RL78 ファミリ
EEPROM エミュレーション・ライブラリ Pack02
パッケージ Ver.3.00 リリースノート

R20UT2645JJ0110
Rev.1.10
2024.03.15

この度は、EEPROM エミュレーション・ライブラリ Pack02 パッケージ Ver.3.00 をご使用いただきまして、誠にありがとうございます。

この添付資料では、EEPROM エミュレーション・ライブラリ Pack02 パッケージ Ver.3.00 をお使いいただく上での制限事項および注意事項等を記載しております。ご使用の前に、必ずお読みくださいますようお願い申し上げます。

目次

第 1 章　対象製品 2
第 2 章　ユーザーズマニュアル 2
第 3 章　バージョンアップ内容 2
第 4 章　対応ツール 3
第 5 章　インストール 3
5.1　インストール方法 3
5.2　アンインストール方法 3
5.3　ファイル構成 4
第 6 章　ビルド方法 5
6.1　使用するソフトウェア 5
6.2　CS+(旧 CubeSuite+)でのビルド方法 5
6.2.1　C 言語の場合 5
6.2.2　アセンブリ言語の場合 8
6.3　e^2 studio でのビルド 11
6.3.1　プロジェクトの作成 11
6.3.2　C 言語の場合 13
6.3.3　アセンブリ言語の場合(CC-RL コンパイラ使用時のみ) 18
6.4　ビルド時の注意事項 20
6.4.1　CA78K0R コンパイラ使用時 20
6.4.2　CC-RL コンパイラ使用時 21
第 7 章　デバッグ方法 22
7.1　デバッグ時の注意事項 22
第 8 章　サンプルプログラム 23
8.1　サンプルプログラムの初期設定 23
8.2　オプション・バイトとオンチップ・デバッグの設定について 24
8.3　内蔵 RAM 領域の定義 26
8.3.1　CA78K0R コンパイラ使用時 26
8.3.2　CC-RL コンパイラ使用時 27
8.3.3　LLVM コンパイラ使用時 33

第1章 対象製品

EEPROM エミュレーション・ライブラリ Pack02 パッケージ Ver.3.00 は、RL78 ファミリ CA78KR コンパイラ用、RL78 ファミリ CC-RL コンパイラ用と RL78 ファミリ LLVM コンパイラ(LLVM for Renesas RL78)用の EEPROM エミュレーション・ライブラリ Pack02 が含まれます。

本リリースノートの対象製品を示します。

製品名	Ver.	インストーラファイル名	Ver.
RL78 ファミリ CA78K0R コンパイラ用 EEPROM エミュレーション・ライブラリ Pack02	V1.01	RENESAS_RL78_EEL-FDL_T02_PACK02_3V00.exe	V3.00
RL78 ファミリ CC-RL コンパイラ用 EEPROM エミュレーション・ライブラリ Pack02	V1.01		
RL78 ファミリ LLVM コンパイラ用 EEPROM エミュレーション・ライブラリ Pack02	V1.01		

第2章 ユーザーズマニュアル

本バージョンは下記のユーザーズマニュアルに対応しています。

対象コンパイラ	マニュアル名	資料番号
CA78K0R, CC-RL, LLVM コンパイラ	RL78 ファミリ EEPROM エミュレーション・ライブラリ Pack02 日本リリース版 ユーザーズマニュアル注	R01US0068JJ0110

注. このユーザーズマニュアルは、弊社 HP からダウンロードしてください。

第3章 バージョンアップ内容

本バージョンのバージョンアップ内容を示します。

No.	パッケージ Ver.	対象	内容
1	V3.00	CA78K0R コンパイラ用 EEPROM エミュレーション・ライブラリ Pack02	パッケージ Ver.2.00 からライブラリ本体に変更はありません。
		CC-RL コンパイラ用 EEPROM エミュレーション・ライブラリ Pack02	パッケージ Ver.2.00 からライブラリ本体に変更はありません。
		LLVM コンパイラ用 EEPROM エミュレーション・ライブラリ Pack02	新規に追加されました。
		ユーザーズマニュアル	Rev.1.01 から Rev.1.10 に改版 改版内容につきましては、ユーザーズマニュアルの改版履歴をご参照ください(R01US0068JJ0110)。

第4章 対応ツール

EEPROM エミュレーション・ライブラリ Pack02 を使用する場合には、下記のバージョンを使用してください。

対象ライブラリ	ツール名	バージョン
CA78K0R コンパイラ用ライブラリ	統合開発環境：CubeSuite+	V1.00.00 以降
	統合開発環境：CS+	V3.00.00 以降
CC-RL コンパイラ用ライブラリ	統合開発環境：CS+	V3.01.00 以降
	統合開発環境：e^2 studio	Version: 2024-01 より掲載[注]
LLVM コンパイラ用ライブラリ	統合開発環境：e^2 studio	Version: 2024-01 以降

注. CC-RL コンパイラ V1.00 以降搭載バージョンで使用可能。

第5章 インストール

この章では、EEPROMエミュレーション・ライブラリ Pack02 パッケージ Ver.3.00のインストールとアンインストールの手順について説明します。

5.1 インストール方法

EEPROMエミュレーション・ライブラリ Pack02のインストールは次の手順で行います。

(1) Windowsを起動します。
(2) EEPROMエミュレーション・ライブラリ Pack02パッケージの圧縮ファイルを解凍し、インストーラを実行します。
(3) プルダウンメニューから"Asia/Oceania - Japanese"を選択します（図5-1）。
(4) "OK"ボタンを押下し、以降、インストーラの指示に従い、実行します。

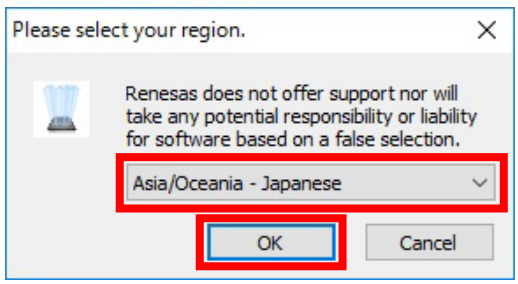

図 5-1 インストーラ"Asia/Oceania - Japanese"の選択

5.2 アンインストール方法

EEPROMエミュレーション・ライブラリ Pack02のアンインストールは次の手順で行います。

(1) Windowsを起動します。
(2) ユーザ任意の場所に配置したEEPROMエミュレーション・ライブラリが入っているフォルダを削除します。

5.3　ファイル構成

EEPROMエミュレーション・ライブラリ Pack02パッケージのインストーラを実行することによって展開されるファイルの構成は、次のとおりです。

```
配置先フォルダ
 ├ r20ut2645jjxxxx_rl78.pdf 注1          : リリースノート(本書)
 ├ support.txt                           : EEL のサポート情報
 └ CA78K0R_110, CCRL_100,またはLLVM_202312 : CA78K0R コンパイラ V1.10 以降、CC-RL コンパイラ V1.00 以降、
                                           LLVM for Renesas RL78 10.0.0.202312 以降が対象です。
    ├ lib
    │  ├ eel.lib or libeel.a             : EEPROM エミュレーション・ライブラリ(EEL)
    │  ├ fdl.lib or libfdl.a             : データ・フラッシュ・ライブラリ(FDL)
    │  ├ eel.h                           : C 言語用 EEL ヘッダファイル
    │  ├ eel.inc                         : アセンブラ用 EEL ヘッダファイル注5
    │  ├ eel_types.h                     : C 言語用 EEL 定義設定ヘッダファイル
    │  ├ eel_types.inc                   : アセンブラ用 EEL 定義設定ヘッダファイル注5
    │  ├ fdl.h                           : C 言語用 FDL ヘッダファイル
    │  ├ fdl.inc                         : アセンブラ用 FDL ヘッダファイル注5
    │  └ fdl_types.h                     : C 言語用 FDL 定義設定ヘッダファイル
    └ Sample
       ├ asm 注5
       │  ├ eel_descriptor.inc           : EEL ディスクリプタヘッダファイル
       │  ├ eel_descriptor.asm           : EEL ディスクリプタソースファイル
       │  ├ eel_sample_linker_file.dr    : EEL サンプルリンクディレクティブファイル(CA78K0R 版のみ)
       │  ├ fdl_descriptor.inc           : FDL ディスクリプタヘッダファイル
       │  └ fdl_descriptor.asm           : FDL ディスクリプタソースファイル
       └ C
          ├ eel_descriptor.h             : EEL ディスクリプタヘッダファイル
          ├ eel_descriptor.c             : EEL ディスクリプタソースファイル
          ├ eel_sample_linker_file.dr    : EEL サンプルリンクディレクティブファイル(CA78K0R 版のみ)
          ├ eel_user_types.h             : EEL ユーザ定義ヘッダファイル
          ├ fdl_descriptor.h             : FDL ディスクリプタヘッダファイル
          ├ fdl_descriptor.c             : FDL ディスクリプタソースファイル
          ├ r_eel_sample_c.c             : サンプルプログラムファイル注2,3,4
          └ r_eel_sample_c.dr            : サンプル用リンクディレクティブファイル(CA78K0R 版のみ) 注2
            or r_eel_sample_c.ld         : サンプル用リンカスクリプトファイル(LLVM 版のみ) 注4
```

注 1. xはバージョン番号、またはRev番号の為、省略しています。

2. CA78K0R用のサンプルプログラムを使用する場合は、プログラムファイル(*.c)とリンクディレクティブファイル(*.dr) [リンク情報の設定ファイル]を一緒に組み込んでください。

3. CC-RL用のサンプルプログラムを使用する場合は、プログラムファイル(*.c)を組み込んでください。リンク情報は、CS+、またはe^2 studioのリンクの設定画面で指定してください。

4. LLLVM用のサンプルプログラムを使用する場合は、プログラムファイル(*.c)とリンカスクリプトファイル(*.ld) [リンク情報の設定ファイル]を一緒に組み込んでください。

5. アセンブラ用のファイルは、CA78K0R、CC-RL用のフォルダにのみ含まれます。

第6章　　ビルド方法

この章では、EEPROMエミュレーション・ライブラリを用いたプログラムのビルドの手順を説明します。

6.1　使用するソフトウェア

EEPROM エミュレーション・ライブラリを用いたプログラムをビルドする際には、次の統合開発環境が必要です。

・CA78K0R コンパイラ用：統合開発環境 CS+ V3.00.00 以降、または CubeSuite+ V1.00.00 以降
・CC-RL コンパイラ用：統合開発環境 CS+ V3.01.00 以降、または e^2 studio Version: 2024-01 より掲載[注]
　注．CC-RL コンパイラ V1.00 以降搭載バージョンで使用可能
・LLVM コンパイラ用：統合開発環境 e^2 studio Version: 2024-01 以降

6.2　CS+（旧 CubeSuite+）でのビルド方法

CS+を用いて EEPROM エミュレーション・ライブラリ Pack02 をユーザプログラムに組み込んで、ビルドする手順を説明します。CS+の対象コンパイラは CA78K0R コンパイラと CC-RL コンパイラです。

6.2.1　　C言語の場合

(1) プロジェクトの作成とソースファイルの設定

CS+でプロジェクトを作成し、表示された画面の左側にある[プロジェクト・ツリー]から、[ファイル]を右クリックして表示されるリストの[追加]を選択し、[既存のファイルを追加]をクリックすると、[既存のファイルを追加]画面が表示されます（図 6-1）。

次に画面中にある[ファイルの種類]のプルダウンメニューをクリックすると、ファイルの種類の一覧が表示されるので、その中にある[C ソースファイル(*.c)]を選択し、ソースファイルとしてユーザプログラムのファイル（サンプルの場合は r_eel_sample_c.c)、及び EEPROM エミュレーション・ライブラリ、データ・フラッシュ・ライブラリのディスクリプタファイル(eel_descriptor.c, fdl_descriptor.c)を登録してください。

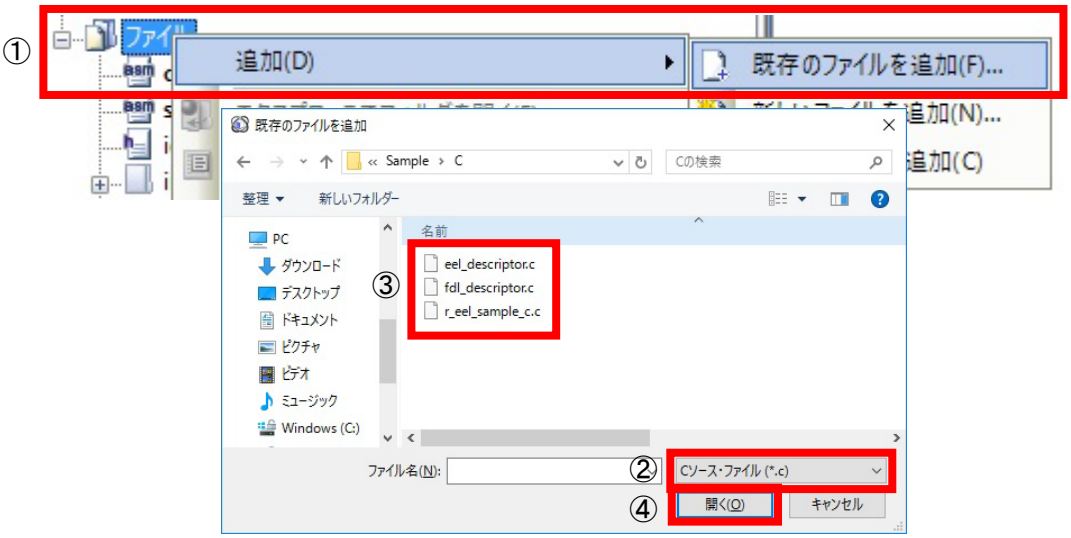

図 6-1 ユーザプログラムファイルの登録

(2) インクルードファイルの設定

CS+の[プロジェクト・ツリー]にある、[ファイル]を右クリックして表示されるリストから[追加]を選択し、[既存のファイルを追加]をクリックすると、[既存のファイルを追加]画面が表示されます(図 6-2)。

