フィルタ処理の初期化
R_DSP_FIR_Change_Coef	・ FIR フィルタ係数の設定
R_DSP_FIR_Operation	・ FIR フィルタ処理の実行

4. 注意事項

4.1 HOCO クロックの誤差による FIR フィルタ結果の誤差

Target Board 上の RX140 は HOCO をクロックソースとして動作します。HOCO は最大±2%の誤差を含んでおり、サンプルプログラムの A/D 変換処理のサンプリング周波数の誤差となります。サンプリング周波数の誤差により FIR フィルタのカットオフ周波数等の誤差となる場合があります。より正確な FIR フィルタ処理が必要なシステムに応用する場合、RX140 のクロックソースとして精度の高い発振子を使用し、サンプルプログラムのクロック設定の変更を行ってください。

4.2 エイリアシング

本書で使用する評価ボードは、RX140 の A/D 変換に入力する信号のエイリアシングへの対策を行っていません。サンプリング周波数の 1/2 を超える周波数の信号を A/D 変換に入力するとエイリアシングが発生します。本書を参考にシステムを設計する場合、必要に応じアンチエイリアシングフィルタを RX140 に外付けするなどの対応を行ってください。

5.　参考

5.1　e² studio による信号処理のモニタ

e² studio の Waveform レンダリング機能を活用することで、RX140 に取り込まれた入力信号や FIR フィルタ処理結果をモニタすることができます。

