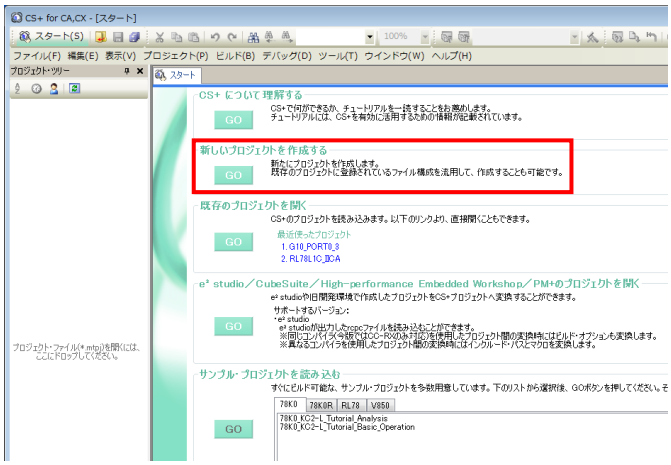
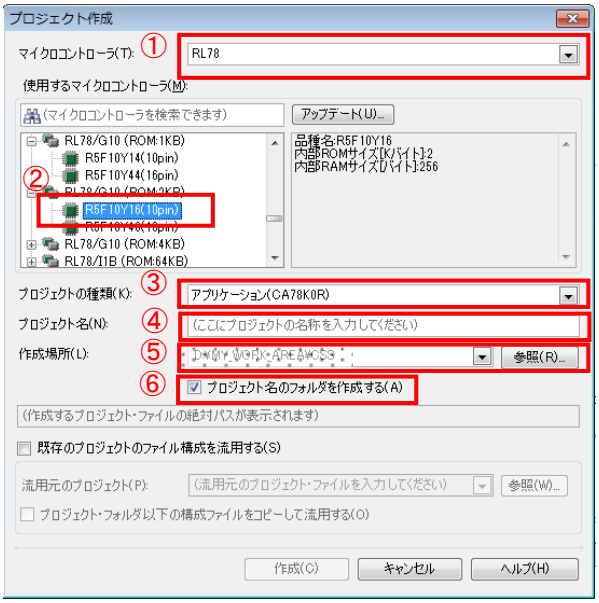
番外編　統合開発環境 CS+とコード生成機能

ここでは，RL78 のプログラム開発には統合開発環境 CS+を使用します。
CS+には、RL78 の言語ツールとして CA78K0R が含まれており，プログラミングを支援するためのコード生成機能も含まれています。
CS+を使用する方法について，重要なポイントを簡単に説明します。

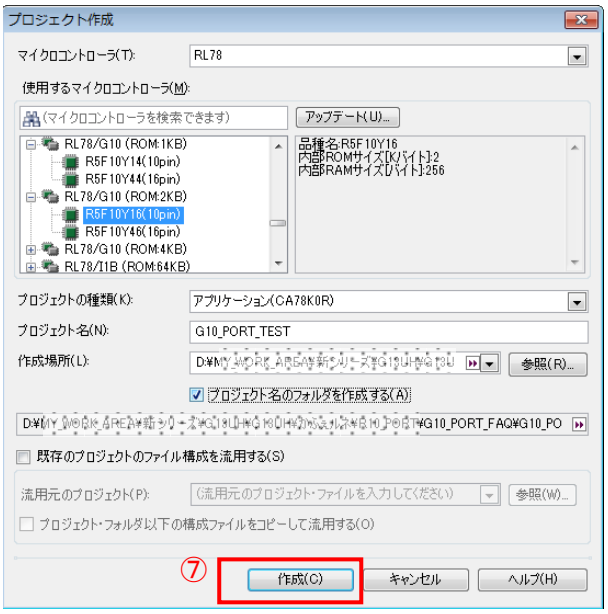
(1)プロジェクトの作成
CS+を用いて開発するプログラムはプロジェクトとして管理されます。最初にやることは，プロジェクトを作成することです。
CS+の起動画面を示します。新しいプロジェクトを作成するには，起動画面の赤で囲んだ「新しいプロジェクトを作成する」の「GO」ボタンをクリックします。