次に画面中にある[ファイルの種類]のプルダウンメニューをクリックすると、ファイルの種類の一覧が表示されるので、その中にある[ヘッダファイル(*.h; *.inc)]を選択し、EEPROM エミュレーション・ライブラリ、データ・フラッシュ・ライブラリのヘッダファイル、及びディスクリプタ用ヘッダファイル(eel.h, eel_types.h, fdl.h, fdl_types.h, eel_descriptor.h, fdl_descriptor.h, eel_user_types.h)を登録してください。

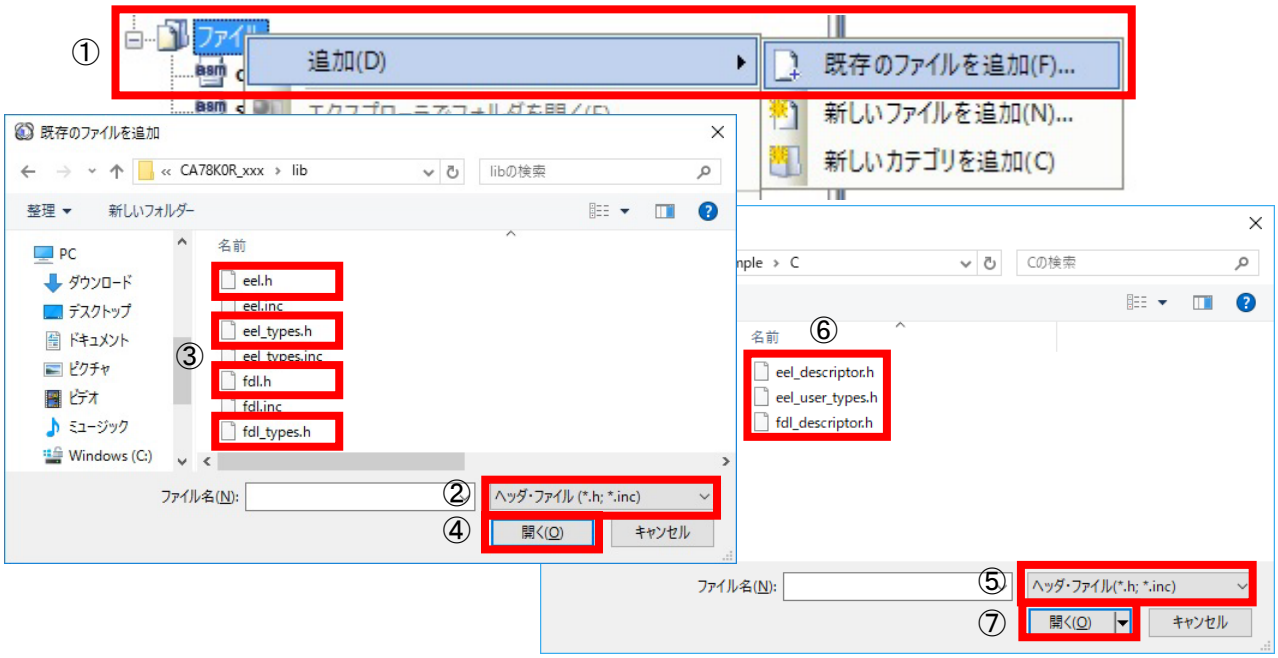

図 6-2 インクルードファイルの登録

(3) ライブラリファイルの設定

CS+の[プロジェクト・ツリー]にある、[ファイル]を右クリックして表示されるリストから[追加]を選択し、[既存のファイルを追加]をクリックすると、[既存のファイルを追加]画面が表示されます(図 6-3)。

次に画面中にある[ファイルの種類]のプルダウンメニューをクリックすると、ファイルの種類の一覧が表示されるので、その中にある[ライブラリファイル(*.lib)]を選択し、EEPROM エミュレーション・ライブラリ、データ・フラッシュ・ライブラリ(eel.lib, fdl.lib)のファイルを登録してください。

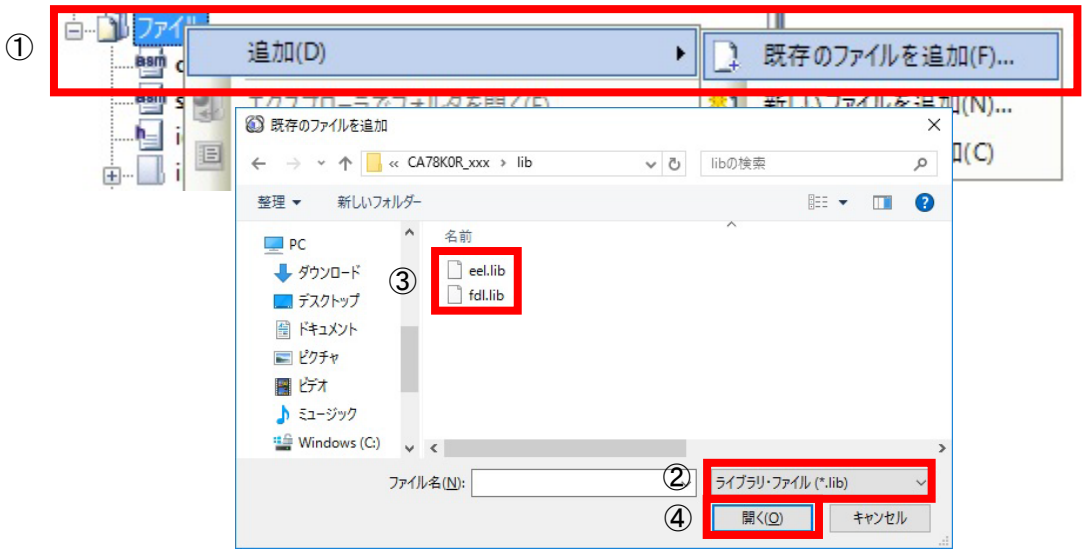

図 6-3 ライブラリファイルの登録

(4) リンクディレクティブの設定(CA78K0R コンパイラ使用時のみ)

CS+の[プロジェクト・ツリー]にある、[ファイル]を右クリックして表示されるリストから[追加]を選択し、[既存のファイルを追加]をクリックすると、[既存のファイルを追加]画面が表示されます(図 6-4)。

次に画面中にある[ファイルの種類]のプルダウンメニューをクリックすると、ファイルの種類の一覧が表示されるので、その中にある[リンクディレクティブファイル(*.dr; *.dir)]を選択し、リンクディレクティブファイル名のユーザファイル(サンプルの場合は r_eel_sample_c.dr 注) を登録してください。

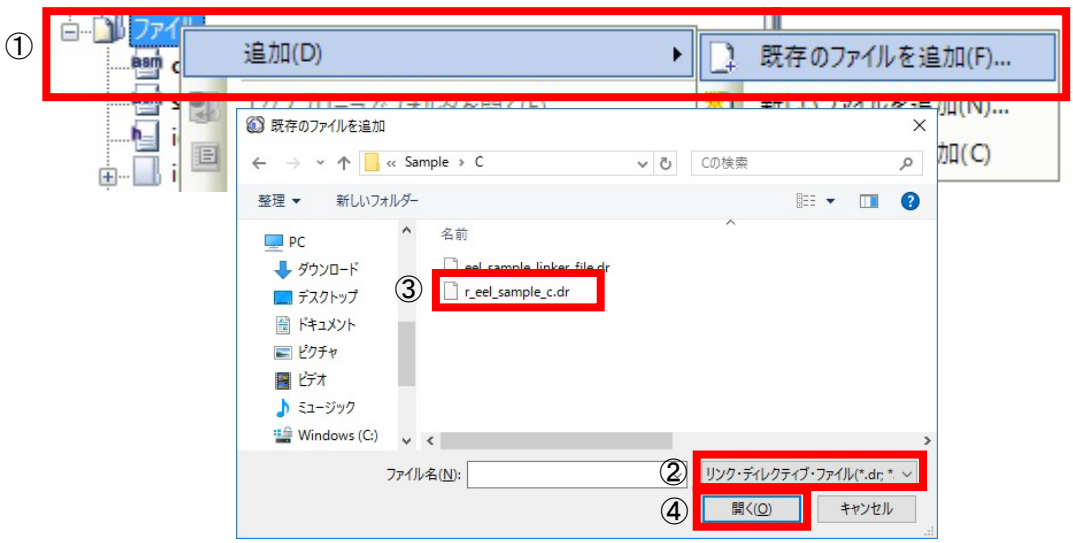

図 6-4 リンクディレクティブファイルの登録

注. 添付のサンプルのリンクディレクティブファイルを使用する場合は編集および修正が必要となる場合があります。

(5) ビルド

CS+のメニュー [ビルド]→[ビルド・プロジェクト] をクリックして、ビルドを実行します。

6.2.2 アセンブリ言語の場合

(1) プロジェクトの作成とソースファイルの設定

CS+でプロジェクトを作成し、表示された画面の左側にある[プロジェクト・ツリー]から、[ファイル]を右クリックして表示されるリストの[追加]を選択し、[既存のファイルを追加]をクリックすると、[既存のファイルを追加]画面が表示されます(図 6-5)。

次に画面中にある[ファイルの種類]のプルダウンメニューをクリックすると、ファイルの種類の一覧が表示されるので、その中にある[アセンブルファイル(*.asm)]を選択し、ソースファイルとしてユーザプログラムのファイル、及び EEPROM エミュレーション・ライブラリ、データ・フラッシュ・ライブラリのディスクリプタファイル(eel_descriptor.asm, fdl_descriptor.asm)を登録してください。

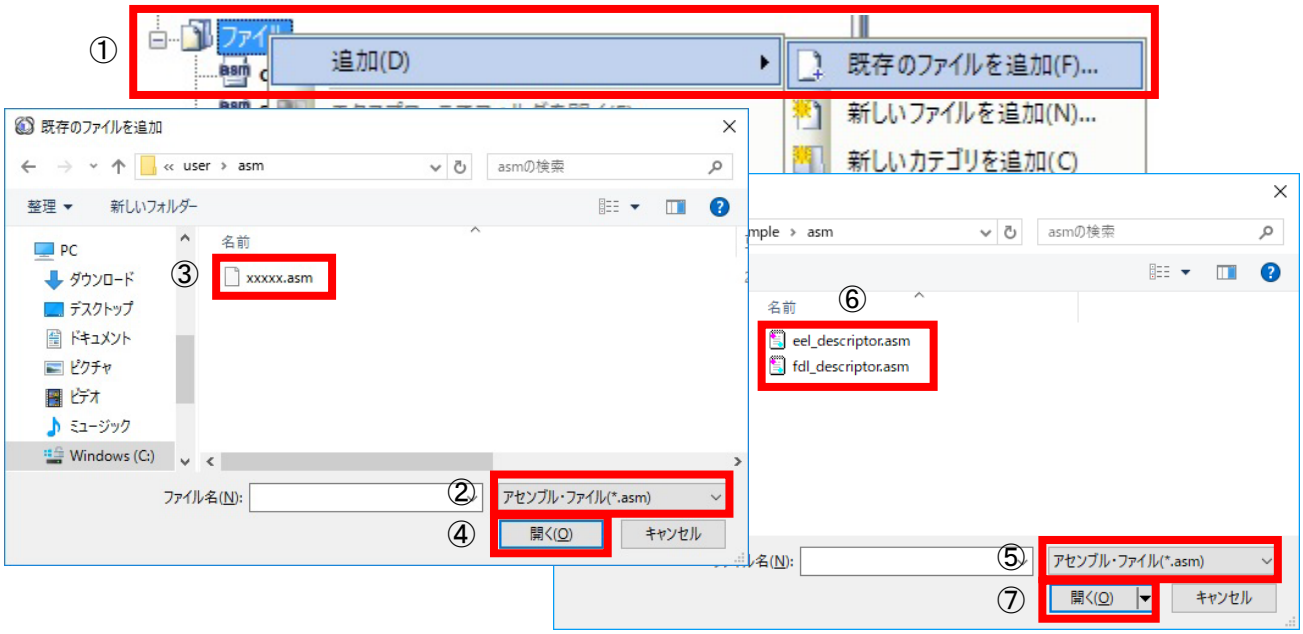

図 6-5 ユーザプログラムファイルの登録

(2) インクルードファイルの設定

CS+の[プロジェクト・ツリー]にある、[ファイル]を右クリックして表示されるリストから[追加]を選択し、[既存のファイルを追加]をクリックすると、[既存のファイルを追加]画面が表示されます(図 6-6)。

次に画面中にある[ファイルの種類]のプルダウンメニューをクリックすると、ファイルの種類の一覧が表示されるので、その中にある[ヘッダファイル(*.h; *.inc)]を選択し、EEPROM エミュレーション・ライブラリ、データ・フラッシュ・ライブラリのヘッダファイル、及びディスクリプタ用ヘッダファイル(eel.inc, eel_types.inc, fdl.inc, eel_descriptor.inc, fdl_descriptor.inc)を登録してください。

