Renesas Synergy™
LCD-KIT サンプルプログラム解説 (AP-S5D9-0A)

3.2 版　2023 年 10 月 02 日

1.　概要

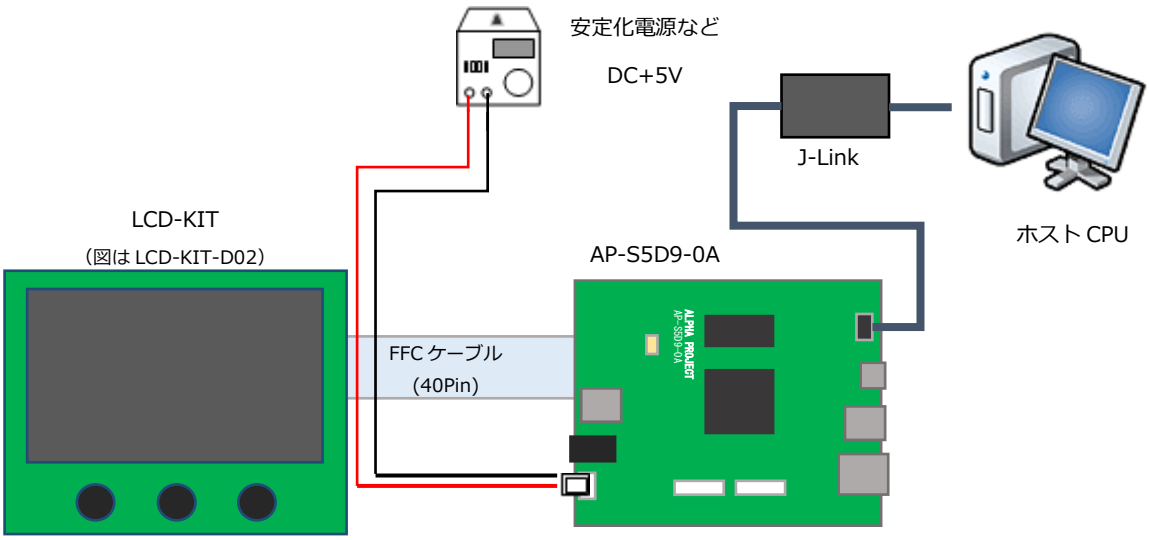
1.1　概要

本アプリケーションノートでは、AP-S5D9-0A(S5D9 CPU)を用いて、Renesas Synergy™の Synergy Software Package を使用した LCD-KIT サンプルプログラムについて解説します。
ソフトウェアは、静電容量式 LCD キット（LCD-KIT-B01、LCD-KIT-B02、LCD-KIT-D02）および抵抗膜式 LCD キット（LCD-KIT-C01、LCD-KIT-C02）に対応しています。
本サンプルプログラムで使用する主な機能を以下に記します。

デバイス	機能	動作内容
AP-S5D9-0A	・グラフィック LCD コントローラ（GLCDC）	グラフィック表示
	・シリアルコミュニケーションインタフェース（SCI3）	LCD-KIT との通信（簡易 I2C 通信）
LCD-KIT ・LCD-KIT-B01 ・LCD-KIT-B02 ・LCD-KIT-C01 ・LCD-KIT-C02 ・LCD-KIT-D02	・LCD パネル	・各画面の表示
	・タッチパネル（静電容量式・抵抗膜式）	・タッチ検出
	・バックライト	・バックライトの点灯
	・スイッチ	・各種機能の切り替え
	・ブザー（LCD-KIT-B01/B02/C01/C02）	・ブザー音出力
	・スピーカ（LCD-KIT-D02）	・音声出力

1.2　接続概要

本サンプルプログラムの動作を確認する上で必要な CPU ボードの接続例を以下に示します。



※AP-S5D9-0A と J-Link を直接接続することはできません。
　AP-S5D9-0A 側(ハーフピッチコネクタ)と J-Link 側(フルピッチコネクタ)を接続するための変換アダプタが必要となります。
　変換アダプタについては、J-Link 取扱店へご確認ください。

1.3　本サンプルプログラムについて

本サンプルプログラムおよび本書含むアプリケーションノートは、弊社 Web サイトのボード紹介ページで公開されています。

株式会社アルファプロジェクト
AP-S5D9-0A 製品ページ　https://www.apnet.co.jp/product/synergy/ap-s5d9-0a.html

1.4　開発環境について

本サンプルプログラムは統合開発環境「e2 studio」と「Synergy Software Package（以下、SSP）」を用いて開発されています。
本サンプルプログラムに対応する開発環境、SSP、コンパイラ、デバッガのバージョンは次の通りです。

ソフトウェア	バージョン	備考
e2 studio	v2021-07	–
SSP	v2.1.0	–
GCC ARM Embedded	v7.2.1	–
AP-S5D9-0A 用 Custom BSP	v2.1.0 - ap010000	–

デバッガ	ファームバージョン	備考
J-Link	V10	Segger Microcontroller Systems 社

1.5　ワークスペースについて

本サンプルプログラムのプロジェクトファイルは次のフォルダに格納されています。

サンプルプログラム	フォルダ
LCD-KIT-B01/B02/C01/C02 サンプルプログラム プロジェクトフォルダ	¥sample¥ap_s5d9_0a_sample_lcdkit_b01_c01
LCD-KIT-D02 サンプルプログラム プロジェクトフォルダ	¥sample¥ap_s5d9_0a_sample_lcdkit_d02

2.　サンプルプログラムの構成

2.1　フォルダ構成

サンプルプログラムは下記のようなフォルダ構成になっています。

sample	
├── Custom BSP	AP-S5D9-0A 用 Custom BSP フォルダ
├── ap_s5d9_0a_sample_lcdkit_b01_c01	LCD-KIT-B01/B02/C01/C02 サンプルプログラムフォルダ
│　├── .settings	設定ファイルフォルダ
│　├── data	LCD-KIT-D02 用 音声データ
│　├── script	リンカ設定ファイルフォルダ
│　└── src	ソースファイルフォルダ
│　　└── lcdkit	LCD-KIT デバイスドライバフォルダ
└── ap_s5d9_0a_sample_lcdkit_d02	LCD-KIT-D02 サンプルプログラムフォルダ
├── .settings	設定ファイルフォルダ
├── data	LCD-KIT-D02 用 音声データ
├── script	リンカ設定ファイルフォルダ
└── src	ソースファイルフォルダ
└── lcdkit	LCD-KIT デバイスドライバフォルダ

2.2　ファイル構成

サンプルプログラムは以下のファイルで構成されています。
本節では、サンプルプログラムの作成にあたって追加したファイルについてのみ記述し、自動生成ファイルなどに関しては説明を省略します。

＜¥sample¥Custom BSP フォルダ内＞

AlphaProject.ap_s5d9_0a.2.1.0-ap010000.pack	･･･	AP-S5D9-0A 用 Custom BSP

＜¥sample¥ap_s5d9_0a_sample_lcdkit_b01_c01 フォルダ内＞

.cproject	･･･	CPROJECT ファイル
.project	･･･	PROJECT ファイル
configuration.xml	･･･	Synergy コンフィギュレータファイル
ap_s5d9_0a(QSPI_ON)_R7FS5D97E3A01CFC.pincfg	･･･	AP-S5D9-0A 用 QSPI 対応ピンコンフィグファイル
ap_s5d9_0a(SDHI_ON)_R7FS5D97E3A01CFC.pincfg	･･･	AP-S5D9-0A 用 SDHI 対応ピンコンフィグファイル
R7FS5D97E3A01CFC.pincfg	･･･	S5D9 CPU 用　デフォルト ピンコンフィグファイル ※ AP-S5D9-0A 用の設定はしてありません。
ap_s5d9_0a_sample_lcdkit_b01_c01 Debug.jlink	･･･	AP-S5D9-0A　LCD-KIT-B01/B02/C01/C02 サンプルプログラム デバッグおよびランタイム設定ファイル
ap_s5d9_0a_sample_lcdkit_b01_c01 Debug.launch	･･･	AP-S5D9-0A　LCD-KIT-B01/B02/C01/C02 サンプルプログラム デバッグおよびランタイム設定ファイル