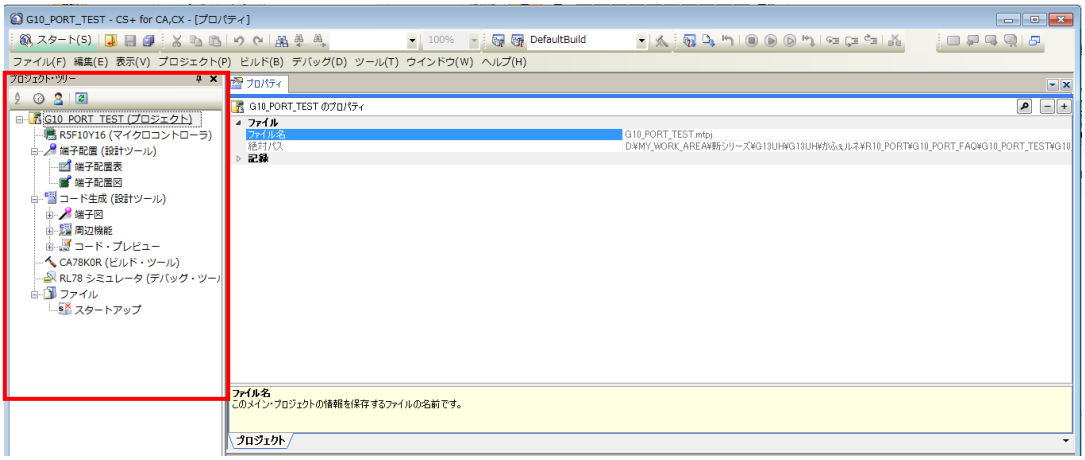
すると，下に示すような「プロジェクト作成」用のウィンドウがポップアップしてきます。
ここで，最初に①使用するマイクロコントローラ（以下マイコンと称します）のシリーズ名を指定し，②具体的なデバイスを指定します。これで，マイコンの指定は完了です。次は作成するプロジェクトの指定です。③プロジェクトの種類は「アプリケーション」のままで構いません。④プロジェクト名は判別し易（やす）い名前を付けてください。⑤の作成場所は適切なフォルダを指定します。⑥は通常チェックしておきます。



④のプロジェクト名を入力すると，⑦の作成のボタンがクリックできるようになります。
設定が完了したら，設定内容を確認後に⑦の作成ボタンをクリックします。



作成が完了すると，CS+の画面が開きます。左側にプロジェクト・ツリーが表示されます。ここで，対象を選んで，右側で設定していくことになります。



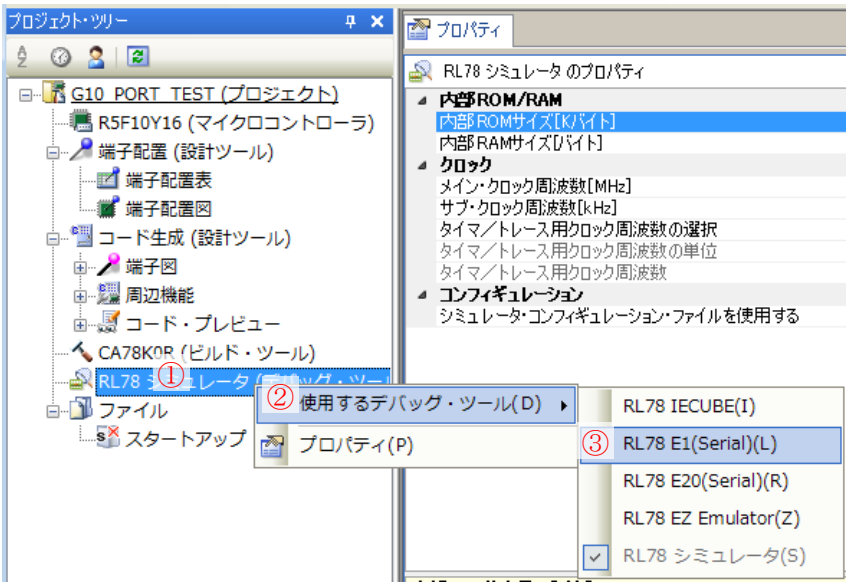
(2)CS+の設定

最初に共通的な設定を行います。まずは，デバッグ・ツールの設定を変更します。デバッグ・ツールとしては，初期状態では「RL78 シミュレータ」が選択されています。RL78/G10 では，周辺機能を含めたシミュレーションが可能ですが，ここでは「E1 エミュレータ」に変更します。

