

# 「E2 エミュレータ Lite 使用法」

E2エミュレータLiteはエミュレータ側からの電源供給に5Vの設定がございません。接続した際にはマイコンボードも3.3Vでご使用いただく事となります。

A/Dコンバータのリファレンスなど、VDD電圧に依存するプログラムの場合ご注意ください。

(マイコンボードに直接電源を供給する場合は5Vでのご使用が可能です)

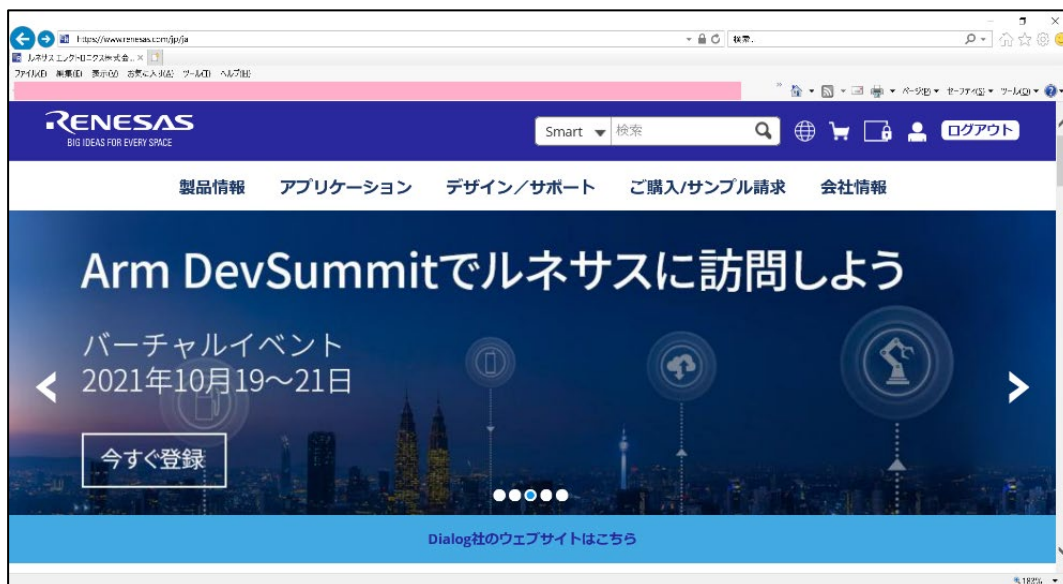
- ・統合開発環境CS+のインストール説明

E2エミュレータLiteは統合開発環境CS+でお使いいただけます。HEWでは使用できません。

RENESAS 社様ホームページより下記リンクに飛び、ファイルをダウンロードします。

(\*ダウンロードにはユーザー登録が必須となります。)

- 1) RENESAS社ホームページにアクセスします。



- 2) 検索窓に 評価版ソフトウェアツール ダウンロード製品一覧 最新版 と入力して検索。



3) 検索結果が出たらクリック。

## 検索

10 件以上見つかりました

並び替え基準 関連性 ▼ 順序 降順 ▼

評価版ソフトウェアツール ダウンロード製品一覧 最新版

Search tips: use quotes to search for specific strings (e.g. "wireless power") and regular expressions to exclude terms with +/- (e.g. wireless power +receiver, or power -wireless)

検索 ▶

↓

評価版ソフトウェアツール ダウンロード製品一覧 最新版

基本ページ - 2021年8月25日

4) このページに飛びます。

RENESAS BIG IDEAS FOR EVERY SPACE
Smart 検索 
🌐 🛒 📁 👤 ログアウト

製品情報
アプリケーション
デザイン/サポート
ご購入/サンプル請求
会社情報

デザイン/サポート > 評価版ソフトウェアツール ダウンロード製品一覧 最新版

## 評価版ソフトウェアツール ダウンロード製品一覧 最新版

[評価ソフトウェアツールとは](#) | 
 [CS+](#) | 
 [e<sup>2</sup> studio](#) | 
 [コンパイラ/アセンブラ](#) | 
 [シミュレータデバッグ](#) | 
 [SQMint](#) | 
 [フラッシュ書き込みツール](#)

### 評価版ソフトウェアツールとは

ソフトウェアツール製品のご購入前に、製品の機能や性能を評価するために無償でご利用いただけます。試用期限、機能、性能およびサービス\* において製品版と異なる場合があります。各評価版ソフトウェアツール製品の詳細は以下の「評価版ソフトウェアツール詳細一覧」をご覧ください。

### 評価版ソフトウェアツールの入手方法

このページの「評価版ダウンロード」または「ダウンロード」をクリックすると現在配布している評価版ソフトウェアツールが一覧表示されます。そこからダウンロードしてください。

5) 下にスクロールしてこの画面から「評価版ダウンロード」をクリック。

### 統合開発環境 CS+ (旧CubeSuite+)

製品名	仕様・性能	試用期限
統合開発環境 CS+ for CC (RL78, RX, RH850用)	・ 試用期限内は製品版(professional版)と同じ。試用期限を過ぎると各MCUにより以下の制限があります。 [RH850ファミリ] リンクサイズを256Kバイト以内に制限しています。 professional版の機能は使用できません。	60日
	[RXファミリ] リンクサイズを128Kバイト以内に制限しています。 professional版の機能は使用できません。	初めて評価版ソフトウェアツールをインストールした後、最初にビルドを行った日から60日間の試用期間があります。
<span style="border: 2px solid red; border-radius: 50%; padding: 2px;">製品ページ</span> <span style="border: 2px solid red; border-radius: 50%; padding: 2px;">評価版ダウンロード</span>	[RHファミリ] リンクサイズを64Kバイト以内に制限しています。 professional版の機能は使用できません。	試用期間内は、機能に制限はありません。  61日目以降は、リンクサイズ、professional版の機能が制限されます。
・ コンパイラ※、デバッグを同梱。 ・ ※ CC-RL, CC-RX, CC-RH		

- 6) 【無償評価版】統合開発環境 CS+ for CC V8.06.00(一括ダウンロード版)をクリック。  
 (「V8.06.00」の部分は最新の物に変更されている場合があります)

↓ ダウンロード

Title, start typii	すべてのタイプ	形式	サイズ	日付
【無償評価版】統合開発環境 CS+ for CA,CX V4.06.00(一括ダウンロード版)	English	ソフトウェア/ツール 評価版ソフトウェア	ZIP 482.72 MB	2021年7月19日
【無償評価版】統合開発環境 CS+ for CC V8.06.00(一括ダウンロード版)	English	ソフトウェア/ツール 評価版ソフトウェア	ZIP 728.33 MB	2021年7月19日
【無償評価版】統合開発環境 CS+ for CA,CX V4.05.00(一括ダウンロード版)	English	ソフトウェア/ツール 評価版ソフトウェア	ZIP 483.78 MB	2021年1月20日
【無償評価版】統合開発環境 CS+ for CC V8.05.00(一括ダウンロード版)	English	ソフトウェア/ツール 評価版ソフトウェア	ZIP 719.07 MB	2021年1月20日

- 7) 契約ページで、文章に同意する場合「同意します」をクリック。

RENASAS  
BIG IDEAS FOR EVERY SPACE

Smart 検索

製品情報 アプリケーション デザイン/サポート ご購入/サンプル請求 会社情報

### 【無償評価版】統合開発環境 CS+ for CC V8.06.00 (一括ダウンロード版)

お客様が「同意します」ボタンもしくはDisclaimer8-JPN(以下、「本契約」といいます)の電子コピーの契約条件に同意することを確認するために設計されたその他のボタンもしくはメカニズムをクリックし、または本契約のライセンス許諾対象のソフトウェア(以下、「本ソフトウェア」といいます。)の全部もしくは一部をダウンロード、インストール、アクセスもしくはその他の手段により複製もしくは使用することで、(a)お客様は、お客様が権限を有する被許諾者(以下、「ライセンス」といいます。)を代理または代表して本契約を締結し、それによりライセンスが本契約に法的に拘束されることを承諾の上、本契約を締結する意思表示を行ったこととなり、また、(b)お客様はライセンスを代理または代表し、ライセンスを拘束する権利、権能および権限を有することを表明かつ保証したことになります。

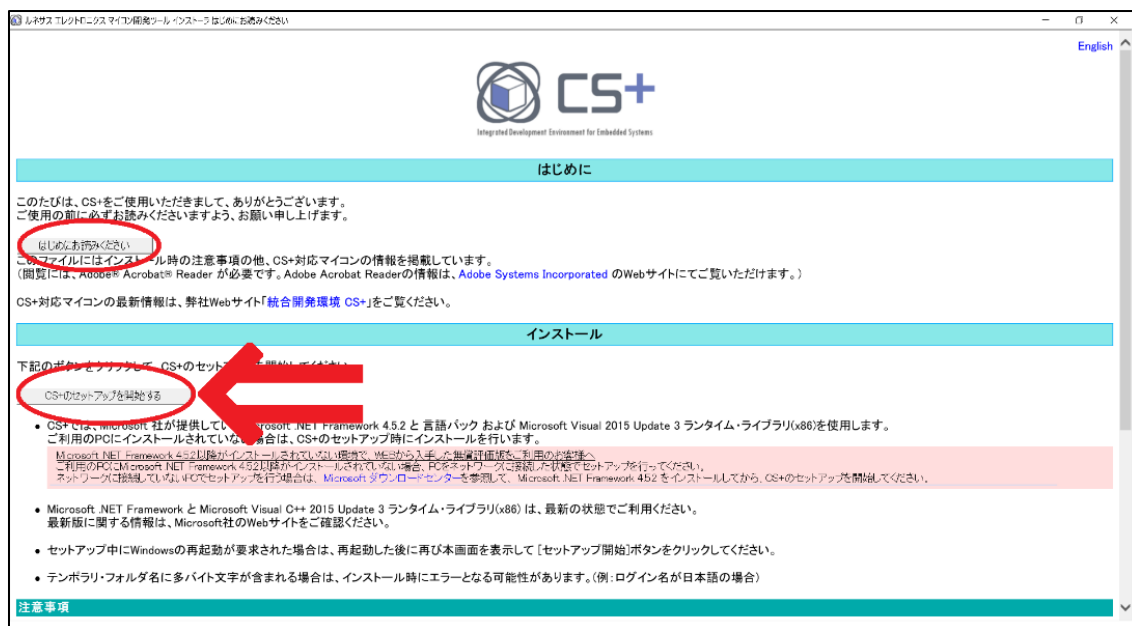
ライセンスが本契約上の契約条件に同意しない場合またはお客様がライセンスを代理もしくは代表して本契約を締結し、ライセンスを拘束する権利、権能および権限を有しない場合、「同意します」ボタンまたは本契約に同意することを確認するために設計されたその他のボタンもしくはメカニズムを選択せず、かつ本ソフトウェアの全部または一部をダウンロード、インストール、アクセスまたはその他の手段により複製もしくは使用しないでください。当社は、本契約に従う限りにおいて、ライセンスに対し、本ソフトウェア(その機能は機構を含みます)をダウンロード、インストール、アクセスまたはその他の手段により複製もしくは使用することを許諾します。

- 8) 同意すると「CSPlus\_CC\_Package\_V8000-doc-j.zip」のダウンロードが始まります。  
 ダウンロードした zip ファイルをクリックすると下記のファイルが解凍されます。

名前	更新日時	種類	サイズ
CSPlus_CC_Package_V80600.EXE	2021/06/18 12:17	アプリケーション	745,850 KB
CSPlus_CC_Package_V80600_readme_J.txt	2021/07/06 16:04	テキスト文書	6 KB

