

ROM化の指定方法

RL78、78K0R用 Cコンパイラ CA78K0R

株式会社ルネサス ソリューションズ
ツールビジネス本部 ツール技術部

2014/6/20 Rev. 1.00

R20UT3041JJ0100

- ROM化について
- ROM化プロセッサの処理イメージ
- マイコンプログラム上での実行時イメージ
- ROM化するには
 - Cソースファイル
 - リンクディレクティブファイル
 - ROM化プロセッサの実行

ROM化について

■ 本資料でのROM化の対象は次になります。

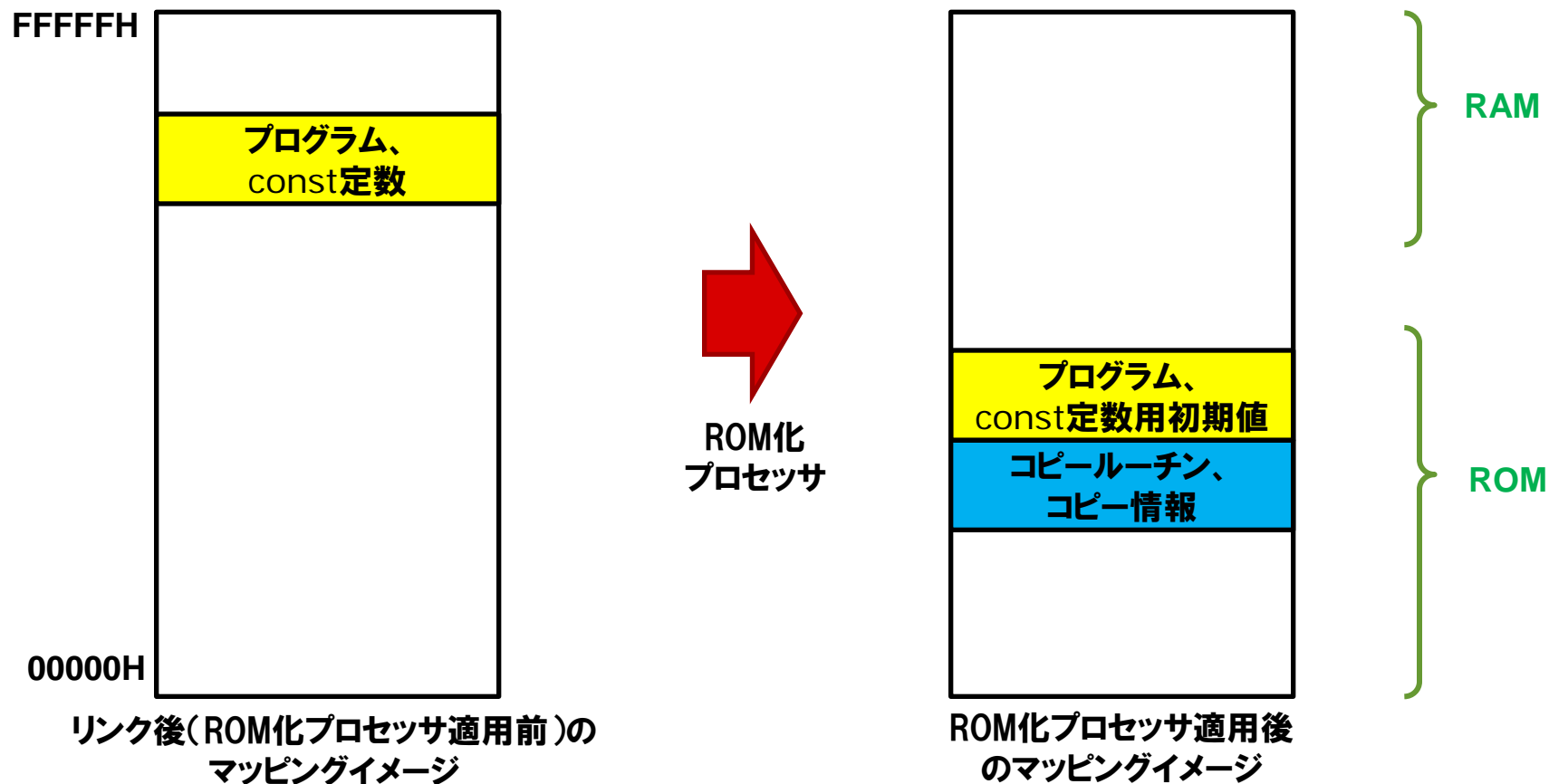
- RAMに配置するプログラム
- RAMに配置するconst定数
- 注意

– 初期値あり変数については、デフォルトのセクションに出力されるものは、コピー用のROMへの初期値の配置、その値のROMからRAMへのコピーは行われていますので、本資料の説明の対象外としています。