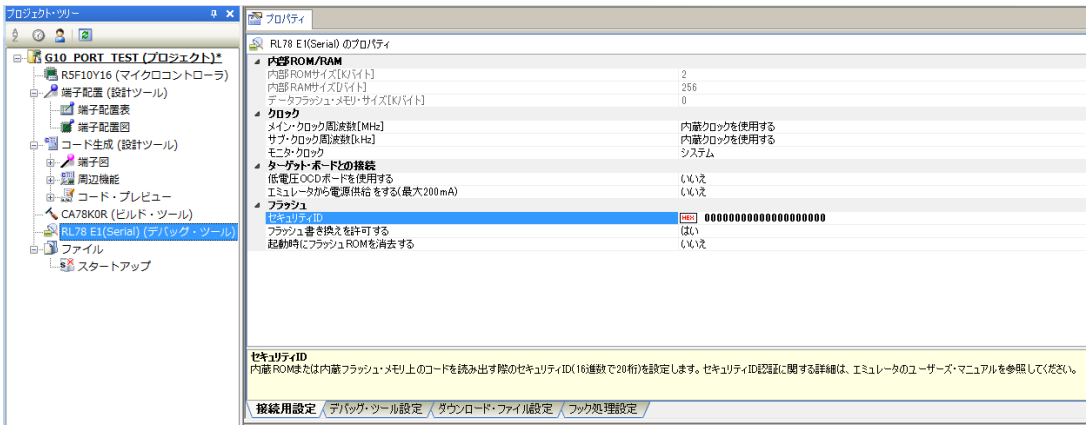
①プロジェクト・ツリーの「RL78 シミュレータ・・・」を選択し，マウスの右ボタンをクリックしてポップ・アップ・メニュを表示します。

②「使用するデバッグ・ツール」をクリックすると，右側にリストが表示されます。

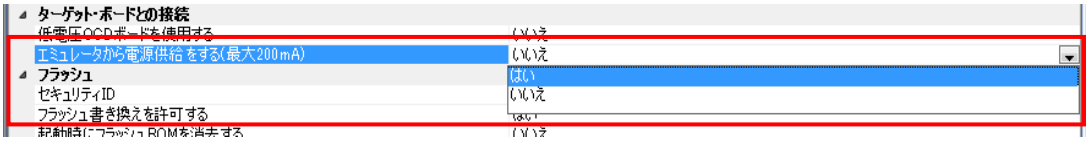
③リストから「RL78 E1・・・」を選択します。



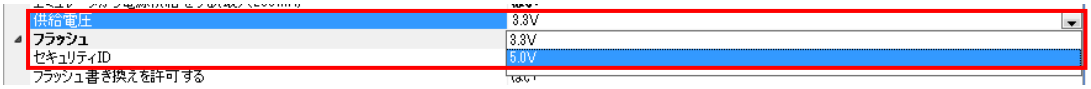
選択が完了すると，右側には E1 エミュレータの設定内容が表示されます。
「セキュリティ ID」が反転表示され，値として 20 桁の 0 が並んでいます。
これが，オンチップ・デバッグでデバイス内部のプログラムの安全のために
設定する 10 バイトの ID コードです。製品として使用する訳ではないので，
ここではそのままにしておきます。
これに関連した項目がコード生成のところでも出てきます。



ここで，変更するのは，「ターゲット・ボードとの接続」の中の「エミュレータから
電源を供給する」の項目です。
ここの「いいえ」となっている部分をクリックすると右端に項目を選択するための
下向き三角が表示されるので，それをクリックして「はい」を選択します。

