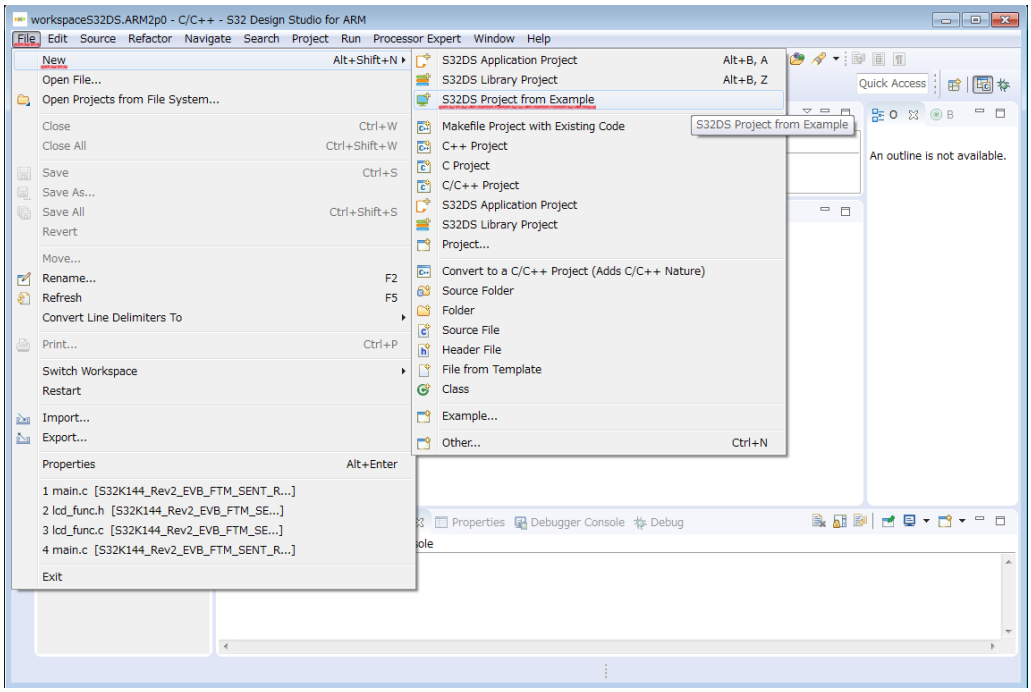
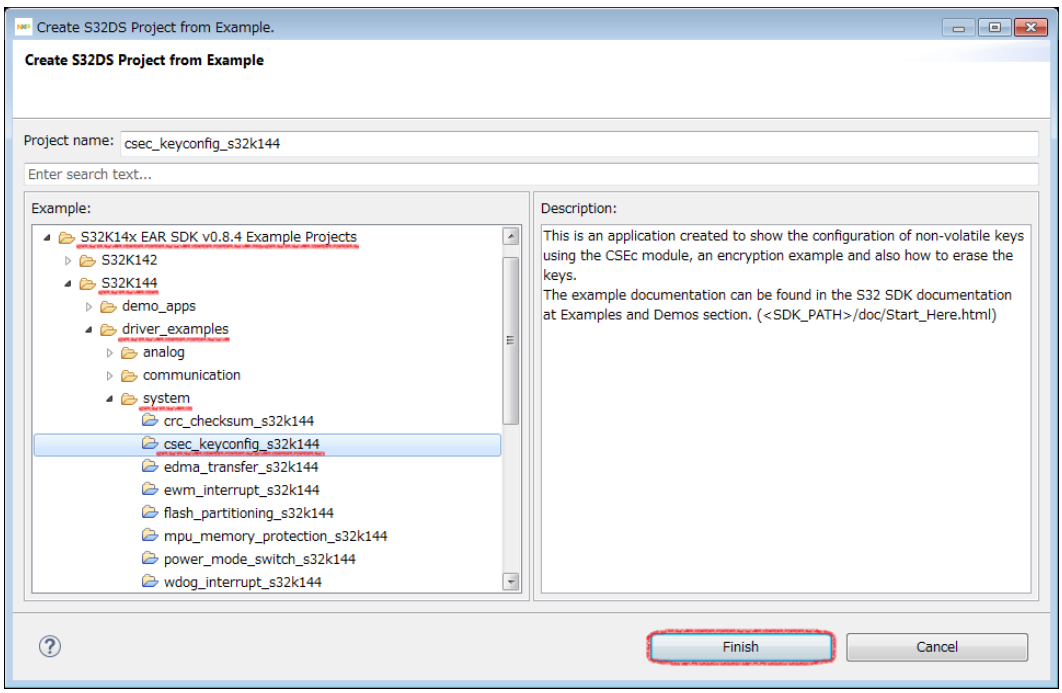
CSEcを使った鍵の設定、および暗号化の手順

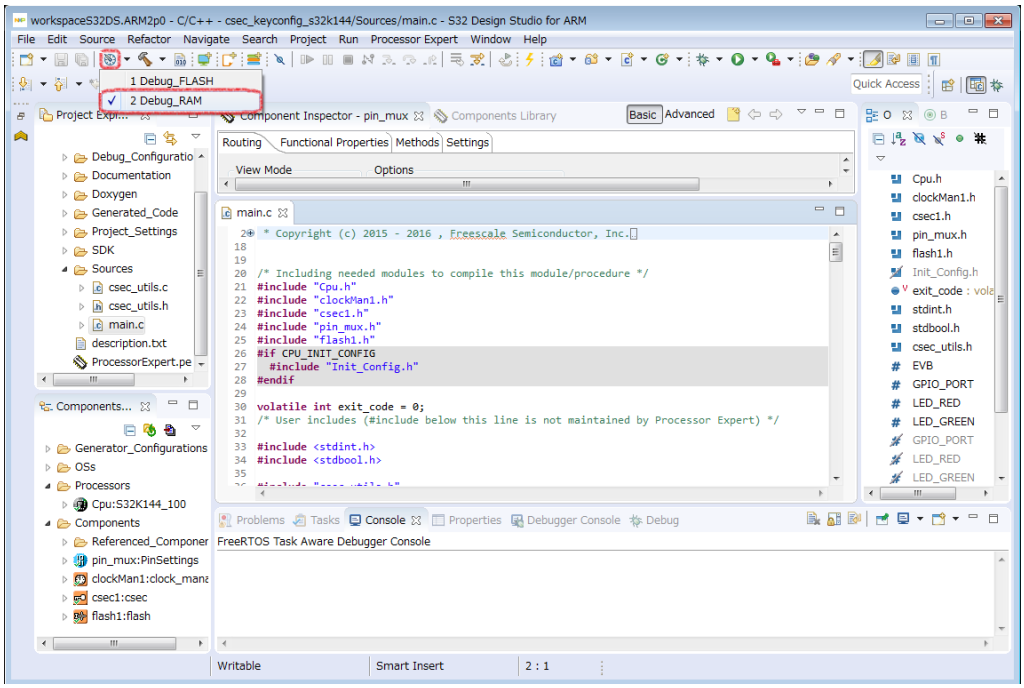
1. S32DSアプリを開き、Fileメニューからサンプル・プロジェクト選択メニューを開く（File → New S32DS Project from Example）。



