

## RL78/I1C(512KB) Continuous Operation FOTA

### Continuous Operation FOTA Example Project

## 目次

1. はじめに .....	4
1.1. 前提条件および注意事項.....	4
1.2. 必要となる環境.....	5
1.3. 用語の定義.....	5
2. Continuous Operation FOTA サンプルプロジェクトの特徴.....	6
2.1. パスコード正誤判定機能.....	6
2.2. LCD 表示機能 .....	6
2.3. FOTA 機能 .....	7
3. サンプルプロジェクトの動作.....	8
3.1. プロジェクト内容 .....	8
3.2. MCU のプログラミング .....	8
3.3. 実行手順 .....	10
3.3.1. イメージファイル転送 .....	14
3.3.2. バンクスワップ機能によるアプリケーションの切り替え .....	17
4. サンプルプロジェクトの設定.....	19
4.1. プロジェクト構成 .....	19
4.2. メモリ配置.....	20
4.2.1. ユーザアプリケーションプロジェクトのメモリ配置 .....	21
4.2.2. ミドルウェアサブプロジェクトのメモリ配置.....	25
4.2.3. ブートローダサブプロジェクトのメモリ配置.....	28
4.3. リンクオプションの補足説明 (e <sup>2</sup> studio).....	31
4.4. 外部定義シンボルファイルの利用について .....	32
4.5. ROM から RAM へのマッピング設定 .....	33
4.5.1. LCD 更新処理ルーチン .....	33
4.5.2. ユーザアプリケーションプロジェクトのマッピング .....	35
4.5.3. ミドルウェアサブプロジェクトのマッピング .....	36
4.5.4. ブートローダサブプロジェクトのマッピング .....	37

4.6.	ブランチテーブル .....	38
4.7.	Continuous Operation FOTA サンプルプロジェクトの API 関数 .....	40
4.7.1.	API 関数一覧.....	40
4.7.2.	Continuous Operation FOTA シーケンス .....	41
4.8.	サンプルプログラムのビルド.....	42
Revision History .....		45

## 図表目次

図 1-1 :	ハードウェア構成 .....	5
図 2-1 :	パスコード認証イメージ.....	6
図 2-2 :	LCD 表示機能 .....	6
図 2-3 :	バンク切り替えを含む内部動作.....	7
図 3-1 :	RL78/I1C(512KB) Fast Prototyping Board の外形.....	8
図 3-2 :	Renesas Flash Programmer .....	9
図 3-3 :	RL78/I1C (512 KB) Fast Prototyping Board と周辺機器の接続 .....	10
図 3-4 :	Windows デバイスマネージャでの確認方法 .....	11
図 3-5 :	Tera Term での Serial Port の設定 .....	11
図 3-6 :	Tera Term での Serial Port 設定.....	12
図 3-7 :	Tera Term での Start-up メッセージの表示 .....	12
図 3-8 :	コマンド一覧.....	13
図 3-9 :	イメージファイル転送の開始 .....	14
図 3-10 :	XMODEM を選択 .....	14
図 3-11 :	XMODEM ダイアログ .....	15
図 3-12 :	イメージ転送終了メッセージ .....	15
図 3-13 :	“hash” コマンド .....	15
図 3-14 :	“binfo” コマンド .....	16
図 3-15 :	Ver. 1.00 での “OK”表示.....	17
図 3-16 :	“bswap”コマンド.....	17
図 3-17 :	“bswap”コマンドによるバージョン表示変化 .....	18
図 3-18 :	パスコード認証 “NG”表示 .....	18
図 3-19 :	パスコード認証 “OK”表示 .....	18
図 4-1 :	プロジェクト構成 .....	19
図 4-2 :	メモリマッピング .....	20
図 4-3 :	ユーザアプリケーション メモリ種別のアドレス範囲 (CS+).....	21
図 4-4 :	ユーザアプリケーション メモリ種別のアドレス範囲 (e2studio).....	22
図 4-5 :	ユーザアプリケーション デバッグ・モニタ領域 (CS+).....	23
図 4-6 :	ユーザアプリケーション デバッグ・モニタ領域 (e2studio).....	23
図 4-7 :	ユーザアプリケーション セクション設定 (CS+).....	24
図 4-8 :	ユーザアプリケーション セクション設定 (e2studio) .....	24
図 4-9 :	ミドルウェア メモリ種別のアドレス範囲 (CS+).....	25
図 4-10 :	ミドルウェア メモリ種別のアドレス範囲 (e2studio) .....	25

図 4-11 : ミドルウェア セクション設定と外部定義シンボル (CS+) .....	26
図 4-12 : ミドルウェア セクション設定と外部定義シンボル (e2studio) .....	27
図 4-13 : ブートローダ メモリ種別のアドレス範囲 (CS+).....	28
図 4-14 : ブートローダ メモリ種別のアドレス範囲 (e2studio) .....	28
図 4-15 : ブートローダ セクション設定と外部定義シンボル (CS+) .....	29
図 4-16 : ブートローダ セクション設定と外部定義シンボル (e2studio) .....	30
図 4-17 : AUTO_SECTION_LAYOUT ユーザオプション (e2studio).....	31
図 4-18 : 外部定義シンボルファイル (CS+).....	32
図 4-19 : LCD 更新処理 .....	33
図 4-20 : 画面更新 Continuous Operation .....	33
図 4-21 : ROM から RAM へのコード展開 .....	34
図 4-22 : ユーザアプリケーション ROM から RAM へマップするセクション (CS+) .....	35
図 4-23 : ユーザアプリケーション ROM から RAM へマップするセクション (e2studio)..	35
図 4-24 : ミドルウェア ROM から RAM へマップするセクション (CS+).....	36
図 4-25 : ミドルウェア ROM から RAM へマップするセクション (e2studio) .....	36
図 4-26 : ブートローダ ROM から RAM へマップするセクション (CS+).....	37
図 4-27 : ブートローダ ROM から RAM へマップするセクション (e2studio) .....	37
図 4-28 : ベクタテーブルの分離 (CS+).....	38
図 4-29 : ベクタテーブルの分離 (e2studio) .....	38
図 4-30 : ブランチテーブルフロー.....	39
図 4-31 : Continuous Operation FOTA の API 使用例 .....	41
図 4-32 : パスコードおよびバージョンの設定 (platform.h) .....	42
図 4-33 : バージョン設定 (CS+) .....	43
図 4-34 : バージョン設定 (e2studio) .....	43

## 表目次

表 1-1 : 用語の定義.....	5
表 4-1 : API 関数.....	40

## 1. はじめに

本ドキュメントでは、RL78/I1C(512KB)の Continuous Operation FOTA を用いたサンプルアプリケーション、および、そのソフトウェア設計について説明します。

### Continuous Operation FOTA とは？

RL78/I1C(512KB)は、主に電力メータの計量 (Metrology) 用のマイコンとして使用されています。また、電力計量機能を停止することなく、ファームウェア更新を行う機能を持っており、この機能を Continuous Metrology FOTA と呼んでいます。

本サンプルソフトウェアは、Continuous Metrology FOTA について、計量以外のアプリケーションへの応用例を示しています。そのため、本ドキュメント内では、Continuous Metrology FOTA を Continuous Operation FOTA に置き換えて記載しています。

**Continuous Metrology FOTA : Metrology 機能を停止することなくファームウェアを更新**

**Continuous Operation FOTA : マイコンのある機能を停止することなくファームウェアを更新**

### 1.1. 前提条件および注意事項

- (1) 本ドキュメントは CS+や e<sup>2</sup>studio などの IDE (統合開発環境) および Tera Term などを含めたターミナルソフトウェアの使用経験があるユーザを対象としています。
- (2) ユーザは、このドキュメントで説明されているようなサンプルプロジェクトの作成および変更のために必要とされるマイクロコントローラ、組み込みシステム、および CS+のコードジェネレータに関する基本的な知識を持っているものとします。
- (3) 本ドキュメント中で使用される画像やスクリーンショットは参考用のものです。実際の画面の表示内容は、プログラムや開発ツールのバージョンによって異なる場合があります。

## 1.2. 必要となる環境

ハードウェア構成（図 1-1 参照）：

- RL78/I1C (512KB) Fast Prototyping Board [RTK5RL10N0CPL000BJ]（図中 1）
- PMOD OLEDrgb [Digilent Pmod OLEDrgb (Revision B)]（図中 2 の 96x64 RGB Display）
- PMOD KYPD [Digilent PmodKYPD (Revision B)]（図中 3 の 4x4 キーパッド）
- Micro USB ケーブル（図中 4）
- PC（USB ポート）（図中 5）

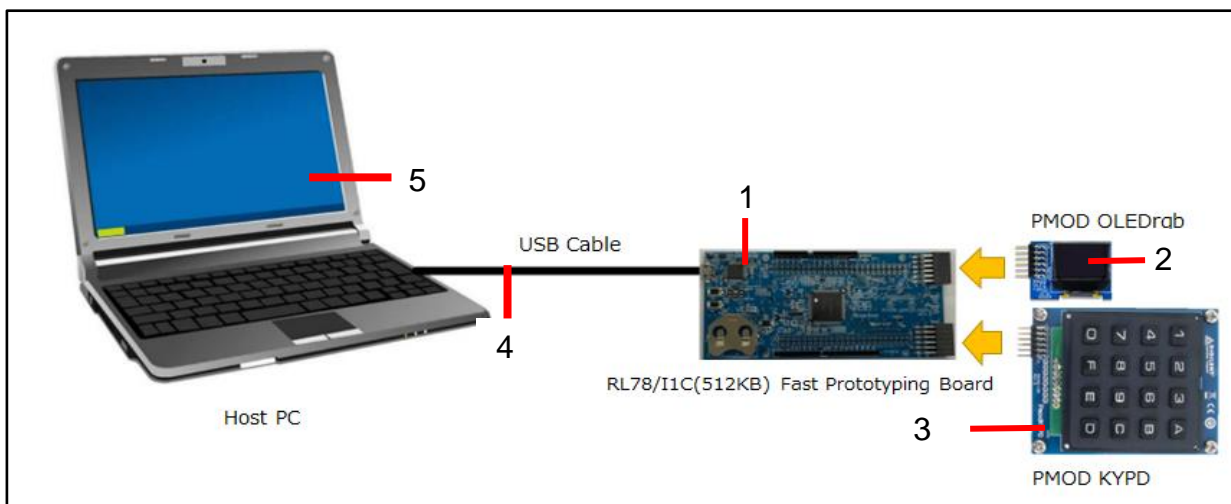


図 1-1：ハードウェア構成

ソフトウェア関連：

- Windows® 10 オペレーティングシステム
- USB シリアルドライバ（Windows 10 標準）
- Tera Term 等のターミナルソフトウェア
- CS+ Ver. 9.06.00 以上 または e²studio 2021-10 以上
- CC-RL コンパイラ V1.10.00 以上
- Renesas Flash Programmer V3.08.03 以上

## 1.3. 用語の定義

表 1-1：用語の定義

FOTA	Firmware update Over-The-Air
バンクスワップ	実行中のバンク 0 (256KB) と新しいファームウェア用のバンク 1 (256KB) の 2 つのバンクを、バンクスワップ用のライブラリを使って切り替えること
セルフ・プログラミング	ファームウェアの更新対象が自身である場合のその操作自体のこと
FSL	フラッシュ・セルフ・プログラミング・ライブラリ

## 2. Continuous Operation FOTA サンプルプロジェクトの特徴

本サンプルプロジェクトでは、ユーザが入力するパスコードの正誤判定機能および LCD 表示機能が動作します。パスコードを変更する場合、認証用ファームウェアの更新を行います。その際、RL78/I1C(512KB) Continuous Operation FOTA により、LCD 表示を継続しながら、ファームウェア更新（パスコードの変更）を行うことができます。

### 2.1. パスコード正誤判定機能

PMOD KYPD（4x4 キーパッド）からパスコードの入力を受け付け、正誤判定を行います。パスコードは、0～9の数字と E を除く A～F の文字の 4 桁で構成され、正しいパスコードは、1 組のみです。

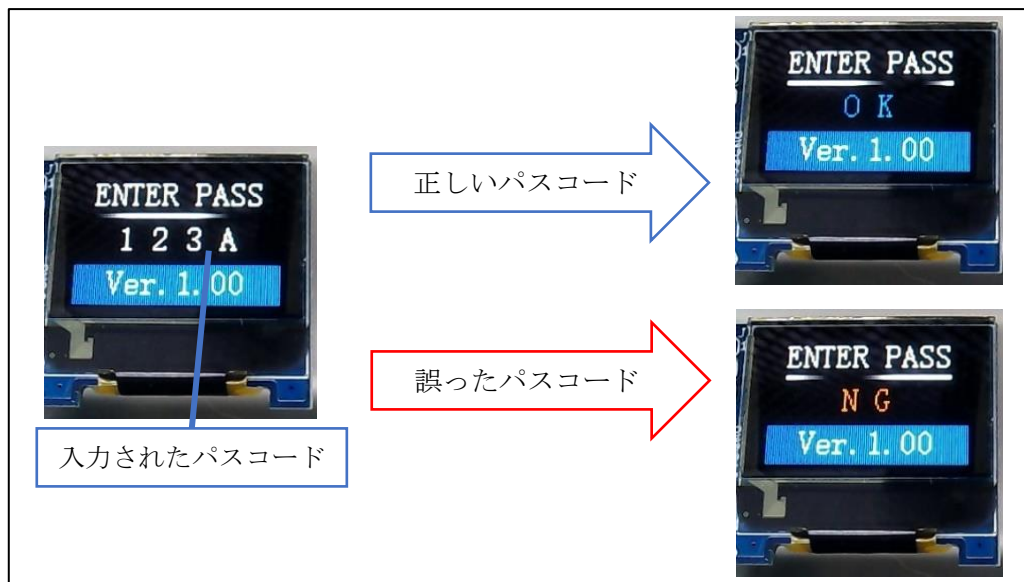


図 2-1 : パスコード認証イメージ

### 2.2. LCD 表示機能

LCD（96x64 RGB Display）に表示しているインジケータ・バーは、ファームウェアアップデート処理を行う期間も含め、常時動作します。LCD では、他に、“ENTER PASS”のメッセージ、入力したパスコード、正誤判定結果および現在のファームウェアバージョン表示を行います。

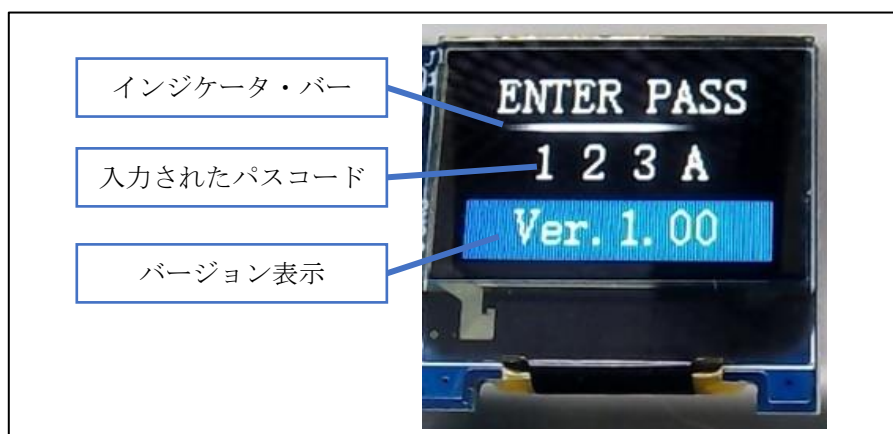


図 2-2 : LCD 表示機能

## 2.3. FOTA 機能

本サンプルプロジェクトでは、LCD 表示機能（OLED 制御 F/W）が停止することなく、新パスコード判定アプリケーション C のダウンロード・書き込み、および、アプリケーション（B→C）の切り替えを行います。この Continuous Operation FOTA は、RL78/I1C(512KB)の2つのフラッシュ・メモリバンク、バンクスワップ機能を用い、さらに、RAM 上でプログラムを実行することによって実現されます。

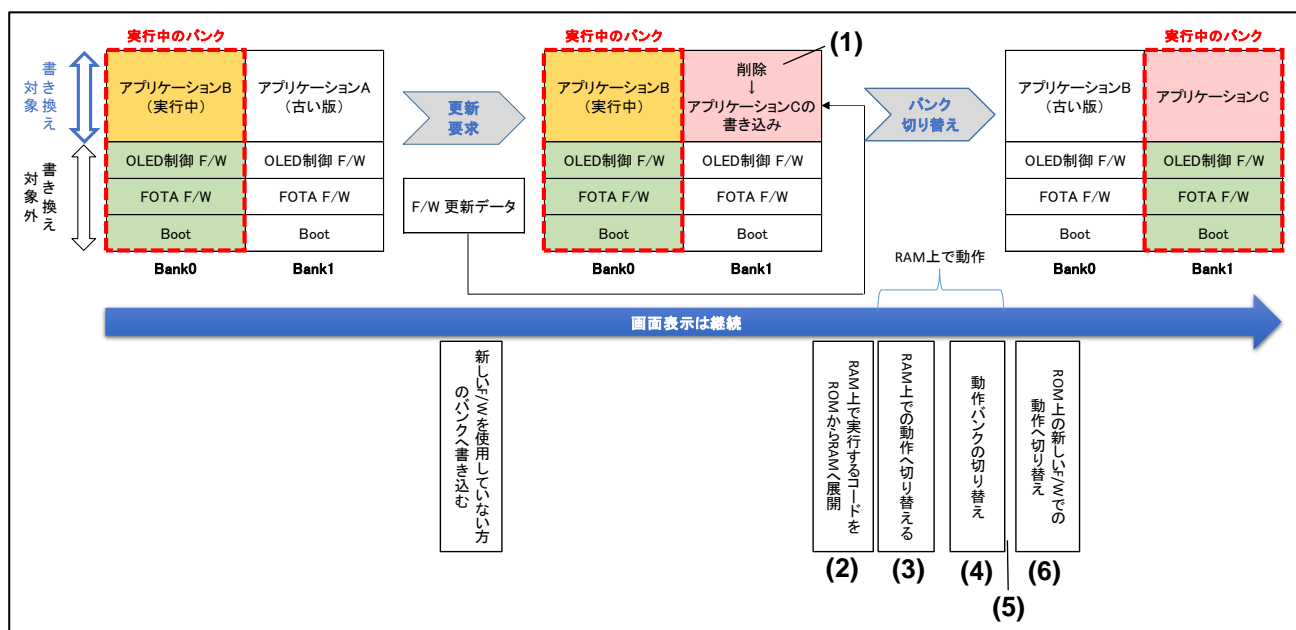


図 2-3: バンク切り替えを含む内部動作

- アプリケーション A : 現在使用されていない古いアプリケーション
- アプリケーション B : 現在動作中のアプリケーション
- アプリケーション C : 書き換える新しいアプリケーション
- OLED 制御 F/W : 画面表示を行うファームウェア
- Boot : 起動制御を行うブートローダ