9) CSplus\_CC\_Package\_V80600.eEXE をクリックすると、インストールが始まります。

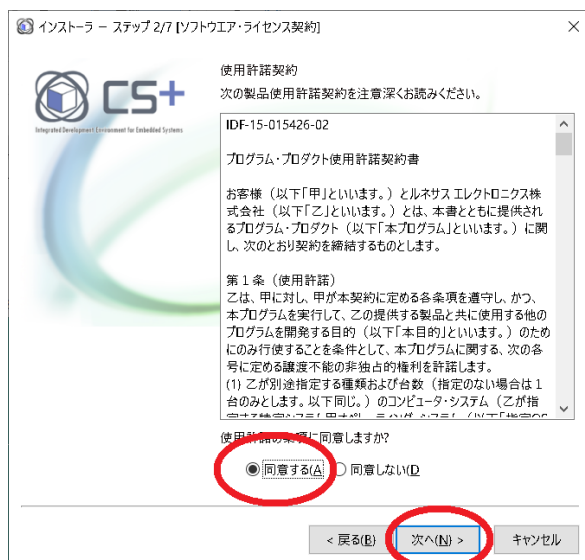
「はじめにお読みください」をクリックして説明を読んだ後、「CS+のセットアップを開始する」をクリックしてセットアップを開始します。



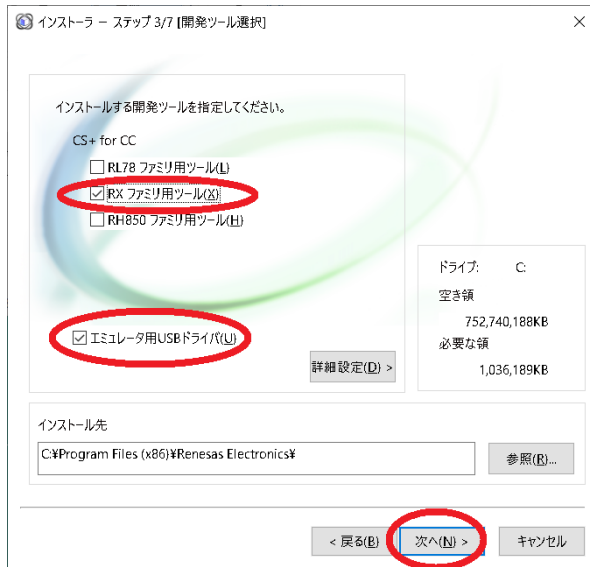
10) セットアップを開始しますので「次へ」をクリック。



11) 使用許諾契約を読んだ後、「同意する」にチェックを入れて「次へ」をクリック。



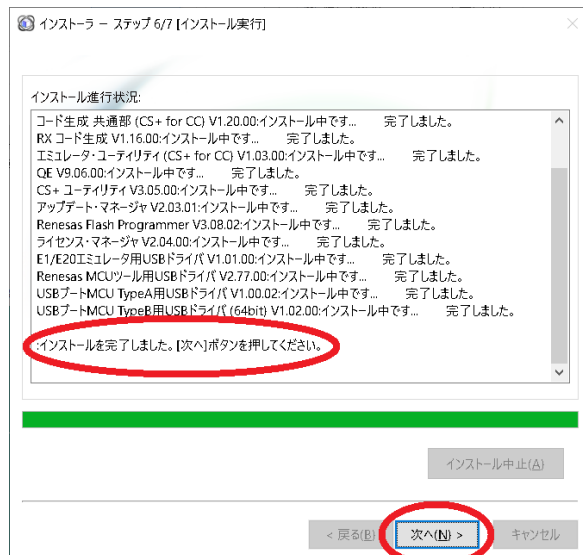
1 2) 開発ツール選択で「RX ファミリ用ツール」「エミュレータ用 USB ドライバ」にチェックを入れて「次へ」をクリック。



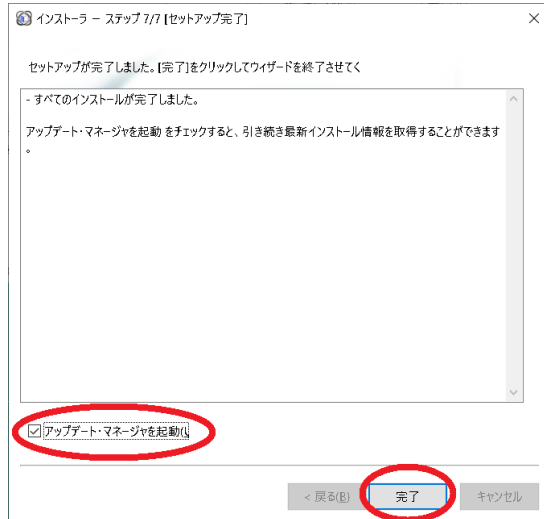
1 3) インストール設定を確認して「次へ」をクリック。



1 4) インストールが完了したら、確認して「次へ」をクリック。

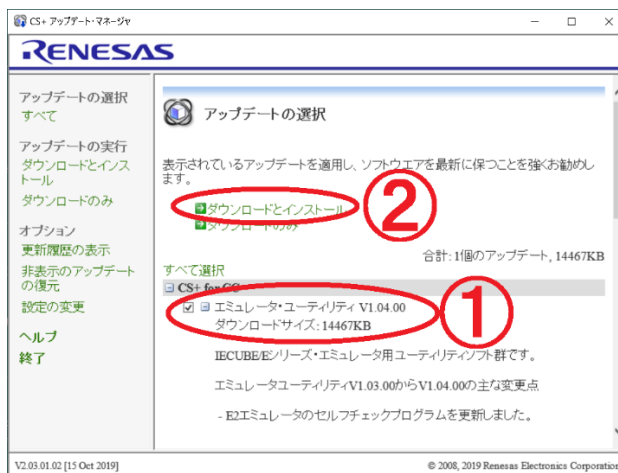


15) セットアップが完了したのでアップデートを確認します。「アップデート・マネージャを起動」にチェックを入れて「完了」をクリック

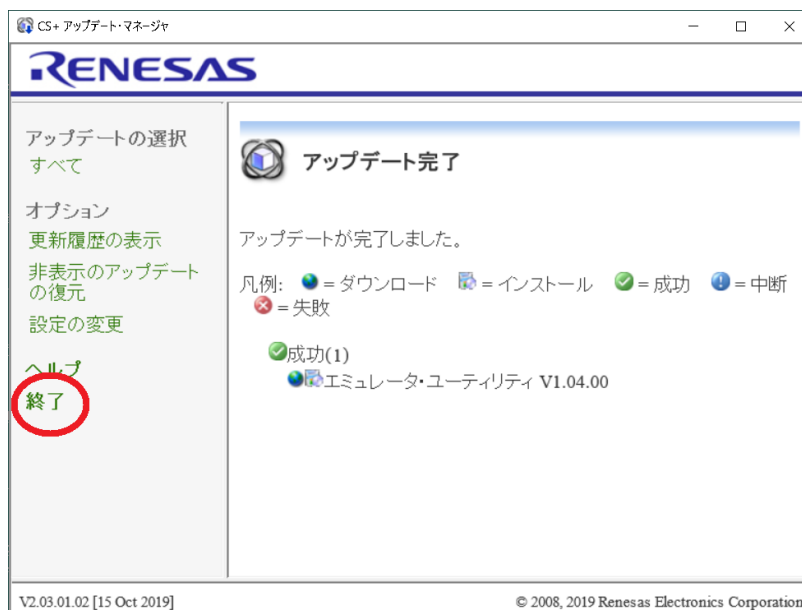


16) アップデート・マネージャが起動するので①「エミュレータ・ユーティリティ V1.04.00」にチェックを入れて②「ダウンロードとインストール」をクリック。

(「V1.04.00」の数字はダウンロード時の最新の物になっています。)



