Rev 1.10.00

e2studio(GNU)ツールチェインの設定と必要事項の説明

（ルネサス　ＲＺ／Ｔ１用）

e2studio(GNU)ツールチェインの設定方法とサンプルプロジェクトに必要な設定を説明します。

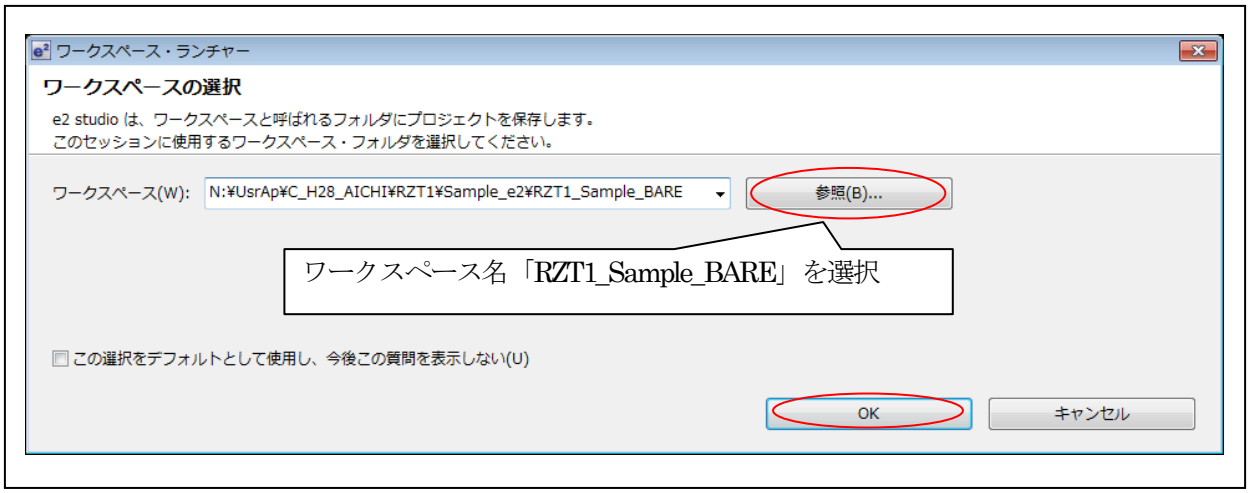
下記５通りのサンプルプロジェクトがありますが、toolchain の設定例は「RZT1_Sample_BARE」をもとに説明を進めます。なお、他のサンプルの相違は、各サンプル用アプリケーションノートを参照して下さい。

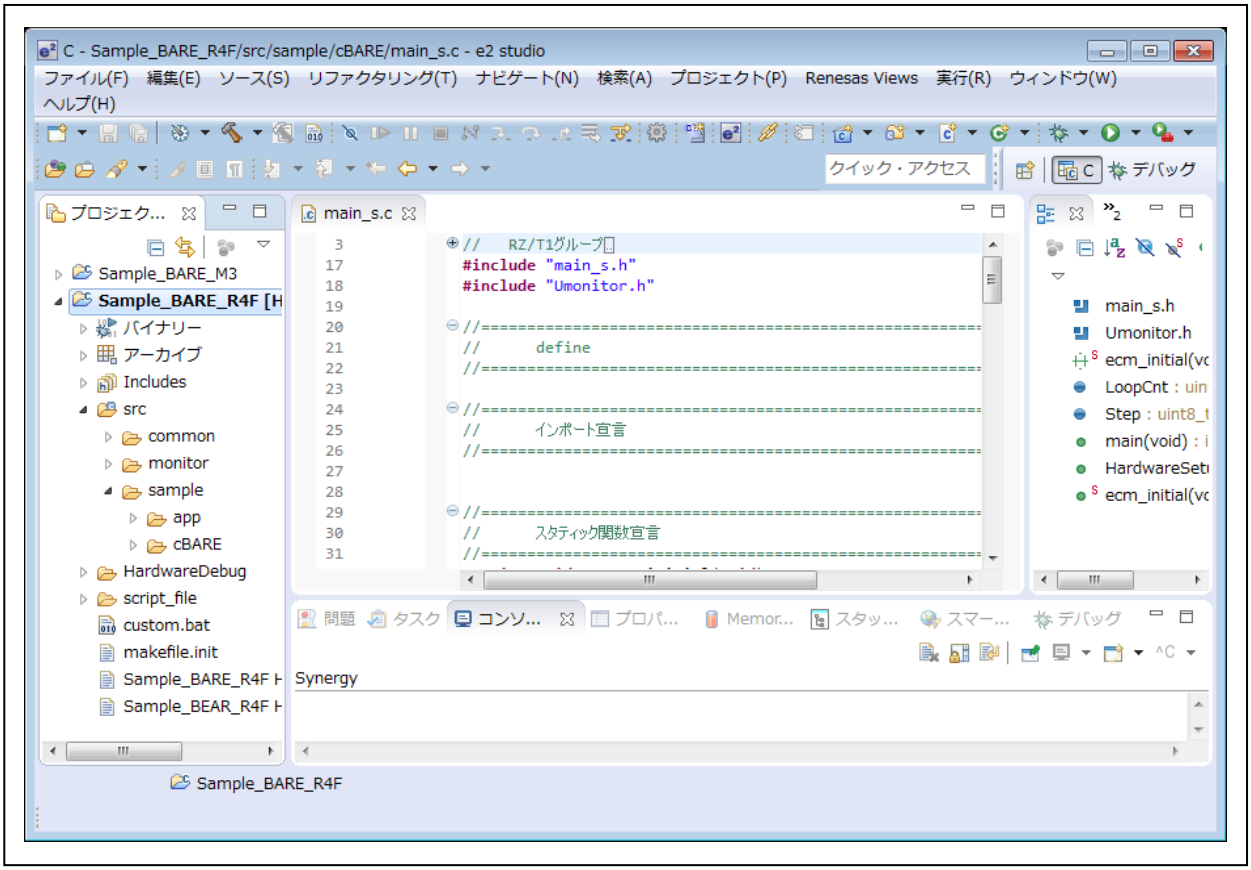
サンプル名	用途	ワークスペース名	コア	プロジェクト名
RZT1_Sample_BARE	ベアメタル版サンプル	RZT1_Sample_BARE	M3	Sample_BARE_M3
			R4F	Sample_BARE_R4F
RZT1_Sample_HWRTOS	M3 側で HWRTOS を使用したサンプル	RZT1_Sample_HWRTOS	M3	Sample_HWRTOS_M3
			R4F	Sample_HWRTOS_R4F
RZT1_Sample_ECAT	EtherCAT 制御サンプル	RZT1_Sample_ECAT	M3	Sample_ECAT_M3.mot を使用（ソース非公開）
			R4F	Sample_ECAT_R4F
RZT1_Demo_BARE	ベアメタル版デモソフト	RZT1_Demo_BARE	M3	Sample_ECAT_M3.mot を使用（ソース非公開）
			R4F	Demo_BARE_R4F
RZT1_Demo_NORTi	RTOS-NORTi 版デモソフト	RZT1_Demo_NORTi	M3	Sample_ECAT_M3.mot を使用（ソース非公開）
			R4F	Demo_NORTi_R4F

注記
コア【M3】側 EtherCAT®制御用サンプルのソース公開を希望される場合は、JSL Technology 社とのライセンス契約が必要です。

1．e2studio を起動する。

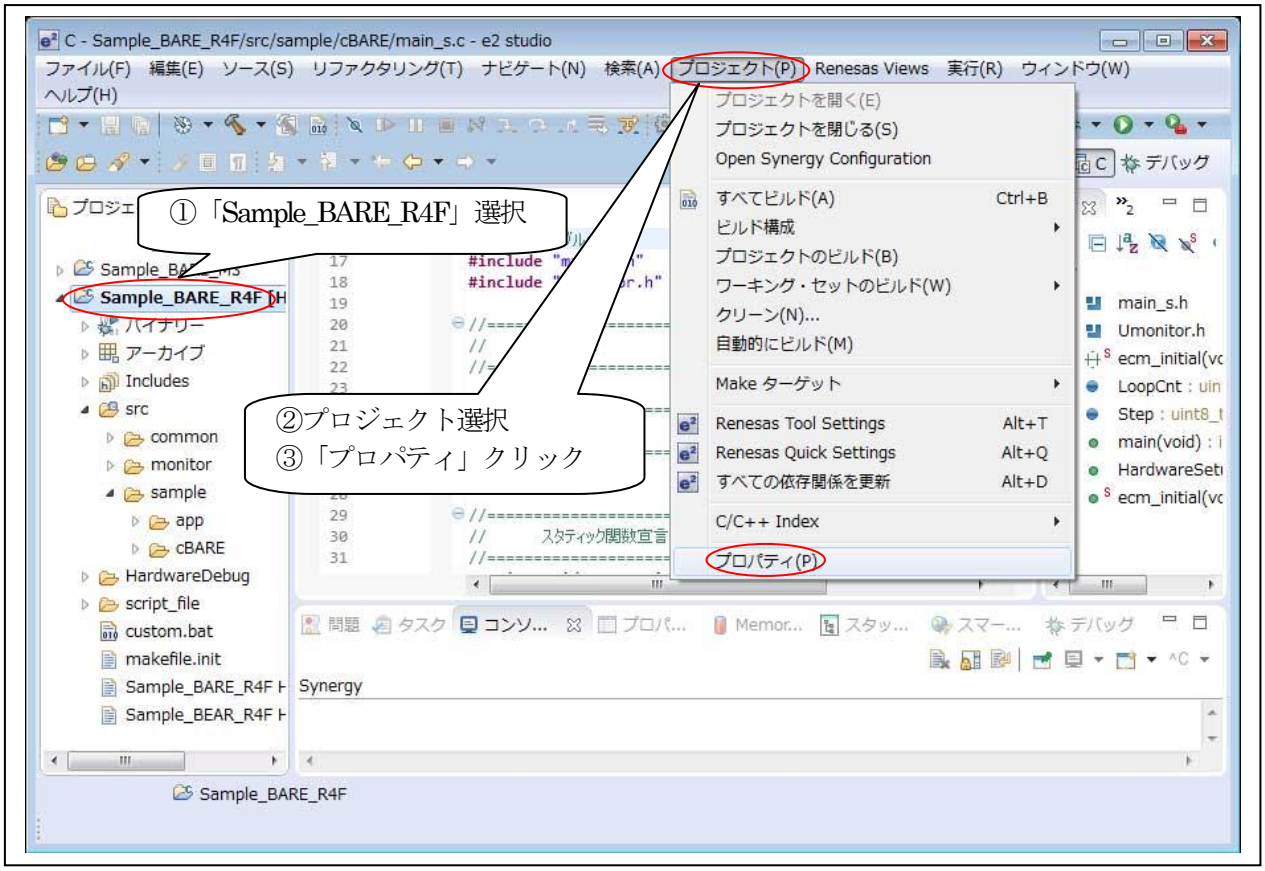
1－1．ワークスペースを選択する。





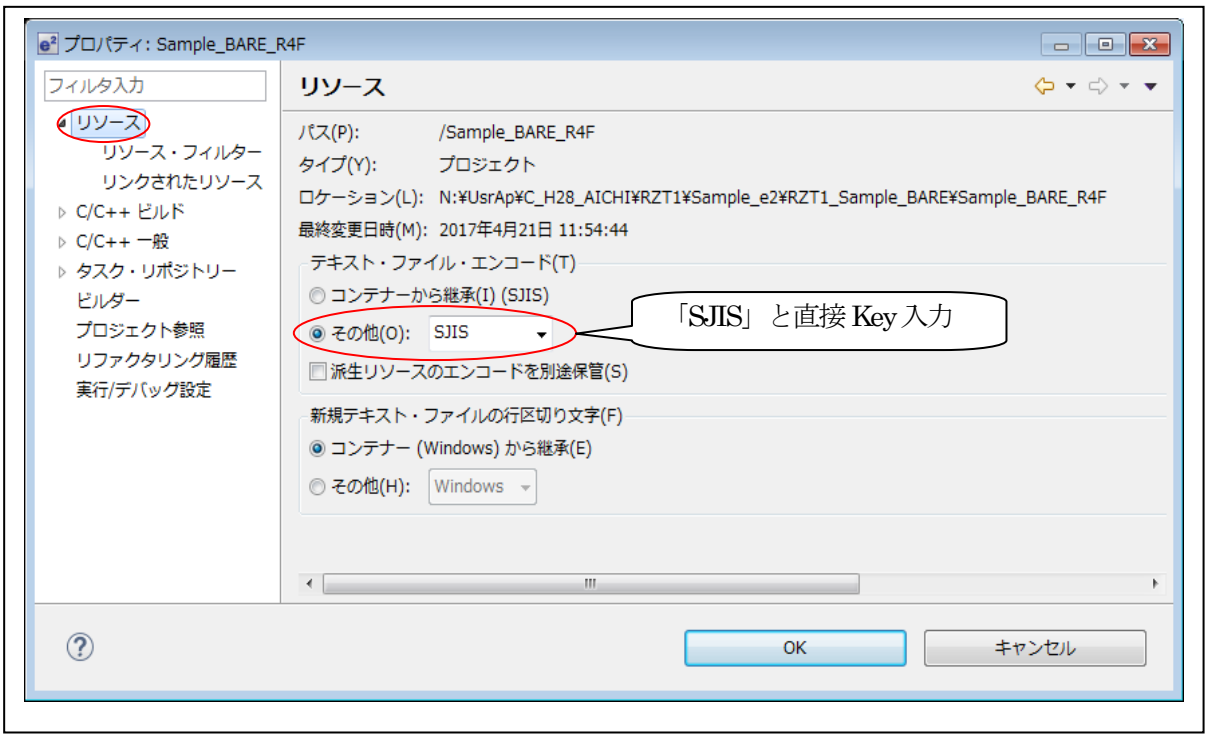
2．各ツールの設定内容を確認する。

2－1．コア【R4F】側の確認



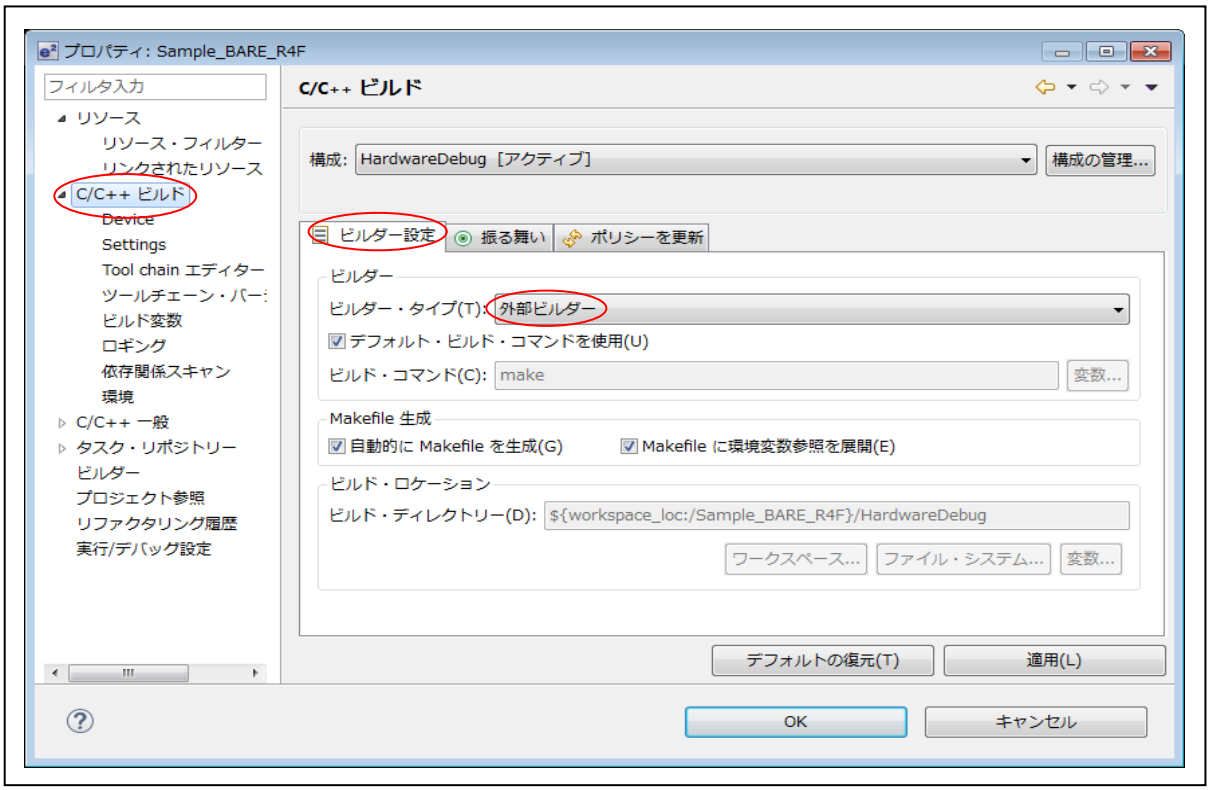
2－1－1．リソース

1）テキスト・ファイル・エンコード

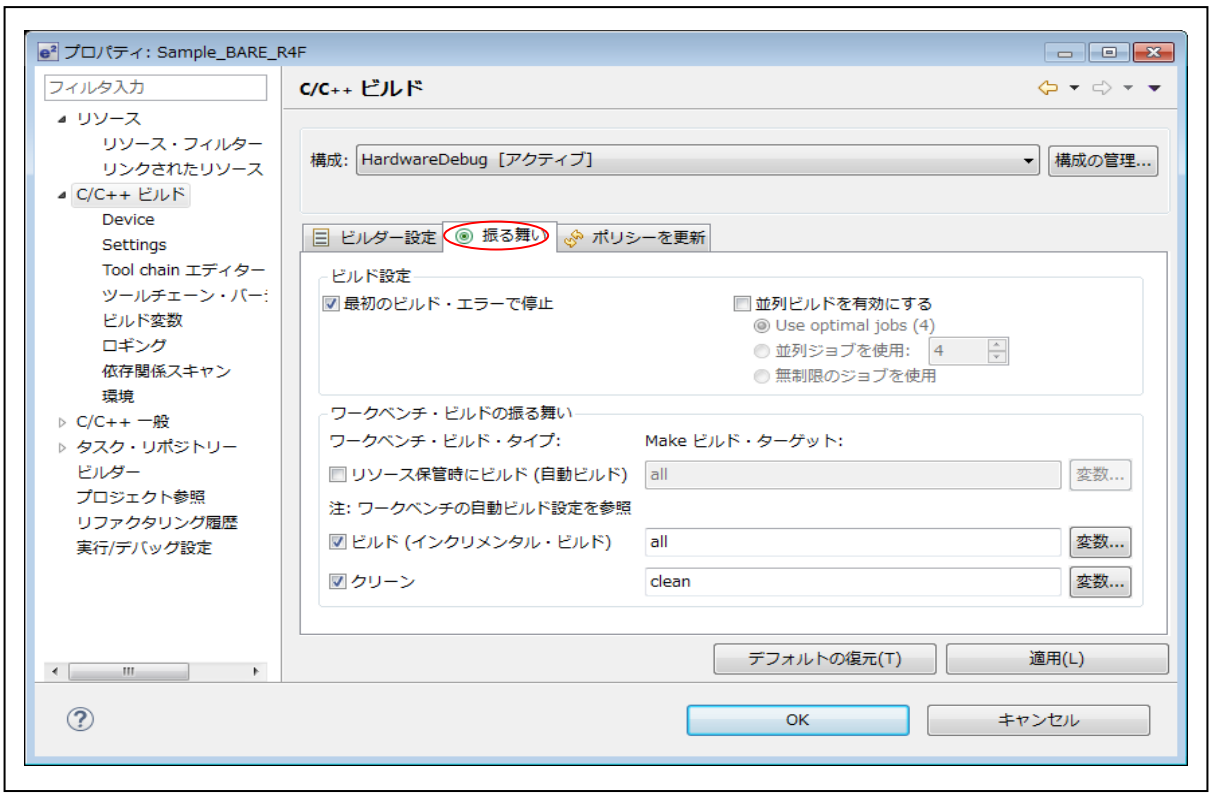


２－１－２．C/C++ビルド（TOP）

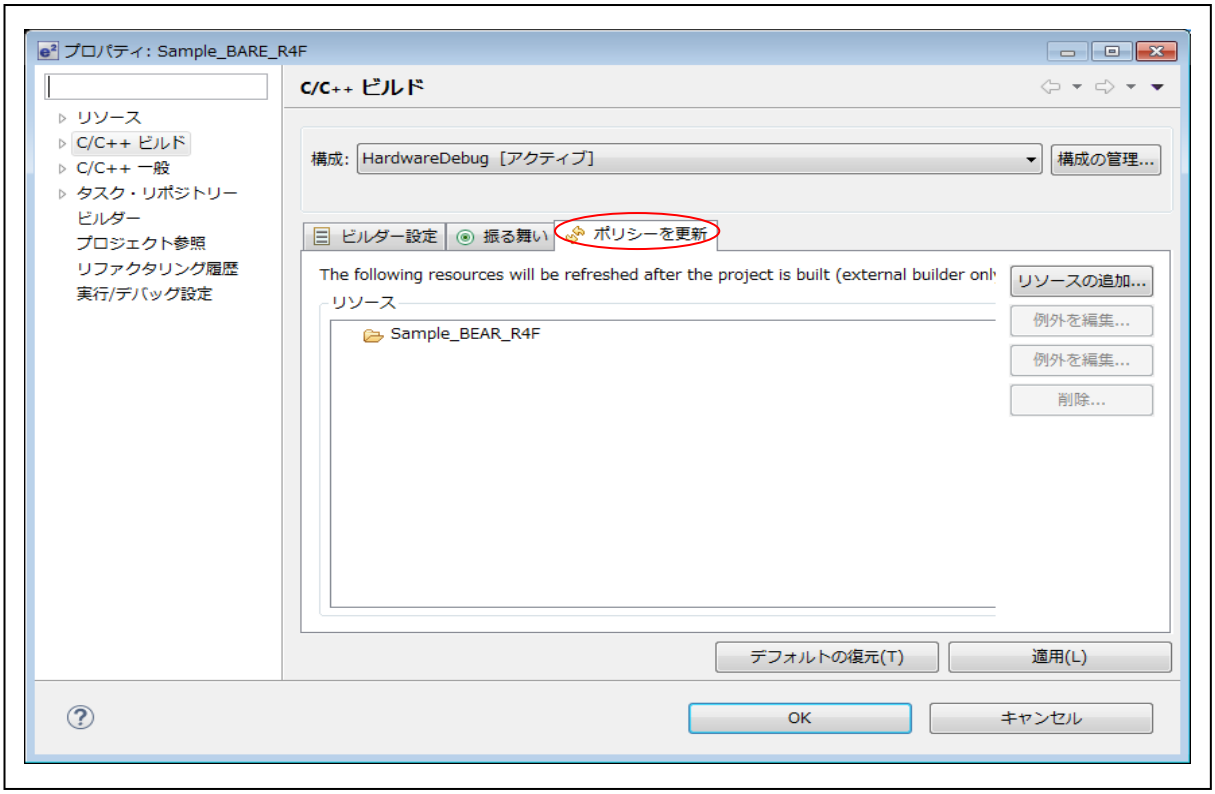
１）ビルダー設定（デフォルト）



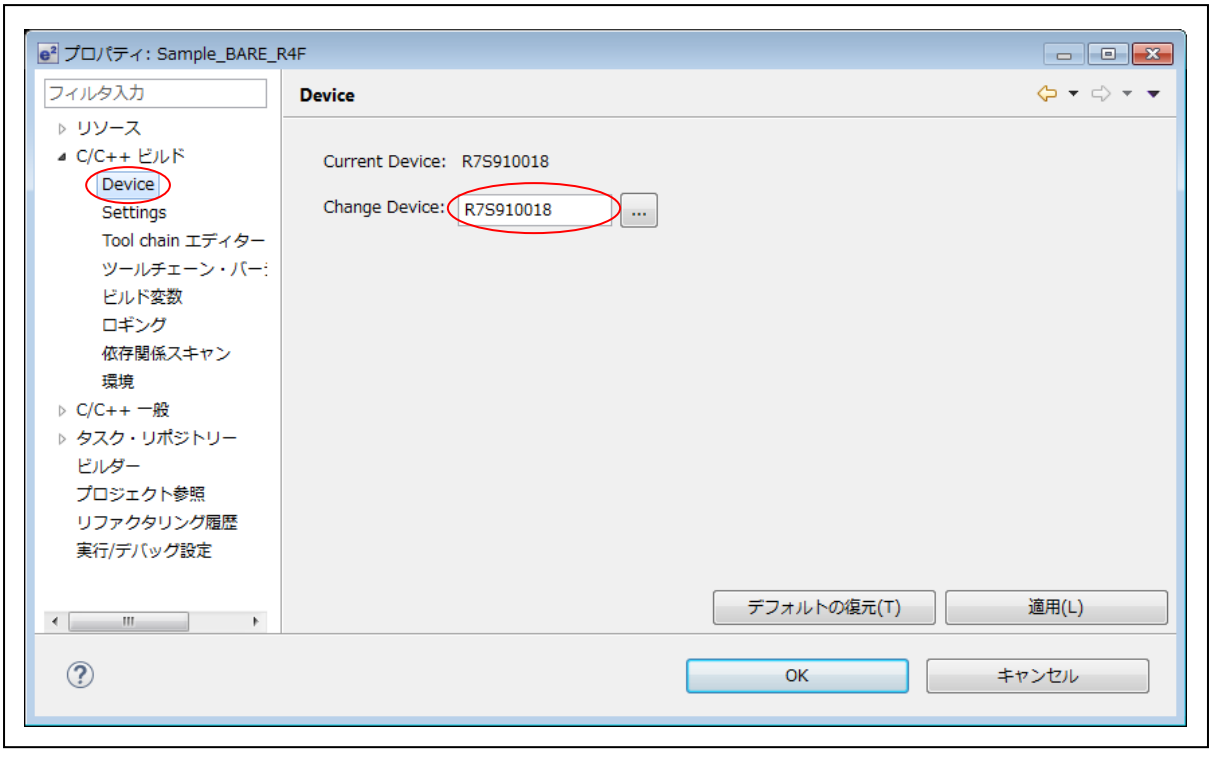
２）振る舞い（デフォルト）



3）ポリシーを更新（デフォルト）

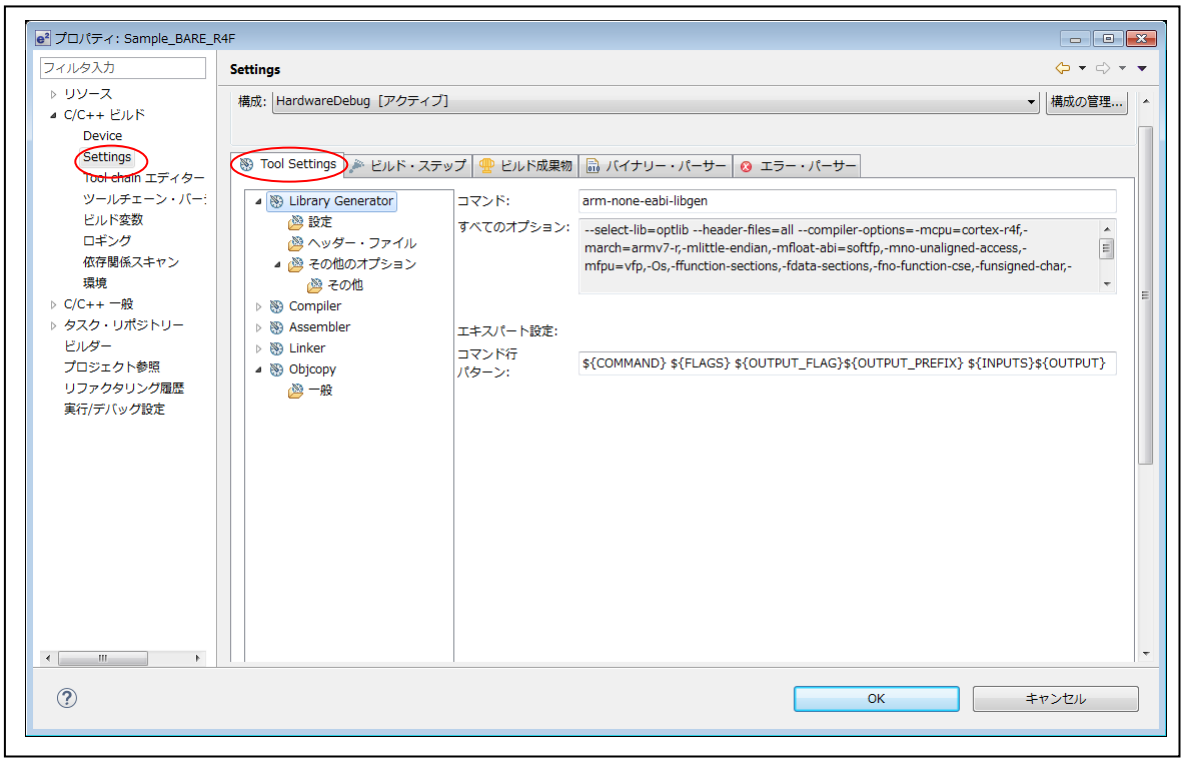


2－1－3．C/C++ビルド（Device）

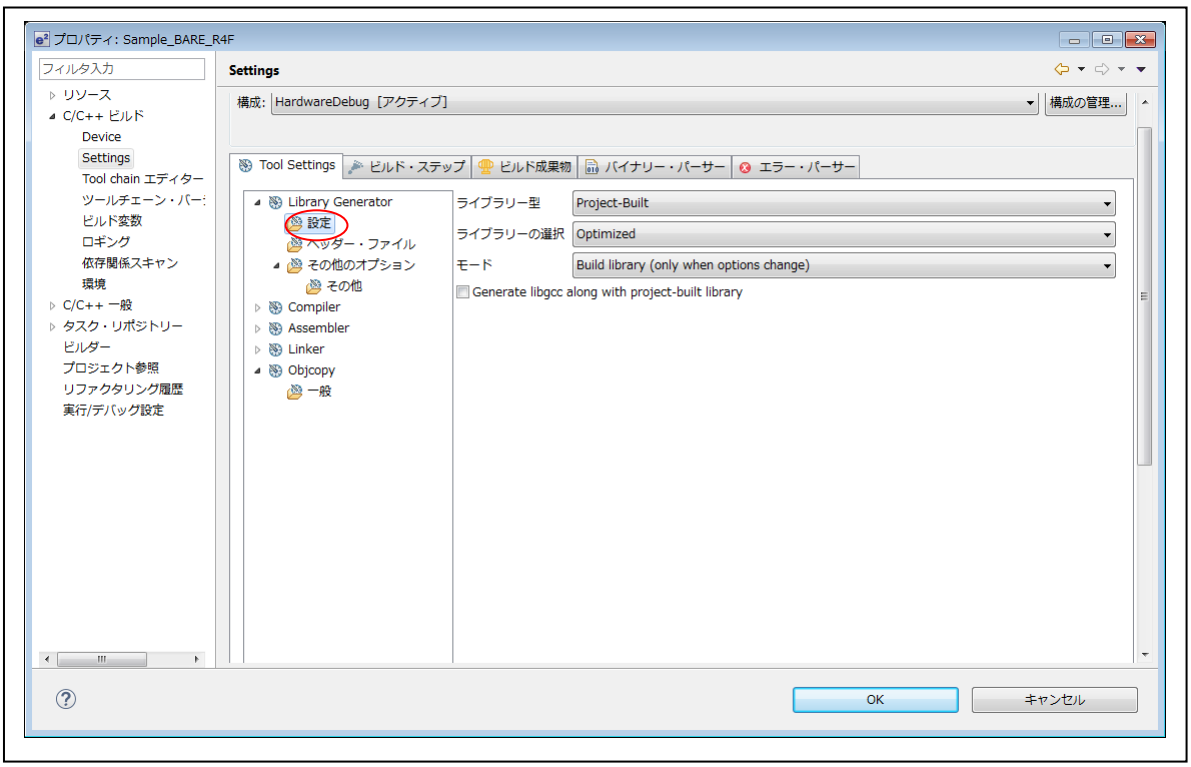


２－１－４．C/C++ビルド（Settings）

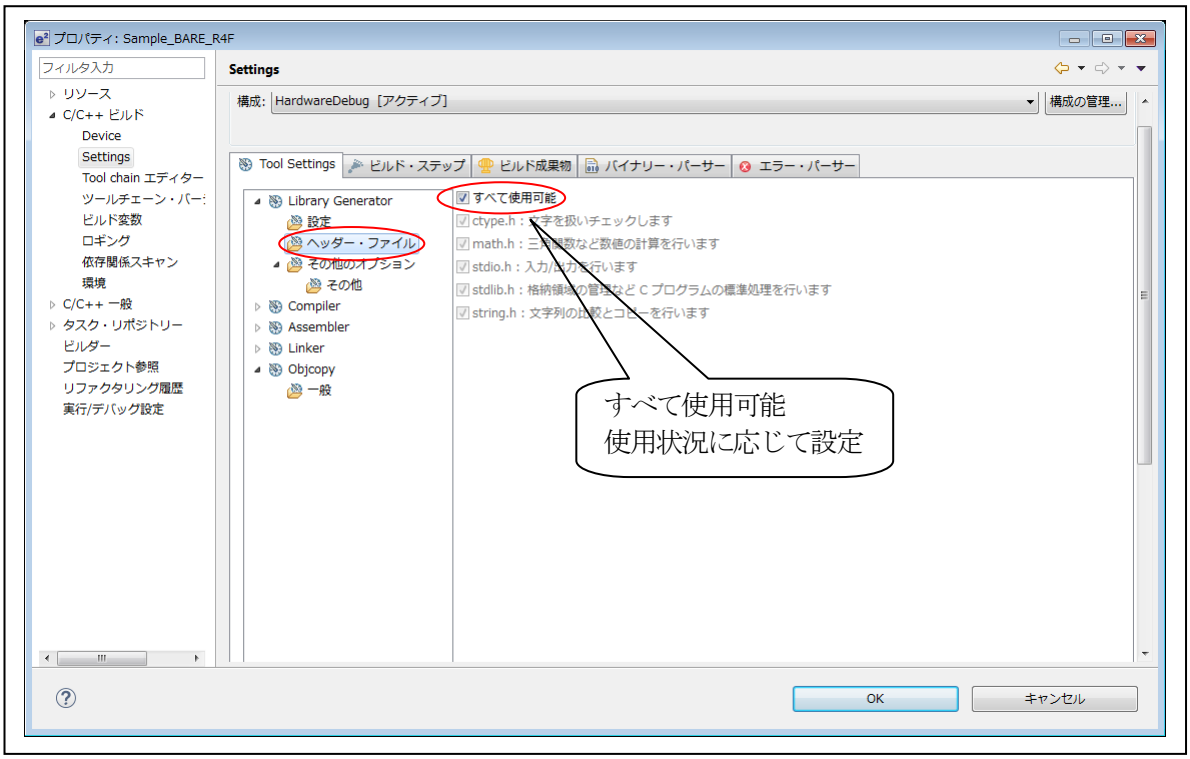
２－１－４－１．Tool　Settings(Library Generator)