### <内部動作>

- (1) アプリケーション B の動作中に、古いアプリケーション A のある未使用バンク（Bank1）へ新たなアプリケーション C を書き込む。
- (2) 書き込み後、バンクスワップコマンドの受信により、RAM 上で実行するコード（LCD のインジケータ・バーの更新など）を ROM から RAM へ展開する。
- (3) 割り込みベクタテーブルを RAM 上のものへ切り替え、RAM からの動作へ切り替える。
- (4) バンクスワップが実行される。その間も RAM 上のプログラムは動作を続ける。
- (5) バンクスワップ完了後、割り込みベクタテーブルを ROM 上のものに戻す。
- (6) 更新後のアプリケーションの動作を開始する。

### 3. サンプルプロジェクトの動作

この章では、Continuous Operation FOTA サンプル実行についての操作手順を示します。

#### 3.1. プロジェクト内容

サンプルプロジェクトは以下の3つのサブフォルダを含み、サンプルの実行には、(1)と(2)のファイルを使用します。

- (1) RFP RL78I1C Production フォルダ : Renesas Flash Programmer のプロジェクトファイル [i1c\_512k\_production.rpj]と、“Ver.1.00”の MOT ファイル ([rl78i1c\_production.mot]) が入っています。MOT ファイルの作成の方法については本書の 4.8 を参照してください。
- (2) New Application File フォルダ : [rl78i1c\_v100.mot] と [rl78i1c\_v200.mot]が入っており、それぞれ 4.8 で生成される“Ver.1.00”と“Ver.2.00”の MOT ファイルとなります。
- (3) Source フォルダ : 設定ファイル、ソースコードなどソフトウェア一式が入っています。

#### 3.2. MCU のプログラミング

MCU へのプログラミングを、下記の手順で行います。

- (1) RL78/I1C (512KB) Fast Prototyping Board 上のディップスイッチ(SW3)を“Debug”に設定し、MicroUSB ケーブルを Micro USB コネクタに接続します。

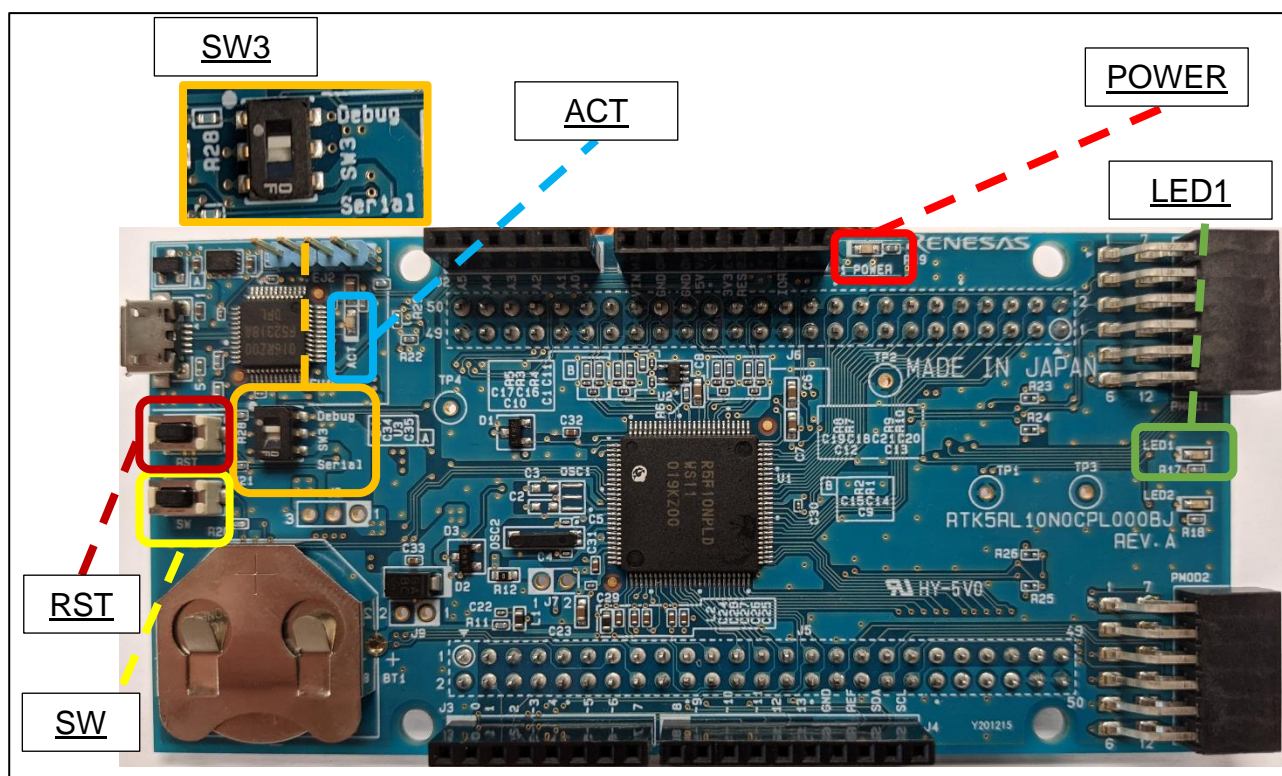


図 3-1 : RL78/I1C(512KB) Fast Prototyping Board の外形



- (2) MicroUSB ケーブルのもう一方の端をホスト PC に接続します。(LED3 (POWER) 点灯)
- (3) Renesas Flash Programmer で、3.1 (1)の[i1c\_512k\_production.rpj]を開きます。
- (4) 3.1 (1)の[rl78i1c\_production.mot]を指定、Start ボタンをクリックして、MCU にダウンロードします。

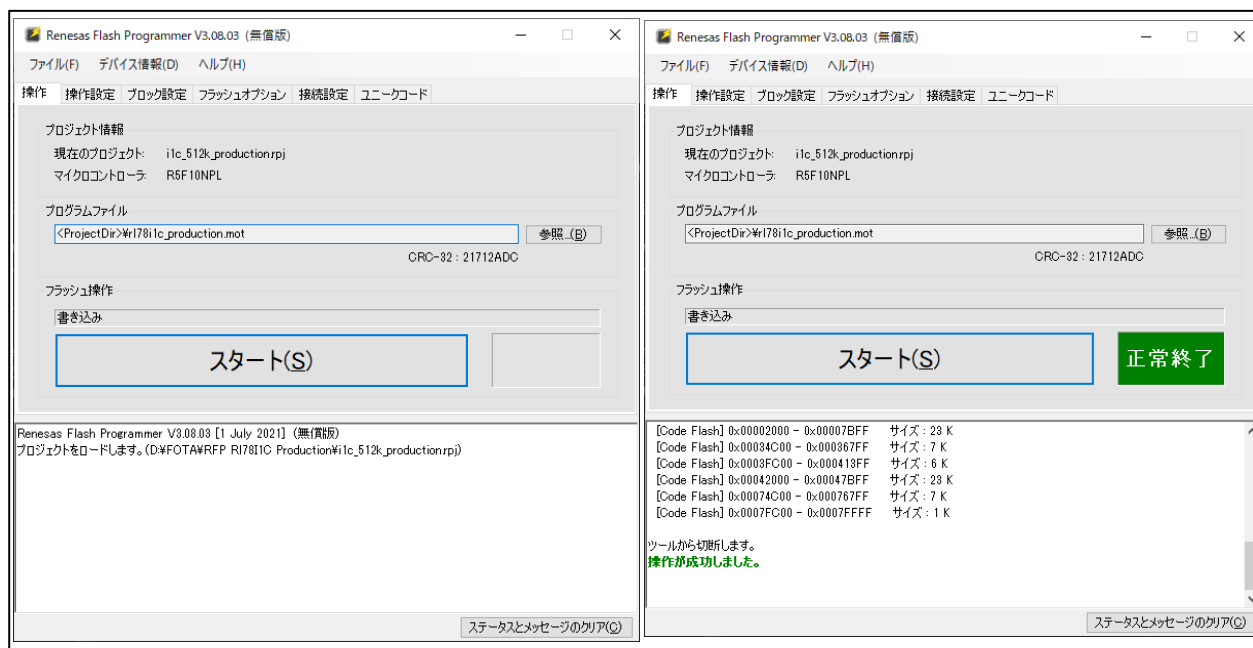


図 3-2 : Renesas Flash Programmer

### 3.3. 実行手順

サンプルプロジェクトを動作させる手順を示します。

1. RL78/I1C (512KB) Fast Prototyping Board 上のディップスイッチ(SW3) “Serial” に設定し、Micro USB ケーブルを RL78/I1C (512KB) Fast Prototyping Board の Micro USB コネクタに接続します。
2. Pmod OLEDrgb と Pmod KYPD を RL78/I1C (512KB) Fast Prototyping Board に接続します。

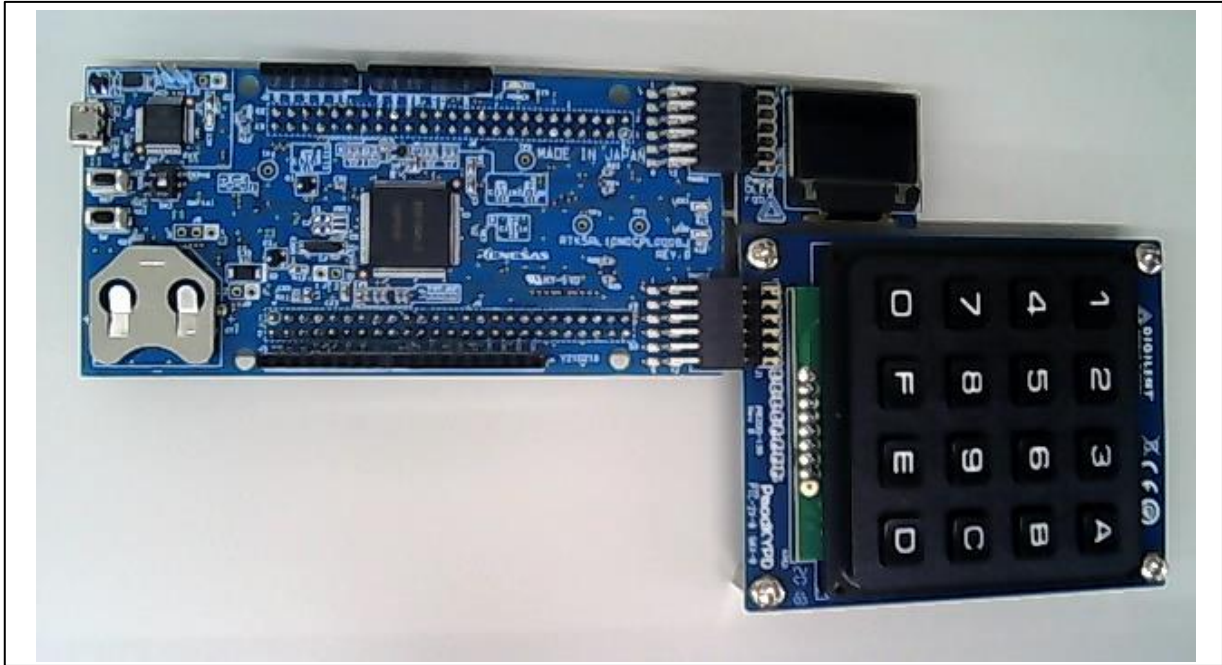


図 3-3 : RL78/I1C (512 KB) Fast Prototyping Board と周辺機器の接続

3. Micro USB ケーブルのもう一方の端をホスト PC に接続します。(LED3 (POWER) 点灯)

4. ホスト PC で、Windows デバイスマネージャを開きます。USB Serial Device (COMxx) にある Ports (Com & LPT) を展開し、次の手順で参照できるように COM ポート番号を書き留めます。

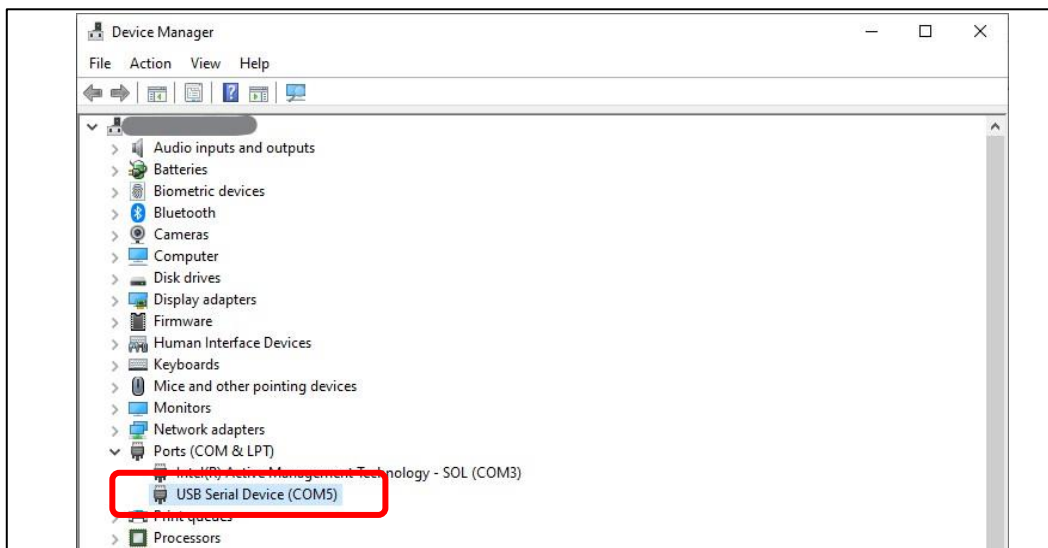


図 3-4 : Windows デバイスマネージャでの確認方法

補足 : RL78/I1C (512KB) Fast Prototyping Board とホスト PC 上のターミナルアプリケーション間の通信には、USB シリアルデバイスドライバが必要です。

5. Tera Term を開き、Serial と COMxx (Serial Device(COMxx)) を選択し、OK をクリックします。

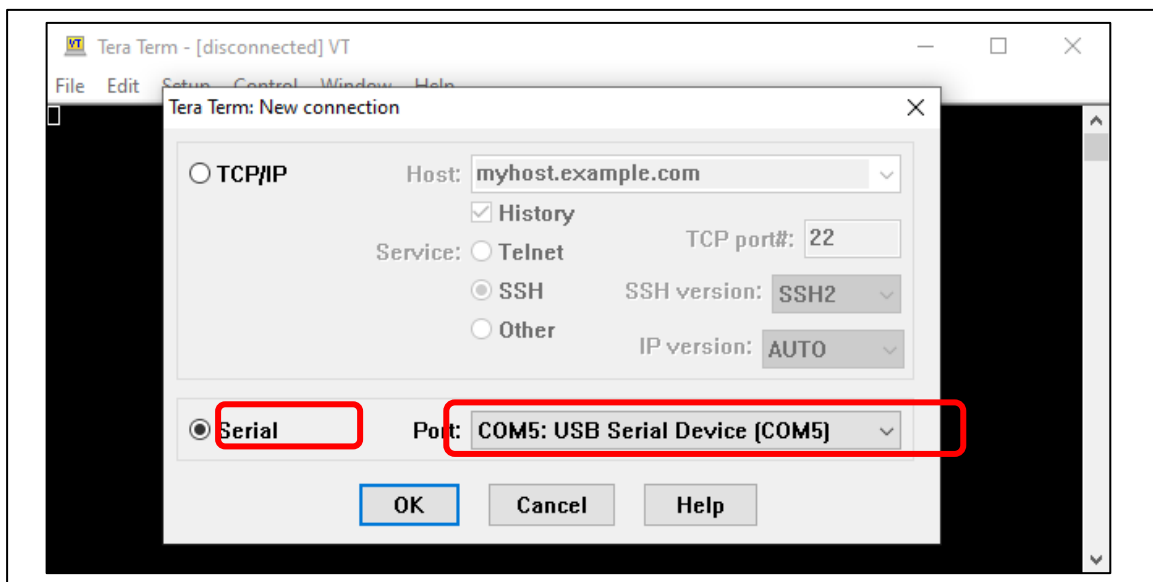


図 3-5 : Tera Term での Serial Port の設定

6. Tera Term の Serial port setup and connection ウィンドウで、Setup and Serial Port...を選択します。図 3-6 のように設定し、“New setting”をクリックします。