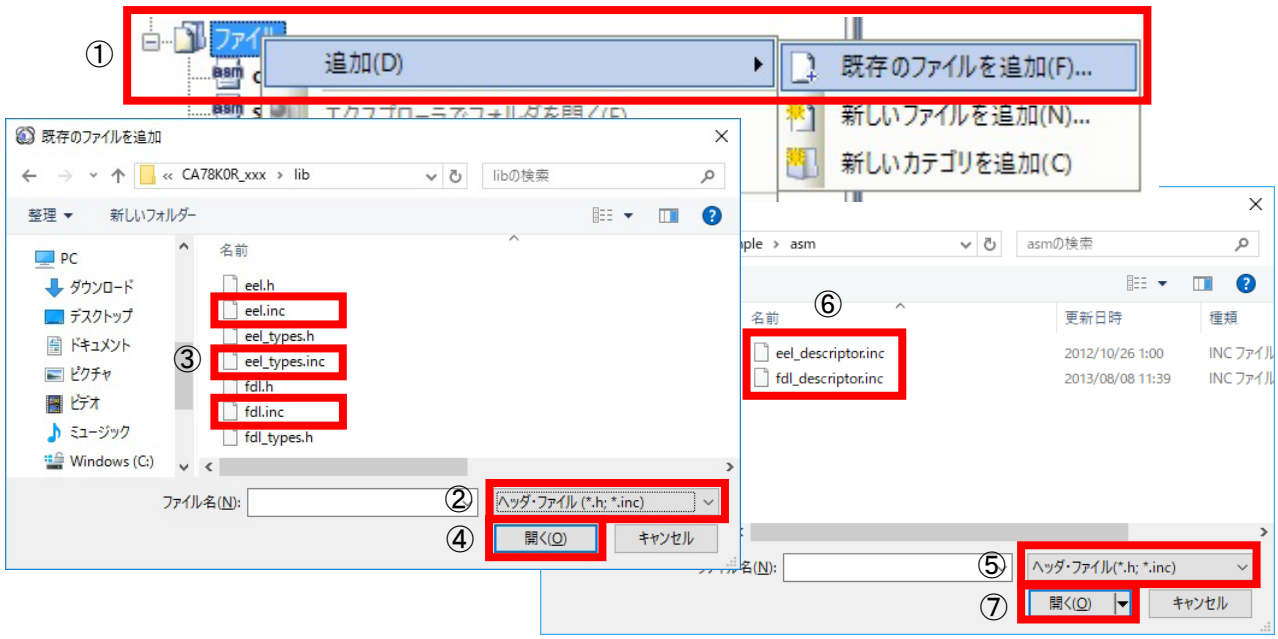

図 6-6 インクルードファイルの登録

(3) ライブラリファイルの設定

CS+の[プロジェクト・ツリー]にある、[ファイル]を右クリックして表示されるリストから[追加]を選択し、[既存のファイルを追加]をクリックすると、[既存のファイルを追加]画面が表示されます(図 6-7)。

次に画面中にある[ファイルの種類]のプルダウンメニューをクリックすると、ファイルの種類の一覧が表示されるので、その中にある[ライブラリファイル(*.lib)]を選択し、EEPROM エミュレーション・ライブラリ、データ・フラッシュ・ライブラリ(eel.lib, fdl.lib)のファイルを登録してください。

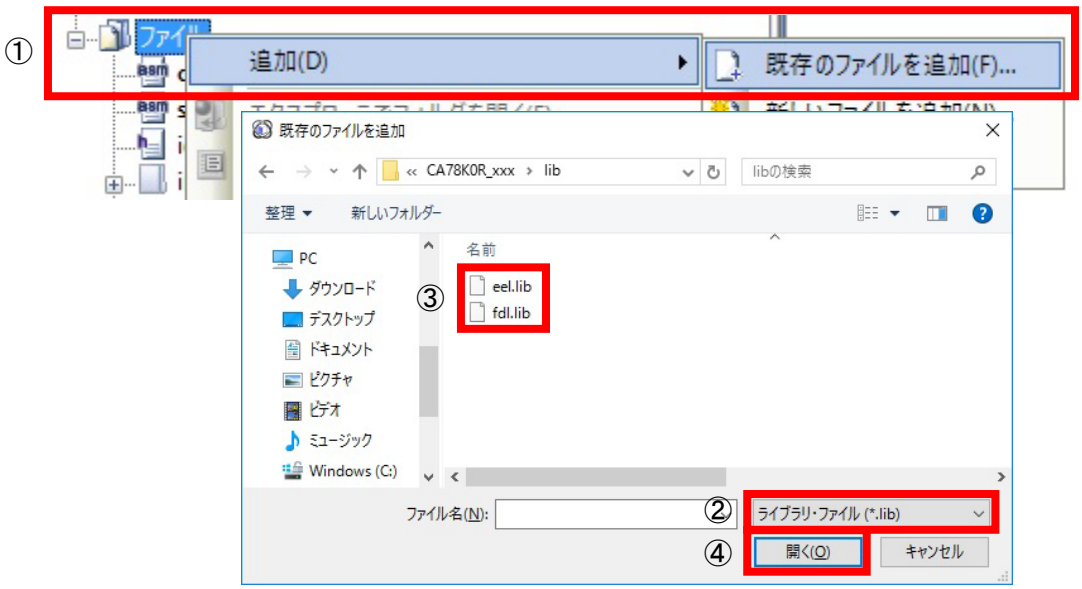

図 6-7 ライブラリファイルの登録

(4) リンクディレクティブの設定(CA78K0R コンパイラ使用時のみ)

CS+の[プロジェクト・ツリー]にある、[ファイル]を右クリックして表示されるリストから[追加]を選択し、[既存のファイルを追加]をクリックすると、[既存のファイルを追加]画面が表示されます(図 6-8)。

次に画面中にある[ファイルの種類]のプルダウンメニューをクリックすると、ファイルの種類の一覧が表示されるので、その中にある[リンクディレクティブファイル(*.dr; *.dir)]を選択し、リンクディレクティブファイル名のユーザファイルを登録してください。

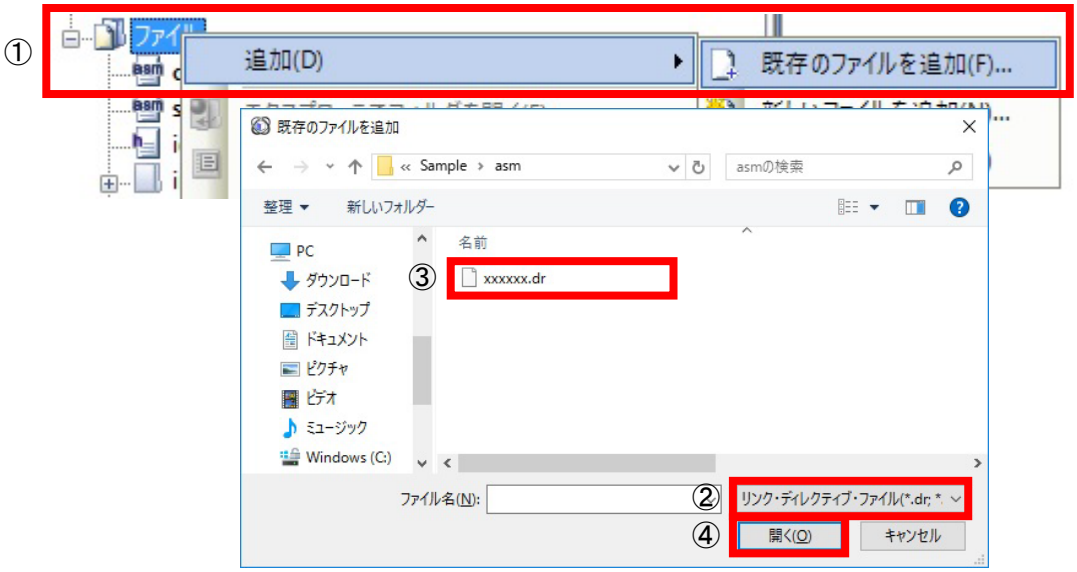

図 6-8 リンクディレクティブファイルの登録

(5) 自動生成されたファイルの除外(CC-RL コンパイラ使用時のみ)

CC-RL コンパイラ用 CS+では、[プロジェクト・ツリー]の[ファイル]にいくつかの自動生成されるファイルがあります。この中で、"main.c"の処理は、既に EEPROM エミュレーション・ライブラリのサンプルプログラムに含まれています。その為、このファイルをプロジェクトから除外する必要があります(図 6-9)。

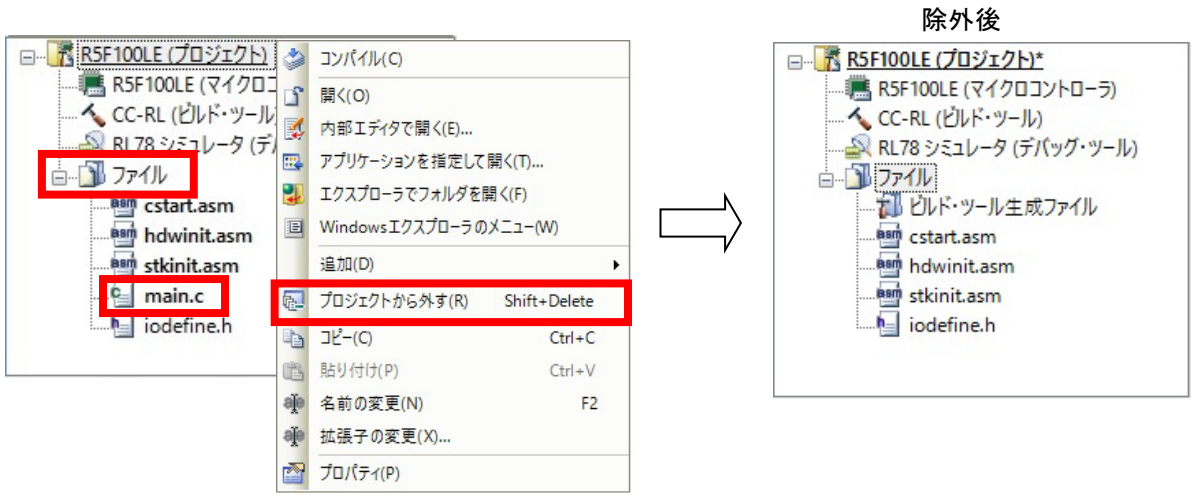

図 6-9 自動生成ファイルの除外

(6) ビルド

CS+のメニュー [ビルド] → [ビルド・プロジェクト] をクリックして、ビルドを実行します。

6.3 e² studioでのビルド

e² studio を用いて EEPROM エミュレーション・ライブラリ Pack02 をユーザプログラムに組み込んでビルドする手順を説明します。e² studio の対象コンパイラは CC-RL コンパイラと LLVM コンパイラです。

6.3.1 プロジェクトの作成

e² studio を起動し、[ファイル]メニューの[新規]から[C/C++ Project]を選択し、"新規 C/C++ プロジェクトのテンプレートウインドウを起動します（図 6-10）。

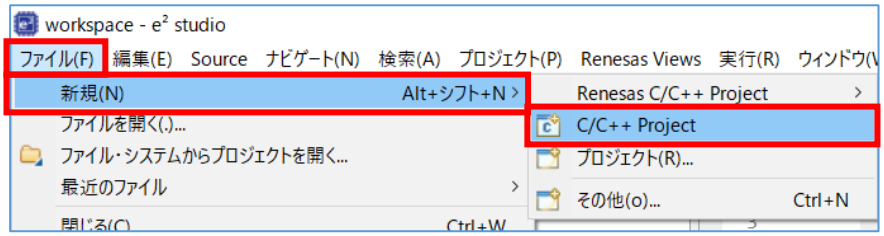

図 6-10 新規プロジェクトの作成

- CC-RL コンパイラを使用する場合は、[Renesas CC-RL C/C++ Executable Project]を選択、"次へ"ボタンを押します（図 6-11）。

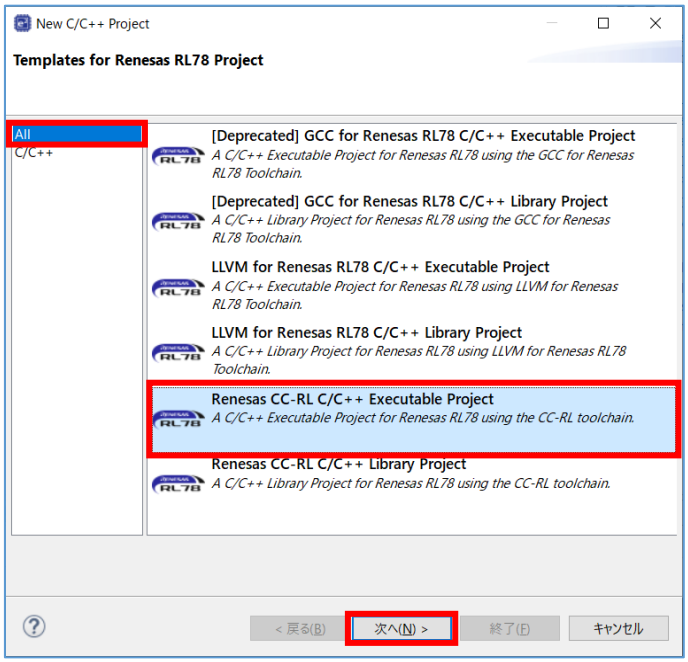

図 6-11 ツールチェインに CC-RL コンパイラを選択する場合

"New Renesas CC-RL Executable Project"ウィンドウで、プロジェクト名を入力して"次へ"ボタンを押します。

- LLVM コンパイラを使用する場合は、[LLVM for Renesas C/C++ Executable Project]を選択、"次へ"ボタンを押します(図 6-12)。