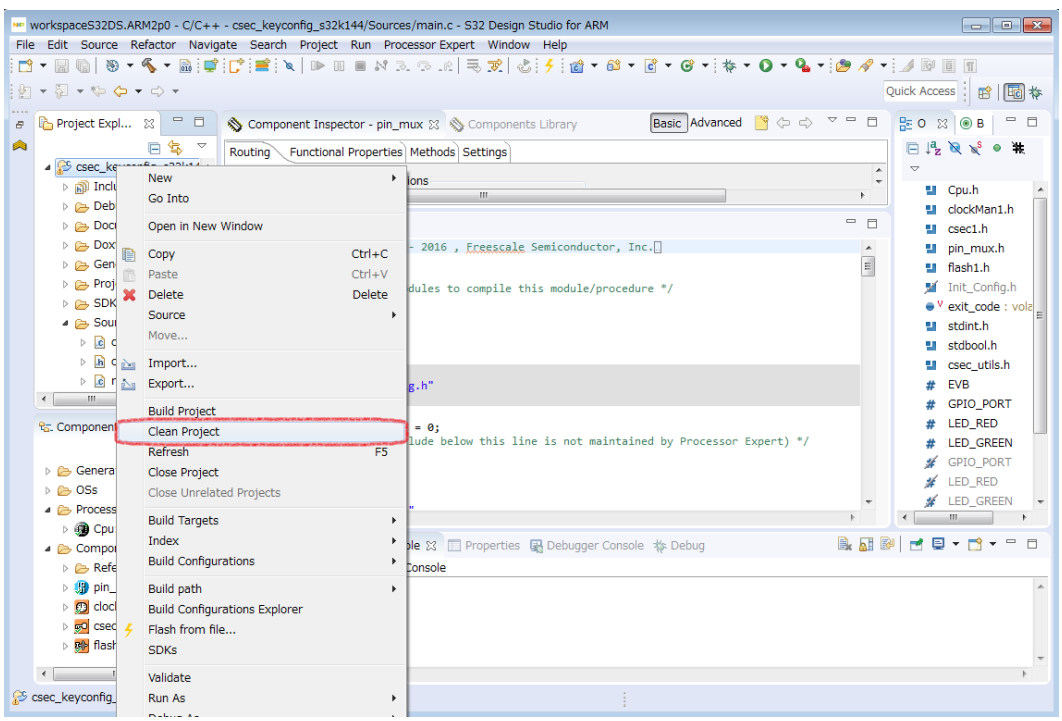
2. 読み込むサンプル・プロジェクトを選択する（S32K14x EAR SDK v0.8.4 Example Projects → S32K144 → driver_examples → csec_keyconfig_s32k144）。



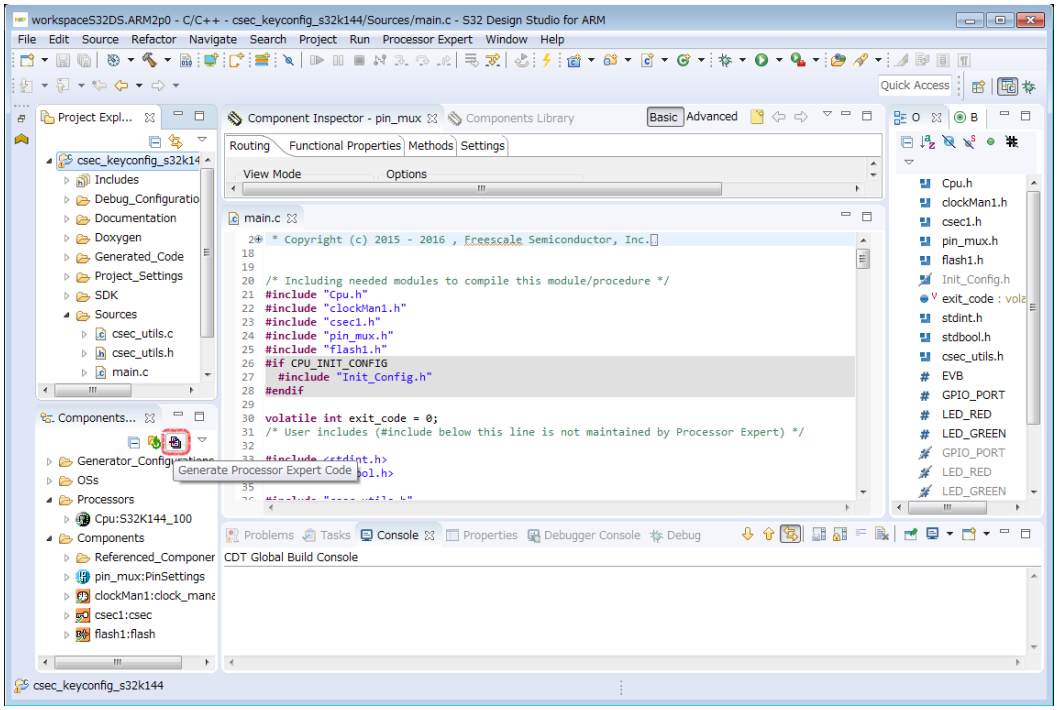
3. 読み込が完了したら、「 Project Explorer 」内のフォルダをクリックし、「構成管理」アイコン・メニューから “Debug_RAM” を選択。CSEcの鍵格納メモリの初期化にフラッシュの消去・書き込みが必要となるため、本コードはマイコン内蔵SRAMに転送する。



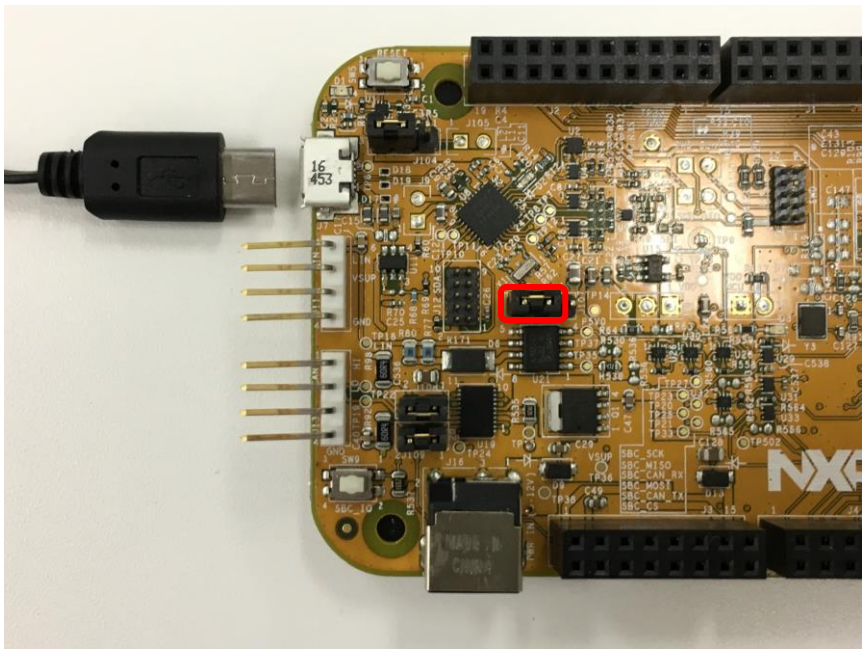
4. プロジェクト・フォルダアイコン上で右クリックメニューを開き「Clean Project」を実行。