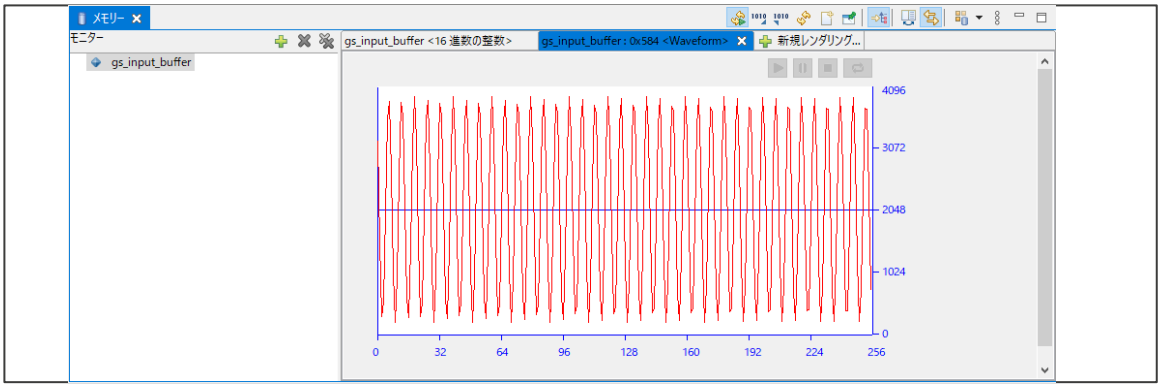


図 5.1 Waveform レンダリングの表示例（入力バッファに格納されたデータ）

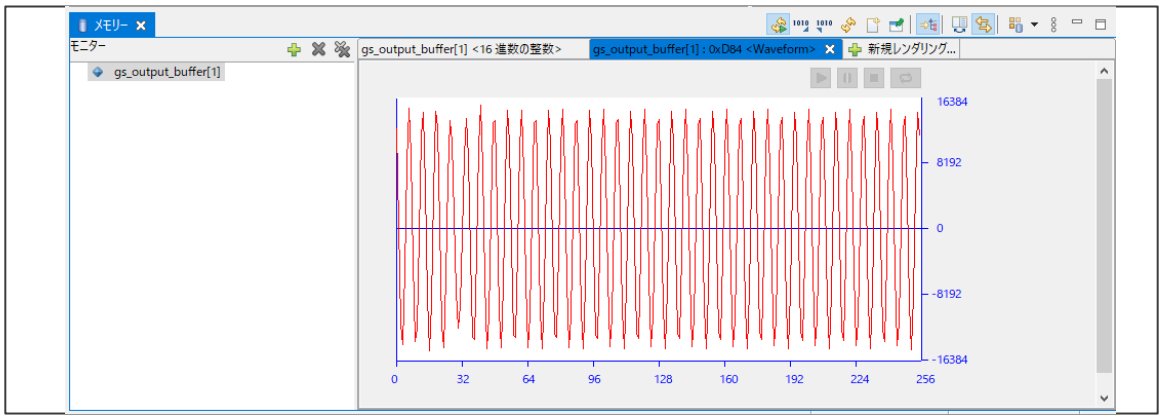


図 5.2 Waveform レンダリングの表示例（FIR フィルタ処理による周波数振幅特性）

RX140 とのデバッグ接続後、「ウィンドウ(W)」→「ビューの表示(V)」から「メモリー」ビューを選択して表示します。「メモリー」ビューが表示されたら、モニタする変数として入力バッファ（gs_input_buffer）を指定します。追加されたら同様に出力バッファ（gs_output_buffer）も指定します。