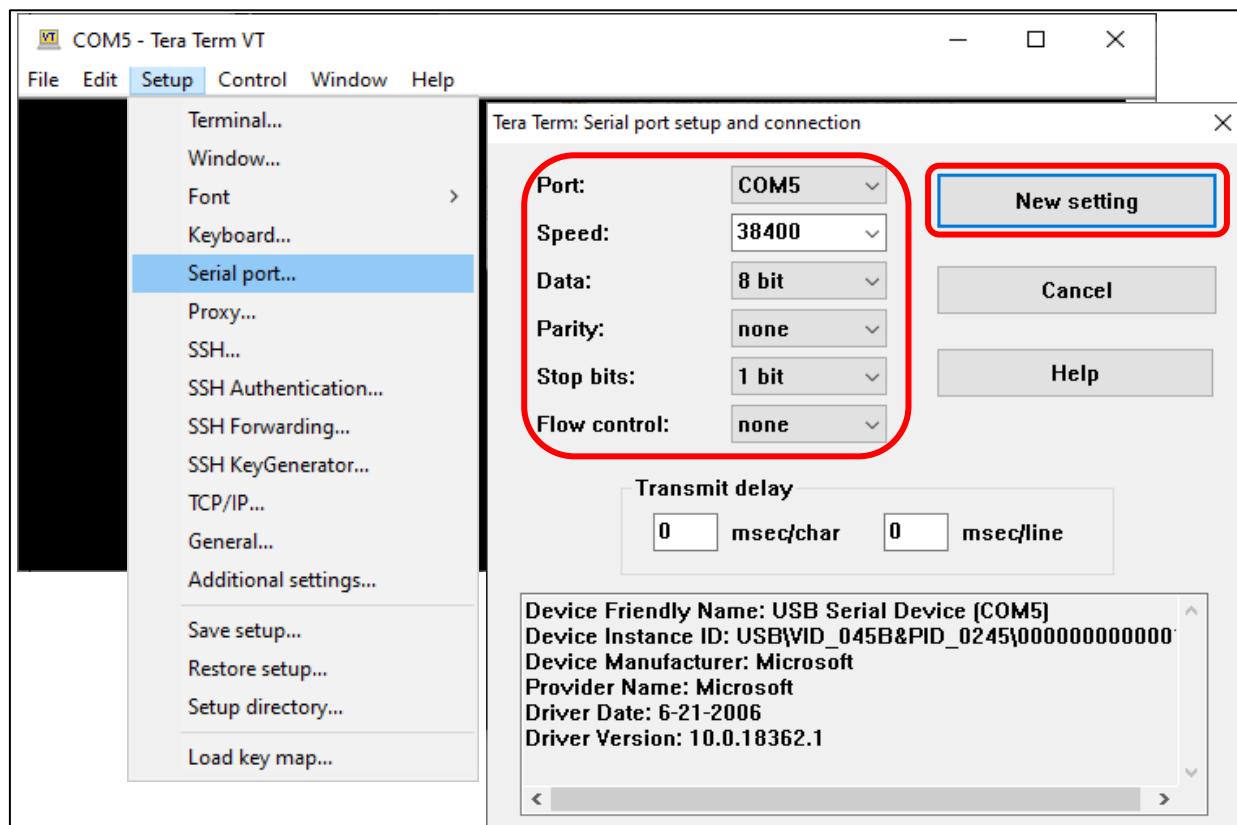


図 3-6 : Tera Term での Serial Port 設定

7. RL78/I1C(512KB) Fast Prototyping Board の RST ボタンを押してください。Tera Term に、図 3-7 のスタートアップメッセージが表示されます。

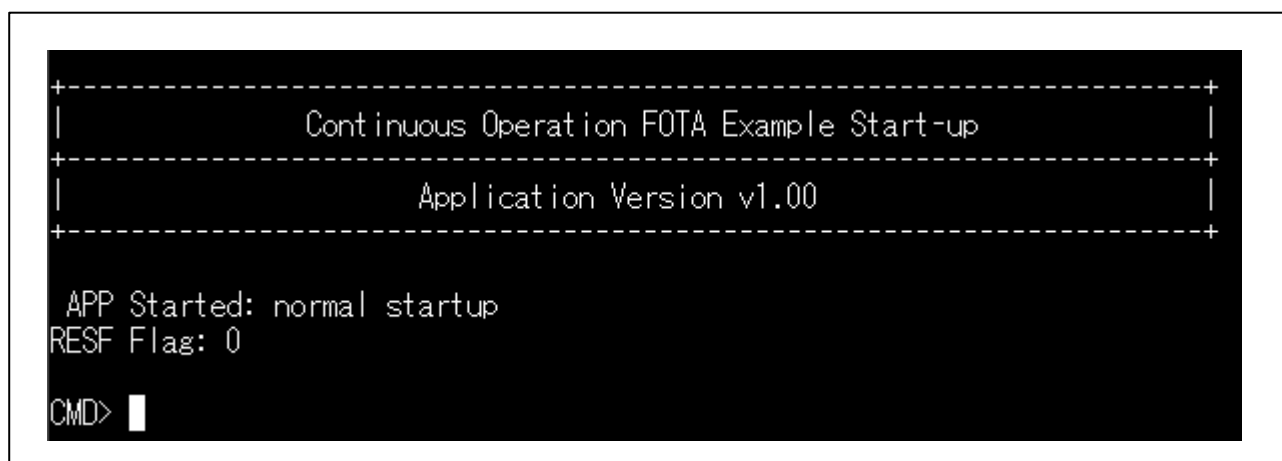


図 3-7 : Tera Term での Start-up メッセージの表示

8. “?” と入力し、Enter キーを押すと、使用可能な コマンド を確認することができます。

```
CMD> ?
-----
Command Name  Description
-----
?             Help
cls          Clear screen
binfo       Get bank status information
bswap       Swap bank
xfer       Transfer image file using XModem Protocol
hash       Hash the secondary bank and compare to the header value
```

図 3-8 : コマンド一覧

9. コマンドを入力し、Enter キーを押すと関数が実行されます。

各コマンドの使い方と動作については、次の項で説明します。

### 3.3.1. イメージファイル転送

Continuous Operation FOTA サンプルプロジェクトのイメージファイル転送について説明します。なお、ファイル転送機能は、Tera Term v4.106 を使用してテストおよび開発しています。

- イメージファイル転送は XMODEM チェックサムプロトコルを使用して実行されます。
- XMODEM CRC などのバリエーションはサポートされていません。

1. Tera Term から “xfer” コマンドを送信し、XMODEM 転送ファイル (rl78i1c\_v200.mot) を選択します。XMODEM プロトコルでは、データの転送は受信側が送信側へ NAK を送信することから始まります。“xfer” コマンドを実行後、受信側であるユーザアプリケーションは、データ転送が開始されるよう、10 秒毎に NAK を送信する状態になります。

```
CMD> xfer
Please start file transfer using XModem protocol.
Transfer will initiate within 10 seconds.
```

図 3-9 : イメージファイル転送の開始

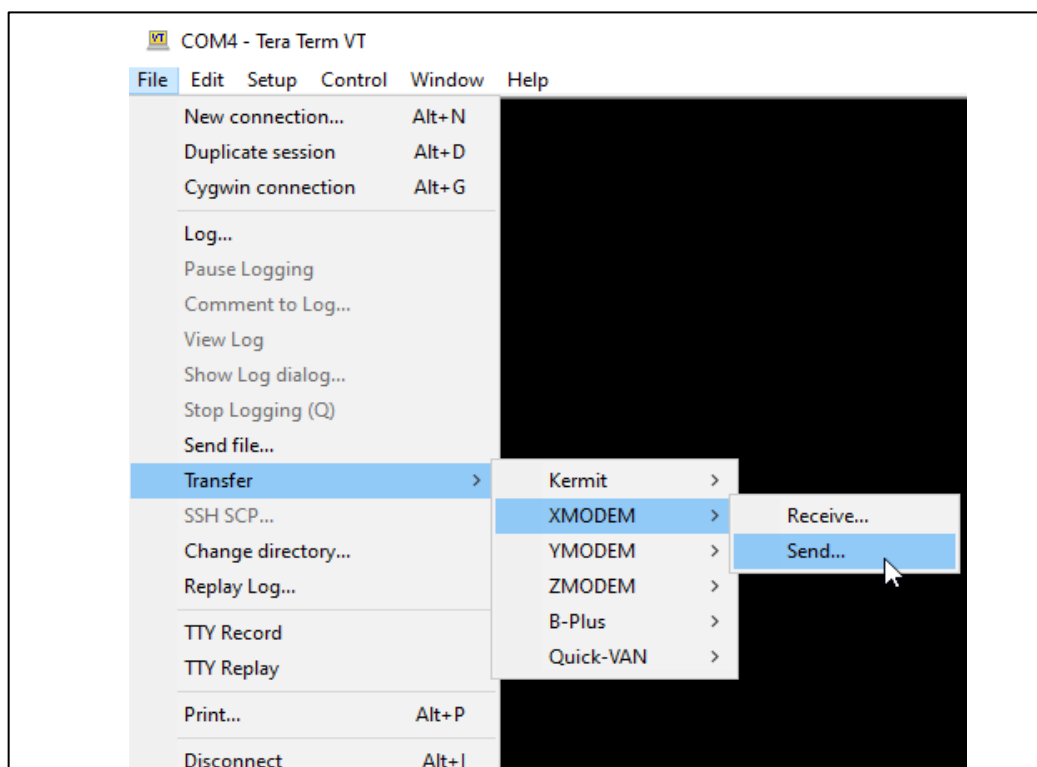


図 3-10 : XMODEM を選択

2. Tera Term 側は、NAK を受信するとデータ転送を開始します。

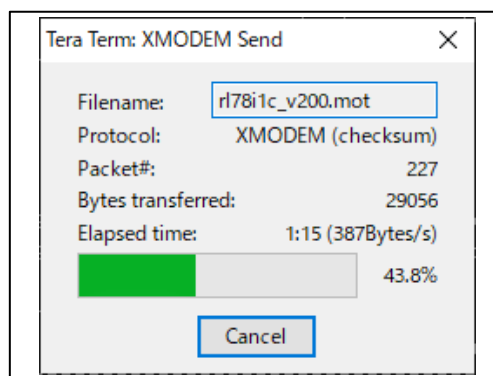


図 3-11 : XMODEM ダイアログ

3. ユーザアプリケーションは、データを受信すると、受信したデータの検証を行い、内容が正しければ書き込みを行います。RL78/I1C(512KB) Fast Prototyping Board 上の LED 1 が、受信したデータを書き込む際に点滅します。それらの処理が完了すると、ユーザアプリケーションは Tera Term 側へ ACK を返信します。Tera Term 側は ACK を受信すると、次のデータを送信します。全てのデータの転送完了までこれを繰り返します。
4. イメージ転送が正常に完了すると、次のようなメッセージが表示されます。

```
CMD> xfer
Please start file transfer using XModem protocol.
Transfer will initiate within 10 seconds.

End of File.
Updating Image Header...
Transfer Complete.
```

図 3-12 : イメージ転送終了メッセージ

5. “hash” コマンドを使用して、転送されたユーザーアプリケーションイメージのハッシュ値が、イメージヘッダー内に格納されている事前計算されたハッシュ値と一致することを検証することができます。

```
CMD> hash
Hash value OK
Calculated program hash: 0x6c30
Embedded program hash: 0x6c30
```

図 3-13 : “hash” コマンド

6. “binfo” コマンドでソフトウェアバージョン等が確認できます。

```
CMD> binfo
Reading device information (embedded in reserved FLASH)
Device name: R5F10NPL
Bankswap support: YES
Code flash size: 0x00080000 (524288) bytes
Code flash size: 0x00000800 (2048) bytes
Reading image header at address 0x01000
Primary Bank Header Info
Platform name: RES_FOTASample_1P2W_I1C512K
Software version: v1.00
User program size: dec: 27487, hex: 0x06b5f (bytes)
User program hash: 0xd84e

Secondary Bank Header Info
Platform name: RES_FOTASample_1P2W_I1C512K
Software version: v2.00
User program size: dec: 27489, hex: 0x06b61 (bytes)
User program hash: 0xaaf0
Reading self-programming flash
FSL Library version: SRL78T01L1000GV221
Boot flag: 0

CMD> █
```

図 3-14 : “binfo” コマンド

7. 転送されたイメージファイル（アップデートファームウェア）は、次の項で説明する“bswap” コマンドを使用して有効にできます。



### 3.3.2. バンクスワップ機能によるアプリケーションの切り替え

“bswap” コマンドを実行すると、ファームウェアの切り替えが行われます。以下の手順で、ファームウェアアップデートを確認することができます。

#### ■“bswap”コマンド実行前

Pmod KYPD で、Ver.1.00 の正しいパスコード（本サンプルでは、1 2 3 A）を入力、“E”を押すと、“OK”が表示されます。

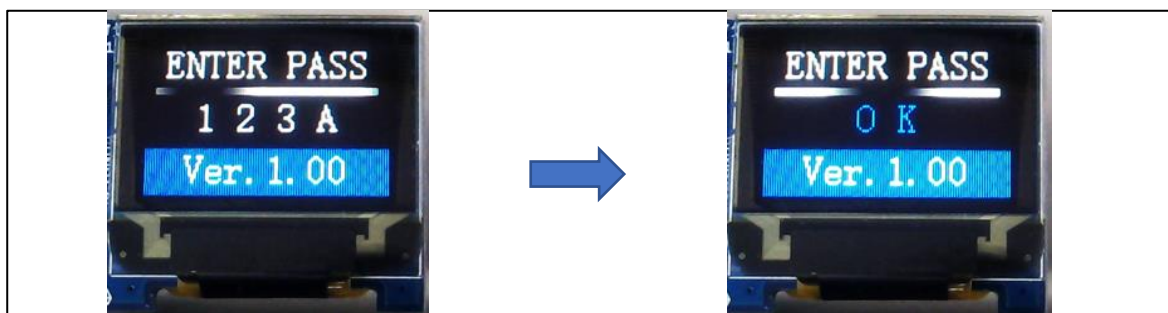


図 3-15 : Ver. 1.00 での “OK”表示

#### ■“bswap”コマンド実行

“bswap” コマンドを実行すると、RL78/I1C(512KB)内のフラッシュバンクの切り替えが行われ、直ちに、新しいファームウェア（Ver.2.00）での動作が開始します。

```
CMD> bswap
Executing bank swap on the fly then branch back to user app entry
-----+-----
|                               |
|   Continuous Operation FOTA Example Start-up   |
|                               |
|   Application Version v2.00   |
|                               |
|-----+-----|
APP Started: bank swapped
CMD> |
```

図 3-16 : “bswap” コマンド

一方、LED表示においても、Ver.2.00に変更されたことを確認することができます。

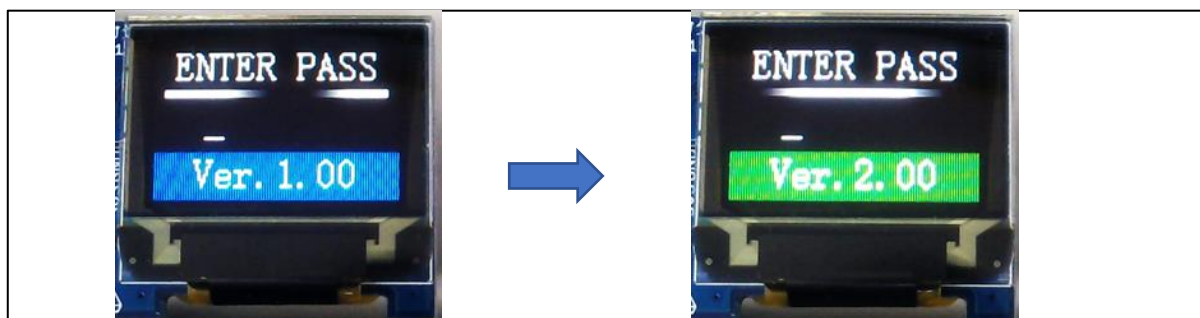


図 3-17 : “bswap”コマンドによるバージョン表示変化

また、本サンプルでは、Ver.1.00 のパスワードを “1 2 3 A”、Ver.2.00 のパスワードを “4 5 6 B” としているため、Ver.2.00 のファームウェアにおけるパスワード認証結果は、それぞれ、図 3-18、図 3-19 のようになります。

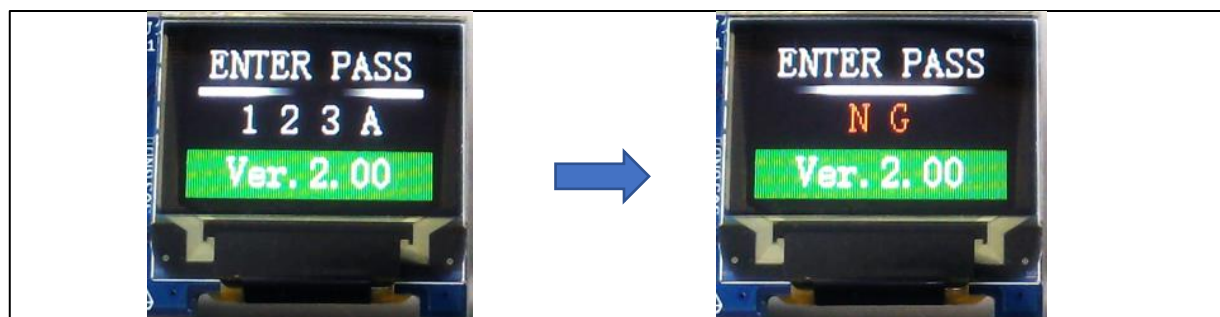


図 3-18 : パスワード認証 “NG”表示

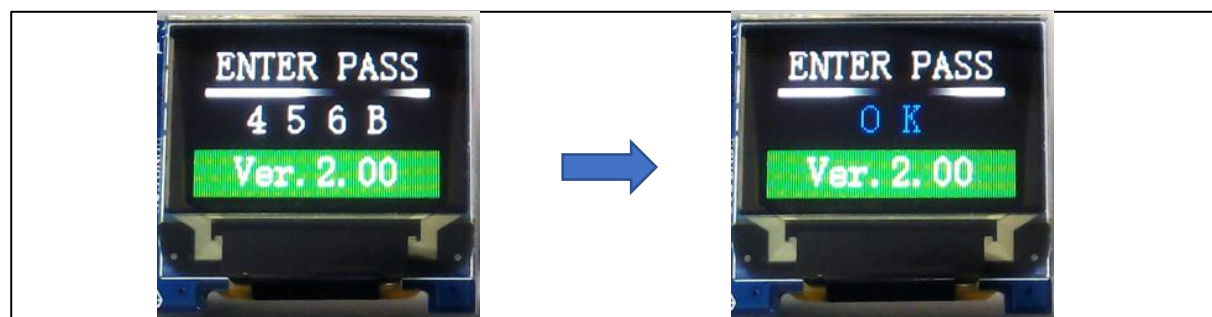


図 3-19 : パスワード認証 “OK”表示

## 4. サンプルプロジェクトの設定

この章では、本サンプルでのプロジェクト設定や動作について説明します。

### 4.1. プロジェクト構成

本サンプルプロジェクトは3つのプロジェクトで構成されています。メインプロジェクトはFOTA機能による更新対象となるユーザアプリケーションプロジェクトです。ミドルウェアサブプロジェクトには、画面の表示機能に関わるプログラムが含まれています。ブートローダサブプロジェクトは、ブートローダ関数とブートローダライブラリを含んでいます。