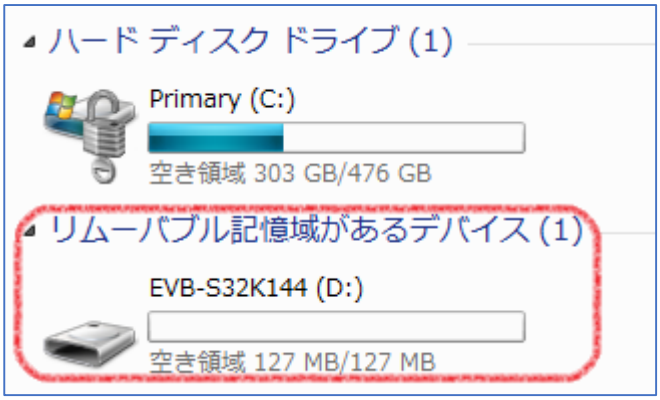
5. 「Components」ウインドウ内の “Generate Processor Expert Code” ボタンをクリックして、ソースコードを生成。



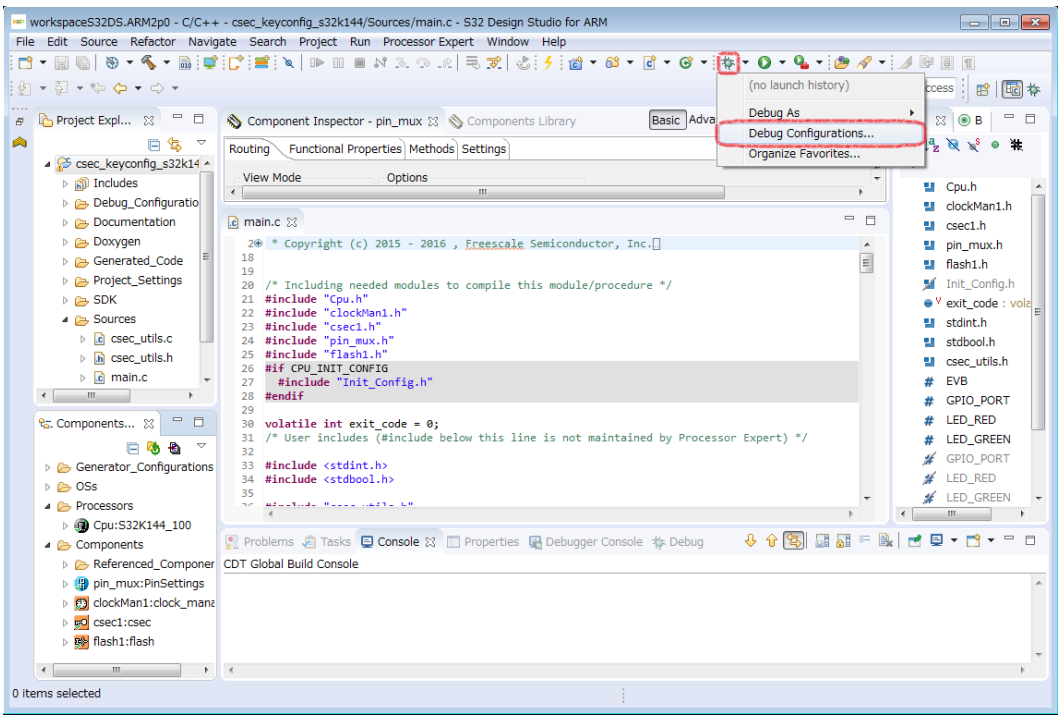
6. EVBのJ107ジャンパを2-3に切り替え、micro-USBでPCと接続する。EVBの電源はUSBバス電源から供給される。



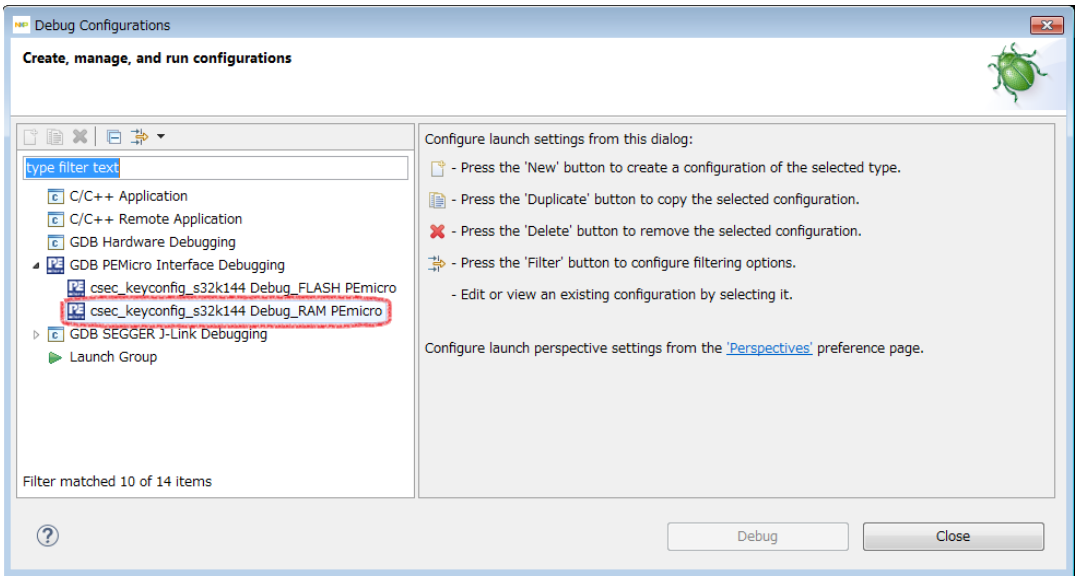
7. リムーバブル・デバイスとして「EVB-S32K144」がマウントされるのを待つ。