＜¥sample¥ap_s5d9_0a_sample_lcdkit_d02 フォルダ内＞

.cproject	･･･	CPROJECT ファイル
.project	･･･	PROJECT ファイル
configuration.xml	･･･	Synergy コンフィギュレータファイル
ap_s5d9_0a(QSPI_ON)_R7FS5D97E3A01CFC.pincfg	･･･	AP-S5D9-0A 用 QSPI 対応ピンコンフィグファイル
ap_s5d9_0a(SDHI_ON)_R7FS5D97E3A01CFC.pincfg	･･･	AP-S5D9-0A 用 SDHI 対応ピンコンフィグファイル
R7FS5D97E3A01CFC.pincfg	･･･	S5D9 CPU 用　デフォルト ピンコンフィグファイル ※ AP-S5D9-0A 用の設定はしてありません。
ap_s5d9_0a_sample_lcdkit_d02 Debug.jlink	･･･	AP-S5D9-0A　LCD-KIT-D02 サンプルプログラム デバッグおよびランタイム設定ファイル
ap_s5d9_0a_sample_lcdkit_d02 Debug.launch	･･･	AP-S5D9-0A　LCD-KIT-D02 サンプルプログラム デバッグおよびランタイム設定ファイル

※以下、「LCD-KIT-B01/C01/C02 サンプルプログラム」のフォルダおよびファイルに関して記しますが、
特に記載のない場合、「LCD-KIT-D02 サンプルプログラム」のファイルはフォルダ名以外共通となります。

<¥sample¥ap_s5d9_0a_sample_lcdkit_b01_c01¥data フォルダ内>

Sound0.bin	・・・	音声データ 0（LCD-KIT-D02 用）
Sound1.bin	・・・	音声データ 1（LCD-KIT-D02 用）
Sound2.bin	・・・	音声データ 2（LCD-KIT-D02 用）

<¥sample¥ap_s5d9_0a_sample_lcdkit_b01_c01¥script フォルダ内>

r7fs5d97e3a01cfc.ld	・・・	リンカスクリプトファイル

<¥sample¥ap_s5d9_0a_sample_lcdkit_b01_c01¥src フォルダ内>

hal_entry.c	・・・	hal_entry 関数ソースファイル
ioport_app.c	・・・	I/O ポート制御ソースファイル
ioport_app.h	・・・	I/O ポート制御ヘッダファイル
lcd_thread_entry .c	・・・	lcd_thread_entry 関数ソースファイル
lcd_thread_main .c	・・・	LCD アプリケーションソースファイル
common_app.h	・・・	共通ヘッダファイル

<¥sample¥ap_s5d9_0a_sample_lcdkit_b01_c01¥src¥lcdkit フォルダ内>

lcdkit.h	・・・	LCD-KIT デバイス情報ヘッダファイル ※このファイル内で使用する LCD-KIT を定義します。 「LCD-KIT-B01/B02/C01 サンプルプログラム」と 「LCD-KIT-D02 サンプルプログラム」では 使用する LCD-KIT のデフォルトの定義が異なります。
lcdkit_app.c	・・・	LCD-KIT デバイスドライバソースファイル
lcdkit_app.h	・・・	LCD-KIT デバイスドライバヘッダファイル
lcdkit_b01.c	・・・	LCD-KIT-B01 デバイスドライバソースファイル
lcdkit_b01.h	・・・	LCD-KIT-B01 デバイスドライバヘッダファイル
lcdkit_b02.c	・・・	LCD-KIT-B02 デバイスドライバソースファイル
lcdkit_b02.h	・・・	LCD-KIT-B02 デバイスドライバヘッダファイル
lcdkit_c01.c	・・・	LCD-KIT-C01 デバイスドライバソースファイル
lcdkit_c01.h	・・・	LCD-KIT-C01 デバイスドライバヘッダファイル
lcdkit_d02.c	・・・	LCD-KIT-D02 デバイスドライバソースファイル
lcdkit_d02.h	・・・	LCD-KIT-D02 デバイスドライバヘッダファイル

3.　サンプルプログラムについて

3.1　動作説明

3.1.1　サンプルプログラム動作説明

サンプルプログラムは、下記の動作を行います。

- グラフィック表示

タッチパネル入力を検出すると、以下の順でグラフィックの切り替えを行います。
・カラーバー
・グラデーション
・市松模様
・カラーバー（以下、繰り返し）

- ブザー（LCD-KIT-B01/B02/C01/C02）

SW 入力に応じて、ブザー音を出力します。
SW1　・・・　ブザー音①を出力します（ブザー音①：低）
SW2　・・・　ブザー音②を出力します（ブザー音②：中）
SW3　・・・　ブザー音③を出力します（ブザー音③：高）

- 音声出力（LCD-KIT-D02）

SW 入力に応じて、音声を出力します。
SW1　・・・　「ピンポーン」
SW2　・・・　「ブブー」
SW3　・・・　「ありがとうございました」

- LED

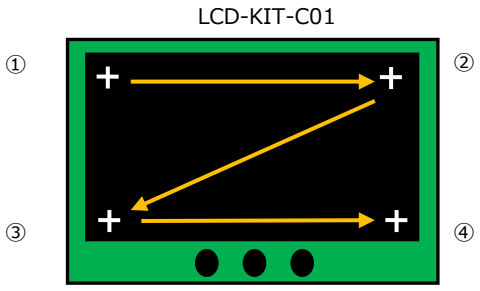
LED は LCD-KIT の制御に合わせて点灯・消灯します。
（電源投入　・・・　LD1：消灯　LD2：消灯）
LCD-KIT 初期化完了　・・・　LD1：点灯　LD2：消灯
- LCD-KIT 初期化失敗　・・・　LD1：点滅　LD2：点滅
カラーバー表示開始　・・・　LD1：点灯　LD2：点灯

3.1.2　LCD-KIT-C01/C02 のキャリブレーション

LCD-KIT-C01/C02 は、抵抗膜式タッチパネル付き LCD ボードです。
静電容量式の LCD-KIT とは異なり、キャリブレーション処理を行う必要があります。

LCD-KIT-C01/C02 使用時に本サンプルプログラムの動作を開始すると、ピピッと短いブザー音がした後、
黒い画面の左上に白色の十字のポインタが表示されます。
ユーザはそのポインタに対してタッチ＆リリースをしてください。
その後も順に右上、左下、右下の順にポインタが表示されますので、同様にユーザは各ポインタに対して
タッチ＆リリースをしてください。

LCD-KIT-C01



画面に対して四回タッチ＆リリースをするとキャリブレーション処理は終了です。
その後の動作は静電容量式の LCD-KIT と同様です。

・キャリブレーション処理の補正は、リリースする直前のタッチ情報を基に計算されます。
・LCD-KIT-C01 はマルチタッチに対応していません。同時に複数箇所に触れると誤動作する場合があります。

3.1.3　LCD-KIT の選択・設定

本サンプルプログラムは、使用する LCD-KIT によって、下記ソースファイルの変更が必要な場合があります。
変更後は、e2 studio 上で [プロジェクトをクリーンにする]を行ってから [プロジェクトのビルド]を選択して、
必ずプロジェクトを再ビルドしてください。