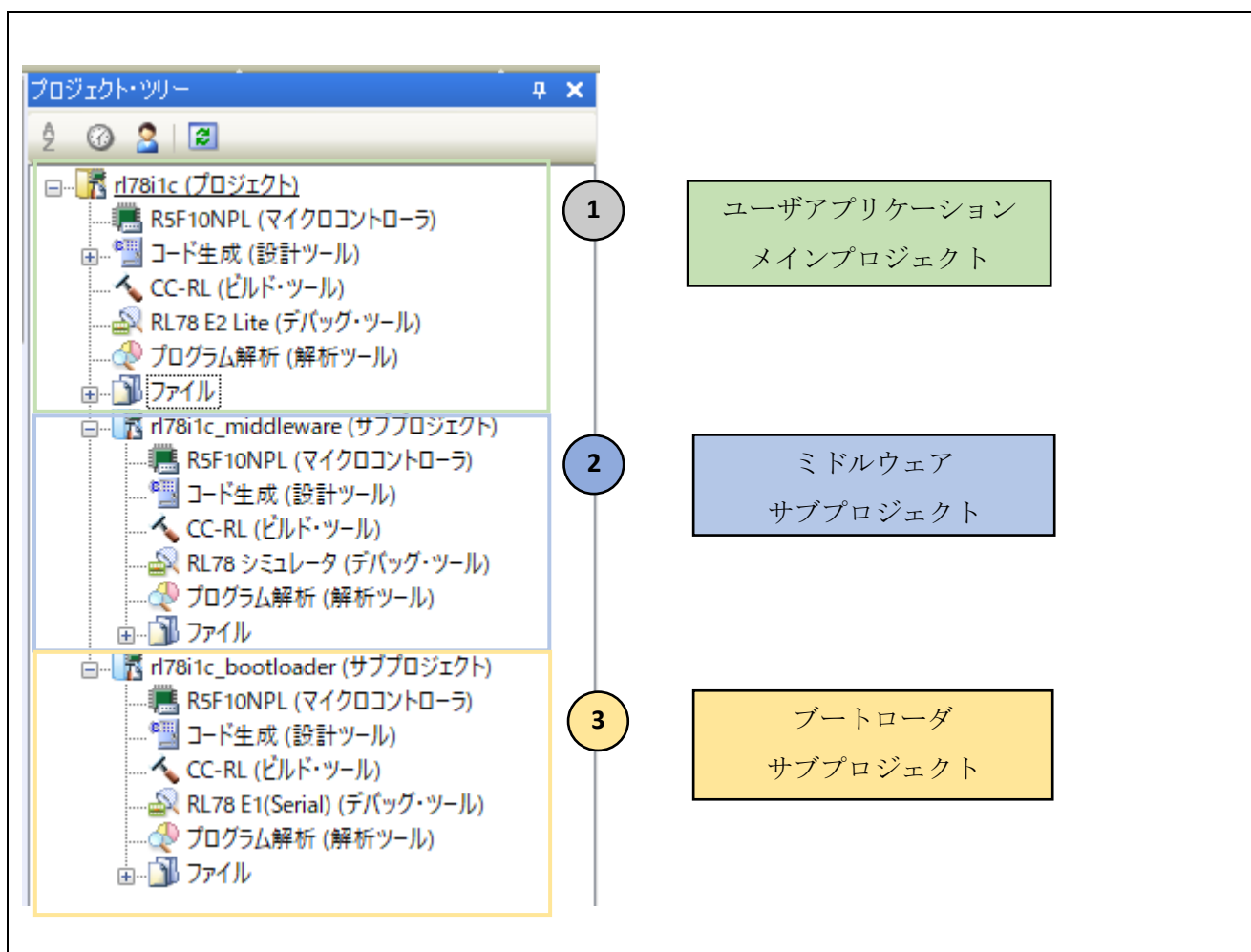


図 4-1: プロジェクト構成

## 4.2. メモリ配置

以下に、内部 ROM および内部 RAM の使用領域を示します。

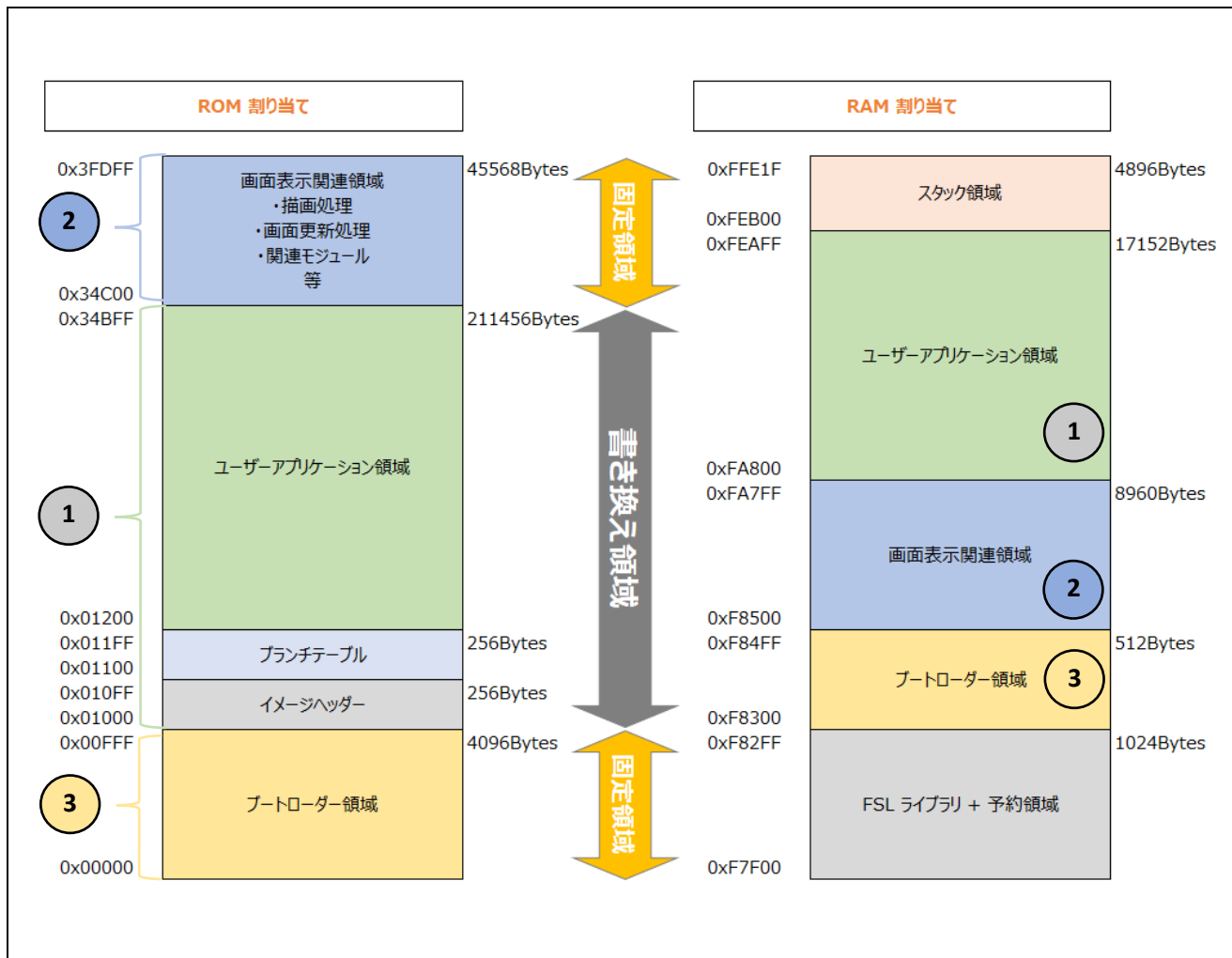


図 4-2: メモリマッピング

上記①、②、③の各番号は、図 4-1 の各プロジェクトとの対応を示しています。

書き換え対象となるのは、図の書き換え領域全体です。ブートローダ及び画面表示関連の機能は固定領域に配置されており、FOTA で書き換えることはできません。

0x00000~0x3FDFF が起動バンクの配置となります。同様のものを 0x40000 以降のもう一つのバンクにも配置します。

### 4.2.1. ユーザアプリケーションプロジェクトのメモリ配置

rl78i1c プロジェクト (図 4-1 ①) について、ROM および RAM を所定の配置にするための設定を行います。割り当ての方法を CS+ と、e<sup>2</sup>studio それぞれの場合について示します。

- 0000-0007f : オンチップデバッガ機能のための ROM 領域
- 000c0-000d7 : MCU の制御に使用するオプションバイトとセキュリティ ID のための ROM 領域
- 01000-34bff : ユーザアプリケーション用の ROM 領域
- 3fe00-3ffff : デバッガの monitor2 のための ROM 領域
- fa800-feaff : ユーザアプリケーション用の RAM 領域

#### ■メモリ種別のアドレス範囲

#### CS+の場合:

CC-RL (ビルド・ツール)→リンクオプション→ベリファイ→メモリ種別のアドレス範囲

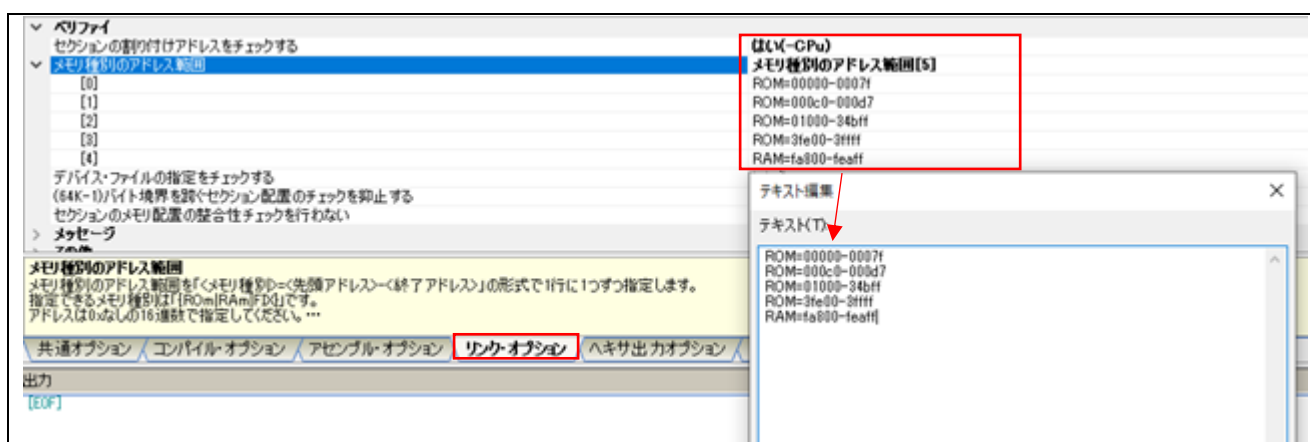


図 4-3 : ユーザアプリケーション メモリ種別のアドレス範囲 (CS+)

**e<sup>2</sup>studio の場合:**

プロパティ→C/C++ ビルド→設定→Linker→デバイス→セクション割り付け領域の整合性をチェックする  
アドレス範囲とメモリ種別

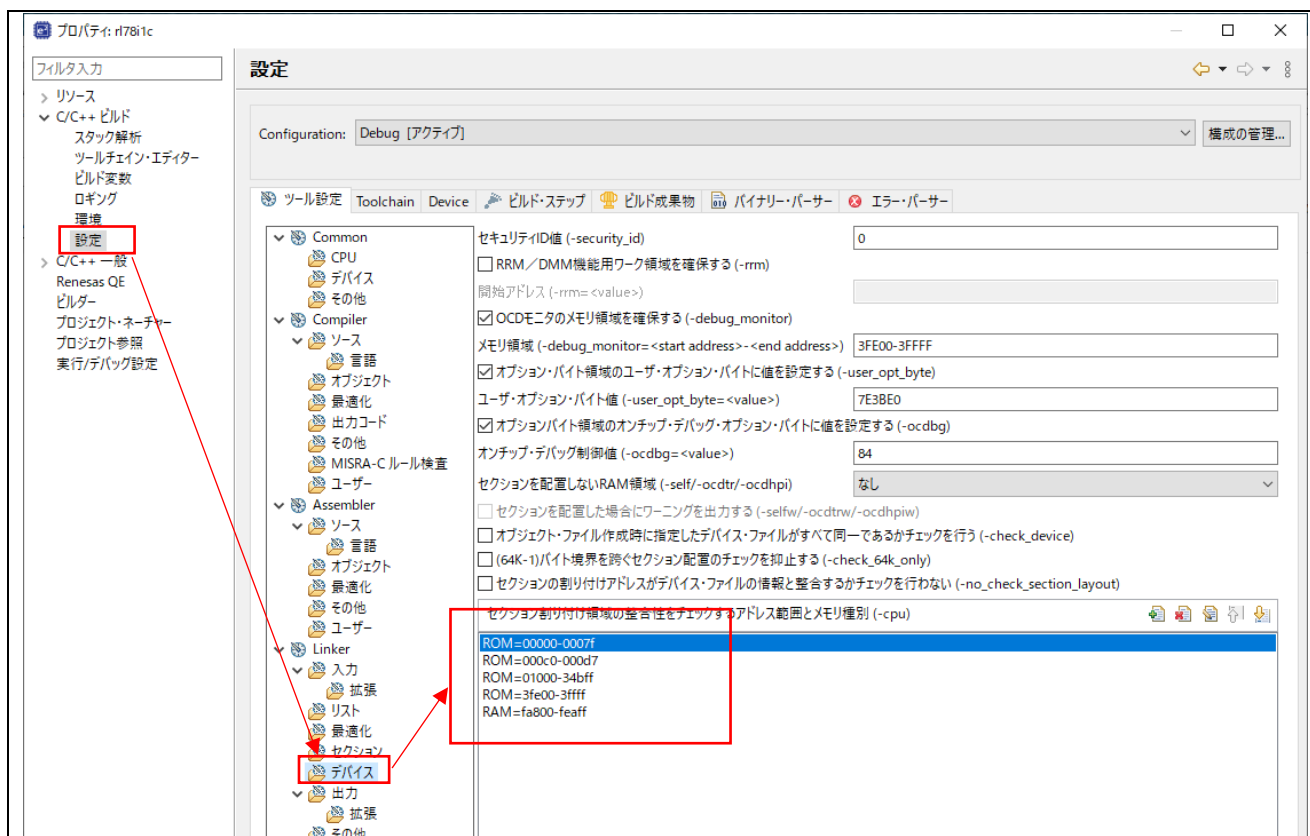


図 4-4 : ユーザアプリケーション メモリ種別のアドレス範囲 (e2studio)

■デバッグ・モニタ領域

**CS+の場合:**

CC-RL (ビルド・ツール)→リンクオプション→デバイス→デバッグ・モニタ領域を設定する



図 4-5 : ユーザアプリケーション デバッグ・モニタ領域 (CS+)

**e<sup>2</sup>studio の場合:**

プロパティ→C/C++ ビルド→設定→Linker→デバイス→メモリ領域

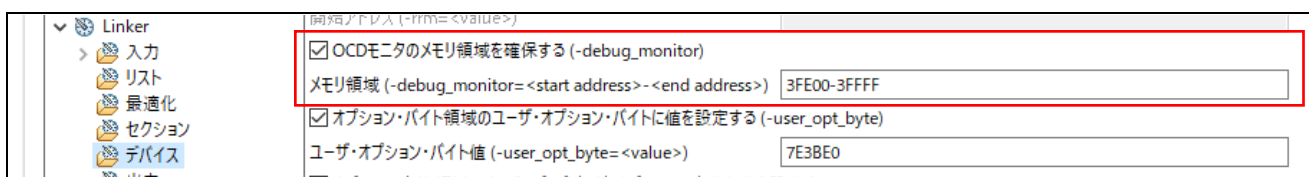


図 4-6 : ユーザアプリケーション デバッグ・モニタ領域 (e2studio)

■セクション設定

**CS+の場合:**

CC-RL (ビルド・ツール)→リンクオプション→セクション→セクションの開始アドレス

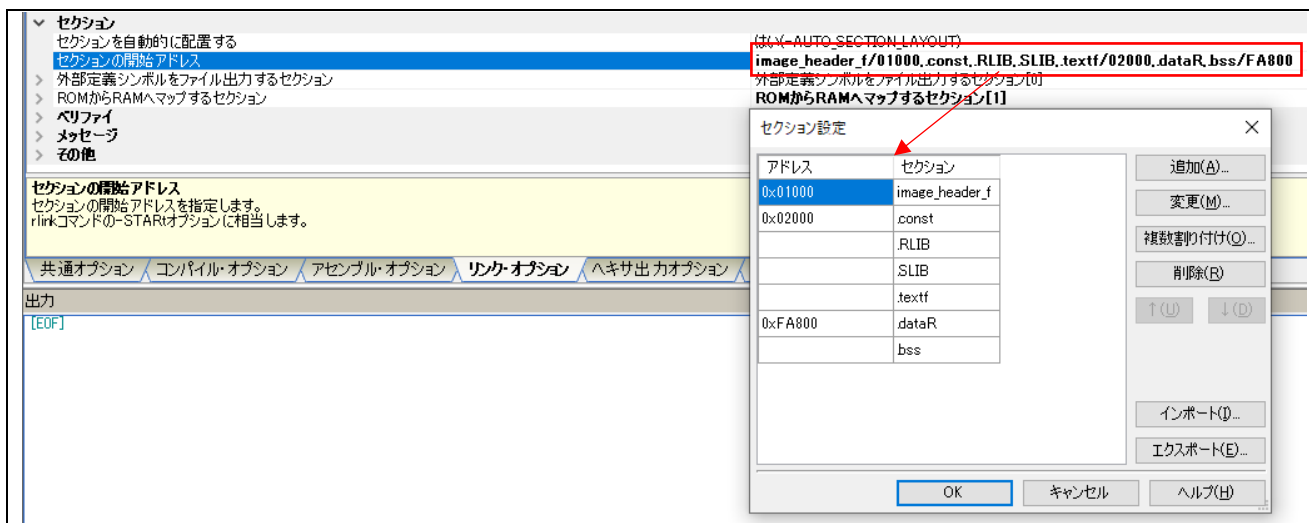


図 4-7 : ユーザアプリケーションセクション設定 (CS+)