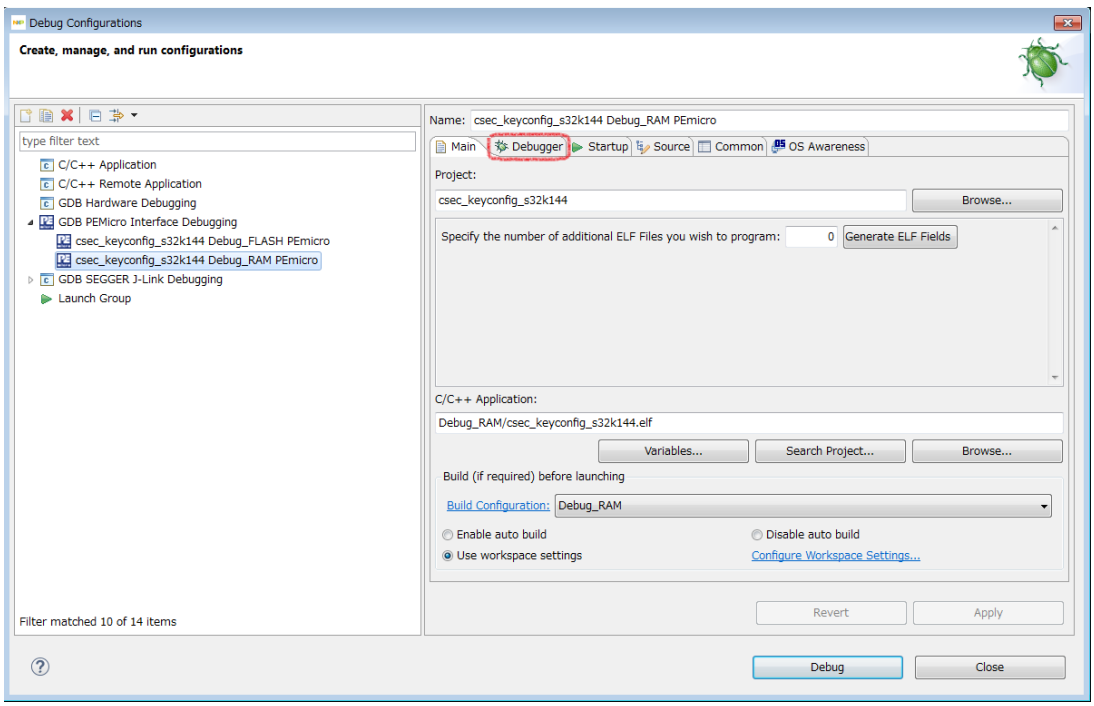
8. デバッグ・アイコン（虫マーク）のメニューから「 Debug Configurations… 」を選ぶ。



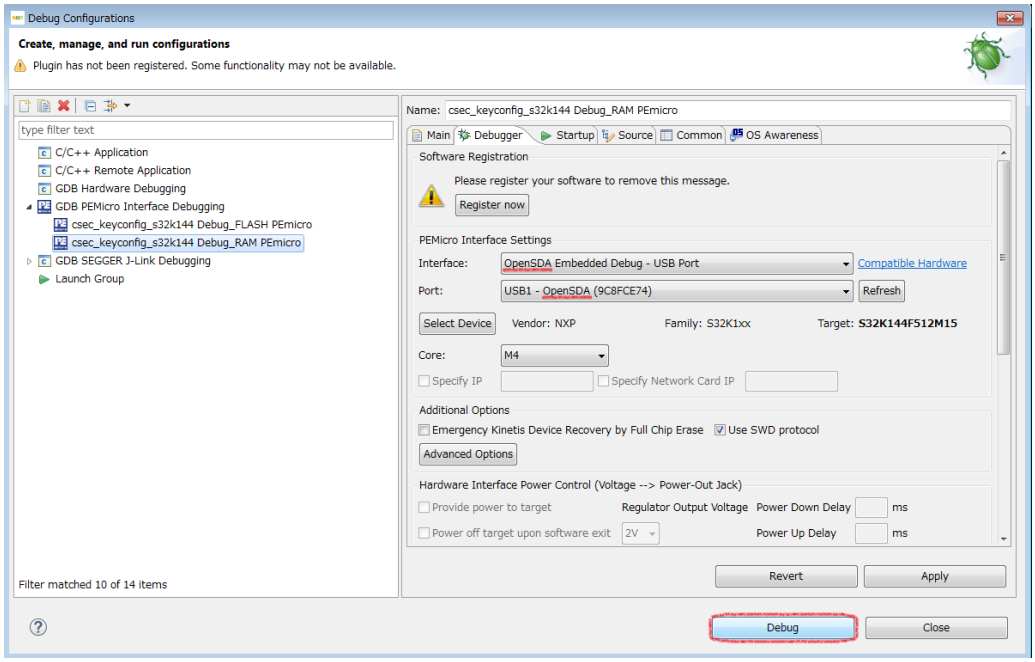
9. 左の窓内にある “csec_keyconfig_s32k144 Debug_RAM PEmicro” をクリック。



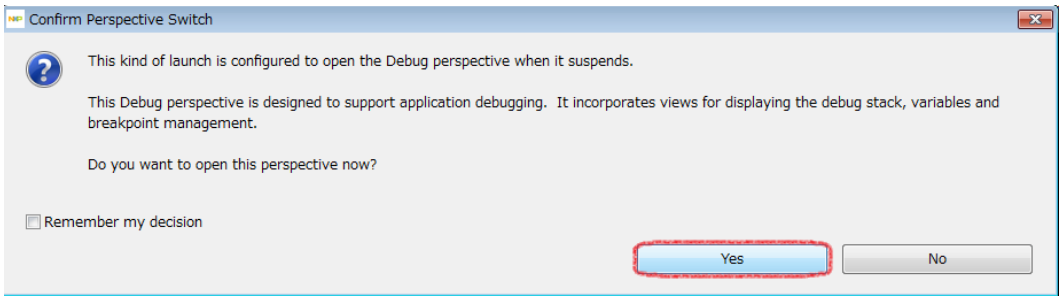
10. “Debugger” タブをクリック。



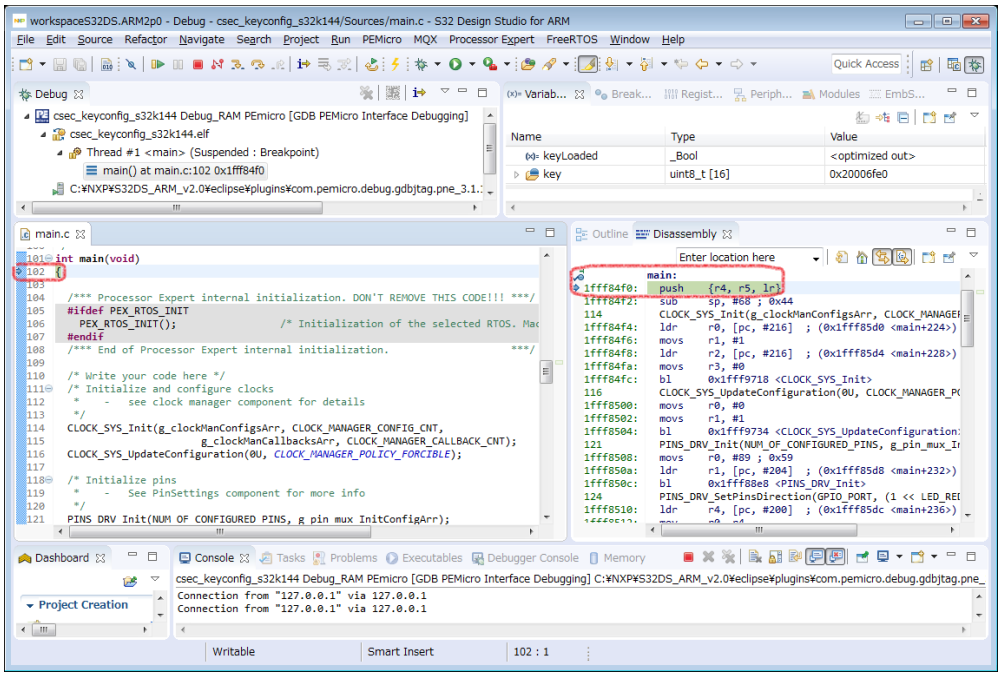
11. Interface および Port がそれぞれ “OpenSDA” となっているのを確認し、Debug ボタンをクリック。これによりコードがマイコン内蔵SRAMに転送される。