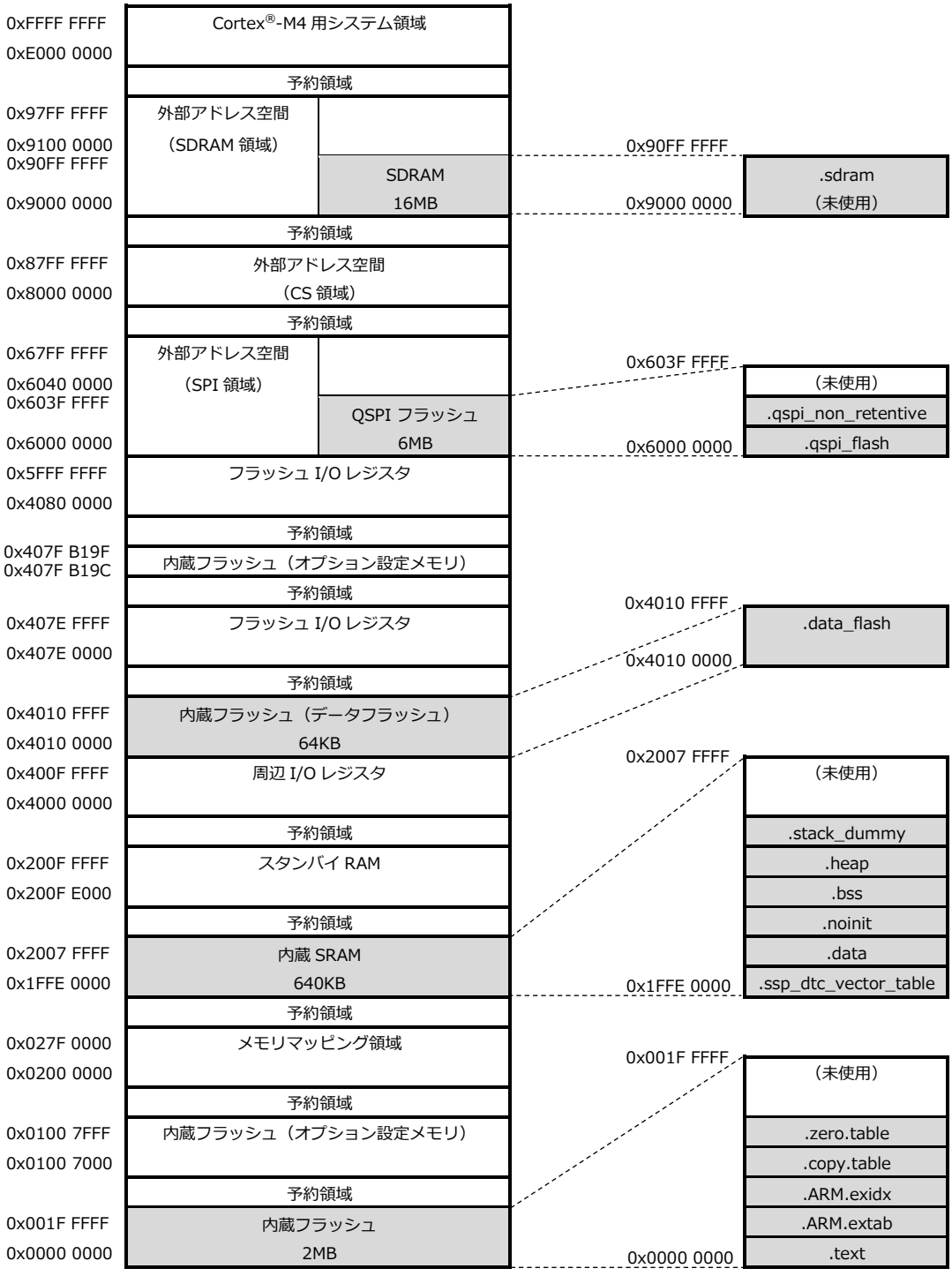
- LCD-KIT-B01/B02/C01/C02 を使用する場合
  プロジェクト「¥sample¥ap_s5d9_0a_sample_lcdkit_b01_c01」を使用します。
  デフォルトでは LCD-KIT-B01 を使用する設定になっています。
- LCD-KIT-D02 を使用する場合
  プロジェクト「¥sample¥ap_s5d9_0a_sample_lcdkit_d02」を使用します。
  デフォルトで LCD-KIT-D02 を使用する設定になっています。

<B01/B02/C01/C02>¥sample¥ap_s5d9_0a_sample_lcdkit_b01_c01¥src¥lcdkit¥lcdkit.h（18～28 行目）
<D02>　　¥sample¥ap_s5d9_0a_sample_lcdkit_d02¥src¥lcdkit¥lcdkit.h（18～28 行目）

```
18 :  #define LCDKIT_NONE    (0)
19 :  #define LCDKIT_B01     (LCDKIT_NONE + 1)          ←LCD-KIT-B01 と接続時の設定値
20 :  #define LCDKIT_B02     (LCDKIT_B01 + 1)           ←LCD-KIT-B02 と接続時の設定値
21 :  #define LCDKIT_C01     (LCDKIT_B02 + 1)           ←LCD-KIT-C01/C02 と接続時の設定値
22 :  #define LCDKIT_D02     (LCDKIT_C01 + 1)           ←LCD-KIT-D02 と接続時の設定値
                                  中略
28 :  #define LCDKIT  (LCDKIT_B01)     /* <- Please specify the LCD-KIT to use. */  ←赤字の値を適切に設定してください
```

3.2　メモリマップ

e2 studio のプロジェクトのメモリマップを以下に示します。



3.3　e2 studio を用いたプロジェクトのビルド・デバッグ

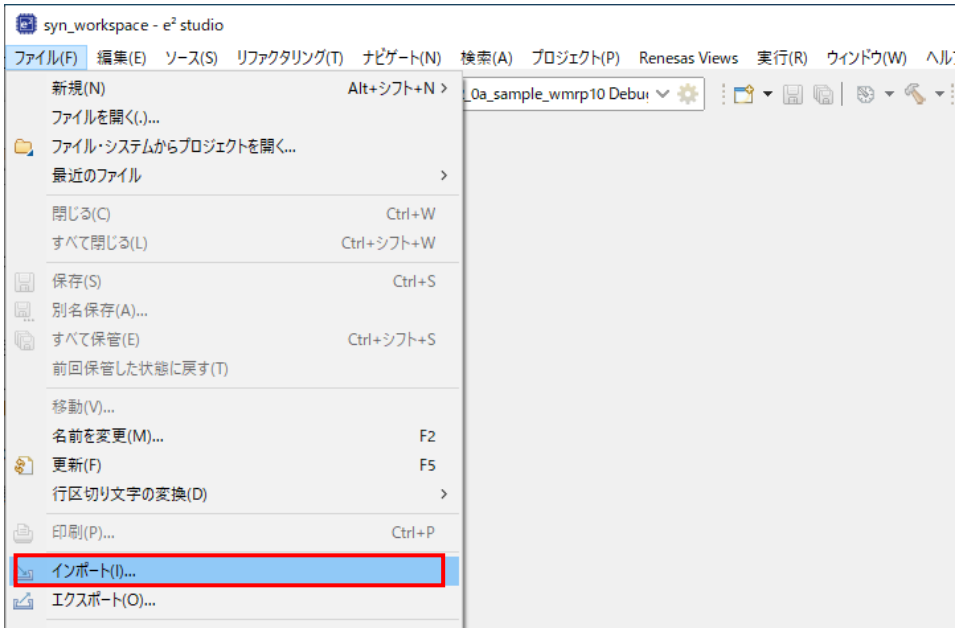
サンプルプログラムを CPU ボード上で実行するためには、e2 studio 上に一度サンプルプログラムをインポートし、ビルドを行う必要があります。

e2 studio 上へのサンプルプログラムのインポート方法、サンプルプログラムのビルド・デバッグ方法については本節で説明します。

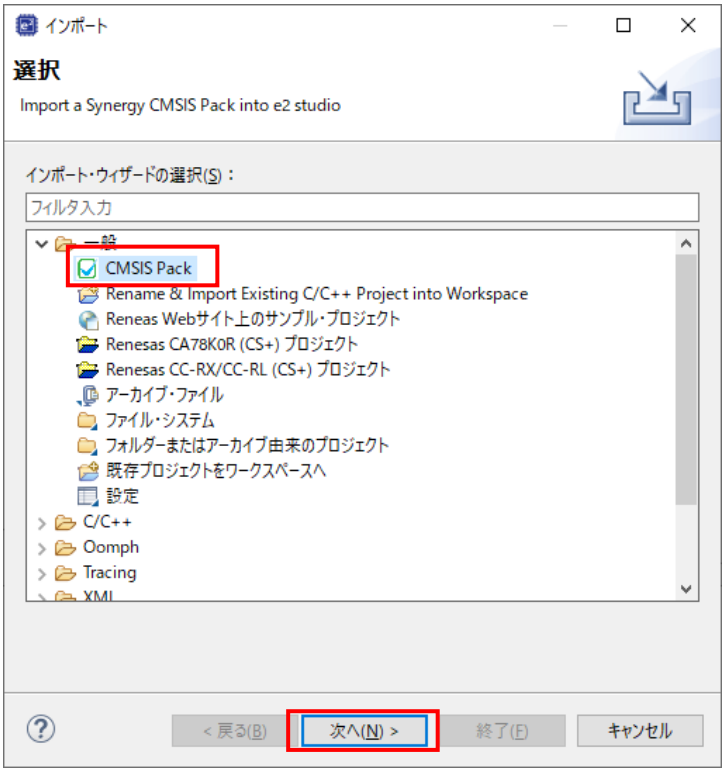
（下記で表示される図は「ap_s5d9_0a_sample_can」をデバッグ・ビルドする際の例として表示しています。プロジェクト名等は、ビルド・デバッグを行うサンプルプログラムにより変化します。）