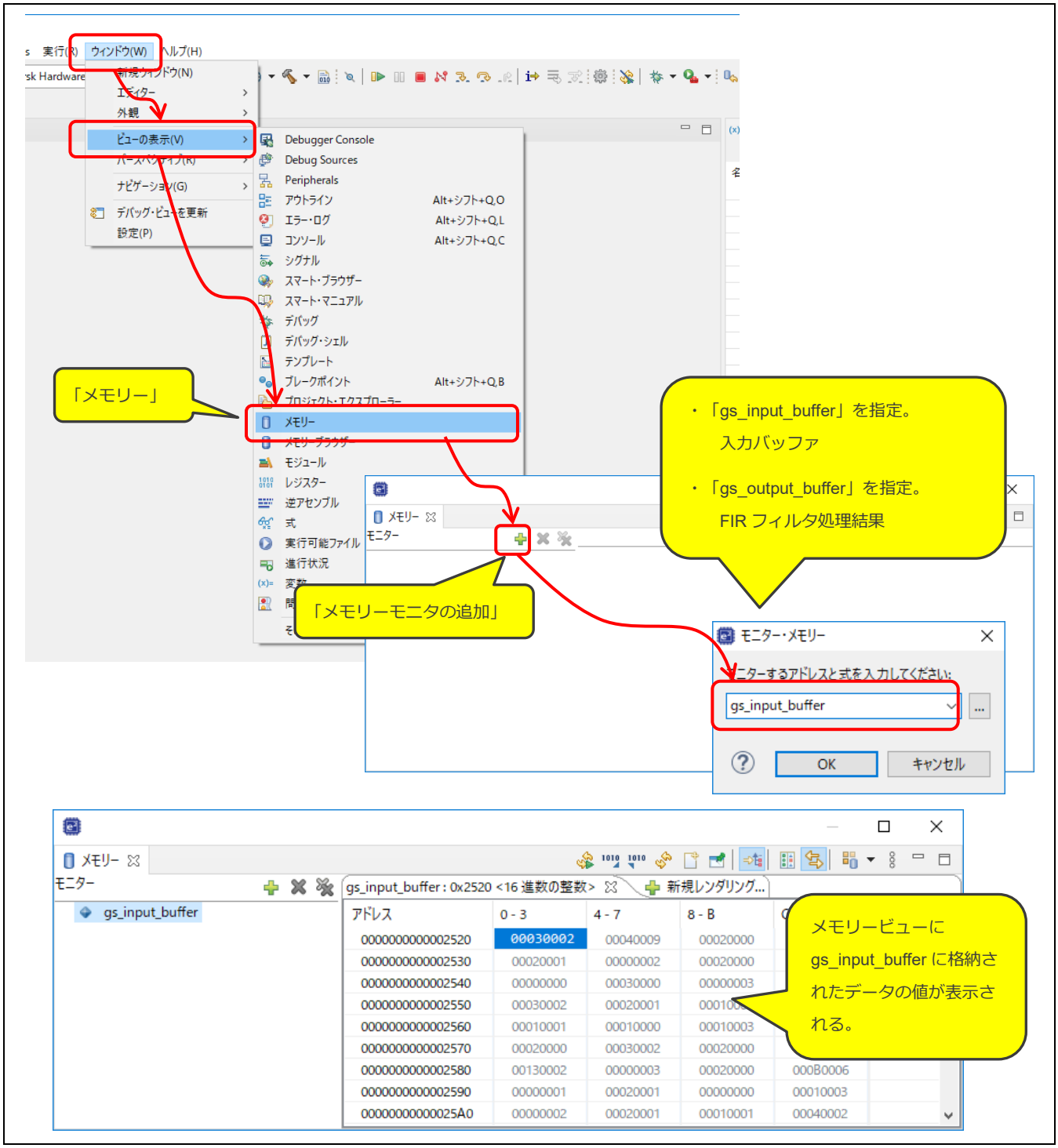


図 5.3 メモリービューの表示手順

次に、グラフ表示したい変数を選択し、「新規レンダリング」→「Waveform」→「レンダリング」の順にクリックします。Waveform プロパティーが表示されますので、変数に応じて必要な設定を行い、最後に「OK」をクリックするとグラフが表示されます。