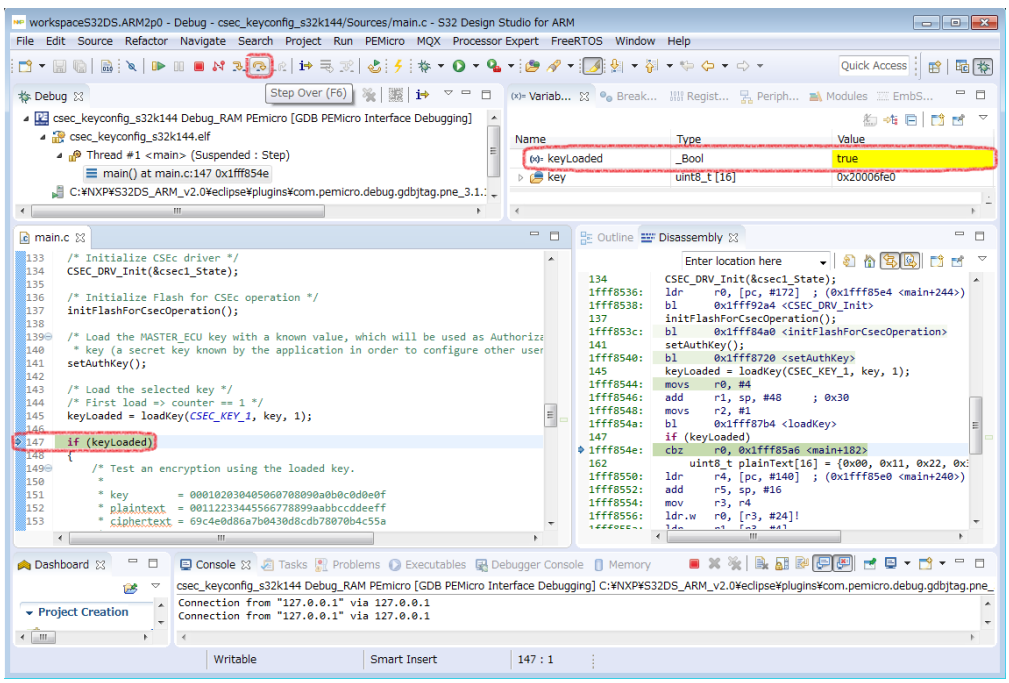
12.「Confirm Perspective Switch」ウインドウ内の“ Yes” をクリックしてデバッグ表示画面に移行する。