3.3.1　インポート方法

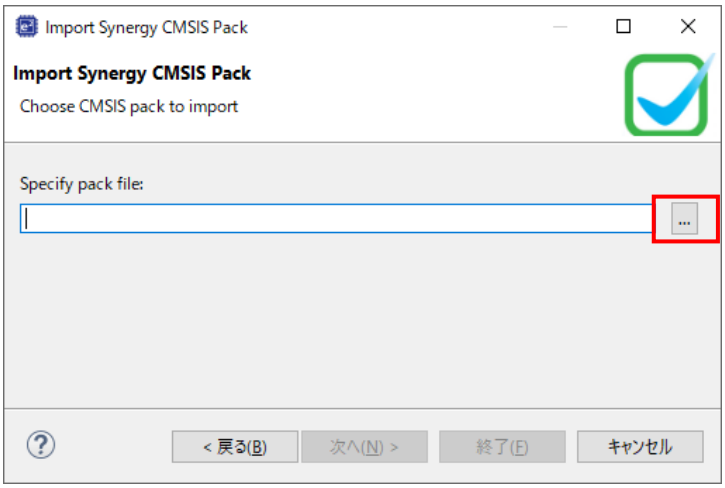
①　e2 studio を起動し、ツールバーの［ファイル］→［インポート］を選択します。



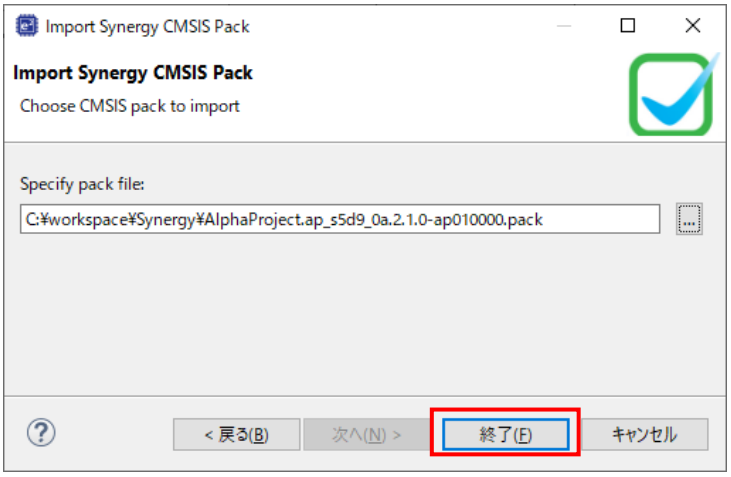
② ［CMSIS Pack］を選択し［次へ］を選択し、pack ファイル「AlphaProject.ap_s5d9_0a.2.1.0-ap010000.pack」をインポートします。
すでに開発環境に pack ファイルをインポート済みである場合は、⑤へお進みください。



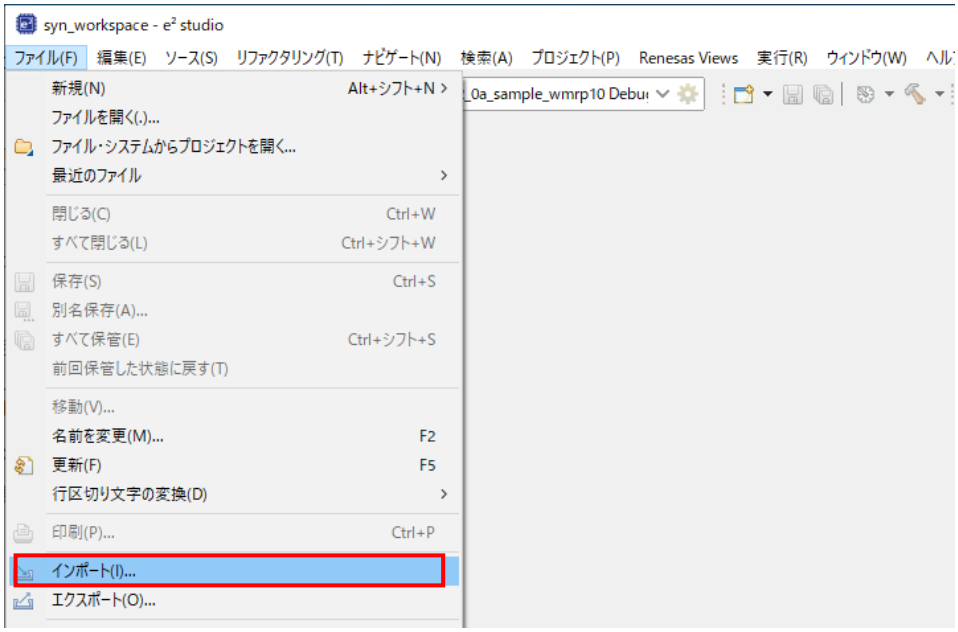
③ ［Import Synergy CMSIS Pack ウィンドウ］が表示されましたら、インポートする pack ファイル「sample¥ CustomBSP¥ AlphaProject.ap_s5d9_0a.2.1.0-ap010000.pack」を選択してください。