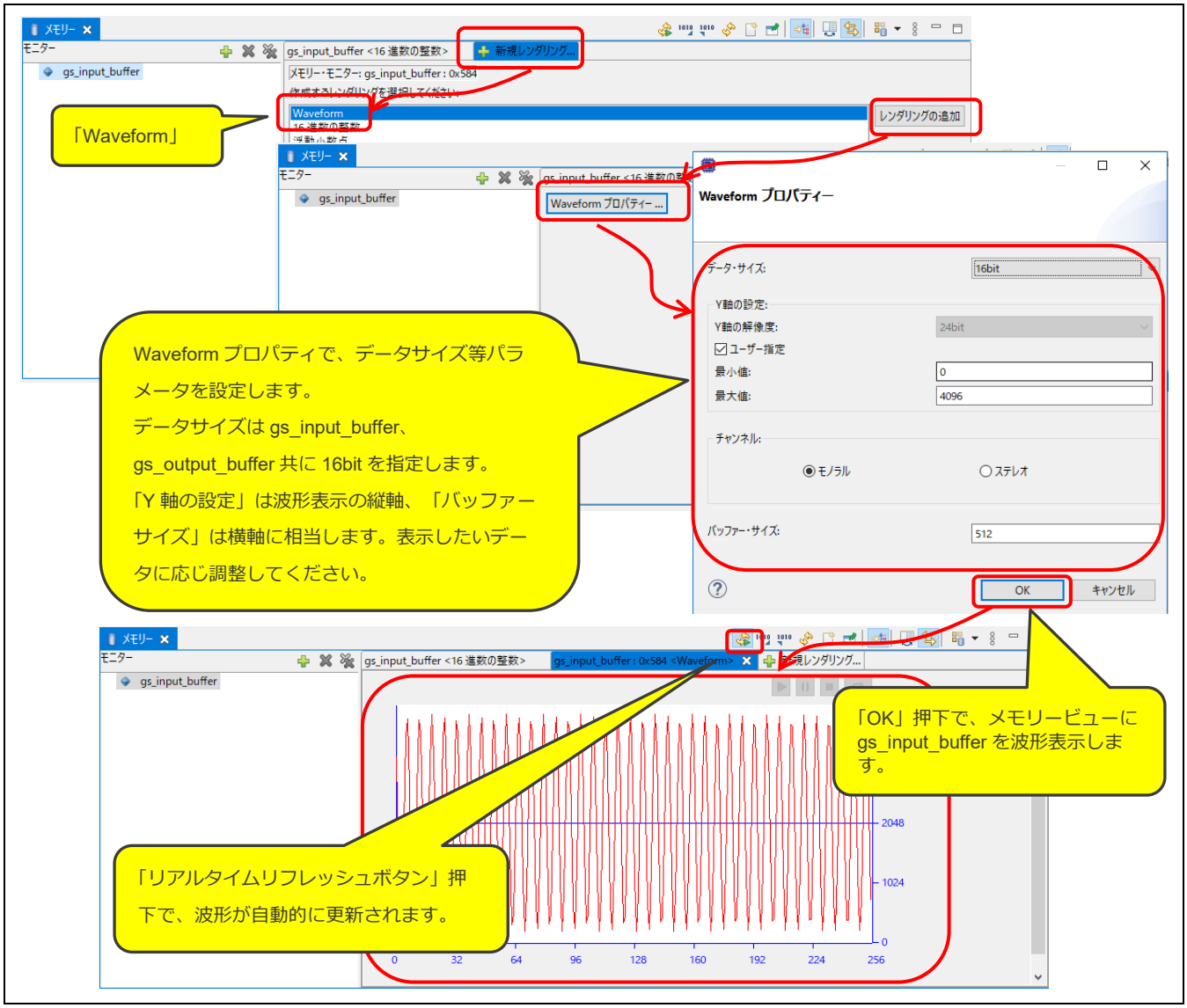


図 5.4 Waveform レンダリングの表示手順

5.2　使用メモリ

サンプルプログラムのメモリ使用量を表 5.1 に示します。

ROM と RAM は表 5.1 の条件で生成した.map ファイルを元に算出、ユーザスタックと割り込みスタックはサンプルプログラムを実行し実際に使用されたスタックメモリを測定しています。

表 5.1 サンプルプログラムの使用メモリ（参考）

項目	測定値 [byte]	説明
ROM	13801	サンプルプログラムの ROM 使用量。
RAM	12603	サンプルプログラムの RAM 使用量。
ユーザスタック	116	サンプルプログラムのスタック使用量。
割り込みスタック	128	

5.3　FIR フィルタ処理の消費リソース、マイコン選定のヒント

5.3.1　サイクル数、RAM 消費、CPU 占有率、RAM 占有率

サンプルプログラムの FIR フィルタ処理 1 チャンネルの消費リソースを、条件別に表 5.2～表 5.6 に例示します。

- CPU 占有率や RAM 占有率は、システムの実現可否を判断する指標となります。演算語長、サンプリング周波数 Taps 数など条件によって、必要となる実行サイクル数や RAM が RX140 では不足する場合があります。CPU 占有率や RAM 占有率に特に着目し、表 5.2～表 5.6 を実現可否判断やルネサスマイコン選定の参考にしてください。
- 表 5.2 はチャンネル数を除きサンプルプログラムの設定での消費リソースです。表 5.2 とその他の表を比較することをお勧めします。比較により条件ごと消費リソースの増減の傾向をご理解いただけます。
- 各表に示す測定値は、ソースファイル r_dsp_fir_i16i16.c または r_dsp_fir_i32i32.c, r_dsp_fir_f32f32.c に記載された関数および変数、定数などに加え、main.c で定義されたバッファメモリなど、FIR フィルタ処理の実行に必要な要素を含みます。
- 表に示す値は RX140 での測定値です。記載ないものはサンプルプログラムの設定とします。

  詳しい測定条件は後述の 5.3.2 消費リソース測定条件を参照してください。

表 5.2 FIR フィルタ処理の消費リソース（1 チャンネル分）

	Channels:1, Taps:64, Fs:10KHz, Unit Samples:256		
	r_dsp_fir_i16i16.c	r_dsp_fir_i32i32.c	r _dsp_fir_f32f32.c
Cycle count (CPU usage)	59497 cycles (4.84 %)	87713 cycles (7.14 %)	135317 cycles (11.01 %)
RAM (RAM usage)	3108 bytes (18.97 %)	6180 bytes (37.72 %)	6180 bytes (37.72 %)
Stack	SU： 108 bytes SI： 0 bytes	SU： 120 bytes SI： 0 bytes	SU： 120 bytes SI： 0 bytes
ROM (ROM usage)	823 bytes (1.26 %)	2626 bytes (4.01 %)	1142 bytes (1.74 %)