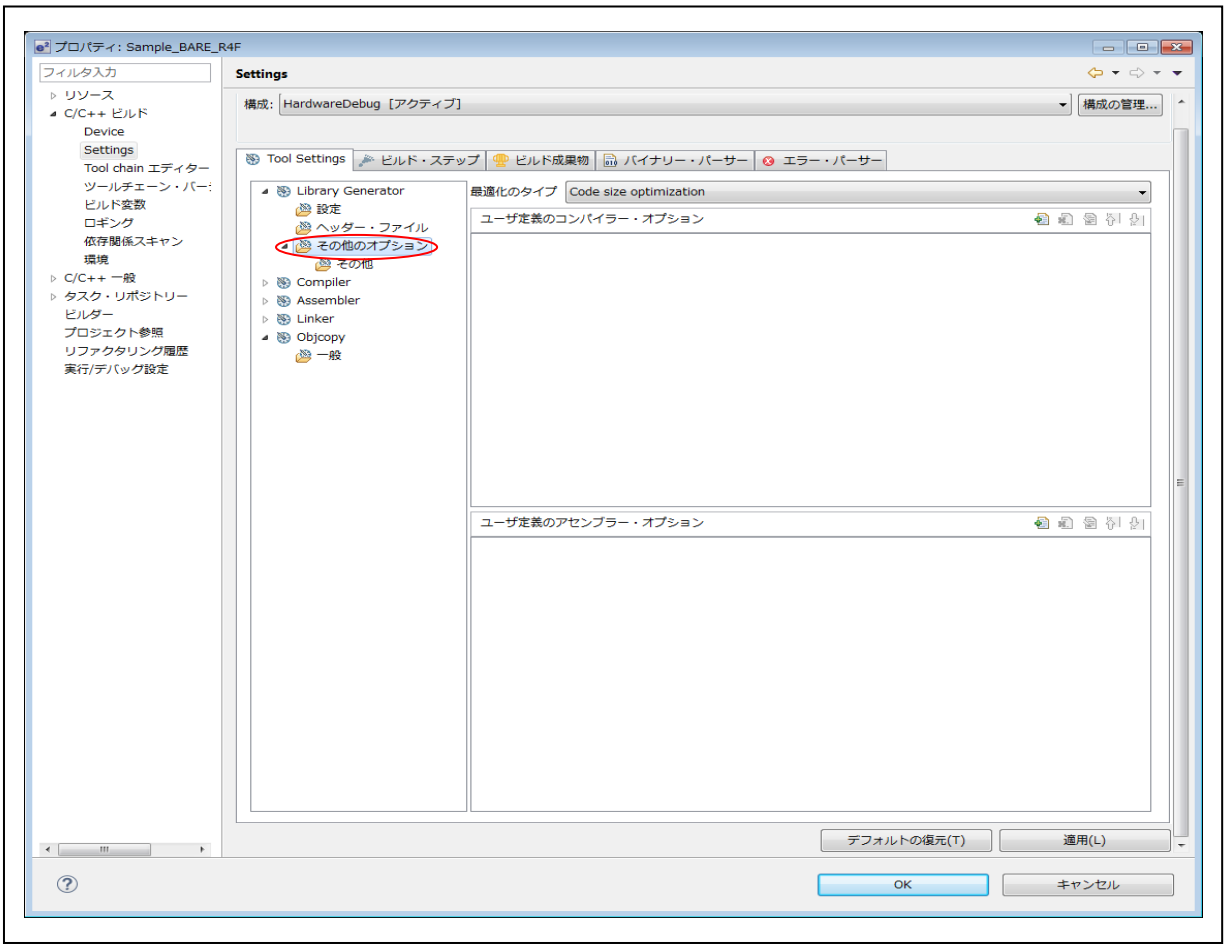
1）設定（デフォルト）



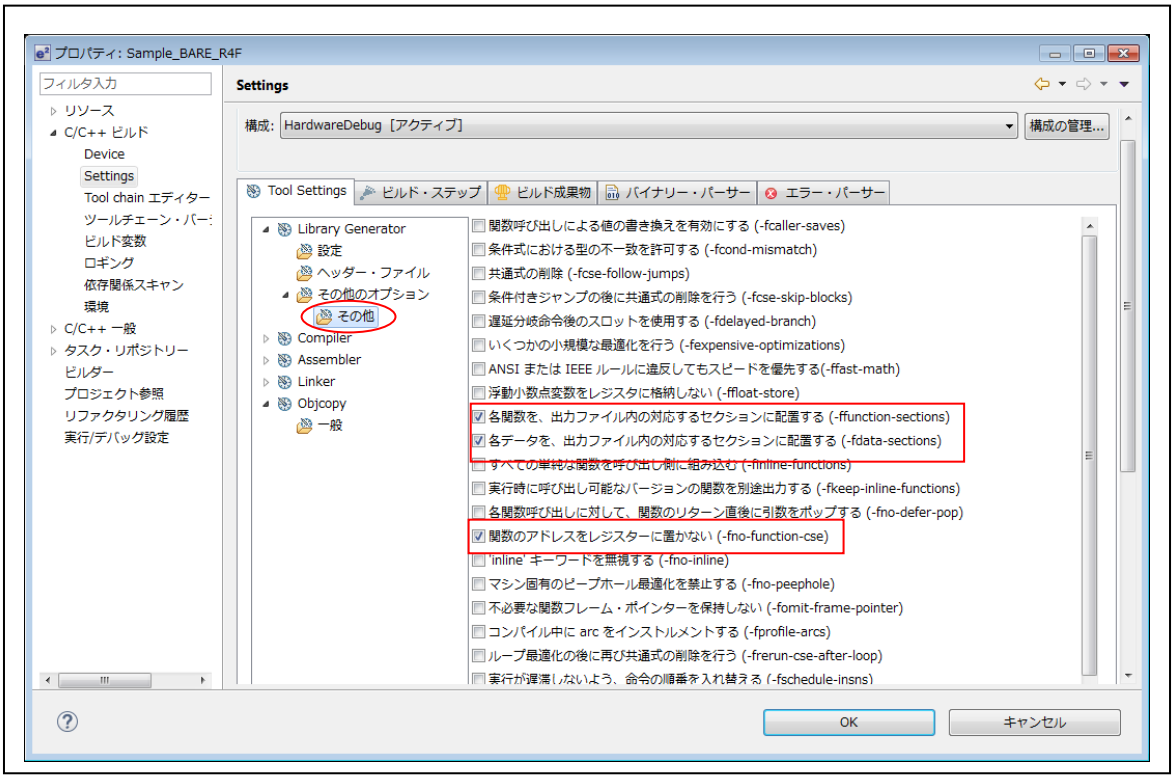
2）ヘッダー・ファイル



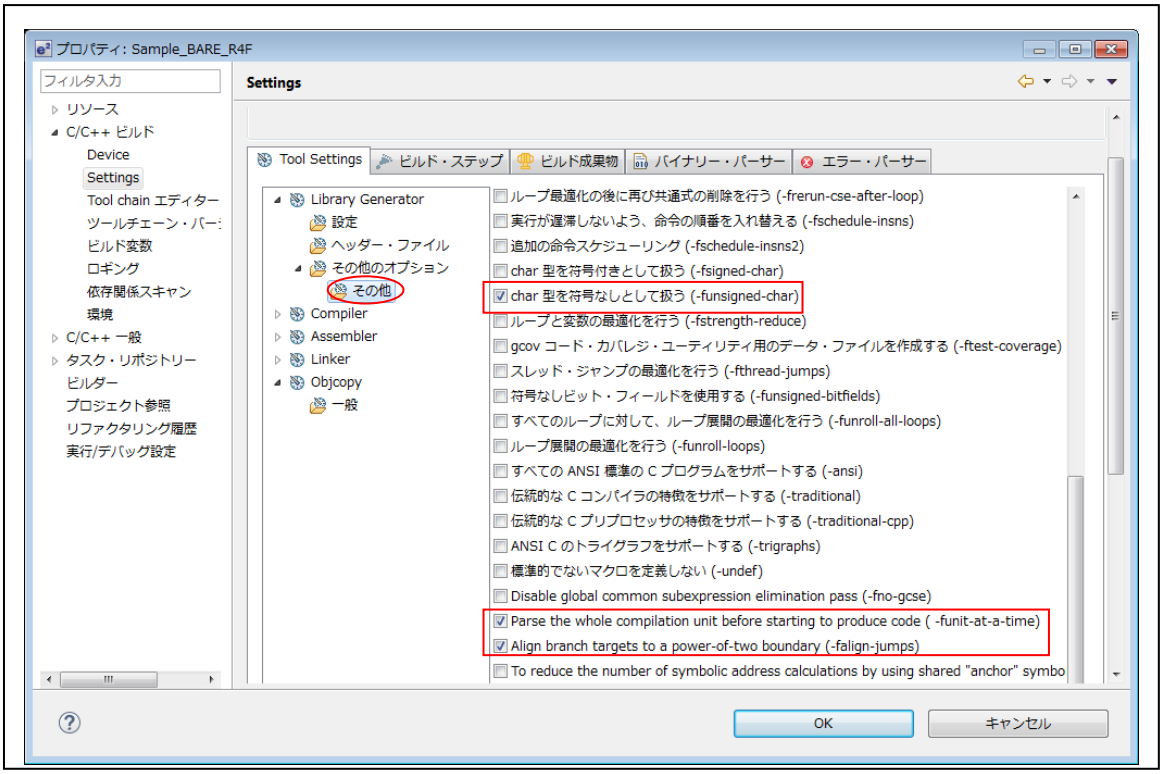
3）その他のオプション（デフォルト）



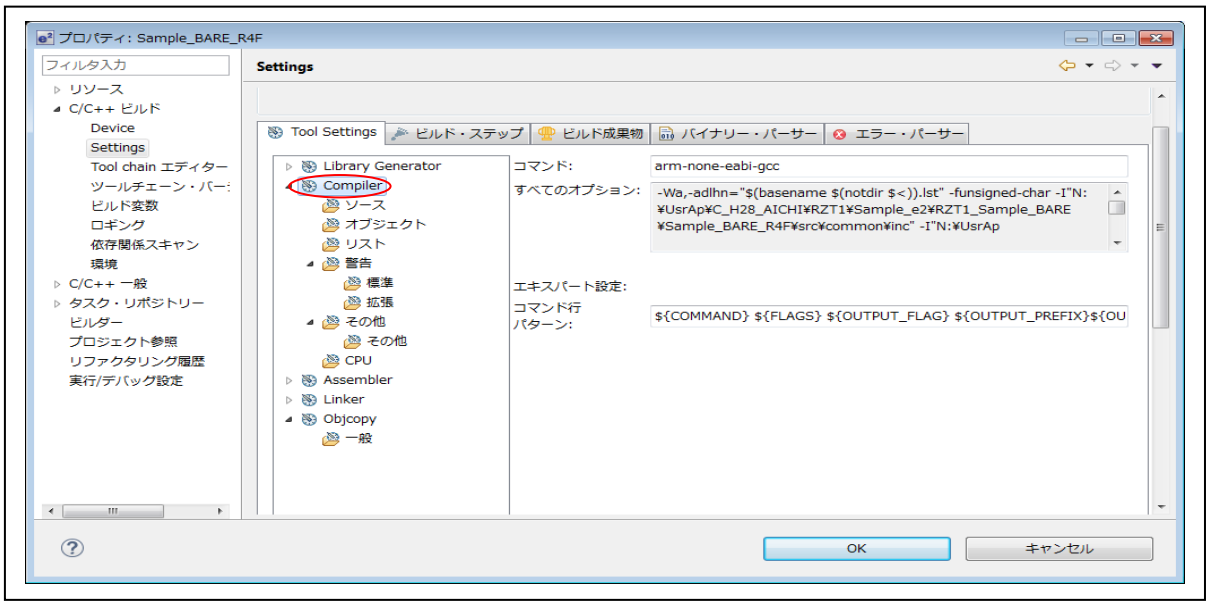
4）その他（1／2）



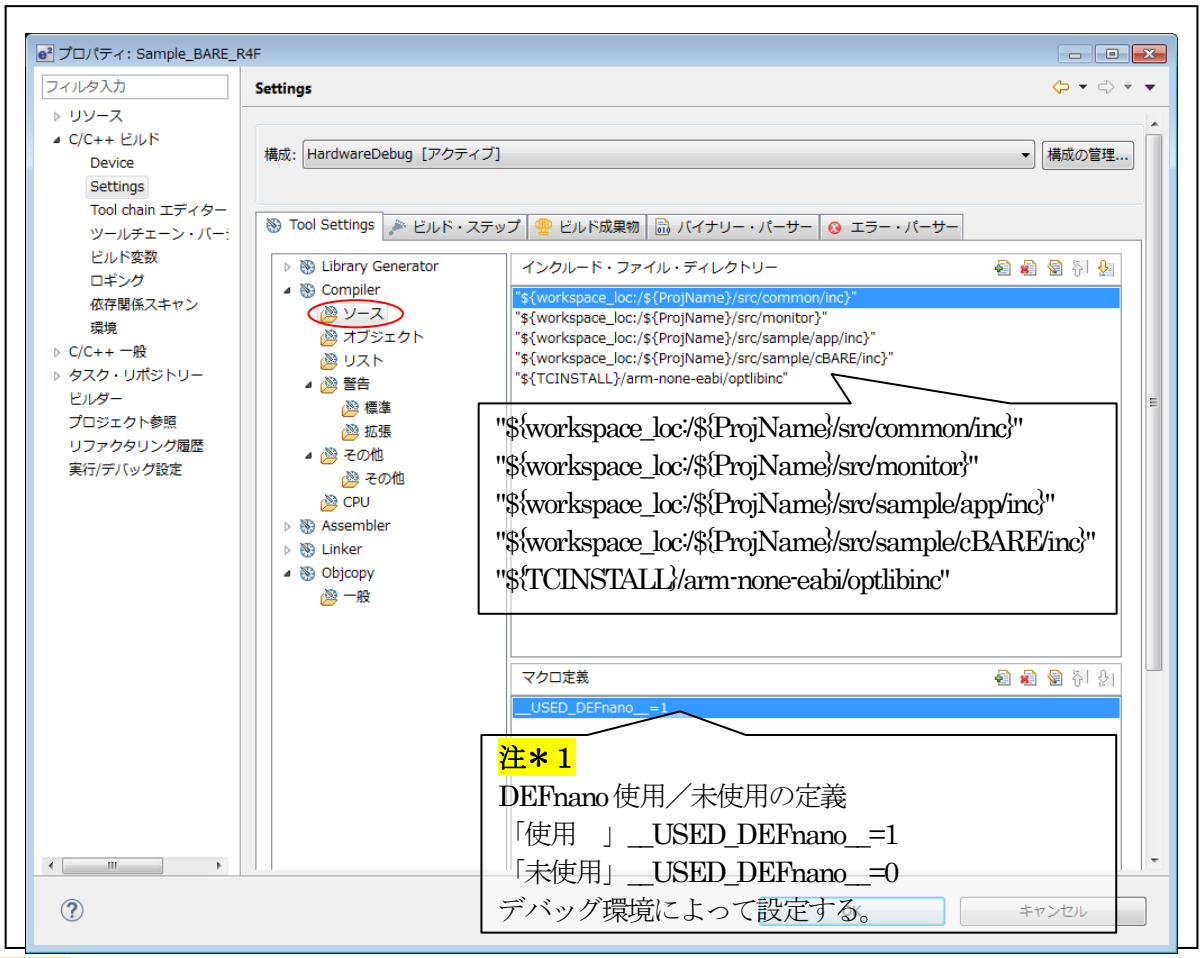
5）その他（2／2）



2－1－4－2．Tool　Settings(Compiler)



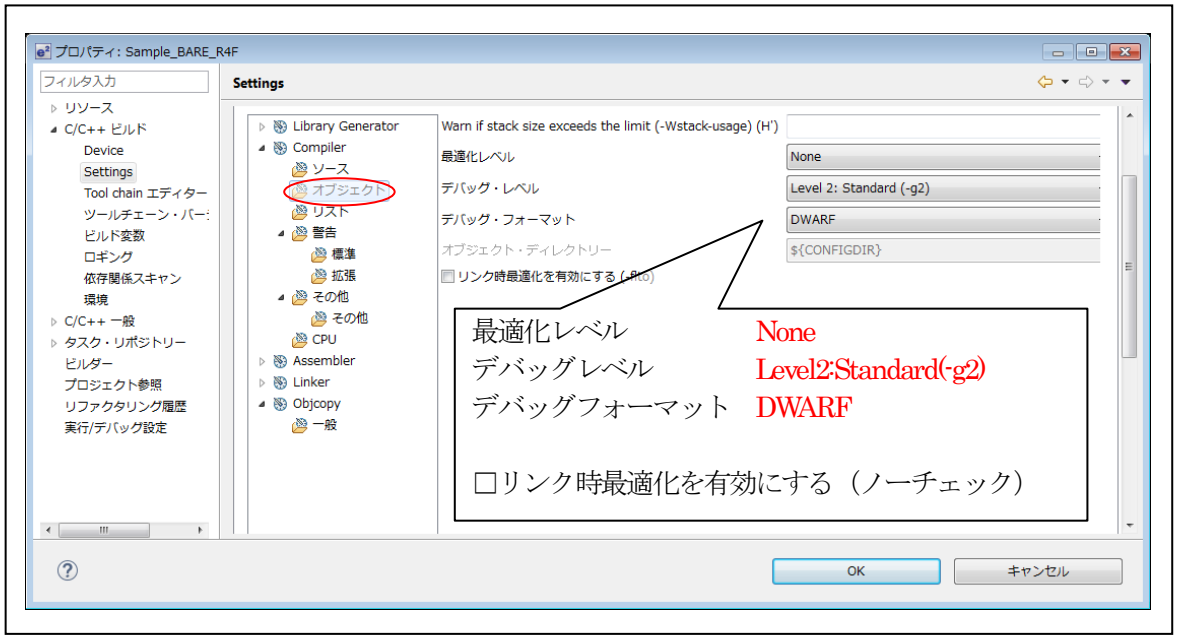
1）ソース



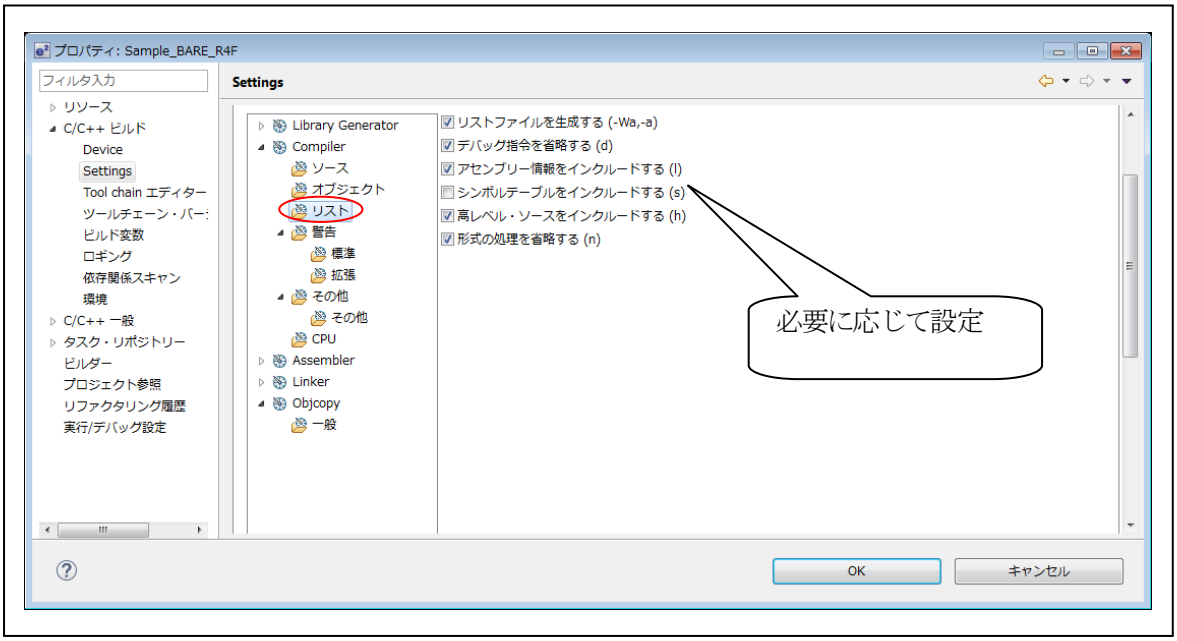
注＊1

「__USED_DEFnano__=0」と使用しない側に定義しても内蔵 RAM へのダウンロードとシリアルフラッシュ ROM への書き込み操作は可能です。ただし、再操作する場合はターゲット側のリセット操作が必要になります。

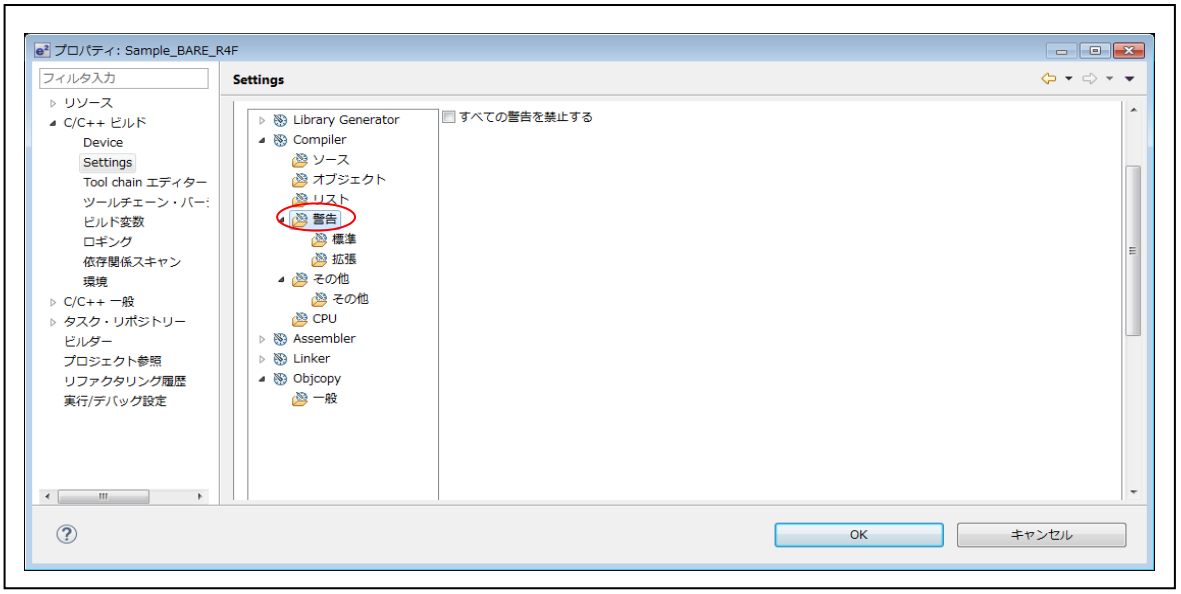
2）オブジェクト



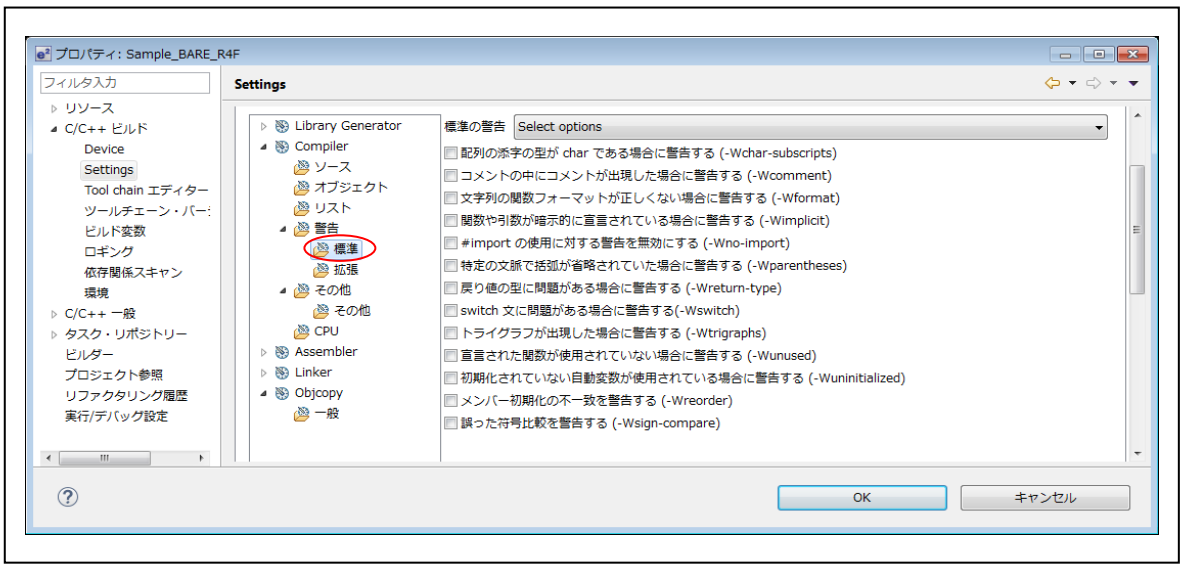
3）リスト（デフォルト）



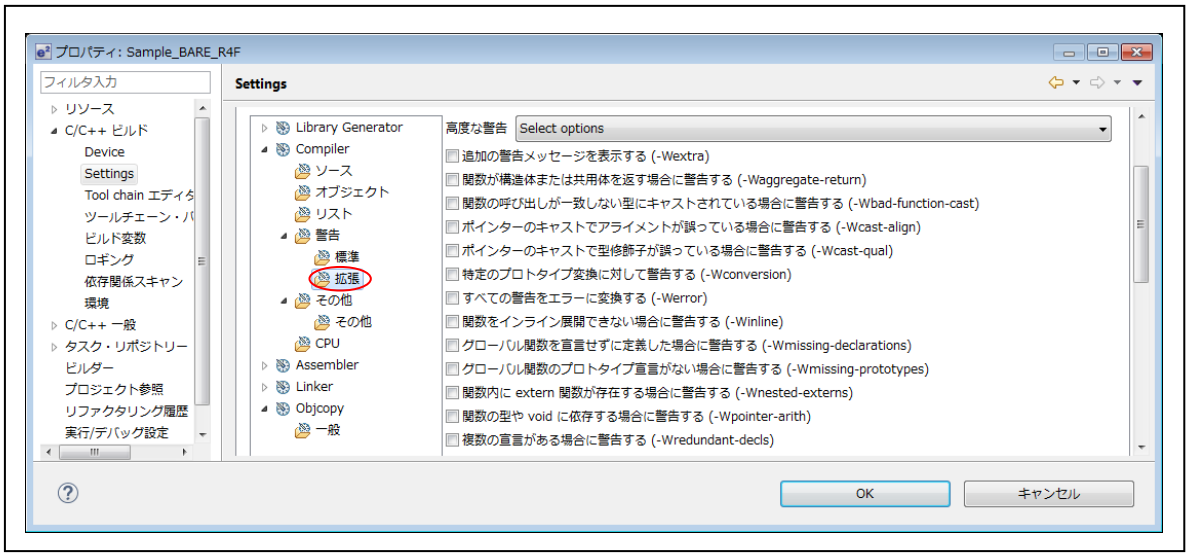
4）警告（デフォルト）



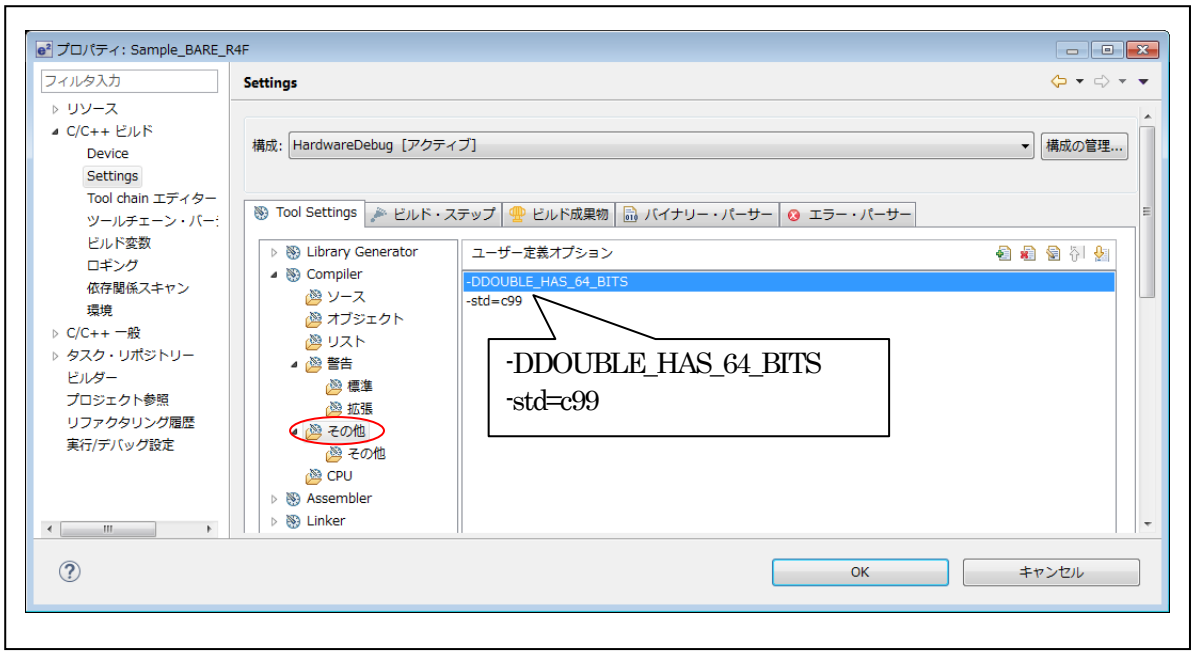
5）警告ー標準（デフォルト）



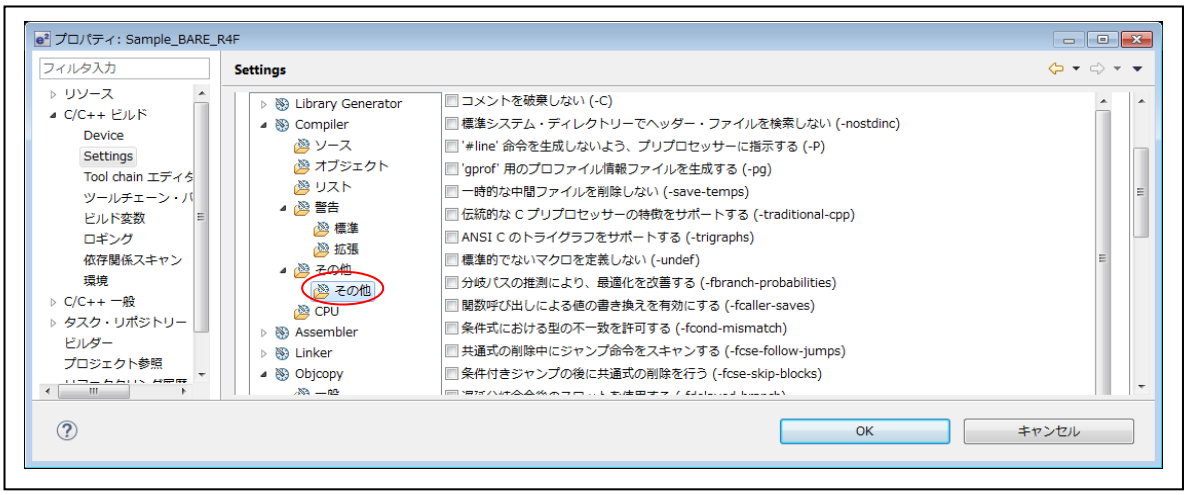
6）警告ー拡張（デフォルト）



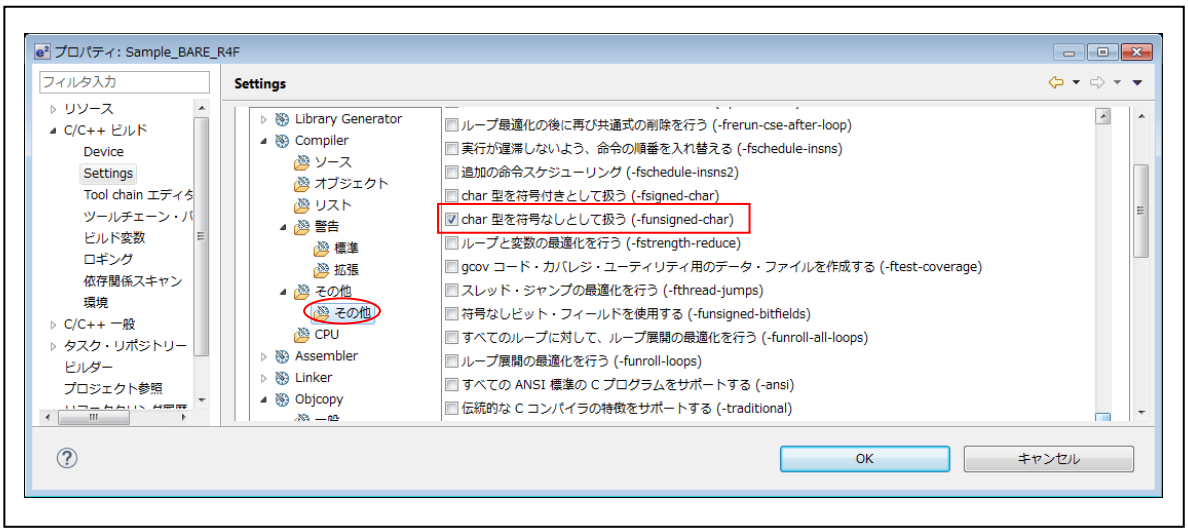
7）その他



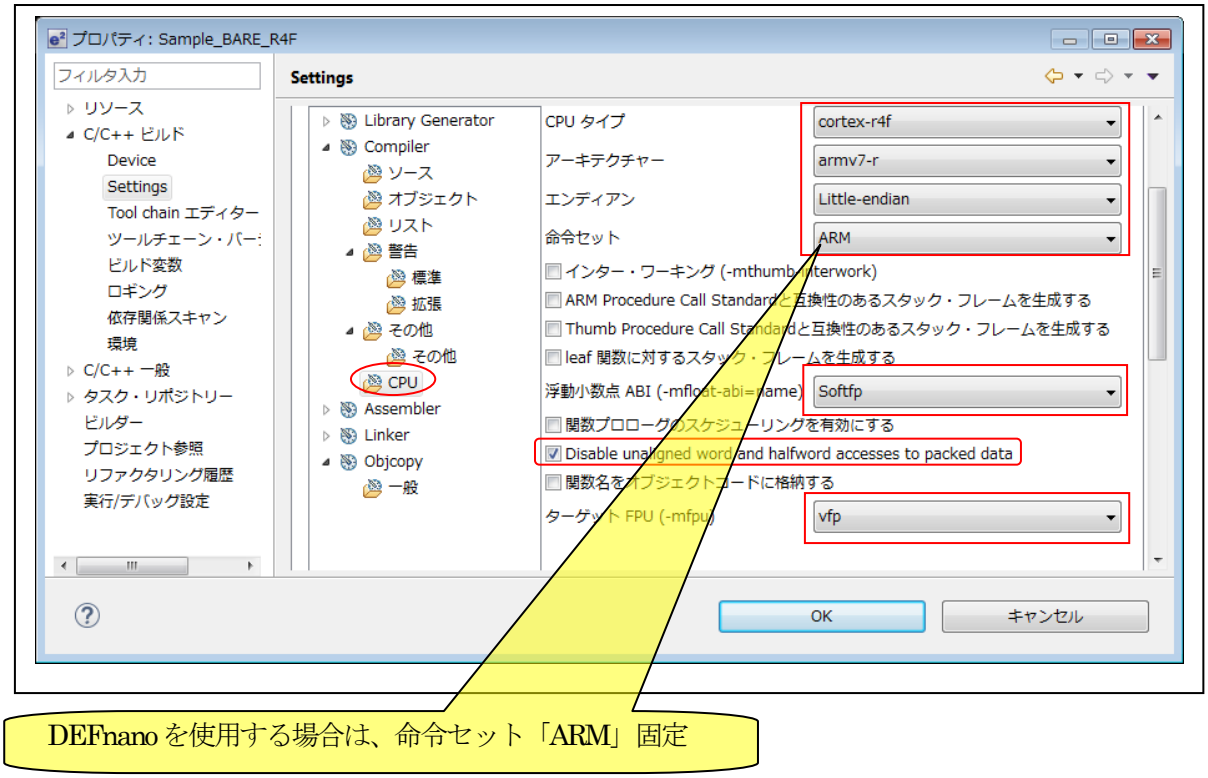
8）その他ーその他（1／2）



8）その他ーその他（2／2）

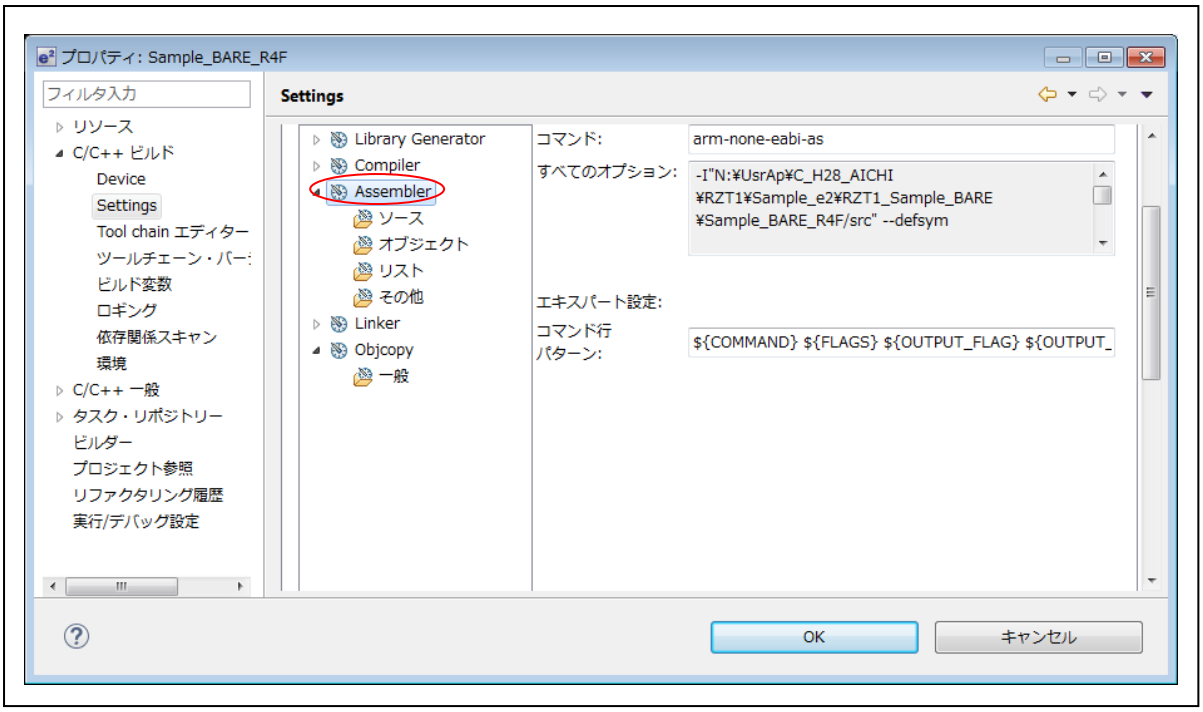


9）CPU

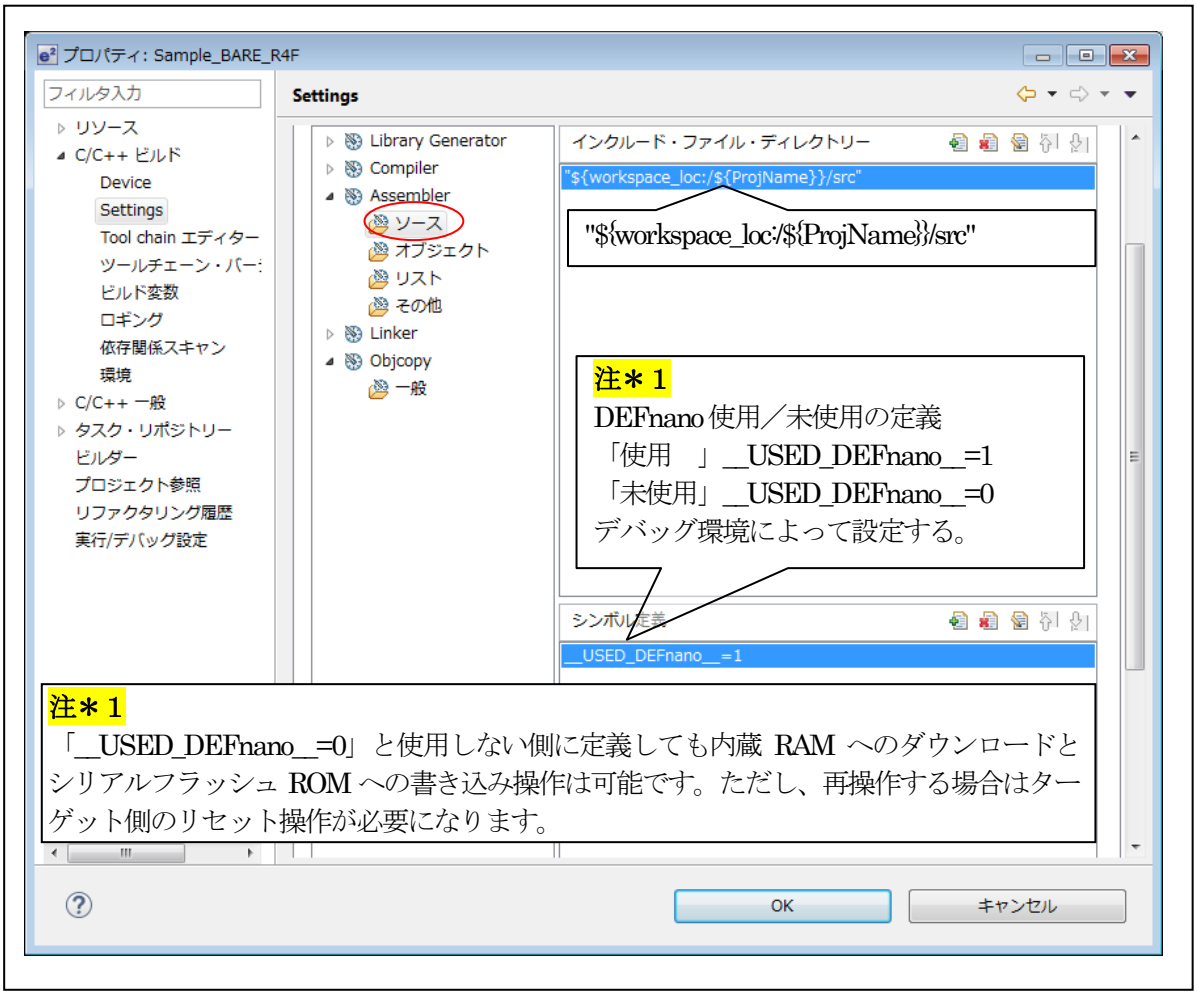


DEFnano を使用する場合は、命令セット「ARM」固定

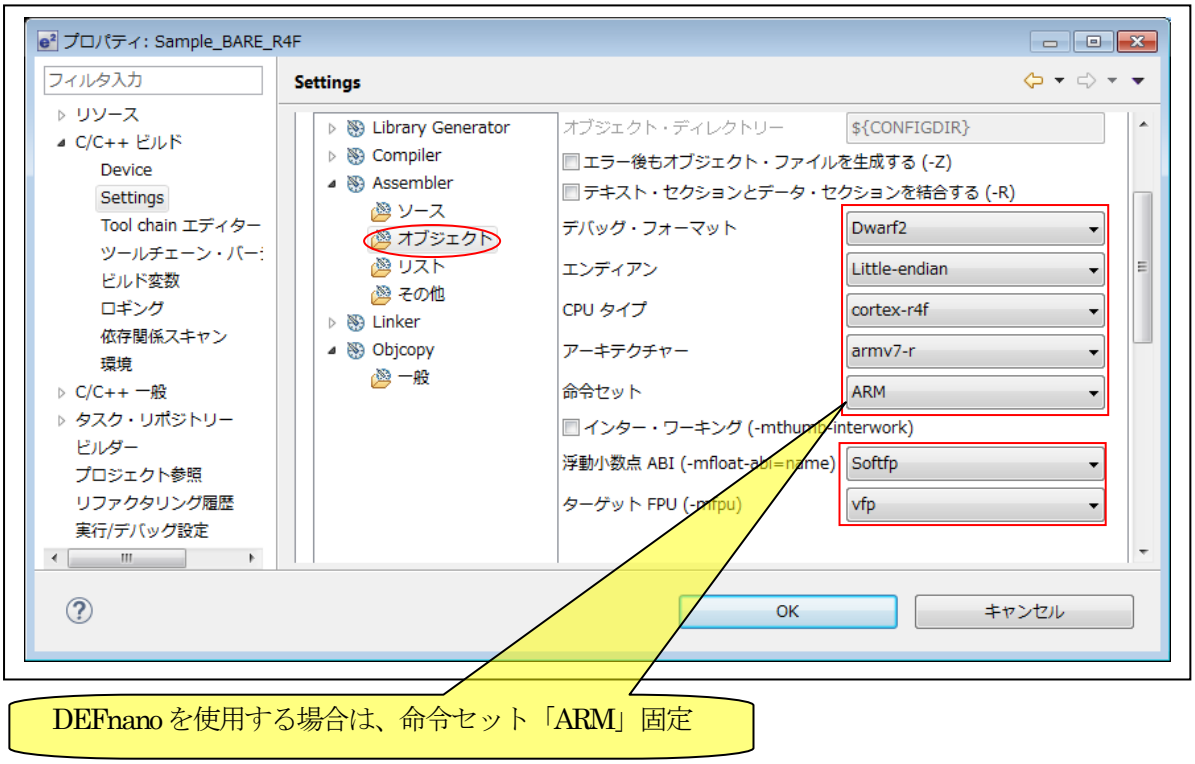
2－1－4－3．Tool　Settings(Assembler)