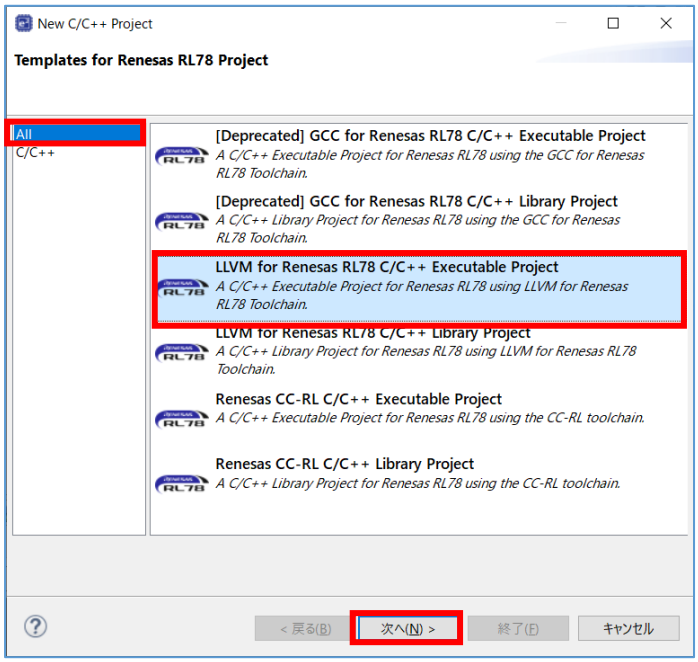

図 6-12 ツールチェインに LLVM コンパイラを選択する場合

"New LLVM for Renesas RL78 Executable Project"ウィンドウで、プロジェクト名を入力して"次へ"ボタンを押します

[Device Settings]の[ターゲット・デバイスで、"RL78 – G13" - "R5F100LE"を選択します。
[Configurations]で"Hardware Debug 構成を生成"にチェックが入った状態で E2 Lite (RL78)を選択し、[終了]ボタンを押します(図 6-13)。デバッグ・ツールに E2 Lite を選択し、オンチップ・デバッグを実施することを前提としています。

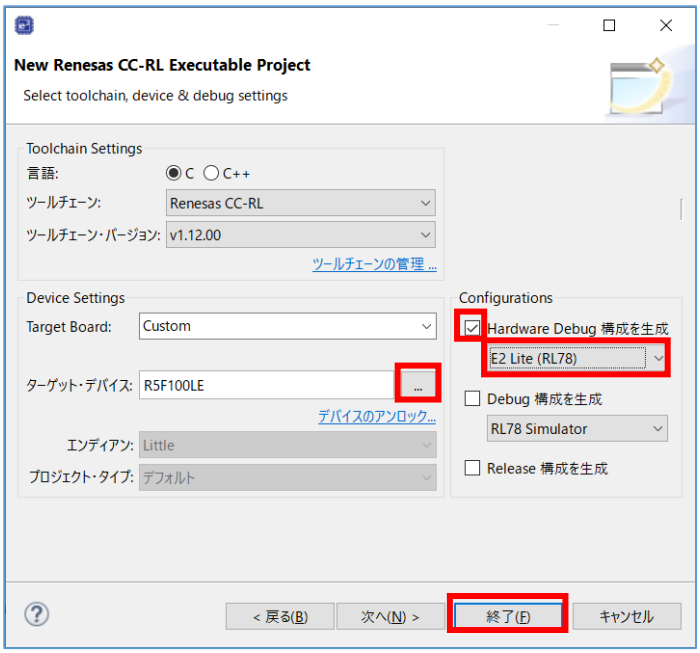

図 6-13 デバイスの選択

6.3.2 C言語の場合

(1) ソース、インクルードファイルの設定

・作成したプロジェクトへの、EEPROM エミュレーション・ライブラリファイル、およびデータ・フラッシュ・ライブラリファイルの追加

- CC-RL では、e2 studio が出力する"src"フォルダに"eel.h", "eel_types.h", "eel.lib", "fdl.h", "fdl_types.h", "fdl.lib"を登録します。また、ディスクリプタファイル"eel_descriptor.c", "eel_descriptor.h", "eel_user_types.h", "fdl_descriptor.c". "fdl_descriptor.h"とサンプルプログラムファイル"r_eel_sample_c.c "を登録してください（図 6-14）。

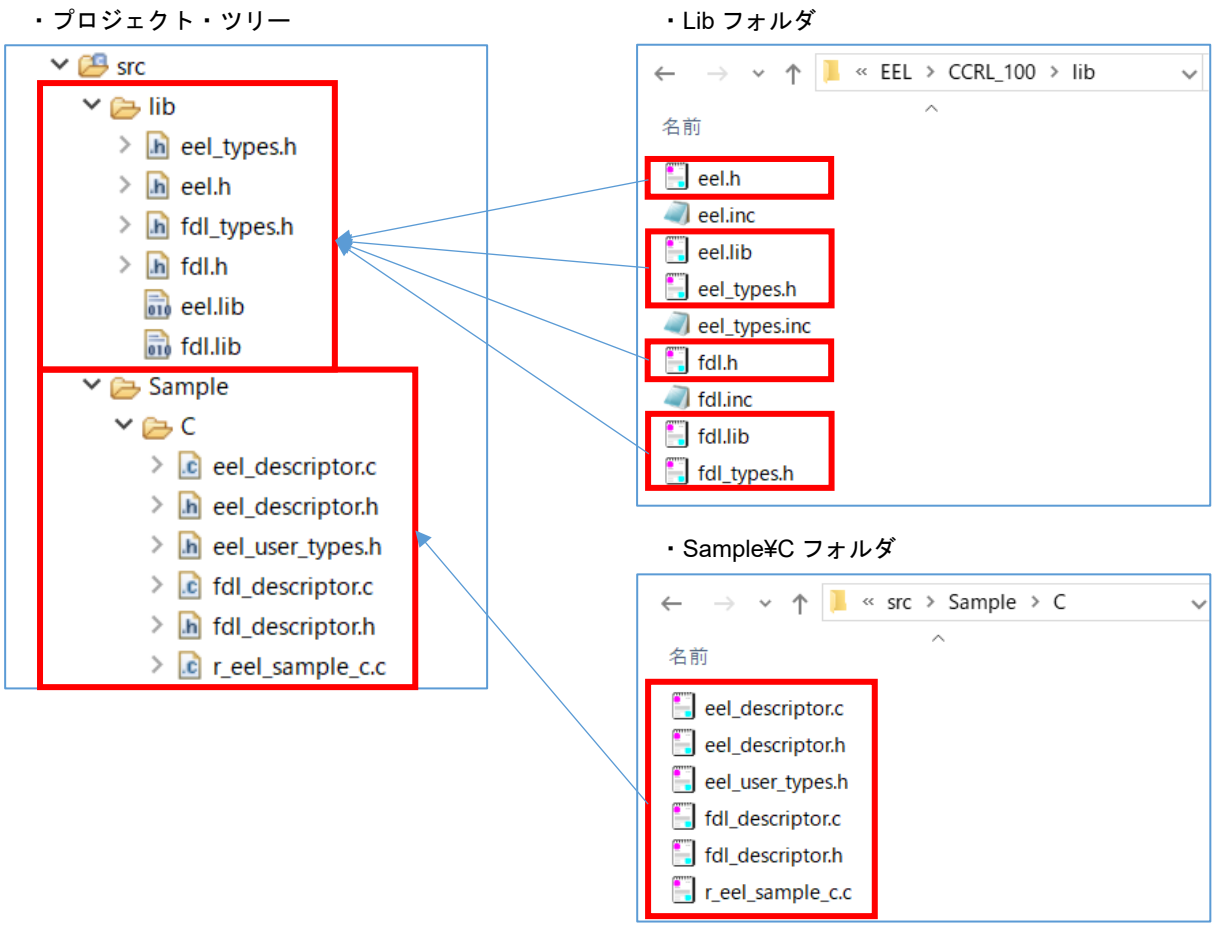

図 6-14 ソース、インクルードファイルの登録(CC-RL)

- LLVM では、e2 studio が出力する"src"フォルダに"eel.h", "eel_types.h", "libeel.a", "fdl.h", "fdl_types.h", "libfdl.a"を登録します。また、ディスクリプタファイル"eel_descriptor.c", "eel_descriptor.h", "eel_user_types.h", "fdl_descriptor.c". "fdl_descriptor.h"とサンプルプログラムファイル"r_eel_sample_c.c ", "r_eel_sample_c.ld"を登録してください。(図 6-15)

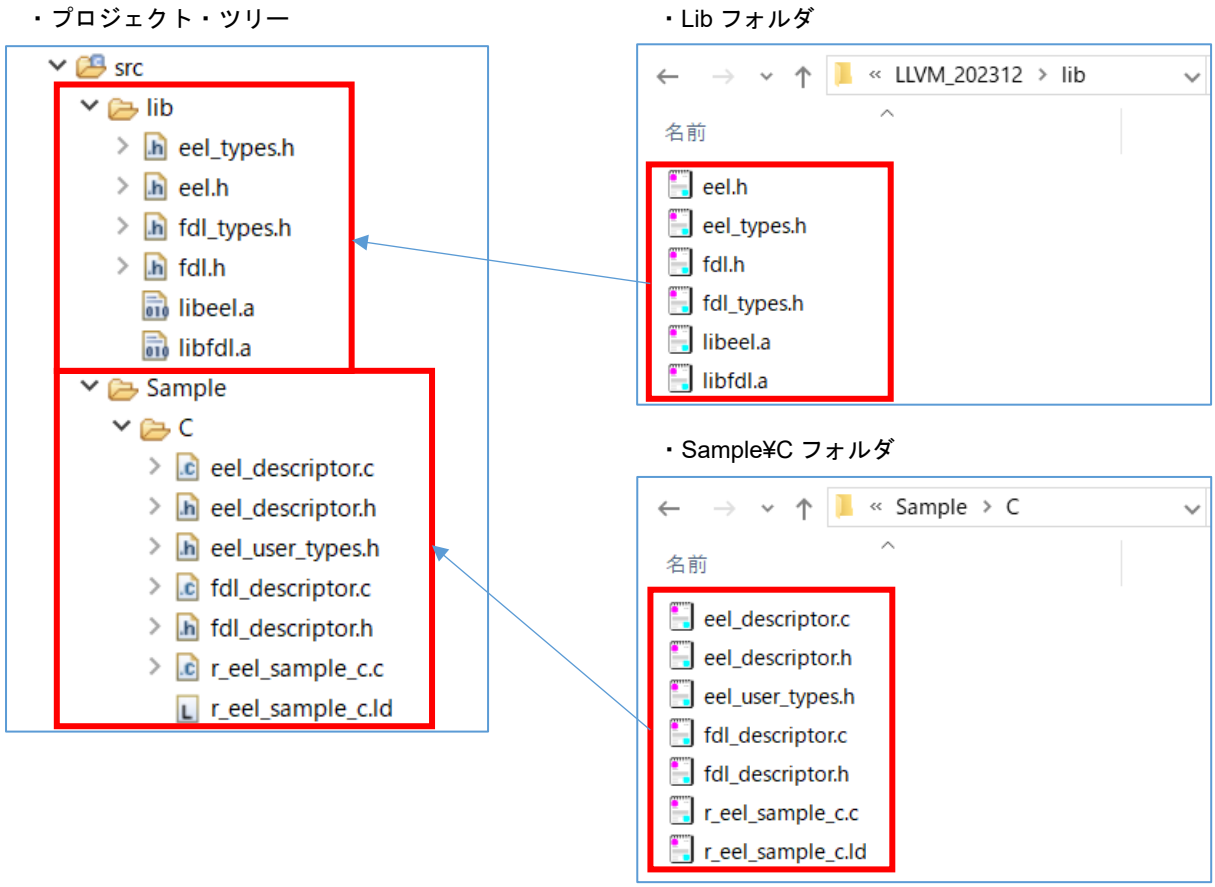

図 6-15 ソース、インクルードファイルの登録(LLVM)

・統合開発環境の機能により自動的に追加されたファイルの除外

作成されたプロジェクトには、自動的に追加されるファイルがあります。これらと同様のファイルは、EEL Pack02 の"sample"フォルダ内にも存在するため、プロジェクト・ツリーで各ファイルを選択し、各統合開発環境の機能を使用して、プロジェクトから外します。

プロジェクト・ツリー上のファイルをマウス右クリック、"プロパティ"で表示された[設定]画面の、"ビルドからリソースを除外"にチェックを入れ、対象ファイルを除外します（フォルダから削除も可能）。

- CC-RL コンパイラ使用時は、[プロジェクト名]/src フォルダ内の[プロジェクト名].c（例："EELPack02_PJ01.c"）が対象。
- LLVM コンパイラ使用時は、[プロジェクト名]/generate フォルダ内の"linker_script.ld"と、[プロジェクト名]/src フォルダ内の[プロジェクト名].c（例："EELPack02_PJ01.c"）が対象。

(2) ライブラリファイルの設定

- CC-RL コンパイラの場合、e^2 studio のプロジェクト・ツリーの[プロジェクト]上でマウスの右クリックで"プロパティ"を選択、"C/C++ビルド" [設定] － "Linker" [入力]で表示された画面の"リンクするリロケータブル・ファイル、オブジェクト・ファイル、およびライブラリー・ファイル"の右側にある＋ボタンをクリックして表示された"ファイルの追加"ウィンドウで[形式]を"library"に変更してから、ライブラリファイル("eel.lib", "fdl.lib")のパスを登録します（図 6-16）。