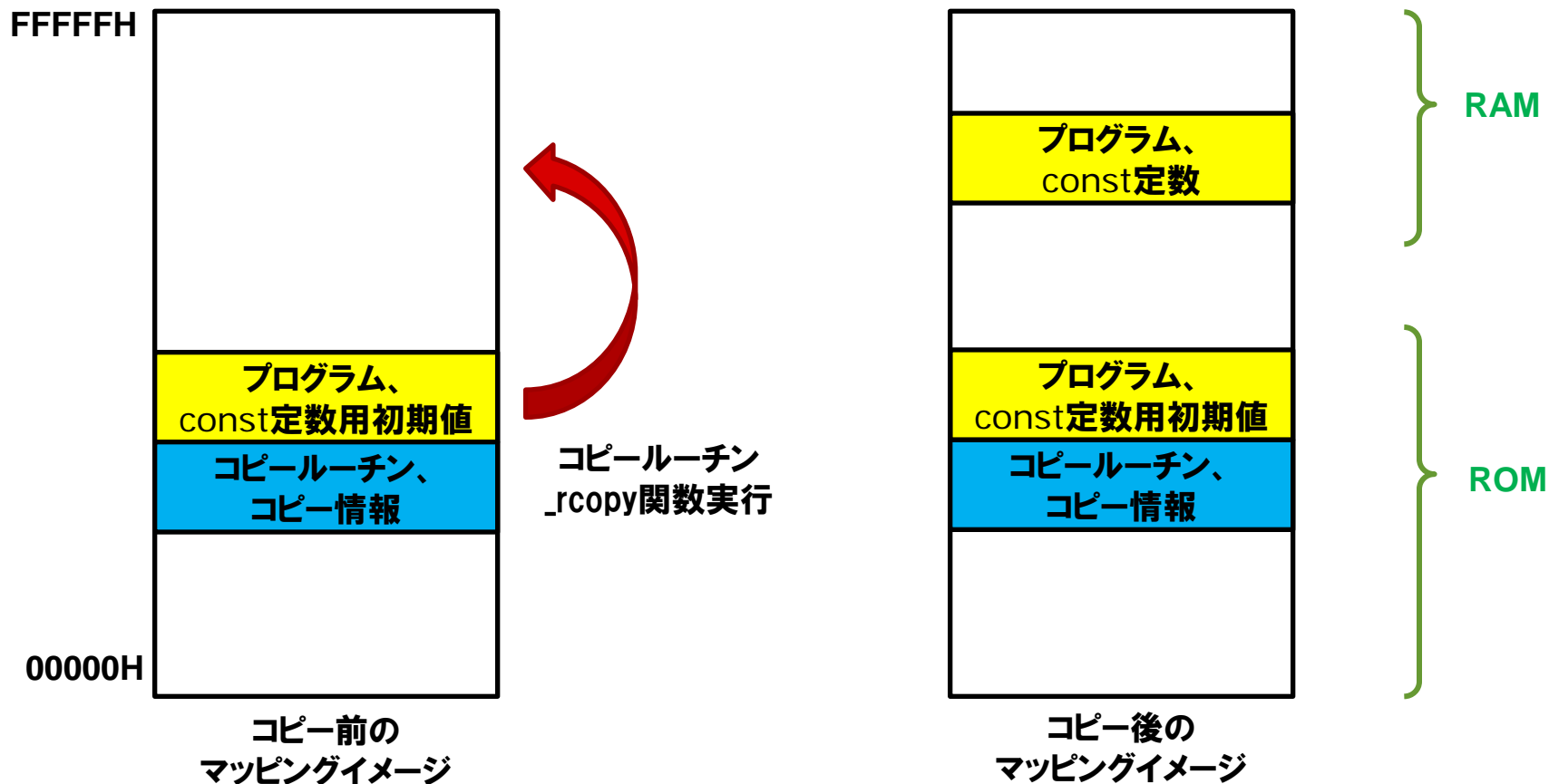
■ ROM化には、次の処理が必要になります。

- プログラム、const定数をRAMに配置する
- ROM化プロセッサで、RAMに配置したプログラム、const定数を、ROMへの配置を自動的に行い、ROMからRAMにコピーできるようにする
- プログラム中で、ROMからRAMにコピーするルーチンを追加する

ROM化プロセッサの処理イメージ



マイコンプログラム上での実行時イメージ(ROM化プロセッサ適用後)



ROM化するには

■ Cソースファイル

- **プログラム、const定数をRAM配置の設定にする**
 - #pragma section でセクション名を変更する(詳細な配置指定が必要な場合)
 - 対象となるCソースファイルにオプションを指定する
- **プログラム中で、ROMからRAMにコピーするルーチンを追加する**
 - _rcopy関数の呼び出しを追加する

■ リンクディレクティブファイル

- **上記のセクションをRAMに配置する様指定する**
 - MERGEにて、セクションの配置するアドレスや領域を指定する

■ ROM化プロセッサの実行

- **RAMに配置したプログラム、const定数を、ROMからRAMにコピーできるようなオブジェクトを生成する**
 - ROM化プロセッサを実行するよう設定する
 - ROM化対象をオプションで指定する
 - ビルドの出力ファイルをROM化用オブジェクトに変更する

ROM化するには ~ Cソースファイル(1/5)

- #pragma section でセクション名を変更する(1)
 - 対象となるプログラム(関数)のデフォルトのセクション名をRAM配置用に任意の名前に変更する

```
#pragma section @@CODER RAM_CODE  
  
extern int x;  
void func(void)  
{  
    x++;  
}
```

デフォルトのセクション名

新たに作成したセクション名

■ デフォルトのプログラムのセクション名

セクション名	説明
@@CODE	コード部用セグメント(near領域配置)
@@CODEL	コード部用セグメント(far領域配置)
@@CODER	RAM配置コード部用セグメント

ROM化するには ~ Cソースファイル(2/5)

■ #pragma section でセクション名を変更する(2)