1）ソース

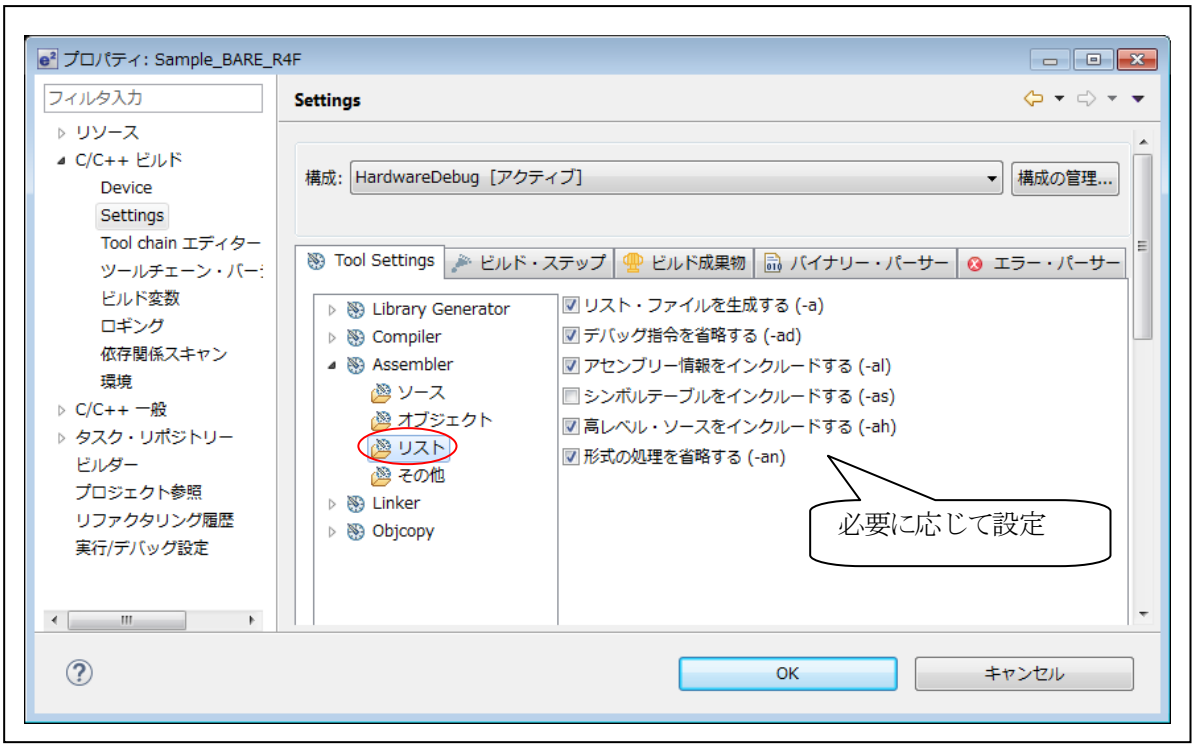


2）オブジェクト

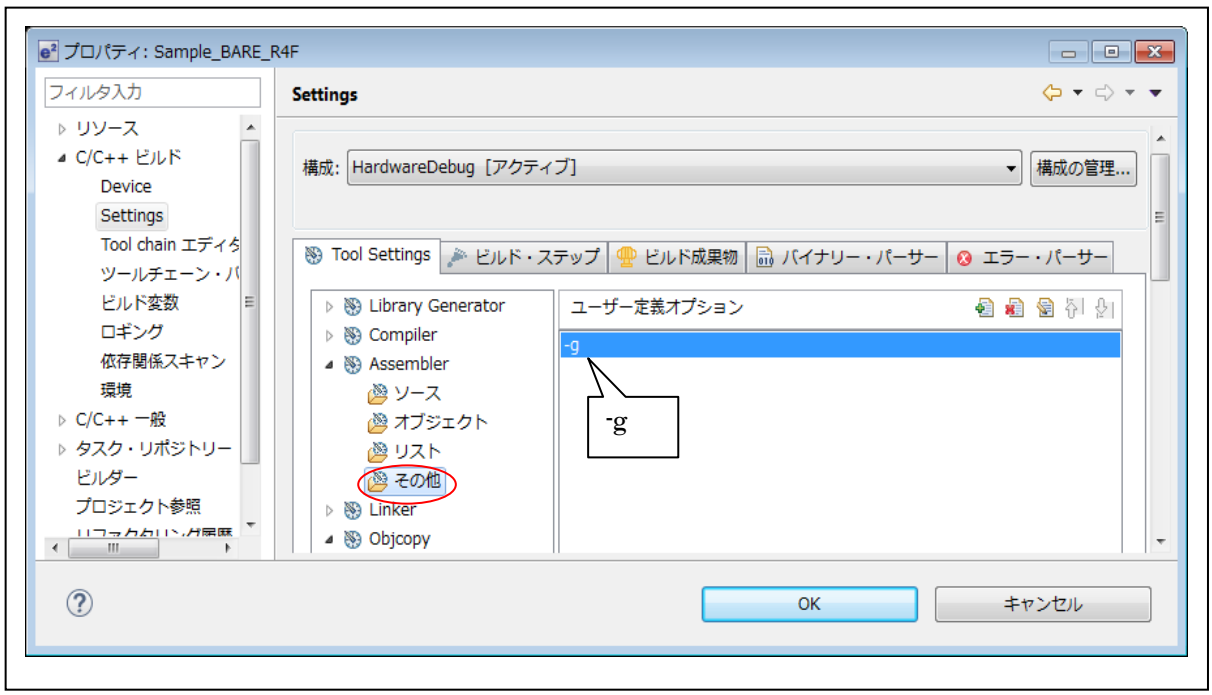


DEFnano を使用する場合は、命令セット「ARM」固定

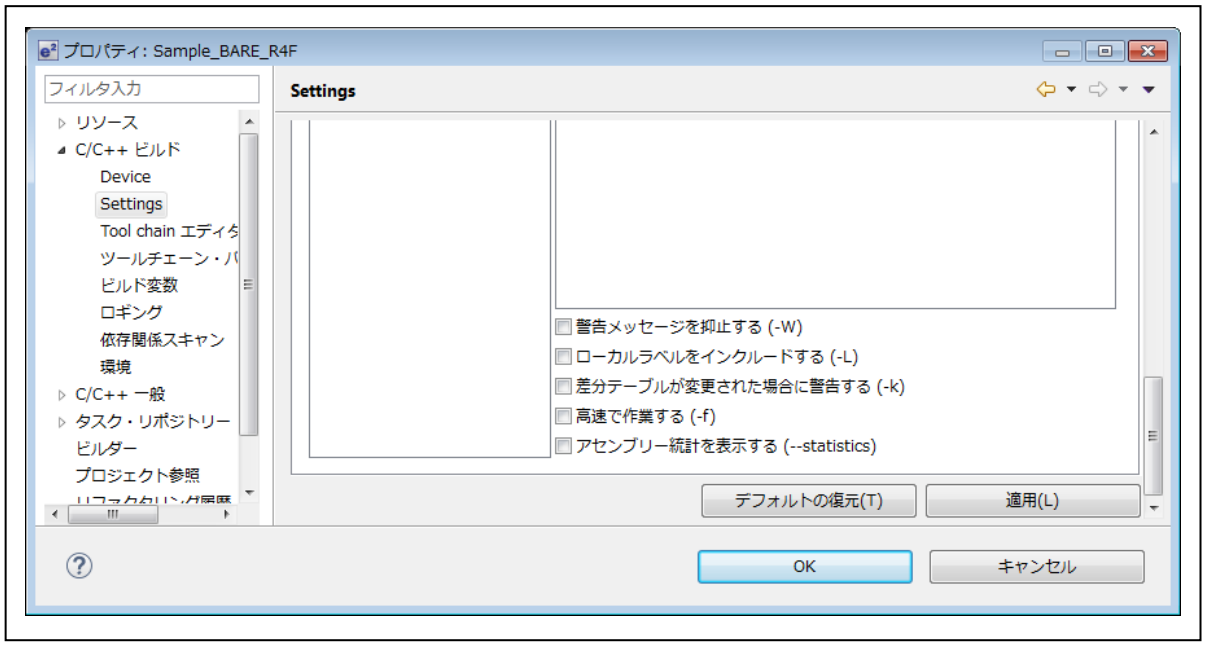
3）リスト（デフォルト）



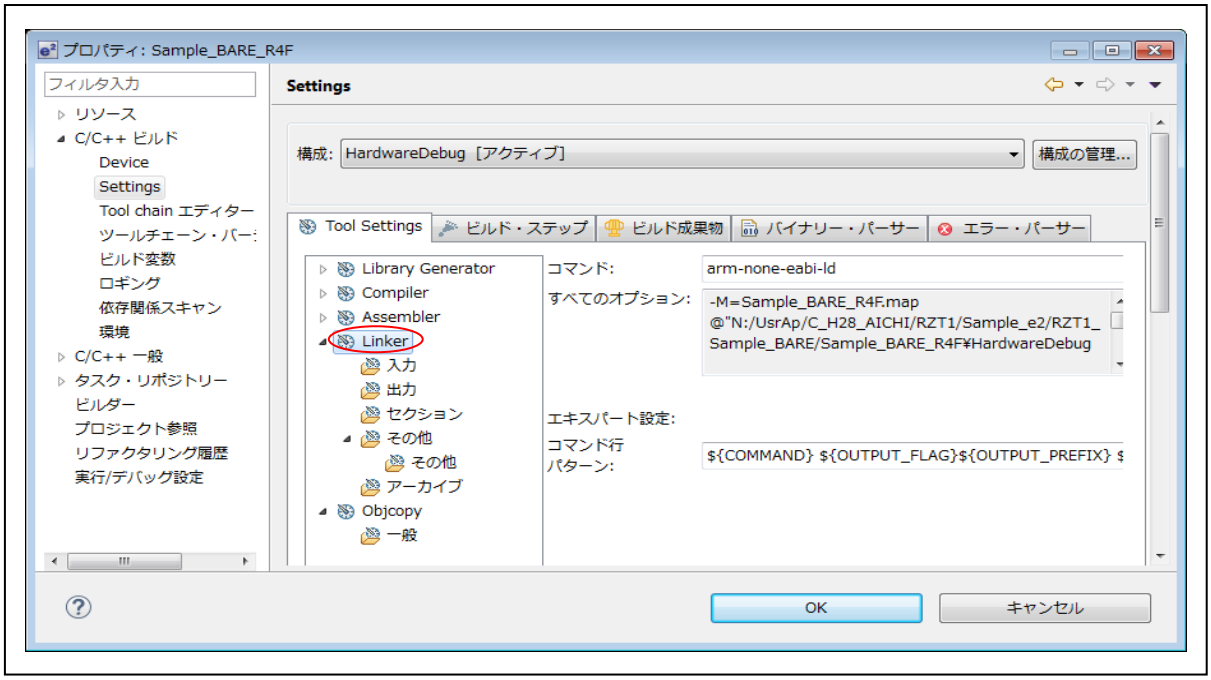
4）その他（1／2）



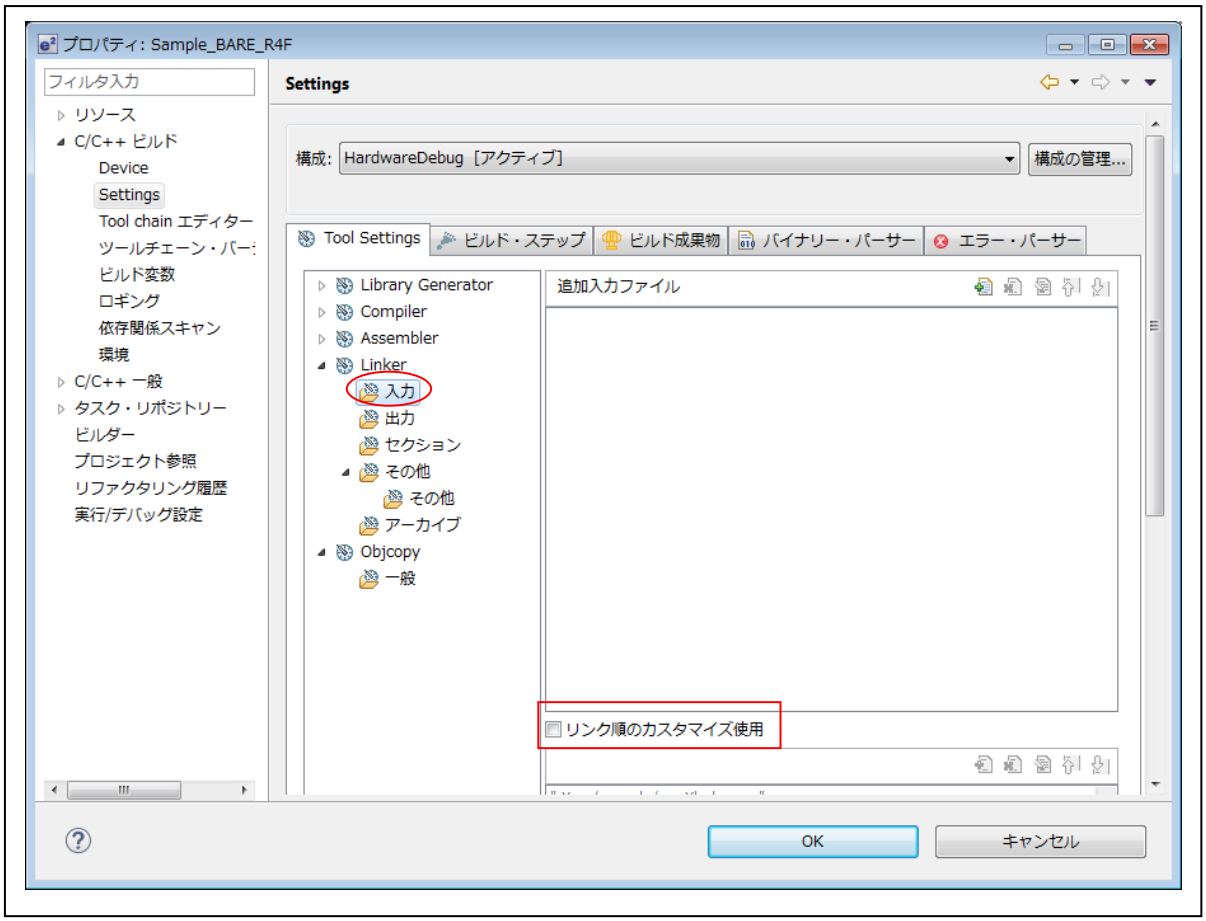
5）その他（2／2）（デフォルト）



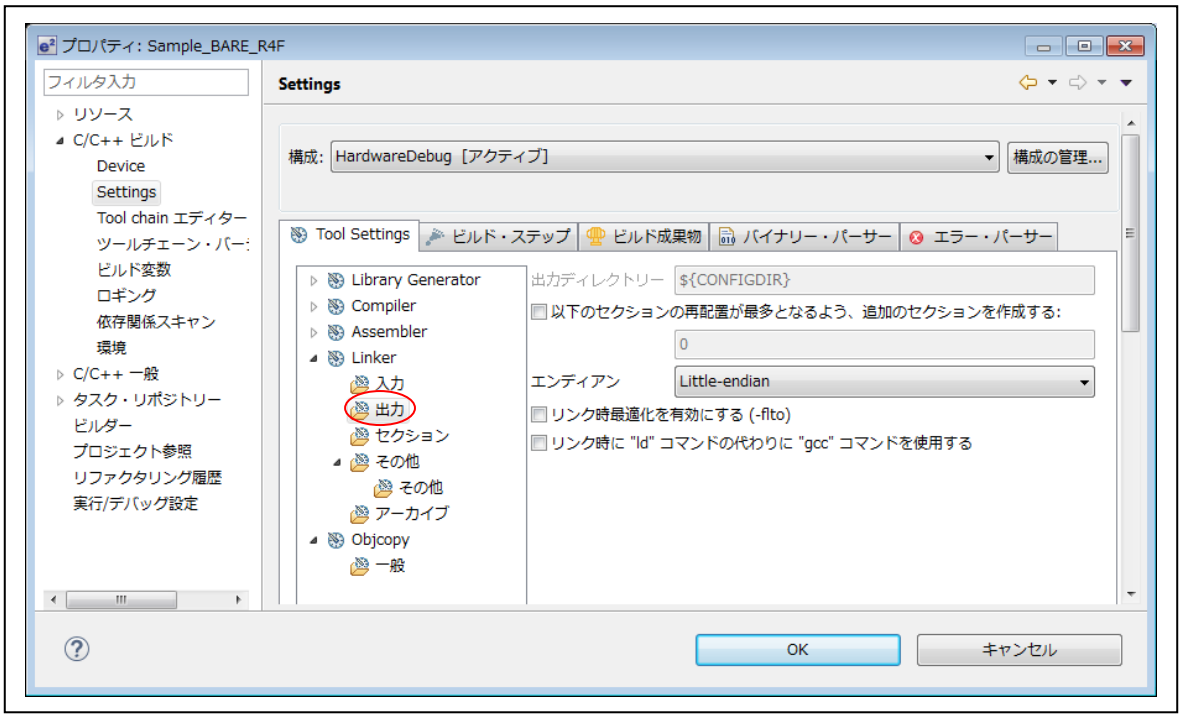
2－1－4－4．Tool　Settings(Linker)



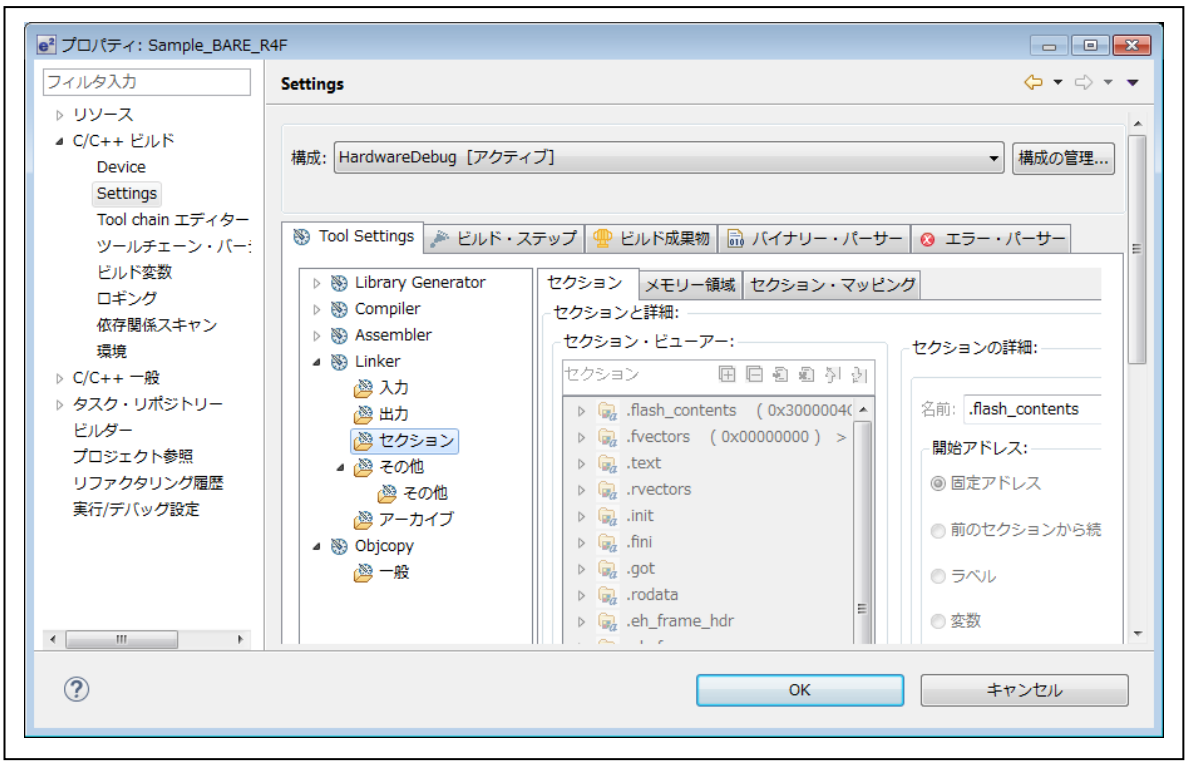
1）入力（デフォルト）



2）出力（デフォルト）



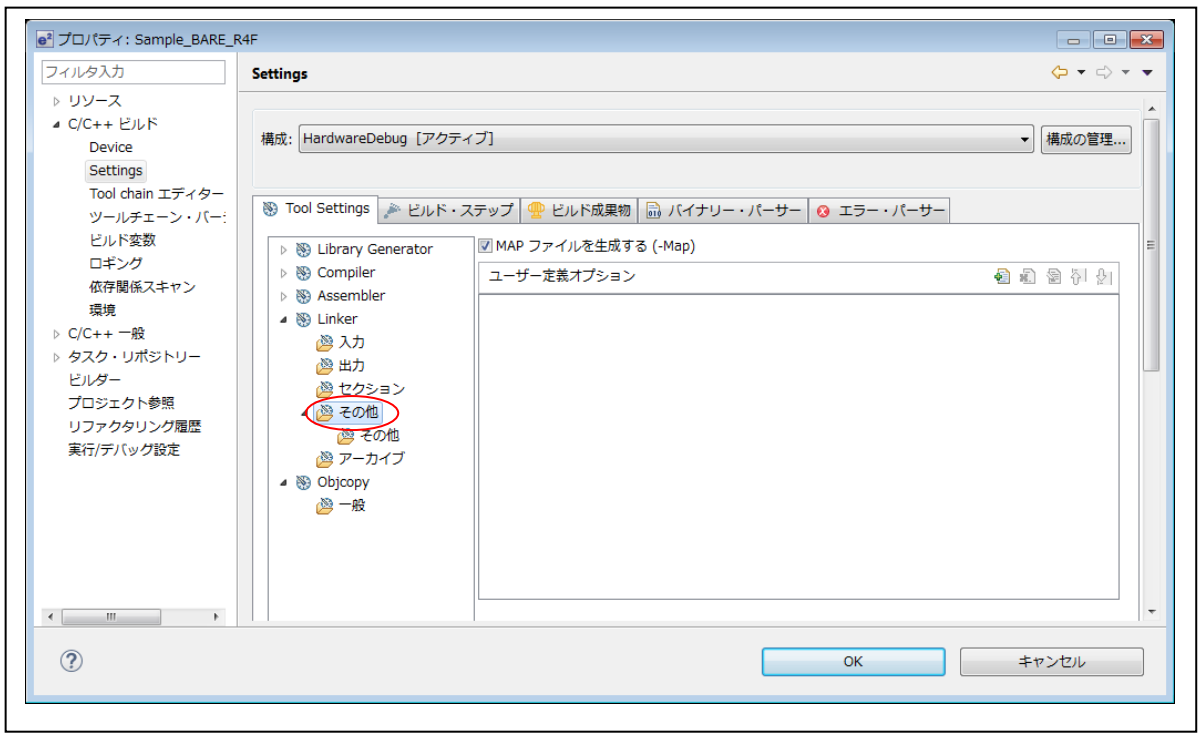
3）セクション（設定なし）



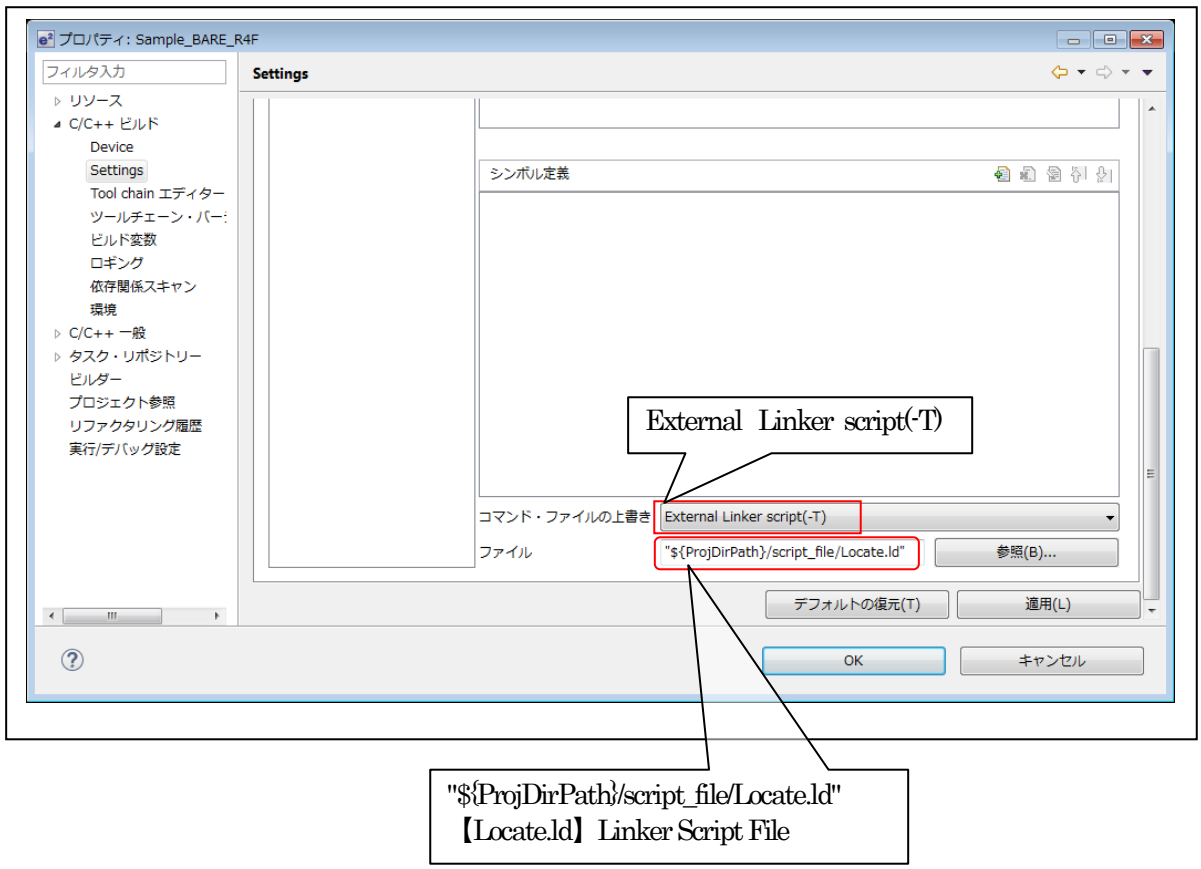
注＊1

サンプルのセクション設定は、後記「その他」の Linker Script ファイルにて設定

4）その他（1／2）（デフォルト）



5）その他（2／2）

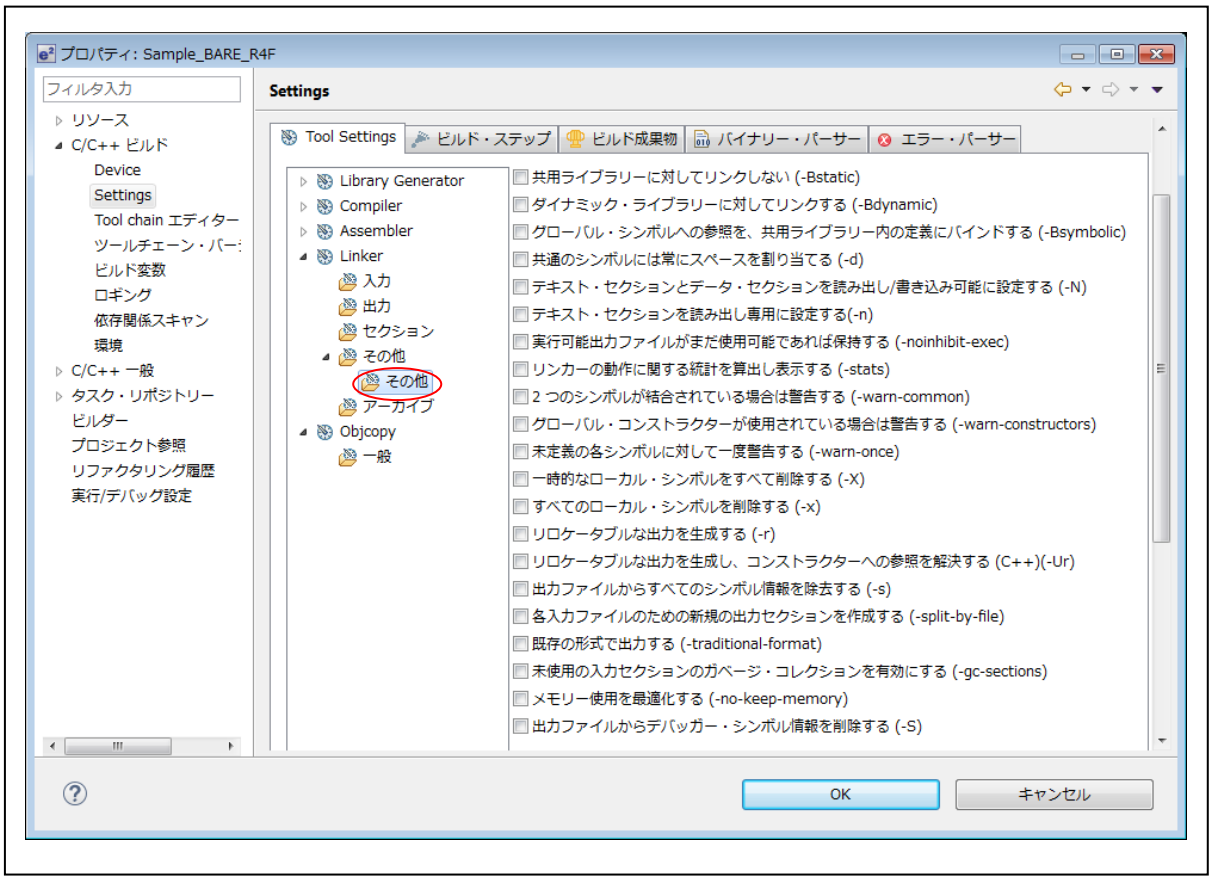


【Locate.ld】

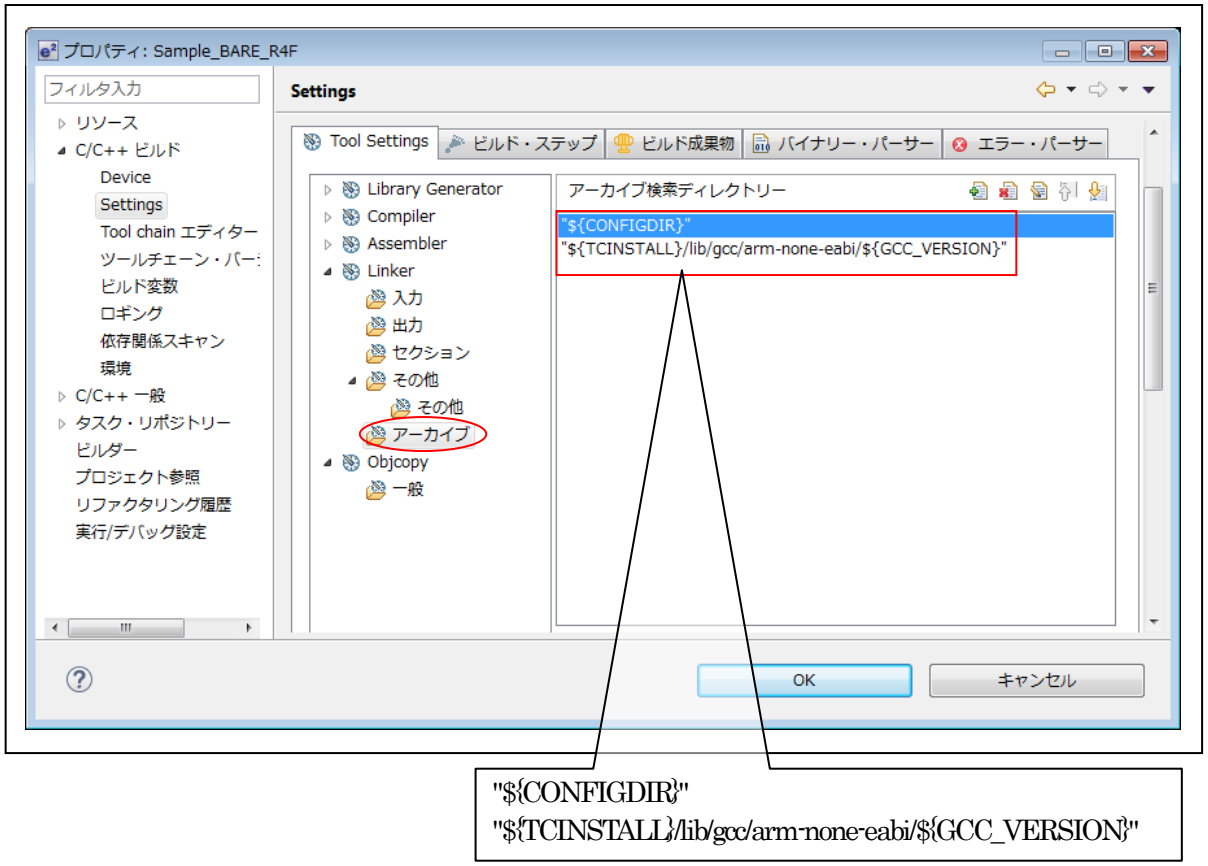
```
MEMORY{
  ROM      (rx) : ORIGIN = 0x00000000, LENGTH = 0x00060000
  RAM      (rw) : ORIGIN = 0x00060000, LENGTH = 0x00018000
  STACK    (rw) : ORIGIN = 0x00078000, LENGTH = 0x00008000
}
SECTIONS
{
  .text : {
    . = 0x00000000;
    * (.fvectors)              /* asm */
    . = 0x00000100;
    * (.loader_text)
    * (.text .text.*)
    * (.rodata .rodata.*)
  } > ROM
  .data : {
    __data_load = .;
    __data_start = LOADADDR(.data) + ( __data_load - ADDR(.data) );
    * (.data .data.*)
    __data_end = LOADADDR(.data) + ( . - ADDR(.data) );
  } > ROM
  .bss : {
    __bss_start = .;
    * (.bss .bss.*)
    * (COMMON)
    __bss_end = .;
    * (_ebss _end)
    end = .;
  } > RAM
  .stack : {
    . = ALIGN( 0x10 );
    _sys_stack_top = .;
    . += 0x0002000;
    _sys_stack = .;

    . = ALIGN( 0x10 );
    _svc_stack_top = .;
    . += 0x00000100;
    _svc_stack = .;

    . = ALIGN( 0x10 );
    _irq_stack_top = .;
    . += 0x00002000;
    _irq_stack = .;

// Next page
```

```
    .=ALIGN(0x10);
    _und_stack_top=.;
    .+=0x00000100;
    _und_stack=.;

    .=ALIGN(0x10);
    _abt_stack_top=.;
    .+=0x00000100;
    _abt_stack=.;
  }>STACK

}
// EOF
```