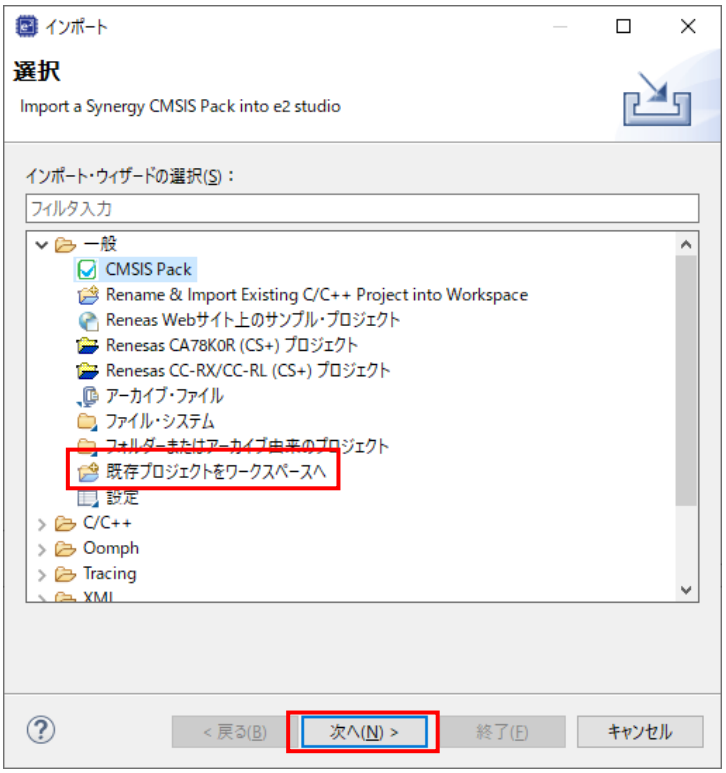
④　［終了］を選択してください。



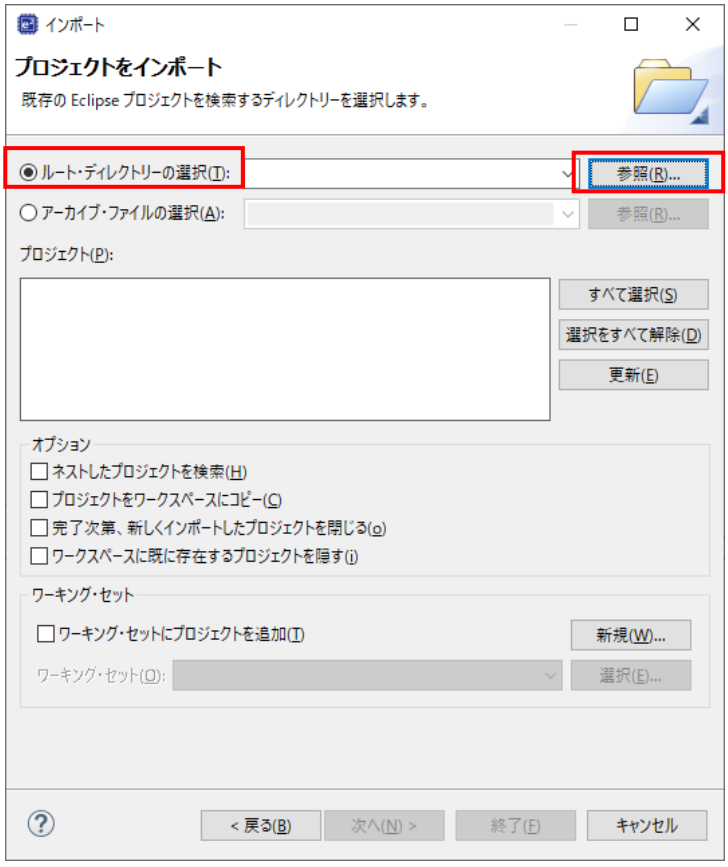
⑤　もう一度ツールバーの［ファイル］→［インポート］を選択します。



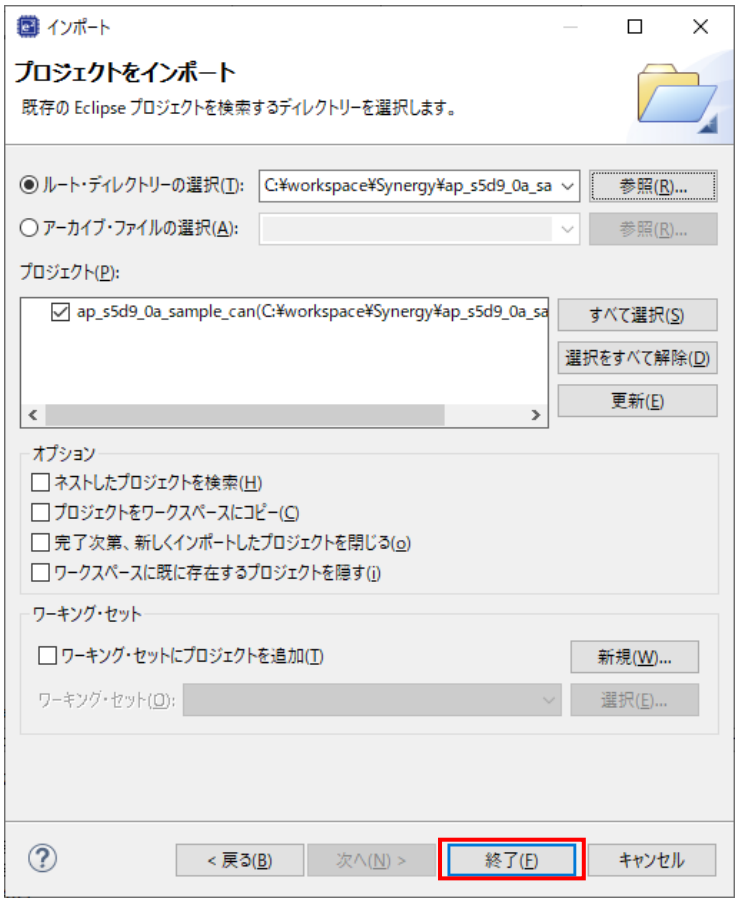
⑥　［既存のプロジェクトをワークスペースへ］を選択し［次へ］を選択します。



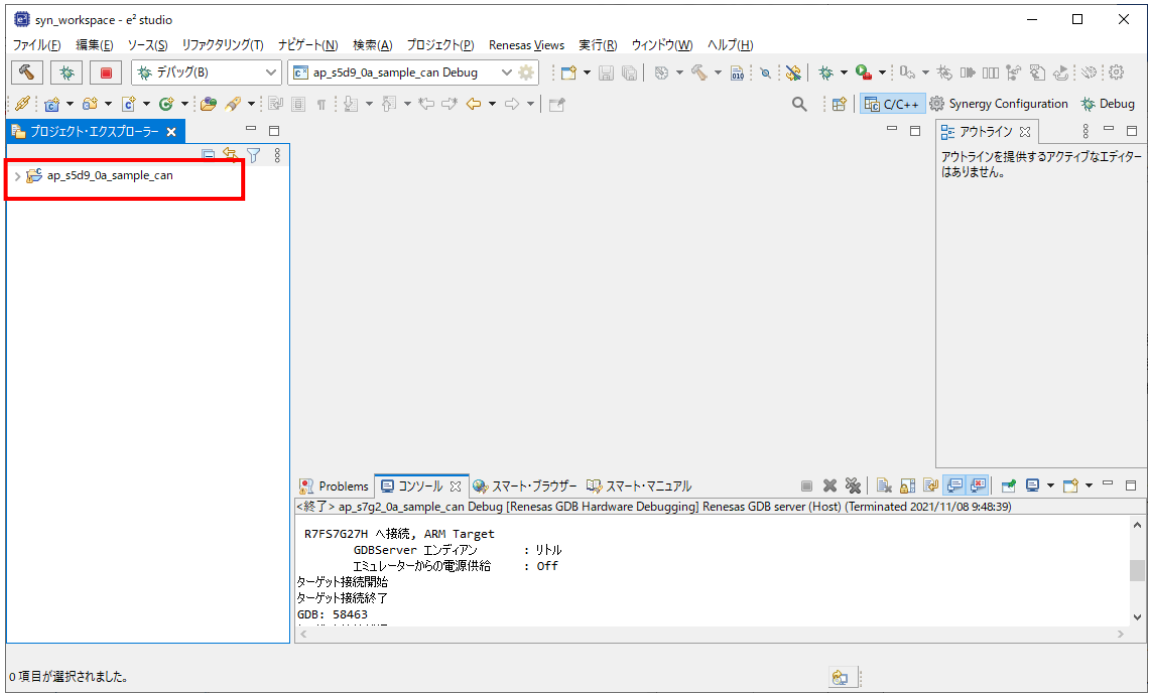
⑦　［ルート・ディレクトリーの選択］を選択し、［参照］からサンプルプログラムのフォルダを選択します。



⑧　[終了]を選択します。



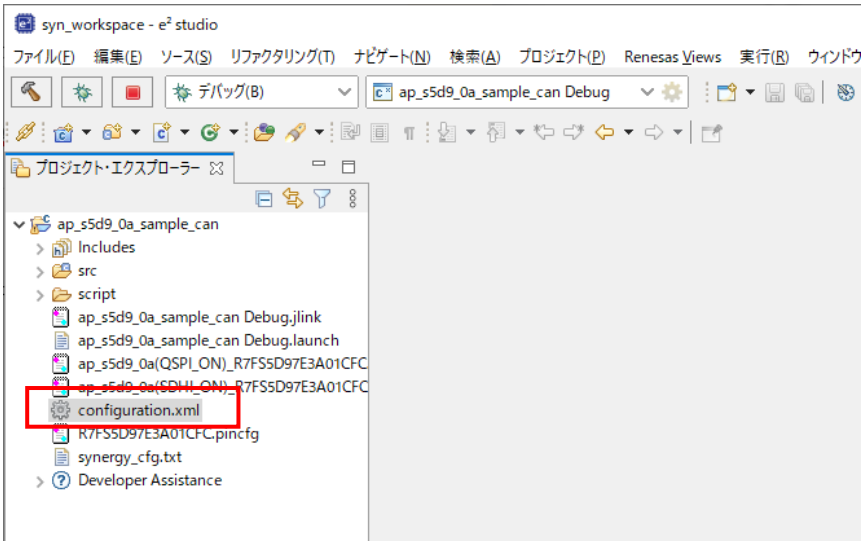
⑨　ナビゲーションウィンドウにサンプルプログラムのプロジェクトが追加されていることを確認します。