- const定数のデフォルトのセクション名をRAM配置用に任意の名前に変更する

(例)

```
#pragma section @@CNSTR RAM_CNST
```

```
const int c = 0xff;
```

デフォルトのセクション名

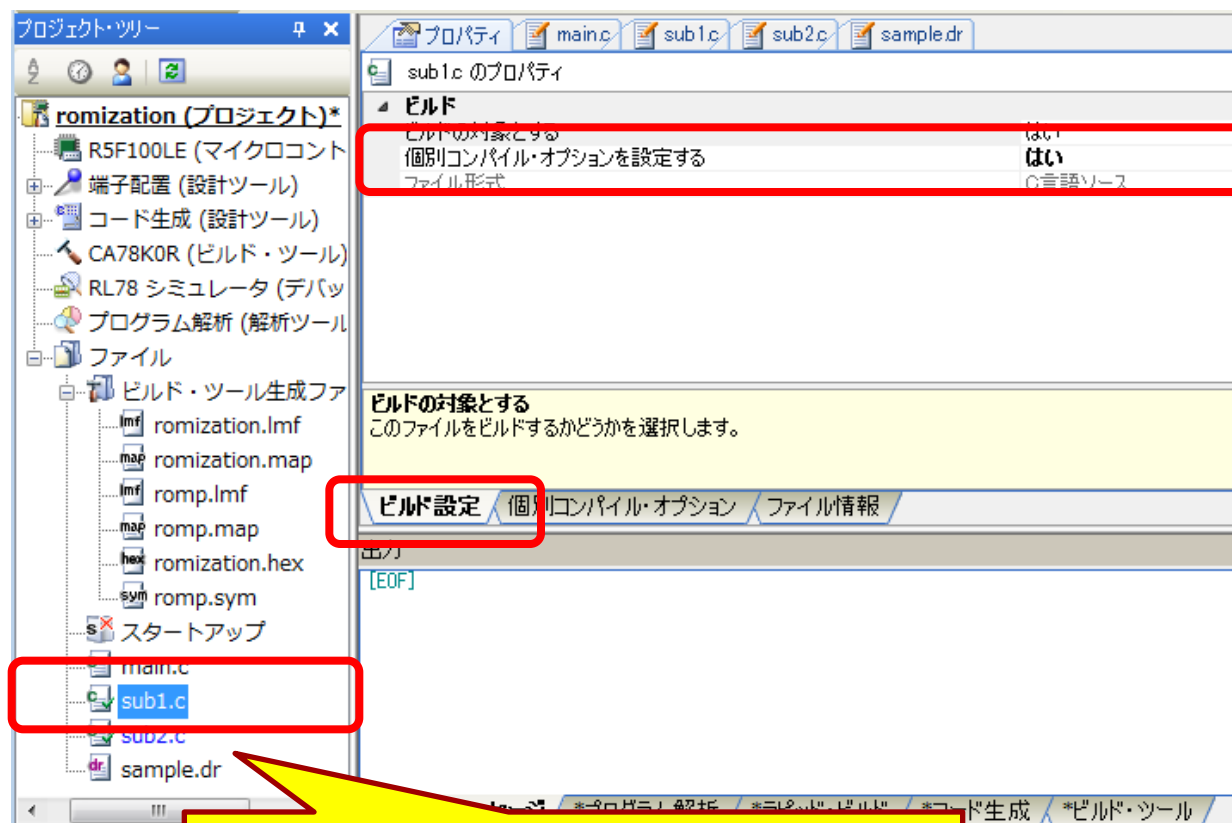
新たに作成したセクション名

■ デフォルトのプログラムのセクション名

セクション名	説明
@@CNST	ROMデータ(near領域配置)
@@CNSTR	RAM配置ROMデータ用セグメント(near領域配置)
@@CNSTL	ROMデータ(far領域配置)
@@CNSTLR	RAM配置ROMデータ用セグメント(far領域配置)

ROM化するには ~ Cソースファイル(3/5)