17) アップデート完了を確認して「終了」をクリック。



18) スタートメニューからインストール完了を確認します。



以上でソフトウェアのインストールは終了です。

続いてコンパイルに移ります。

「E2 エミュレータ Lite コンパイル」  
にお進みください。

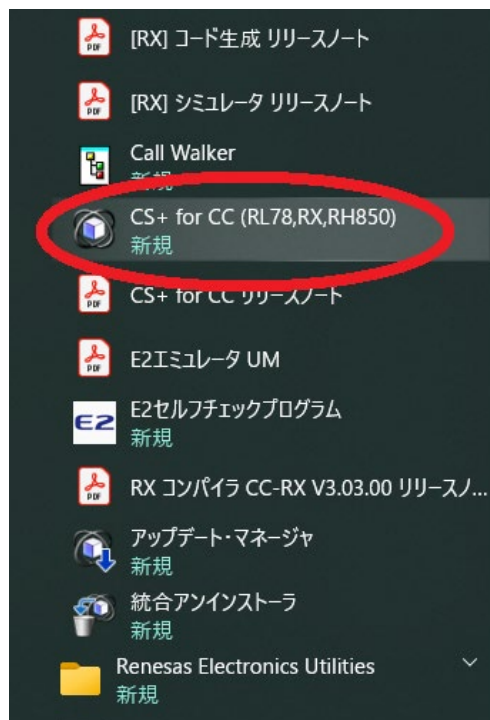
# 「E2 エミュレータ Lite コンパイル」

## ●C言語でのサンプルソフトのコンパイル例説明

本CDの「サンプル」フォルダ内の「TEST\_LED.c」が、Cコンパイラ用サンプルソースファイルです。

## ■TEST\_LED.cのコンパイル例■

1) フォルダ「Renesas Electronics CS+」中の「CS+ for CC (RL78,RX,RH850)」をクリックして開始します。

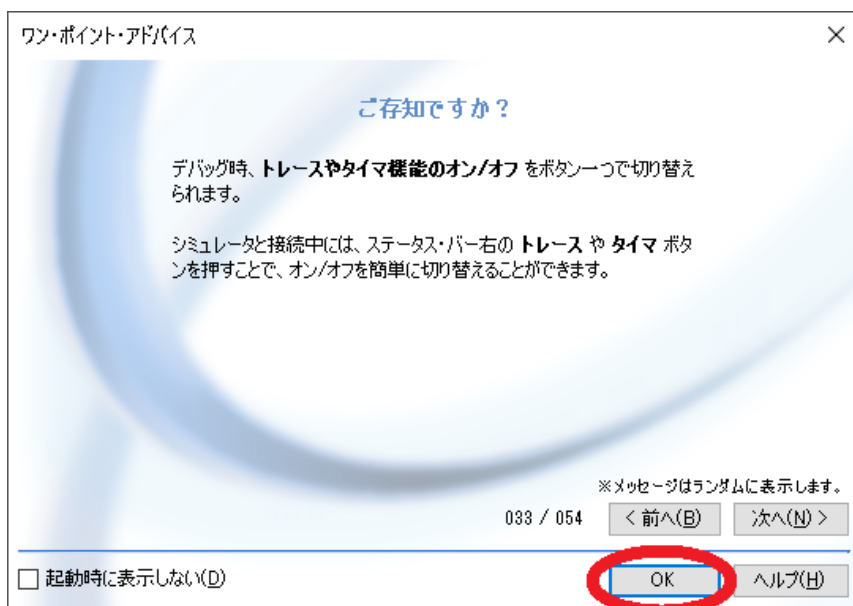


2) こちらが開始画面です。

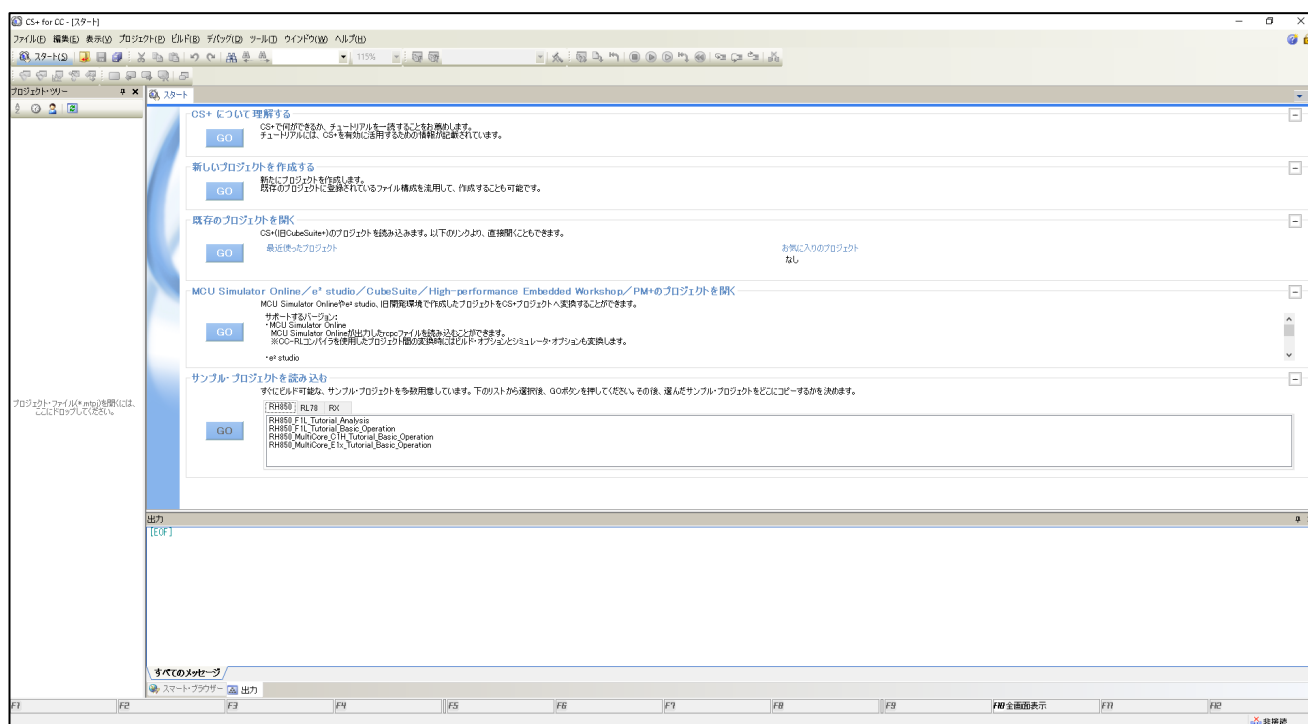




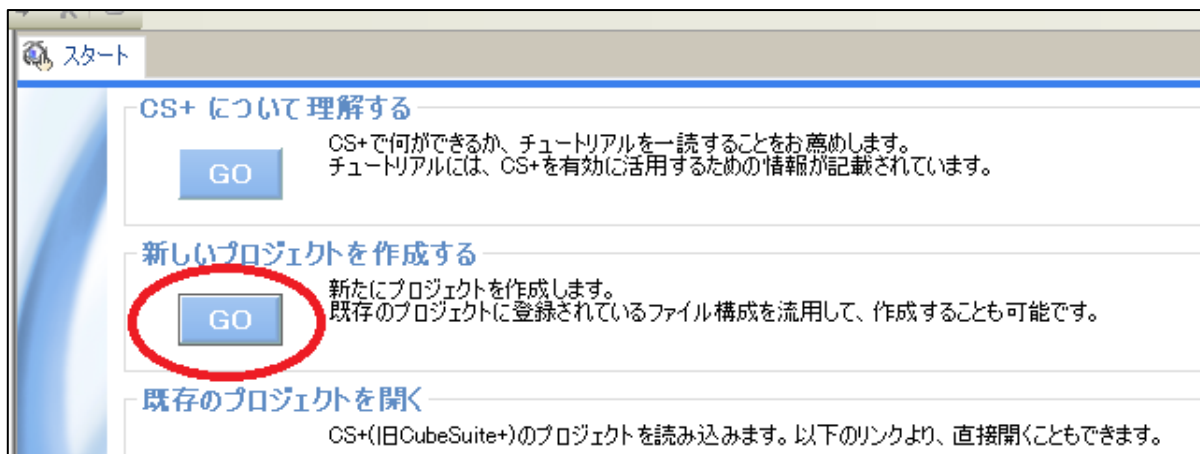
3) 次に現れる「ワンポイント・アドバイス」にはこれから使用するに当たって役立つ情報が記載されています。今回はとりあえず「OK」をクリックして開発を開始します。