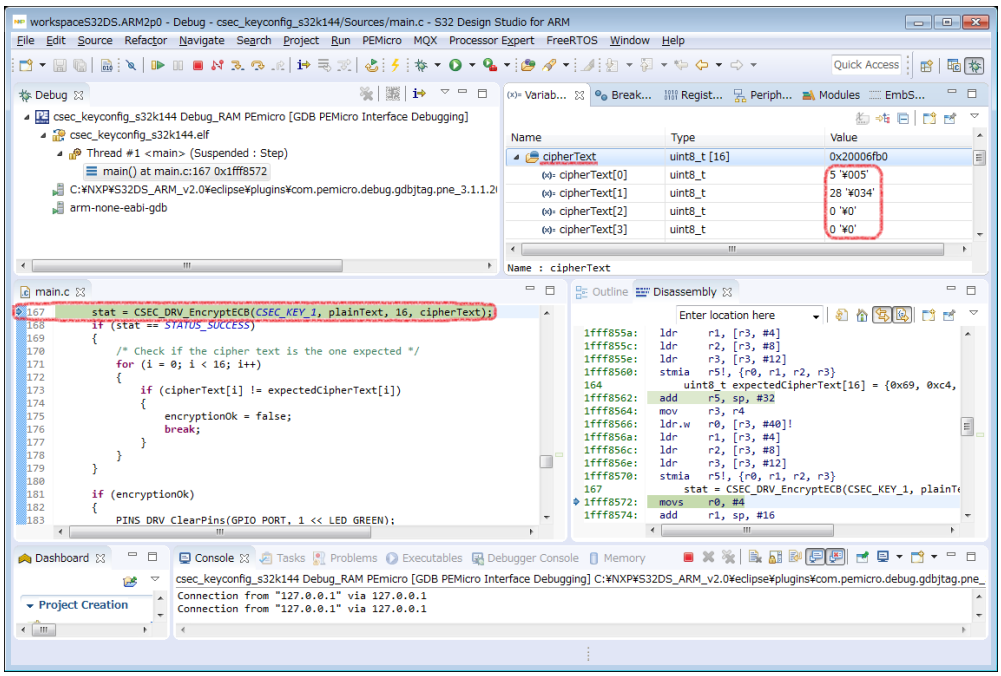
13. main 関数の冒頭でブレークしているのを確認。



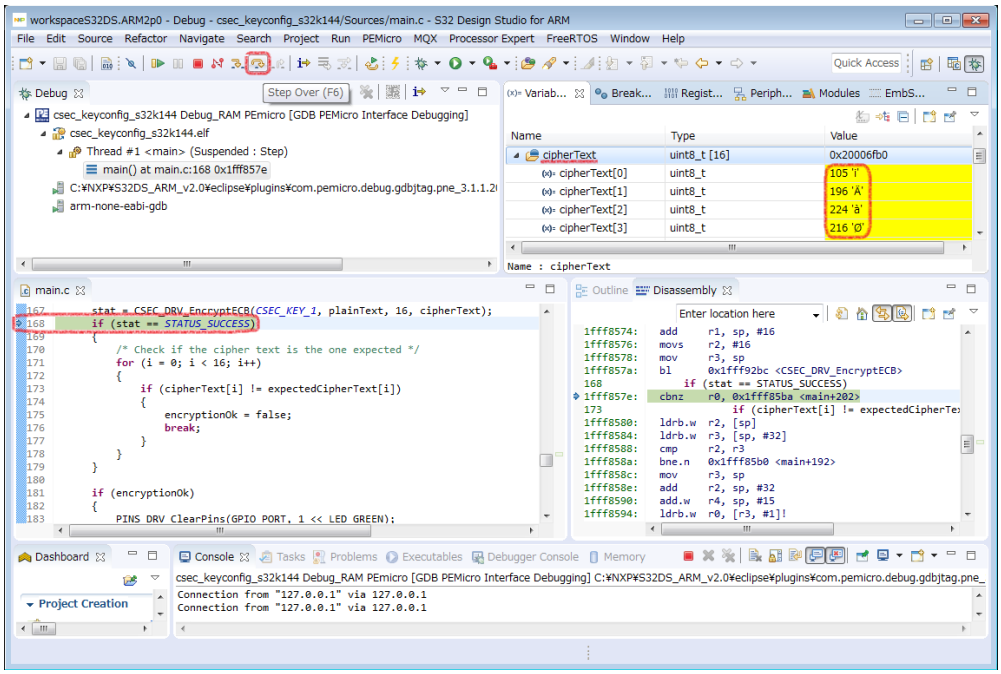
14. main 関数内の147行目まで、「Step Over」ボタン（関数単位での実行）をクリックして実行を進める。これにより、秘密鍵が CSEc の鍵格納用メモリに書き込まれ、“keyLoaded” 変数が “true” となる。ここで “true” とならない場合は、既に鍵が書き込まれている可能性があるため、鍵を消去してから再実行する。



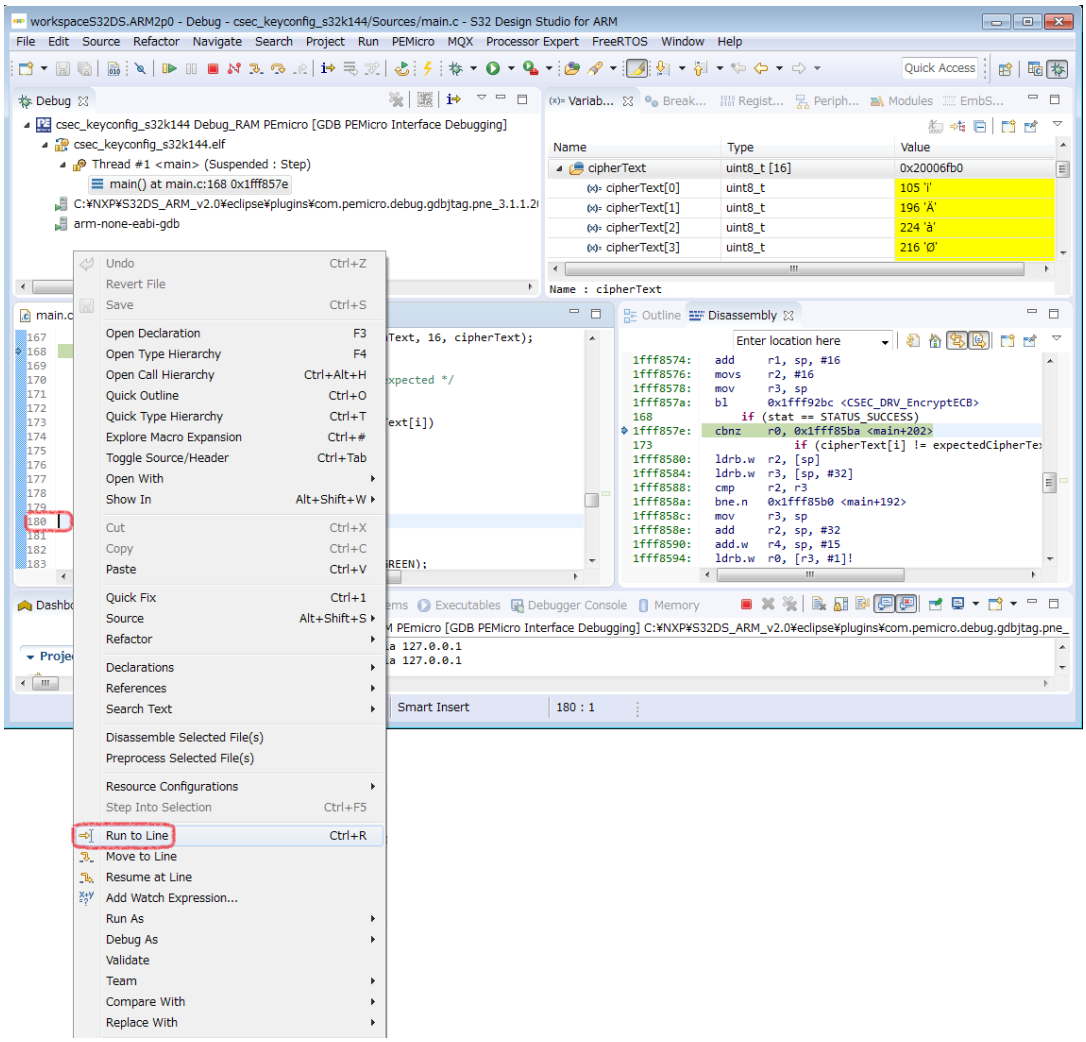
15. 引き続き「Step Over」ボタンをクリックして167行目まで実行を進める。この時点では “cipherText” 配列は未初期化状態であるため、「Variable」ウインドウ内で任意の値が格納されているのを確認する。



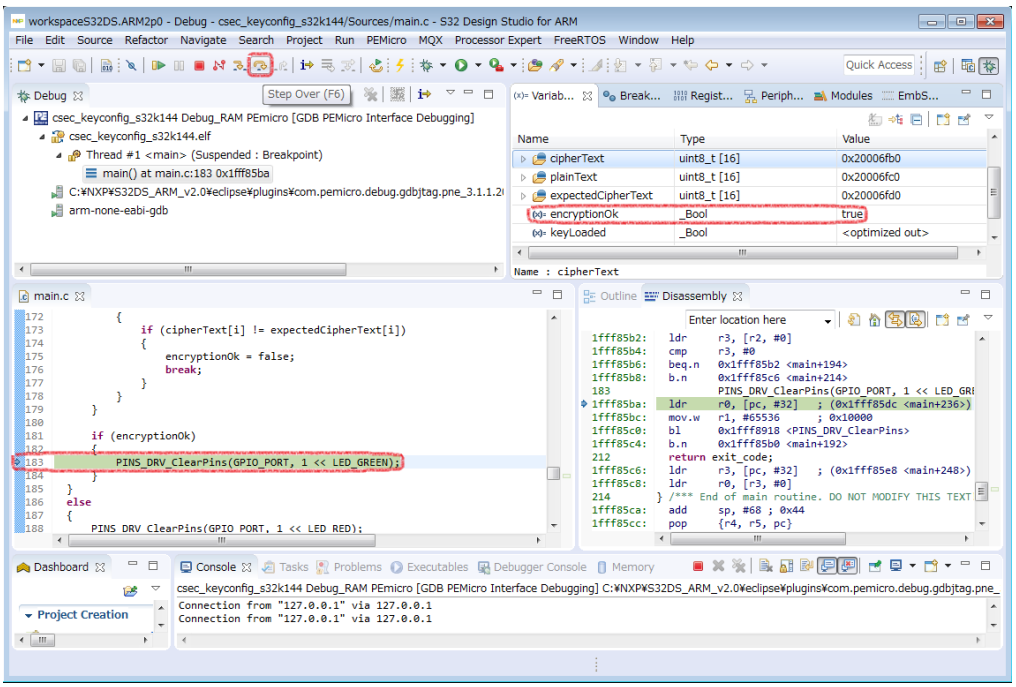
16.「Step Over」ボタンを1回クリックして、「CSEC_DRV_EncryptoECB」関数を実行する。これにより、“plainText” 配列に格納されていた値が暗号化され “cipherText” 配列に返される。