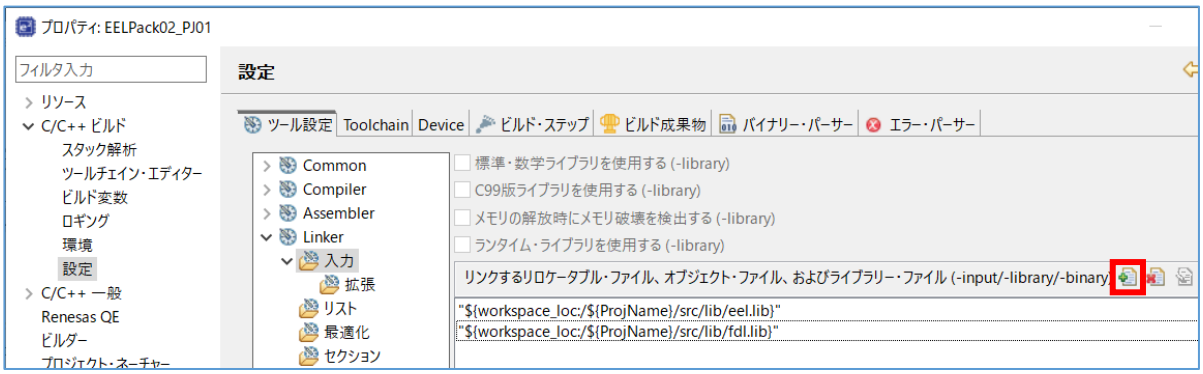

図 6-16 (a) ライブラリファイルの登録(CC-RL)

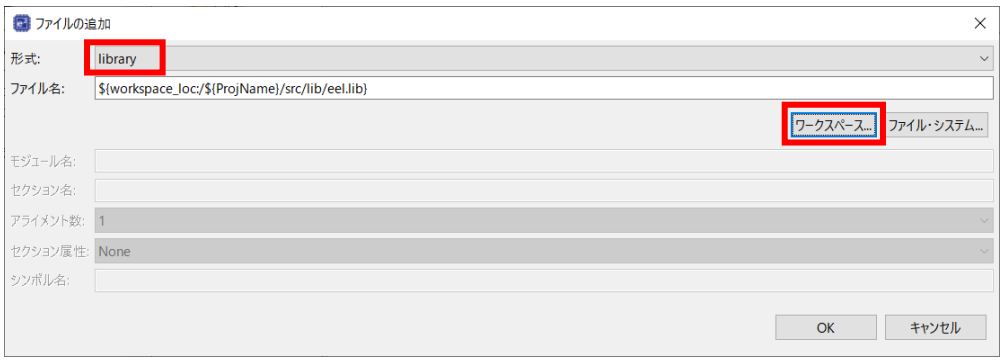

図 6-16 (b) ライブラリファイルの登録(CC-RL)

- LLVM コンパイラの場合、e2 studio のプロジェクト・ツリーの[プロジェクト]上でマウスの右クリックで"プロパティ"を選択、"C/C++ビルド" [設定] － "Linker" [Source]で表示された画面の"Additional input files"欄に、ライブラリファイル("libeel.a", "libfdl.a")のパスを登録します(図 6-17)。

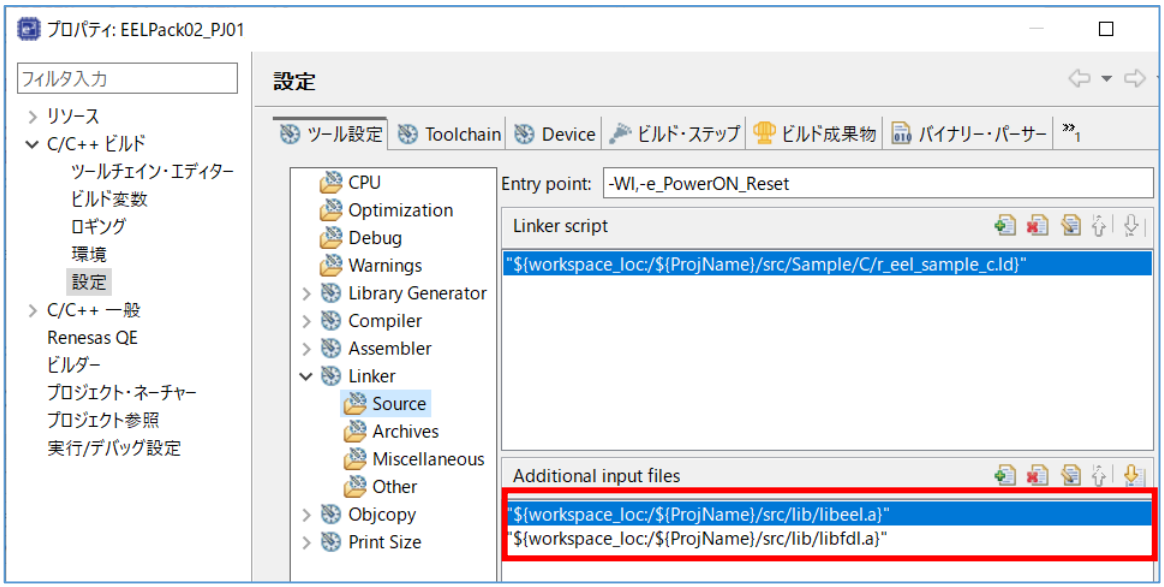

図 6-17 ライブラリファイルの登録(LLVM)

(3) リンカスクリプトファイルの設定(LLVM コンパイラ使用時のみ)

LLVM コンパイラの場合、e2 studio のプロジェクト・ツリーの[プロジェクト]上でマウスの右クリックで"プロパティ"を選択、"C/C++ビルド" [設定] － "Linker" [Source]で表示された画面の"Linker script"欄に、リンカスクリプトファイル"r_eel_sample_c.ld"のパスを設定します(図 6-18)。

ここでは、EEL Pack02 用に準備されている"r_eel_sample_c.ld"ファイルを登録します。

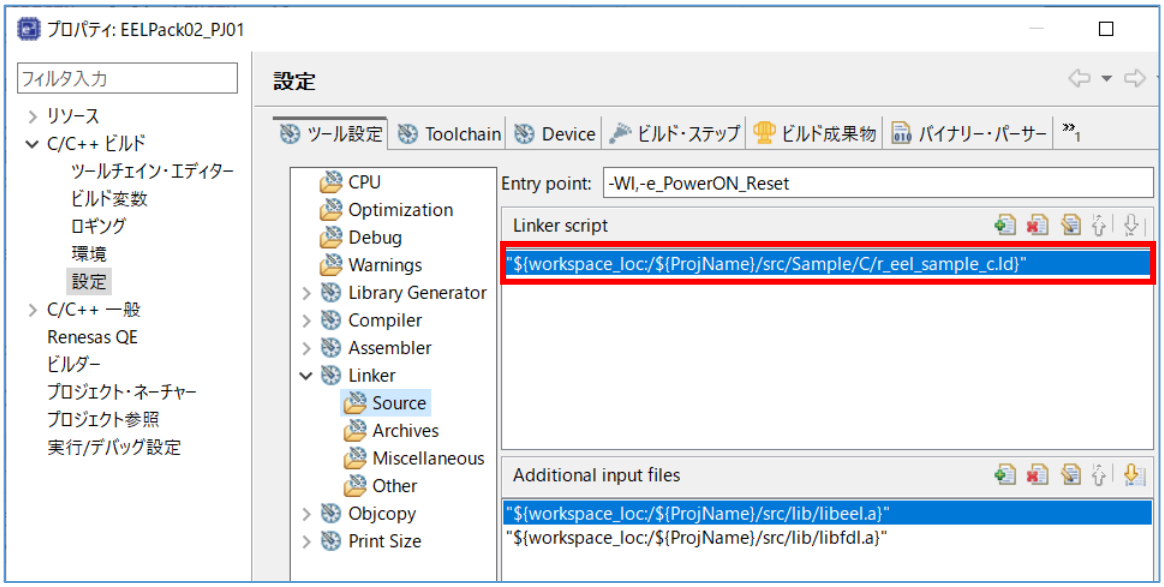

図 6-18 リンカスクリプトファイルの登録(LLVM のみ)

注. リンカスクリプトファイル(*.ld)の記述内容、及び記述方法の詳細については、LLVM のリファレンスマニュアルをご参照ください。

(4) ビルド

e2 studio のプロジェクト・ツリーの[プロジェクト]上でマウスの右クリックで"プロジェクトのビルド"を選択してビルドを実行します。

6.3.3 アセンブリ言語の場合（CC-RLコンパイラ使用時のみ）

(1) ソース、インクルードファイルの設定

・作成したプロジェクトへの、EEPROM エミュレーション・ライブラリファイル、およびデータ・フラッシュ・ライブラリファイルの追加

e^2 studio が出力する"src"フォルダにライブラリファイル"eel.inc", "eel_types.inc", "eel.lib", "fdl.inc", "fdl.lib", "eel_descriptor.inc", "fdl_descriptor.inc"を登録します。また、ディスクリプタファイル"eel_descriptor.asm", "fdl_descriptor.asm"とユーザプログラムファイル"xxxxxx.asm"を登録します（図 6-19）。

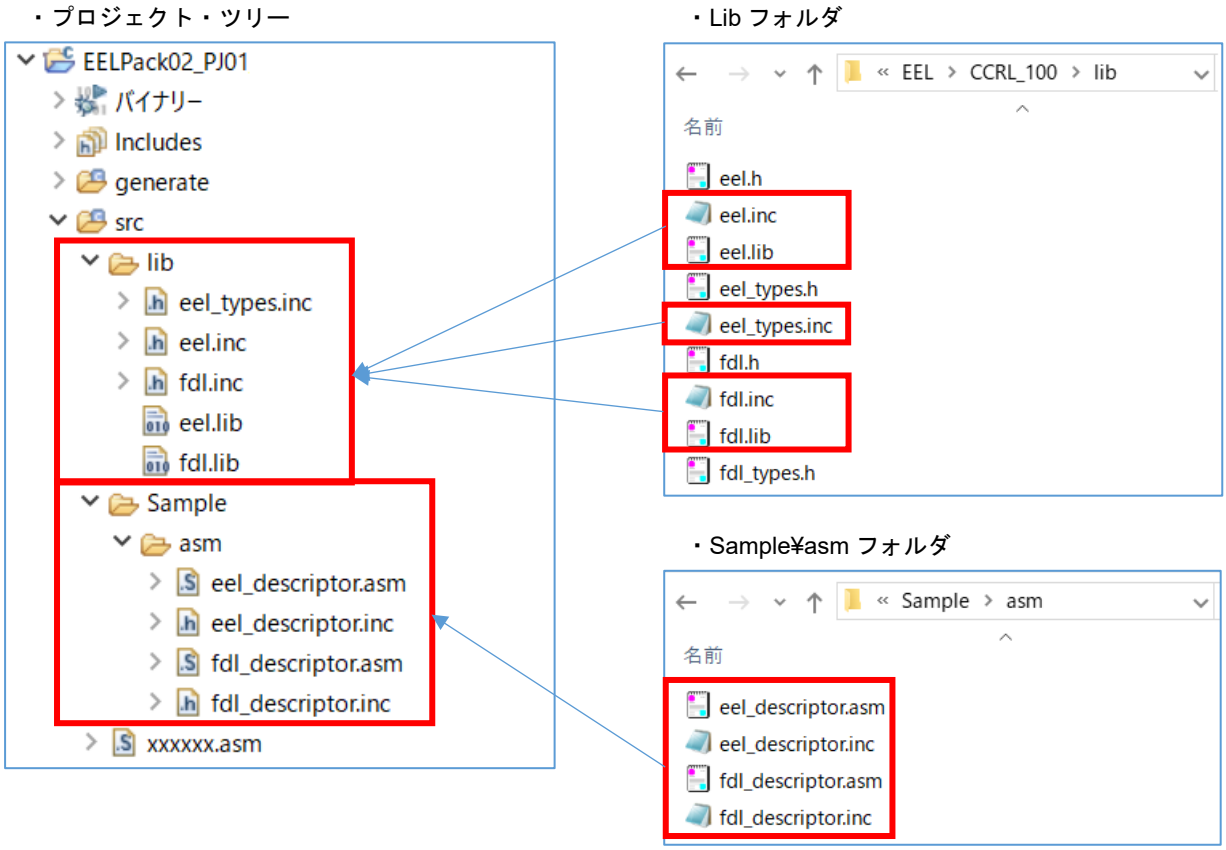

図 6-19 ソース、インクルードファイルの登録

・統合開発環境の機能により自動的に追加されたファイルの除外

作成されたプロジェクトには、自動的に追加されるファイルがあります。これらと同様のファイルは、EEL Pack02 の"sample"フォルダ内にも存在するため、ツリーで各ファイルを選択し、統合開発環境の機能を使用して、プロジェクトから外します。

プロジェクト・ツリー上のファイルをマウス右クリック、"プロパティ"で表示された[設定]画面で、"ビルドからリソースを除外"にチェックを入れ、対象ファイルを除外します（フォルダから削除も可能）。

- [プロジェクト名]/src フォルダ内の[プロジェクト名].c（例："EELPack02_PJ01.c"）が対象。

(2) ライブラリファイルの設定

- e² studio のプロジェクト・ツリーの[プロジェクト]上でマウスの右クリックで"プロパティ"を選択、"C/C++ビルド" [設定] － "Linker" [入力]で表示された画面の"リンクするリロケータブル・ファイル、オブジェクト・ファイル、およびライブラリー・ファイル"の右側にある＋ボタンをクリックして表示された"ファイルの追加"ウィンドウで[形式]を"library"に変更してから、ライブラリファイル("eel.lib", "fdl.lib")のパスを登録します（図 6-20）。