4) CS+が立ち上がると下の画面が出ます。

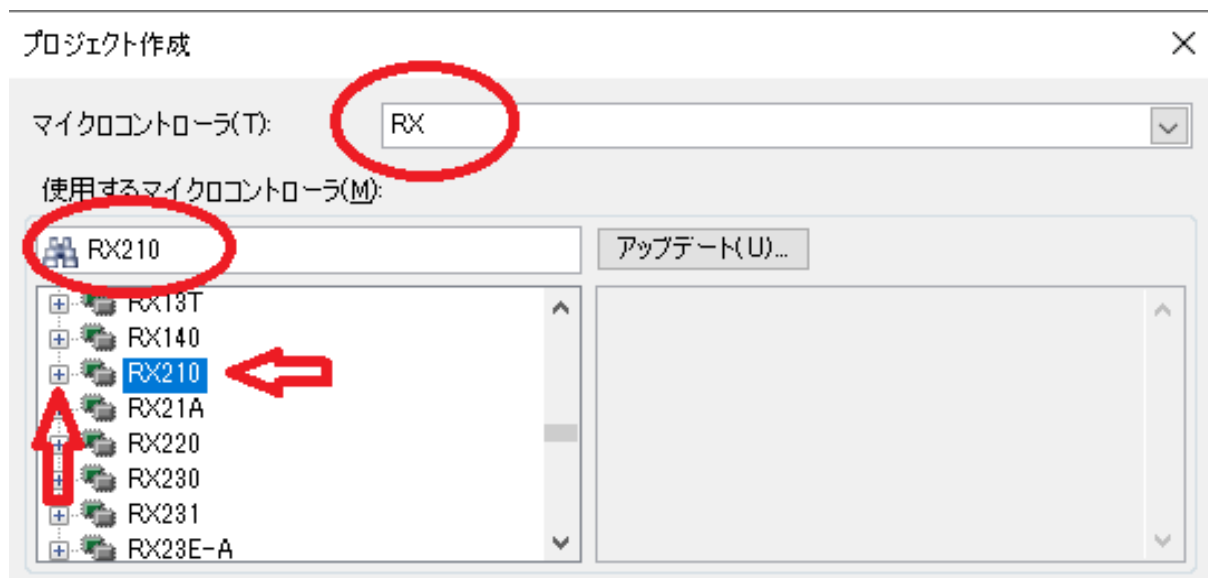


5) プロジェクトを作成します。「新しいプロジェクトを作成する」の「GO」をクリックします。



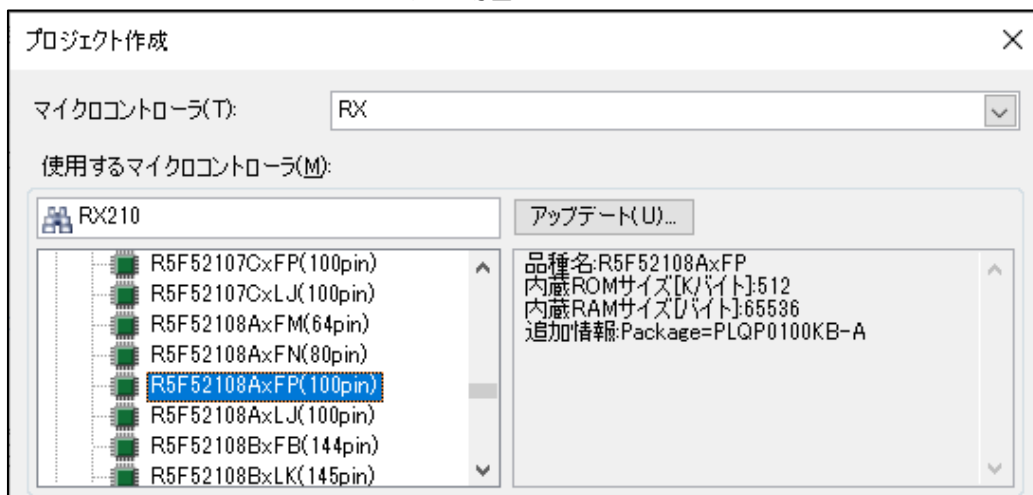
6) 「プロジェクト作成」画面でマイクロコントローラ（CPU）を選択します。

「マイクロコントローラ（T）」にRX、「使用するマイクロコントローラ（M）」にRX210を入力、表示されたRX210の左の田マークをクリックします。

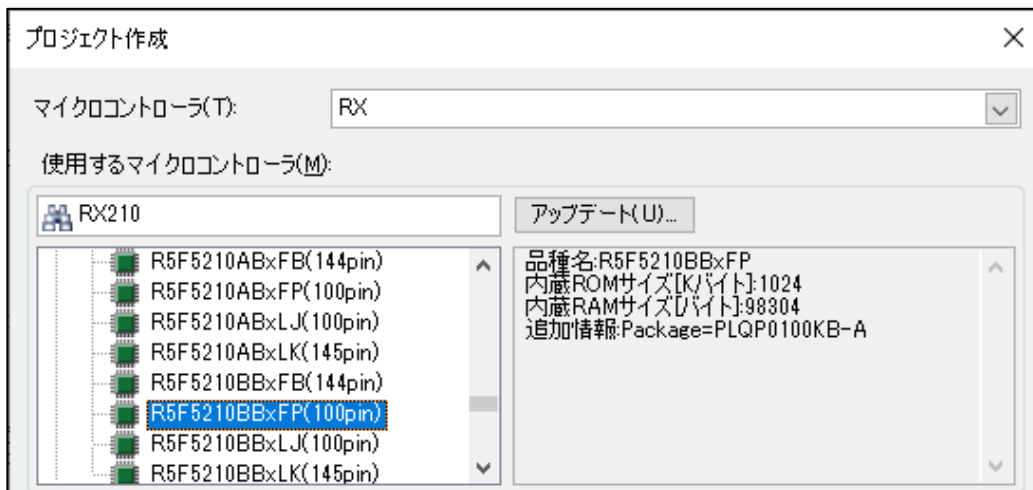


7) マイコンボードに使用しているマイクロコントローラを指定します。

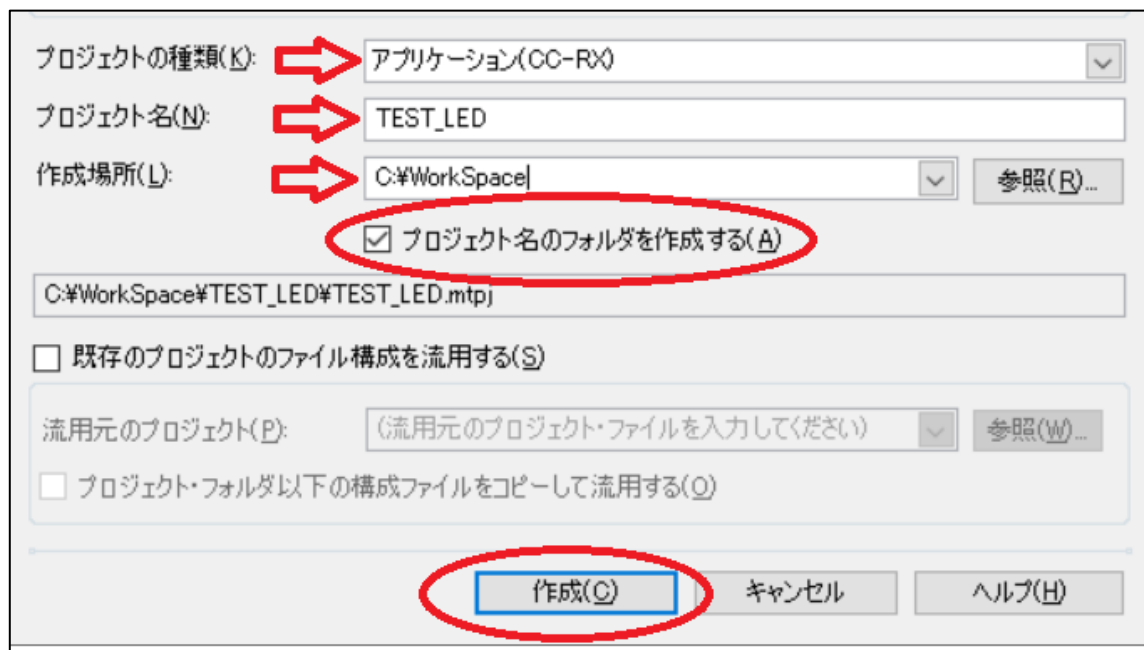
R5F52108ADFPボードの場合



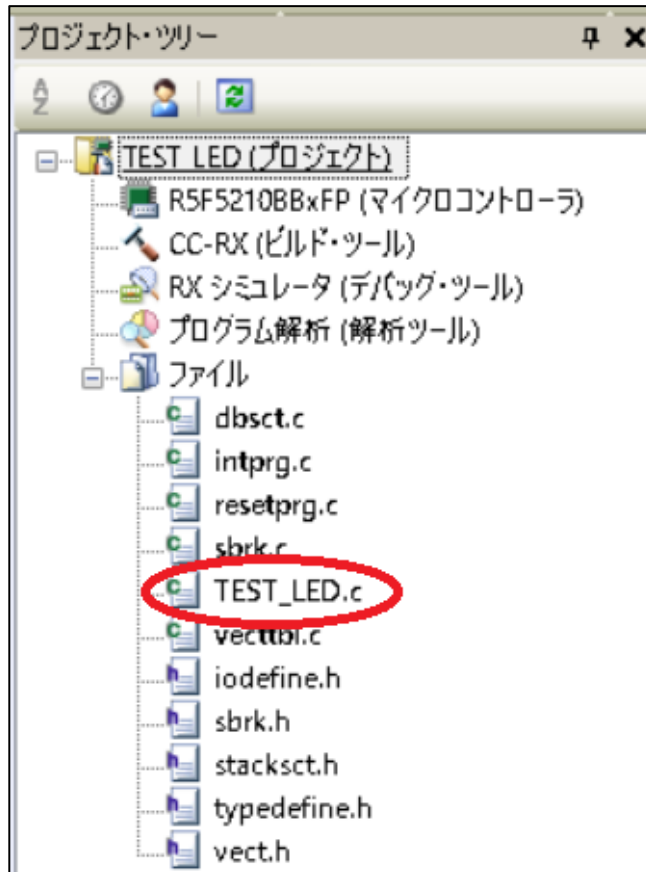
## R5F5210BDFPボードの場合



- 8) 「プロジェクト作成」画面の下部でプロジェクトの種類、プロジェクト名、作成場所を指定し、「作成」をクリックします。ここではプロジェクトの種類を「アプリケーション(CC-RX)」プロジェクト名は「TEST\_LED」作成場所を「C:\¥Workspace」として「プロジェクト名のフォルダを作成する」にチェックをします。



- 9) 左上のプロジェクトツリーを確認すると、「ファイル」の中に「TEST\_LED.c」が生成されています。



- 10) TEST\_LED.c をダブルクリックすると、下図のようなデフォルトの雛形が現れますが、今回はキット付属の CD-ROM 内のサンプルを使用しますので全て消去します。

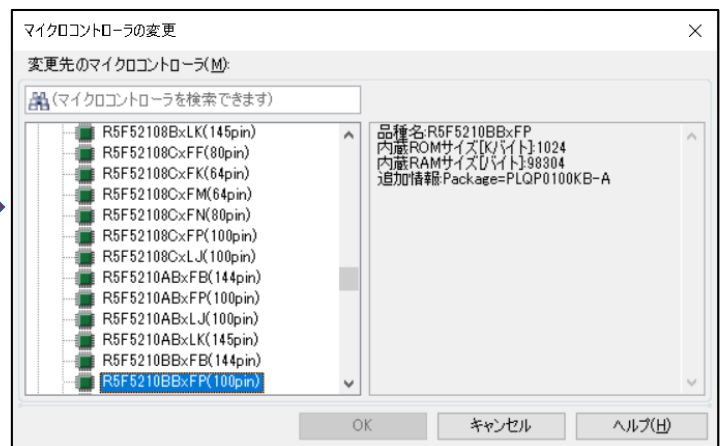
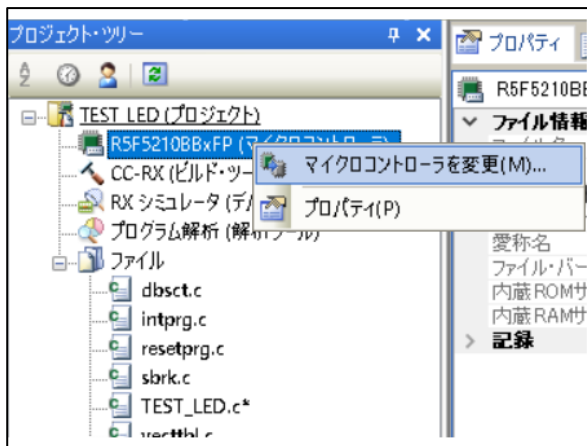
(画面内にカーソルを移動させ C t r l - A で全てを選択してから D E L キーで消去してください。)

- 1 1) 「TEST\_LED.c」を編集して、ソフトを作成しますが、キット添付CD（このCDです）の「サンプル」フォルダに「TEST\_LED.c」がありますので、表示されているソースを削除し、「TEST\_LED.c」をWindows 付属ソフトのメモ帳などで開き、内容をコピーします。  
 下図は、コピー後の状態です。（ファイルの先頭を表示させています）

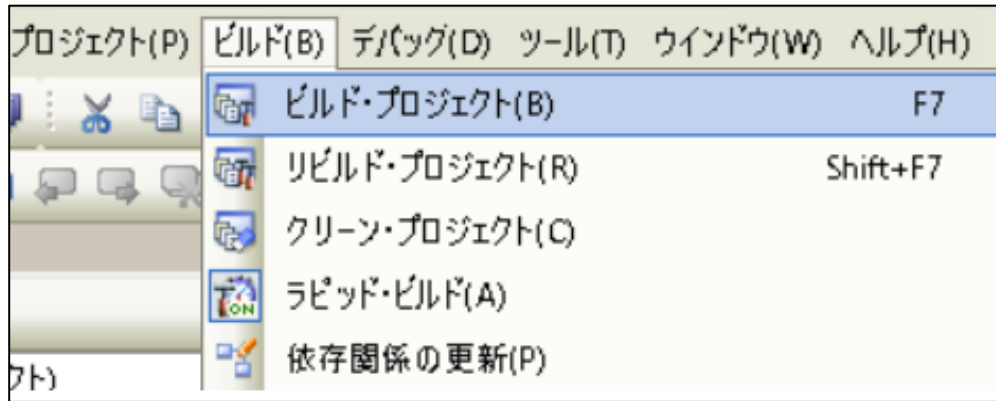
```

1  /******
2  /*
3  /* FILE      :TEST_LED.c
4  /* DATE      :Wed, Jan 29, 2014
5  /* DESCRIPTION :Main Program
6  /* CPU TYPE   :RX210
7  /*
8  /* This file is generated by Renesas Project Generator (Ver.4.53).
9  /* NOTE:THIS IS A TYPICAL EXAMPLE.
10 /*
11 /******
12
13
14 #include "iodefine.h"
15 #define LED_PORT PORT1.POODR.BIT.B5
16 #define DELAY_VALUE 50000
17
18 /* 関数のプロトタイプ宣言 */
19 void main(void);
20 void PORT_INITIALIZE(void);
21 void INTERVAL_DELAY(void);
22
23 /* メイン関数 */
24 void main(void)
25 {
26     PORT_INITIALIZE(); //全ポート出力 全ポートLOW
27     while(1) //繰り返し開始
28     {
29         LED_PORT = 0; //点：VDDに接続されているので0で点灯
30         INTERVAL_DELAY();
31         LED_PORT = 1; //滅：VDDに接続されているので1で滅灯
32         INTERVAL_DELAY();
33     }
34 }
  
```

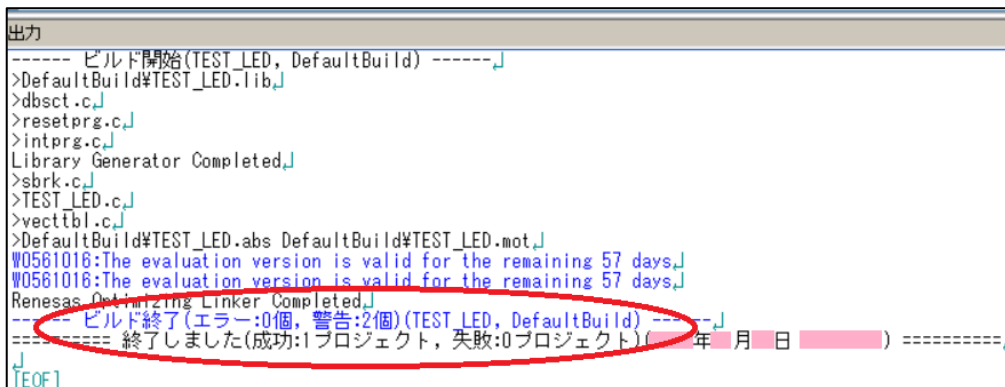
- 1 2) マイコンの指定を間違えていた場合などは下記で修正します。7) をご参照ください。



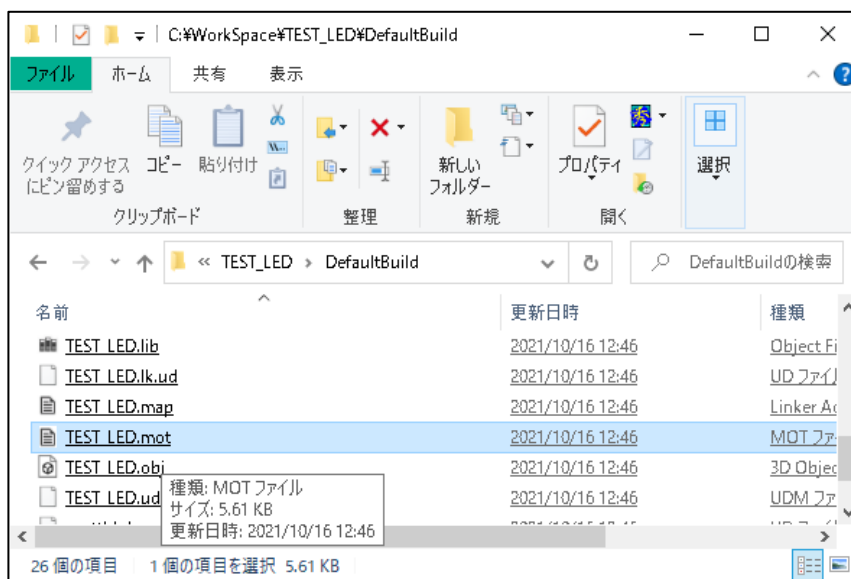
- 1 3) プロジェクトをビルドします。(ソースコード「TEST\_LED.c」を実行可能なファイル形式にします)  
ツールバー「ビルド (B)」⇒「ビルド・プロジェクト (B)」をクリックします。



- 1 4) 正常に終了すると画面下部の「出力」窓にメッセージが表示されます。  
(かかる時間はパソコンの性能により異なります)

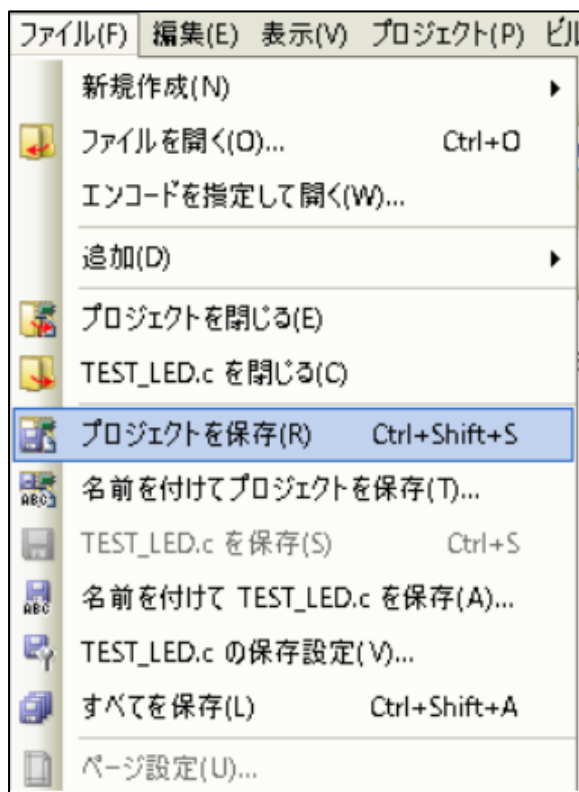


- 1 5) 実行ファイルが生成されている事を確認します。8) で指定したディレクトリ (フォルダ) に  
「TEST\_LED.mot」ができている事を確認してください。