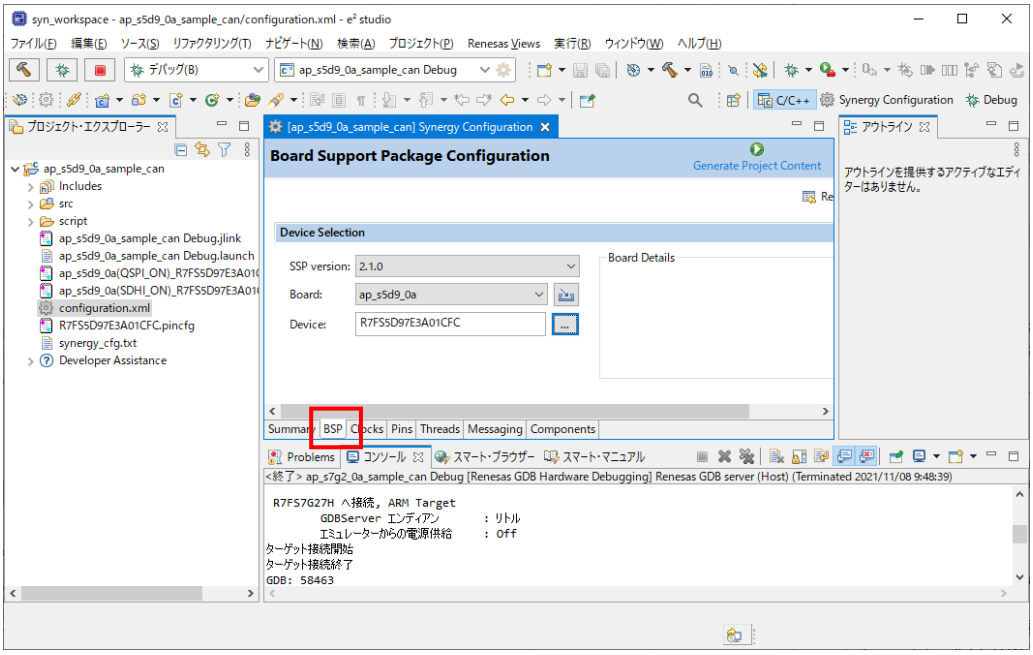
以上でプロジェクトのインポートは完了です。

3.3.2　ビルド方法

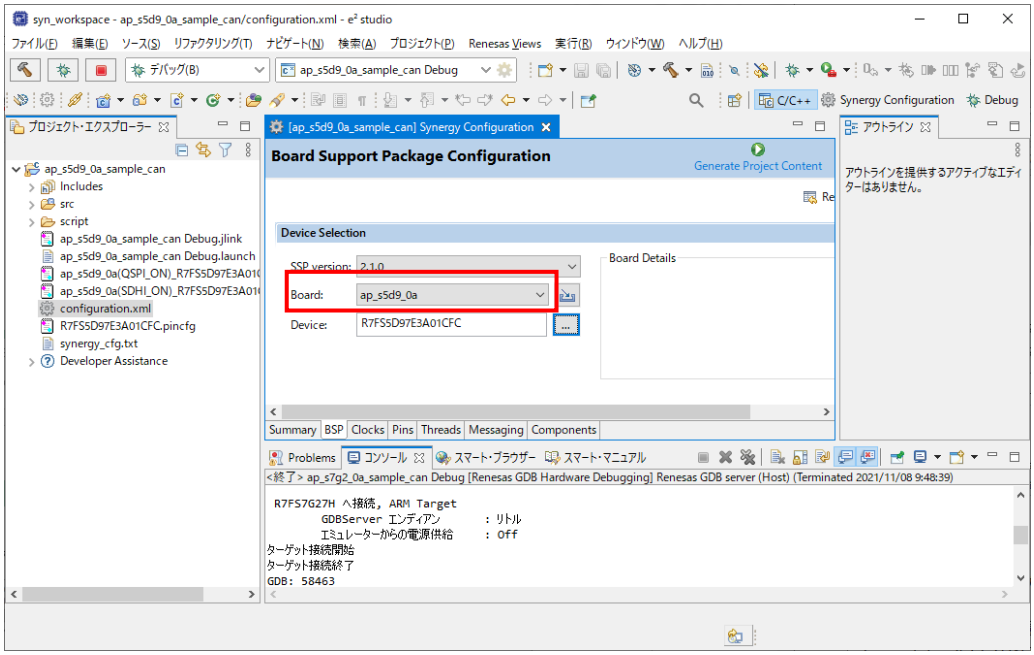
①　プロジェクトのコンフィギュレータファイルを開きます。



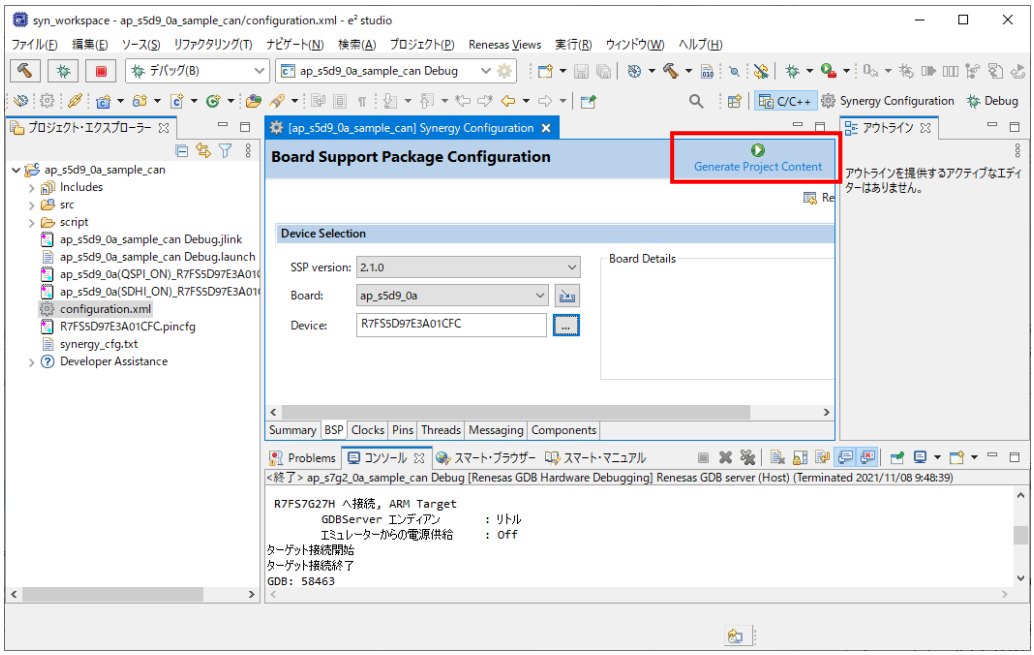
②　[BSP] タブを開きます。



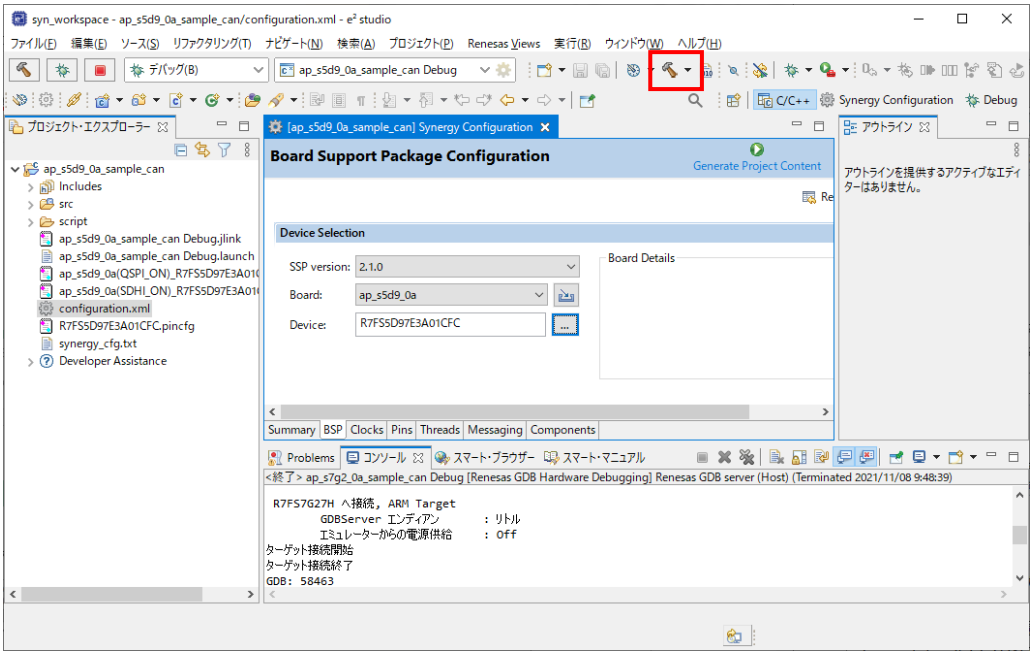
③　[BSP]タブで [Board] が「ap_s5d9_0a」であることを確認します。



④　[Generate Project Content] をクリックし、自動作成ファイルを出力して設定をプロジェクトに適用します。



⑤　ツールバーからビルドアイコンを選択します。
ビルドが成功すると、¥Debug ワークフォルダにオブジェクトファイルが生成されます。

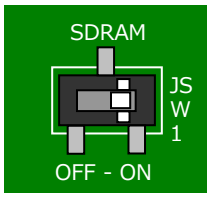