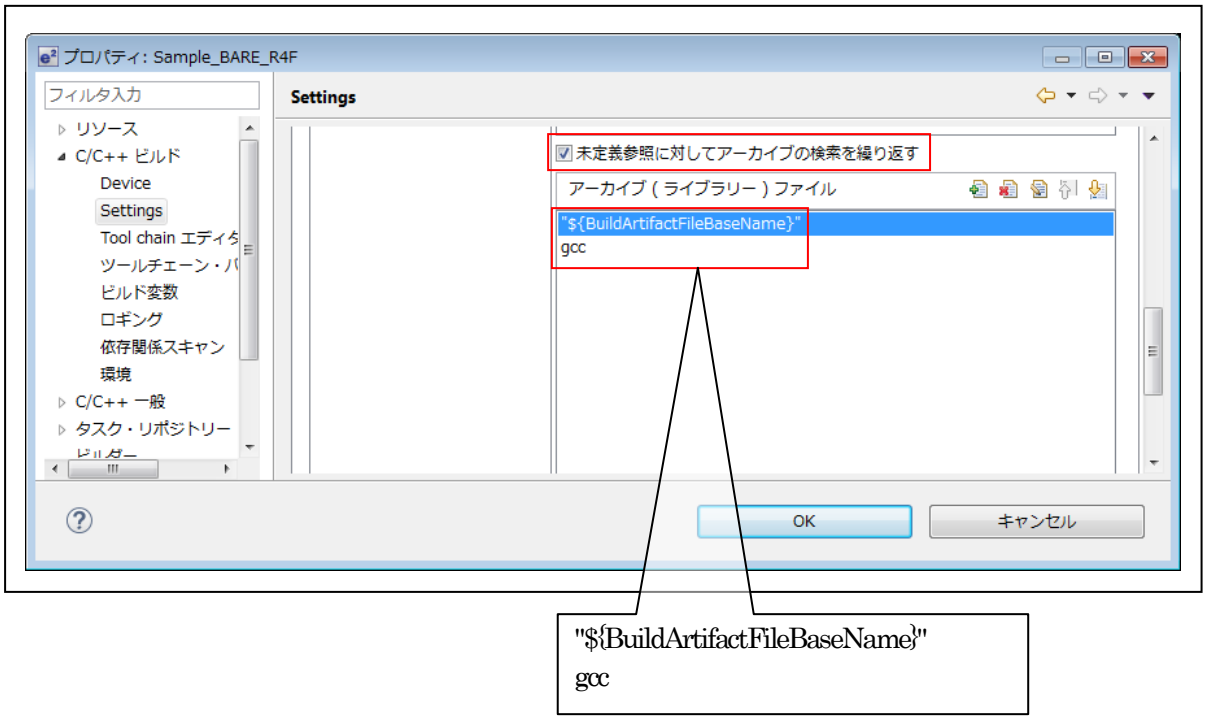
6）その他ーその他（デフォルト）



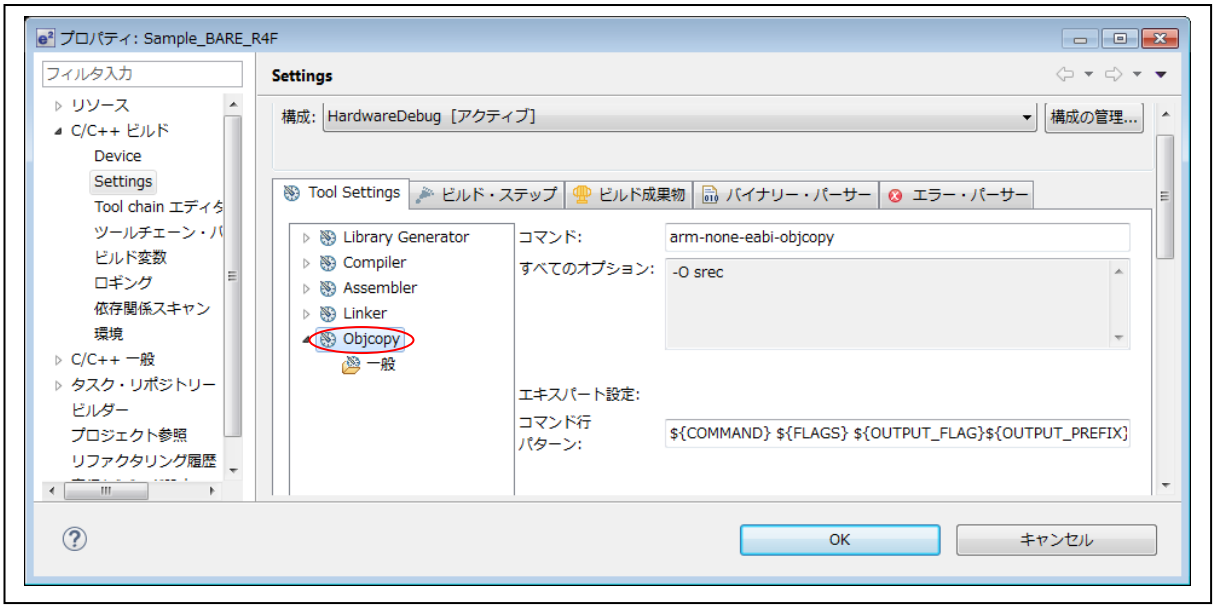
7）アーカイブ（1／2）



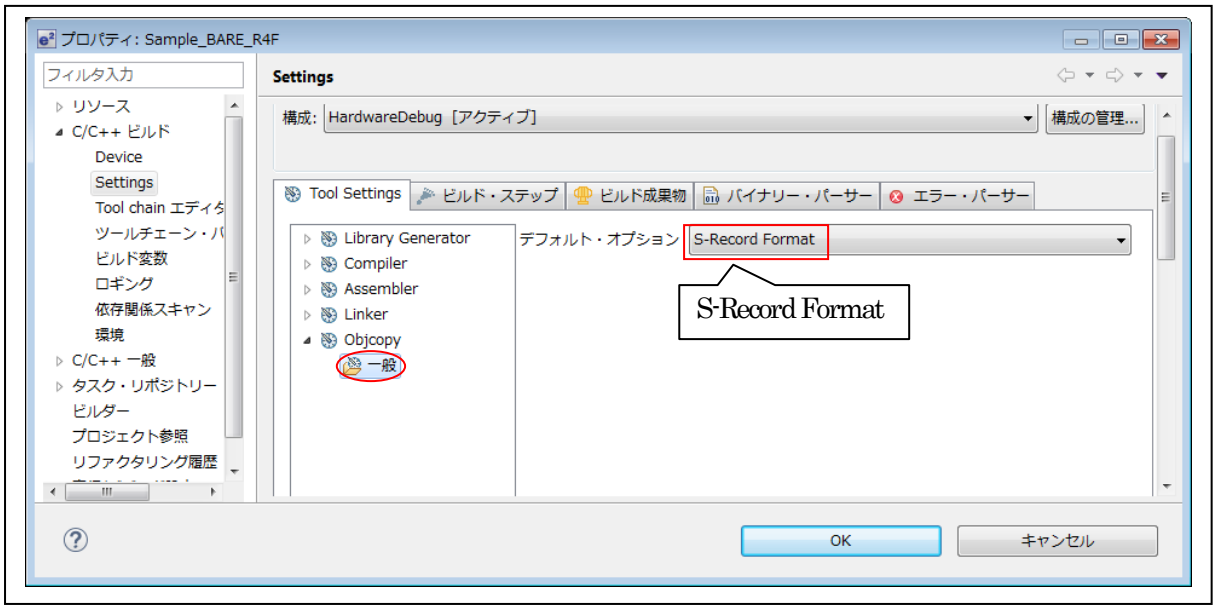
8）アーカイブ（2／2）



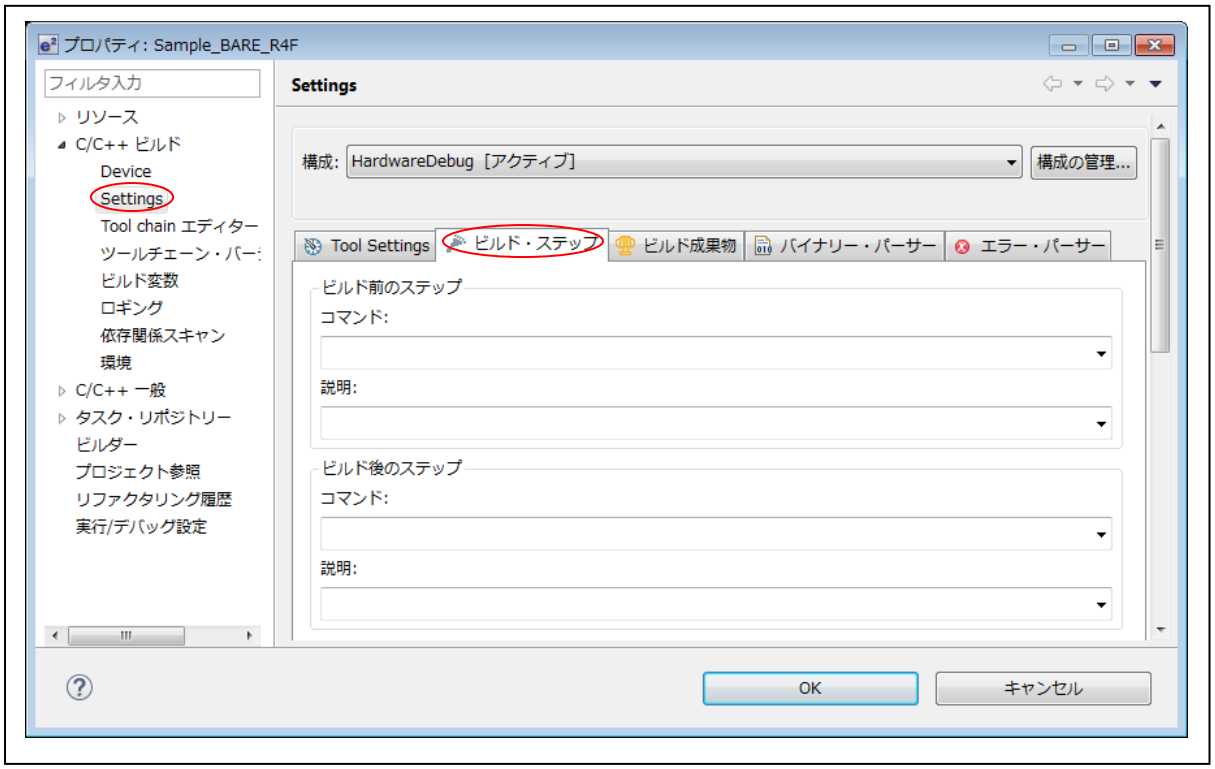
2－1－4－5．Tool　Settings(Objcopy)



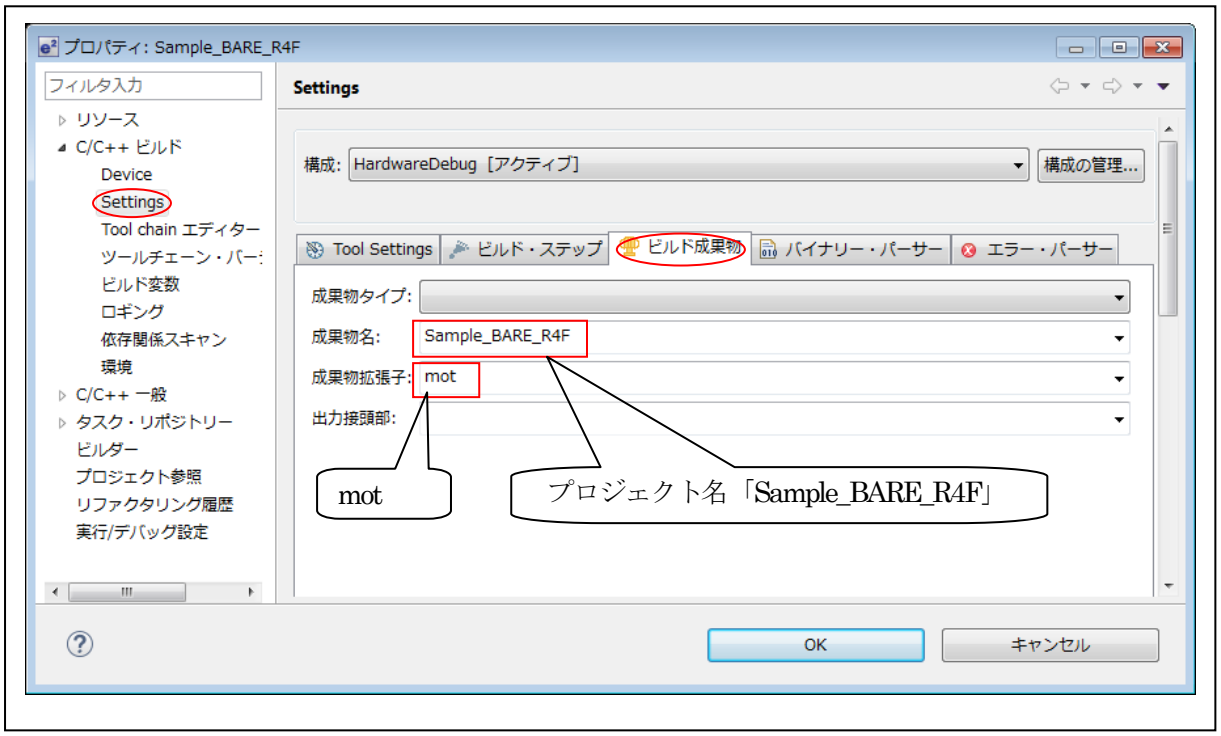
1）一般



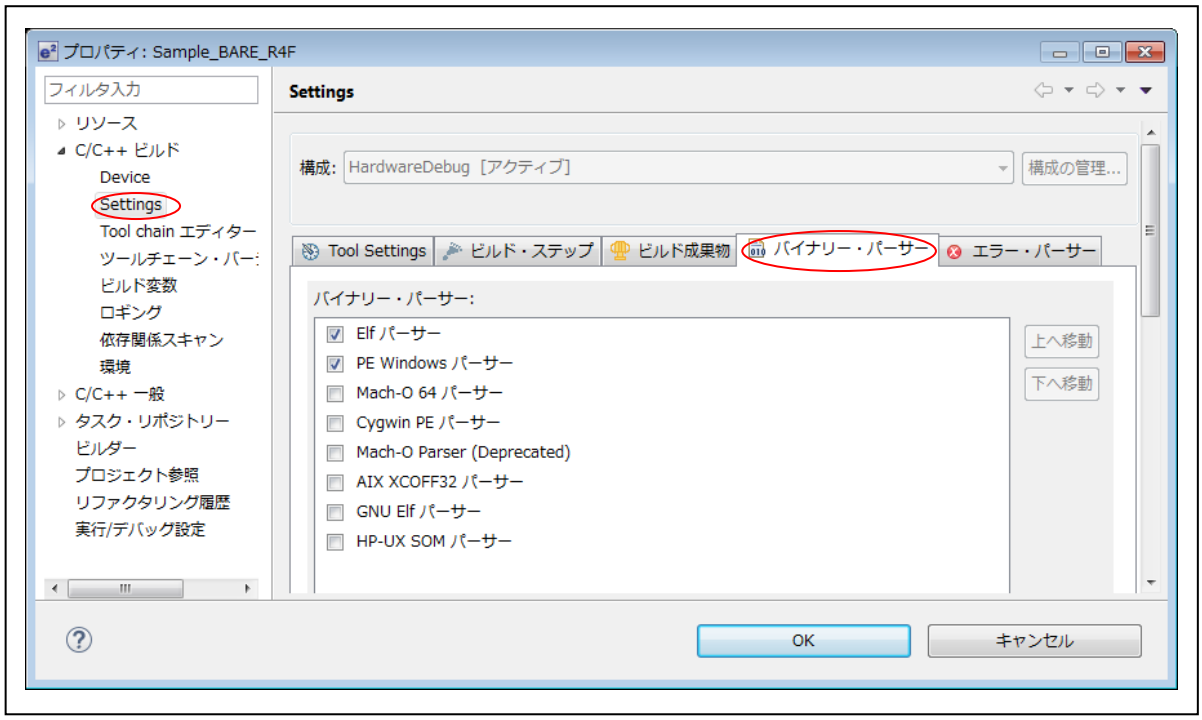
2－1－4－6．ビルド・ステップ（デフォルト）



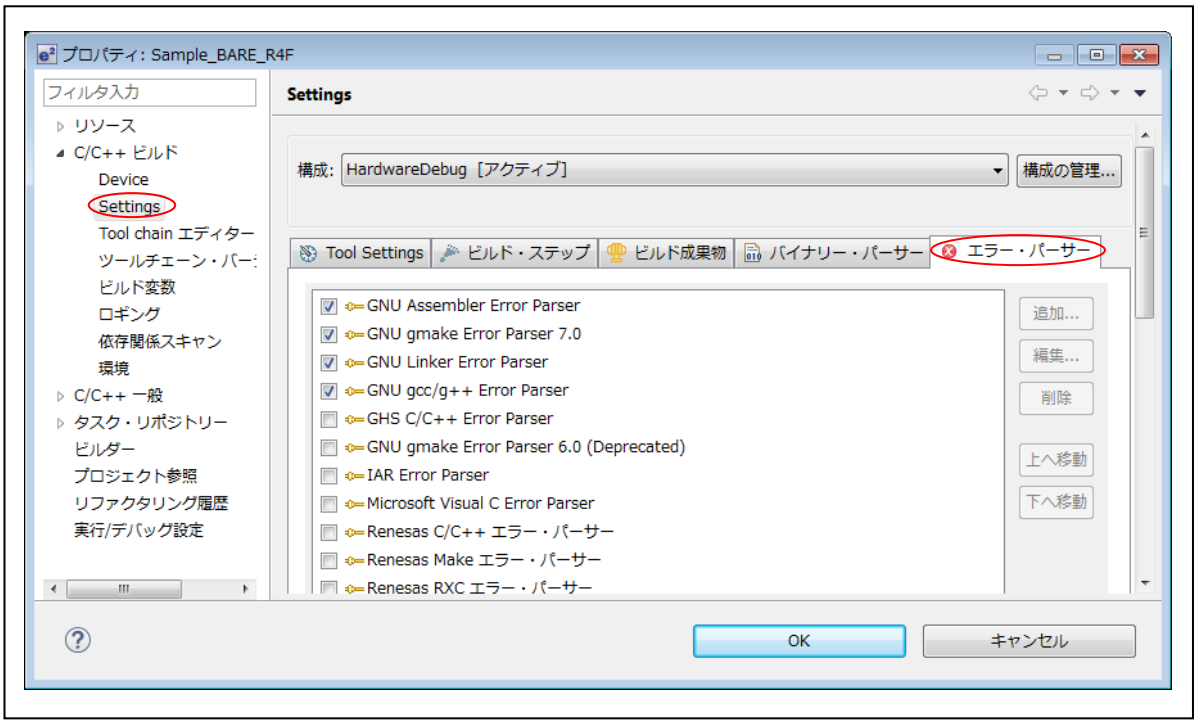
2－1－4－7．ビルド成果物



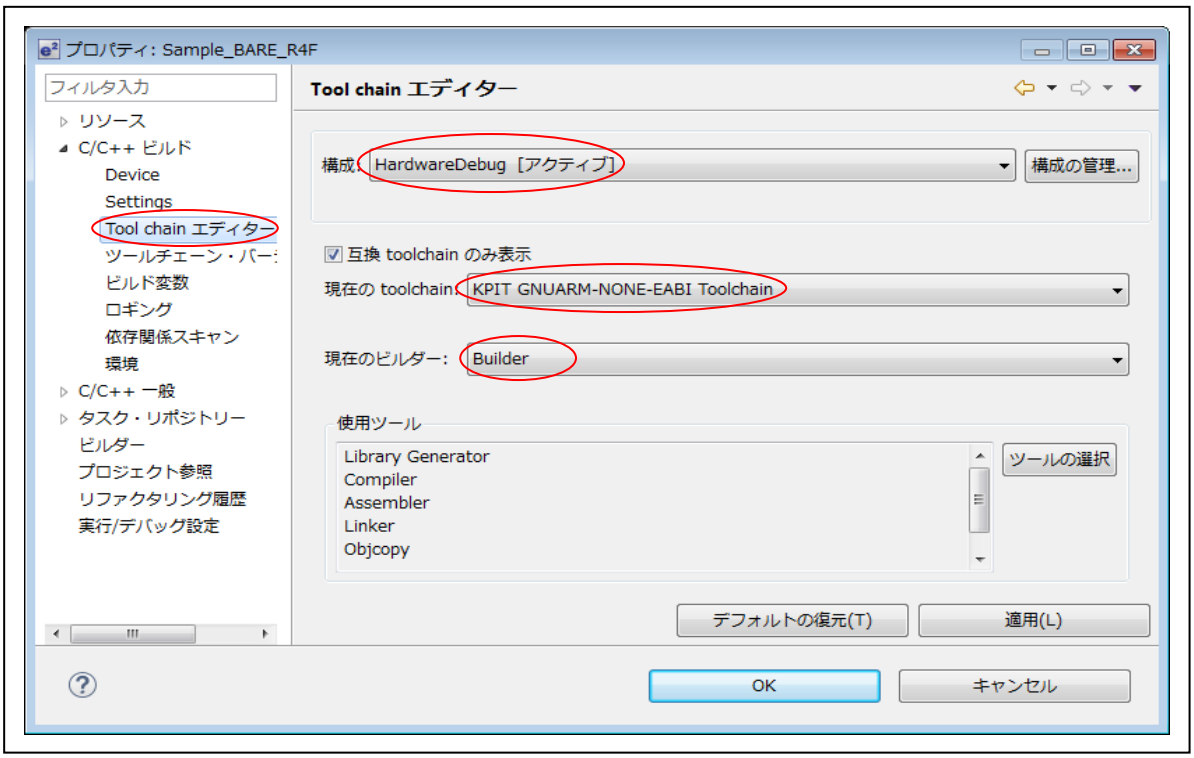
2－1－4－8．バイナリー・パーサー（デフォルト）



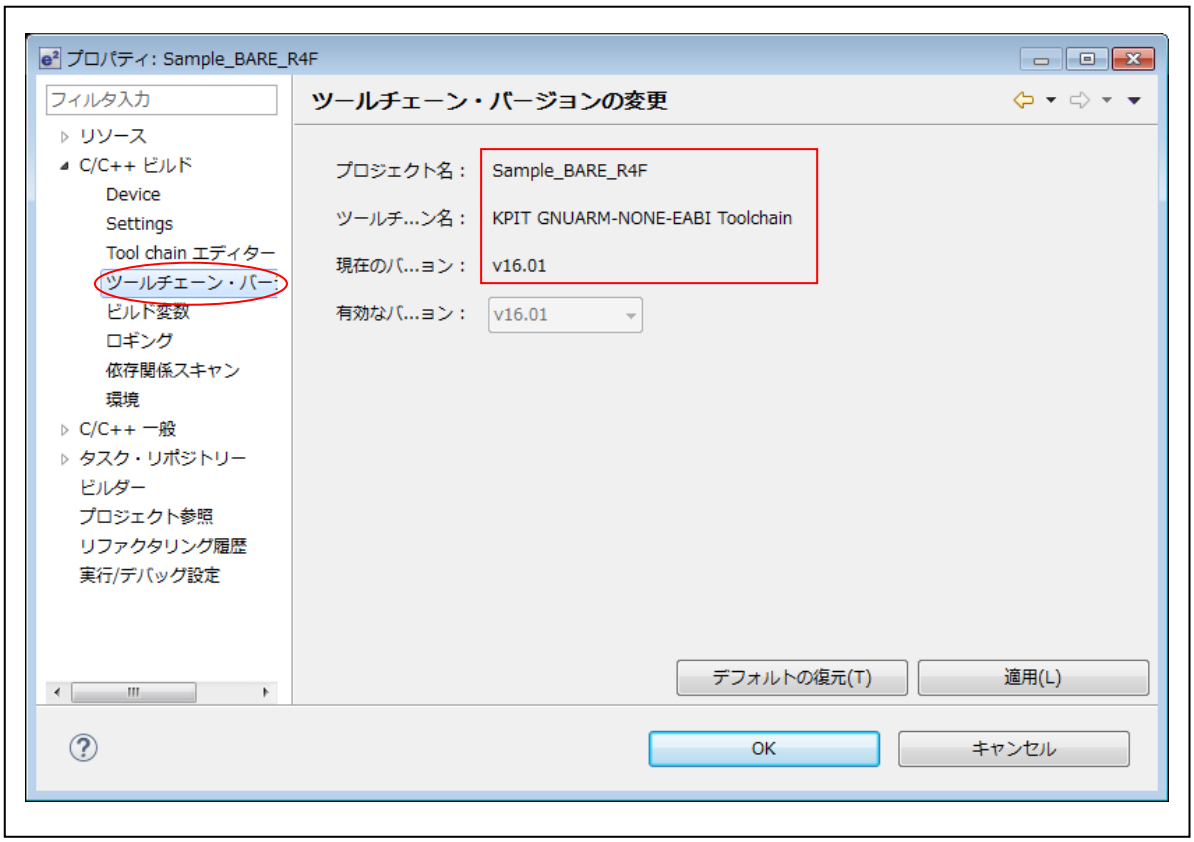
2－1－4－8．エラー・パーサー（デフォルト）



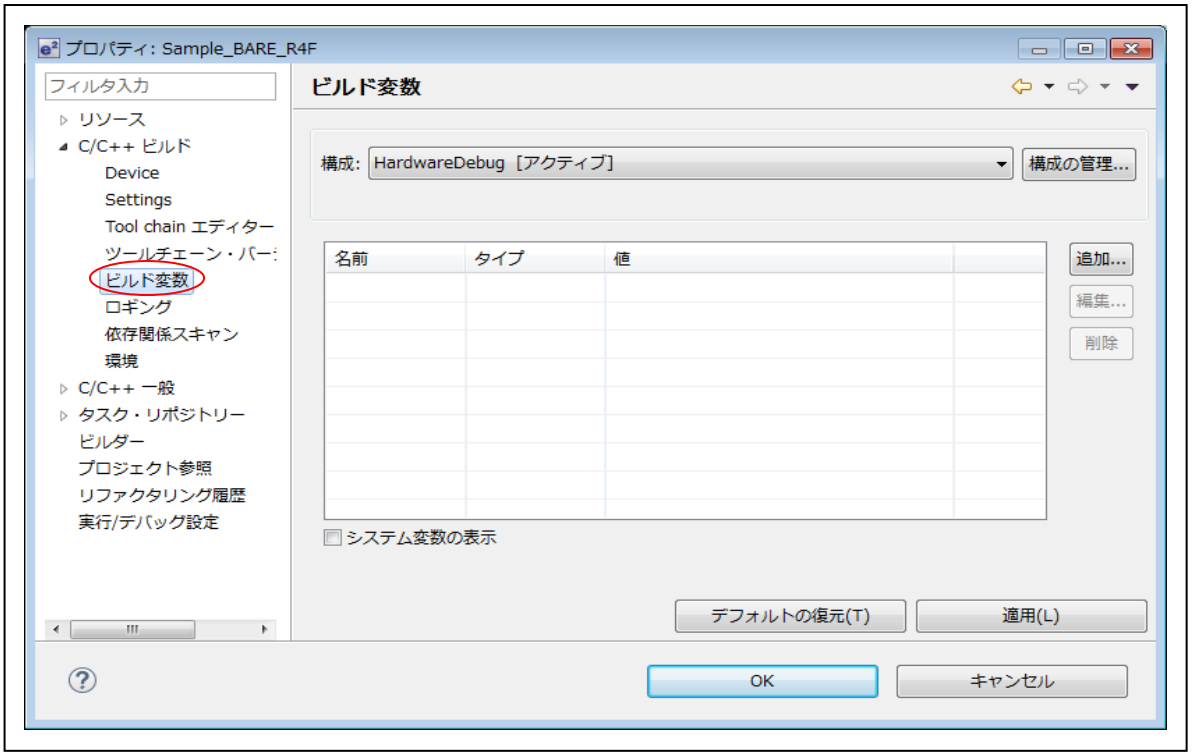
2－1－5．C/C++ビルド（Tool Chain エディター）（デフォルト）



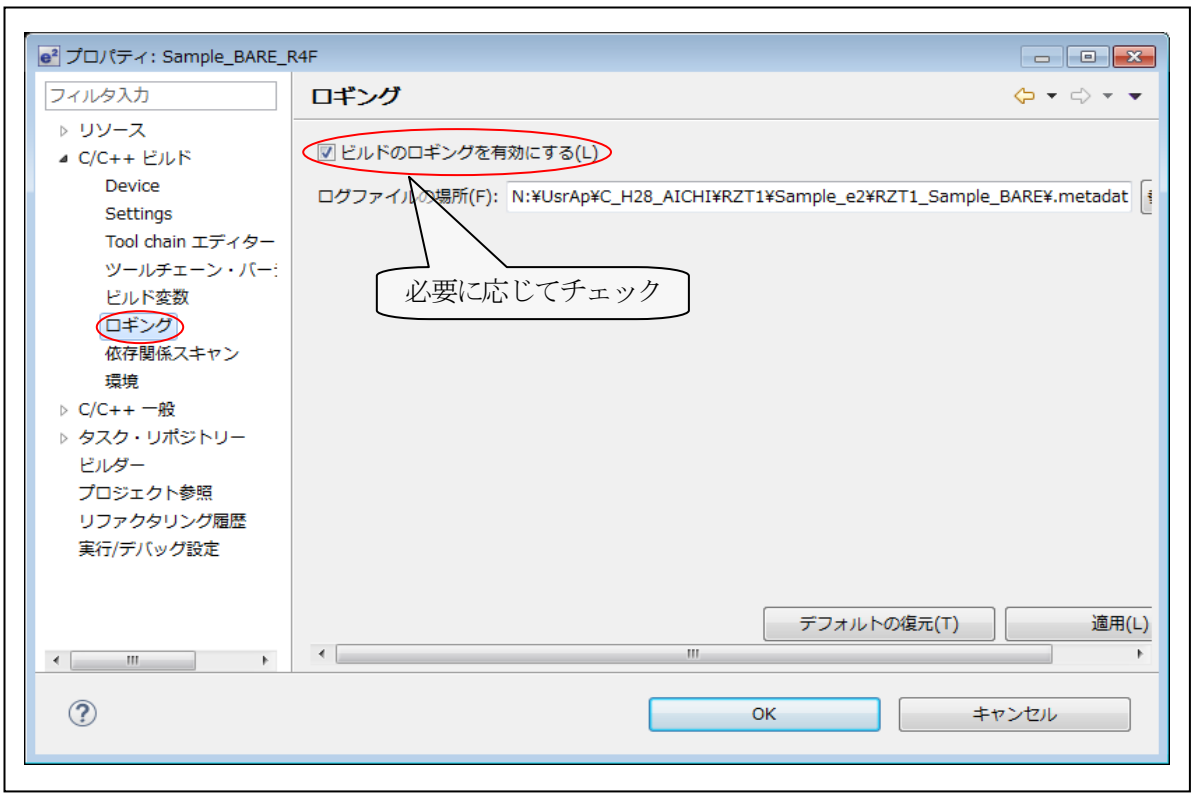
2－1－6．C/C++ビルド（ツールチェーン・バージョン）（デフォルト）



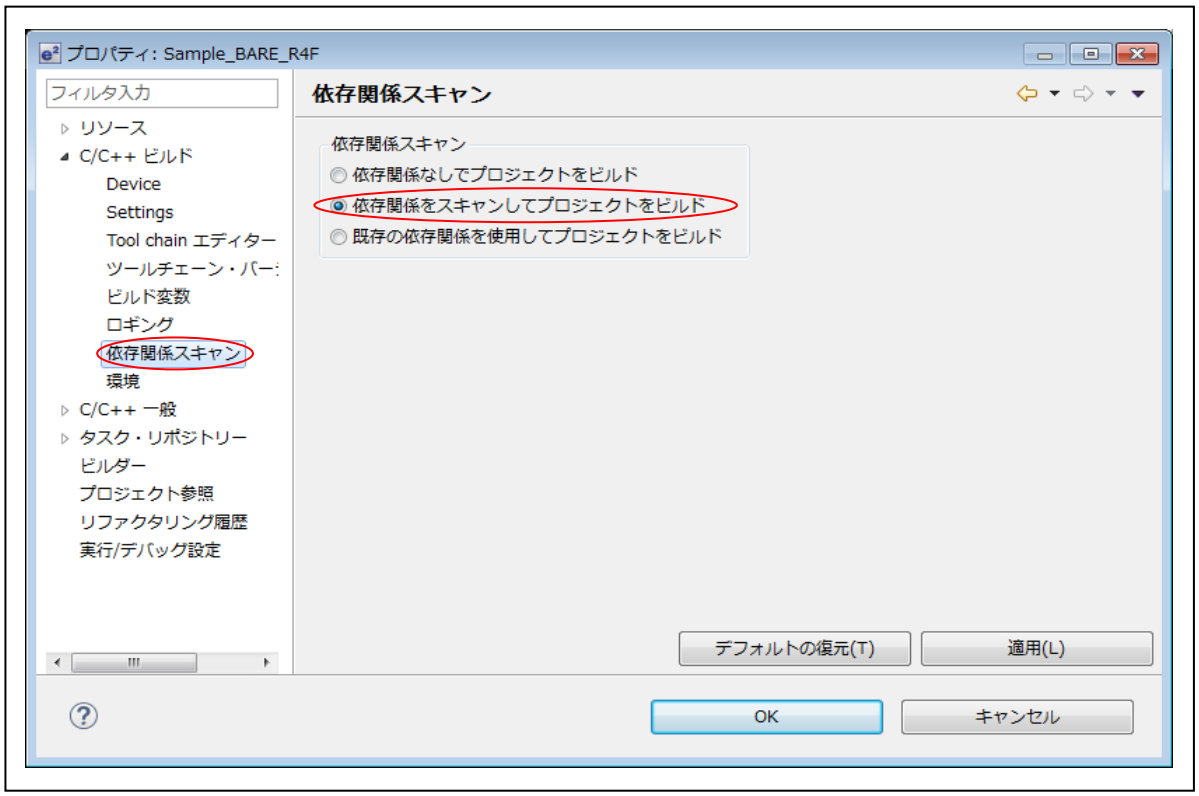
2－1－7．C/C++ビルド（ビルド変数）（デフォルト）



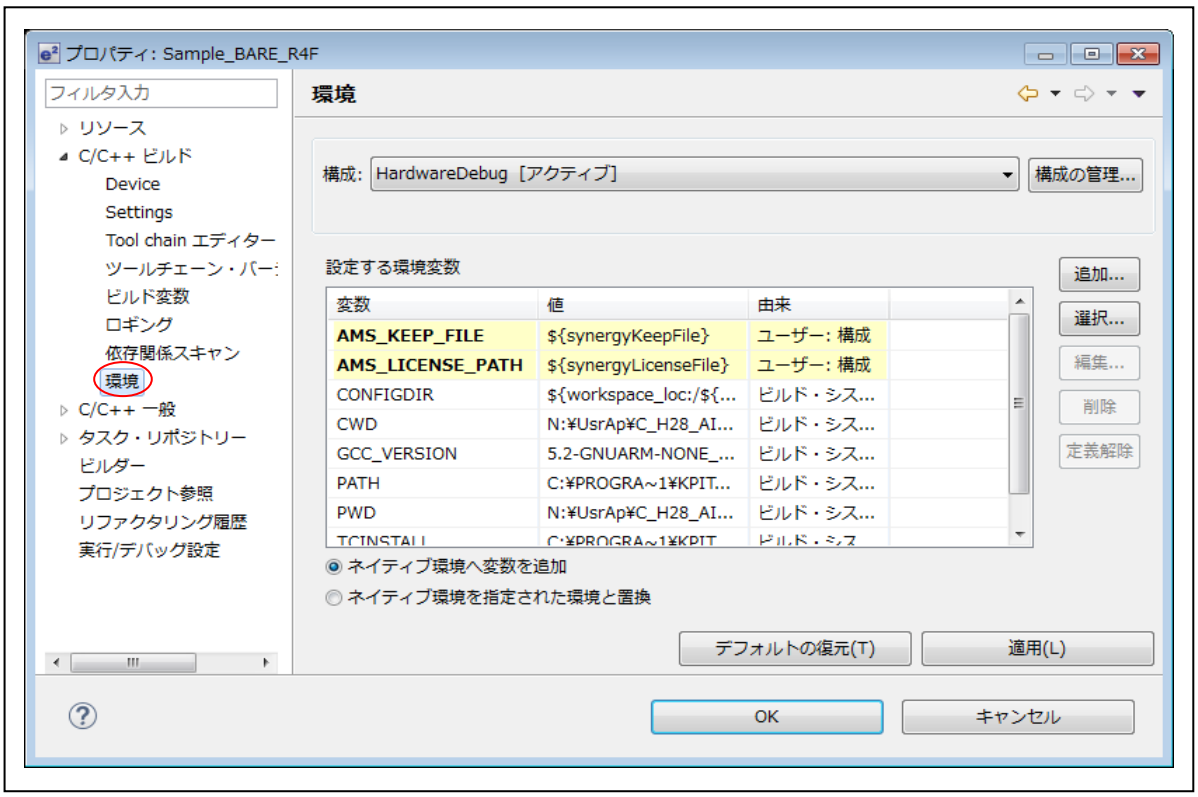
2－1－8．C/C++ビルド（ロギング）（デフォルト）



２－１－９．C/C++ビルド（依存関係スキャン）（デフォルト）



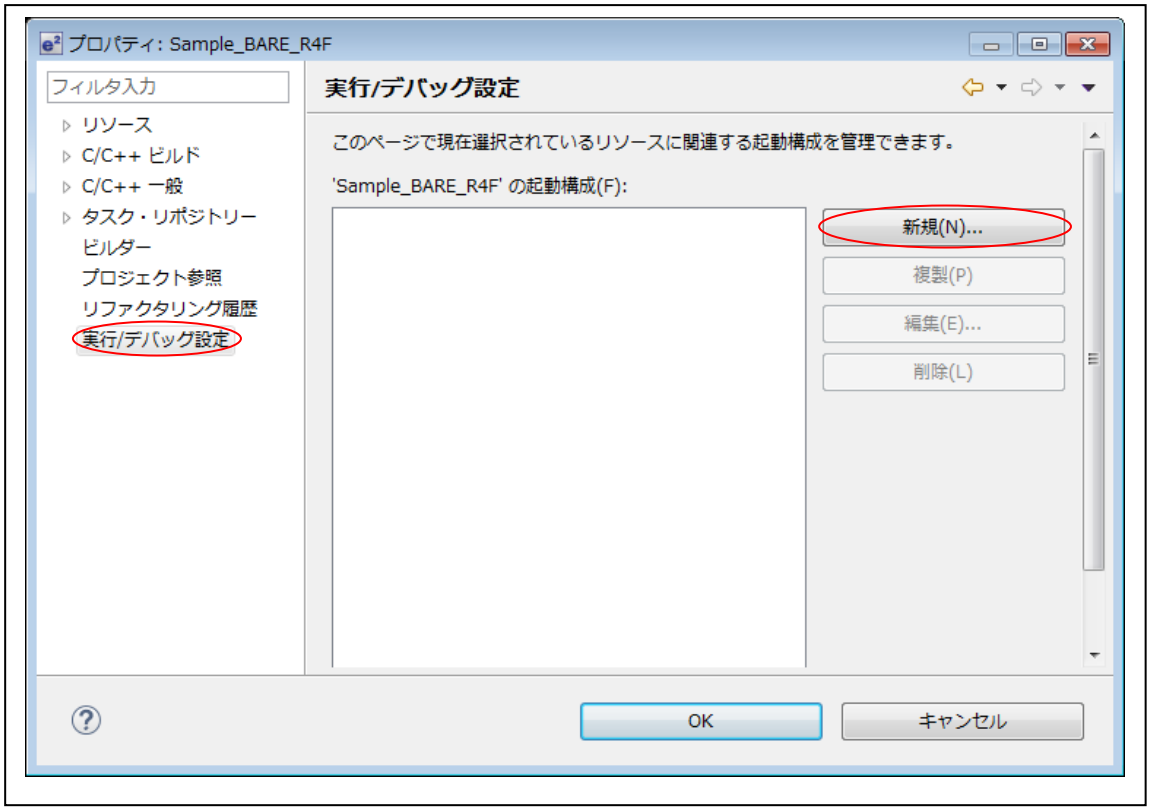
２－１－１０．C/C++ビルド（環境）（デフォルト）



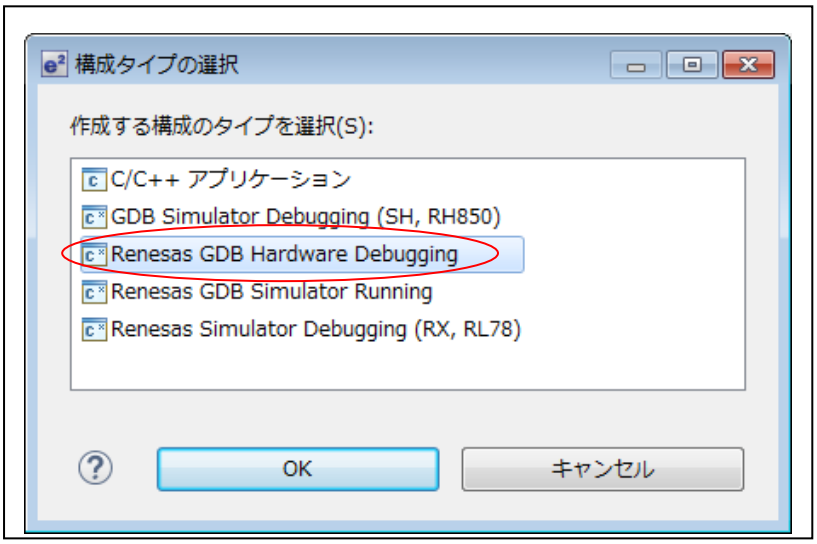
２－１－１１．【説明省略】C/C++　一般（デフォルト）
２－１－１２．【説明省略】タスク・リポジトリー（デフォルト）
２－１－１３．【説明省略】ビルダー（デフォルト）
２－１－１４．【説明省略】プロジェクト参照（デフォルト）
２－１－１５．【説明省略】リファクタリング履歴（デフォルト）