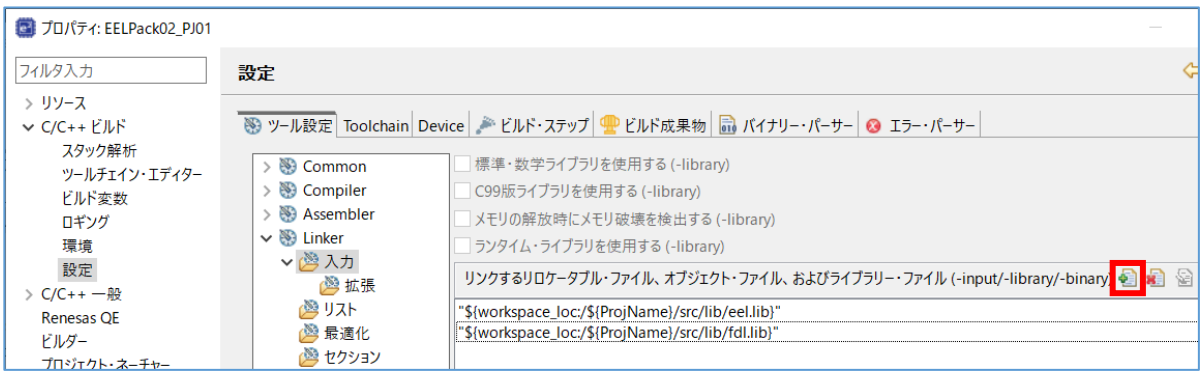

図 6-20 (a) ライブラリファイルの登録

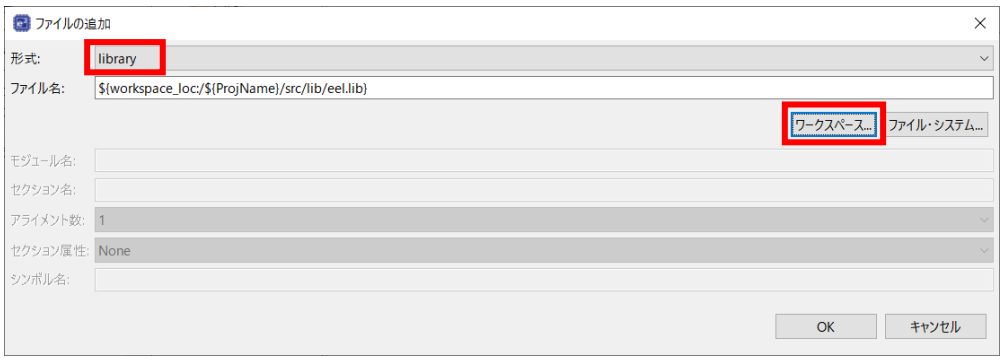

図 6-20 (b) ライブラリファイルの登録

(3) ビルド

e² studio のプロジェクト・ツリーの[プロジェクト]上でマウスの右クリックで"プロジェクトのビルド"を選択してビルドを実行します。

6.4 ビルド時の注意事項

6.4.1 CA78K0Rコンパイラ使用時

(1) オンチップ・デバッグ機能使用時

CS+でオンチップ・デバッグ機能を有効にした後、プログラムのビルドを行うと、以下のようなエラーが発生する場合があります。

```
RA78K0R error E3212: Default segment can't allocate to memory - ignored
                     Segment '??OCDROM' at xxxxxH-200H
```

このエラーは、オンチップ・デバッグ機能で使用するモニター領域(OCDROM)用のセグメント配置ができないために発生していますので、エラーを回避するためには、プロジェクトに組み込んでいるリンクディレクティブファイル(*.dr)に、以下のような内容を追記し、セグメント配置用の領域を別途用意してください。

```
MEMORY OCD_ROM : ( 0xxxxxH, 00200H )
```

備考 1. xxxxx：エラーが発生した箇所の先頭アドレス
2. 領域名「OCD_ROM」は参考例としての表記となります。

6.4.2 CC-RLコンパイラ使用時

(1) オンチップ・デバッグ機能使用時

CS+でオンチップ・デバッグ機能を有効にした後、プログラムのビルドを行うと、以下のようなエラーが発生する場合があります。

```
E0562321:Section ".monitor2" overlaps section "xxxxx"
```

備考 1. xxxxx：セクション名

このエラーは、オンチップ・デバッグ機能で使用するモニター領域(OCDROM)用のセクション配置ができないために発生しています。エラーを回避するためには、CS+の[プロジェクト・ツリー]から[①CC-RL(ビルド・ツール)]を選択、右クリックで表示されるプロパティを選択し、表示された②"CC-RL のプロパティ"の③"リンク・オプション"タブを選択します。④"セクション"項目内の⑤"セクションの開始アドレス"で、オンチップ・デバッガモニタ用のセクション配置領域(monitor2 : R5F100LE では、初期アドレス 0xFE00-0xFFFF)に、他のセクションが重複しないよう配置を修正してください(図 6-21)。

セクションの設定の詳細については、CC-RL コンパイラ ユーザーズマニュアルをご確認ください。

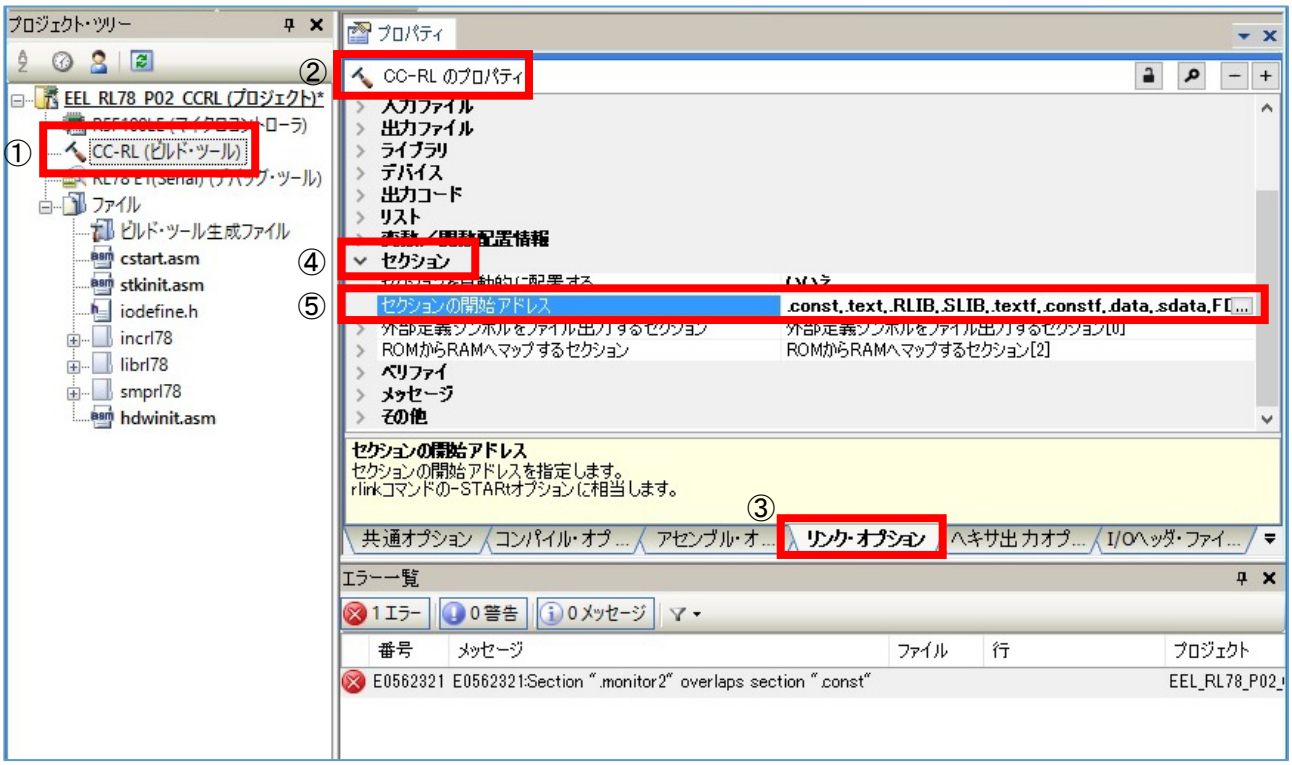

図 6-21 セクション配置の変更

第7章 デバッグ方法

IECUBE、またはオンチップ・デバッグ・エミュレータ E1、E2、E2 エミュレータ Lite、及び E20 を使用してデバッグを行う場合につきましては、以下の資料を参照してください。

タイトル
CubeSuite+ 統合開発環境 ユーザーズマニュアル RL78 デバッグ編 [CS+ for CA,CX]
CS+ 統合開発環境 ユーザーズマニュアル RL78 デバッグ・ツール編 [CS+ for CC]
e² studio 統合開発環境 ユーザーズマニュアル 入門ガイド

注. これらのドキュメントは、弊社HPの「統合開発環境 CS+（旧 CubeSuite+）」、および「統合開発環境 e² studio」のページからダウンロードしてください。

7.1 デバッグ時の注意事項

EEPROM エミュレーション・ライブラリ Pack02 を使用してオンチップ・デバッグ・エミュレータ E1、E2、E2 エミュレータ Lite、もしくは E20 を使用する場合の注意事項

(1) CubeSuite+ Ver.1.01 より前のバージョンで、オンチップ・デバッグ・エミュレータ E1、もしくは E20 を使用して EEPROM エミュレーション・ライブラリ Pack02 のコマンドを実行した場合、シーケンサが完了した事を確認するまでブレイクを実行しないでください。シーケンサが正常に動作できなくなります。

(2) フラッシュ・ライブラリのデバッグはシミュレータでは実行できません。デバッグを行う場合は、RL78 マイクロコントローラのオンチップ・デバッグ機能を使用するか、もしくは IECUBE をご用意ください。

第8章 サンプルプログラム

添付のサンプルプログラム(r_eel_sample_c.c)は、R5F100LEA(RL78/G13)を対象に QB-R5F100LE-TB ボードで EEPROM エミュレーション・ライブラリ Pack02 の使用方法を簡単に確認することが可能なように用意しているプログラムです。あくまで参考例となりますので、必ずサンプルプログラム通りに作成する必要があるわけではありません。簡易的な動作確認用のプログラムとしてご使用ください。

- CA78K0R コンパイラ用サンプルプログラムのリンクディレクティブファイル(r_eel_sample_c.dr)は、サンプルプログラムで使用するスタックやデータバッファ等を配置禁止領域[注 1] に配置しないように指定する事を目的としています。サンプルプログラムを使用する場合は、こちらのファイルも一緒に組み込んでください。[注 2]
- CC-RL コンパイラ用サンプルプログラムでは、CS+、または e^2 studio の画面上で、サンプルプログラムで使用するスタックやデータバッファ等を配置禁止領域[注 1] に配置しないように [セクション]配置を設定する必要があります。[注 2]
- LLVM コンパイラ用リンカスクリプトファイル(r_eel_sample_c.ld)は、サンプルプログラムで使用するスタックやデータバッファ等を配置禁止領域[注 1] に配置しないように指定する事を目的としています。サンプルプログラムを使用する場合は、こちらのファイルも一緒に組み込んでください。[注 2]

注 1. 詳細については、EEPROM エミュレーション・ライブラリ Pack02 のユーザーズマニュアル「6.2 ソフトウェア・リソース」を参照してください。
2. ご使用の環境やプログラムの変更によっては使用中のデータが意図しない領域へ配置される場合があります。実行モジュール作成後はマップファイル(*.map)を確認し、プログラムやデータの配置状態を必ず確認してください。また、各コードやデータの定義方法や配置条件等については、使用するコンパイラのユーザーズマニュアルを参照してください。

8.1 サンプルプログラムの初期設定

サンプルプログラムを以下の初期設定で動作させます。変更が必要な場合は、サンプルプログラムを修正してください。

- CPU の動作周波数 : 高速オンチップ・オシレータ 32MHz
- フラッシュ書き換えモード : フルスピード・モード

8.2 オプション・バイトとオンチップ・デバッグの設定について

(1) CS+でCA78K0R、またはCC-RLコンパイラを使用する場合

オンチップ・デバッグを行う場合は「オンチップ・デバッグの許可／禁止をリンク･オプションで設定する」を「はい」に設定し、「オンチップ・デバッグ・オプション・バイト制御値」を「84」に指定してください。

CC-RLコンパイラ用では、「デバッグ・モニタ領域を設定する」を「はい」に指定してください。

サンプルプログラムは高速オンチップ・オシレータを32MHzに設定する事で正常に動作します。ユーザ・オプション・バイトを設定するには、CS+の「リンク・オプション」タブを選択し、「ユーザ・オプション・バイトを設定する」を「はい」に設定した後、ユーザ・オプション・バイト値を「xxxxE8」に指定してください(図8-1)。

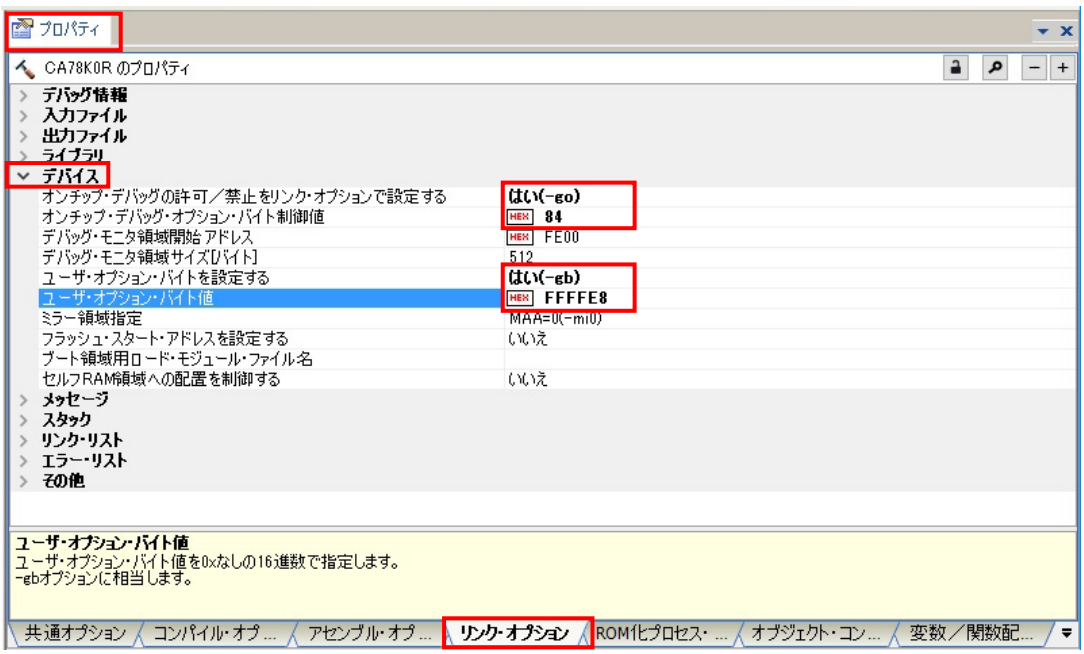

図 8-1(a) CS+使用時のオプション・バイトの設定(CA78K0R コンパイラ)