- 対象となるCソースファイルに、RAM配置用オプションを指定する
 - 対象ファイルの個別コンパイル・オプションを設定する



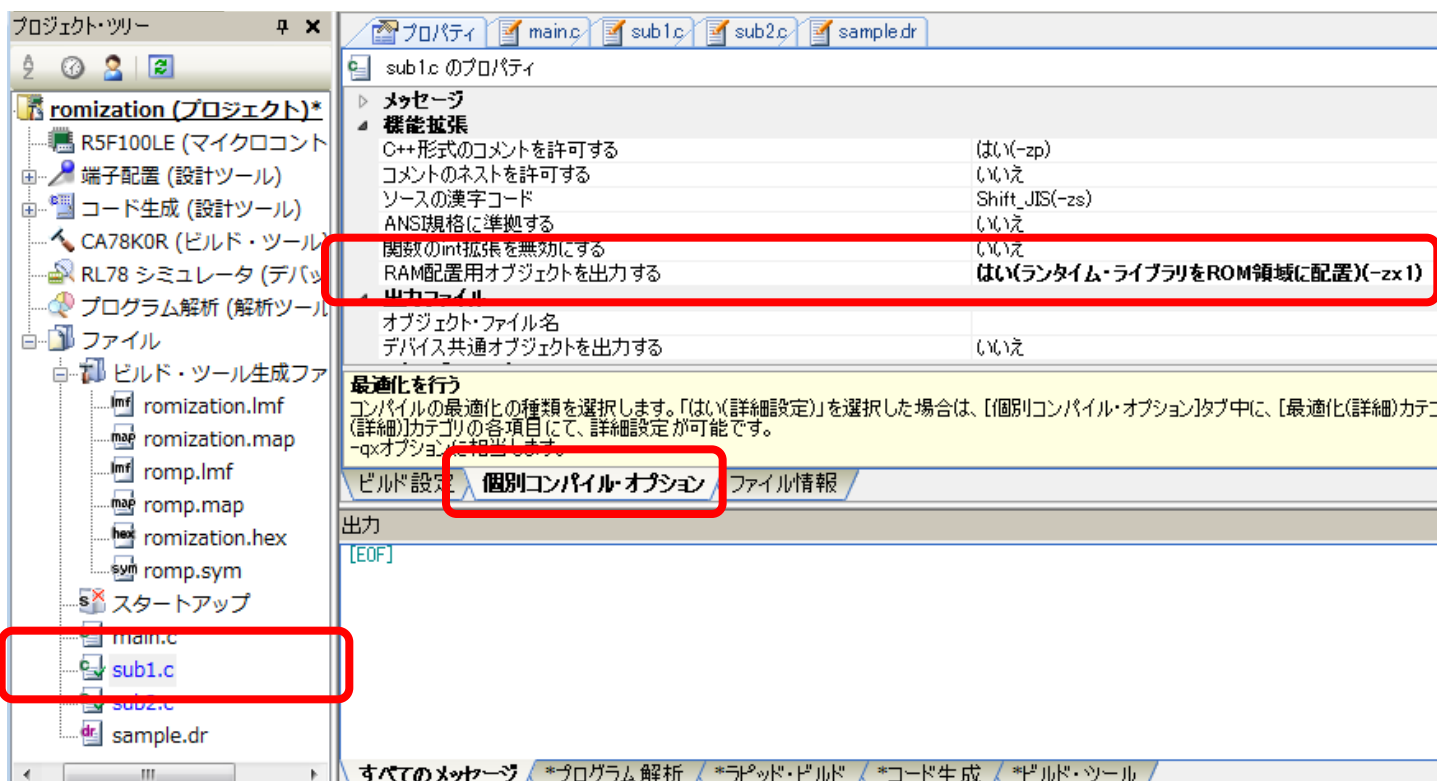
対象ファイルを選択して、
プロパティパネルを表示

ROM化するには ~ Cソースファイル(4/5)

■ 個別コンパイル・オプションの設定

● [個別コンパイル・オプション]タブの「拡張機能」

- 「RAM配置用オブジェクトを出力する」で次のいずれかを選択
 - はい(ランタイム・ライブラリをROM領域に配置(-zx1))
 - はい(ランタイム・ライブラリをRAM領域に配置(-zx2))



ROM化するには ~ Cソースファイル(5/5)

■ プログラム中で、ROMからRAMにコピーするルーチンを追加する

- ROMに配置する関数に、_rcopy関数の呼び出しを追加する

(例)

```
int _rcopy(int) ;  
  
void main(void)  
{  
    _rcopy(1);  
    _rcopy(2);  
};
```

プロトタイプ宣言

ROM化プロセッサのマップ情報

```
*** ROM processs segment map ***  
  
SEGMENT      SEGMENT  
NAME         NUMBER  
RAM_CNST     1  
RAM_CODE     2
```

セクション番号

コピールーチンの呼び出し

- ・引数にROM化されたセクションのセクション番号を指定する。セクション番号は、一度ビルドを実行し、ROM化プロセッサのマップ情報を参照する
- ・引数に -1 を指定した場合には、すべてのROM化されたセクションをコピーする。

ROM化するには ~ リンクディレクティブファイル(1/2)

- 配置アドレスを指定する場合には、リンクディレクティブファイルで指定

(例)

```
MERGE RAM_CODE : AT(OFEF00H)
MERGE RAM_CNST : AT(OFF000H)
```

- 専用のメモリ領域を作成し、そこに配置することも可能

(例)

```
MEMORY RAM_P:(OFEF00H,00200H)
MEMORY RAM:(OFF100H,00F00H)
MERGE RAM_CODE := RAM_P
MERGE RAM_CNST := RAM_P
```

ROM化するには ~ リンクディレクティブファイル(2/2)