e2 studio の詳細な使用方法に関しては、 e2 studio のマニュアルを参照してください。

3.3.3　デバッグ方法

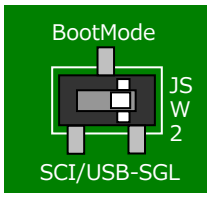
①　「3.3.2　ビルド方法」を参考に、プロジェクトをビルドしてください。

②　ボード上のディップスイッチを以下のように設定してください。



JSW1：ON

ボード上の SDRAM を使用する

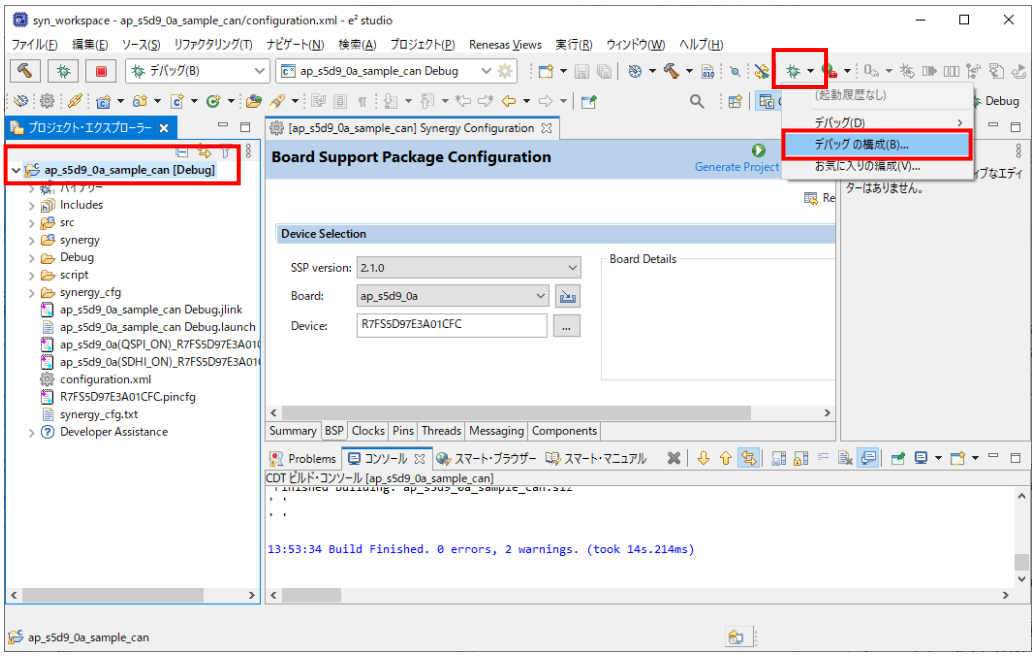


JSW2：SGL

シングルチップモード

③　ボードに電源を投入してください。

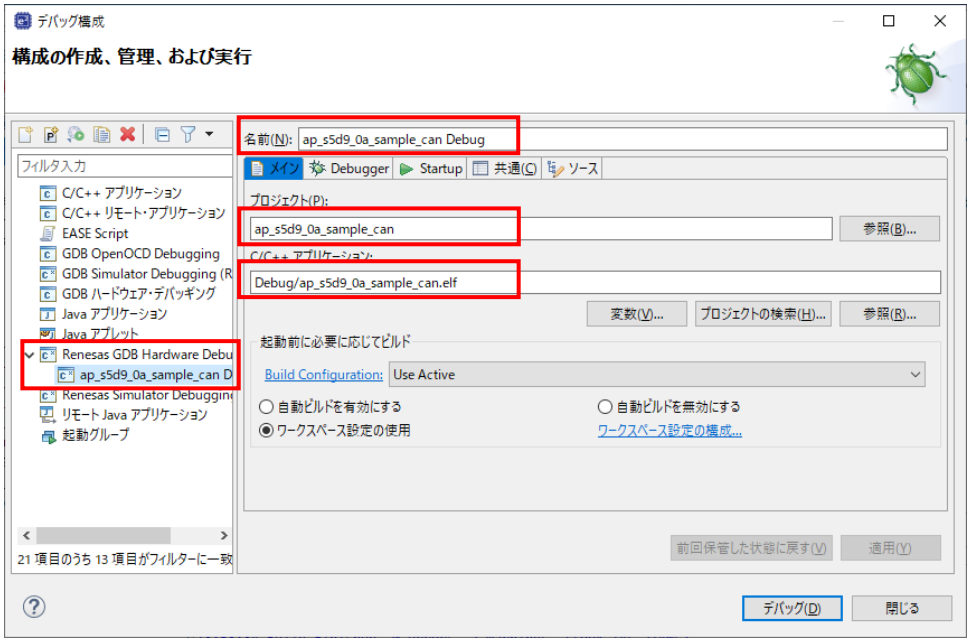
④　プロジェクトを選択し、メニューバーから [デバッグの構成] を開きます。



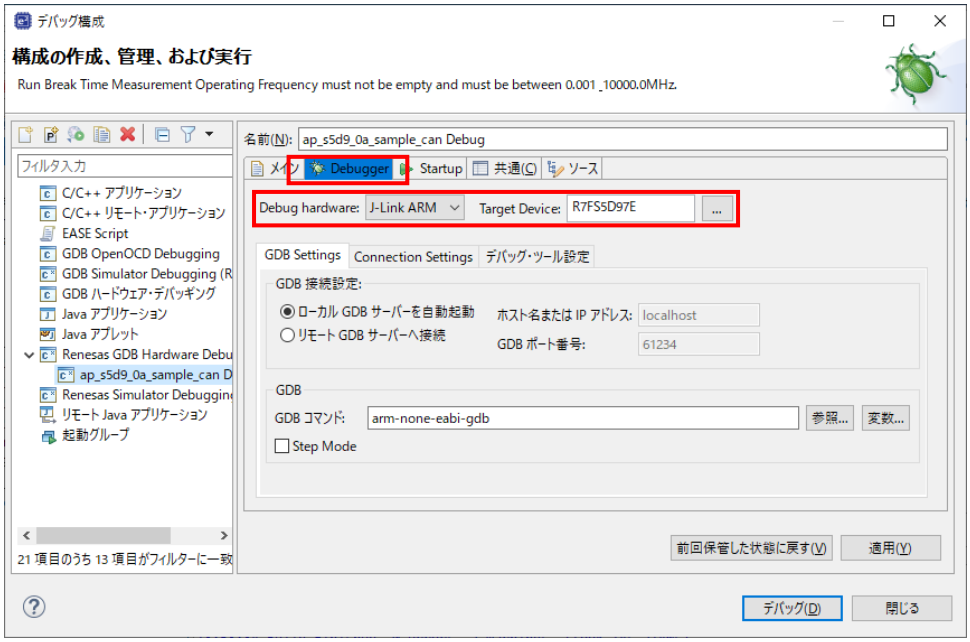
⑤　[Renesas GDB Hardware Debug] の [ap_s5d9_0a_XXXX Debug]を選択し、下記の内容になっていることを確認してください。