２－１－１６．実行／デバッグ設定

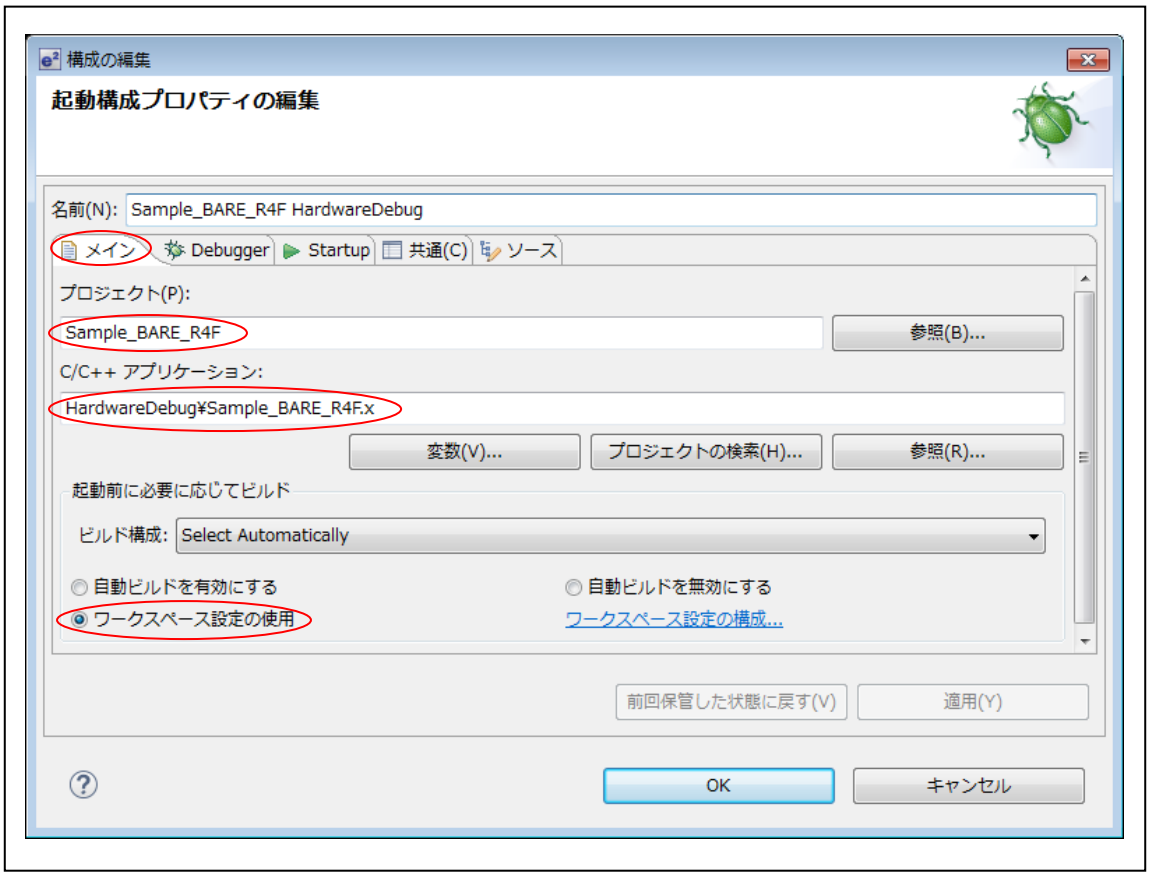
１）新規登録



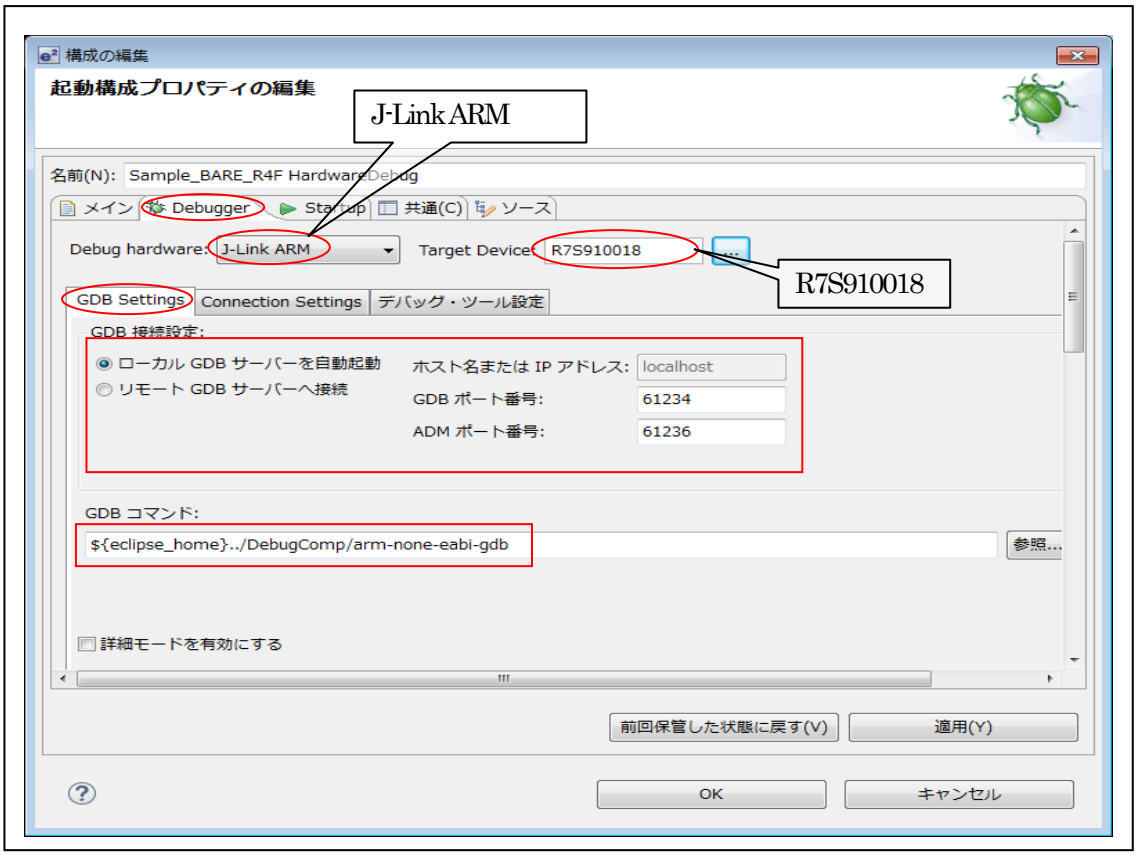
２）構成タイプの選択



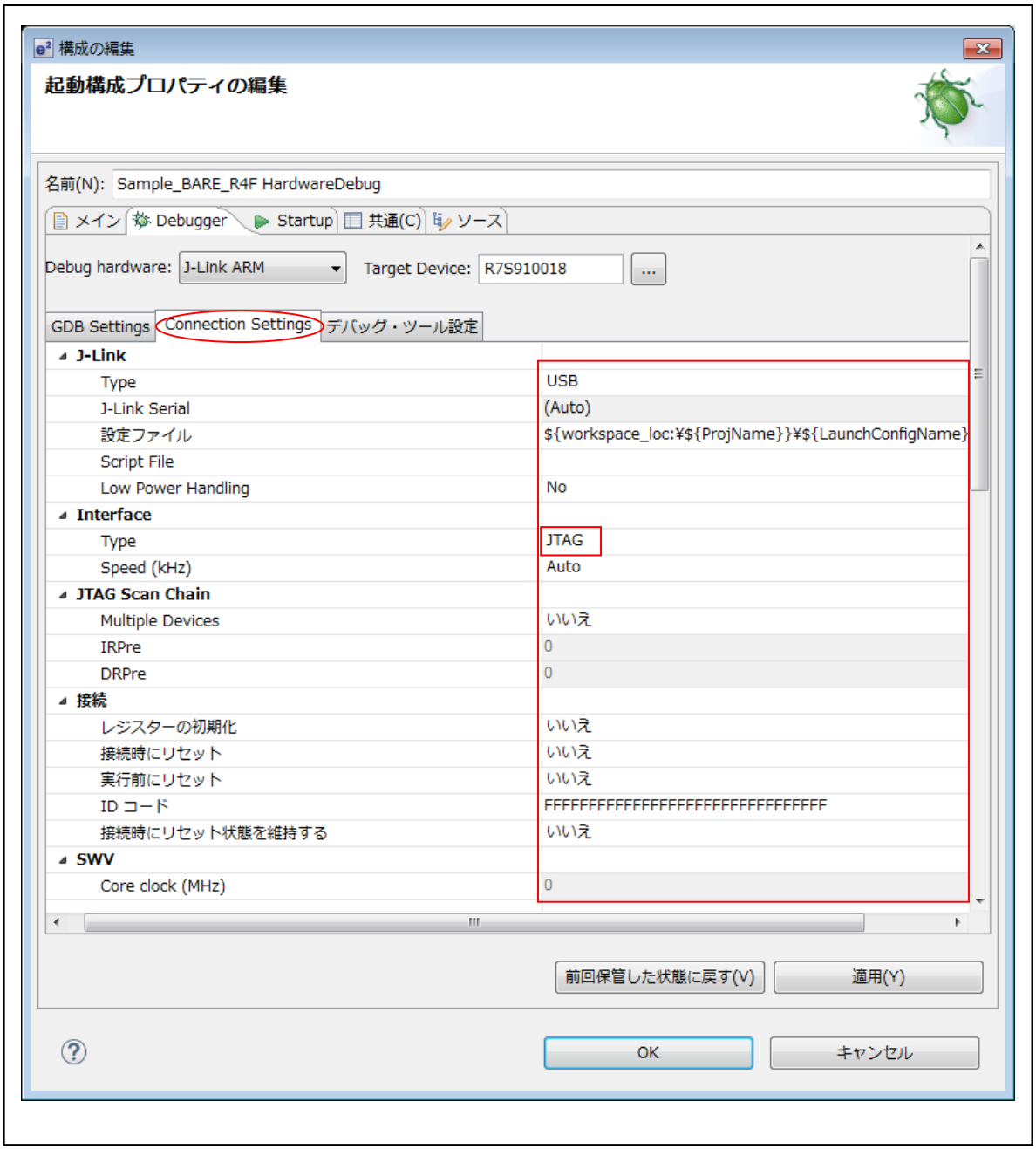
3）起動構成プロパティの編集(メイン)（デフォルト）



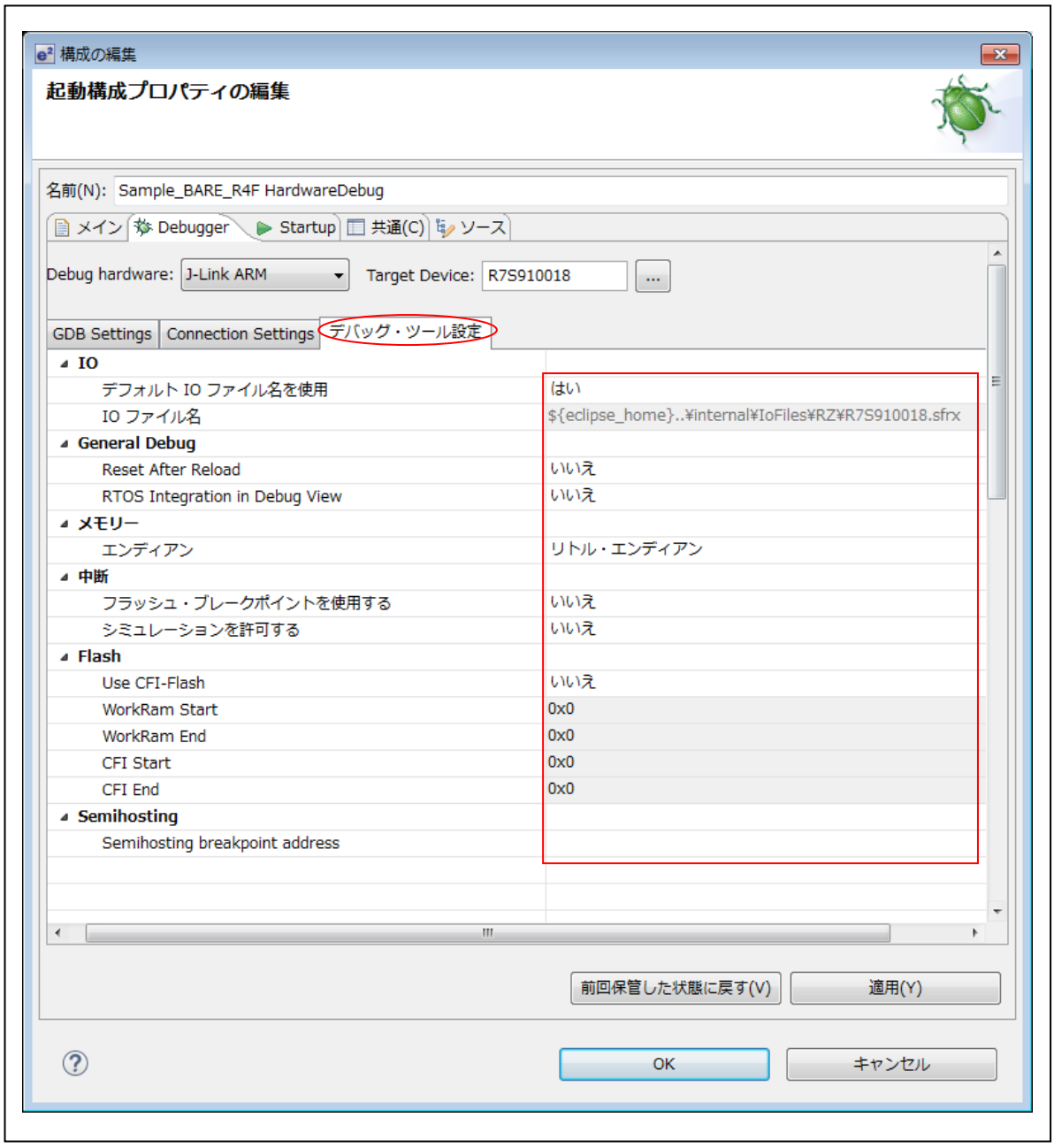
4）起動構成プロパティの編集(Debugger[GDB Settings])



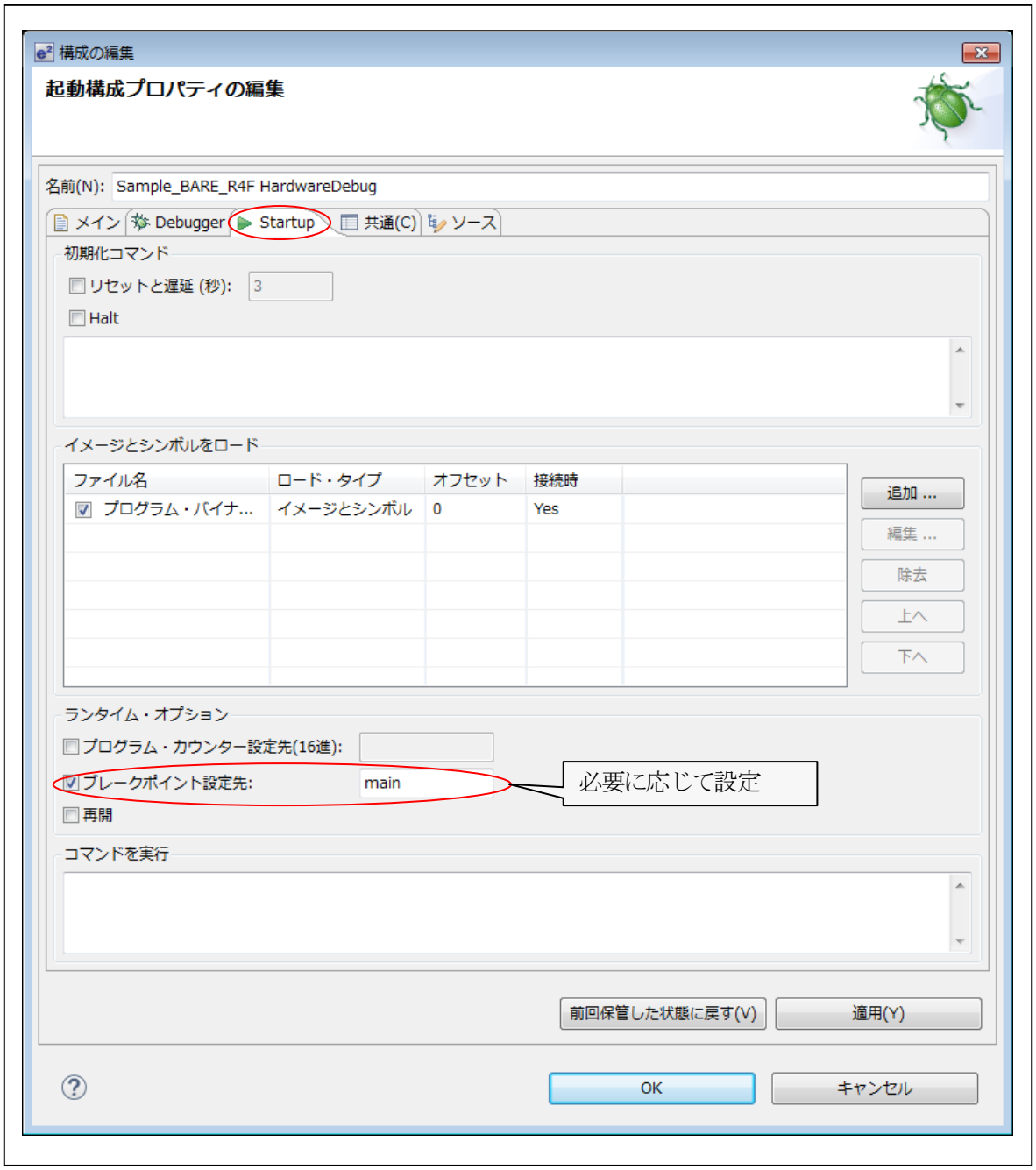
5）起動構成プロパティの編集(Debugger[Connection Settings]) （デフォルト）



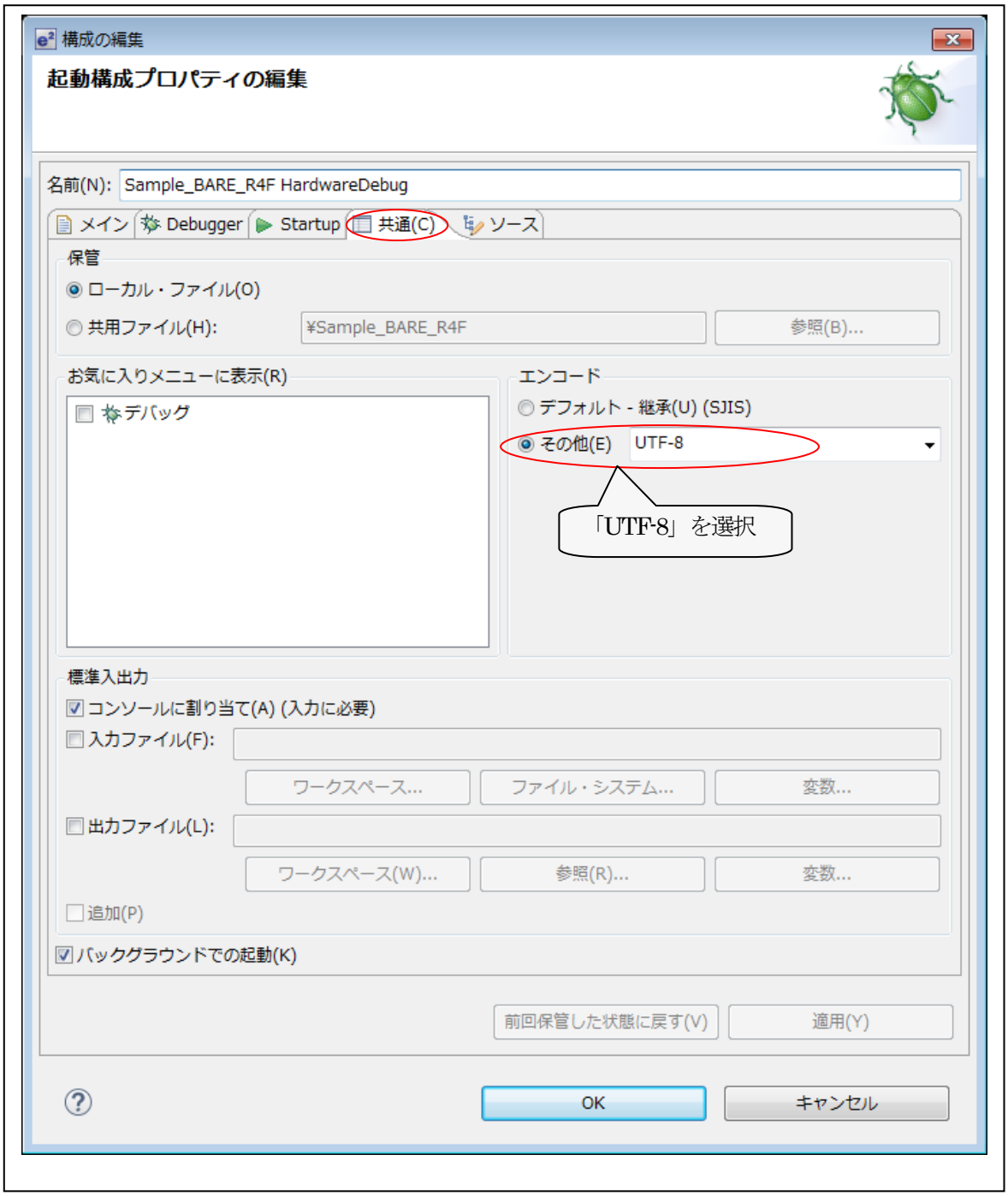
6）起動構成プロパティの編集(Debugger[デバッグ・ツール設定])



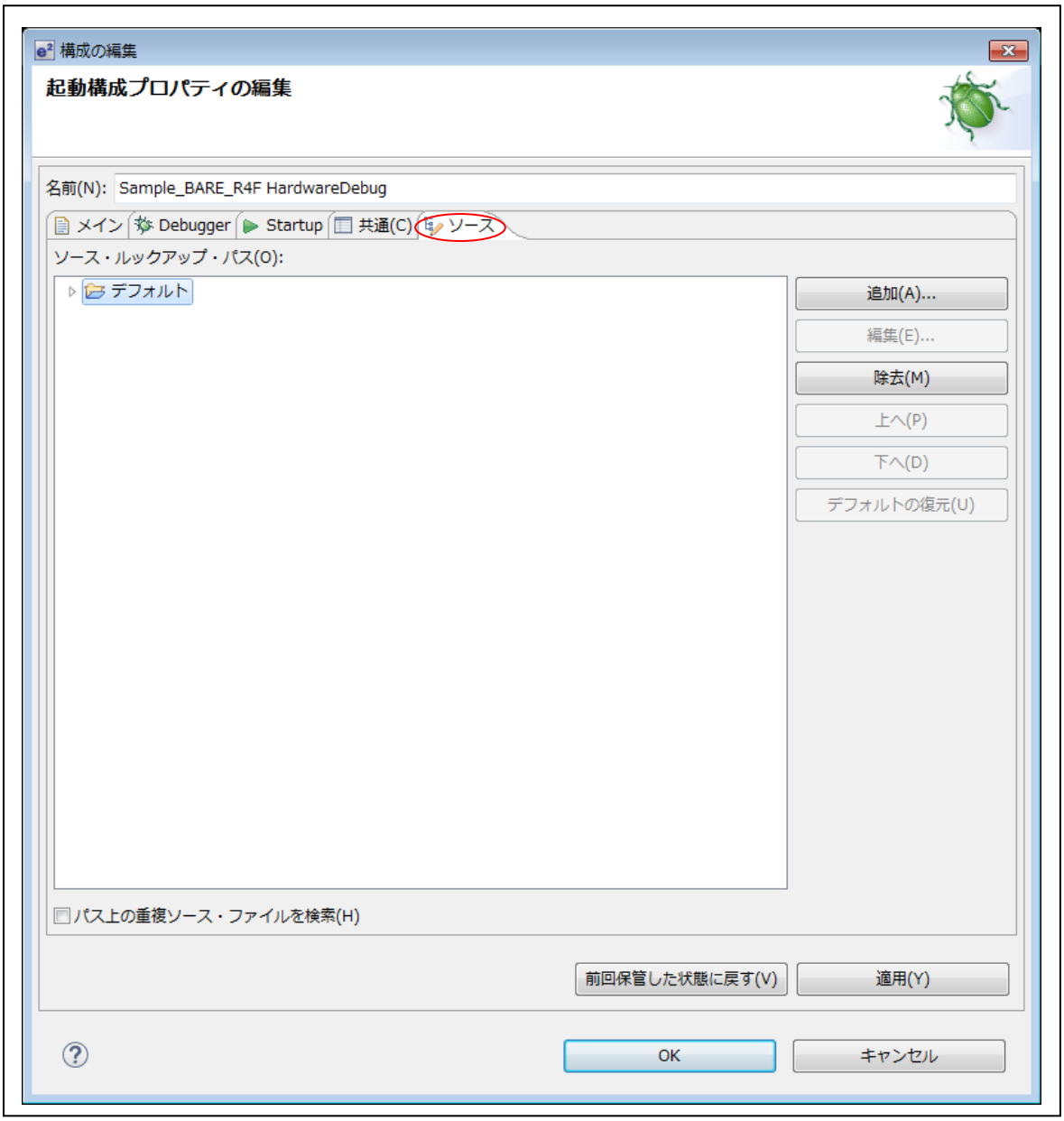
7）起動構成プロパティの編集(Startup)



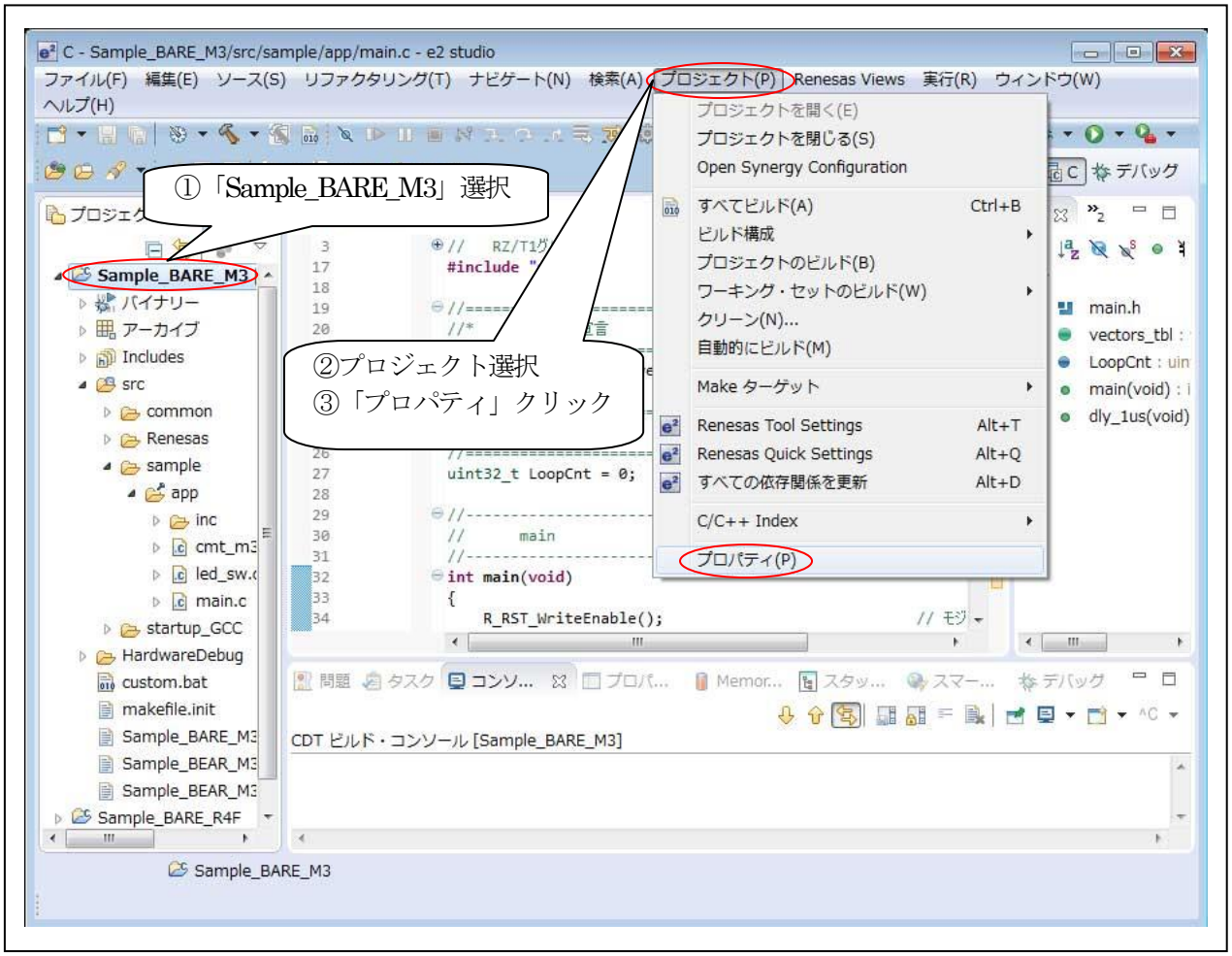
8）起動構成プロパティの編集(共通)



9）起動構成プロパティの編集(ソース)（デフォルト）

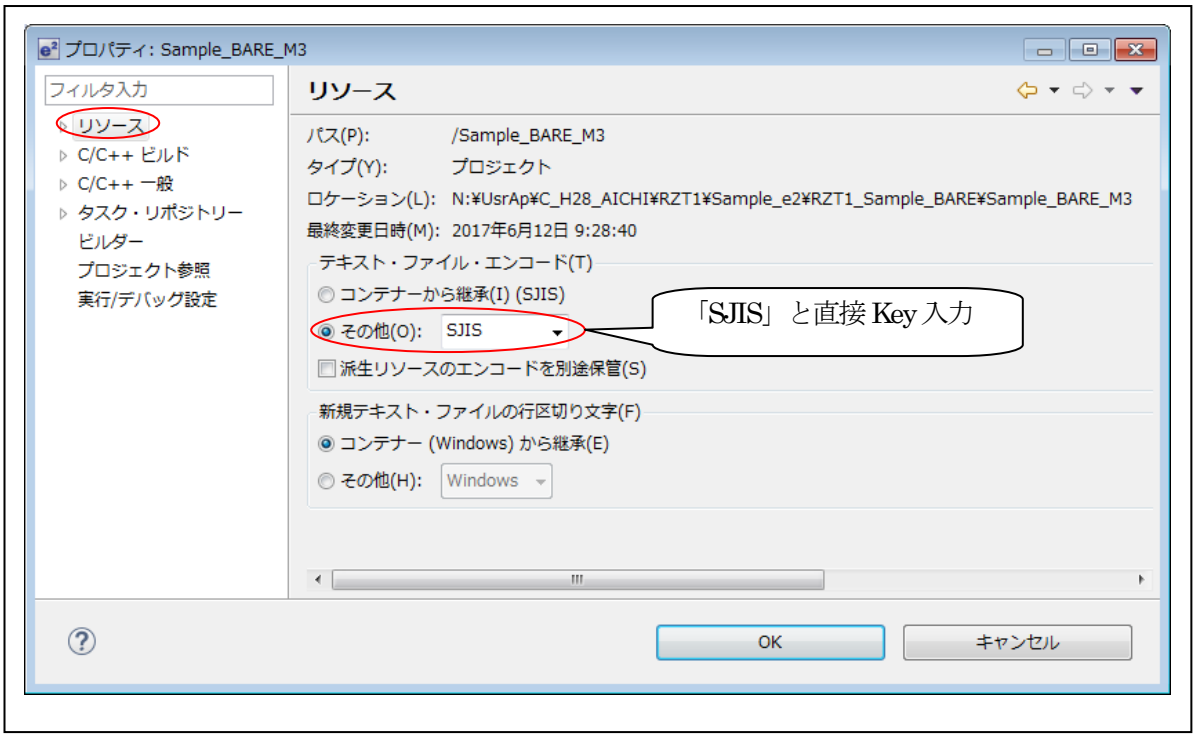


２－２．コア【M3】側の確認


①「Sample_BARE_M3」選択
②プロジェクト選択
③「プロパティ」クリック
C - Sample_BARE_M3/src/sample/app/main.c - e2 studio
ファイル(F) 編集(E) ソース(S) リファクタリング(T) ナビゲート(N) 検索(A) プロジェクト(P) Renesas Views 実行(R) ウィンドウ(W)
ヘルプ(H)
プロジェクトを開く(E)
プロジェクトを閉じる(S)
Open Synergy Configuration
すべてビルド(A) Ctrl+B
ビルド構成
プロジェクトのビルド(B)
ワーキング・セットのビルド(W)
クリーン(N)...
自動的にビルド(M)
Make ターゲット
Renesas Tool Settings Alt+T
Renesas Quick Settings Alt+Q
すべての依存関係を更新 Alt+D
C/C++ Index
プロパティ(P)
Sample_BARE_M3
バイナリー
アーカイブ
Includes
src
common
Renesas
sample
app
inc
startup_GCC
HardwareDebug
custom.bat
makefile.init
Sample_BARE_R4F
CDT ビルド・コンソール [Sample_BARE_M3]
デバッグ
main.h
vectors_tbl :
LoopCnt : uin
main(void) : i
dly_1us(void)
uint32_t LoopCnt = 0;
int main(void)
{
R_RST_WriteEnable();