表 5.3 FIR フィルタ処理の消費リソース（3 チャンネル分）

	Channels:3, Taps:64, Fs:10KHz, Unit Samples:256		
	r_dsp_fir_i16i16.c	r_dsp_fir_i32i32.c*1	r _dsp_fir_f32f32.c*1
Cycle count (CPU usage)	180554 cycles (14.69 %)	-	-
RAM (RAM usage)	8300 bytes (50.66 %)	-	-
Stack	SU： 112 bytes SI： 0 bytes	-	-
ROM (ROM usage)	1207 bytes (1.84 %)	-	-

*1 RX140(RAM 16KB)では RAM が不足するため、記載を省略。

表 5.4 FIR フィルタ処理の消費リソース（Taps：256）

	Channels:1, Taps:256, Fs:10KHz, Unit Samples:256		
	r_dsp_fir_i16i16.c	r_dsp_fir_i32i32.c,	r _dsp_fir_f32f32.c
Cycle count (CPU usage)	213098 cycles (17.34 %)	308899 cycles (25.14 %)	503446 cycles (40.97 %)
RAM (RAM usage)	3108 bytes (18.97 %)	6180 bytes (37.72 %)	6180 bytes (37.72 %)
Stack	SU： 108 bytes SI： 0 bytes	SU： 120 bytes SI： 0 bytes	SU： 120 bytes SI： 0 bytes
ROM (ROM usage)	825 bytes (1.26 %)	2627 bytes (4.01 %)	1143 bytes (1.74 %)

表 5.5 FIR フィルタ処理の消費リソース（単位処理サンプル数：64）

	Channels:1, Taps:64, Fs:10KHz, Unit Samples:64		
	r_dsp_fir_i16i16.c	r_dsp_fir_i32i32.c,	r _dsp_fir_f32f32.c
Cycle count (CPU usage)	14946 cycles (4.87 %)	21987 cycles (7.16 %)	34003 cycles (11.07 %)
RAM (RAM usage)	804 bytes (4.91 %)	1572 bytes (9.59 %)	1572 bytes (9.59 %)
Stack	SU： 108 bytes SI： 0 bytes	SU： 120 bytes SI： 0 bytes	SU： 120 bytes SI： 0 bytes
ROM (ROM usage)	819 bytes (1.25 %)	2622 bytes (4.00 %)	1138 bytes (1.74 %)

表 5.6 FIR フィルタ処理の実行サイクル数（サンプリング周波数別）*1

	Channels:1, Taps:64, Unit Samples:256		
Fs [Hz]	r_dsp_fir_i16i16.c	r_dsp_fir_i32i32.c,	r _dsp_fir_f32f32.c
1000 (CPU usage)	59497 cycles (0.48 %)	87713 cycles (0.71 %)	135059 cycles (1.01 %)
10000 (CPU usage)	59497 cycles (4.84 %)	87713 cycles (7.14 %)	135317 cycles (11.01 %)
16000 (CPU usage)	59497 cycles (7.75 %)	95394 cycles (12.42 %)	135317 cycles (17.62 %)
24000 (CPU usage)	59497 cycles (11.62 %)	95394 cycles (18.63 %)	135317 cycles (26.43 %)

*1 RAM などのリソース消費は、表 5.1 を参照してください。

5.3.2　消費リソース測定条件

表 5.2～表 5.6 の消費リソースの測定条件を列挙する。

- 測定対象は、R_DSP_FIR_Operation の実行に関わる処理やメモリとした。（サンプルプログラム全体の消費リソースは表 5.1 サンプルプログラムの使用メモリ（参考）を参照）
- 条件ごとサンプルプログラムの r_dsp_fir_config.h の設定を変更して測定。