■ リンクディレクティブファイルをCubeSuite+ のプロジェクトに登録

- CubeSuite+ のプロジェクトツリーパネルに、リンクディレクティブファイルを、ドラッグ&ドロップすれば登録できます

The screenshot displays the CubeSuite+ interface. On the left, the 'プロジェクト・ツリー' (Project Tree) shows a project named 'LinkDirective'. Underneath, there are various tools and files, including 'main.c'. In the center, the 'コード生成' (Code Generation) window shows a C program snippet:

```
1
2 unsigned int uia = 10, uib = 20;
3
4 void main(void)
5 {
6     unsigned int a;
7     a = uia + uib;
8 }
9
10 [EOF]
```

On the right, a file explorer shows a folder named 'DefaultBuild' containing three items: 'r178.dr' (DR File, 1 KB), 'LinkDirective.mtpj' (MTPJ File, 151 KB), and another folder named 'DefaultBuild'. A red arrow originates from the 'LinkDirective.mtpj' file and points to the 'main.c' file in the project tree, illustrating the drag-and-drop registration process.

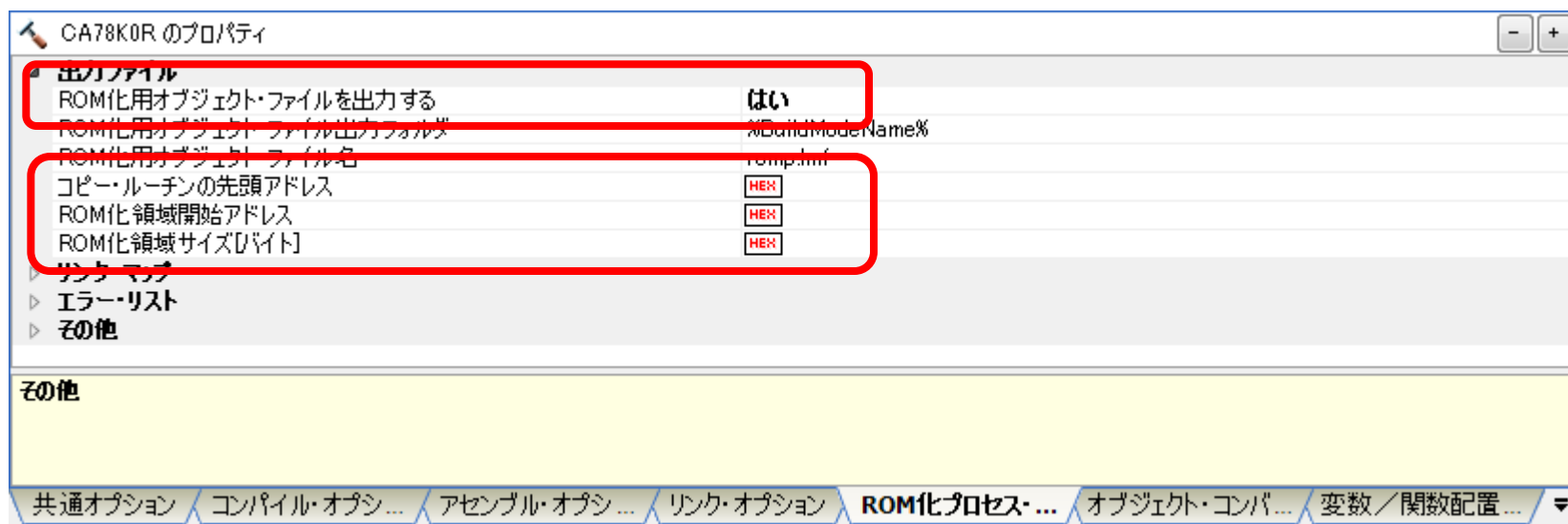
The bottom window shows the output of a build process:

```
出力
計ツール)に反映できません。ビルド・オプションまた
----- 全リビルドの開始(2011年6月27日 11:53:22
----- ビルド開始(LinkDirective, DefaultBuild) ---
>main.c
>DefaultBuild#LinkDirective.lmf
>DefaultBuild#LinkDirective.hex
----- ビルド終了(エラー:0個, 警告:0個) -----
----- 終了しました(成功:1プロジェクト, 失敗:0
-----
↓
[EOF]
```