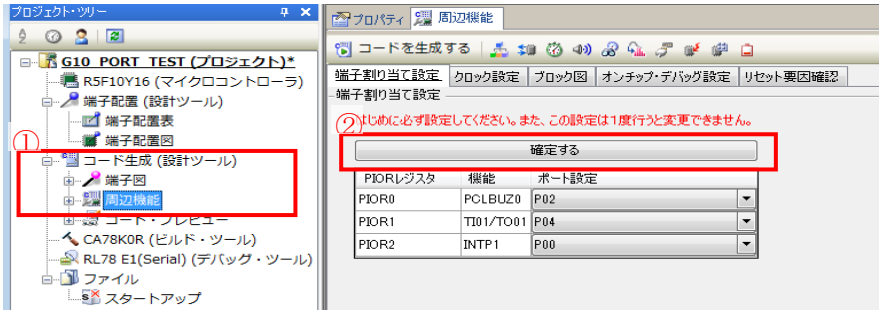


すると，「供給電圧」の項目が表示され，「3.3V」になっています。RL78/G10 は
3.3V での書き込みが保証されていないので，「5.0V」に変更します。



これで，E1 エミュレータの設定はひとまず完了です。

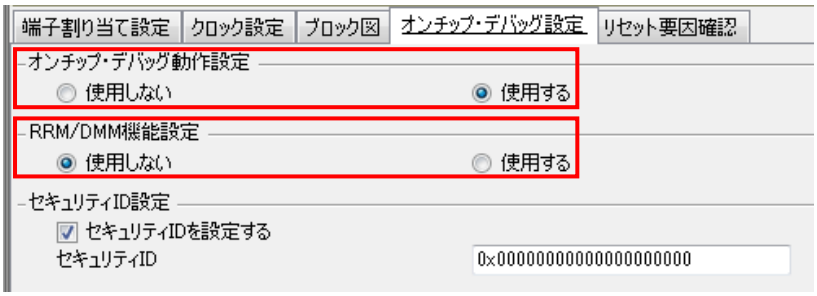
(3)コード生成の設定（共通的な内容）
次にコード生成の設定を行います。①コード生成（設計ツール）の「周辺機能」
をダブルクリックして，右側の画面に「端子割り当て設定」画面を表示します。
ここで，必ず端子の割り当てを行ってください。ここでは PIOR レジスタで設定
するポートのリダイレクション機能の指定です。一度，「確定する」をクリック
すると変更はできませんので，注意が必要です。今回はポートとして使用する
ので，②「確定する」をクリックします。



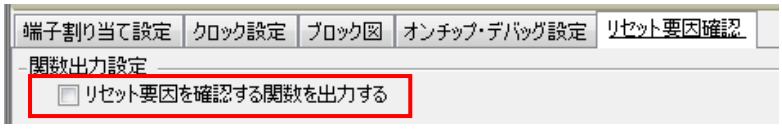
（使用するマイコンを変更すると，コード生成の設定はクリアされるようです。）

引き続いて，「クロック設定」から順にタグを開いて設定を確認します。
ここでは，「ＶＤＤの設定」，「高速オンチップオシレータ設定」および，
「RESET 端子設定」を行います。

次に「オンチップ・デバッグ設定」タグを開きます。
ここでは，「オンチップ・デバッグ動作設定」を「使用する」，「PRM/DMM
機能設定」を「使用しない」に設定します。「セキュリティ ID」はそのまま
にしておきます。