  主な測定条件
  - 統合開発環境：　ルネサス エレクトロニクス e^2 studio 2021-10
  - C コンパイラ：　ルネサス エレクトロニクス RX Compiler CC-RX V3.03.00
  - サンプルプログラム：Version 1.00

  その他の条件は表 1.2 **動作確認条件**を参照。
- Cycle count, CPU 占有率

  R_DSP_FIR_Operation の実行サイクル数を測定。CPU 占有率は、サンプリング周波数を 10kHz とし、256 サンプルのデータ入力にかかる実行サイクル数に対する R_DSP_FIR_Operation 実行サイクル数の比率。

  例）表 5.2 の r_dsp_fir_i16i16.c の CPU 占有率

  (a) R_DSP_FIR_Operation 処理実行サイクル数：　59497[cycle]
  (b)単位処理サンプル数格納にかかるサイクル数：　1228800[cycle]
  (c) CPU 占有率　：　4.84[%]= (a) / (b) x 100 [%]
- RAM, RAM 占有率

  RAM は、r_dsp_fir_i16i16.c, r_dsp_fir_i32i32.c, r_dsp_fir_f32f32.c と gs_intemediate_buffer と gs_output_buffer の RAM 消費量の総和。RAM 占有率は、FIR フィルタ処理の RAM 消費量が RX140 の RAM 全体に示す比率。

  例）表 5.2 の r_dsp_fir_i16i16.c の RAM 消費率（FIR フィルタ処理に必要な RAM）

  (a) FIR フィルタ処理に必要な RAM：　3108[byte]
  (b) RX140 の RAM 容量：　16384[byte]
  (c) RAM 占有率　：　18.97[%]= (a) / (b) x 100 [%]
- Stack

  R_DSP_FIR_Init と R_DSP_FIR_Operation が使用するスタックメモリの最大値。
- ROM , ROM 占有率

  r_dsp_fir_i16i16.c または r_dsp_fir_i32i32.c, r_dsp_fir_f32f32 および 1 つの係数ファイル、DSP ライブラリの ROM 消費量。ROM 占有率は、FIR フィルタ処理の ROM 消費量が RX140 の ROM 全体に示す比率。

  例）表 5.2 の r_dsp_fir_i16i16.c の ROM 占有率（FIR フィルタ処理に必要な ROM）

  (a) FIR フィルタ処理に必要な ROM：　823[byte]
  (b) RX140 の ROM 容量：　65536[byte]
  (c) ROM 占有率　：　1.26[%]= (a) / (b) x 100 [%]

5.4　CPU の負荷低減

S12AD、CMT、ELC、DTC は、一旦設定されれば CPU の介在なしに動作します。この特長を利用し、ソフトウェアによる処理実行時のみ通常動作モードで CPU を動作させ、それ以外は WAIT 命令により CPU をスリープモードに遷移させることで CPU 負荷を低減することができます。

この機能を使用する場合は表 2.1 を参照し、main.h に定義しているマクロ「SLEEP_MODE」の設定値を変更してください。またスリープモードを使用している際は、Waveform レンダリングが正常に動作しない場合があります。Waveform レンダリングを使用する場合は、スリープモードを使用しないでください。

5.5　ソフトウェアモジュールの設定

サンプルプログラムで使用した FIT モジュールおよび e²studio のスマート・コンフィグレータの設定、DSP ライブラリの設定を表 5.7～表 5.12 に示します。スマート・コンフィグレータの設定における各表の項目、設定内容は設定画面の表記で記載しています。各ソフトウェアモジュールの詳細は、8.参考資料に記載のアプリケーションノートを参照してください。