16) 一旦ここでプロジェクトを保存しておきましょう。

ツールバー「ファイル(F)」⇒「プロジェクトを保存(R)」で保存します。



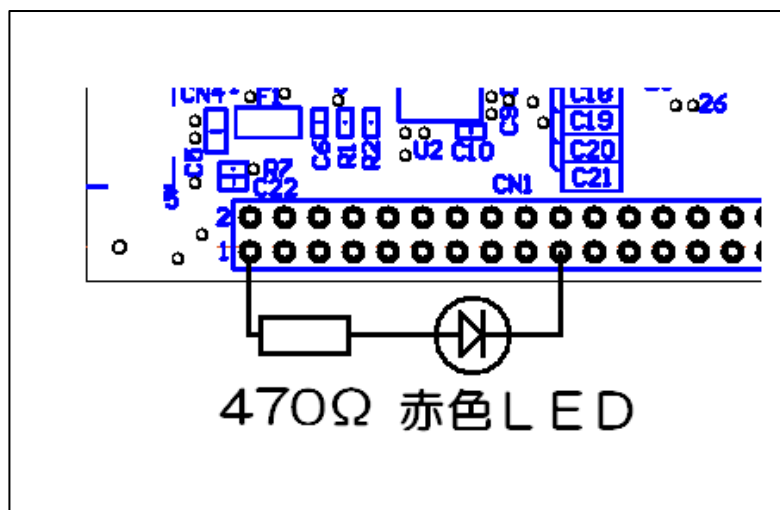
次はE2エミュレータLiteを使ったデバッグに移ります。

「E2 エミュレータ Lite デバッグ」にお進みください。

# 「E2 エミュレータ Lite デバッグ」

## ■デバッグ作業確認用テストLED回路の接続■（470Ω抵抗と赤色LEDは別途ご用意ください）

CN1の1番ピン（VDD）と19番ピン（PORT 1, BIT 5）に470Ω抵抗と赤色LEDを直列に接続します。1番ピンがアノード側、19番ピンがカソード側です。



## ■E2エミュレータLite（以下E2Liteと表記）との接続■

以下の説明では電源をE2Liteから供給するものとして説明いたします。

基板上のSW1（スライドスイッチ）は必ず右下画像内矢印の方向（基板内側向き）にスライドさせておいてください。

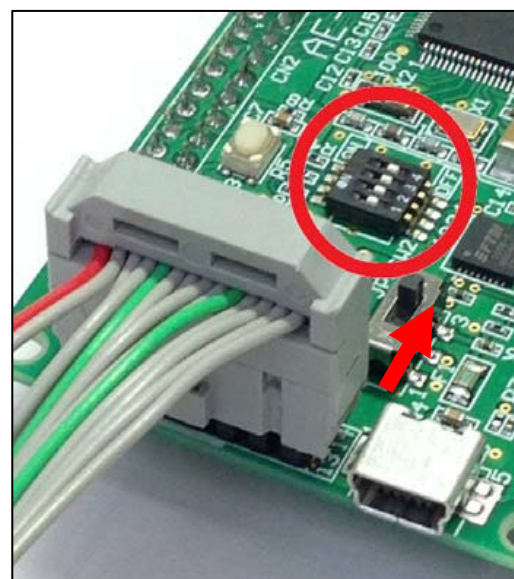
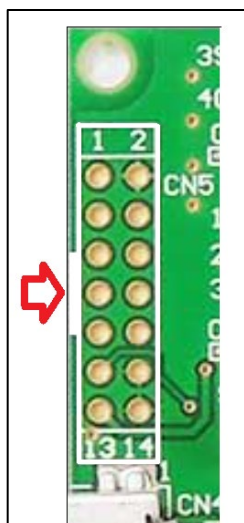
E2LiteはUSBケーブルでパソコンと接続しておきます。

### 1) E2LiteをUSB接続し、マイコンボードと

フラットケーブルで接続します。

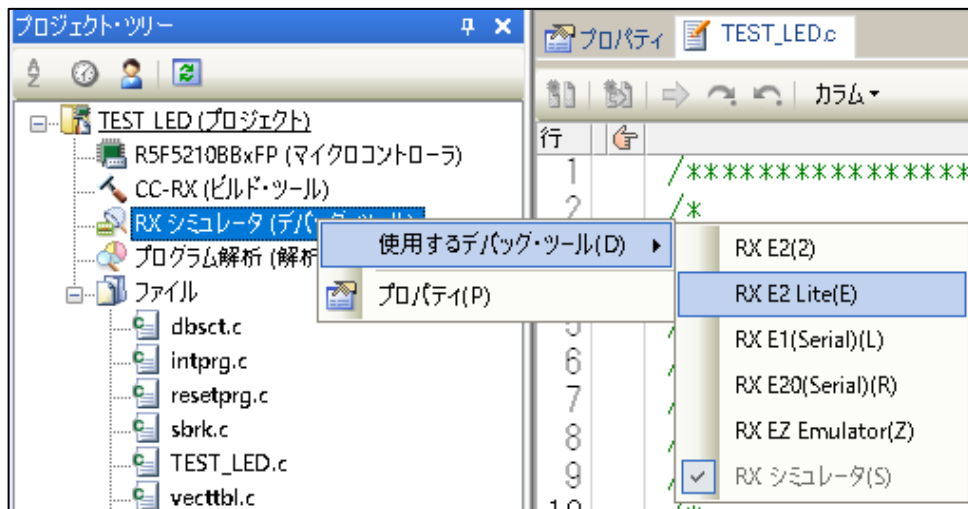
基板の凸印にコネクタを合わせてください。

SW2は1～4全てOFFにします。

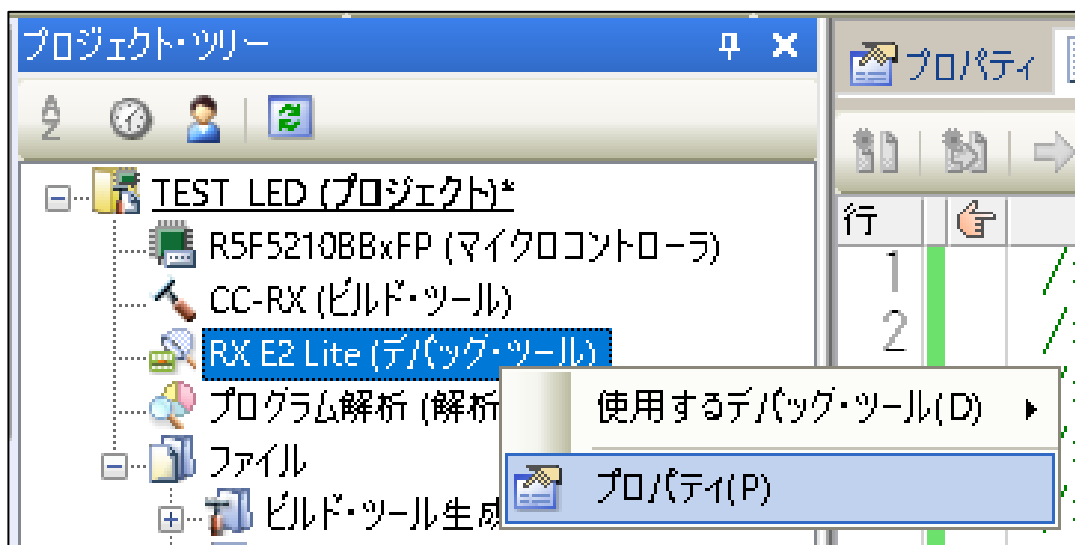




- 2) CS+は「コンパイル編」からの続きで、既に立ち上がっている状態からの説明をいたします。  
デバッグ・ツールにE2Liteを指定します。  
プロジェクト・ツリーから「RXシミュレータ(デバッグ・ツール)」を右クリック、  
「使用するデバッグ・ツール(D)」から「RX E2Lite(E)」を指定します。



- 3) E2Liteのプロパティを設定します。  
プロジェクト・ツリーの「RX E2Lite(デバッグ・ツール)」を右クリックして表示された  
「プロパティ(P)」をクリックします。

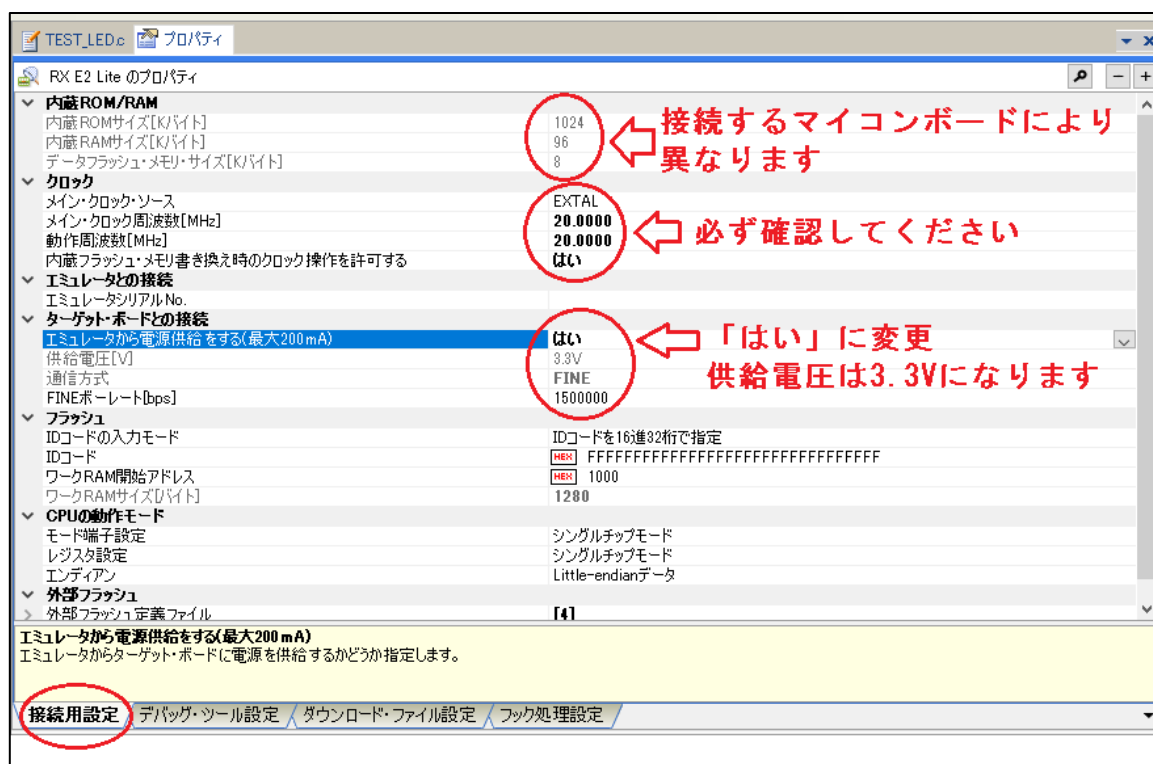


#### 4) プロパティ画面が表示されます。

左下の「接続用設定」タブの画面です。

- 「クロック」内「メイン・クロック・ソース」はEXTAL、
- 「メイン・クロック周波数[MHz]」=20（入力すると 20.0000 と表示されます）
- 「動作周波数[MHz]」=20（入力すると 20.0000 と表示されます）
- 「内蔵フラッシュ・メモリ書き換え時のクロック操作を許可する」枠内をクリックして現れる右端の「V」マークをクリックし「はい」に変更
- 「エミュレータから電源供給をする（最大 200mA）」枠内をクリックして現れる右端の「V」マークをクリックして「はい」に変更
- 「供給電圧」が3.3Vである事を確認