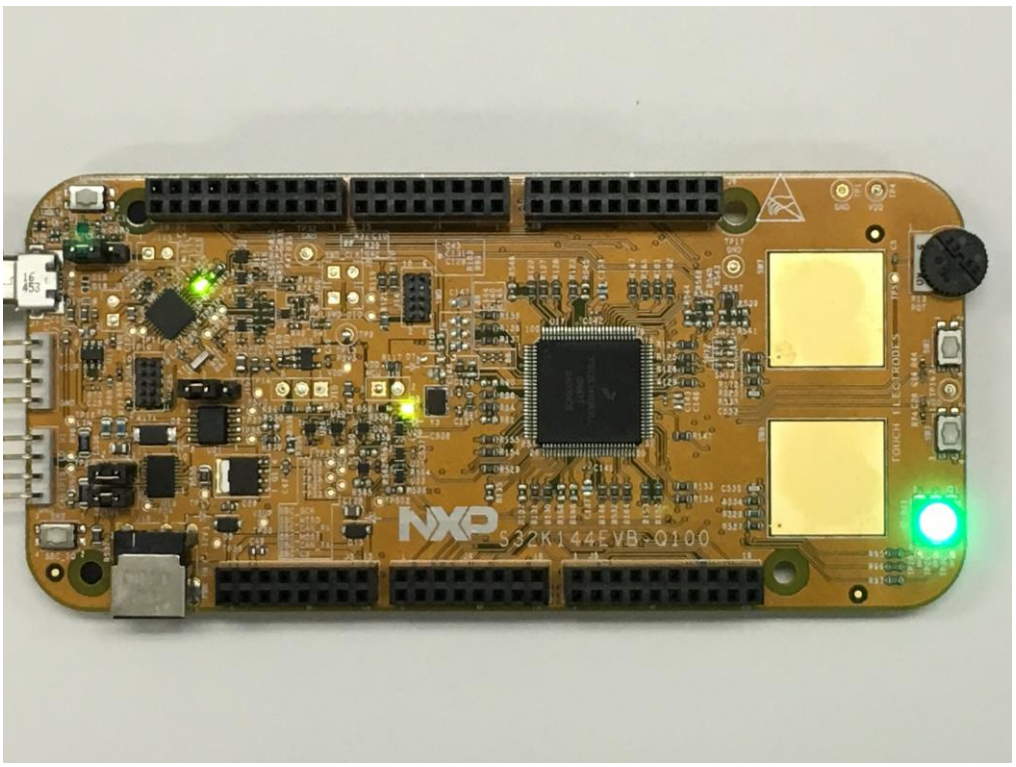


17. 180行目をクリックして、右クリックメニューから「Run to Line」を選択。これにより180行目まで一括で実行され、戻された “cipherText” 配列と、あらかじめ用意しておいた “expectedCipherText” と内容が一致しているかが比較され、結果が “encryptionOk” 変数に格納される。