**e<sup>2</sup>studio の場合:**

プロパティ→C/C++ ビルド→設定→Linker→セクション

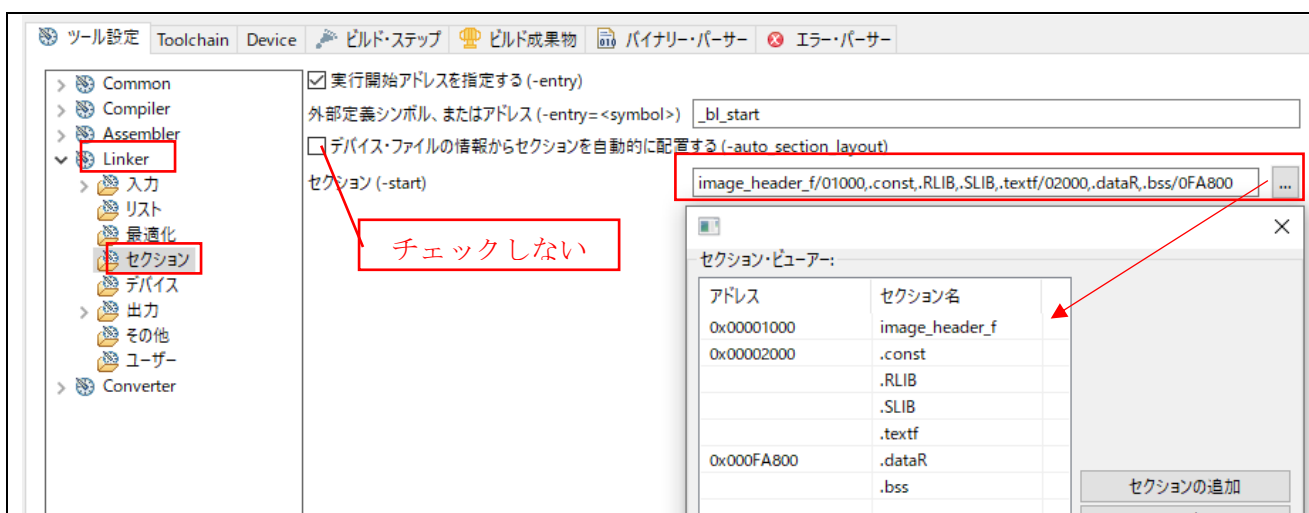


図 4-8 : ユーザアプリケーションセクション設定 (e2studio)



### 4.2.2. ミドルウェアサブプロジェクトのメモリ配置

rl78i1c\_middleware サブプロジェクト (図 4-1 ②) について、ROM および RAM を所定の配置にするための設定を行います。割り当ての方法を CS+ と、e<sup>2</sup>studio それぞれの場合について示します。

#### ■メモリ種別のアドレス範囲

#### CS+の場合:

CC-RL (ビルド・ツール)→リンクオプション→ベリファイ→メモリ種別のアドレス範囲

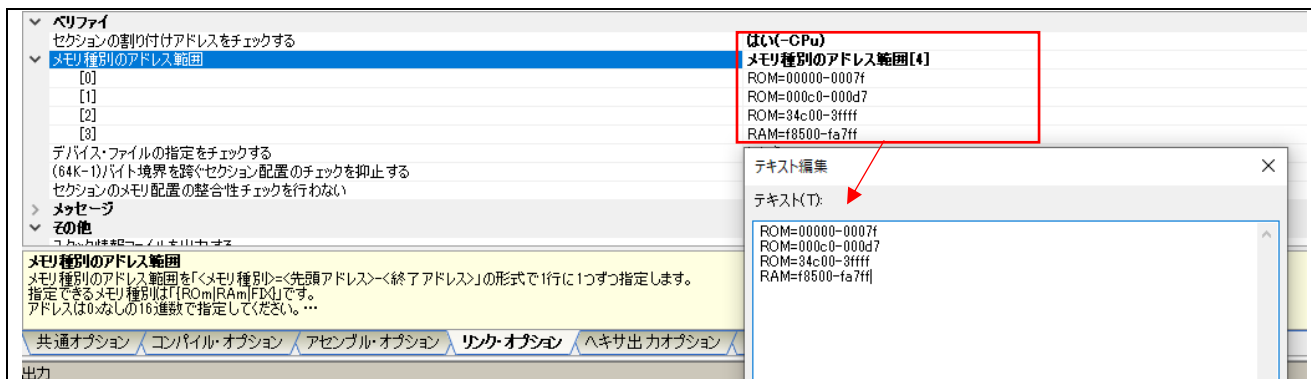


図 4-9: ミドルウェア メモリ種別のアドレス範囲 (CS+)

#### e<sup>2</sup>studio の場合:

プロパティ→C/C++ ビルド→設定→Linker→デバイス→セクション割り付け領域の整合性をチェックする  
アドレス範囲とメモリ種別

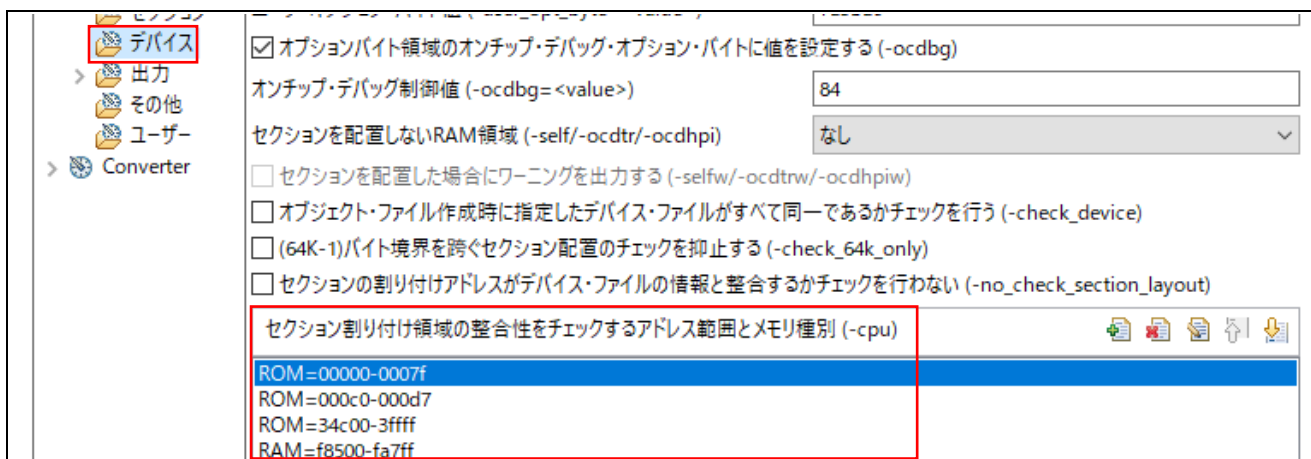


図 4-10: ミドルウェア メモリ種別のアドレス範囲 (e2studio)

■セクション設定と外部定義シンボル

**CS+の場合:**

CC-RL (ビルド・ツール)→リンクオプション→セクション→セクションの開始アドレス

CC-RL (ビルド・ツール)→リンクオプション→セクション→外部定義シンボルをファイル出力するセクション

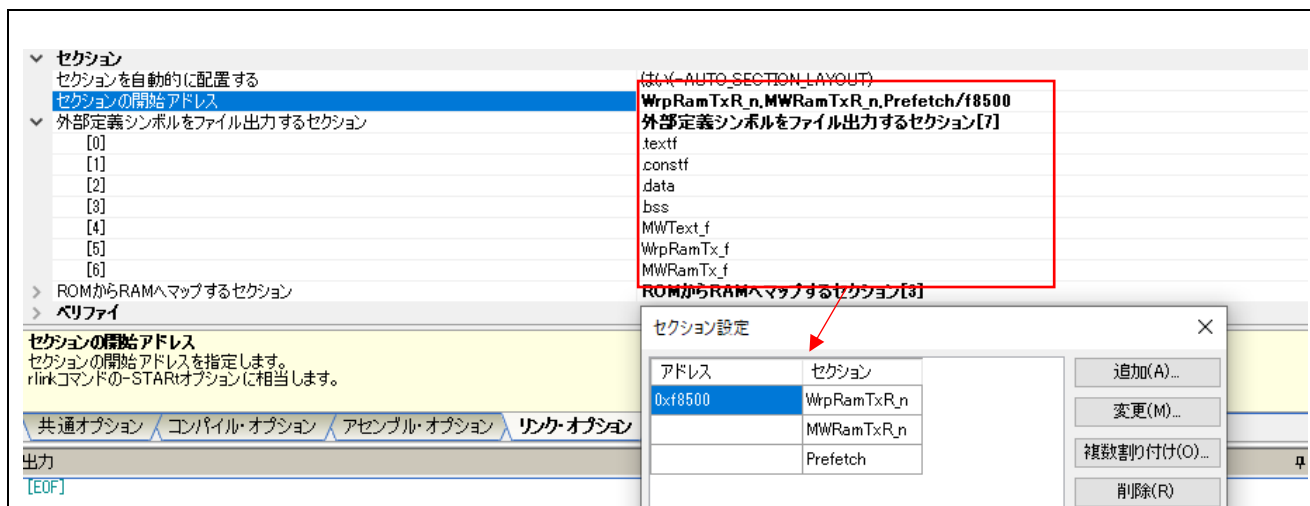


図 4-11: ミドルウェア セクション設定と外部定義シンボル (CS+)

**e<sup>2</sup>studio の場合:**

プロパティ→C/C++ ビルド→設定→Linker→セクション→セクション

プロパティ→C/C++ ビルド→設定→Linker→出力→外部定義シンボルをファイル出力するセクション

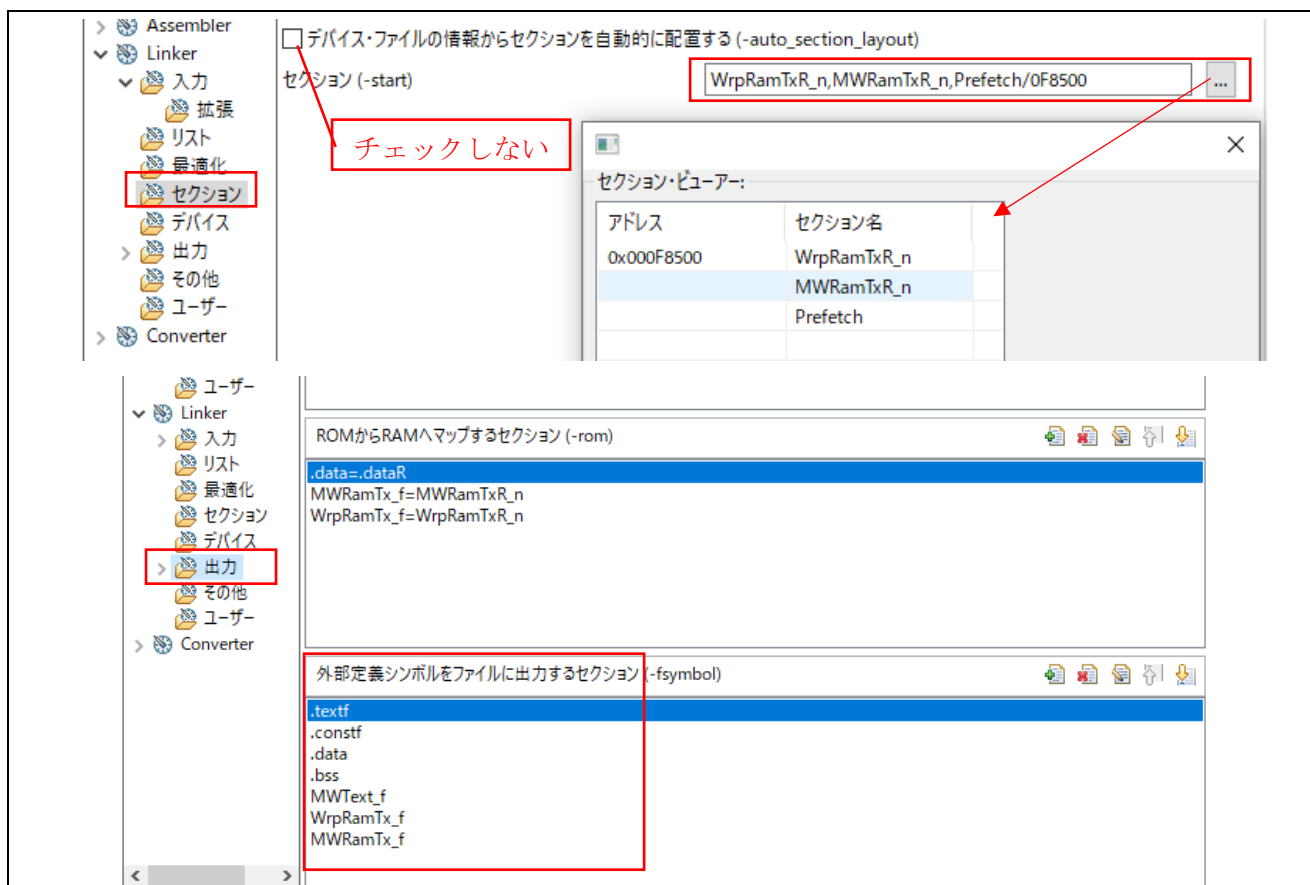


図 4-12 : ミドルウェア セクション設定と外部定義シンボル (e2studio)

### 4.2.3. ブートローダサブプロジェクトのメモリ配置

rl78i1c\_bootloader サブプロジェクト (図 4-1 ③) について、ROM および RAM を所定の配置にするための設定を行います。割り当ての方法を CS+ と、e<sup>2</sup>studio それぞれの場合について示します。

#### ■メモリ種別のアドレス範囲

#### CS+の場合:

CC-RL (ビルド・ツール)→リンクオプション→ベリファイ→メモリ種別のアドレス範囲

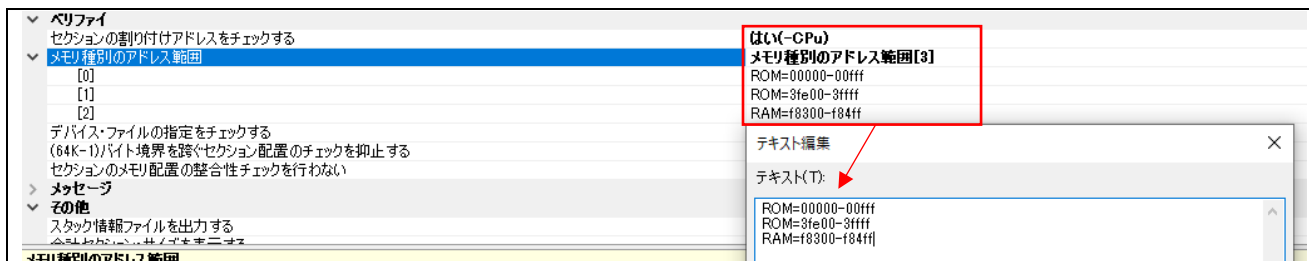


図 4-13 : ブートローダ メモリ種別のアドレス範囲 (CS+)

#### e<sup>2</sup>studio の場合:

プロパティ→C/C++ ビルド→設定→Linker→デバイス→セクション割り付け領域の整合性をチェックする  
アドレス範囲とメモリ種別

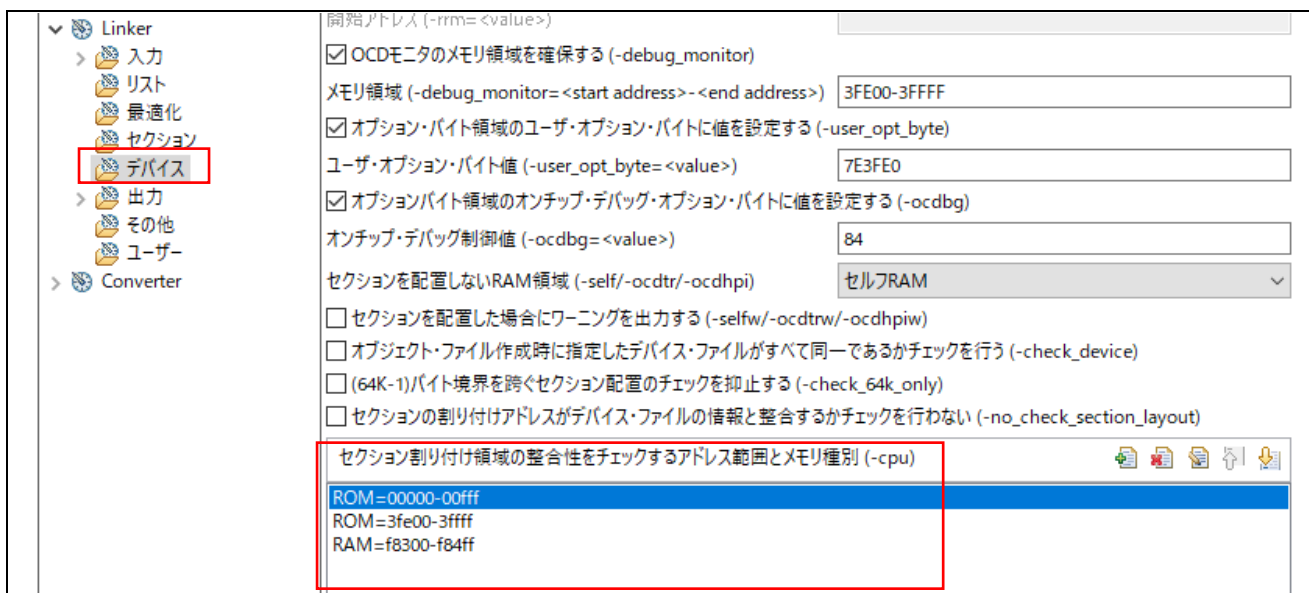


図 4-14 : ブートローダ メモリ種別のアドレス範囲 (e2studio)