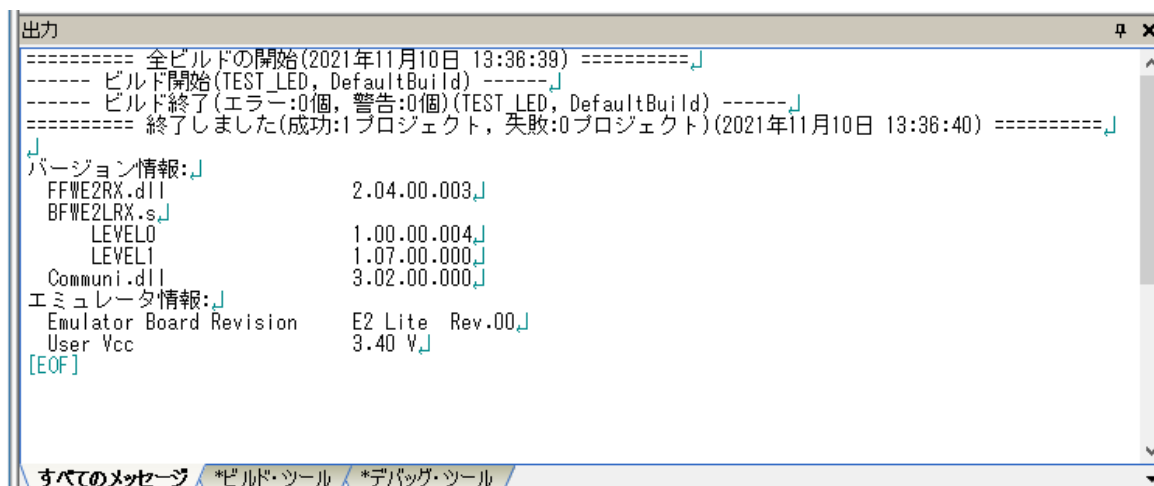
以上を確認し、違っている場合は書き換えてください。



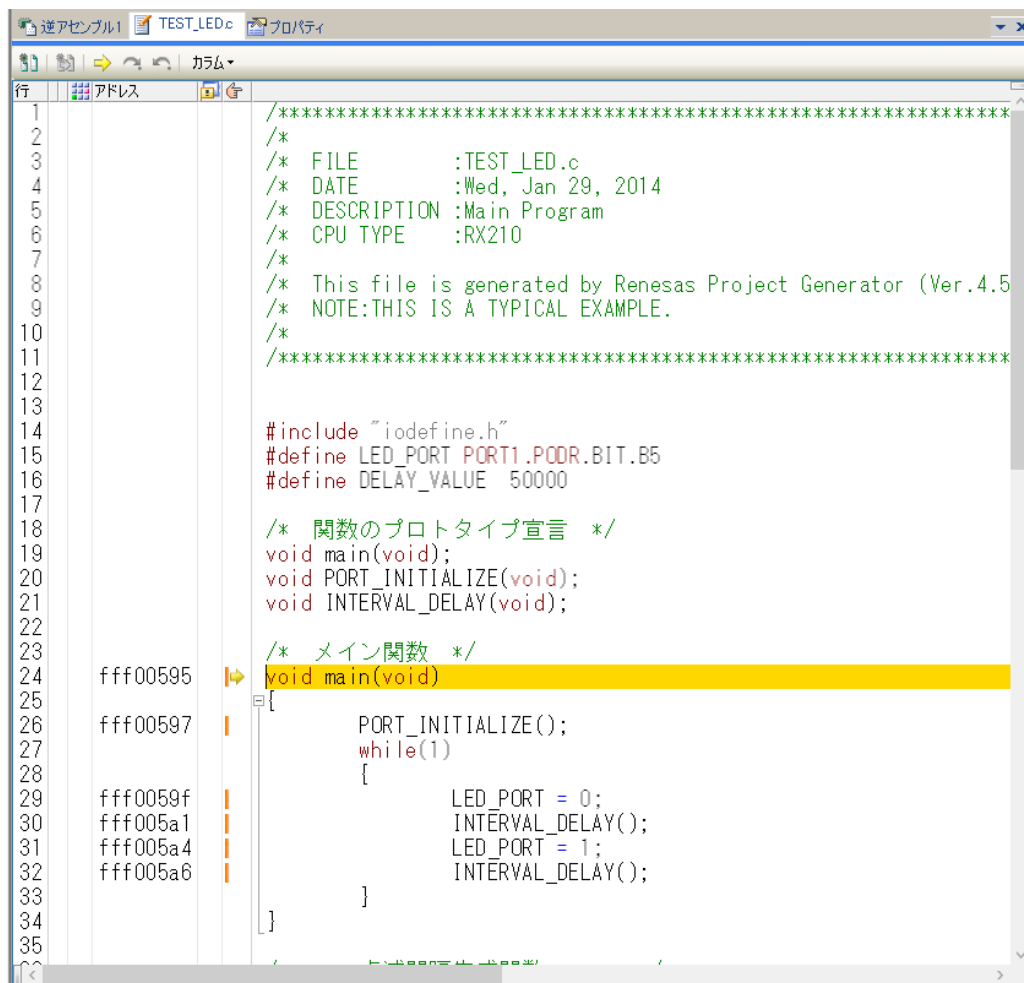
#### 5) ビルドとデバッグ

ツールバー「デバッグ (D)」⇒「ビルド & デバッグ・ツールヘダ ヲロード (B)」をクリック

「出力」画面には下記の様に表示されます。

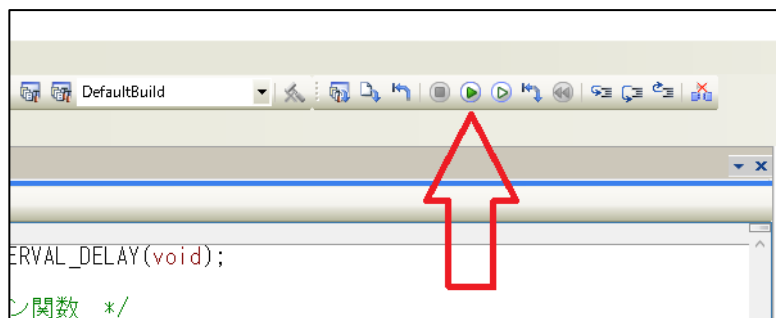


6) 「TEST\_LED.c」のタブの横には「逆アセンブル」タブが現れます。

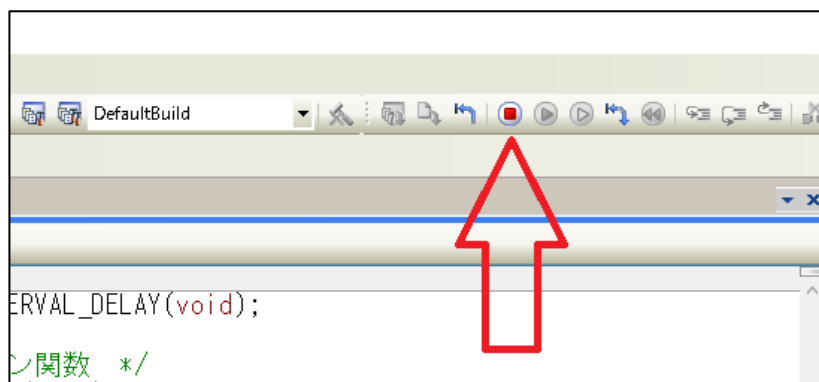


7) ダウンロードが完了するとプログラム実行可能になります。

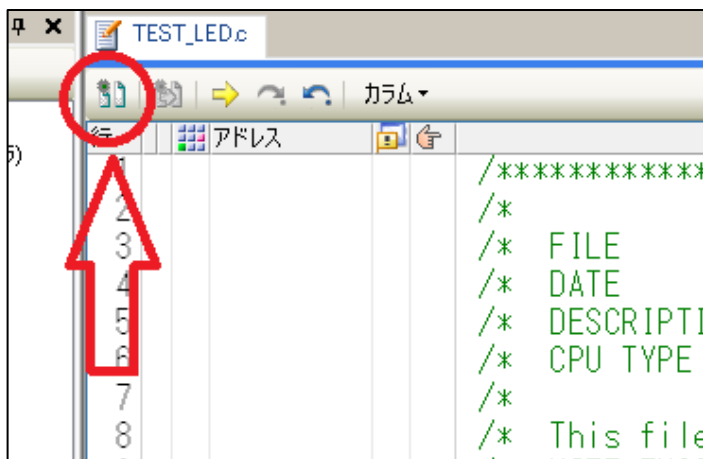
ツールバー下の「横向き▲ (プログラムを現在の位置から実行します)」をクリックするとプログラムが実行されます。



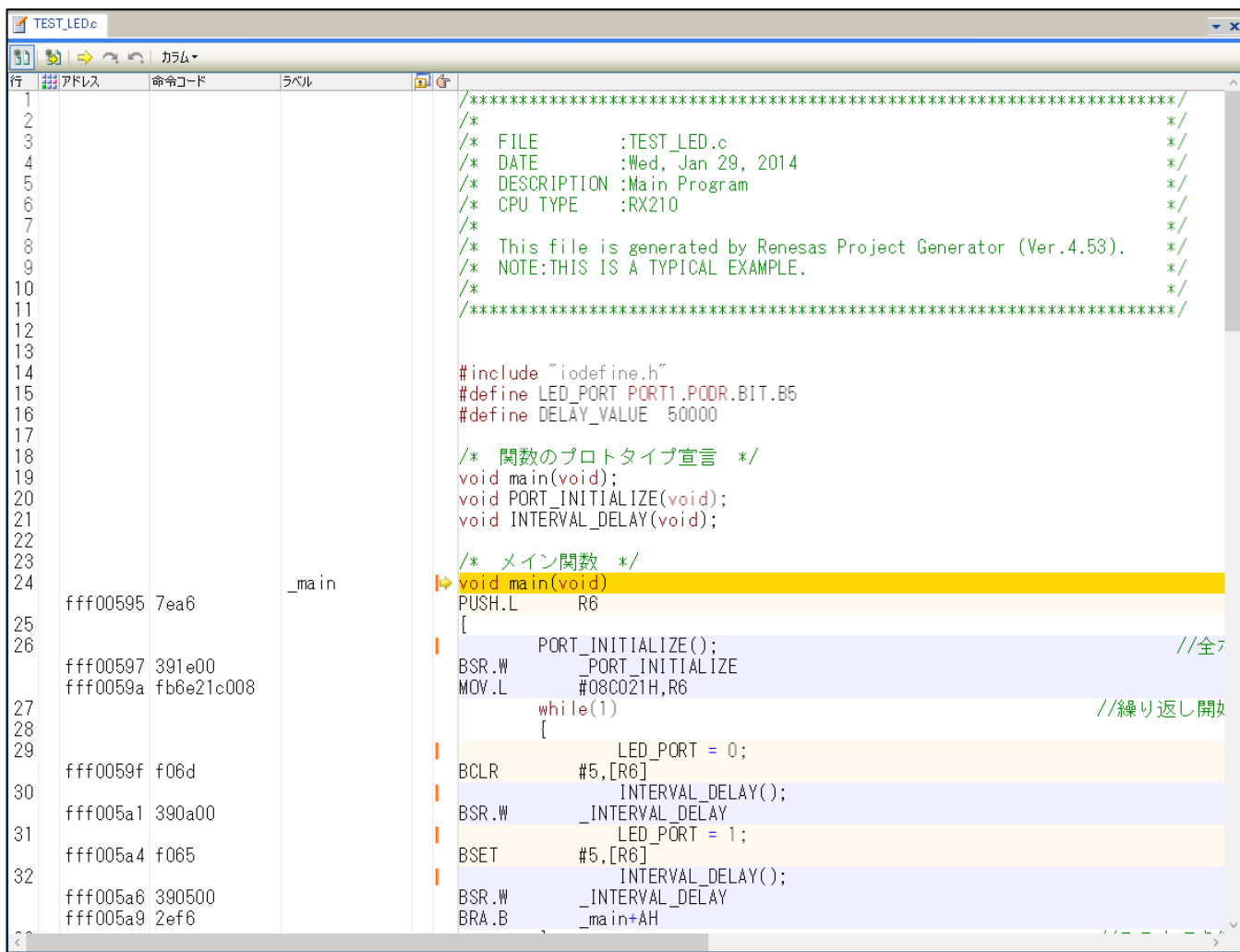
8) 実行中にツールバー下の「■ (実行中のプログラムを停止します)」をクリックすると停止します。