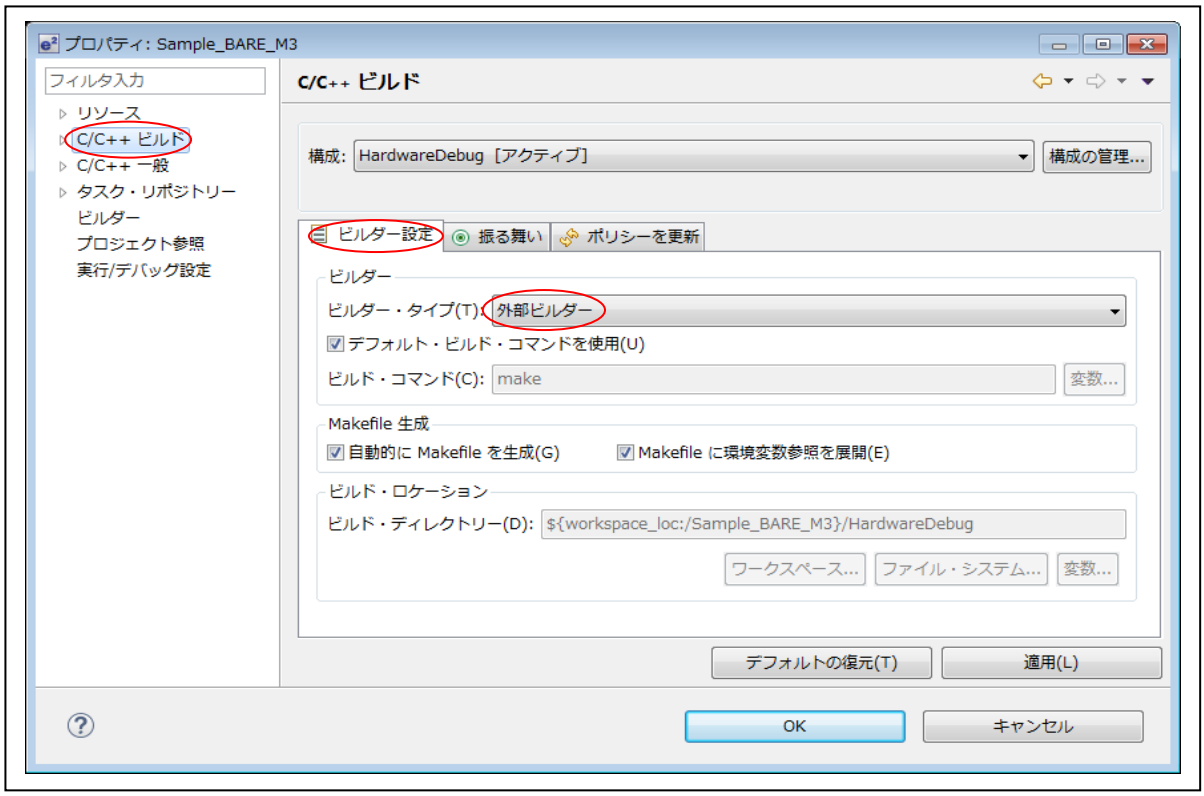
２－２－１．リソース

１）テキスト・ファイル・エンコード

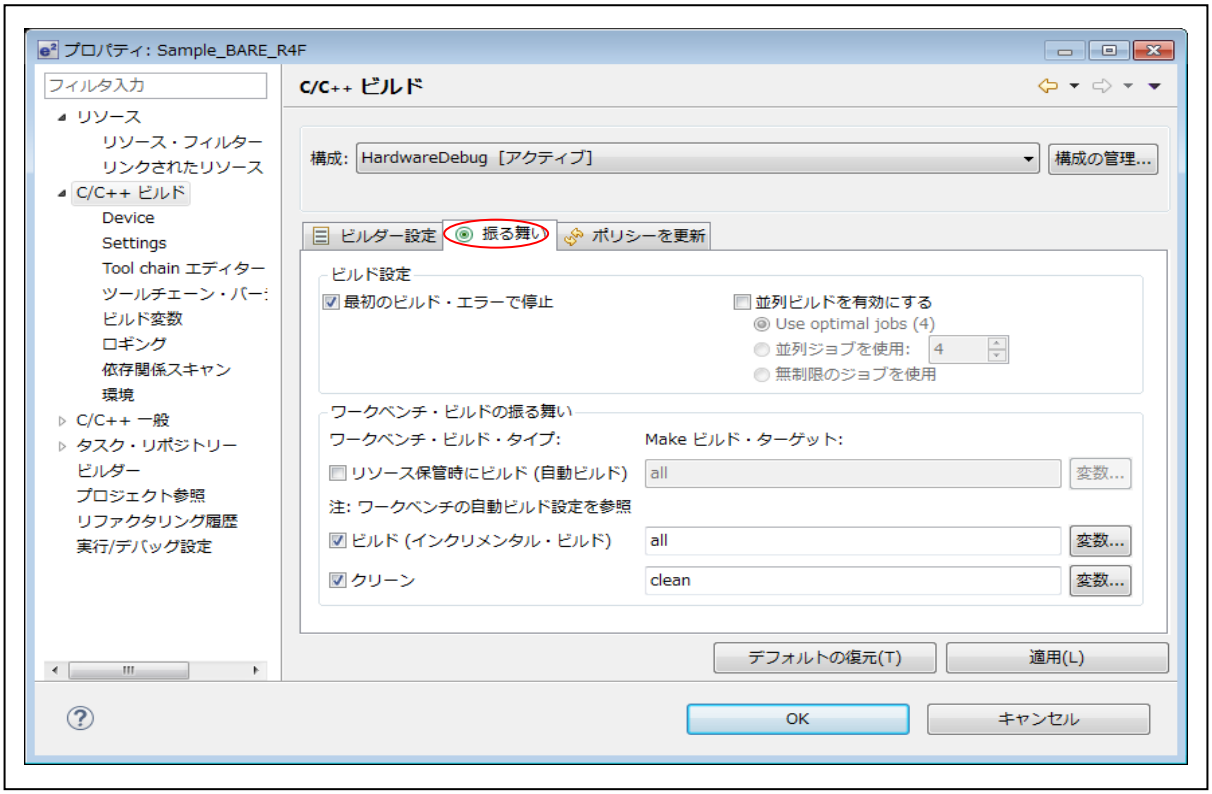


２－２－２．C/C++ビルド（TOP）

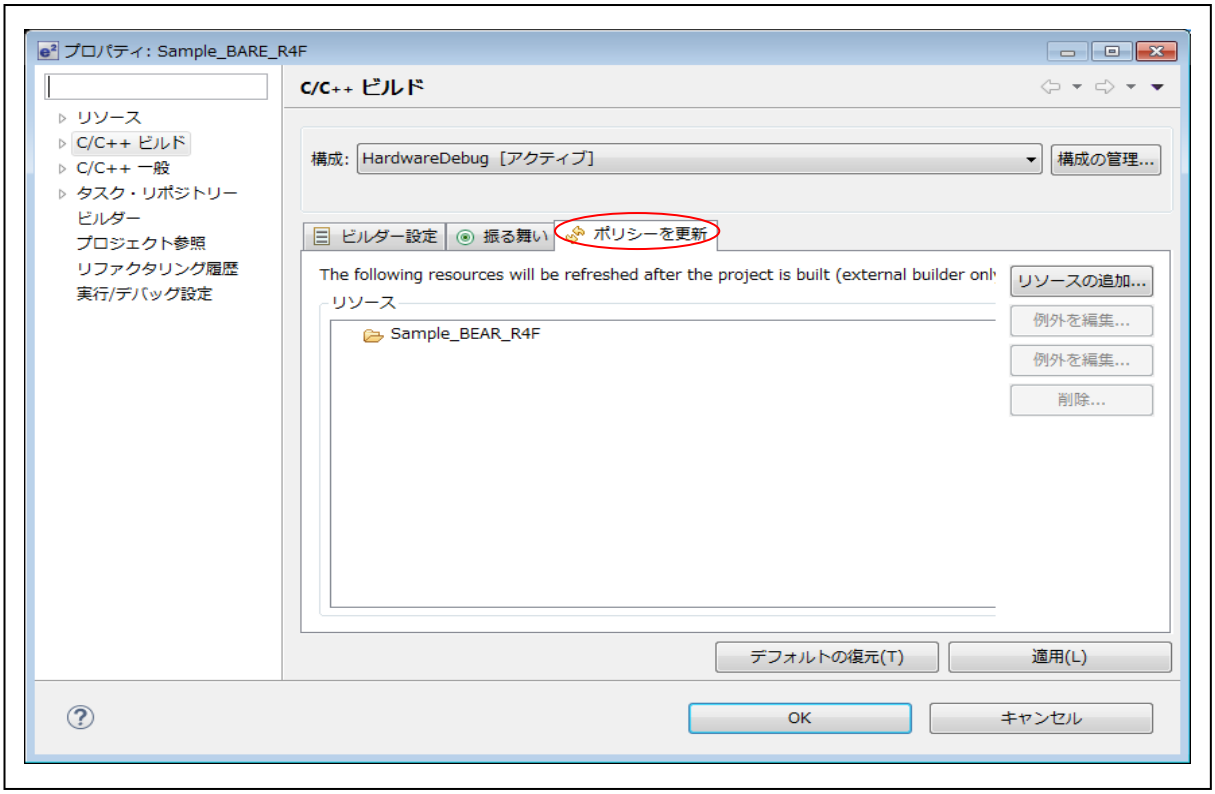
１）ビルダー設定（デフォルト）



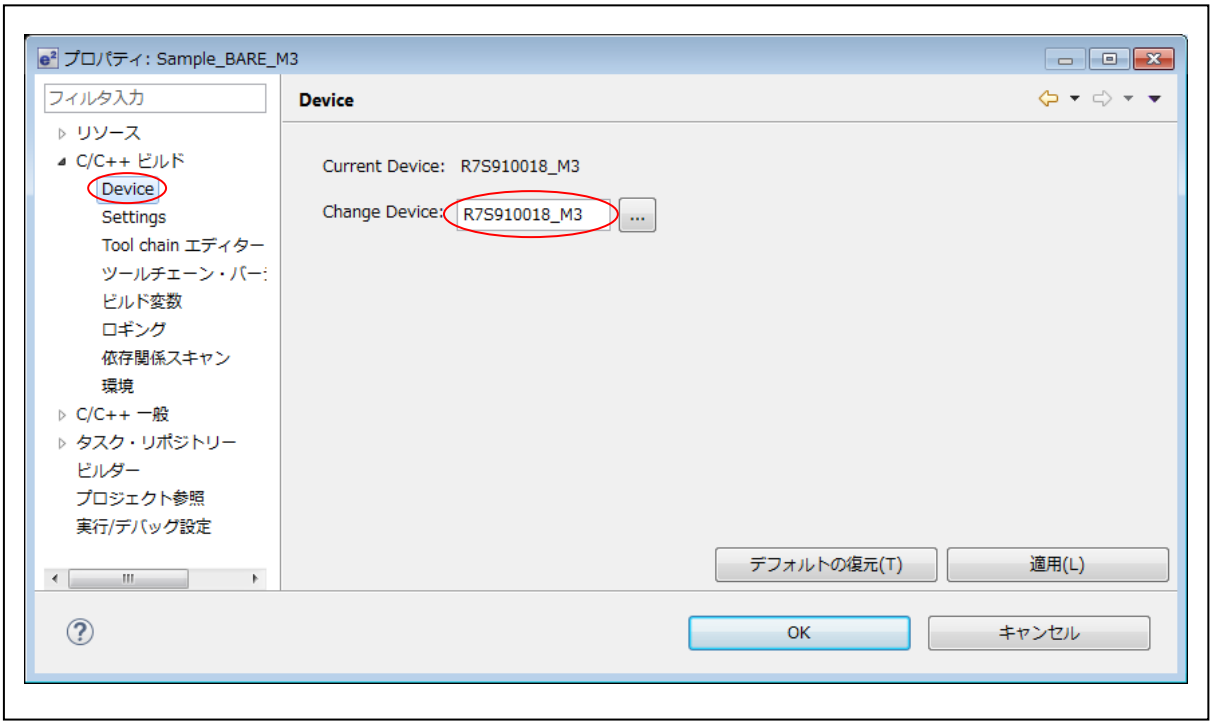
２）振る舞い（デフォルト）



3）ポリシーを更新（デフォルト）

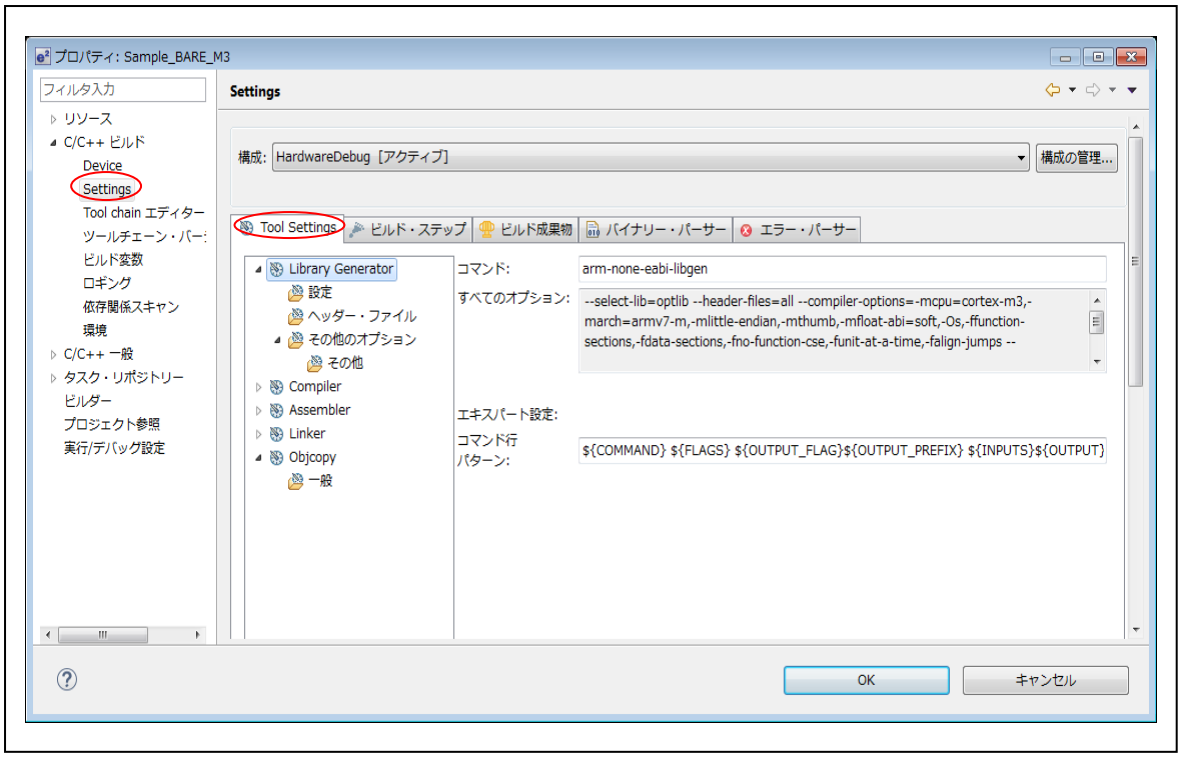


2－2－3．C/C++ビルド（Device）

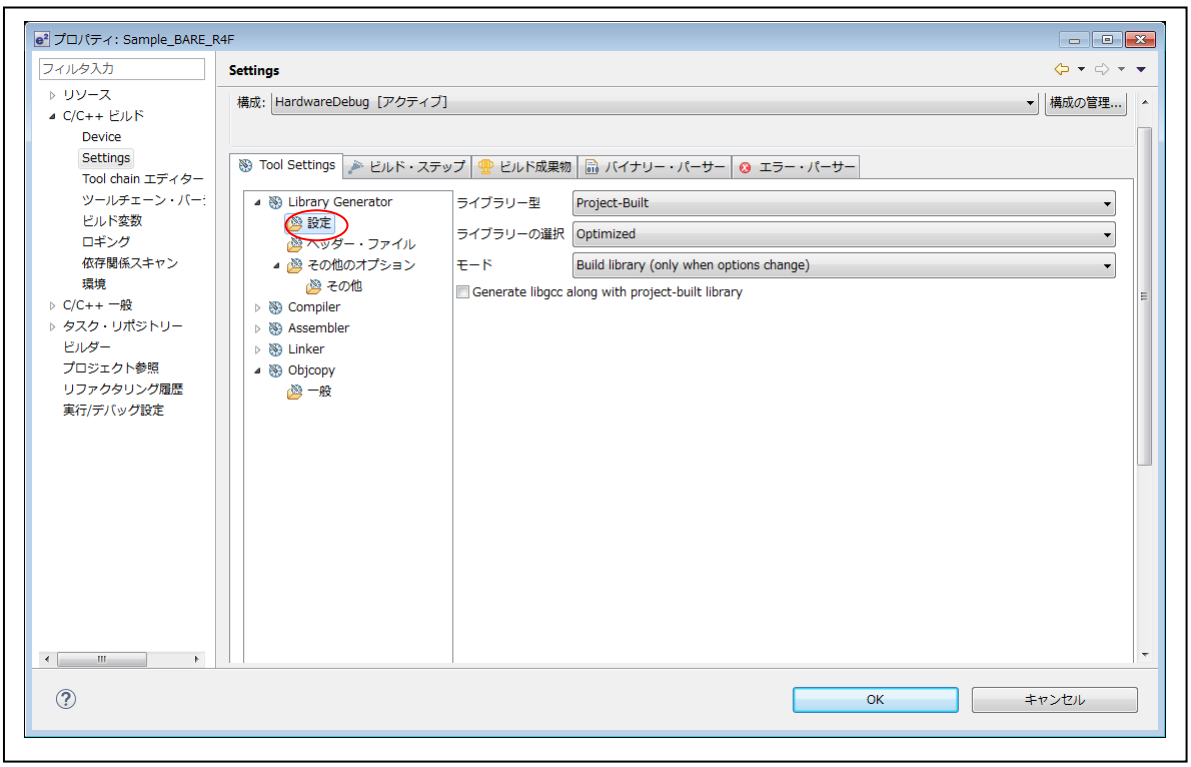


２－２－４．C/C++ビルド（Settings）

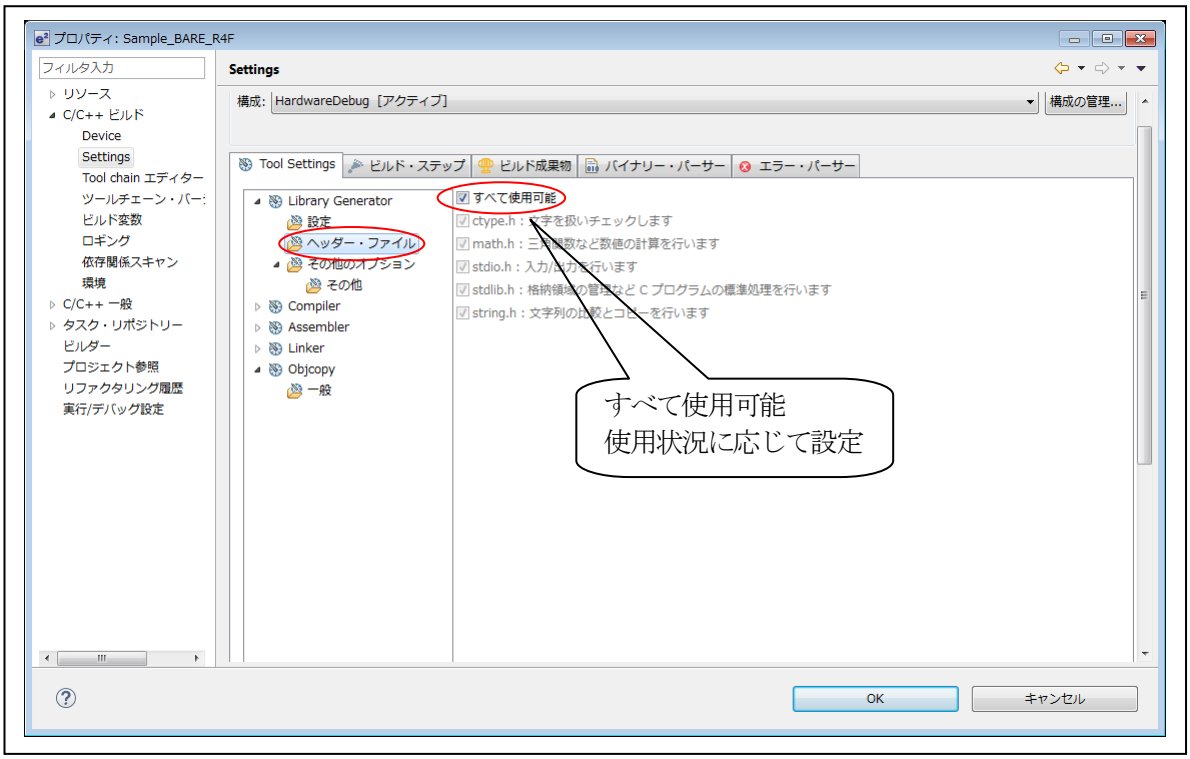
２－２－４－１．Tool　Settings(Library Generator)



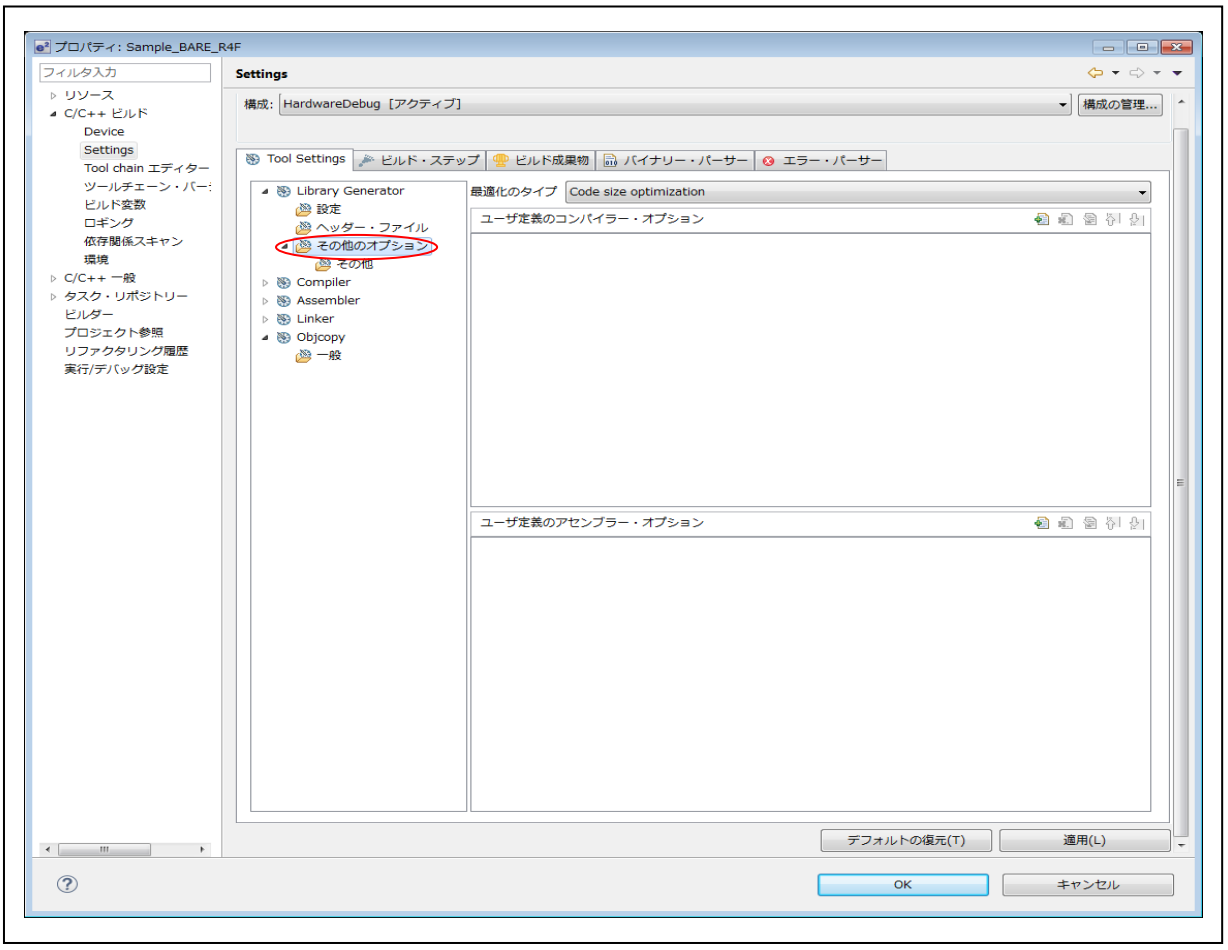
1）設定（デフォルト）



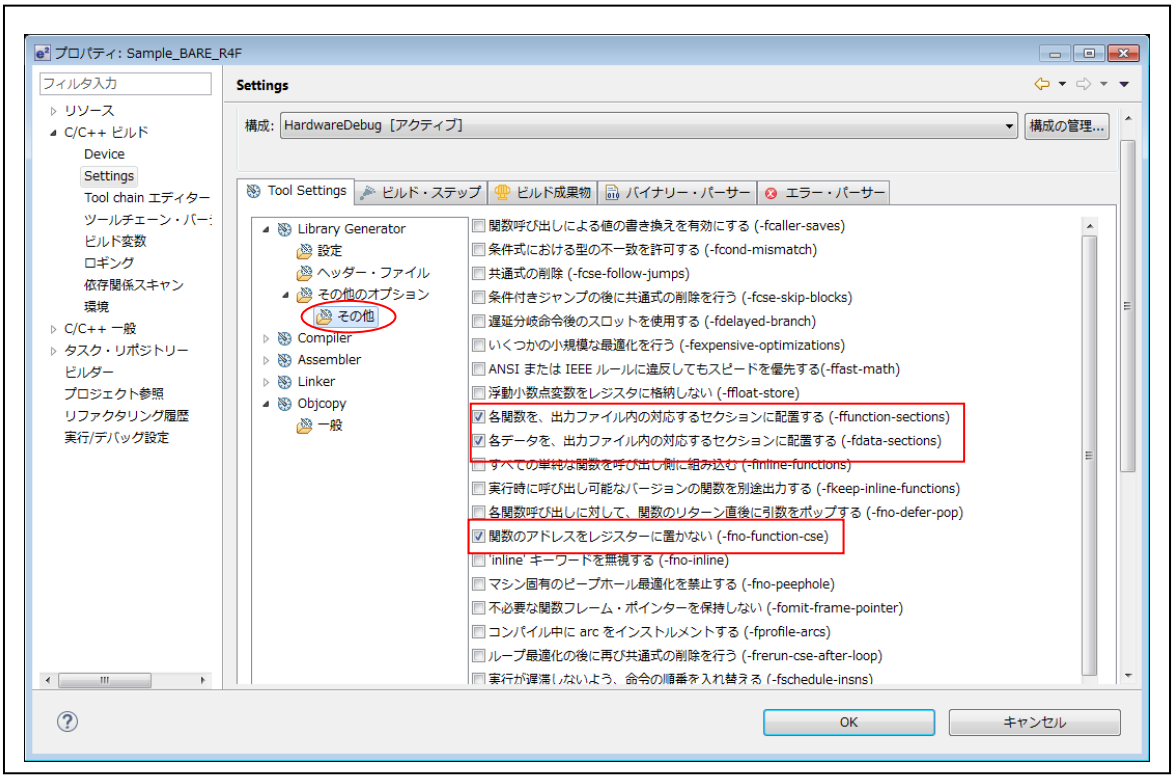
2）ヘッダー・ファイル



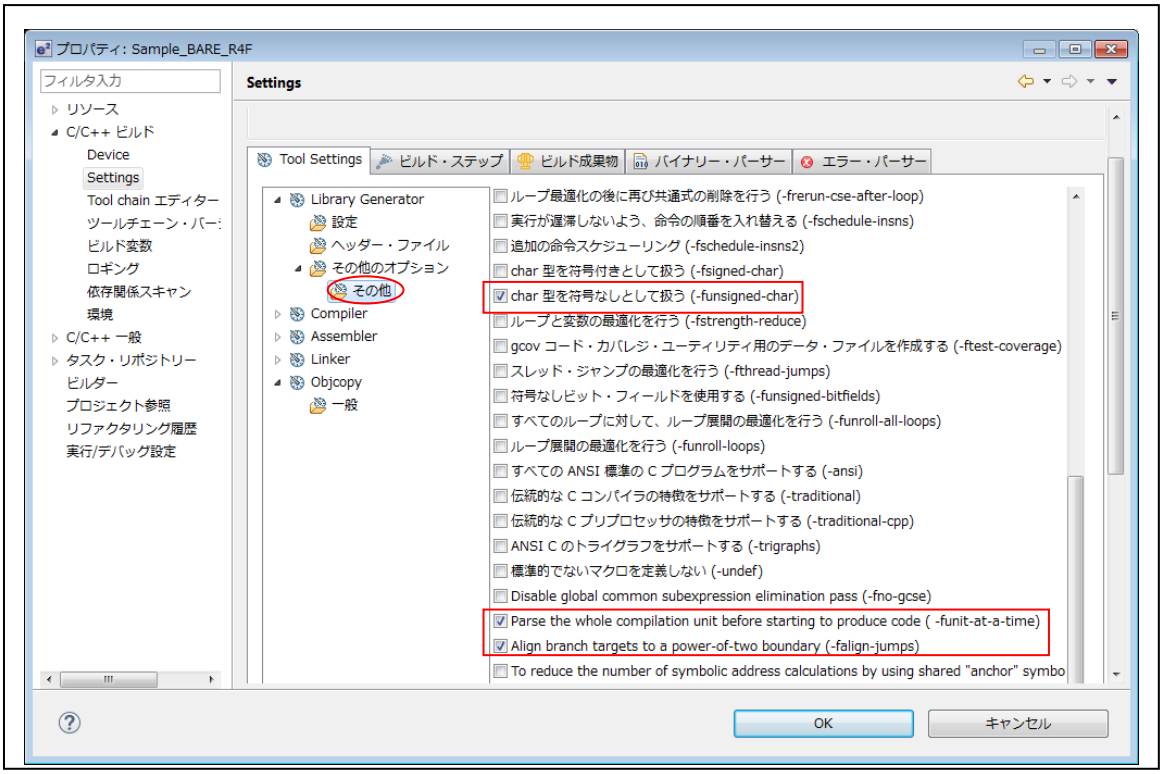
3）その他のオプション（デフォルト）



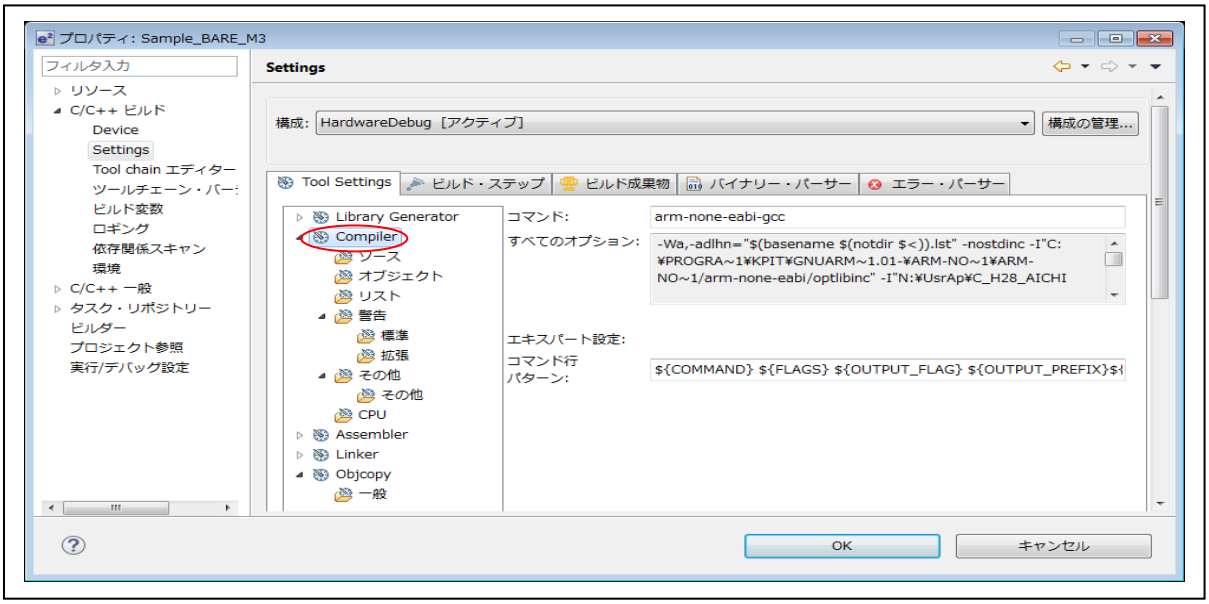
4）その他（1／2）



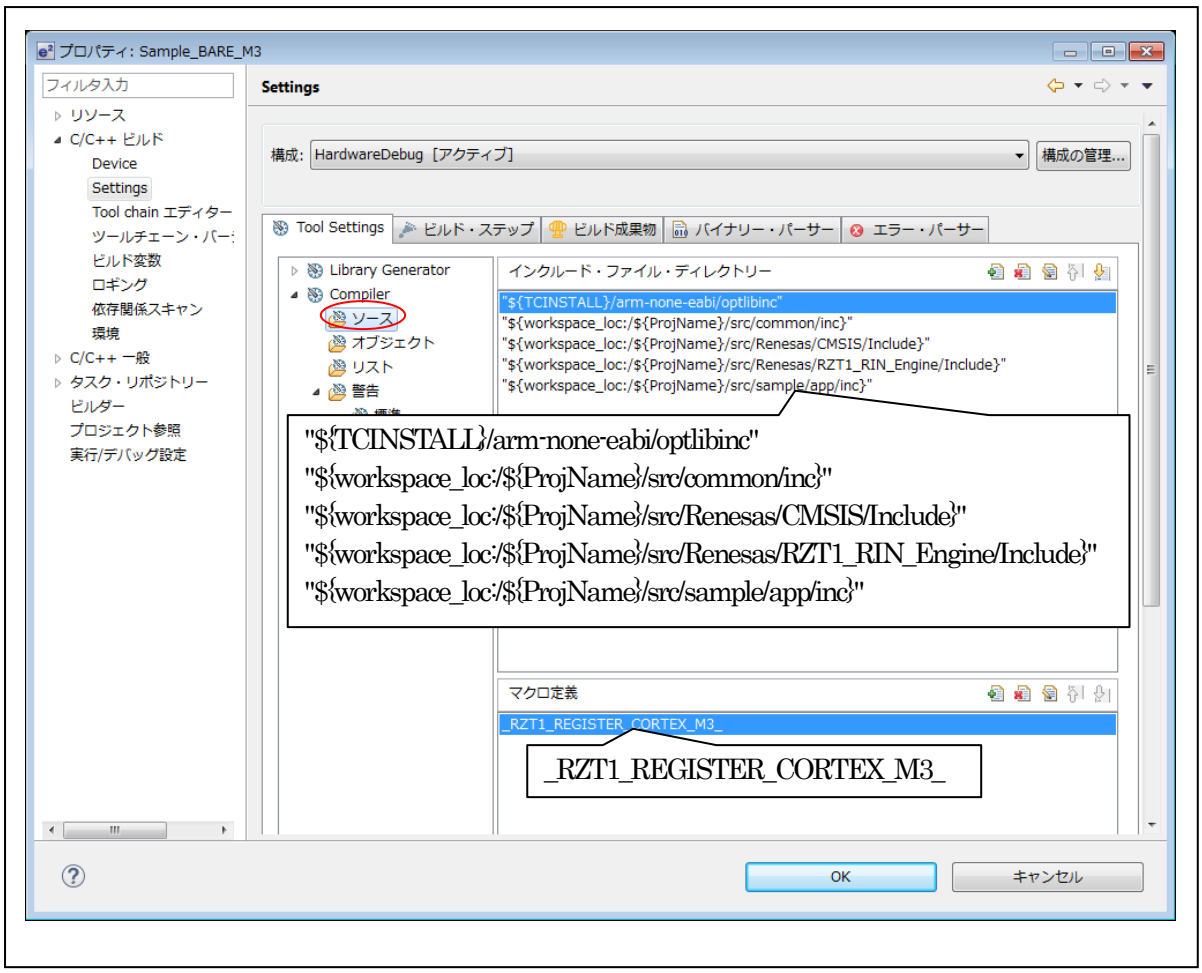
5）その他（2／2）



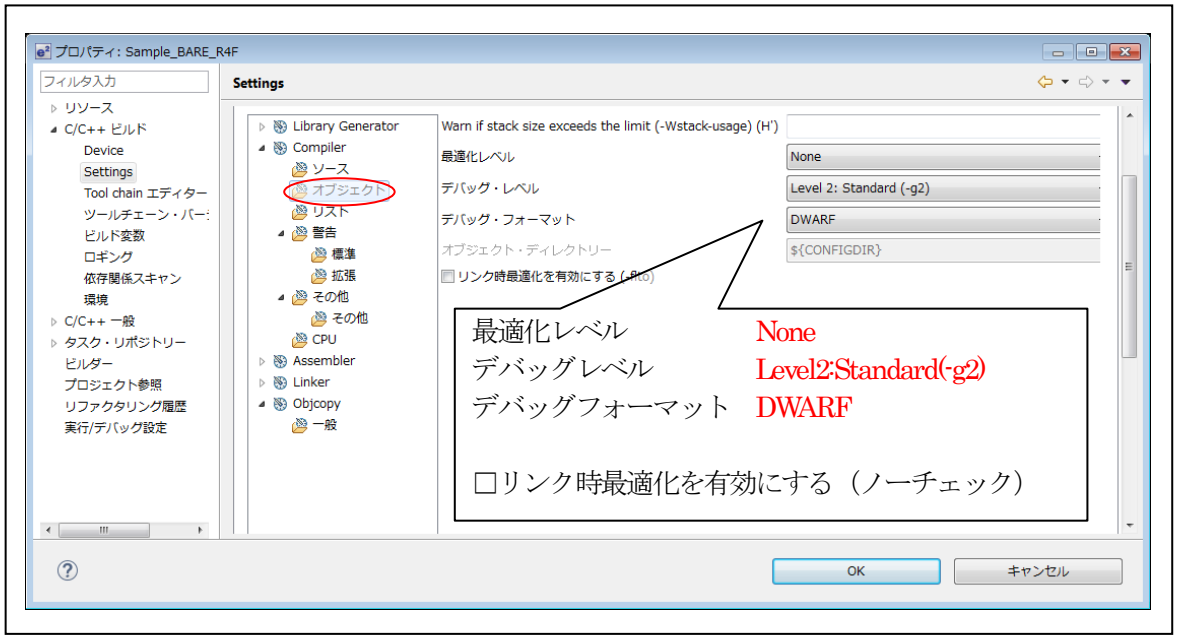
2－2－4－2．Tool　Settings(Compiler)