ROM化するには ~ ROM化プロセッサの実行(1/2)

■ ビルドツールのプロパティで次の設定を行う

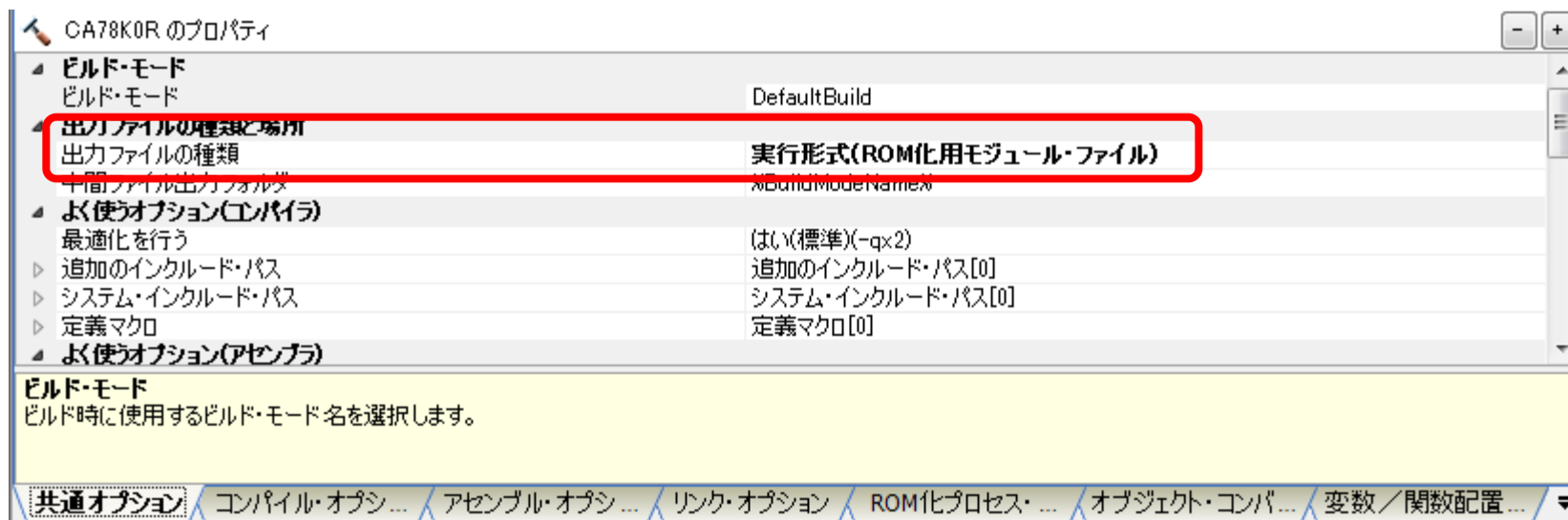
- [ROM化プロセス・オプション]タブの「出力ファイル」
 - 「ROM化用オブジェクト・ファイルを出力する」で「はい」を選択
 - 「コピー・ルーチンの先頭アドレス」を空欄にする(自動でアドレスが決まる)
 - 「ROM化領域開始アドレス」を空欄にする(内蔵RAMが範囲となる)
 - 「ROM化領域サイズ[バイト]」を空欄にする(内蔵RAMが範囲となる)



ROM化するには ~ ROM化プロセッサの実行(2/2)

■ ビルドツールのプロパティで次の設定を行う

- [共通オプション]タブの「出力ファイルの種類と場所」
 - 「出力ファイルの種類」で次の項目を選択
 - 実行形式(ROM化用モジュール・ファイル)



RENEASAS

ルネサス ソリューションズ株式会社

© 2014 Renesas Solutions Corp.