9) デバッグを行います。「TEST\_LED.c」タブの下の（逆アセンブルとソースを混合して表示します）ボタンをクリックして双方を表示させます。



10) 逆アセンブルとソースが同時表示されます。



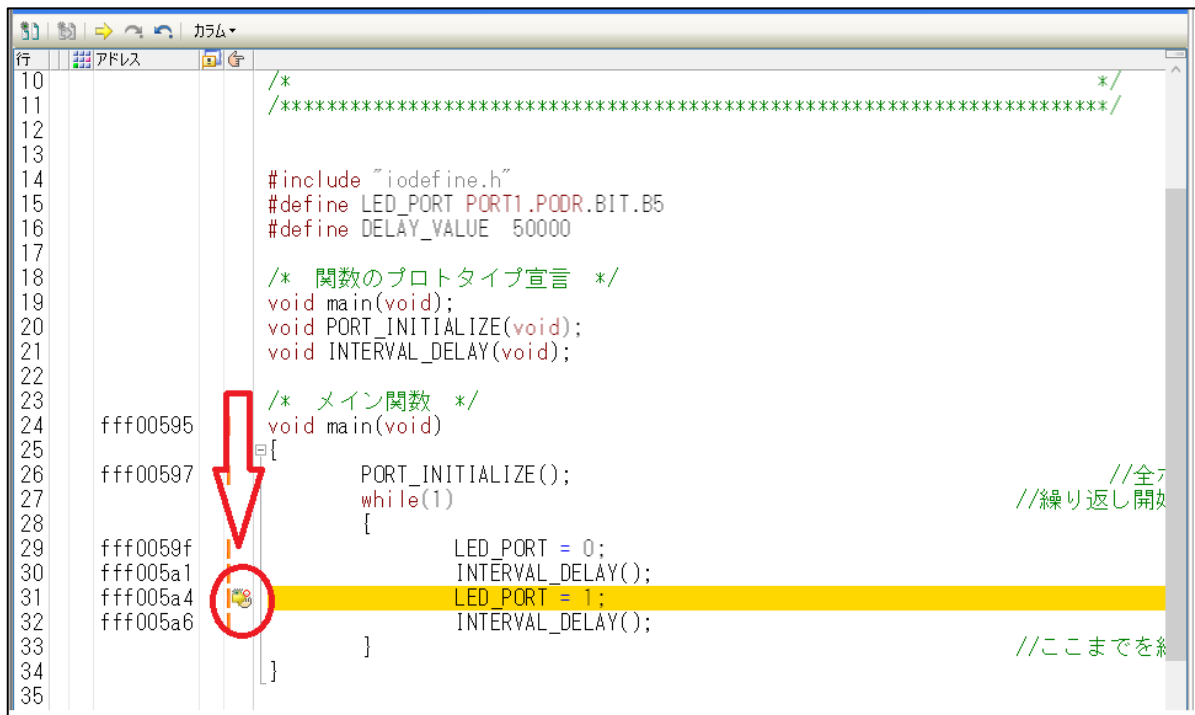
## ■ブレークポイントの設定■

ソースファイル左のブレークポイント指定欄にカーソルを置くとブレークポイントの印が現れます。その直前までを実行して停止させる事ができます。

下の図の例では、31行目の「LED\_PORT = 1;」にブレークポイントを置いています。

緑の横向き▲の（プログラムを現在の位置から実行します。）をクリックすると、直前30行目の「INTERVAL\_DELAY();」関数を実行終了した時点で停止し、LEDは点灯したままになります。

もう一度同じく緑の横向き▲をクリックすると、31行目を実行（LED消灯）、32行目を実行（2秒待ち）した後に29行目（LED点灯）に戻り、再び30行目を実行終了した時点で停止し、LEDは点灯したままになります。

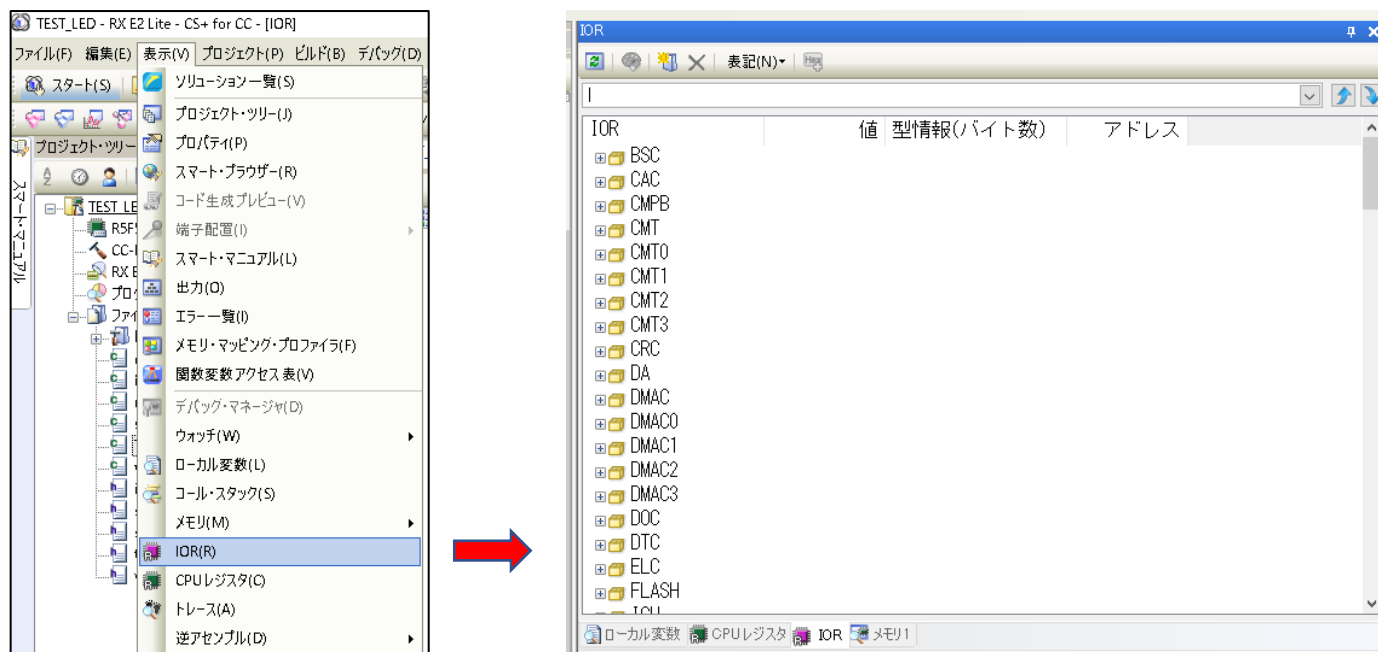


```
10 /*
11  */
12
13
14 #include "iodefine.h"
15 #define LED_PORT PORT1.PODR.BIT.B5
16 #define DELAY_VALUE 50000
17
18 /* 関数のプロトタイプ宣言 */
19 void main(void);
20 void PORT_INITIALIZE(void);
21 void INTERVAL_DELAY(void);
22
23 /* メイン関数 */
24 void main(void)
25 {
26     PORT_INITIALIZE();
27     while(1) //繰り返し開始
28     {
29         LED_PORT = 0;
30         INTERVAL_DELAY();
31         LED_PORT = 1;
32         INTERVAL_DELAY(); //ここまでを繰り返す
33     }
34 }
35
```

## ■I/Oの確認・変更■

1) ツールバーの「表示」 - 「I/O(R)」で、レジスタの内容が表示されます。

ソースコードの隣に表示されますが、位置や幅は見やすい状態に調節してください。



2) 「PORT1」 - 「PODR」の「B5」が LEDのポートです。(PORT1.PODR.B5)

縦スクロールバーで「PORT1」まで移動し、左の田印をクリックすると、PORT1の各レジスタが表示されます。「PORT1.PODR.B5」桁の「値」をダブルクリックして値を変更Enterキーで確定してください。

LEDは抵抗を介してVDDに接続されている為、0で点灯、1で消灯です。

IOR	値	型情報(バ...	アドレス
PORT1			
PORT1.PDR	0xff	IOR(1)	0x0008c001
PORT1.PODR	0x9f	IOR(1)	0x0008c021
PORT1.PIDR	0x9c	IOR(1)	0x0008c041
PORT1.PMR	0x00	IOR(1)	0x0008c061
PORT1.ODR0	0x00	IOR(1)	0x0008c082
PORT1.ODR1	0x00	IOR(1)	0x0008c083
PORT1.PCR	0x00	IOR(1)	0x0008c0c1
PORT1.DSCR	0x00	IOR(1)	0x0008c0e1
PORT1.PDR.B7	0x1	IOR(1ビット)	0x0008c001.7
PORT1.PDR.B6	0x1	IOR(1ビット)	0x0008c001.6
PORT1.PDR.B5	0x1	IOR(1ビット)	0x0008c001.5
PORT1.PDR.B4	0x1	IOR(1ビット)	0x0008c001.4
PORT1.PDR.B3	0x1	IOR(1ビット)	0x0008c001.3
PORT1.PDR.B2	0x1	IOR(1ビット)	0x0008c001.2
PORT1.PODR.B6	0x0	IOR(1ビット)	0x0008c021.6
PORT1.PODR.B7	0x1	IOR(1ビット)	0x0008c021.7
PORT1.PODR.B5	0x01	IOR(1ビット)	0x0008c021.5
PORT1.PODR.B4	0x1	IOR(1ビット)	0x0008c021.4
PORT1.PODR.B3	0x1	IOR(1ビット)	0x0008c021.3
PORT1.PODR.B2	0x1	IOR(1ビット)	0x0008c021.2

3、「■ブレークポイントの設定■」で32行目にカーソルを置いた場合は、直前31行目の

「LED\_PORT = 1;」を実行して停止しますので、「PORT1.PODR.BIT.B5」は1になり、LEDは消灯します。



2) 変更するメモリのアドレスを表示します。

アドレス「fff005ab」の内容は「fbae8813」となっています。アセンブラは「MOV,L #1388H,R14」です。

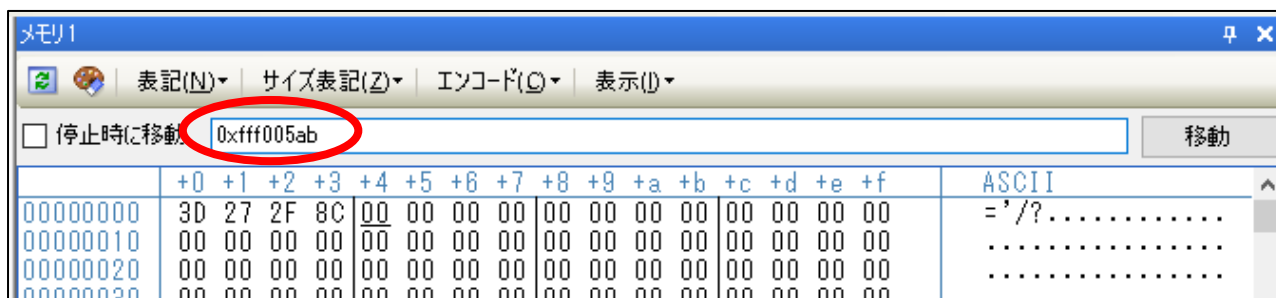
この1388が46)で修正した「DELAY\_VALUE 5000」(5000は16進数の1388H)です。メモリには上位下位が逆に格納されていますので8813となっています。

```

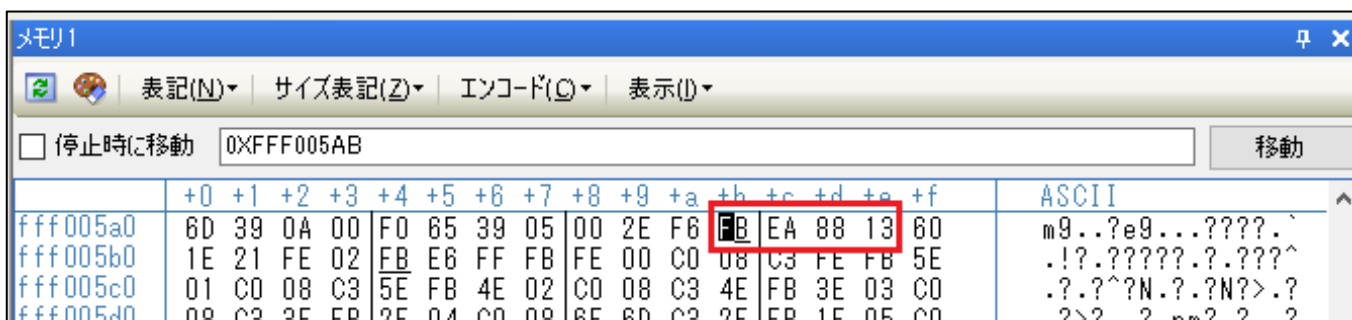
36
37          fff005ab fbae8813  _INTERVAL_DEL  /* 点滅間隔生成関数 */
          fff005af 601e         void INTERVAL_DELAY(void)
          fff005b1 21fe         MOV.L #1388H,R14
                                     SUB #1H,R14
                                     BNE.B _INTERVAL_DELAY+4H
38          [
39                                     unsigned long L;
40                                     for ( L = 0 ; L < DELAY_VALUE ; L ++ ); //WAIT
41          ]
          fff005b3 02         RTS
    
```