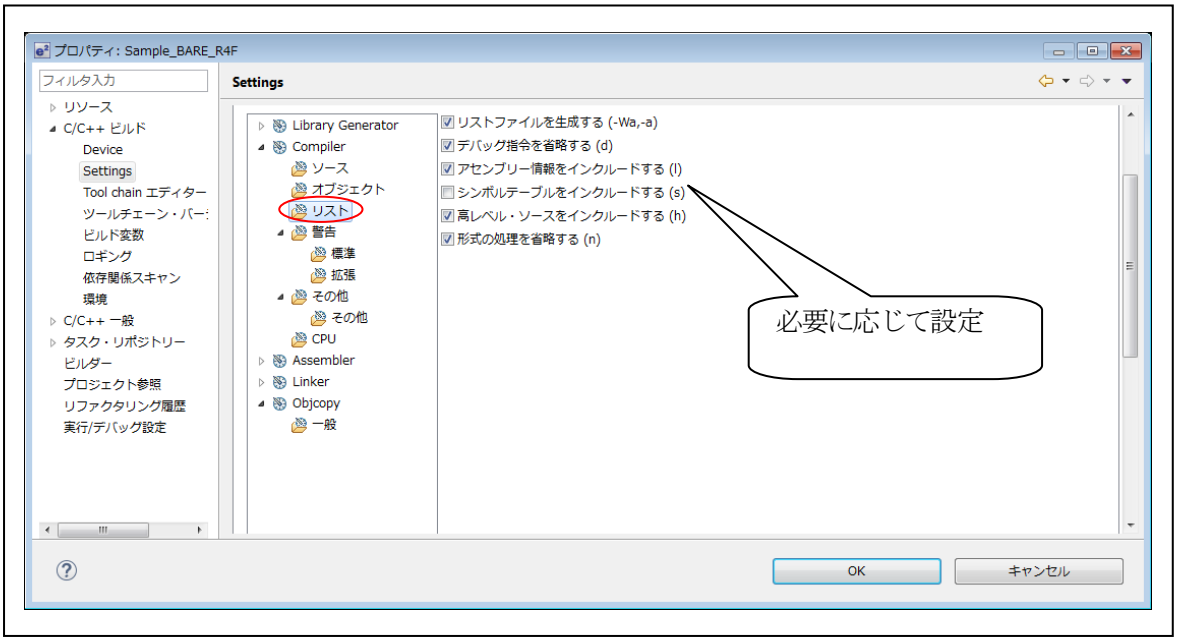
1）ソース



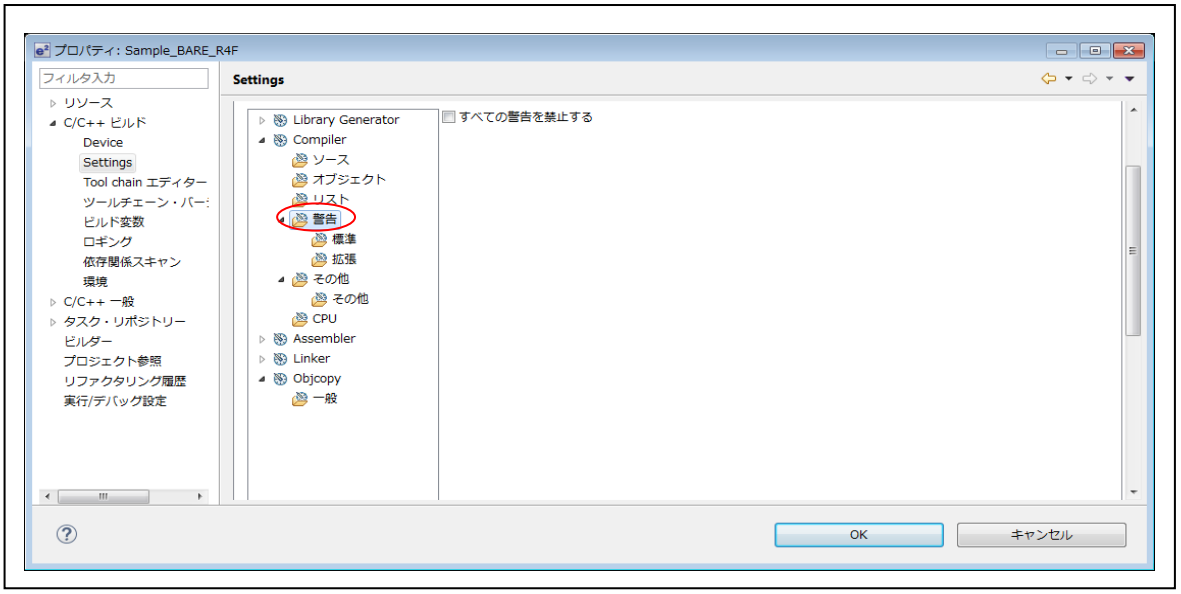
2）オブジェクト



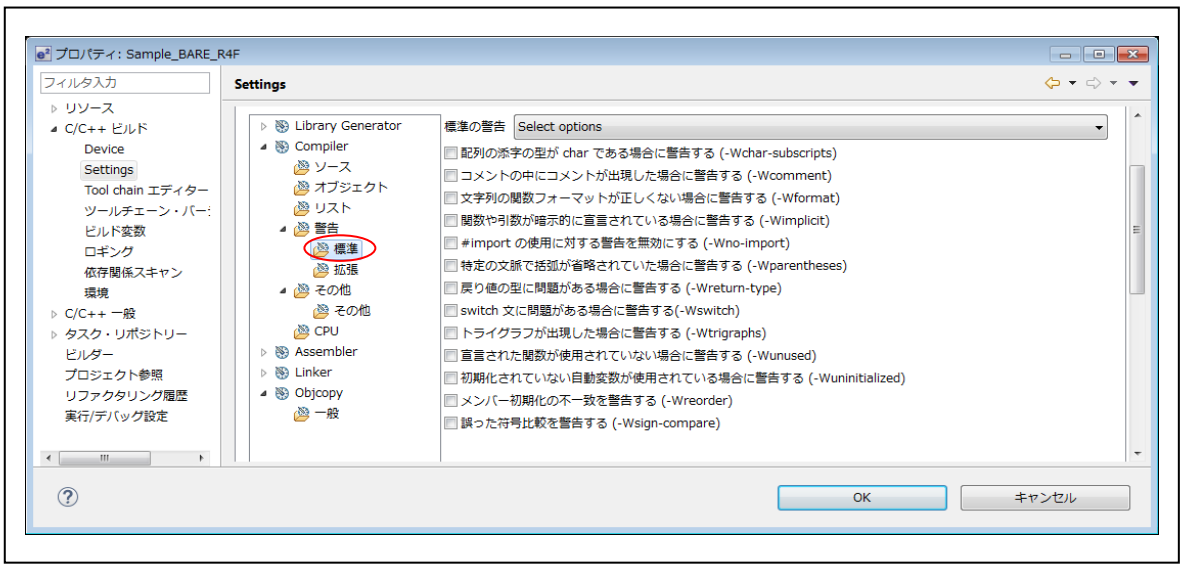
3）リスト（デフォルト）



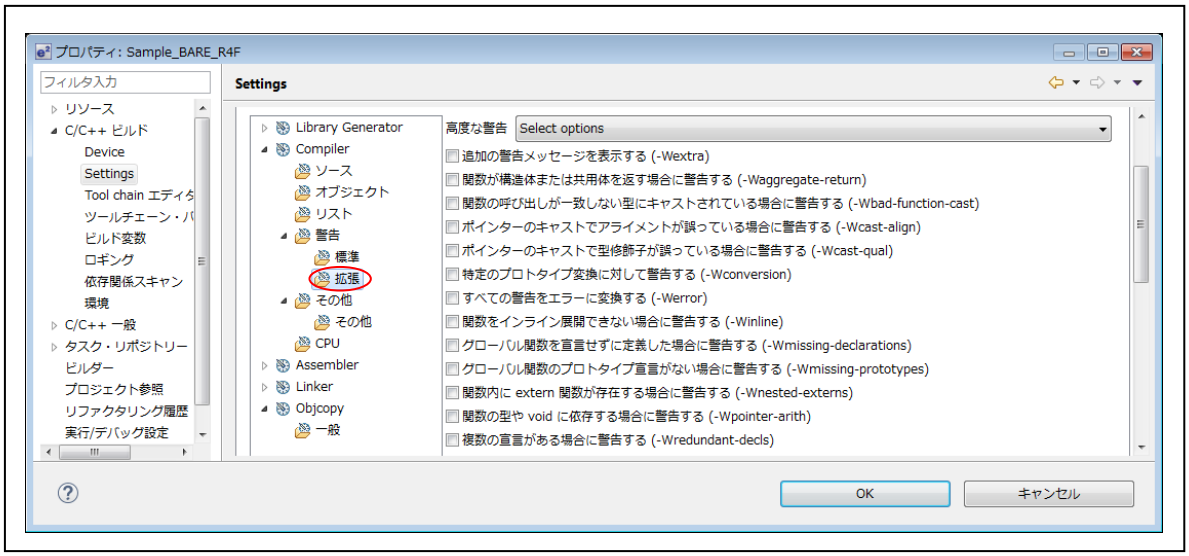
4）警告（デフォルト）



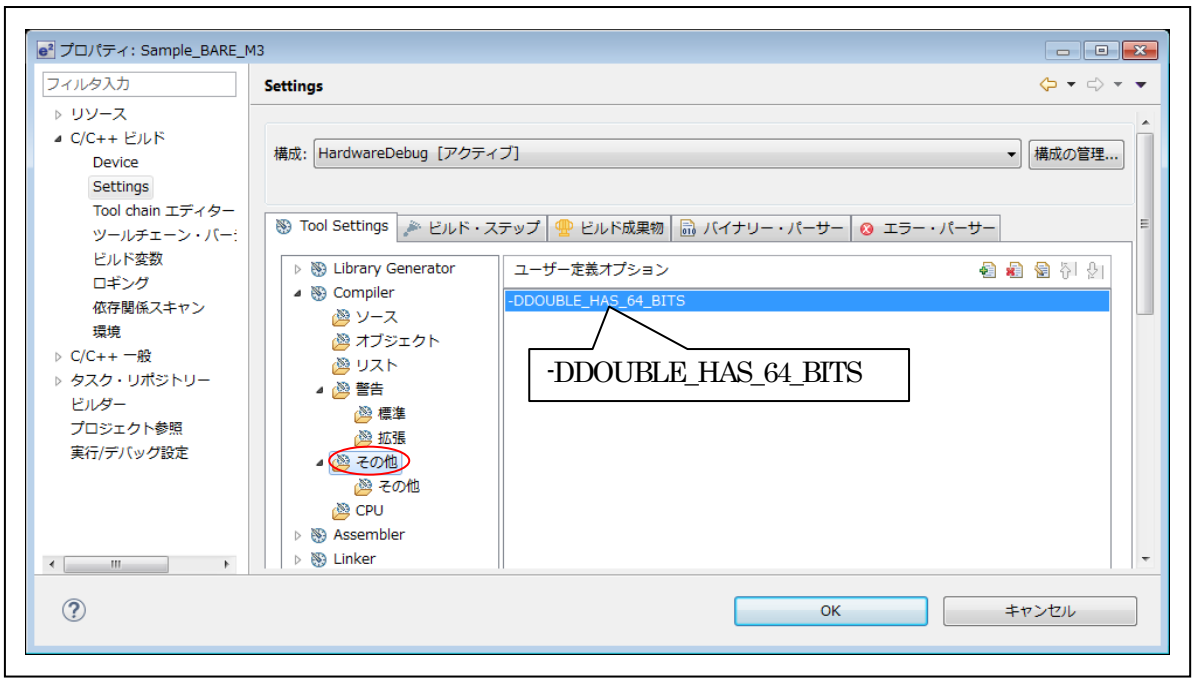
5）警告ー標準（デフォルト）



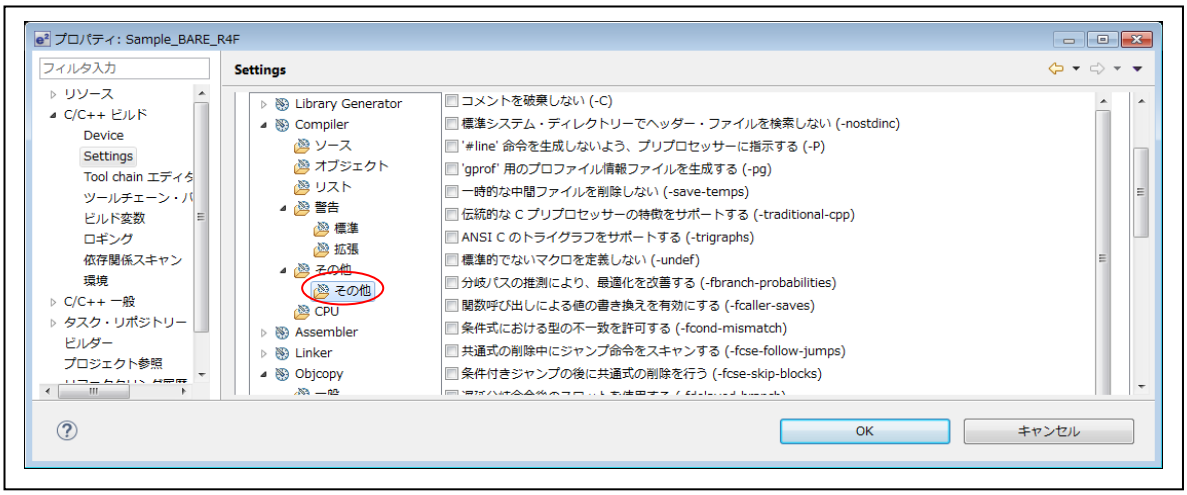
6）警告ー拡張（デフォルト）



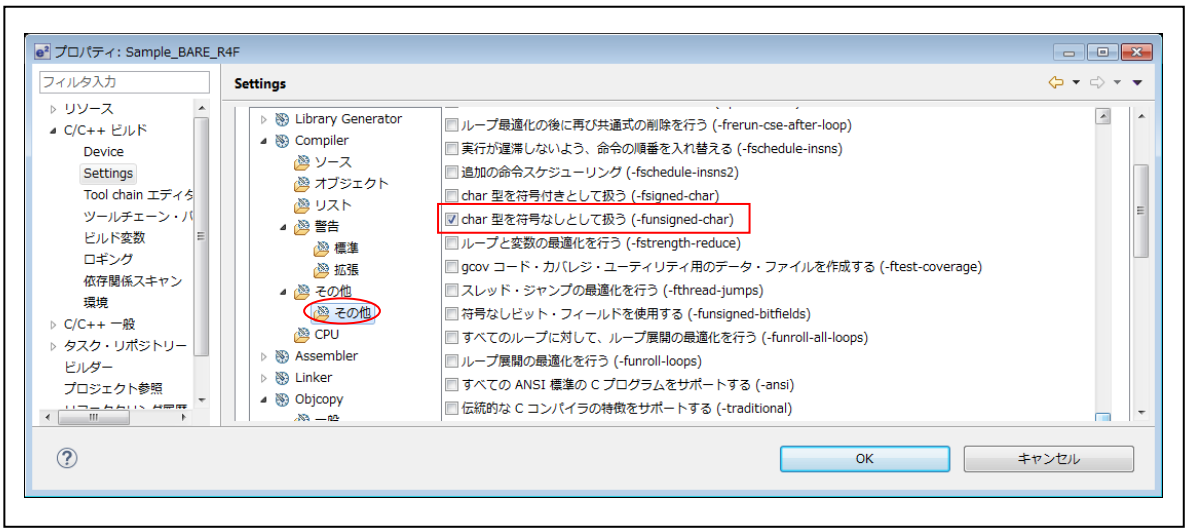
7）その他



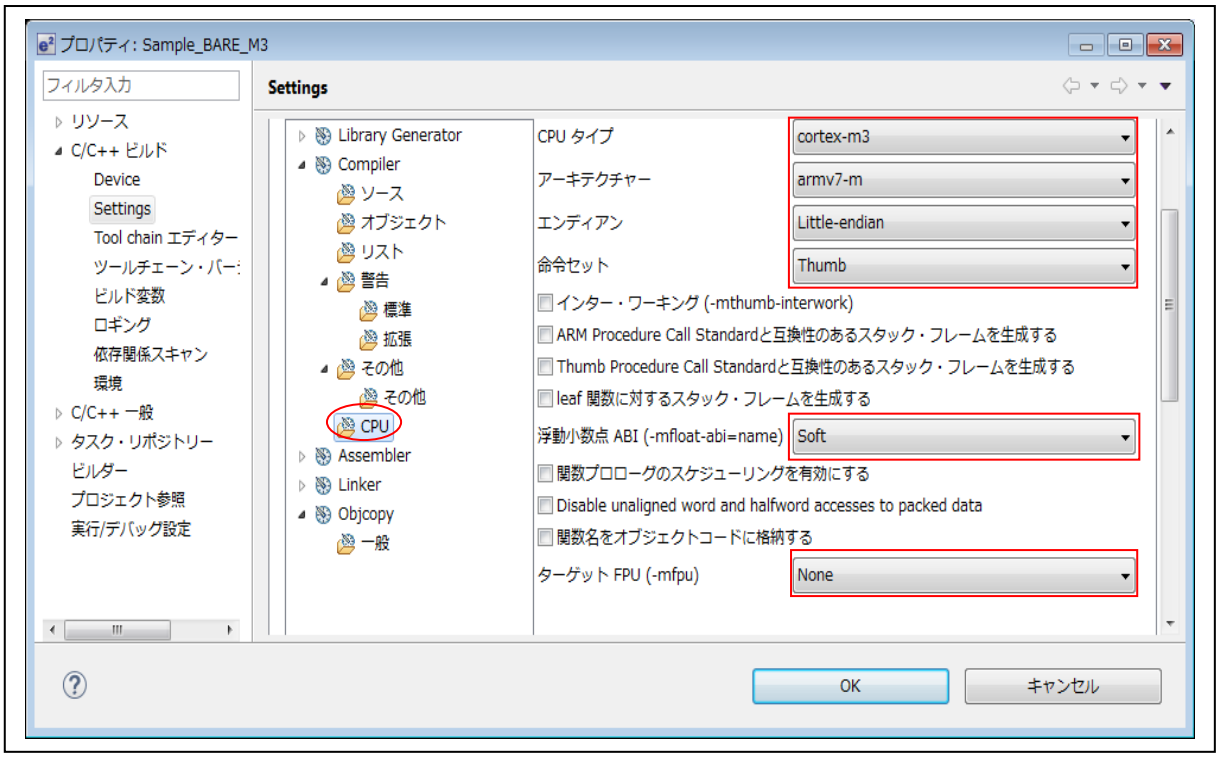
8）その他ーその他（1／2）



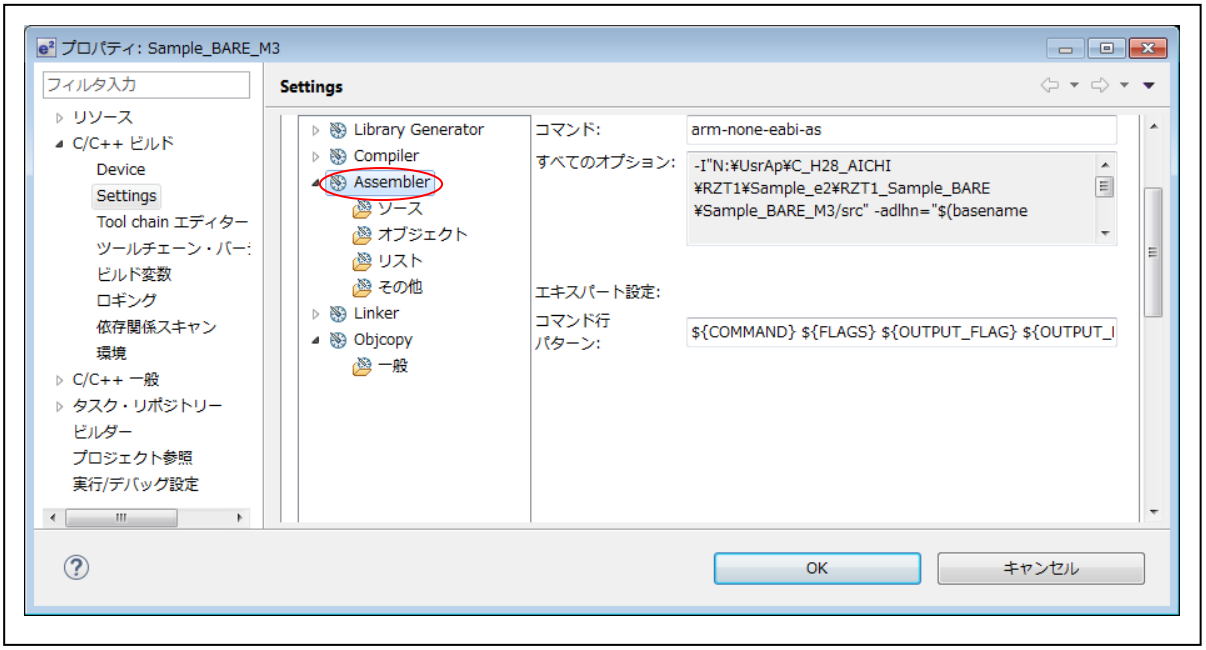
8）その他ーその他（2／2）



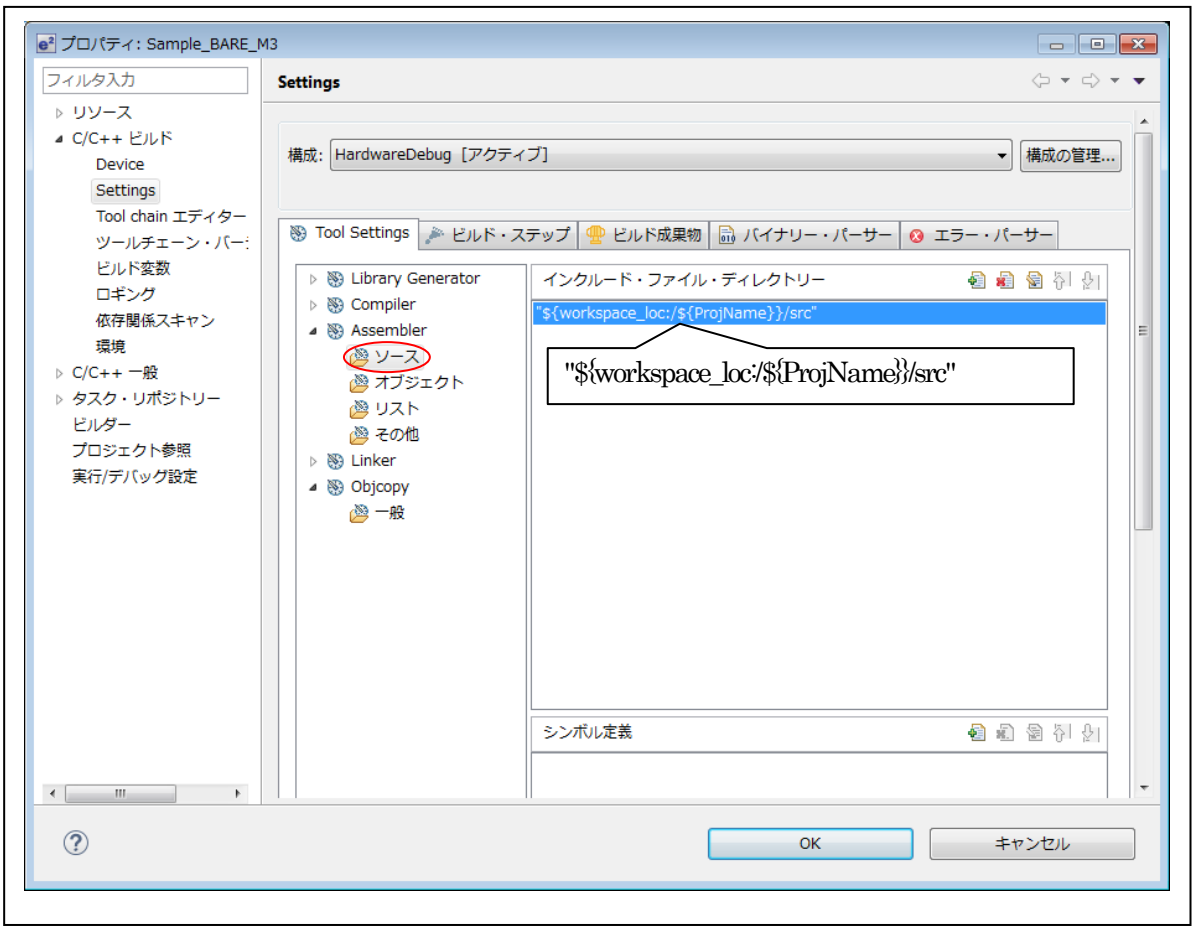
9）CPU


プロパティ: Sample_BARE_M3
フィルタ入力
リソース
C/C++ ビルド
Device
Settings
Tool chain エディター
ツールチェーン・バー
ビルド変数
ロギング
依存関係スキャン
環境
C/C++ 一般
タスク・リポジトリー
ビルダー
プロジェクト参照
実行/デバッグ設定
Settings
Library Generator
Compiler
ソース
オブジェクト
リスト
警告
標準
拡張
その他
その他
CPU
Assembler
Linker
Objcopy
一般
CPU タイプ
cortex-m3
アーキテクチャー
armv7-m
エンディアン
Little-endian
命令セット
Thumb
インター・ワーキング (-mthumb-interwork)
ARM Procedure Call Standardと互換性のあるスタック・フレームを生成する
Thumb Procedure Call Standardと互換性のあるスタック・フレームを生成する
leaf 関数に対するスタック・フレームを生成する
浮動小数点 ABI (-mfloat-abi=name)
Soft
関数プロローグのスケジューリングを有効にする
Disable unaligned word and halfword accesses to packed data
関数名をオブジェクトコードに格納する
ターゲット FPU (-mfpu)
None
OK
キャンセル

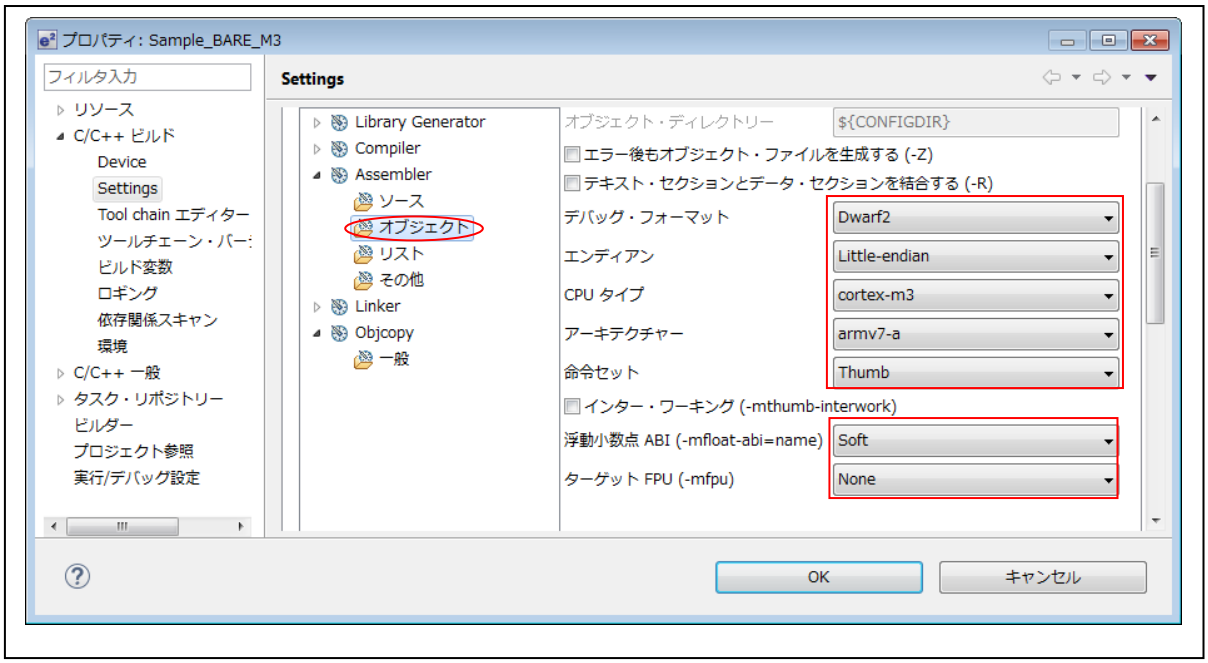
2－2－4－3．Tool　Settings(Assembler)



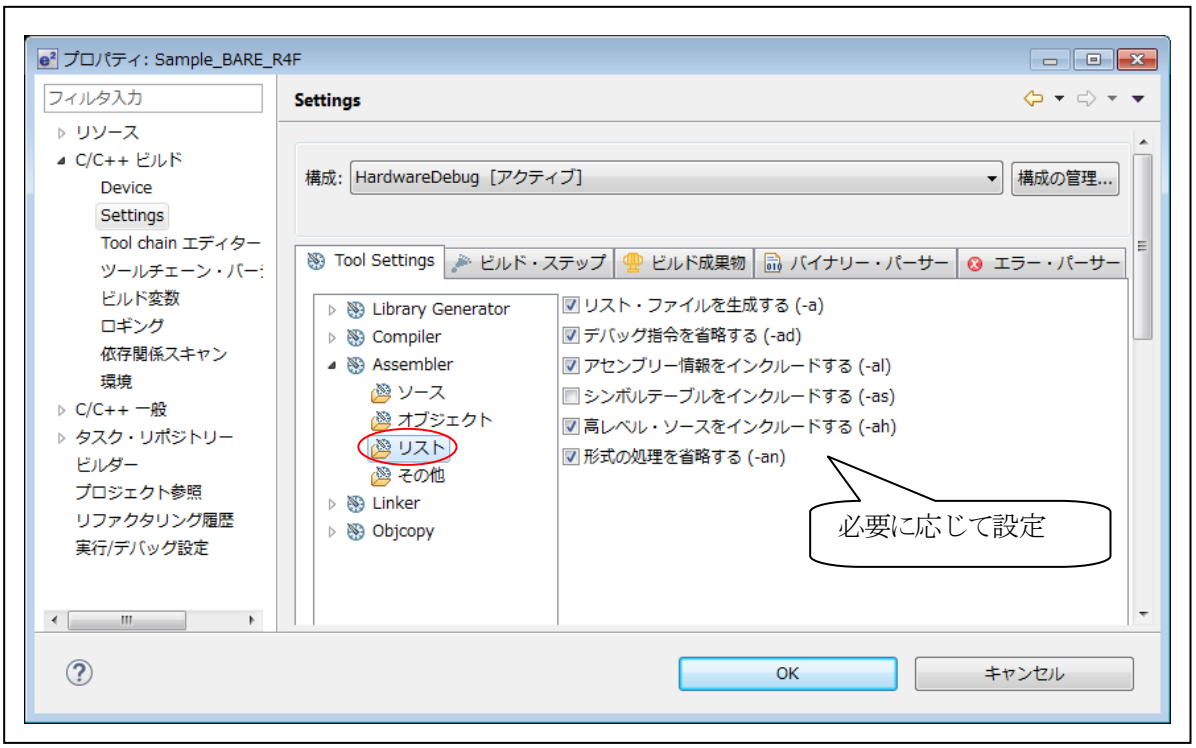
1）ソース



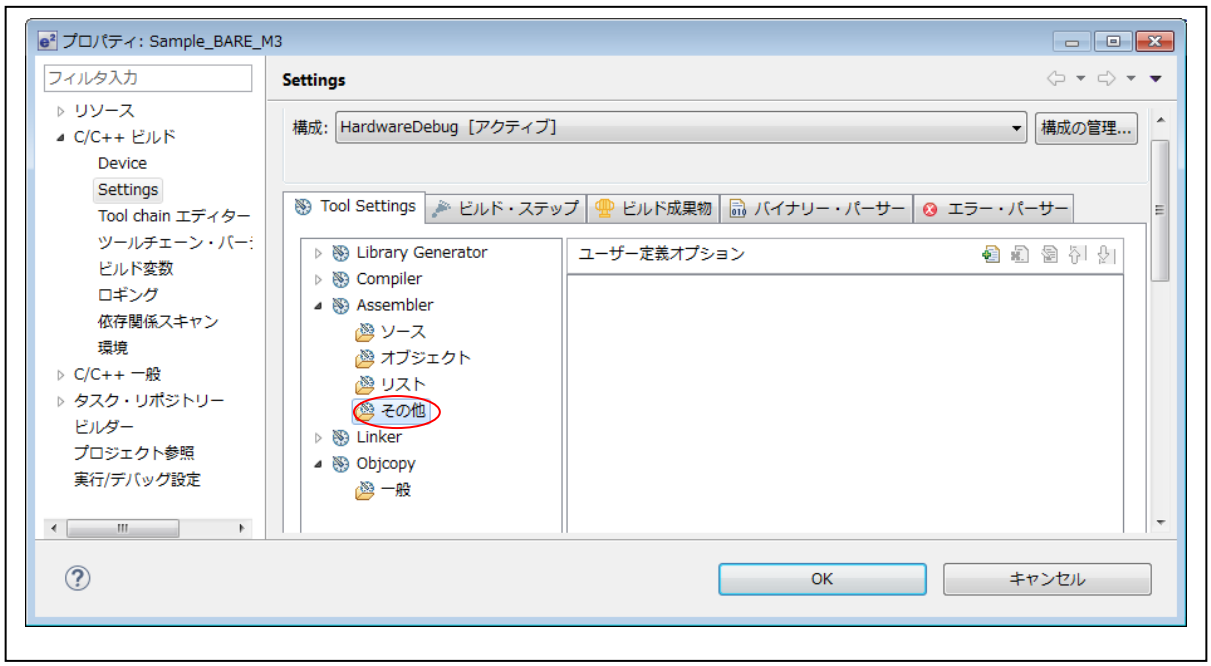
2）オブジェクト



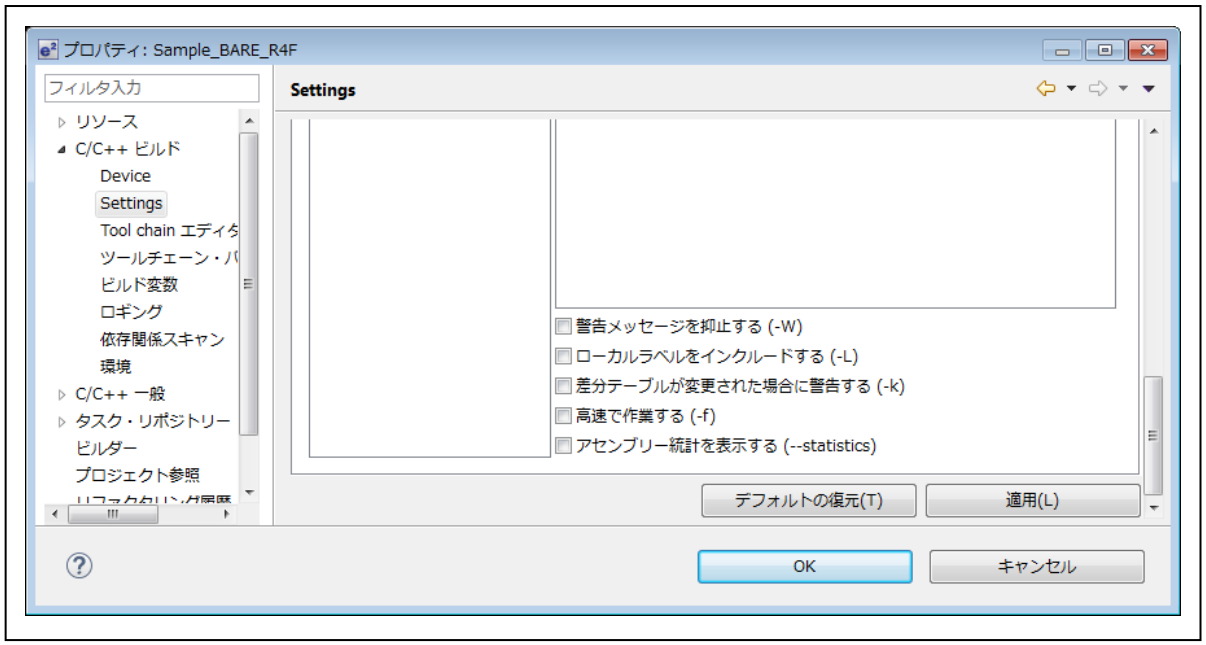
3）リスト（デフォルト）



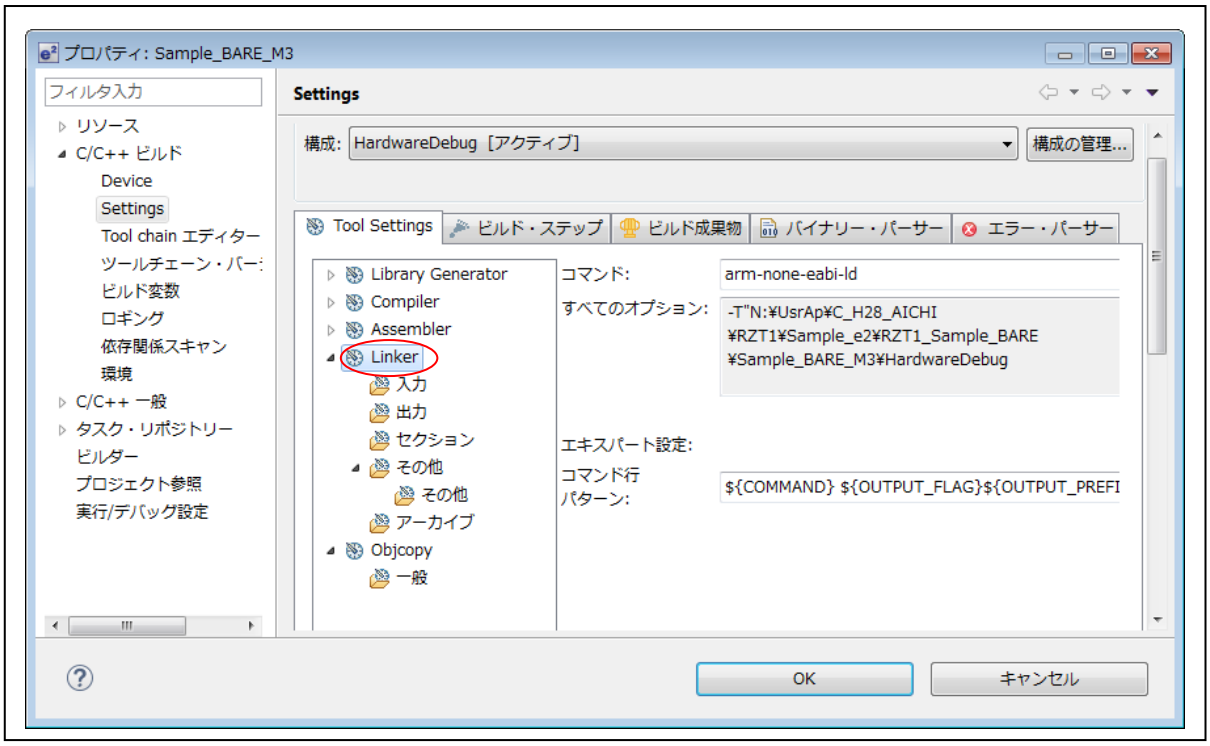
4）その他（1／2）（デフォルト）



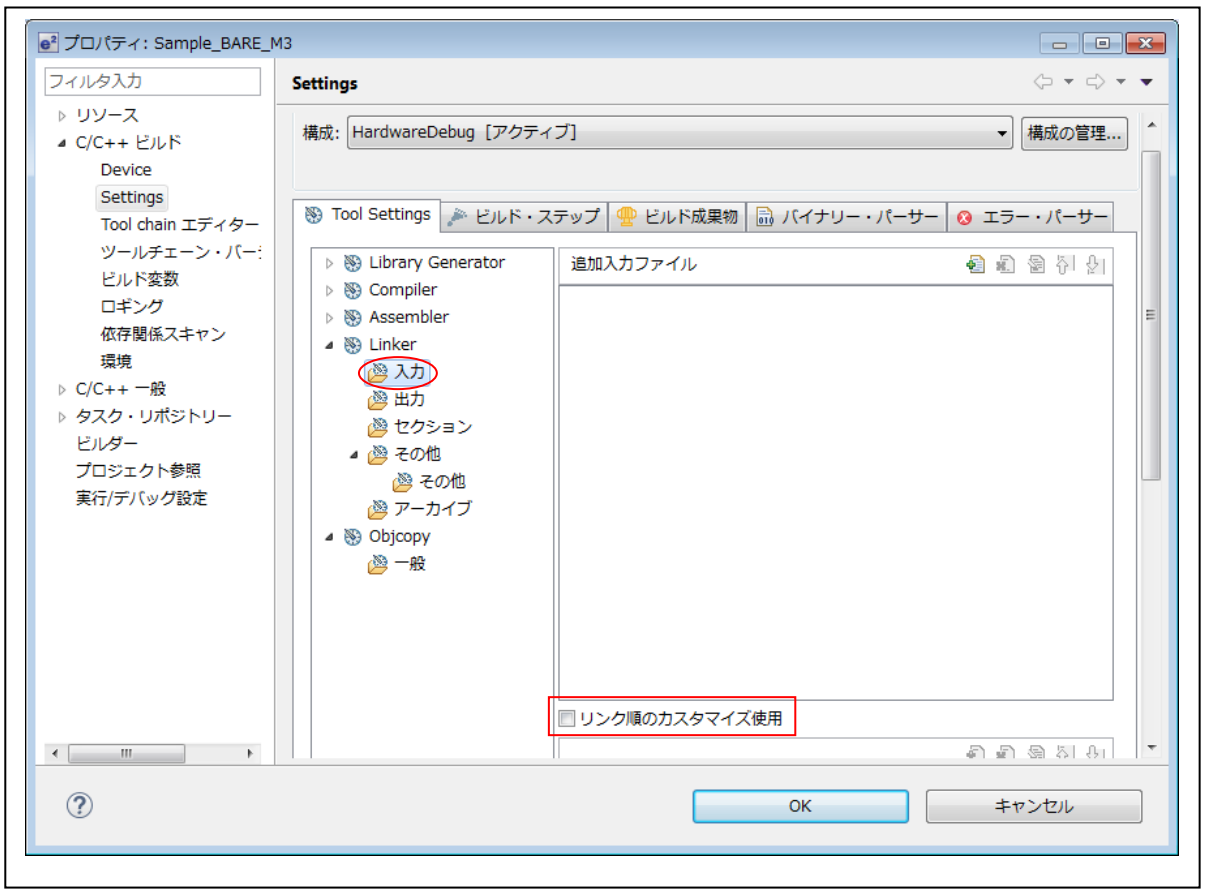
5）その他（2／2）（デフォルト）



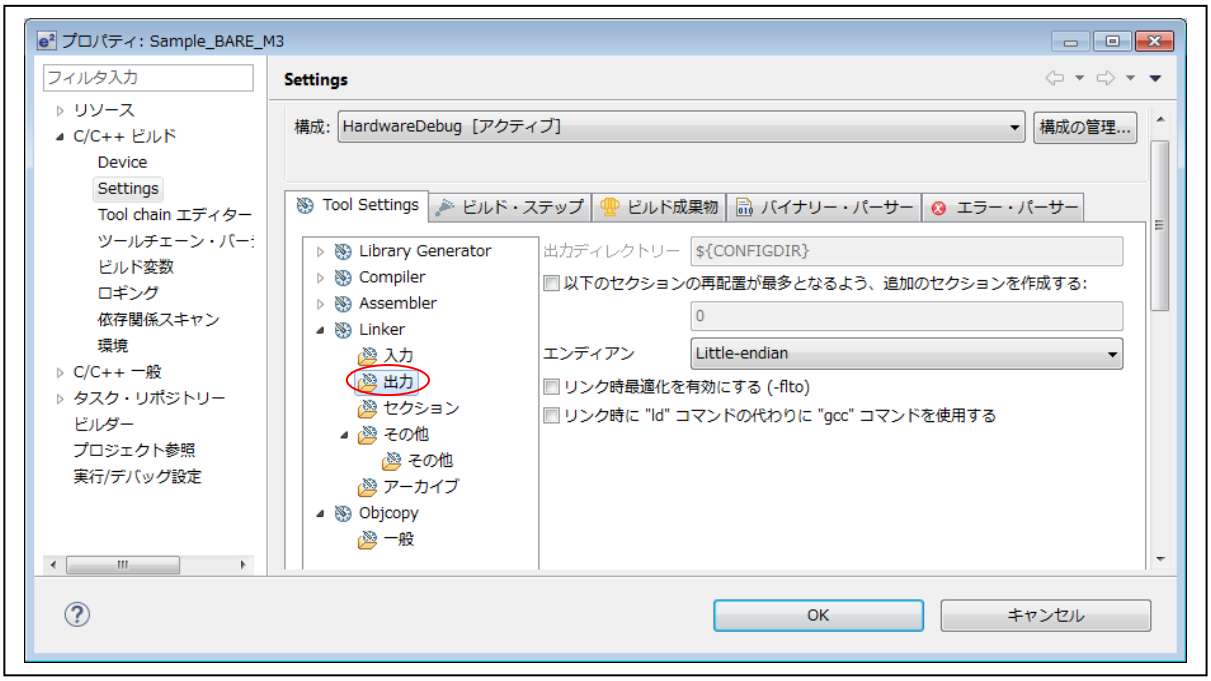
2－2－4－4．Tool　Settings(Linker)