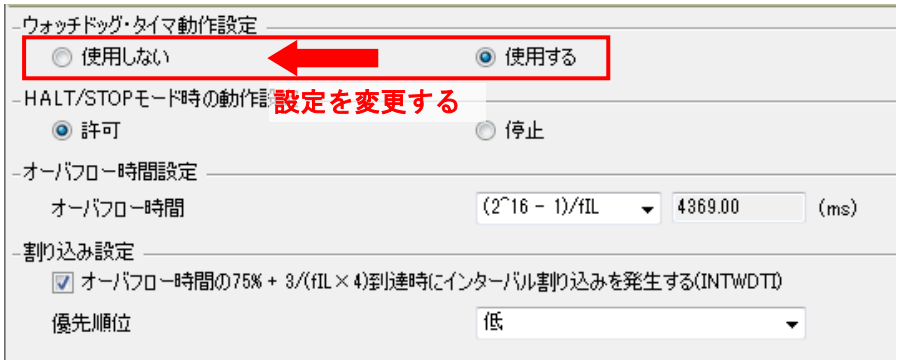


次は，「リセット要因確認」です。ここでは「リセット要因を確認する関数
を出力する」のチェックを外します。

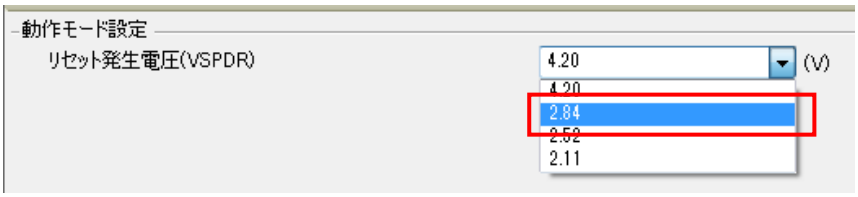


以上が共通の設定です。

それ以外に注意を要する設定が二つあります。
一つは，「ウォッチドッグ・タイマ」の設定です。この設定は，初期状態では
「使用する」となっています。そこで，この設定を「**使用しない**」に変更して
ください。これを忘れると，数秒ごとにリセットがかかってしまいます。
意図して動作させた訳ではないので，よく陥りやすいトラブルです。
もちろん，ウォッチドッグ・タイマを使用したい場合には，この限りではあり
ません。目的に応じて設定してください。



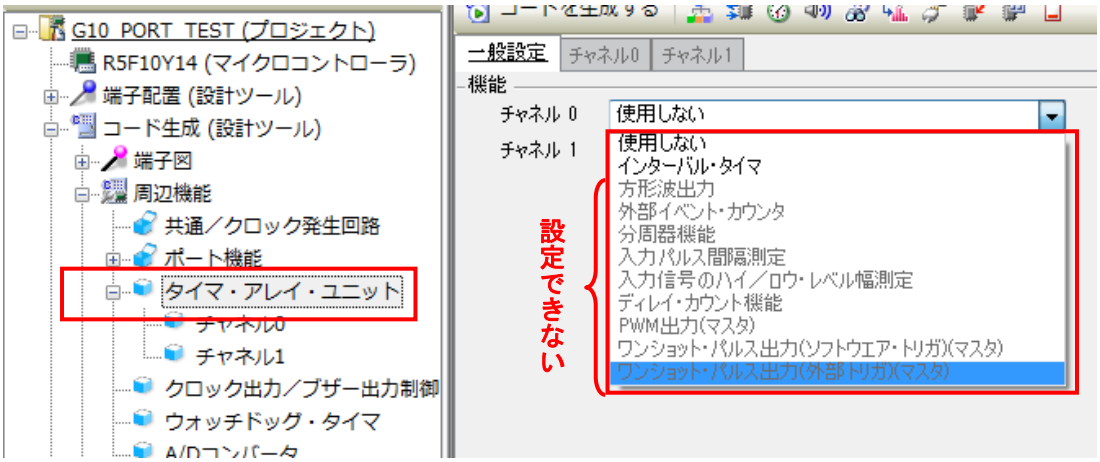
もう一つは，「セレクタブル・パワーオン・リセット」です。ここで設定した電圧よりも電源電圧が低いと，E1 エミュレータが起動できなくなります。また，クロック設定で指定した電圧より低い場合にも，起動できなくなることがあります。通常は 2.84V に設定しておきます。