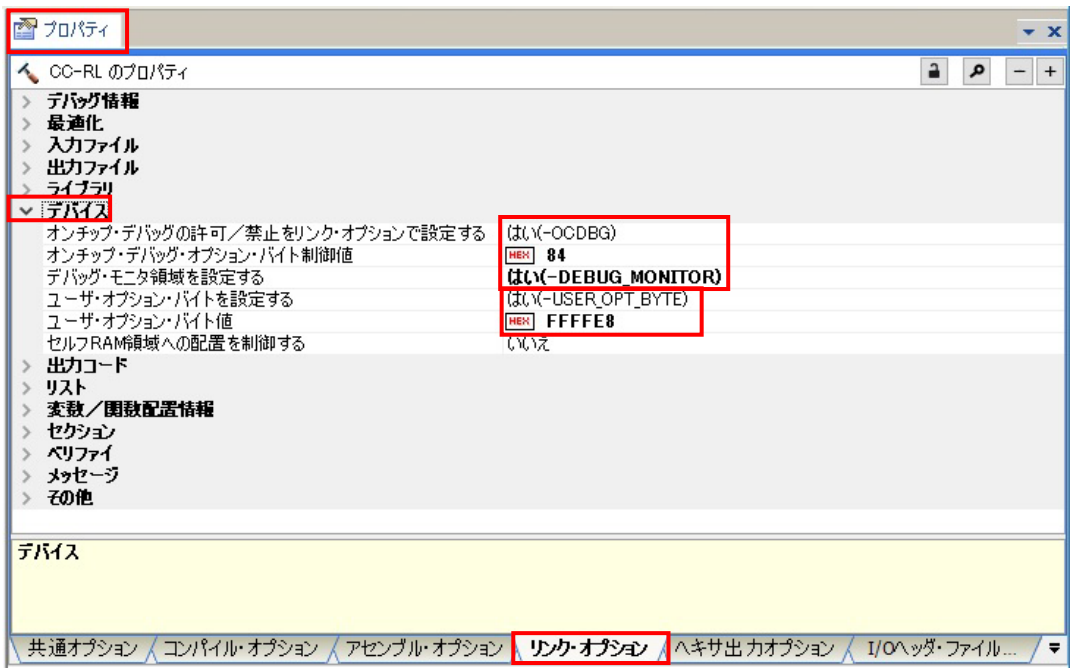

図8-1(b) CS+使用時のオプション・バイトの設定(CC-RLコンパイラ)

(2) e² studio で CC-RL コンパイラを使用する場合

"C/C++ビルド" [設定] – "Linker" [デバイス] で表示された画面でデバイス項目の設定をします。

オンチップ・デバッグを行う場合は、[オプション・バイト領域のオンチップ・デバッグ・オプション・バイトに値を設定する(-ocdbg)]にチェックをいれ、「オンチップ・デバッグ制御値」に「84」を指定してください。

また、サンプルプログラムは高速オンチップ・オシレータを32MHzに設定する事で正常に動作します。オプション・バイト領域のユーザ・オプション・バイト値を設定する(-user_opt_byte)]にチェックをいれ、[ユーザ・オプション・バイト値(-user_opt_byte=<value>)]に「xxxxE8」を指定し、高速オンチップ・オシレータを32MHzに設定してください(図8-2)。

図 8-2 e² studio 使用時のオプション・バイトの設定(CC-RL コンパイラ)

(3) e² studio で LLVM コンパイラを使用する場合

デバイス項目の設定は、e² studio から出力される"vects.c"ファイルで設定します。

対象ファイルパス："プロジェクトフォルダ"¥generate¥vects.c

サンプルプログラムは、高速オンチップ・オシレータを32MHzに設定する事で正常に動作します。そのため、"vects.c"ファイルの"Option_Bytes"にユーザ・オプション・バイト値「xxxxe8」とオンチップ・デバッグ・オプション・バイト値を次のように設定します。

[RL78/G13の設定例]

"0xff, 0xff, 0xe8, 0x84" (WDT動作許可, LVDオフ, HSモード/32MHz/, オンチップ・デバッグ動作許可)

```
const unsigned char Option_Bytes[]  __attribute__ ((section (".option_bytes"))) = {
        0xff, 0xff, 0xe8, 0x84
};
```

注. オンチップ・デバッグを実施することを前提とした設定例です。
対象デバイスのユーザーズマニュアルで「オプション・バイト」の章の「ユーザ・オプション・バイト」「オンチップ・デバッグ・オプション・バイト」の内容をご確認いただき、設定値を書き込んでください。

8.3 内蔵RAM領域の定義

8.3.1 CA78K0Rコンパイラ使用時

CA78K0R コンパイラ使用時の初期状態では、内蔵 RAM 領域については全域が"RAM"という名称の領域として自動的に定義されています。特にリンクディレクティブファイル等で特に指定しない場合、内蔵 RAM 領域については全域が"RAM"という名称の領域として自動的に定義され、特に指定しない限り、スタック等がこの領域に配置[注]される事になりますが、この場合、フラッシュ・ライブラリの使用禁止領域(セルフ RAM、FFE20H-FFEFFH の領域)にスタックやデータバッファ等が配置されてしまい、プログラムが正常に動作できなくなることがあります。

添付されているサンプル用リンクディレクティブファイルでは、解決策の一つとして"RAM"名の領域をフラッシュ・ライブラリの使用禁止領域を含めないように再定義し、スタック等が使用禁止領域に配置されないようにしています。

```
MEMORY RAM        :(0FF080H, 000DA0H)
```

上記は、"RAM"名の領域を、FF080H から DA0H バイトのサイズの領域(FF080H-FFE1FH)[注]に再定義し、フラッシュ・ライブラリの使用禁止領域が"RAM"名の領域に含まれないようにしています。

ただし、この設定だけでは「FFE20H-FFEFFH」の領域が他の用途でも使用できなくなるため、別途以下の定義を追加する必要があります。この領域の名称に関しては特に制限等はありません。

```
MEMORY SADDR_RAM:(0FFE20H, 0000E0H)
```

また、セルフ RAM 領域がある場合は、その範囲を以下のように"SELFRAM"名の領域として定義する事により、この領域に対し、自動的に変数等が配置されないように制限する事が可能です。

```
MEMORY SELFRAM   :(0FEF00H, 000180H)
```

以下に、RL78/G13(RAM 4KB/ROM 64KB 製品)の場合の設定例を記載します。

```
; ------------------------------------------------------------
; Define new memory entry for Self-RAM
; ------------------------------------------------------------
MEMORY SELFRAM    : ( 0FEF00H, 000180H )   <-- セルフ RAM 領域の定義

; ------------------------------------------------------------
; Redefined default data segment RAM
; ------------------------------------------------------------
MEMORY RAM        : ( 0FF080H, 000DA0H )   <-- 標準的に使用する RAM 領域の定義

; ------------------------------------------------------------
; Define new memory entry for saddr area
; ------------------------------------------------------------
MEMORY RAM_SADDR : ( 0FFE20H, 0000E0H )    <-- FFE20H-FFEFFH 領域の定義
```

注. CA78K0Rのリンカでは、配置先が指定されないデータ(セグメント・タイプDSEGおよびBSEG)はそのデータの再配置属性に従い、内蔵RAM領域に配置されます。そのため、状況によっては特定のデータが"RAM"名の領域に配置されない場合があります。

各データの定義や配置方法等の詳細についてはCS+のユーザーズマニュアルを参照してください。

また、ビルド時に生成されるマップファイル(*.map)を必ず参照し、各データの配置状態を確認してください。

8.3.2 CC-RLコンパイラ使用時

(1) インクルード・パスの追加

CS+、e2 studio は、初期状態ではインクルード･パスは設定されていません。EEPROM エミュレーション・ライブラリで使用するヘッダファイルのインクルード･パスを追加する必要があります。EEPROM エミュレーション・ライブラリで使用するヘッダファイルは、"eel.h", "eel_types.h", "eel_user_types.h", "eel_descriptor.h", "fdl.h", "fdl_types.h", "fdl_descriptor.h"、および CS+、または e2 studio が自動生成する"iodefine.h" です。

- CS+では、[コンパイル・オプション] –[プリプロセス] - [追加のインクルード・パス]で、各ファイルが存在するインクルード･パスを追加してください。
- e2 studio では"プロパティ"ウィンドウで、"C/C++ビルド" [設定] ‐ "Compiler" [ソース] で表示された画面の"インクルード・ファイルを検索するフォルダ(-I)"の欄に、各ファイルが存在するインクルード・パスを追加してください。

(2) セクションの定義

ROM 領域、RAM 領域で使用する各セクションを定義する必要があります。

- CS+上でセクション項目の設定は、CS+の画面の[リンク・オプション] – [セクション]で指定することができます。[セクションを自動的に配置する]を"いいえ"に設定し、[セクションの開始アドレス]の設定画面を開き、ROM 領域に EEPROM エミュレーション・ライブラリのセクション（ここでは、本サンプル動作に必要なセクション、FDL_CODE, FDL_CNST_f, EEL_CODE, EEL_CNST_f）を追加します（図 8-3）。また、RAM 領域に EEPROM エミュレーション・ライブラリのセクション（ここでは、本サンプル動作に必要なセクション、FDL_SDAT, EEL_SDAT）を追加します（図 8-4）。

  セクションを追加した後に、[セクションを自動的に配置する]を"はい"に戻します。

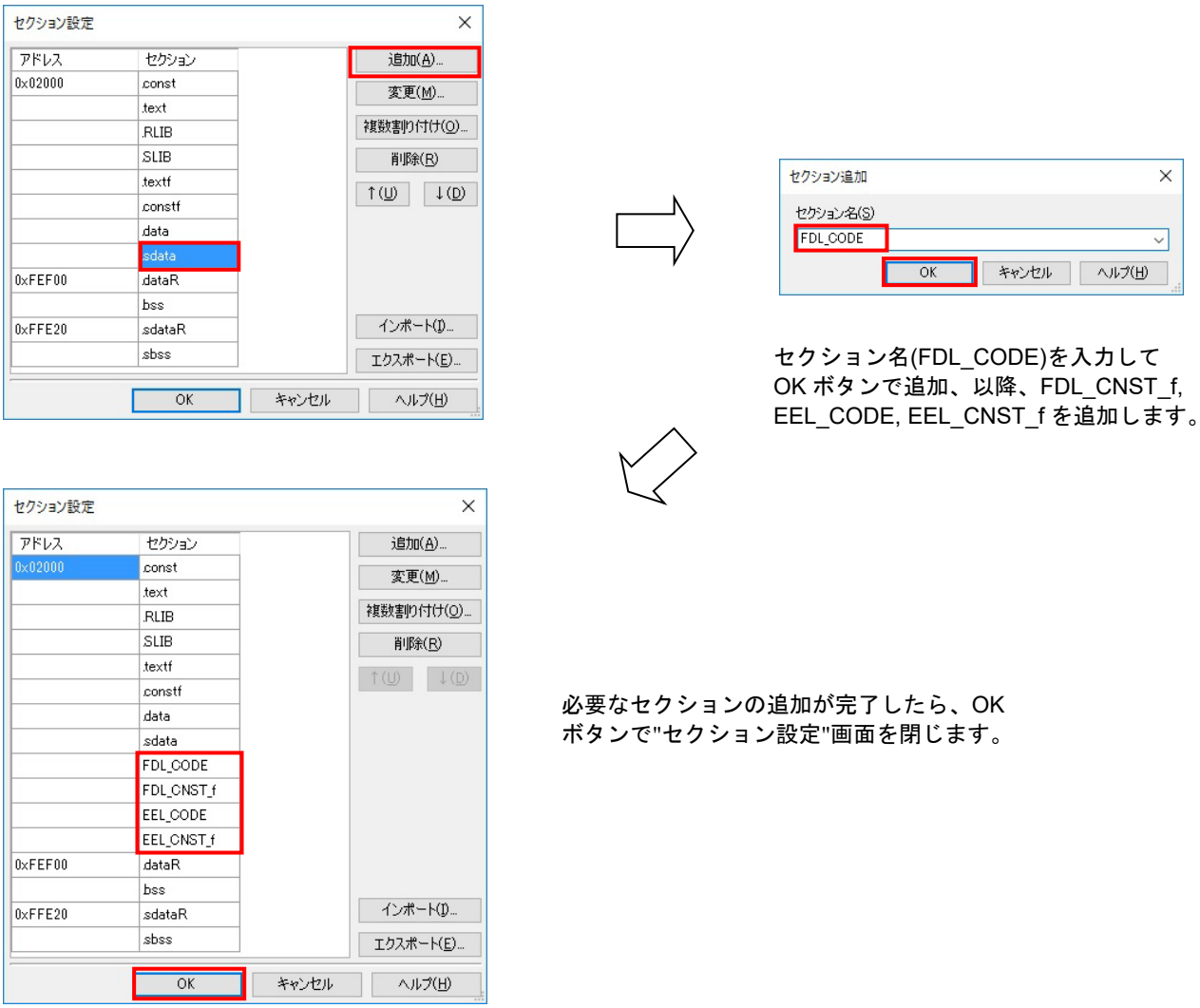

図 8-3 CS+使用時の EEPROM エミュレーション・ライブラリのセクション設定例(ROM 領域)