3) 「メモリ1」表示窓にアドレス fff005ab を表示させます。

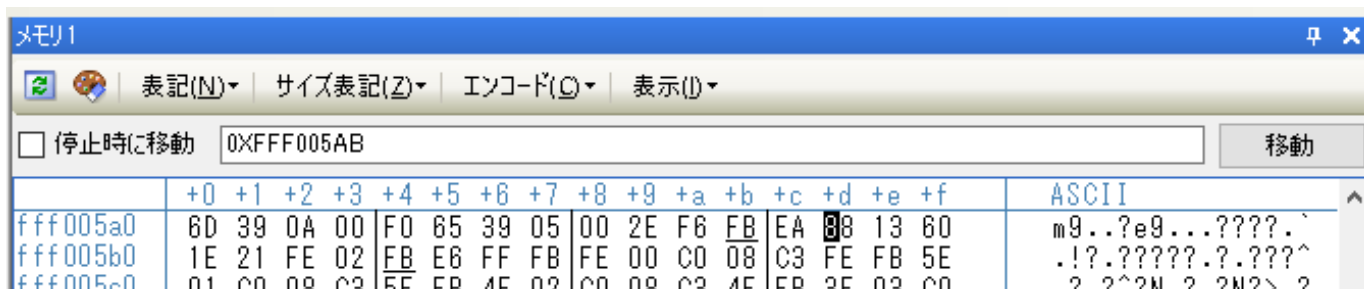
アドレス入力窓に「0xfff005ab」または「0XFFF005AB」(0XFFF005AB)と入力して右の「移動をクリック」(最初の「OX」「Ox」は16進数を表します)



4) アドレス fff005a0 から表示され、指定したアドレス fff005ab がアンダーライン付きで表示されます。FBEA8813 がメモリに格納されている機械語です。8813 が16進数の1388、十進数の5000になるので、ここを変更します。



5) 変更するアドレス fff005ad の88 をクリックすると反転表示になりますので、ここから十進数 50000、16進数でC350を格納していきます。上位下位が逆になるので50C3です。88の部分から50C3と入力してキーボードからEnterを押して確定します。





- 6) メモリを変更した値を確認します。緑色横向き▲で実行し■で停止します。  
LEDの点滅周期が遅くなっている事を確認してください。
- 7) この状態ではC言語ソースファイルは変更されていません（マイコン内のメモリは変更されています）。  
必要に応じてソースファイル「TEST\_LED.c」の内容を変更してください。
- 8) マイコンからユーザインタフェースケーブルをはずし、CN4（USBコネクタ）とパソコンを接続し電源を供給すると、この変更されたメモリの状態でプログラムが実行されます。  
（このサンプルプログラムの場合、SW2のDIPスイッチは全てOFFで実行に問題ありません）
- 9) メニューバーの「ファイル」－「終了」でプログラムを終了してください。  
（変更を保存する必要がある場合は、保存して終了してください）

より詳しい操作説明はRENESES社ホームページのE2エミュレータLite  
のマニュアルをご覧ください。

- ・次ページからはR5F52108ADFP ボードとR5FT210BBD FP ボードの違いによるプログラミングの相違点を説明いたします。

# R5F52108ADFP ボードと R5FT210BBDFP ボード の違いによるプログラミングの相違点

R5F52108ADFP ボード（以下 52108A と記述）と R5F5210BBDFP ボード（以下 5210BB と記述）は以下の部分で違いがあります。プログラミングの際にはご注意ください。

CPU	5 F 5 2 1 0 8 A D F P (チップバージョンA)	R 5 F 5 2 1 0 B B D F P (チップバージョンB)
ROM	5 1 2 K	1 M
RAM	6 4 K	9 6 K
システムクロックコントロールレジスタ3 (SCKCR3)	クロックソースとしてのメインクロック発振器 選択不可	選択可
PLL電源コントロールレジスタ (PLLPCR)	PLLの電源を切り低消費電力にする機能 なし	あり
クロック周波数精度測定回路 CACコントロールレジスタ1 (CACR1)	周波数測定クロックとしてのメインクロック発振器出力クロック 選択不可	選択可
クロック周波数精度測定回路 CACコントロールレジスタ2 (CACR2)	基準信号生成クロックとしてのメインクロック発振器出力クロック 選択不可	選択可
動作電力コントロールレジスタ (OPCCR)	中速動作モード2A、2B なし	あり
スリープモード復帰クロックソース切り替えレジスタ (RSTCKCR)	スリープモード復帰クロックソースとしてのメインクロック発振器 選択不可	選択可
フラッシュHOCOソフトウェアスタンバイコントロールレジスタ (FHSSBYCR)	ソフトウェアスタンバイモードのフラッシュメモリの電源供給 制御必要	制御不要
5Vトレラント対応I/Oポート	P12、P13、P16	P12、P13、P16、P17
DC特性	消費電流特性の改善なし	消費電流特性の改善あり
ROM (コード格納用フラッシュメモリ) 特性	プログラム・イレース	
	1000回	10000回
	データ保持	
	1000回/10年	1000回/30年 10000回/1年
コールドスタート時のサブクロック制御回路初期化	必要	必要
1を書き込み必須の ポート方向レジスタ (PDR) 予約ビット	なし	P00~P02、P56 P60~P67、P70~P77 P80~P87、P90~P97 PF5、PJ5、PK2~PK5 PL0、PL1
リアルタイムクロック (RTC) のクロック設定手順	なし	RTC ソフトウェアリセット実施前に PCR2レジスタのb7を 0に設定
シリアルペリフェラルインターフェイス RSPIビットレジスタ (SPBR)	16.0Mbps 設定禁止	16.0Mbps 設定可能

次ページではプログラムに関連する各項目の詳細を説明いたします。

### ■システムクロックコントロールレジスタ3 (SCKCR3) ■

このレジスタの書き換えにはプロテクトレジスタ (PRCR) のPRKEYビットをA5h、該当プロテクトビット (PRCR0) を1にして書き込みを許可し、書き換え後は0にして書き込みを禁止します。

詳細はマニュアル内「プロテクトレジスタ (PRCR)」を参照してください。

- 52108A ではメインクロック発振器 (20MHz 水晶発振子) からの出力をそのままシステムクロックとして選択する事が禁止されています。この発振器出力を使用して20MHzのシステムクロックを得るには
  - ①システムクロックコントロールレジスタ (SCKCR3) のクロックソース選択ビット (CKSEL) でPLL回路を選択
  - ②PLLコントロールレジスタ (PLLCR) で分周器比率をPLLDIVで÷4設定、PLL回路倍率をSTCで×8設定
  - ③システムクロックコントロールレジスタ (SCKCR) のシステムクロック選択ビット (ICK) で2分周を設定上記で $20\text{MHz} \div 4 \times 8 \div 2 = 20\text{MHz}$ のシステムクロックが得られます。
- 5210BB ではシステムクロックコントロールレジスタ (SCKCR3) のクロックソース選択ビット (CKSEL) で直接メインクロック発振器出力をシステムクロックとして選択できます。

### ■PLL電源コントロールレジスタ (PLLPCR) ■

このレジスタの書き換えも前項と同じくプロテクトビット (PRCR0) を1にする必要があります。

- 52108A にはこのレジスタはありませんがアドレスは存在し、CS+上でも定義されています。レジスタビット0 (PLLPCNT) の値は読み出し書き込み共に可能ですが、低消費電力機能はありません。ビット7~1は予約ビットです。読むと0が読めます。書く場合、0としてください。

### ■クロック周波数精度測定回路CACコントロールレジスタ (CACR1) ■

- 52108A は周波数測定クロックとしてメインクロック発振器出力クロックを使用できません。FMCSビット2~0に000を設定した場合、動作は保証されません。
- 5210BB には制約はありません。

### ■クロック周波数精度測定回路CACコントロールレジスタ (CACR2) ■

- 52108A は基準信号生成クロックとしてメインクロック発振器出力クロックを使用できません。RSCSビット2~0に000を設定した場合、動作は保証されません。
- 5210BB には制約はありません。

### ■動作電力コントロールレジスタ (OPCCR) ■

b2~0がOPCMビットで、クロック (システム、周辺モジュール、外部バス) を制限する事で動作電力をコントロールします。

- 52108A ではビット2~0 (OPCM) に100、101の設定 (中速動作モード2A、2B) は禁止されています。読み出し書き込み共に可能ですが、動作は保証されません。

### ■スリープモード復帰クロックソース切り替えレジスタ (RSTCKCR) ■

b2~0がRSTCKSELビットで、下記の様に機能します。

- 52108A ではシステムクロックにメインクロックを設定できません。それによりスリープモードからの復帰の際はHOCO (高速オンチップオシレータ) の一択となります。ビット2~0に001以外を設定しないで

ください。

- 5201BB ではスリープモードからの復帰にHOCOとメインクロックの二択となります。ビット 2~0 には 001 (HOCO)、010 (メインクロック) 以外を設定しないでください。

#### ■フラッシュHOCOソフトウェアスタンバイコントロールレジスタ (FHSSBYCR) ■

b 2~0 がSOFT CUTビットで、下記の様に機能します。

ビット2 = 電源検出回路の動作とPOR (パワーオンリセット) の低消費電力機能決定ビット。

1 で電源検出回路は動作、POR低消費電力機能有効

0 で電源検出回路は停止、POR低消費電力機能無効

ビット1 = HOCO (高速オンチップオシレータ) 電源供給ビット

1 でHOCO電源供給OFF

0 でHOCO電源供給ON

ビット0 = フラッシュメモリへの電源供給ビット (5210BB ではどちらを選択しても大丈夫です)

1 でフラッシュメモリ電源OFF

0 でフラッシュメモリ電源ON

- 52108A ではソフトウェアスタンバイモード時のフラッシュメモリの電源供給を制御する必要があります。プログラム時にビット0 を書き込んでください。
- 5210BB ではフラッシュメモリの電源供給を考慮する必要がありません。それによりフラッシュメモリへの電源供給関連ビット0 は二択同義 (don't care) となります。

#### ■5Vトレラント対応 I/Oポート■

- 52108A ではP1 2、P1 3、P1 6が5Vトレラントです。
- 5201BB では上記3ポートに加えてP1 7が5Vトレラントです。

#### ■ポート方向レジスタ (PDR) 予約ビット■

- 52108A では下記制約はありません。
- 5210BB では下記ビットに「1」を書き込む事が必須です。  
P00~P02、P56、P60~P67、P70~P77、P80~P87、P90~P97、  
PF5、PJ5、PK2~PK5、PL0~PL1

#### ■リアルタイムクロック (RTC) のクロック設定手順■

- 52108A では制約はありません。
- 5210BB ではRTCソフトウェアリセット実施前にPCR2レジスタのb 7を0に設定してください。

#### ■シリアルパライフェラルインターフェイス (RSPI) の設定注意事項■

RSPIビットレートレジスタ (SPBR) および

RSPIコマンドレジスタのビットレート分周設定ビット (BRDV)

- 52108A では上記2つのレジスタを同時に0にしてビットレートに16Mbpsを選択することはできません。
- 5201BB では制約はありません。