以降は，使用する周辺機能について設定となります。

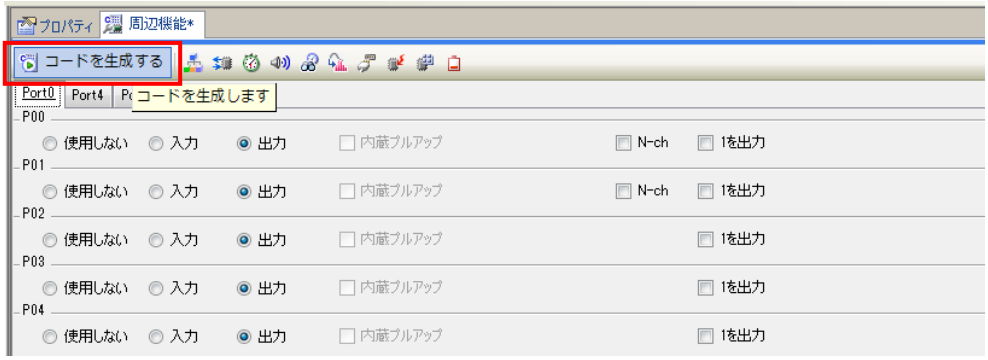
(4)コード生成での個別機能の設定
ポート機能で，P0 は全て出力ポートに設定する場合の設定は以下のようになります。すべて出力なので、出力を選択します。初期値としては 0 でいいので、右端の「1 を出力」はチェックしません。

そのほかのタイマ、シリアル、・・・は必要に応じて設定します。
タイマ（タイマ・アレイ・ユニット）の設定画面を示します。チャネル 0 のプルダウン・メニュを見ると、「使用しない」「インターバル・タイマ」以外薄い表示で選択できません。これはすでにタイマ関係の端子がほかの機能で使用されているためです。具体的には上で、ポート 0 を出力ポートとして使用するように設定したからです。

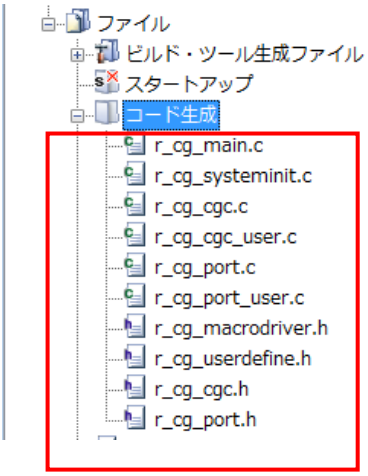


(5)コード生成の便利な設定
周辺機能の設定が完了したなら，コード生成を行いますが，その前にコード生成のプロパティを確認します。「コード生成・・・」を選択して，右クリックして「プロパティ」をクリックします。右側に表示された「ファイル生成モード」の「API 関数の出力制御」の項目で，「設定に合わせてすべて出力する」が初期設定となっています。通常は，この状態でいいのですが，プログラムのサイズの問題などで API を使いたくない場合には，ここを「初期化関数にみを出力する」に設定することもできます。

(6)コード生成の出力
最後に，「コードを生成する」をクリックすると，指定された条件でコードが生成されます。

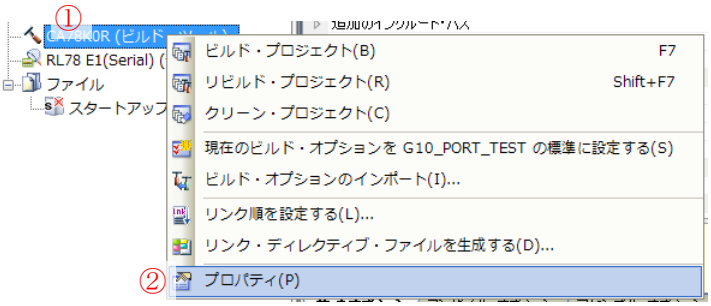


生成されたコードはプロジェクト・ツリーの「ファイル」の下の「コード生成」で確認できます。
「r_cg_main.c」は，main 関数が記述されたファイルです。「r_cg_systeminit.c」は，内蔵周辺機能の初期設定を行っているファイルです。以下の「r_cg_cgc.c」は動作クロックの設定を，「g_cg_port.c」はポートの初期設定を行っているファイルで，共に「r_cg_systeminit.c」から呼び出されています。
「r_cg_port_user.c」のように「・・・・・_user.c」は割り込み処理が記述されているファイルです。ここでは，何も割り込みは使用しないので，中にプログラムは記述されていません。