- [名前] : ap_s5d9_0a_XXXX Debug
- [プロジェクト] : ap_s5d9_0a_XXXX
- [C/C++アプリケーション] : Debug¥ ap_s5d9_0a_XXXX.elf

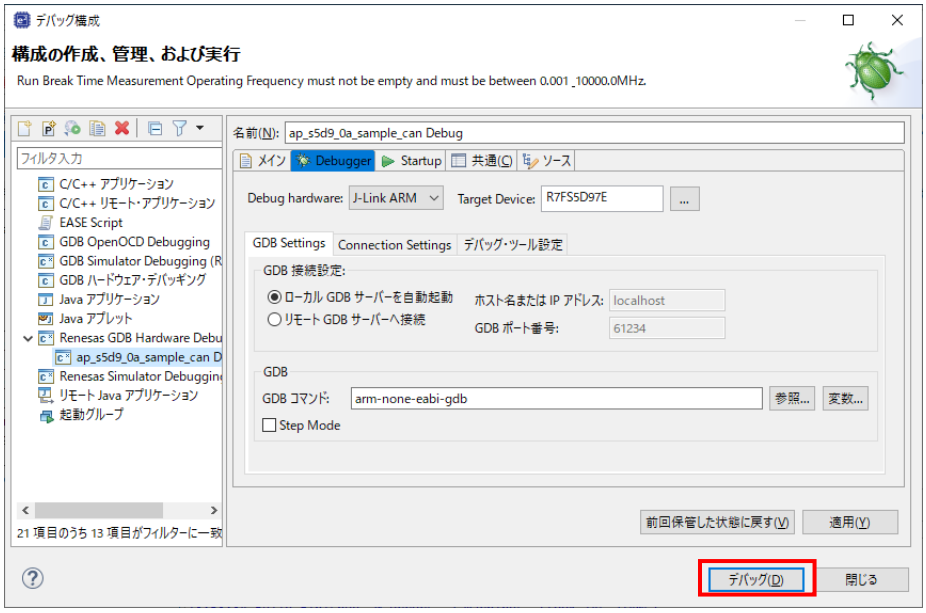
※.XXXX の個所は、デバッグ対象のサンプルプログラムにより名称が異なります。
「2.2 フォルダ構成」を参考に、デバッグ対象のサンプルプログラムに合わせたファイルを選択してください。



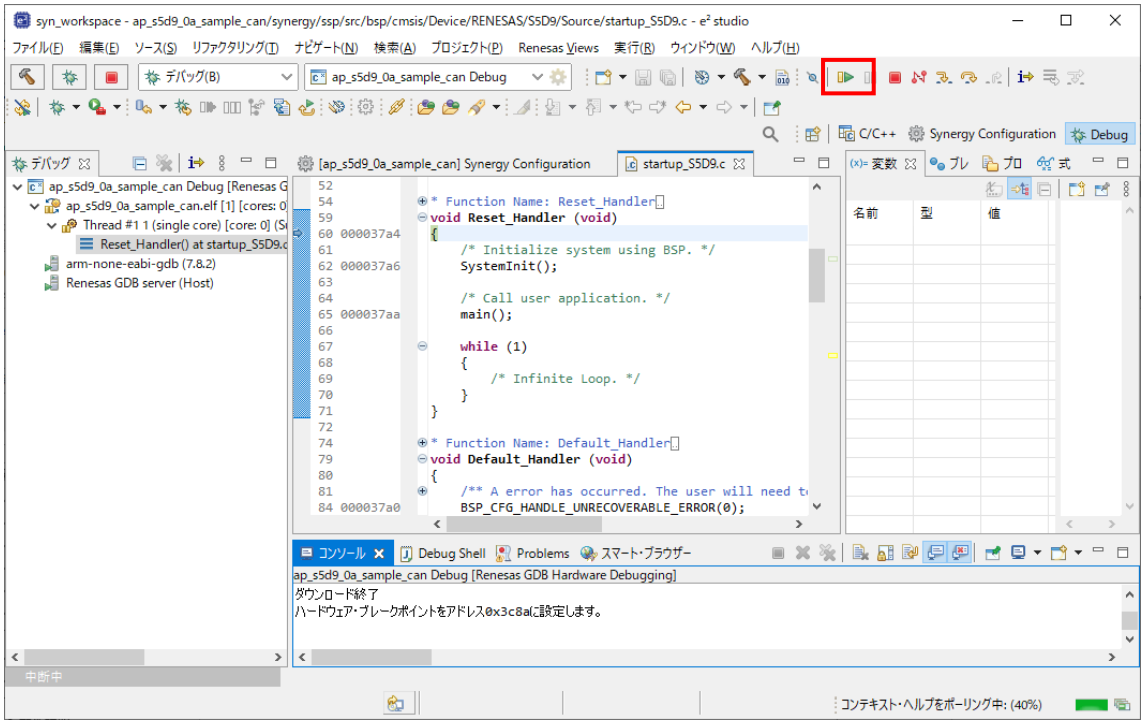
⑥　[Debugger] タブを選択し、 [Debug hardware] が [J-Link ARM] 、 [Target Device] が「R7FS5D97E」に設定されていることを確認してください。



⑦　[デバッグ] を選択します。



⑧　ボードとの接続が完了したらプログラムを実行し、サンプルプログラムを動作させてください。



⑨　プログラムの動作が確認できましたら、CPU ボードへのプログラムのダウンロードも完了しています。
以降、電源投入によりダウンロードされたプログラムの動作が開始されます。

ご注意

・本文書の著作権は株式会社アルファプロジェクトが保有します。
・本文書の内容を無断で転載することは一切禁止します。
・本文書に記載されているサンプルプログラムの著作権は株式会社アルファプロジェクトが保有します。
・本サンプルプログラムで使用されているミドルウェアおよびドライバの著作権はルネサス エレクトロニクス株式会社が保有します。
・本文書に記載されている内容およびサンプルプログラムについてのサポートは一切受け付けておりません。
・本文書の内容およびサンプルプログラムに基づき、アプリケーションを運用した結果、万一損害が発生しても、弊社では一切責任を負いませんのでご了承ください。
・本文書の内容については、万全を期して作成いたしましたが、万一ご不審な点、誤りなどお気付きの点がありましたら弊社までご連絡ください。
・本文書の内容は、将来予告なしに変更されることがあります。

商標について

・Renesas Synergy™および S5D9 は、ルネサス エレクトロニクス株式会社の登録商標、商標または商品名称です。
・e2 studio は、ルネサス エレクトロニクス株式会社の登録商標、商標または商品名称です。
・Synergy Software Package は、ルネサス エレクトロニクス株式会社の登録商標、商標または商品名称です。

・その他の会社名、製品名は、各社の登録商標または商標です。

ALPHA PROJECT
株式会社アルファプロジェクト
〒431-3114
静岡県浜松市中央区積志町 834
https://www.apnet.co.jp
E-Mail: query@apnet.co.jp