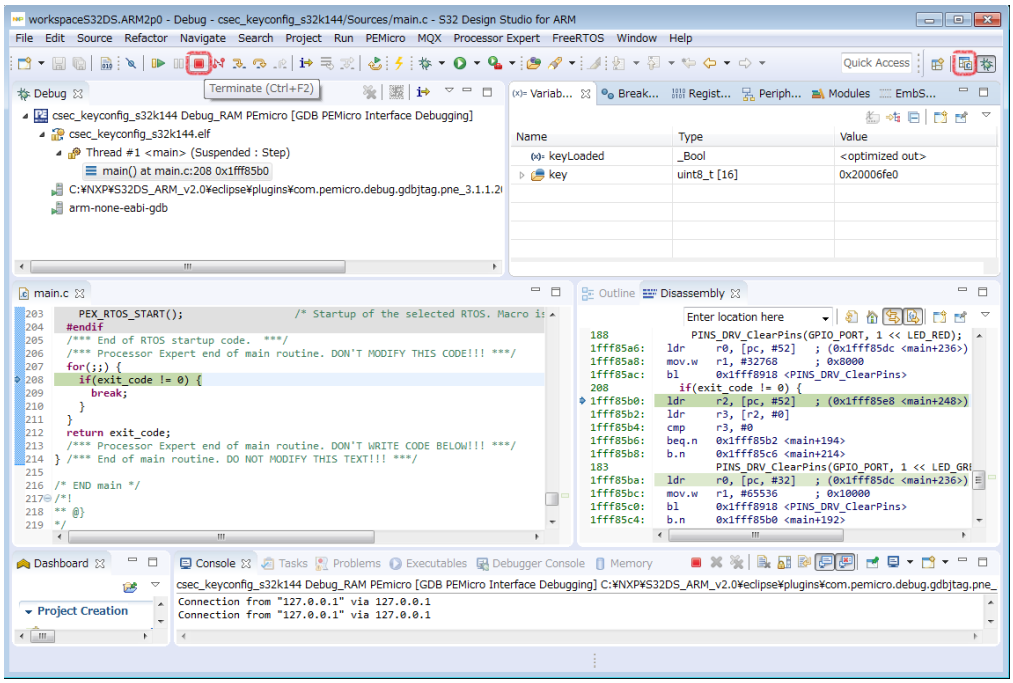


18. 配列内容が一致したため、183行目でブレークしているのを確認。ここで「Step Over」ボタンを1回クリックすると、EVBの緑LEDが点灯する。

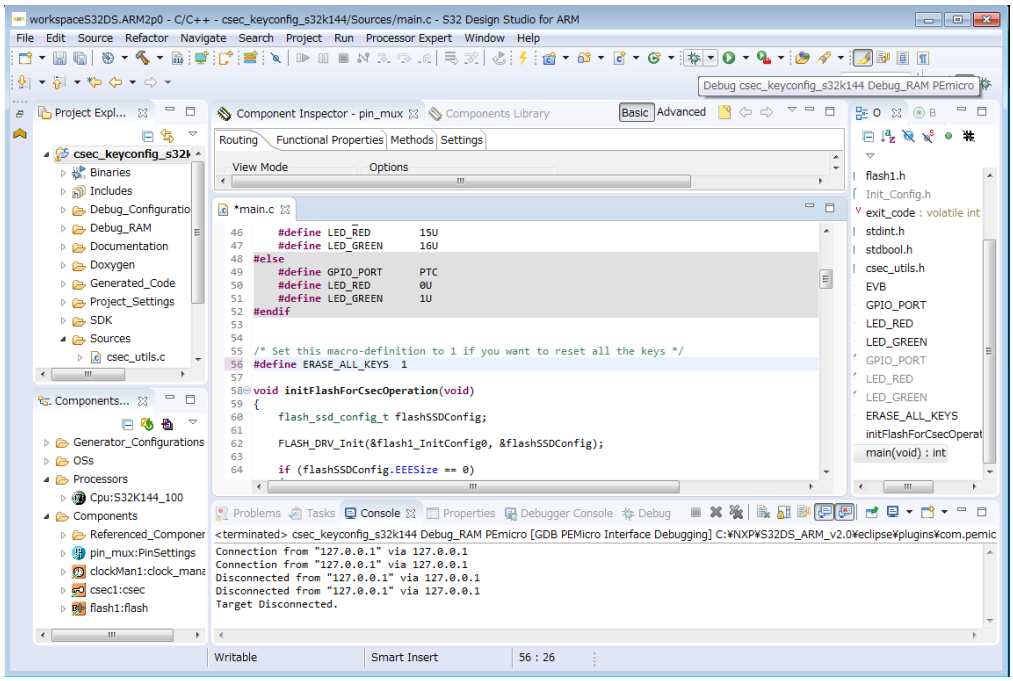




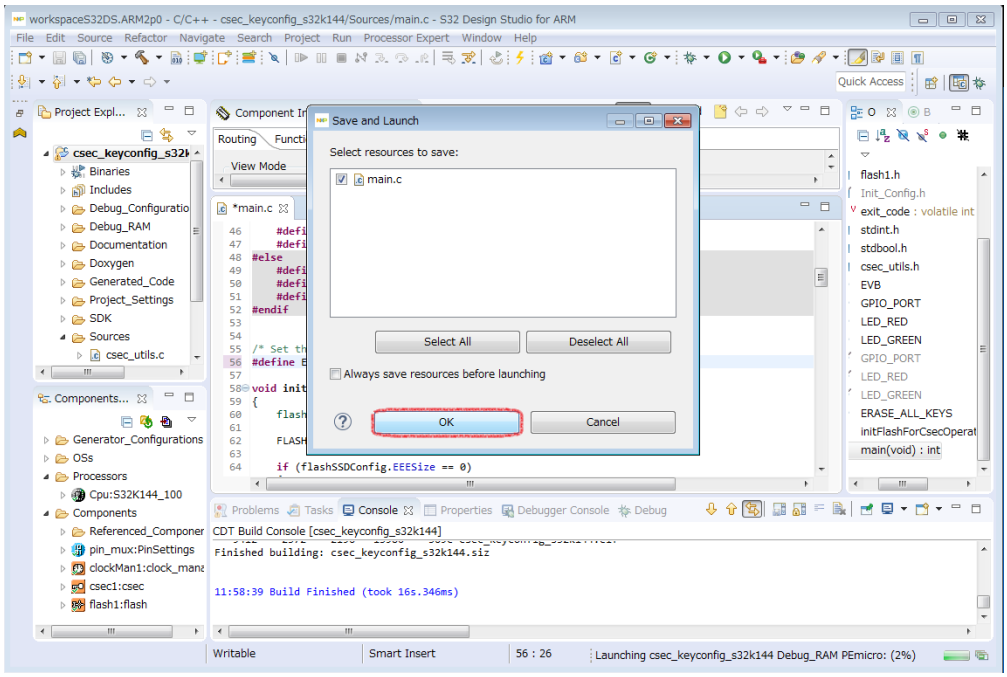
19.「Terminate」ボタンをクリックし、デバッガを終了する。



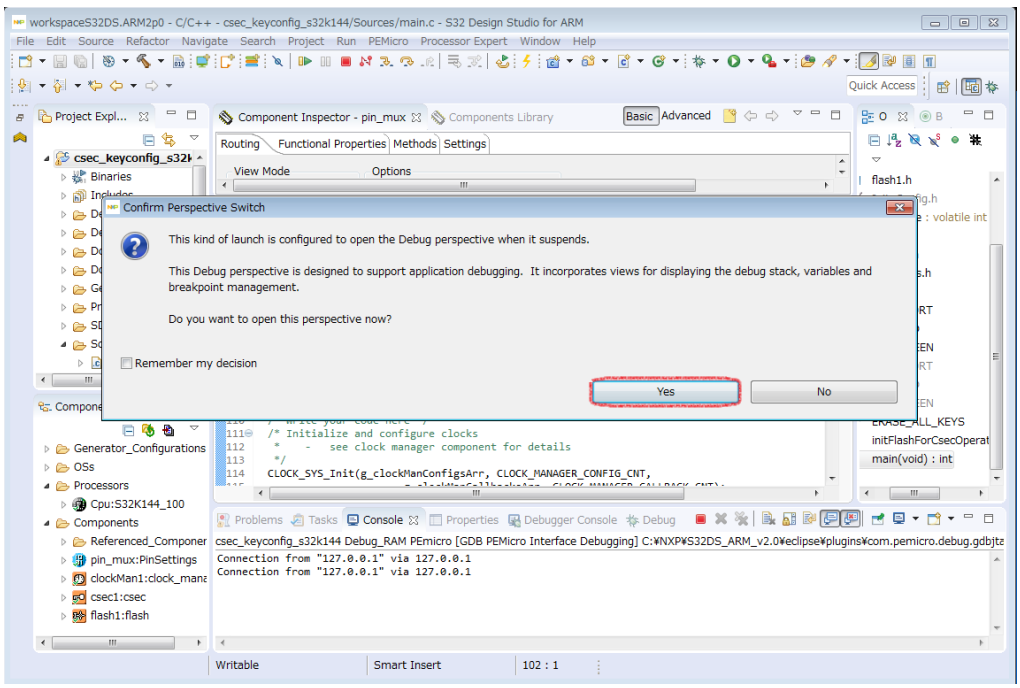
20. 次に、書き込んだ鍵を消去するため、main関数内56行目の
#define ERASE_ALL_KEYSのマクロ定義を “1” に変更し、虫マークをクリック。



21. 変更したソースをセーブするため、OKをクリック。



22.「Confirm Perspective Switch」ウインドウ内の“ Yes” をクリックしてデバッグ表示画面に移行する。



23.「Resume」ボタンをクリックする。これによりコードが一括実行され、
CSEc鍵格納メモリから鍵が消去される。