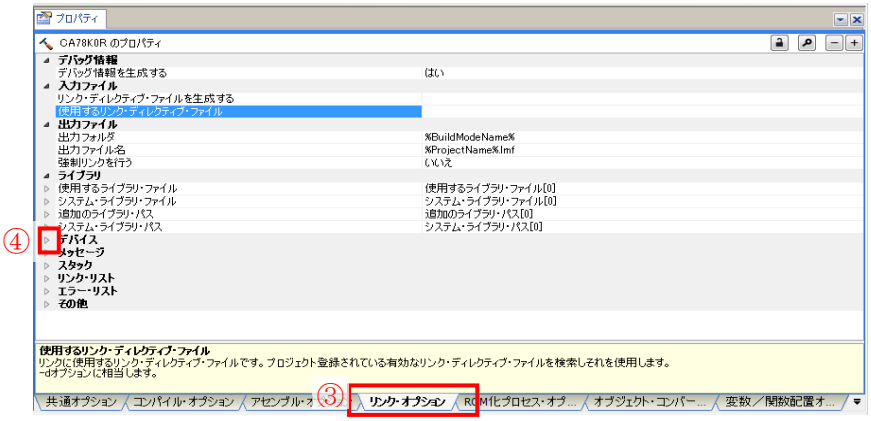


生成されたコードには、ユーザが自由に変更できる部分とできない部分に分かれています。自由に変更できる部分意外にプログラムを記述すると、次にコード生成を行った時に消されてしまいますので、くれぐれも間違えないようにしてください。

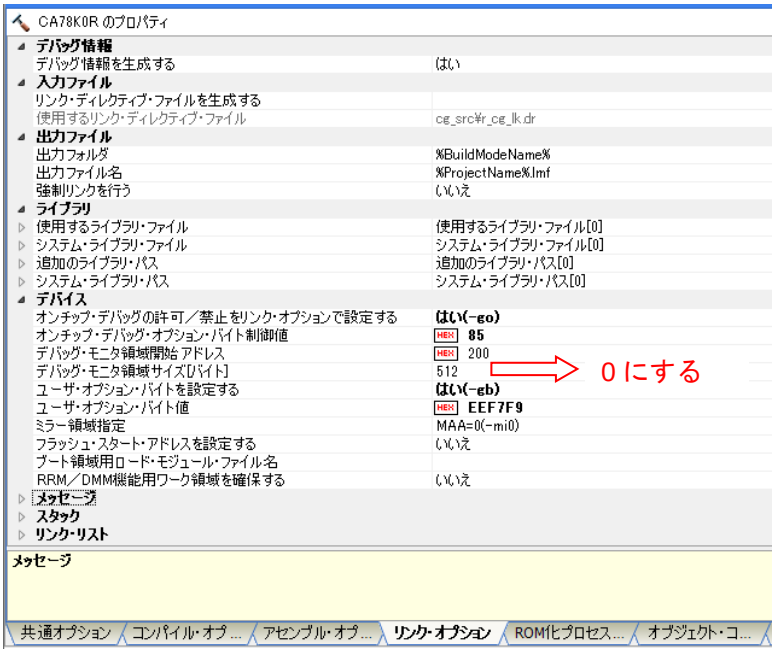
(7)CA78K0R の設定
RL78/G10 の場合には，絶対に変更しておいた方がよい設定があります。
プロジェクト・ツリーで①「CA78K0R・・・」を右クリックして，
プルダウン・メニュを表示させ，そこから「プロパティ」を選択します。



プロパティ画面が表示されたら，下側の「リンク・オプション」タグを選択して，③リンクに関係した項目を表示させます。「デバイス」の項目が閉じているので，④「デバイス」の左の三角をクリックして開きます。



コード生成を行った後は，「デバッグ・モニタ領域サイズ」が 512 バイトになっています。RL78/G10 では，この領域は必要ないので，0 に変更します。これで，コード・フラッシュを全てプログラムで使用可能となります。



これは、RL78/G10 ではソフトウェア・ブレークは使えないために可能なことです。
512 バイトは RL78/G10 にとっては大きな値です。特に上記の例の R5F10Y14 では、使用可能なフラッシュ・メモリの容量が倍になります。