表 5.7 BSP モジュールの設定

分類	項目	設定、説明
スマート・コンフィグレータ >> コンポーネント >> r_bsp		プロパティは下記の変更以外はデフォルトの設定とする。
	Parameter checking	Disabled
	Heap size	0x900
スマート・コンフィグレータ >> クロック		「クロック」タブを下記の設定とし r_bsp_config.h に反映させる。
	VCC 設定	3.3(V)
	(Target Board 用サンプルプロジェクトの場合) メインクロック設定	動作：チェックを外す
	サブクロック発振器設定	動作：チェックを外す
	(Target Board 用サンプルプロジェクトの場合) HOCO クロック設定	動作：チェックする 周波数：48 (MHz)
	LOCO クロック設定	動作：チェックを外す
	(Target Board 用サンプルプロジェクトの場合) システムクロック設定	クロックソース：HOCO システムクロック(ICLK)：x1　48 (MHz) 周辺モジュールクロック(PCLKB)：x1/2　24 (MHz) 周辺モジュールクロック(PCLKD)：x1　48 (MHz) FlashIF クロック(FCLK)：x1　48 (MHz)
	IWDT 専用クロック設定	動作：チェックを外す

表 5.8 DTC モジュールの設定

分類	項目	設定、説明
r_dtc_rx_config.h		下記の変更以外はデフォルトの設定とする。
	DTC_CFG_USE_DMAC_FIT_MODULE	DTC モジュールを使用時に DMAC モジュールを使用する設定となっている。RX140 に DMAC は搭載されていないため、DTC_DISABLE に変更。

表 5.9 スマート・コンフィグレータの設定（S12AD）

分類	項目	設定
スマート・コンフィグレータ>> コンポーネント >> シングルスキャンモード S12AD（Config_S12AD0）		下記の設定とし、コードを生成する。
	アナログ入力モード設定	ダブルトリガモード：チェックを外す
	アナログ入力チャネル設定	AN000 のみチェックする
	変換開始トリガ設定	開始トリガソース：ELC からのトリガ
	割り込み設定	AD 変換終了割り込みを許可（S12ADI0）にチェックする。 優先順位：レベル 0（割り込み禁止）
	AD 変換値を加算/平均	AN000 のチェックを外す
	A/D 変換動作選択ビット	高速変換動作
	高電位側基準電圧選択ビット	AVCC0
	低電位側基準電圧選択ビット	ACSS0
	自己診断設定	モード：未使用
	断線検出アシスト設定	チャージ設定：未使用
	データレジスタ設定	データレジスタフォーマット：右詰にする 自動クリアイネーブル：自動クリアを禁止 加算/平均モード選択：加算モード 加算回数：1 回変換
	データ格納バッファ設定	禁止
	ウィンドウ機能設定	禁止
	ウィンドウ A/B 動作設定	比較ウィンドウ A 有効：チェックを外す 比較ウィンドウ B 有効：チェックを外す
	入力サンプリング時間設定	AN000：0.407 (μs)
	イベントリンクコントロールビット設定	ELC 用スキャン終了イベント：すべてのスキャン終了時にイベント発生

表 5.10 スマート・コンフィグレータの設定（CMT1）

分類	項目	設定、説明
スマート・コンフィグレータ>> コンポーネント >> コンペアマッチタイマ （Config_CMT1）		下記の設定とし、コードを生成する。
	クロック設定	PCLK/8
	コンペアマッチ設定	インターバル時間：100 μs (実際の値:100) レジスタ（CMCOR）：299 コンペアマッチ割り込みを許可（CMI1）：チェックを外す

表 5.11 スマート・コンフィグレータの設定（ICU）

分類	項目	設定、説明
スマート・コンフィグレータ>> コンポーネント >> 割り込みコントローラ（Config_ICU）		下記の設定とし、コードを生成する。
	IRQ0	IRQ0：チェックする 検出タイプ：立ち下がりエッジ デジタルフィルタ：PCLK/64 優先順位：レベル 13

表 5.12 スマート・コンフィグレータの設定（GPIO）

分類	項目	設定、説明
スマート・コンフィグレータ>> コンポーネント >> 入出力ポート（r_gpio_rx）		下記の設定とし、コードを生成する。
	Parameter Checking	System Default