図 8-4 CS+使用時の EEPROM エミュレーション・ライブラリのセクション設定例(RAM 領域)

- e2 studio でのセクション項目の設定は、"プロパティ"ウィンドウで設定します。"C/C++ビルド" [設定] - "Linker" [セクション] で表示した画面でセクション項目を設定します。[デバイス・ファイルの情報からセクションを自動的に配置する(-auto_section_layout)]のチェックを一度外し、[セクション(-start)]の最も右の" ... "ボタンで、"セクションビューアー"画面を開き、ROM 領域に EEPROM エミュレーション・ライブラリのセクション(ここでは、本サンプル動作に必要なセクション、FDL_CODE, FDL_CNST_f, EEL_CODE, EEL_CNST_f)を追加します(図 8-5)。また、RAM 領域に EEPROM エミュレーション・ライブラリのセクション(ここでは、本サンプル動作に必要なセクション、FDL_SDAT, EEL_SDAT) を追加します(図 8-6)。
セクションを追加した後に、[デバイス・ファイルの情報からセクションを自動的に配置する (-auto_section_layout)] にチェックをいれます。

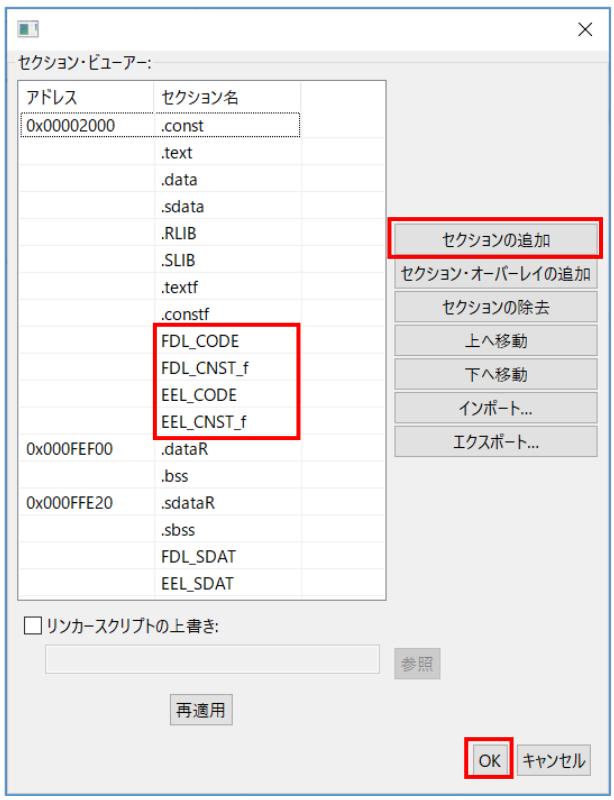

図 8-5 e2 studio 使用時の EEPROM エミュレーション・ライブラリのセクション設定例(ROM 領域)

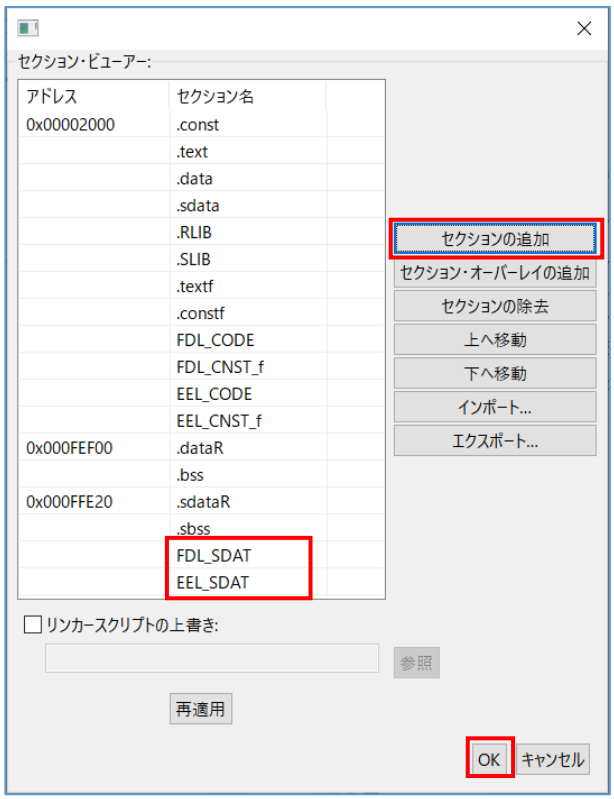

図 8-6 e^2 studio 使用時の EEPROM エミュレーション・ライブラリのセクション設定例(RAM 領域)

(3) セルフ RAM 領域の確保

セクション設定の初期状態では、ユーザ RAM 領域が内蔵 RAM 領域の先頭（サンプルプログラム対象の R5F100LEA では 0xFEF00）から確保されています。しかし、R5F100LEA では、EEPROM エミュレーション・ライブラリがセルフ RAM 領域として 0xFEF00 - 0xFF07F を使用する為、この領域を避けてユーザ RAM 領域を設定する必要があります。ここでは、アドレス 0xFEF00 から設定されているユーザデータの先頭アドレスを 0xFF080 に変更する例を示します（図 8-7，図 8-8）。

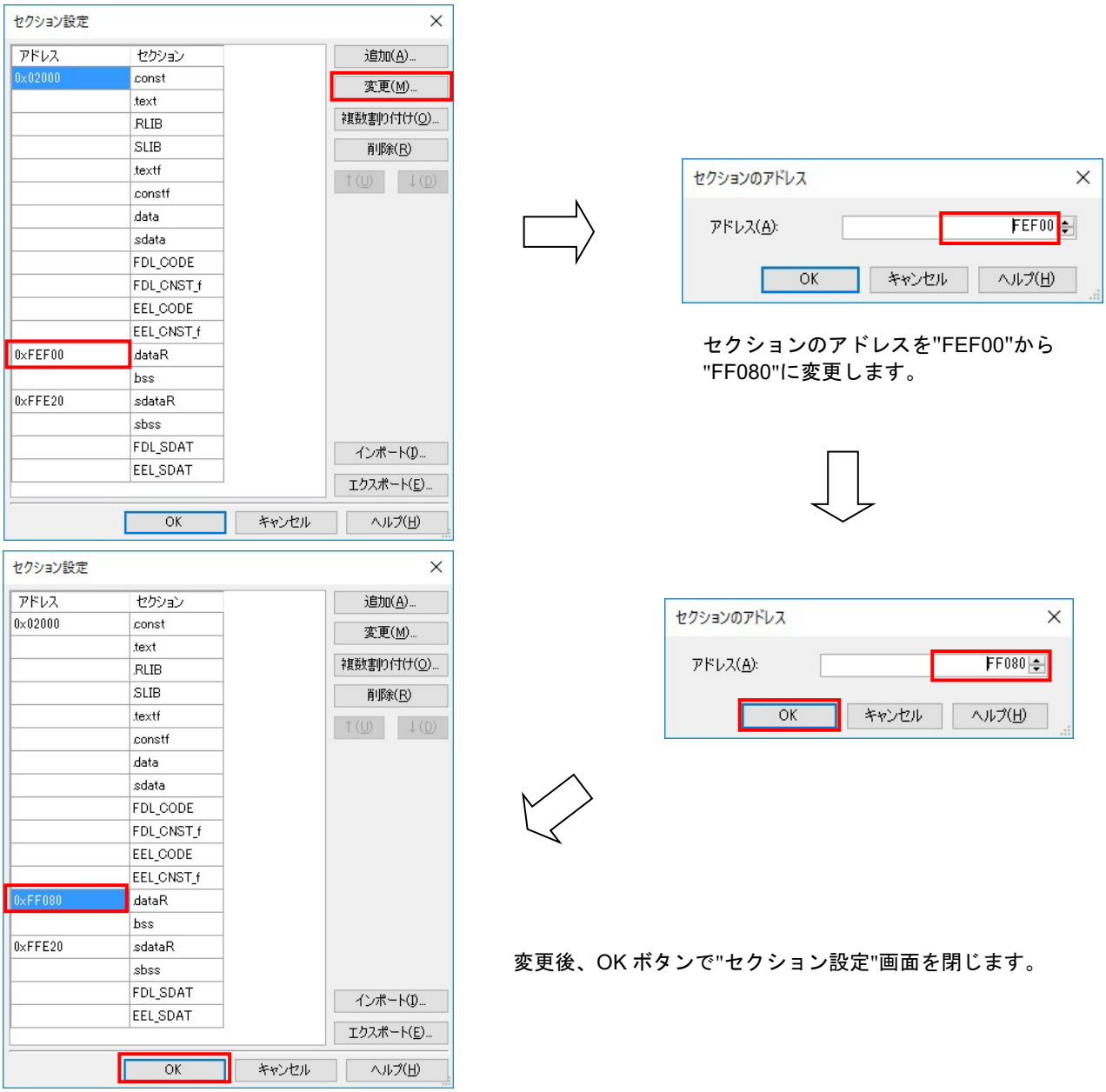

図 8-7 CS+使用時のユーザ RAM 領域範囲の変更例（RAM 領域）

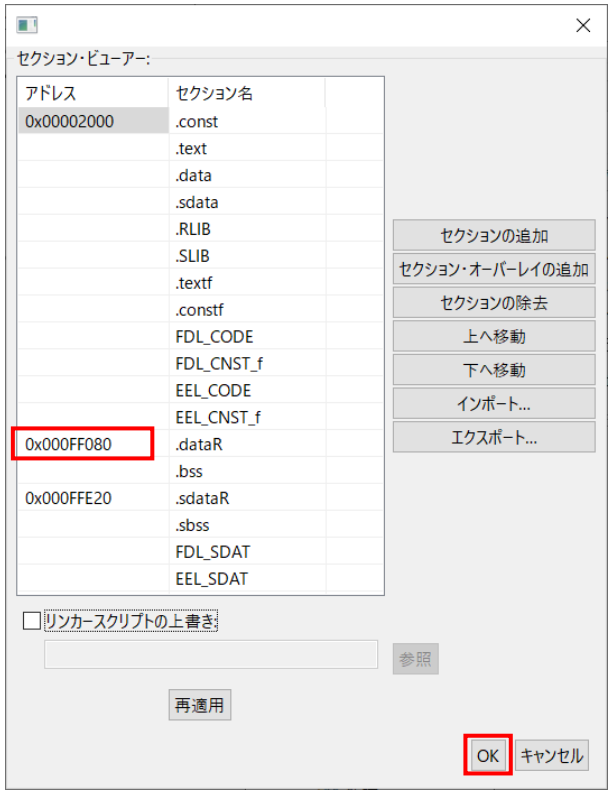

図 8-8 e2 studio 使用時のユーザ RAM 領域範囲の変更例（RAM 領域）

注意）セクション設定で、セクションを自動的に配置する設定を解除しユーザ RAM 領域の変更をした後に、再度セクションを自動的に配置した場合、ユーザが定義したセクションも含め、再自動配置されます。セクションの自動配置では、ユーザが定義していない領域に対してもセクションを配置する可能性があり、意図していなかった領域にデータが配置される可能性があります。必ず、EEPROM エミュレーション・ライブラリで使用するソフトウェアリソース（特に RAM 上データ）が、配置可能領域に配置されていることをマップファイル(*.map)等で確認してください。

8.3.3 LLVMコンパイラ使用時

(1) インクルード・パスの追加

e2 studio は、初期状態ではインクルード･パスが設定されていません。EEPROM エミュレーション・ライブラリで使用するヘッダファイルのインクルード･パスを追加する必要があります。EEPROM エミュレーション・ライブラリで使用するヘッダファイルは、"eel.h", "eel_types.h", "eel_user_types.h", "eel_descriptor.h", "fdl.h", "fdl_types.h", "fdl_descriptor.h"、および e2 studio が自動生成する"iodefine.h", "iodefine_ext.h"です。

e2 studio では、"プロパティ"ウィンドウで、"C/C++ビルド" [設定] － "Compiler" [includes] で表示された画面の "Include file directories (-I)"の欄に、各ファイルが存在するインクルード・パスを追加してください。

(2) セルフ RAM 領域の確保

LLVM コンパイラでは、ビルドで実行するリンク設定をリンカスクリプトファイル(*.ld)に記述します。

e^2 studio が出力するリンカスクリプトファイル(linker_script.ld)では、内蔵 RAM 領域については全域が"RAM"という名称の領域として定義され、EEPROM エミュレーション・ライブラリで使用するソフトウェア・リソースが必要なデバイスでは、"SELFRAM"という名称の領域として定義されています。

サンプルプログラム用に添付されているリンカスクリプトファイル(r_eel_sample_c.ld)では、"RAM"名の領域を EEPROM エミュレーション・ライブラリの使用禁止領域を含めないように再定義し、SELFRAM（セルフ RAM）領域を定義し確保することでユーザデータやスタック等がセルフ RAM 領域に配置されないようにしています。

注. サンプルプログラムで提供している r_eel_sample_c.ld は、R5F100LE を使用することを前提に用意されています。そのほかのデバイスをご使用の場合は、セルフ RAM リストをご確認いただきデバイスにあわせて修正を行ってください。
リンカスクリプトファイル(*.ld)の記述内容、及び記述方法の詳細については、LLVM のリファレンスマニュアルをご参照ください。

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。お客様の機器・システムの設計において、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には、お客様の責任において行ってください。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、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、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。
5. 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。
6. 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。
7. 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。
8. 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。
9. 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。
10. 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。
11. 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。
12. 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.4.0-1　2017.11)

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。

© 2024 Renesas Electronics Corporation. All rights reserved.