■セクション設定と外部定義シンボル

**CS+の場合:**

CC-RL (ビルド・ツール)→リンクオプション→セクション→セクションの開始アドレス

CC-RL (ビルド・ツール)→リンクオプション→セクション→外部定義シンボルをファイル出力するセクション

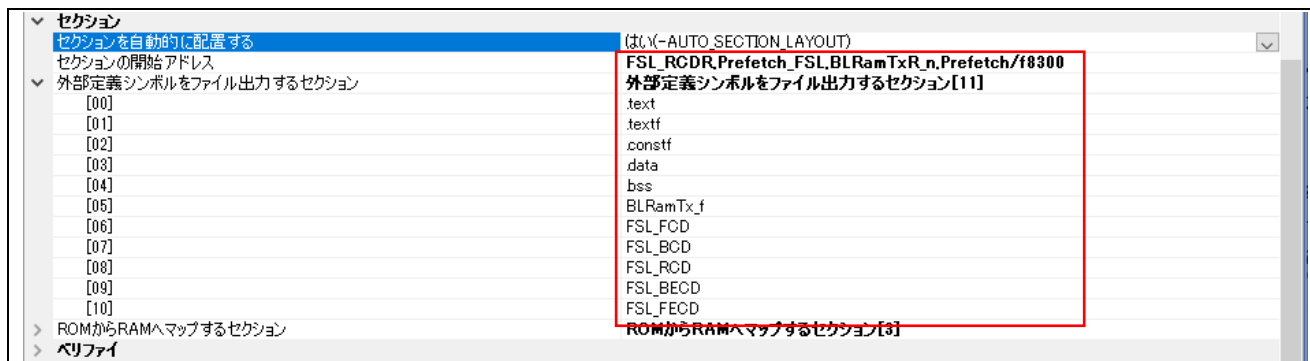


図 4-15 : ブートローダ セクション設定と外部定義シンボル (CS+)

**e<sup>2</sup>studio の場合:**

プロパティ→C/C++ ビルド→設定→Linker→セクション→セクション

プロパティ→C/C++ ビルド→設定→Linker→出力→外部定義シンボルをファイル出力するセクション

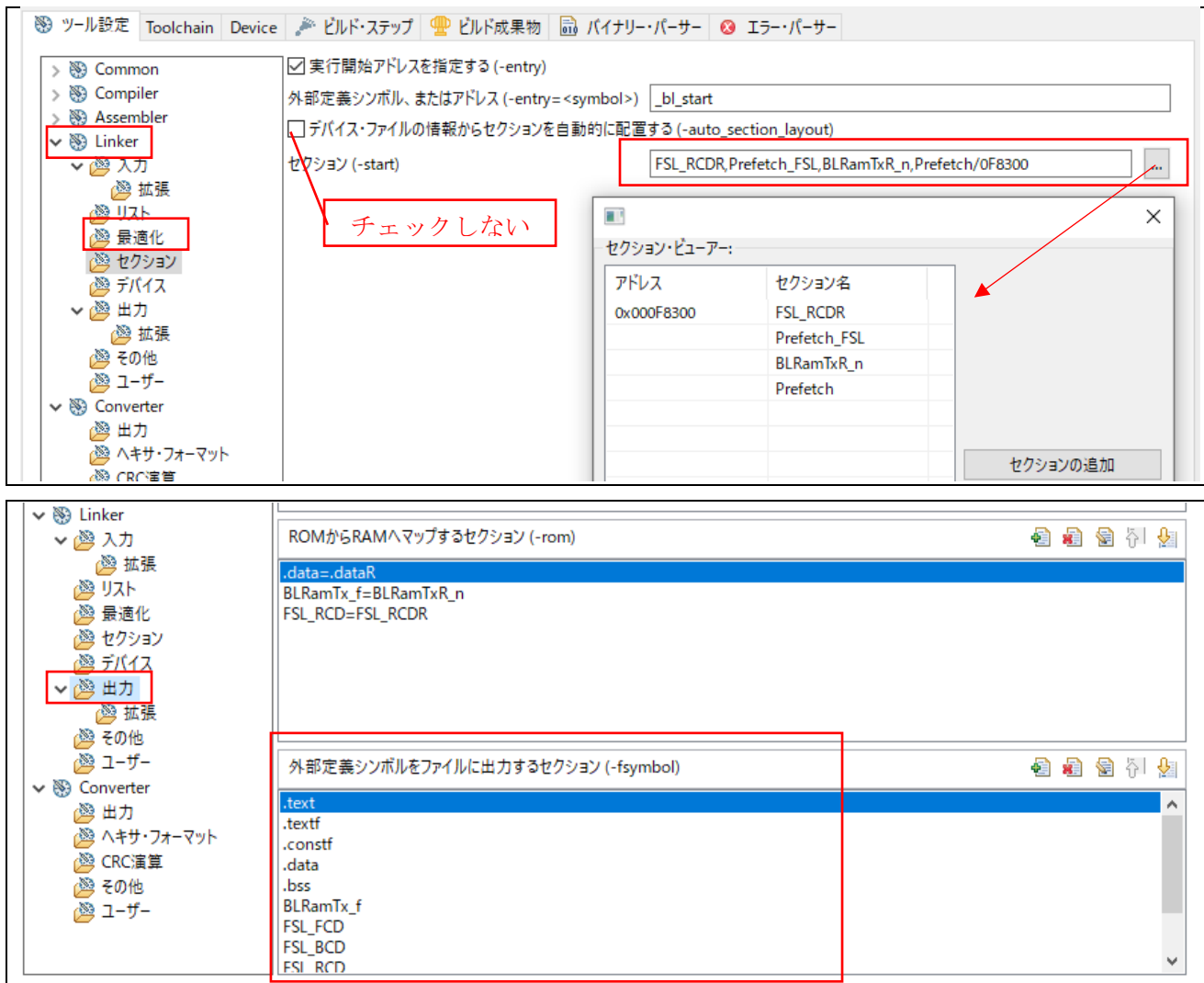


図 4-16 : ブートローダ セクション設定と外部定義シンボル (e2studio)

### 4.3. リンクオプションの補足説明 (e<sup>2</sup>studio)

e<sup>2</sup>studio では、[C/C++ ビルド→設定 → Linker→セクション]の[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) にチェックを入れると、リンカオプション(-start)が出力されなくなります。そのため、ユーザオプション (図 4-17) で、-auto\_section\_layout を指定してください。

これは、rl78i1c プロジェクト (図 4-1 ①)、rl78i1c\_middleware サブプロジェクト (図 4-1 ②)、rl78i1c\_bootloader サブプロジェクト (図 4-1 ③) の全てで必要になります。

#### **e<sup>2</sup>studio** のみ:

プロパティ→C/C++ ビルド→設定→Linker→ユーザ→追加するオプション

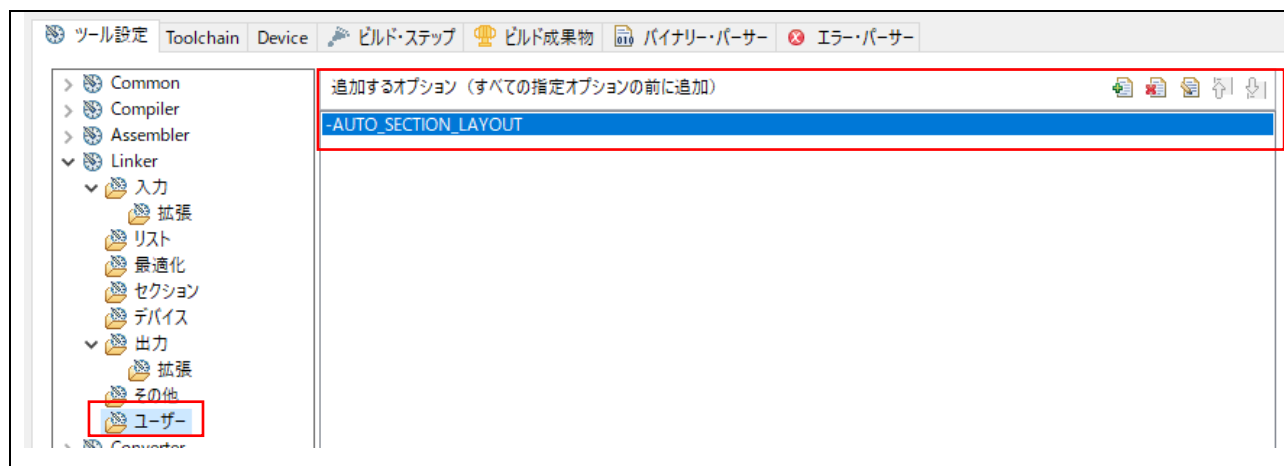


図 4-17 : AUTO\_SECTION\_LAYOUT ユーザオプション (e<sup>2</sup>studio)

#### 4.4. 外部定義シンボルファイルの利用について

メインプロジェクト・サブプロジェクト間での呼び出しや情報の共有については、外部定義シンボルファイルを使用しています。これを利用するためには、[4.2.2](#)および[4.2.3](#)で設定した外部定義シンボルのファイル出力の後、参照したいプロジェクトに含める必要があります。

本サンプルでは、外部シンボルの参照について、以下のように設定しています。

- メインプロジェクトでミドルウェアサブプロジェクトとブートローダサブプロジェクトを参照
- ミドルウェアサブプロジェクトからブートローダサブプロジェクトを参照

#### CS+の場合:

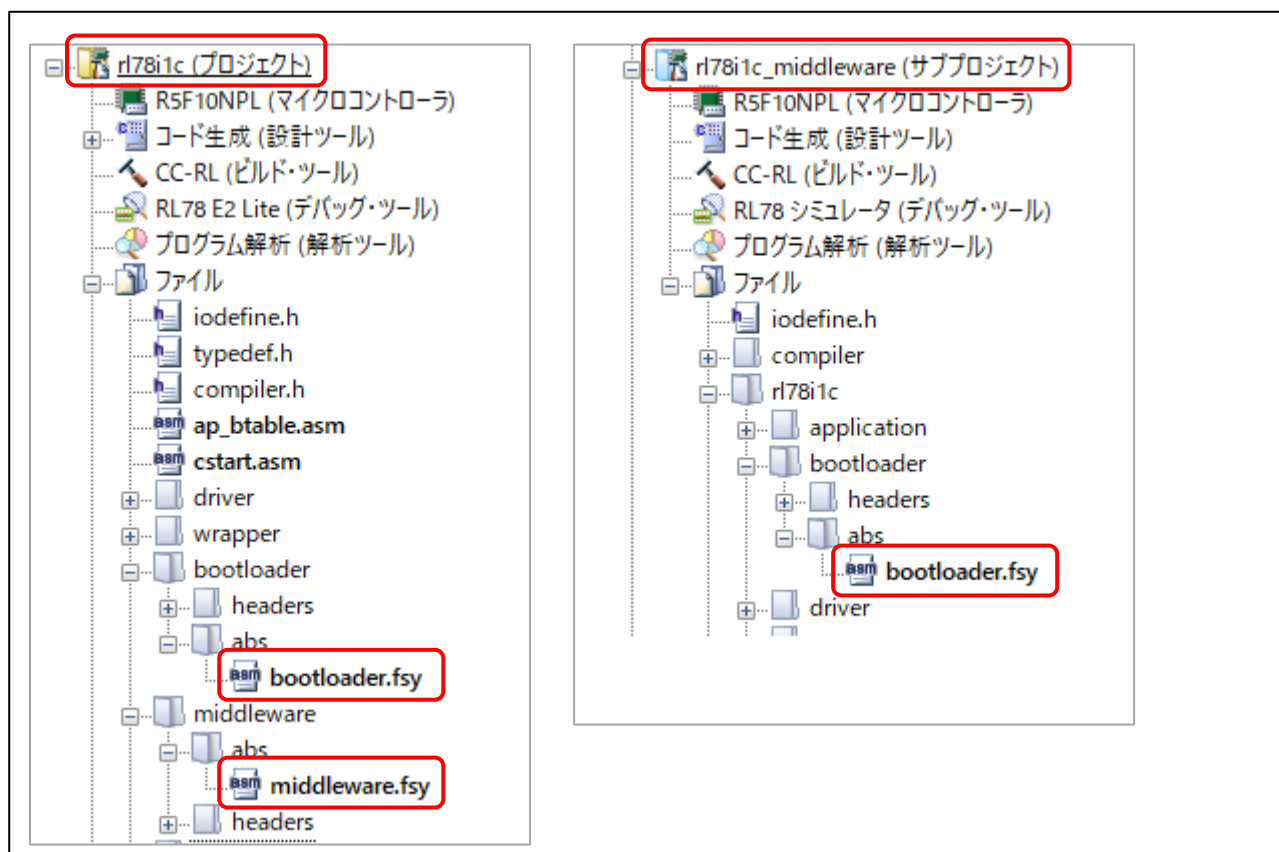


図 4-18 : 外部定義シンボルファイル (CS+)

#### e<sup>2</sup>studio の場合:

CS+と同様のため、省略。



## 4.5. ROM から RAM へのマッピング設定

### 4.5.1. LCD 更新処理ルーチン

画面の表示はタイマー割り込みで制御されています。この割り込み処理では画面上のインジケータ・バーの更新や、入力された文字等の表示を行います。

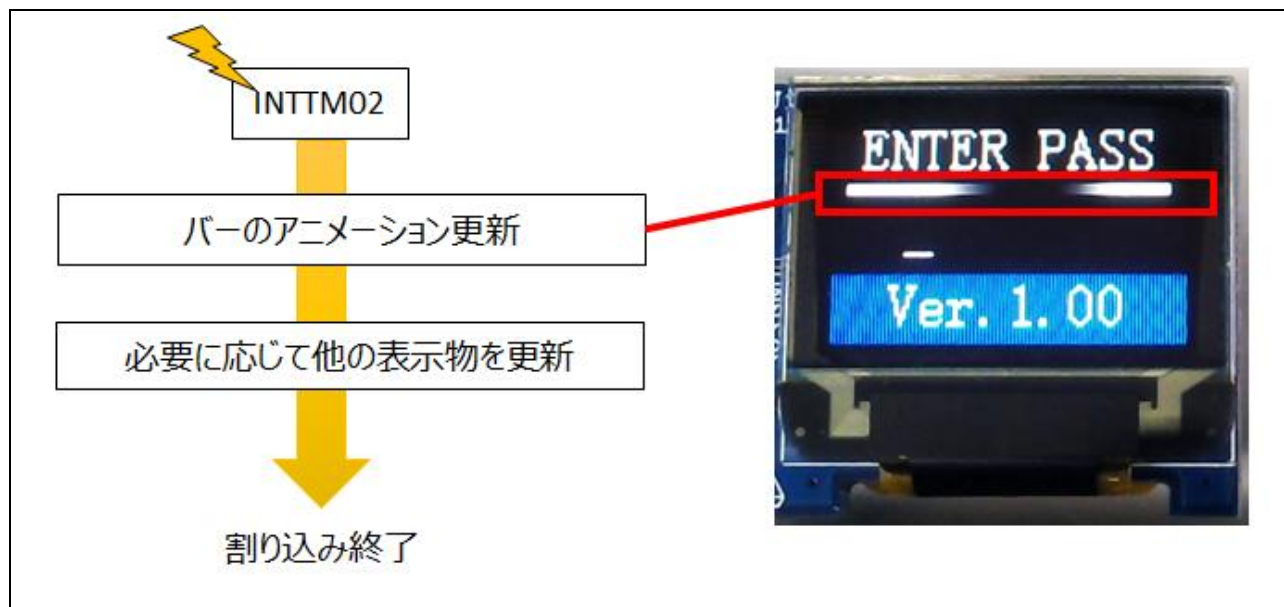


図 4-19 : LCD 更新処理

通常は ROM 上から動作しますが、バンク切り替え中は ROM を参照できないため、RAM 上から動作させることでインジケータ・バーのアニメーション動作を継続しています。

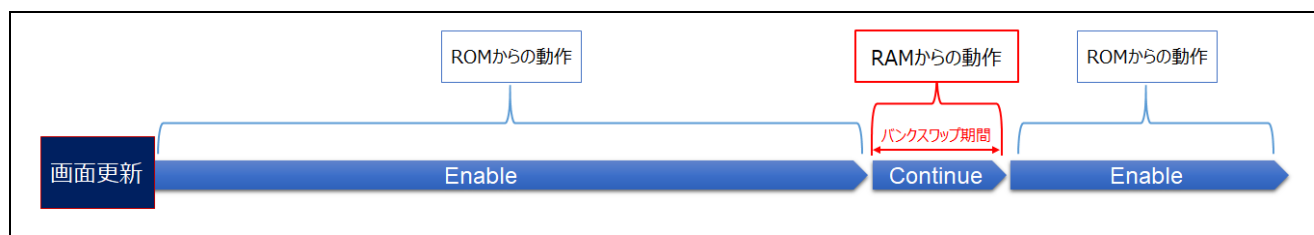


図 4-20 : 画面更新 Continuous Operation

図 4-21 は、RAM 実行前の RAM への実行コード展開イメージです。

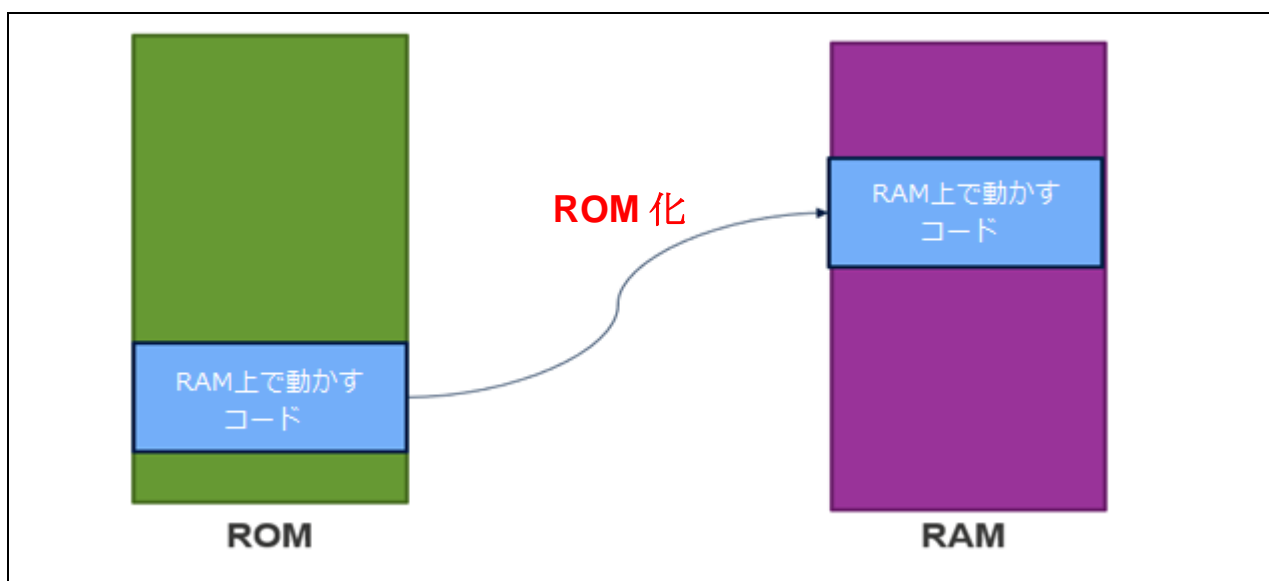


図 4-21 : ROM から RAM へのコード展開

ROM 上で動作している間の割り込みコールバック関数は、RAM 上で動作している間、全ての割り込みを制御する `MW_RunOnRam_RamIsr` 関数へ置き換えられます。この関数で、各割り込みフラグ（具体的には `TMIF02` と `CSIF30`）をチェックして、各割り込み相当の動作を行っています。また、処理後は各割り込みフラグを手動でクリアする必要があります。