6. 開発環境の入手

6.1 e² studio の入手方法

以下の URL にアクセスし、e² studio をダウンロードしてください。

https://www.renesas.com/products/software-tools/tools/ide/e2studio.html

なお、本ドキュメントは e² studio 2022-01 以降を使用することを前提としています。2022-01 よりも古い Ver.を使用した場合、e² studio の一部機能を使用できない可能性があります。ダウンロードする場合、ホームページに掲載されている最新 Ver.の e² studio を入手してください。

6.2 コンパイラパッケージの入手方法

以下の URL にアクセスして、RX ファミリ用 C/C++コンパイラパッケージをダウンロードしてください。

https://www.renesas.com/products/software-tools/tools/compiler-assembler/compiler-package-for-rx-family.html

7. 補足

7.1 無償評価版の「RX ファミリ用 C/C++コンパイラパッケージ」を利用する場合の注意事項

無償評価版の「RX ファミリ用 C/C++コンパイラパッケージ」には、使用期限と使用制限があります。使用期限が過ぎた場合、リンクサイズが 128K バイト以内に制限されるためロードモジュールが正しく生成されなくなる場合があります。

詳しくは、ルネサスのホームページにある、無償版ソフトウェアツールのページを参照してください。

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

7.2 RX ファミリ用 DSP ライブラリについて

本サンプルプログラムの DSP 処理（FFT など）には DSP ライブラリを応用しています。
DSP ライブラリの詳しい情報やダウンロードについては、ルネサスのホームページにある、RX ファミリ用 DSP ライブラリのページを参照してください。
https://www.renesas.com/software-tool/dsp-library-rx-family

8. 参考資料

- RX ファミリ ボードサポートパッケージモジュール Firmware Integration Technology (R01AN1685)
- e² studio コード生成ツール ユーザーズマニュアル RX API リファレンス編 (R20UT2864)
- Renesas e² studio スマート・コンフィグレータ ユーザーガイド(R20AN0451)
- RX140 グループ ユーザーズマニュアル ハードウェア編 (R01UH0905)

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。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Nov. 19. 21	-	初版発行
1.01	Mar. 18. 22	本書全体	・誤字、脱字、紛らわしい文の修正。 ・フォントや体裁の修正
		6	・誤記修正：表 1.1 から CMT の FIT モジュールを削除
		7	・表 1.2 バージョン更新：統合開発環境、C コンパイラ、サンプルプログラム ・誤記修正：iodefine.h のバージョンを 1.00A に修正
		9	・2.2 機器の接続 図を更新：図 2.2
		10	・2.3 サンプルプログラムの実行と動作確認 e² studio の操作の操作の説明を改善
		10, 11	・2.3.2 e² studio の機能を使った FIR フィルタ動作のモニタ 表題を修正 説明を簡素化 名称変更：パースペクティブ「fir_demo」を「FIR_Filter」に 図を更新：図 2.3、図 2.4、図 2.5
		18	・3.4.1 初期化 説明の修正：2. FIR フィルタの初期化
		20	・3.4.5 FIR フィルタ処理結果による周波数帯域判定 説明の簡素化 図の更新：図 3.7、脱字修正と文字の拡大
		27	・表 5.1 サンプルプログラムの使用メモリ サンプルプログラムの改訂に伴い更新
		28	・表 5.3 誤記修正：表の注記
		30	・5.3.2 消費リソース測定条件 測定条件について補足
		サンプルプログラム	・ FIR フィルタ出力信号の不連続を修正 r_dsp_fir_i16i16.c, r_dsp_fir_i32i32.c, r_dsp_fir_f32f32.c ・ バージョン更新：e² studio, C コンパイラ、FIT モジュールなど ・ e² studio の FIR_Filter パースペクティブ 名称変更：「FIR_Filter」に変更 修正：「モニター・メモリー」の表示順、式（変数名）など

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。