ROM から RAM へのコードのコピーはセクション単位で行います。そのため、あらかじめセクションを分割し、RAM 実行が必要なコードを分離しておきます。

RAM 上での動作に必要なコードは `MW_RunOnRam_PrepareFunctions` 関数で ROM から RAM へコピーしています。

### 4.5.2. ユーザアプリケーションプロジェクトのマッピング

rl78i1cプロジェクト（図 4-1 ①）について、ROM から RAM へマップする領域を以下のように設定します。

#### CS+の場合:

CC-RL (ビルド・ツール)→リンクオプション→セクション→ROM から RAM へマップするセクション

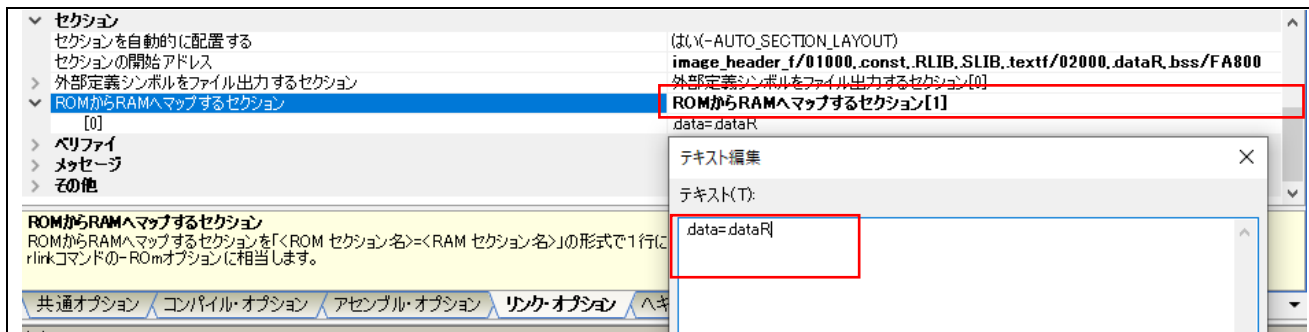


図 4-22 : ユーザアプリケーション ROM から RAM へマップするセクション (CS+)

#### e<sup>2</sup>studio の場合:

プロパティ→C/C++ ビルド→設定→Linker→出力→ROM から RAM へマップするセクション

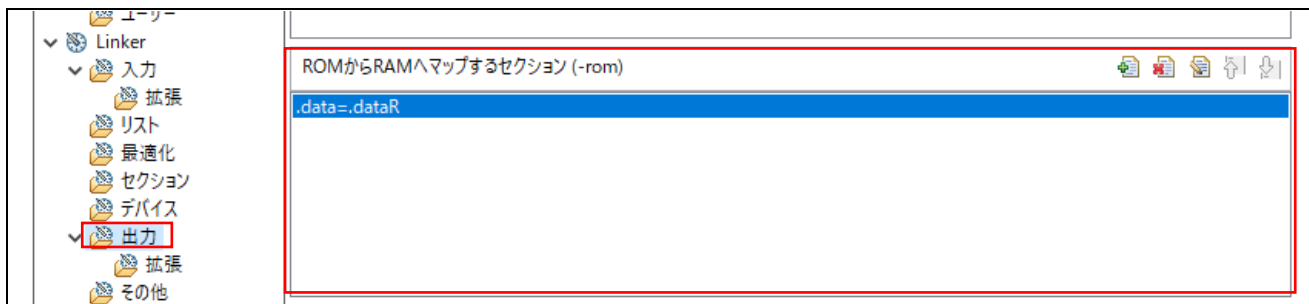


図 4-23 : ユーザアプリケーション ROM から RAM へマップするセクション (e2studio)

### 4.5.3. ミドルウェアサブプロジェクトのマッピング

rl78i1c\_middleware サブプロジェクト (図 4-1 ②) について、ROM から RAM へマップする領域を以下のように設定します。

#### CS+の場合:

CC-RL (ビルド・ツール)→リンクオプション→セクション→ROM から RAM へマップするセクション

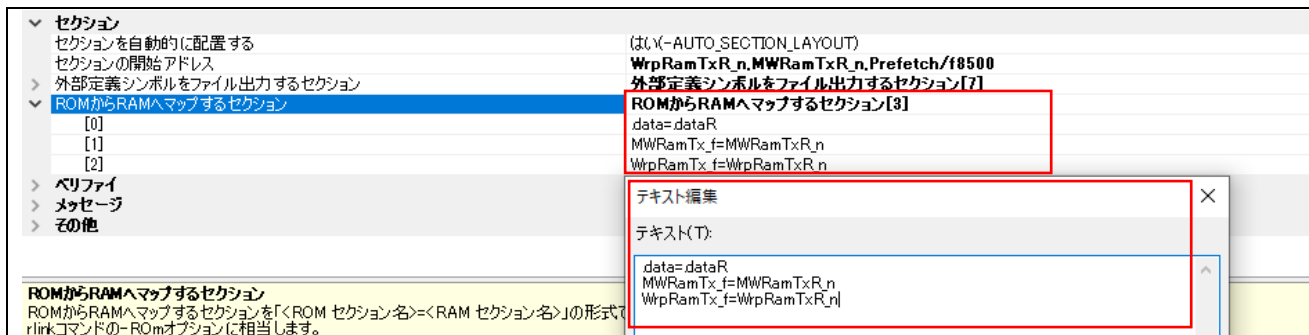


図 4-24 : ミドルウェア ROM から RAM へマップするセクション (CS+)

#### e<sup>2</sup>studio の場合:

プロパティ→C/C++ ビルド→設定→Linker→出力→ROM から RAM へマップするセクション

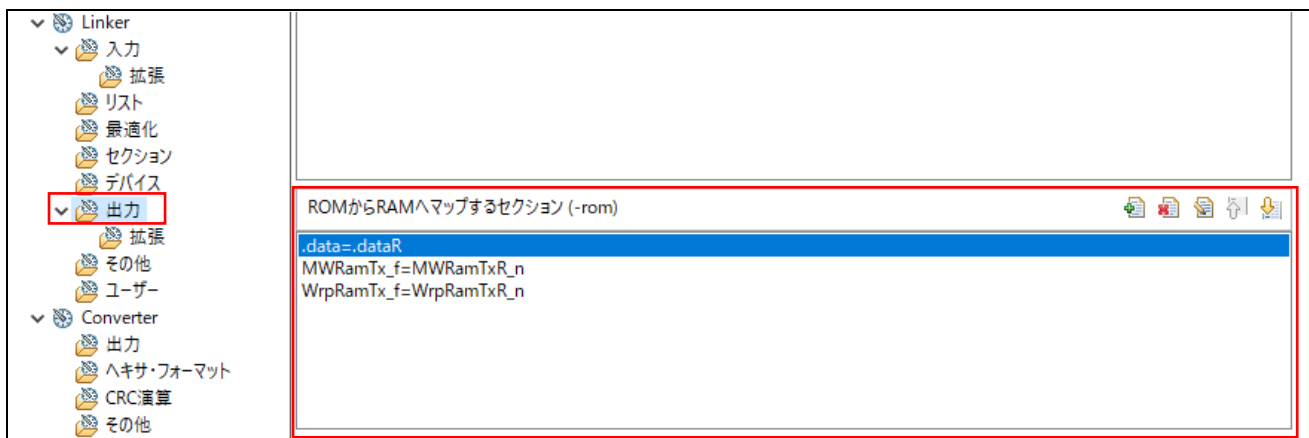


図 4-25 : ミドルウェア ROM から RAM へマップするセクション (e2studio)

MWRamTx : RAM 上から画面更新を行うための関数を集めたセクションです。

WrpRamTx : RAM 上で MWRamTx 内の関数が使用するユーティリティ関数セクションです。

#### 4.5.4. ブートローダサブプロジェクトのマッピング

rl78i1c\_bootloader サブプロジェクト (図 4-1 ③) について、ROM から RAM へマップする領域を以下のように設定します。

##### CS+の場合:

CC-RL (ビルド・ツール)→リンクオプション→セクション→ROM から RAM へマップするセクション

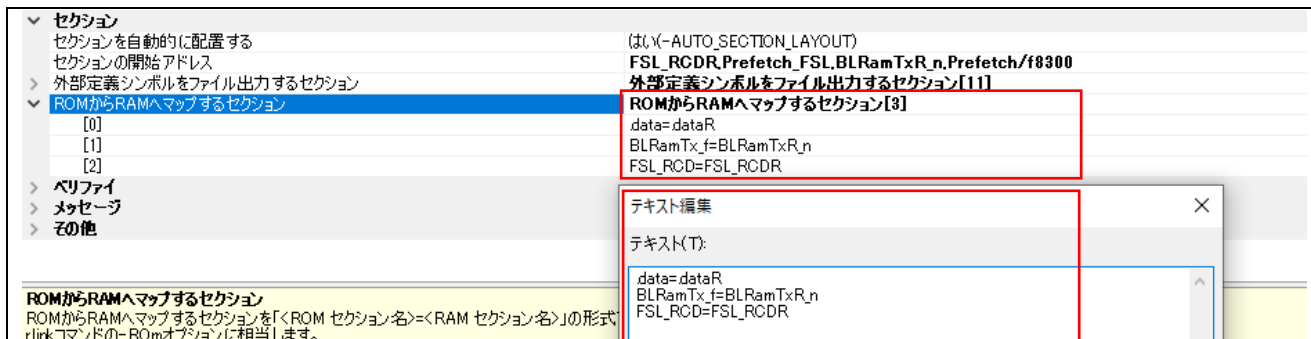


図 4-26 : ブートローダ ROM から RAM へマップするセクション (CS+)

##### e<sup>2</sup>studio の場合:

プロパティ→C/C++ ビルド→設定→Linker→出力→ROM から RAM へマップするセクション

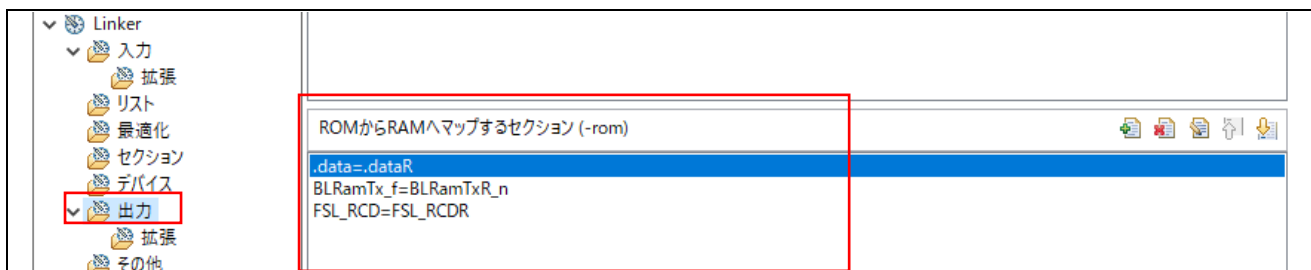


図 4-27 : ブートローダ ROM から RAM へマップするセクション (e2studio)

BLRamTx : ブートローダが RAM 上から使用する関数のセクションです。

FSL\_RCD : FSL ライブラリのセクションです。

### 4.6. ブランチテーブル

ユーザアプリは、ブートローダのベクタテーブルとは別に、割り込み処理用のブランチテーブルを持ちます。ブランチテーブル方式の方が関数ポインタでの呼び出しより高速となります。

ユーザアプリケーションとブートローダはベクタテーブルを共有できますが、ベクタテーブルは固定であり、ユーザアプリの更新に合わせて変更することができません。

また、表示物関連のコードは **far** 領域に配置します。そのため、ブランチテーブルはユーザアプリケーションプロジェクト用の **ap\_btable.asm** と画面表示関連のミドルウェアサブプロジェクト用の **mw\_btable.asm** の二つに分かれています。ブートローダのサブプロジェクトのベクタテーブルを起点として、それぞれに分岐します。そのため、ブートローダのサブプロジェクトでは、ベクタテーブルのセクションを分離する必要があります。

#### CS+の場合:

CC-RL (ビルド・ツール)→リンクオプション→出力コード→ベクタ・テーブル・セクションの分割生成

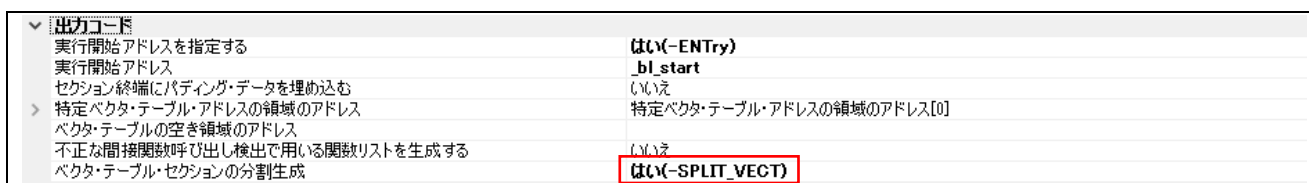


図 4-28 : ベクタテーブルの分離 (CS+)

#### e<sup>2</sup>studio の場合:

プロパティ→C/C++ ビルド→設定→Linker→ベクタテーブルのセクションを分割生成する

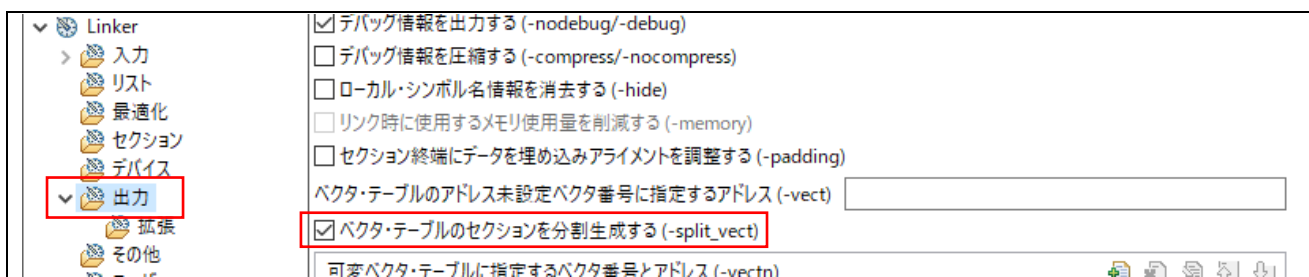


図 4-29 : ベクタテーブルの分離 (e2studio)

次の図は、表示関連処理のブランチテーブルによる処理の流れを示しています。

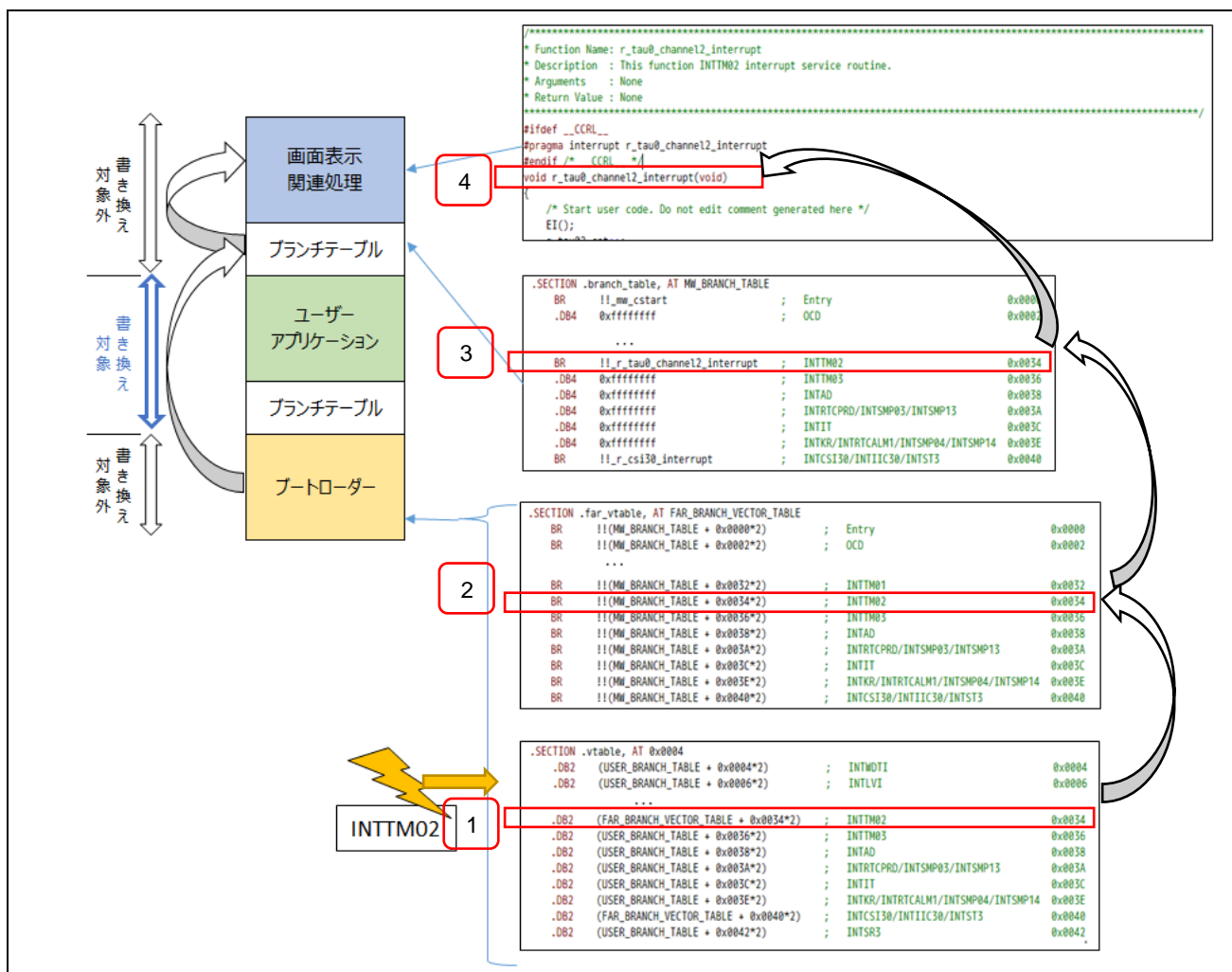


図 4-30 : ブランチテーブルフロー

表示処理は、INTTM02を使用したタイマー割り込みを起点に動作しています。INTTM02 割り込みが発生すると、指定された位置にジャンプします (ステップ 1)。

```
.DB2 (FAR_BRANCH_VECTOR_TABLE + 0x0034*2) ; INTTM02
```

また、画面表示関連の機能は far 領域に置かれているため、中間テーブルへジャンプします (ステップ 2)。その後、表示関連機能のブランチテーブルへジャンプし (ステップ 3)、割り込み関数そのものへ到達します (ステップ 4)。

補足：関連するソースファイルは以下の通りです。

- 1 **bl\_vtable.asm** - rl78i1c\_bootloader subproject
- 2 **bl\_far\_vtable.asm** - rl78i1c\_bootloader subproject
- 3 **mw\_btable.asm** - rl78i1c\_middleware subproject
- 4 **r\_tau\_user\_mw.c** - rl78i1c\_middleware subproject

## 4.7. Continuous Operation FOTA サンプルプロジェクトの API 関数

### 4.7.1. API 関数一覧

表 4-1 : API 関数

Function	Explanation
COMMAND_PollingProcessing	UART でのコマンド受信を監視します。
COMMAND_InvokeBankSwap	"bswap" コマンドの実行関数です。
MW_RunOnRam_NonStopBankSwap	バンクスワップ実行関数です。
MW_RunOnRam_PrepareFunctions	バンクスワップの準備を行う関数です。 RAM 上での動作を行うコードを ROM から RAM にコピーします。
MW_RunOnRam_DisableInterruptsExceptDisplayRelated	画面表示関連以外のマスカブル割り込みを禁止します。
BL_RunOnRam_PrepareFunctions	バンクスワップの準備を行う関数です。 RAM 上での動作を行うコードを ROM から RAM にコピーします。
BL_FLASH_RAM_SwapBankWithRamlSr	割り込みベクタを RAM 上のものへ置き換えた後、バンクスワップを行います。
FSL_ChangeInterruptTable	割り込みベクタを RAM 上のものへ置き換えます。
FSL_SwapActiveBootCluster	バンクスワップを行います。
FSL_RestoreInterruptTable	割り込みベクタを ROM 上のものへ戻します。
BL_FLASH_RAM_JumpBankSwapEntry	バンクスワップ後のエントリポイントへ移動します。



### 4.7.2. Continuous Operation FOTA シーケンス

図 4-31 に、Continuous Operation FOTA の API 使用例を示します。

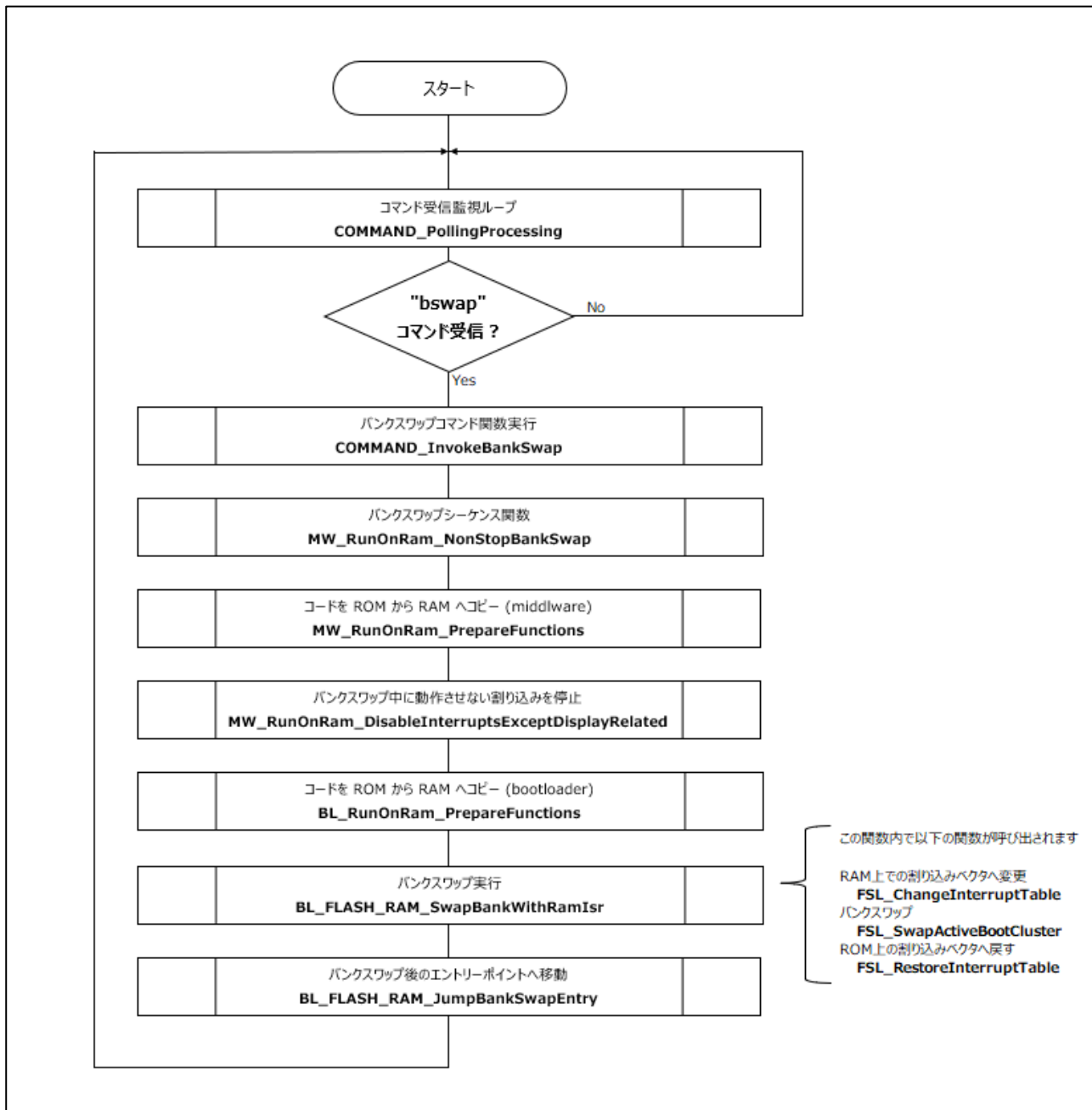


図 4-31 : Continuous Operation FOTA の API 使用例

## 4.8. サンプルプログラムのビルド

3.1 (1)の[rl78i1c\_production.mot]と、3.1 (2)の[rl78i1c\_v100.mot]および[rl78i1c\_v200.mot]の作成方法について説明します。以下のように、2.1に記載のパスコードと、2.2に記載のバージョン情報を設定してください。

### ■パスコードの設定

4桁のパスコードは、[platform.h]に定義しています。APP\_PASSCODE\_1~4がそれぞれパスコード1~4桁目に対応し、“0123456789ABCDF”の内の任意のキャラクタを設定できます。（“E”は決定キーとして使用しているため設定できません。）

```
23  /*****  
24  Macro definitions  
25  *****/  
26  /* Software version to be printed on start-up of FOTA Demo */  
27  #define APP_SOFTWARE_VERSION          (1)  
28  /* Passcode */  
29  #define APP_PASSCODE_1                ('1')  
30  #define APP_PASSCODE_2                ('2')  
31  #define APP_PASSCODE_3                ('3')  
32  #define APP_PASSCODE_4                ('A')  
33
```

図 4-32 : パスコードおよびバージョンの設定 (platform.h)

### ■バージョン情報の設定

バージョン情報は、[platform.h] および ビルド設定ファイルで指定します。（0~9の値が設定可能）  
指定した値は、バージョン表記の一桁目に反映されます。

#### ・[platform.h] のマクロ定義 APP\_SOFTWARE\_VERSION

0~9の値を設定してください。

#### ・ビルド設定ファイル

図 4-33、図 4-34 に示した個所に、0x000000~0x000009の値（上記、APP\_SOFTWARE\_VERSIONと同じ値）を設定してください。

**CS+の場合:**

CC-RL (ビルド・ツール)→リンクオプション→その他→リンク後に実行するコマンド

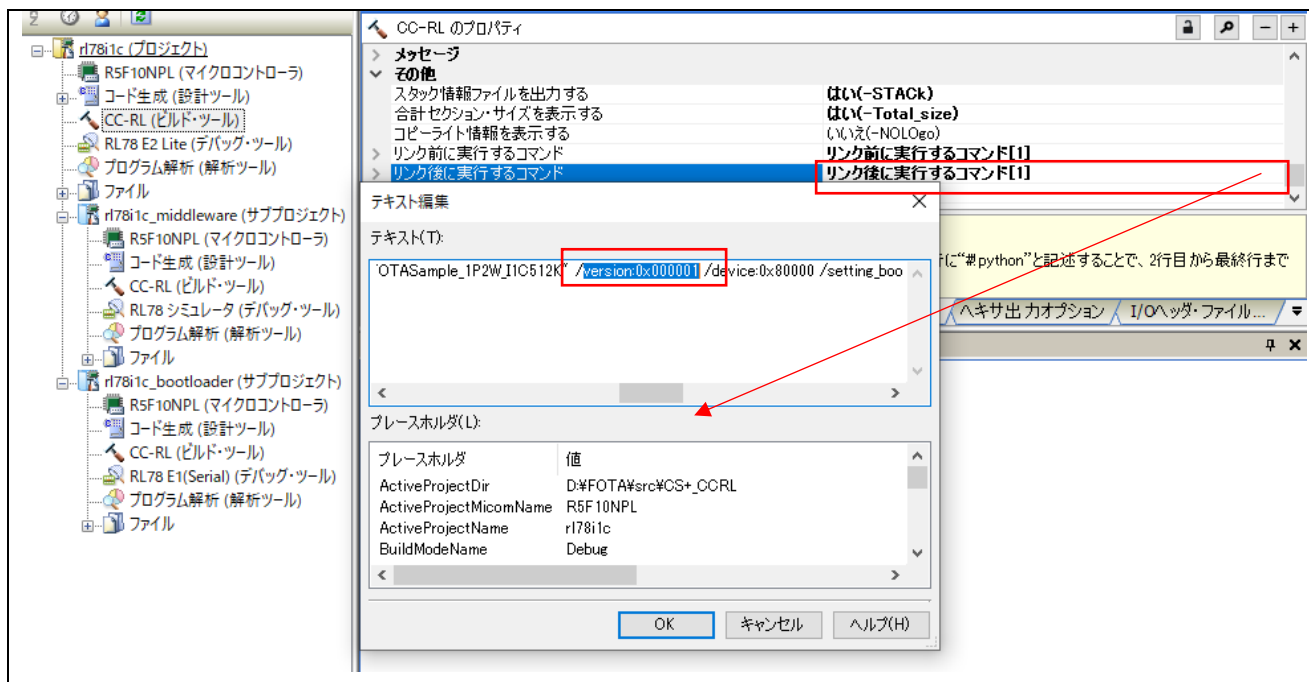


図 4-33 : バージョン設定 (CS+)

**e<sup>2</sup>studio の場合:**

プロパティ→C/C++ ビルド→設定→ビルド・ステップ→ビルド後のステップ

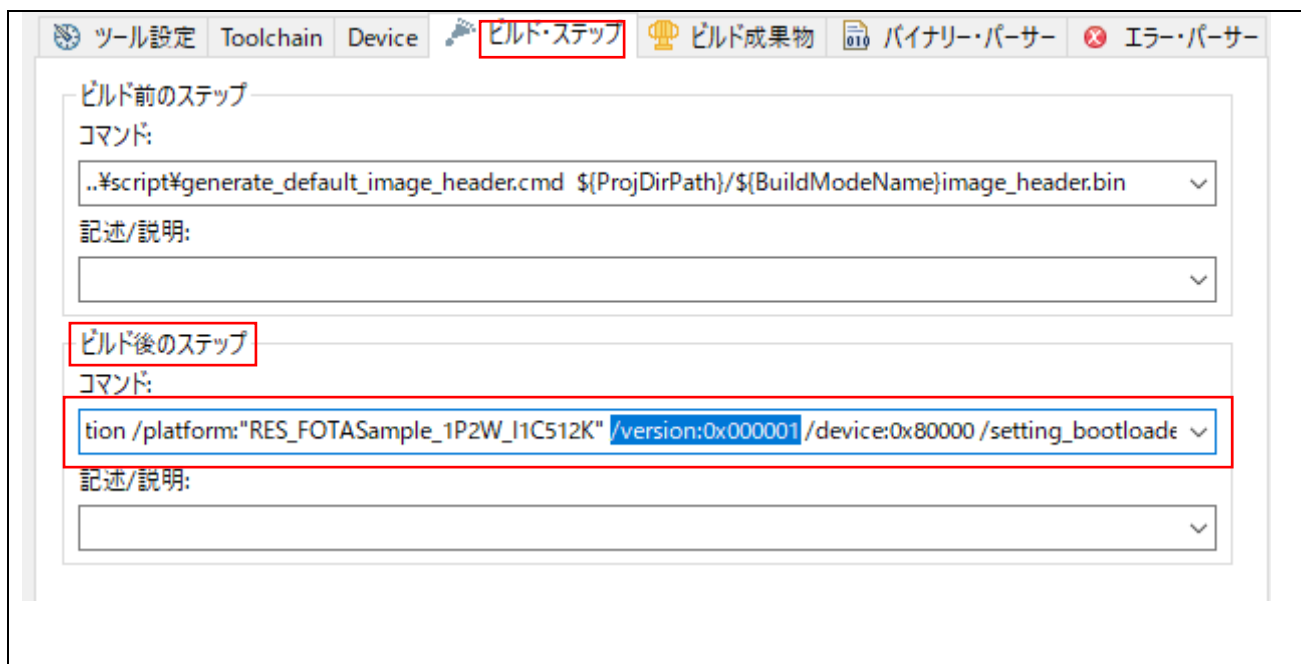


図 4-34 : バージョン設定 (e2studio)

## ■モトローラファイルの出力

### ・Ver.1.00 のファームウェア

[platform.h] および ビルド設定ファイルで、バージョン情報 "1" を設定します。(必要に応じてパスワードを変更してください。)

ビルドすると、以下のフォルダに、[rl78i1c\_production.mot] が生成されます。  
[rl78i1c\_production.mot]は、3.1 (1)で使用するファイルです。

### CS+

\Source\CS+\_CCRL\Debug\Image

### e<sup>2</sup>studio

\Source\e2studio\rl78i1c\Debug\Image

また、同フォルダに[rl78i1c0.mot]が生成されます。ファイル名を[rl78i1c\_v100.mot]に変更し、3.1 (2)で使用してください。

### ・Ver.2.00 のファームウェア

[platform.h] および ビルド設定ファイルで、バージョン情報 "2" を設定します。(必要に応じてパスワードを変更してください。)

ビルドすると、以下のフォルダに、[rl78i1c0.mot]が生成されます。ファイル名を[rl78i1c\_v200.mot]に変更し、3.1 (2)で使用してください。

同様に、[rl78i1c\_v300.mot]、[rl78i1c\_v400.mot]、・・・、[rl78i1c\_v900.mot]を生成することができます。

## Revision History

Rev.	Date	Description	
		Page	Summary
1.00	February 25, 2022	-	Initial release

## General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

### 1. Precaution against Electrostatic Discharge (ESD)

A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.

### 2. Processing at power-on

The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.

### 3. Input of signal during power-off state

Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.

### 4. Handling of unused pins

Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.

### 5. Clock signals

After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.

### 6. Voltage application waveform at input pin

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between  $V_{IL}$  (Max.) and  $V_{IH}$  (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between  $V_{IL}$  (Max.) and  $V_{IH}$  (Min.).

### 7. Prohibition of access to reserved addresses

Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.

### 8. Differences between products

Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

## Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

"Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.

"High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user's manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION ("Vulnerability Issues"). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user's manuals, application notes, "General Notes for Handling and Using Semiconductor Devices" in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

## Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,  
Koto-ku, Tokyo 135-0061, Japan

[www.renesas.com](http://www.renesas.com)

## Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

## Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit: [www.renesas.com/contact/](http://www.renesas.com/contact/).