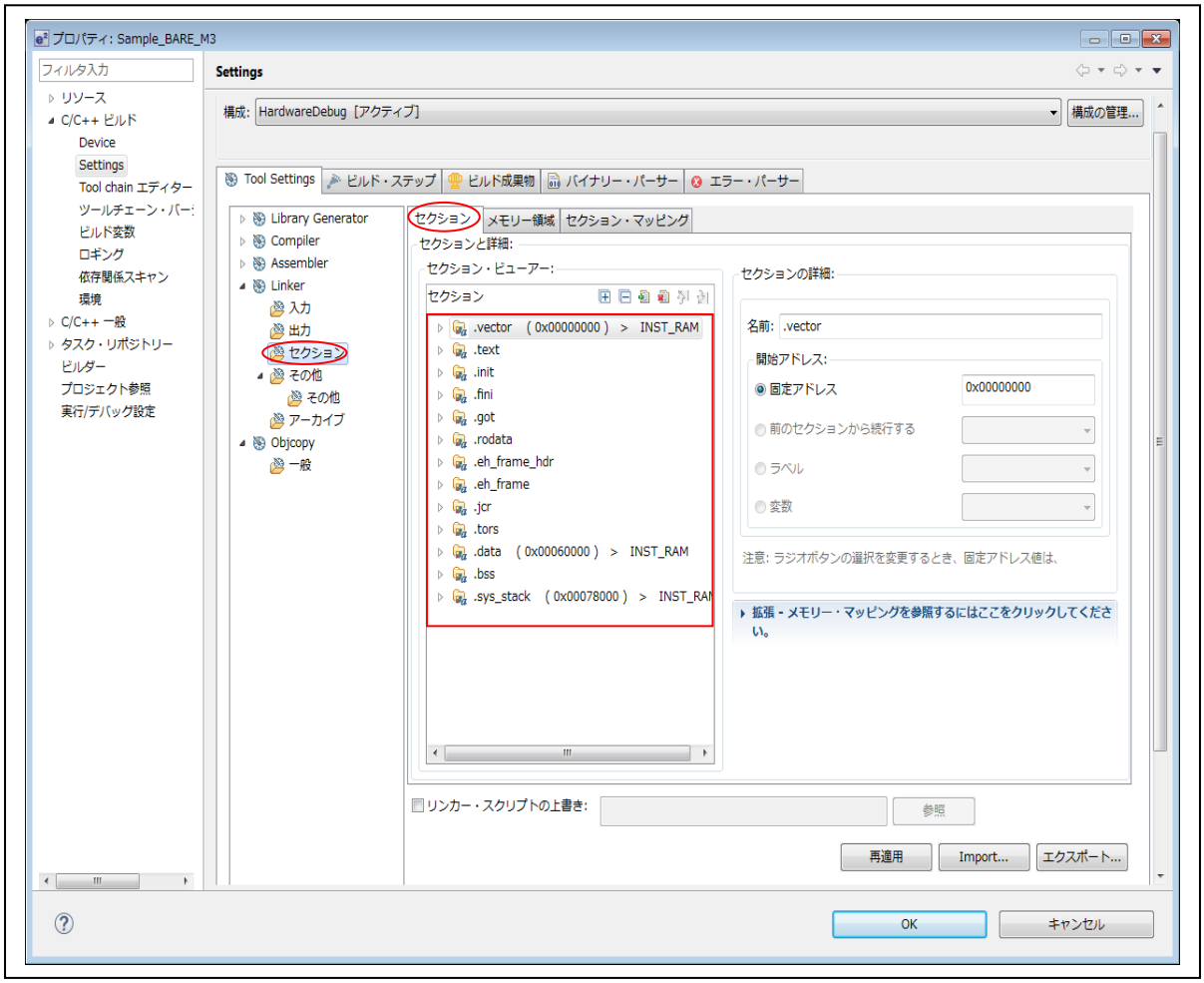
1）入力（デフォルト）



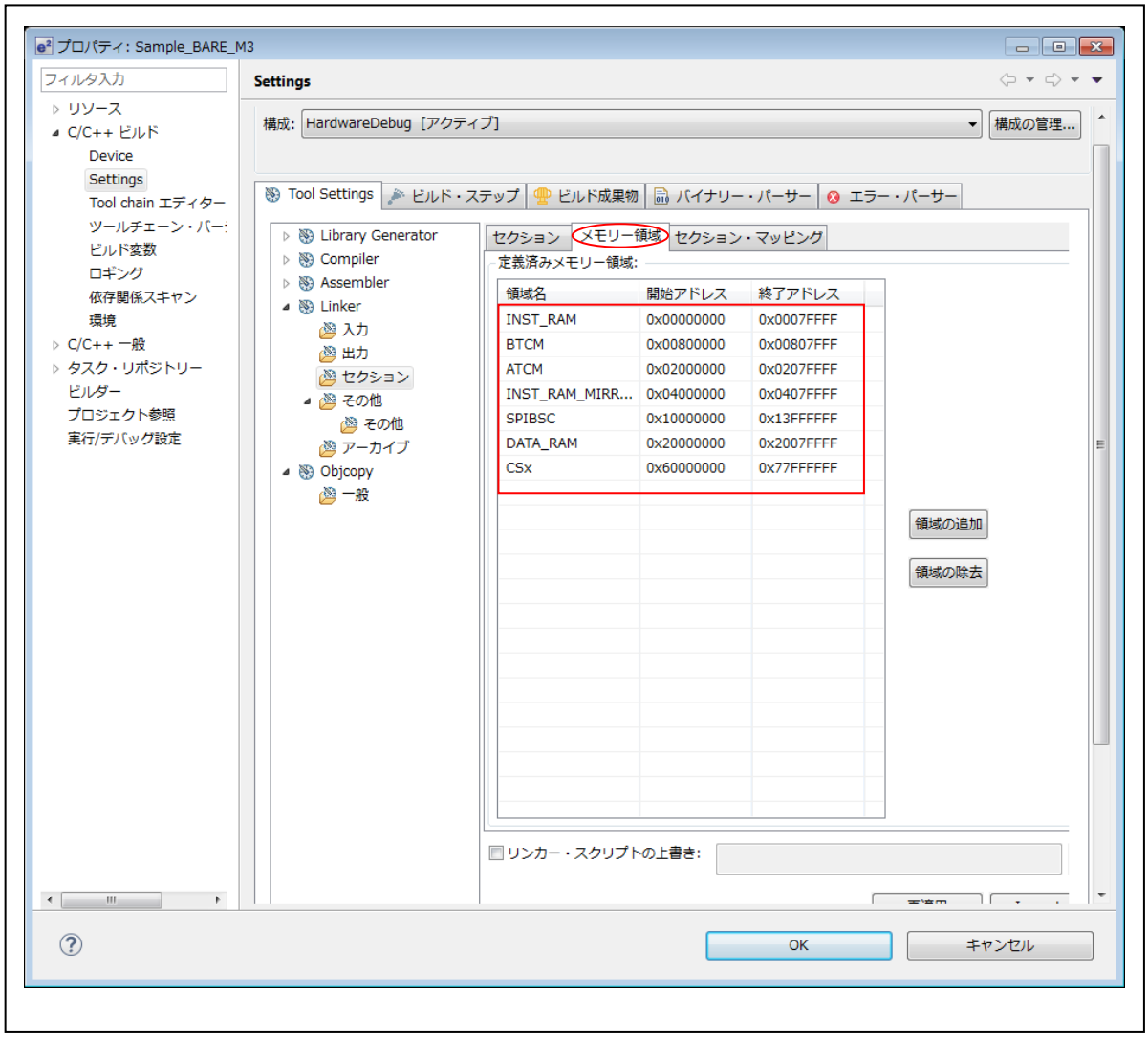
2）出力（デフォルト）



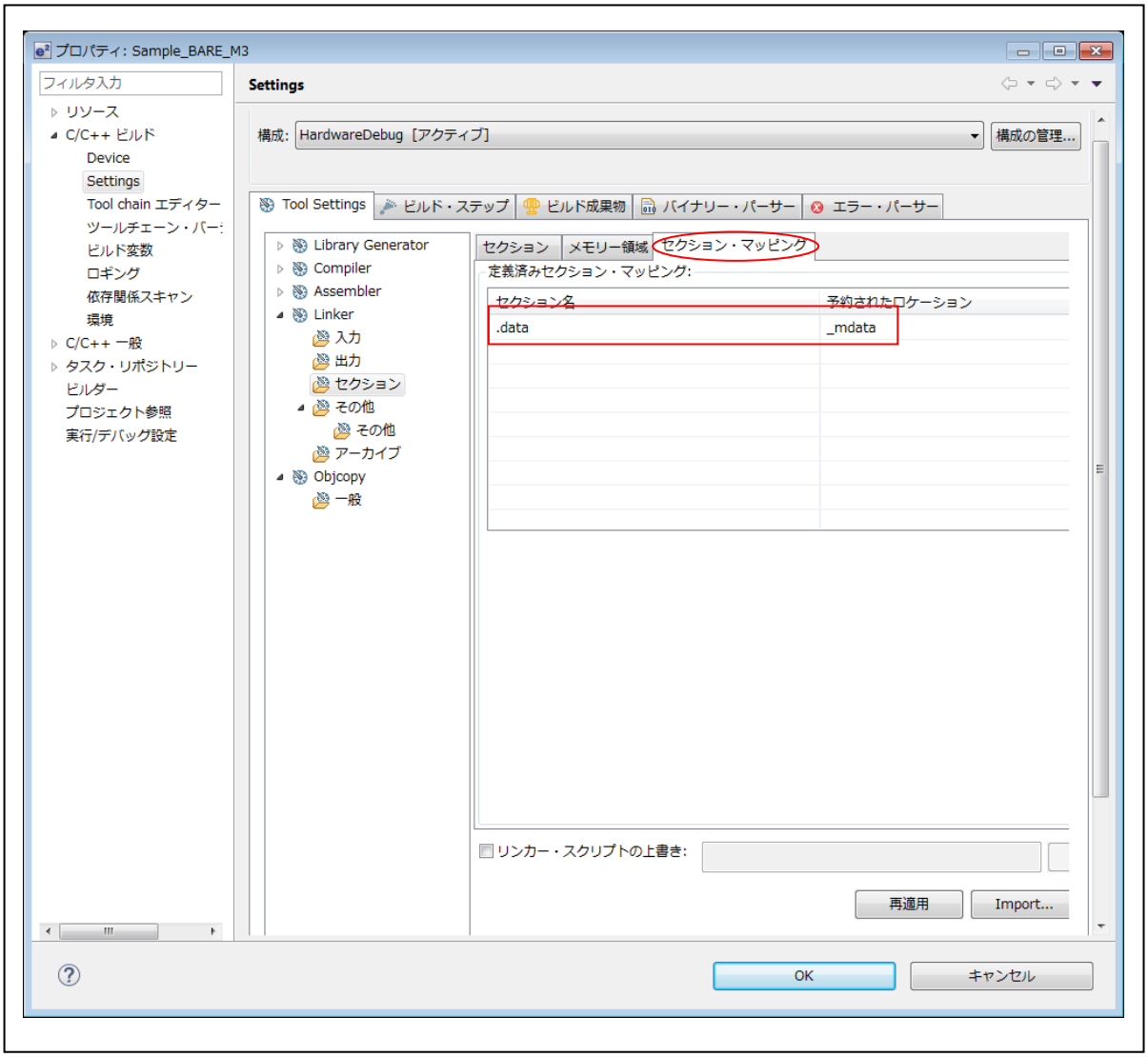
3）セクション



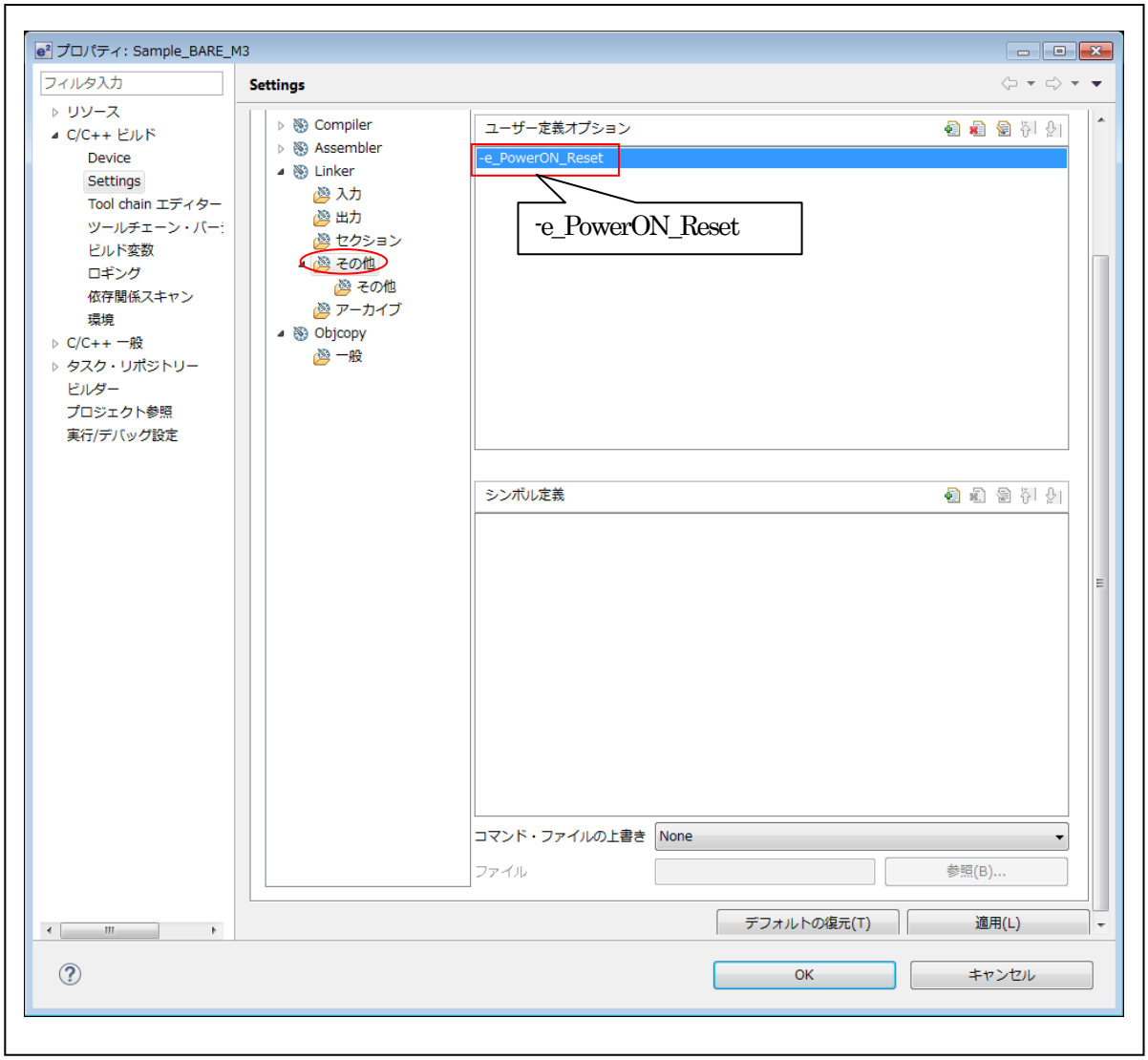
4）セクション（メモリー領域）（デフォルト）



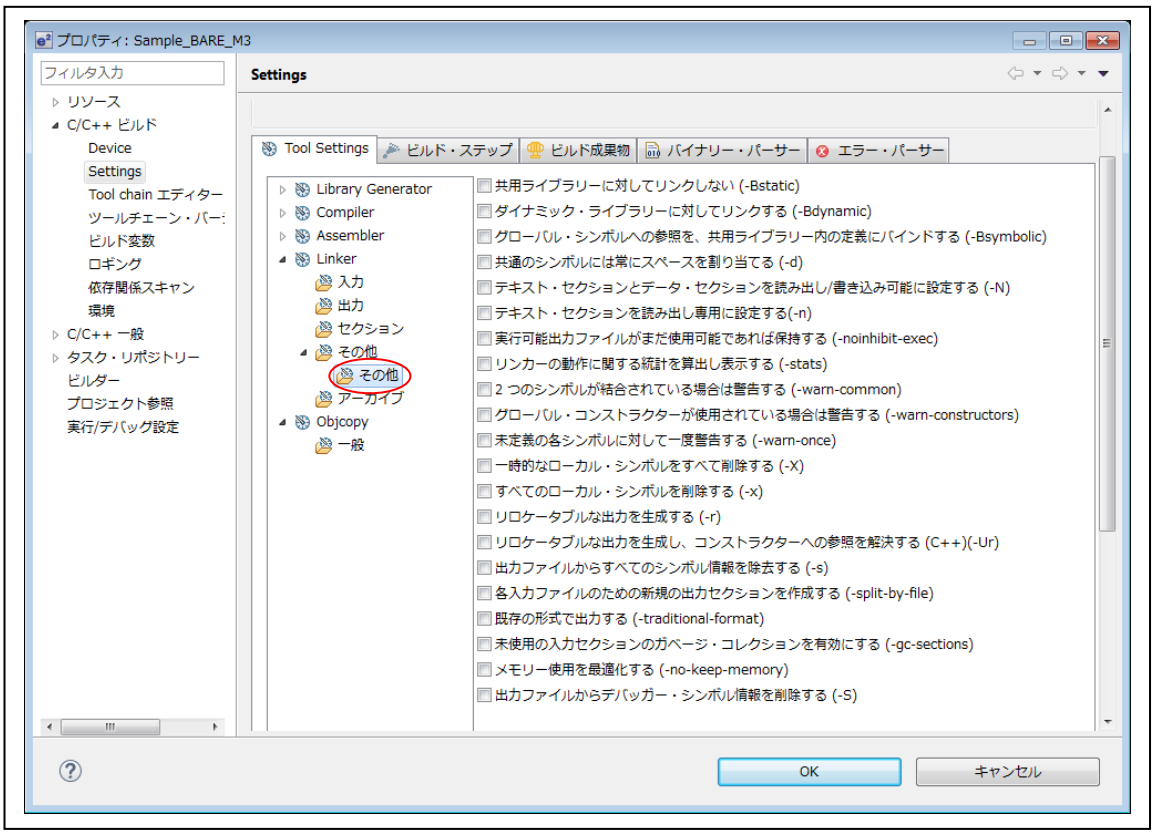
5）セクション（セクション・マッピング）（デフォルト）



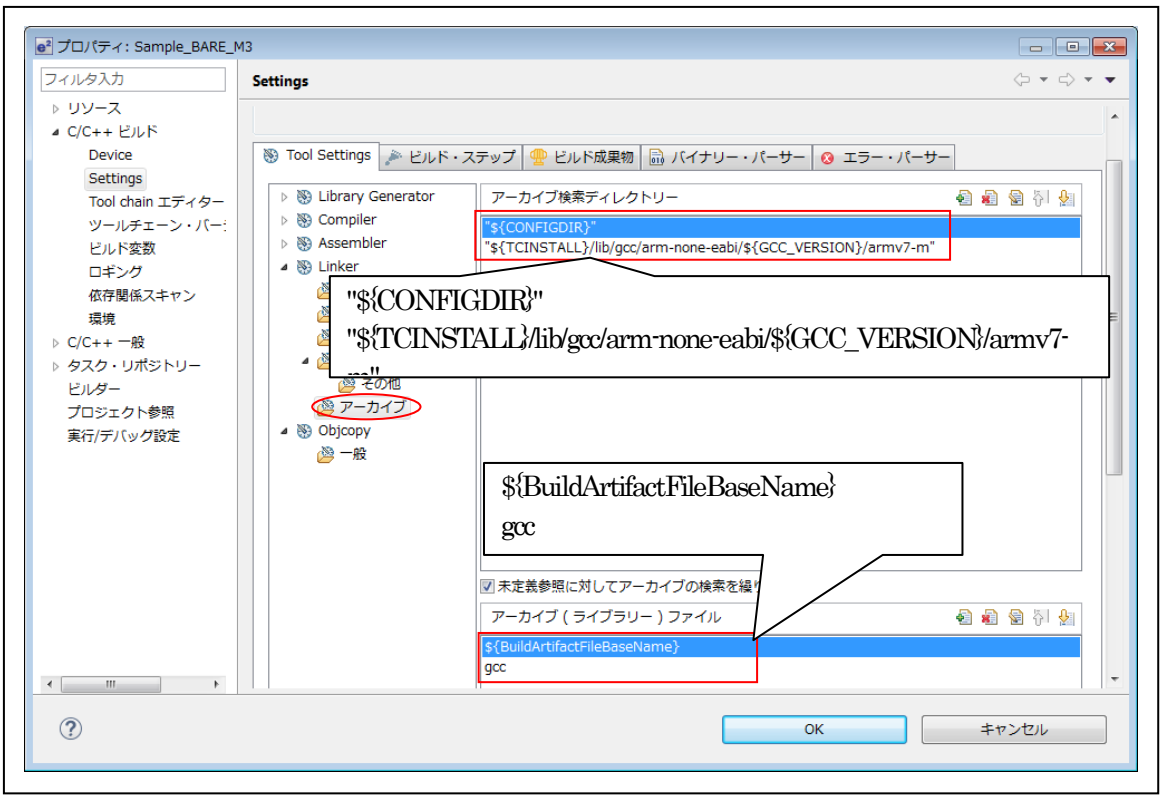
6）その他（デフォルト）


プロパティ: Sample_BARE_M3
フィルタ入力
リソース
C/C++ ビルド
Device
Settings
Tool chain エディター
ツールチェーン・バー
ビルド変数
ロギング
依存関係スキャン
環境
C/C++ 一般
タスク・リポジトリー
ビルダー
プロジェクト参照
実行/デバッグ設定
Settings
Compiler
Assembler
Linker
入力
出力
セクション
その他
その他
アーカイブ
Objcopy
一般
ユーザー定義オプション
-e_PowerON_Reset
-e_PowerON_Reset
シンボル定義
コマンド・ファイルの上書き
None
ファイル
参照(B)...
デフォルトの復元(T)
適用(L)
OK
キャンセル

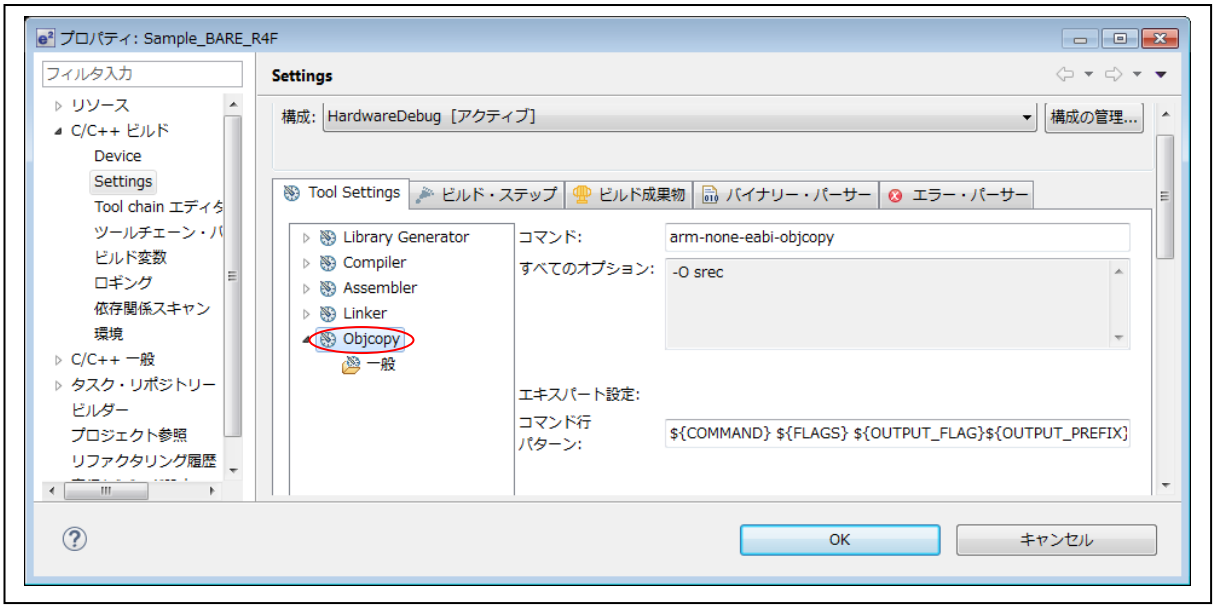
6）その他ーその他（デフォルト）



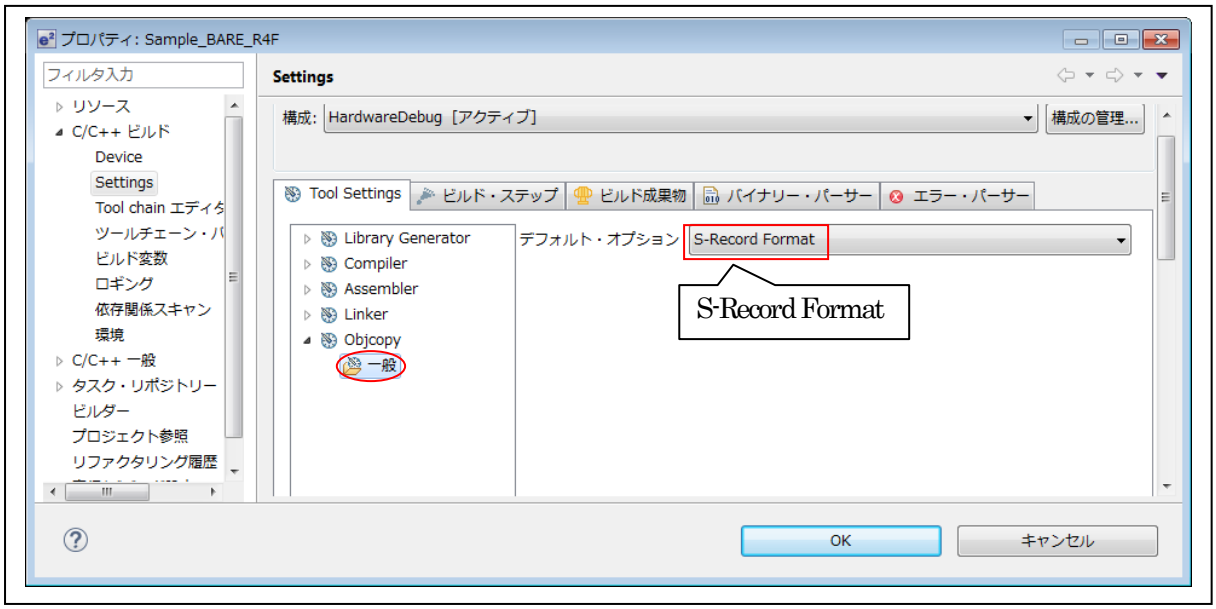
7）アーカイブ



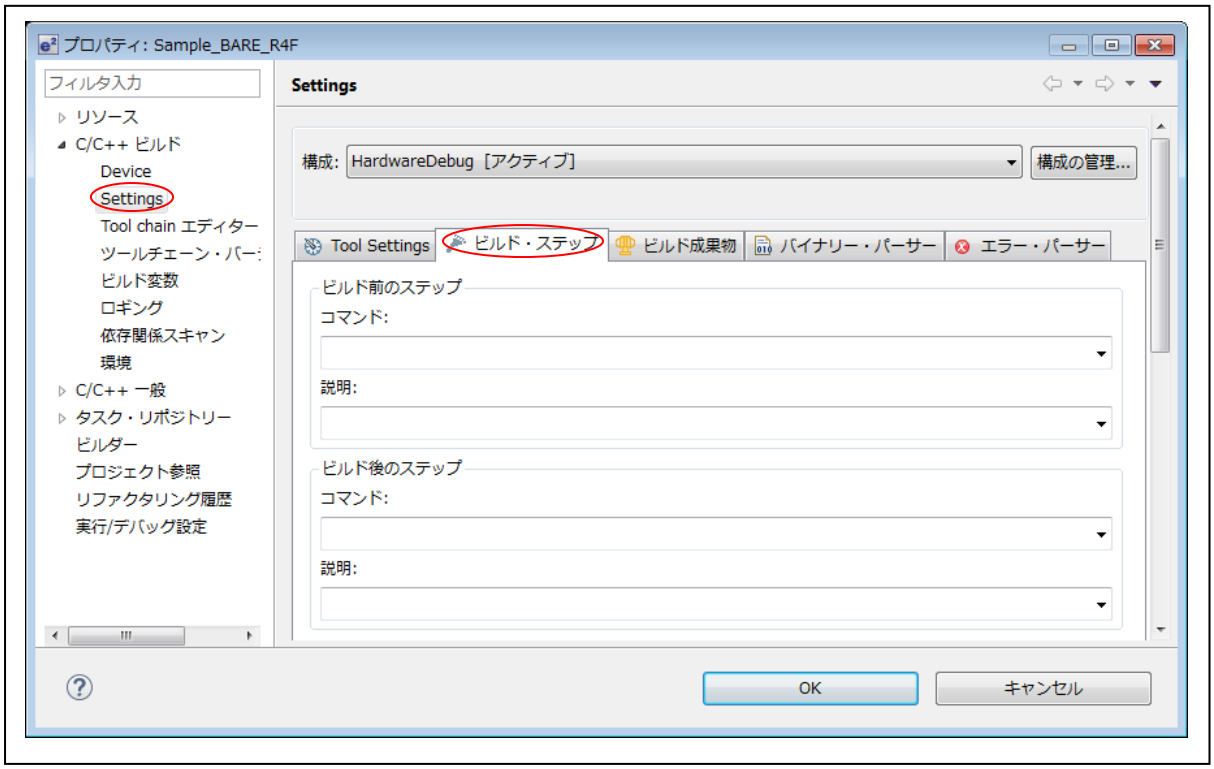
2－2－4－5．Tool　Settings(Objcopy)



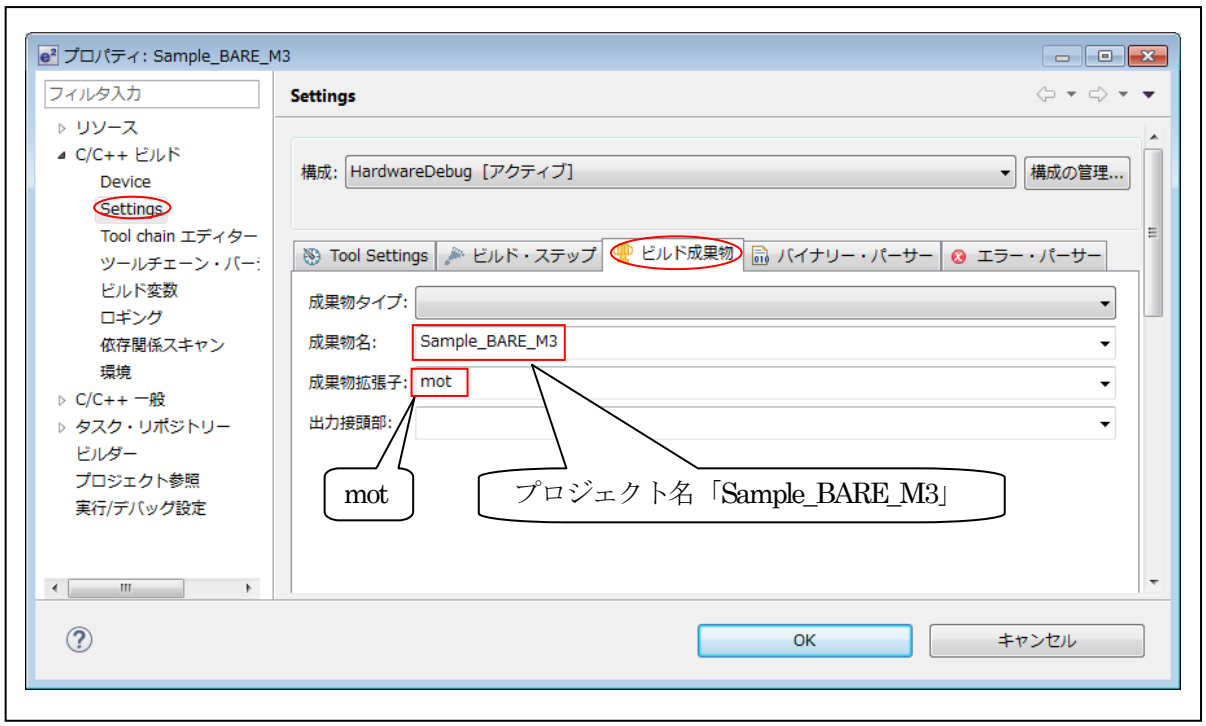
1）一般



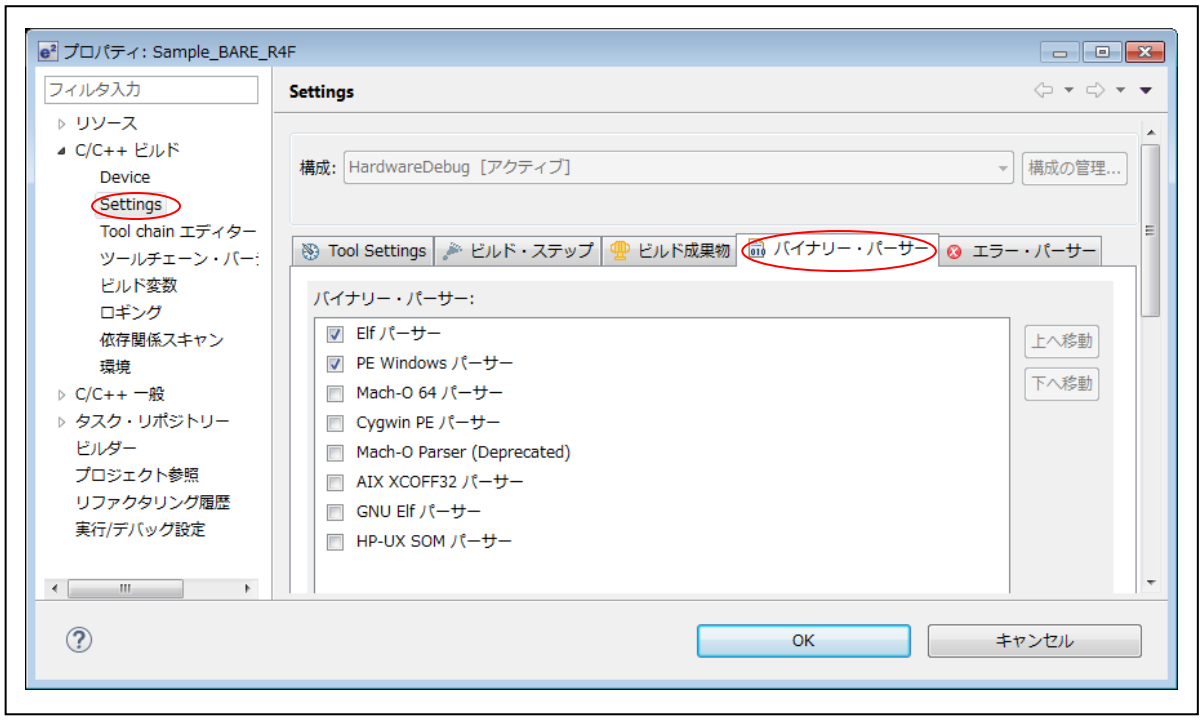
2－2－4－6．ビルド・ステップ（デフォルト）



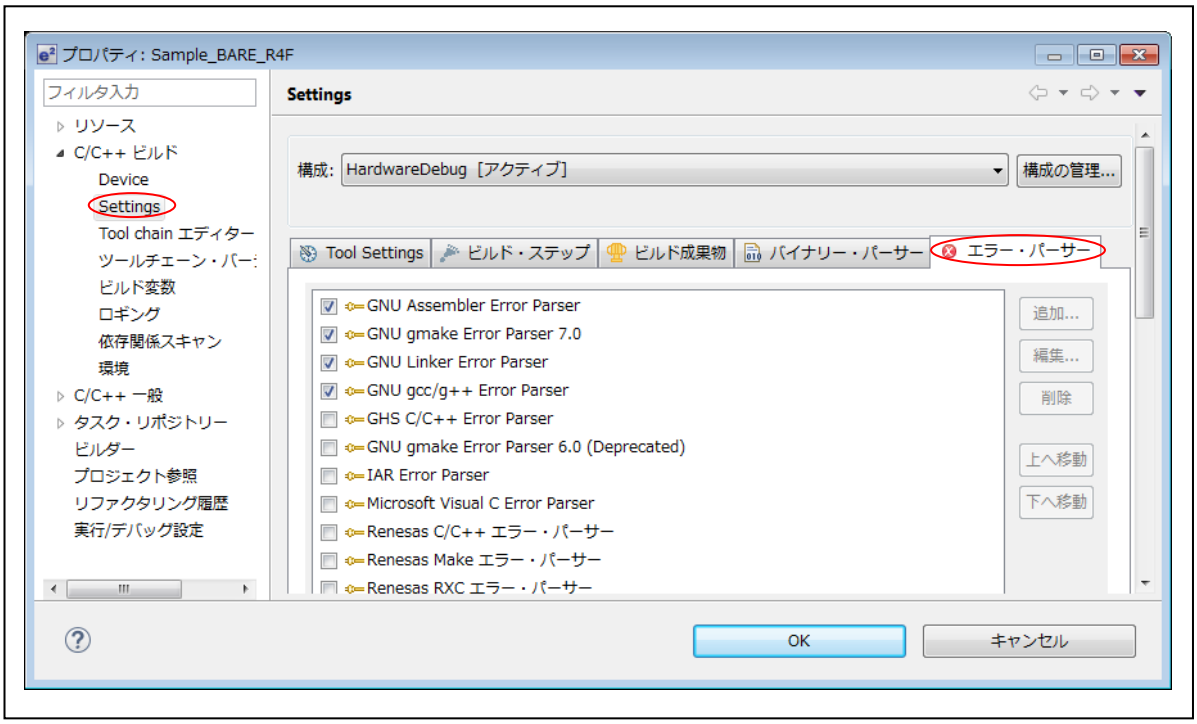
2－2－4－7．ビルド成果物



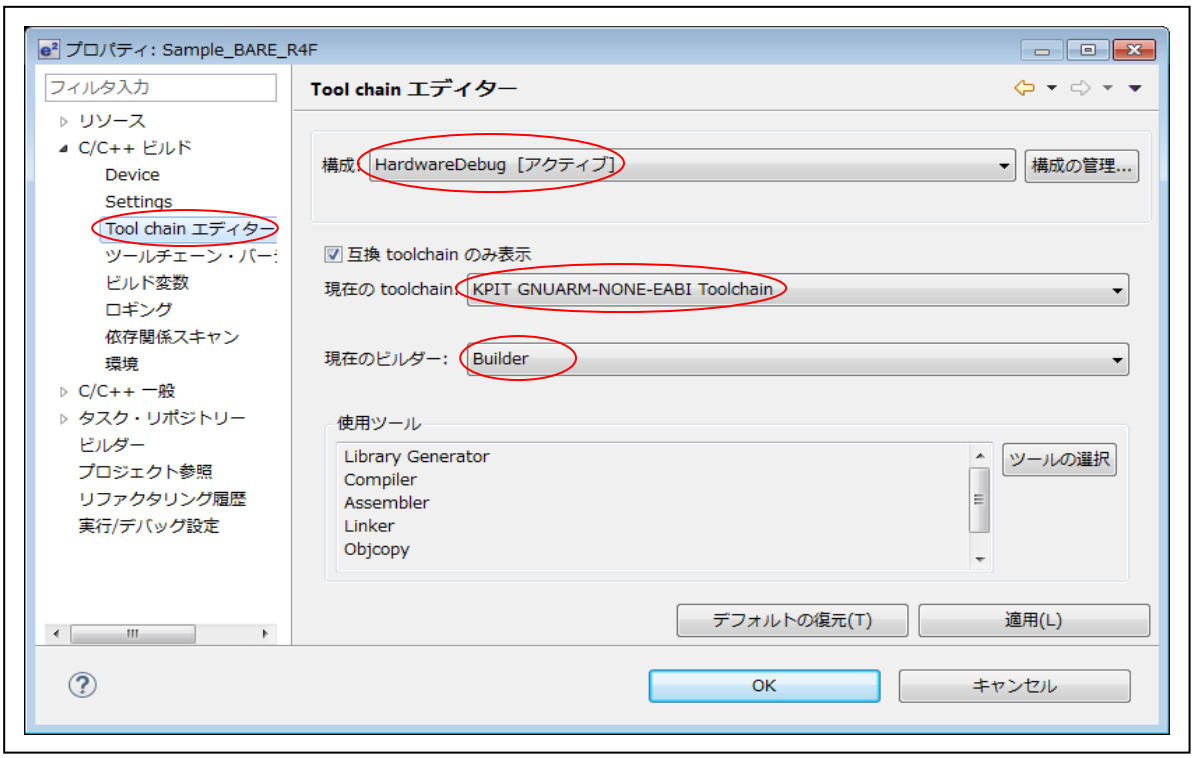
２－２－４－８．バイナリー・パーサー（デフォルト）



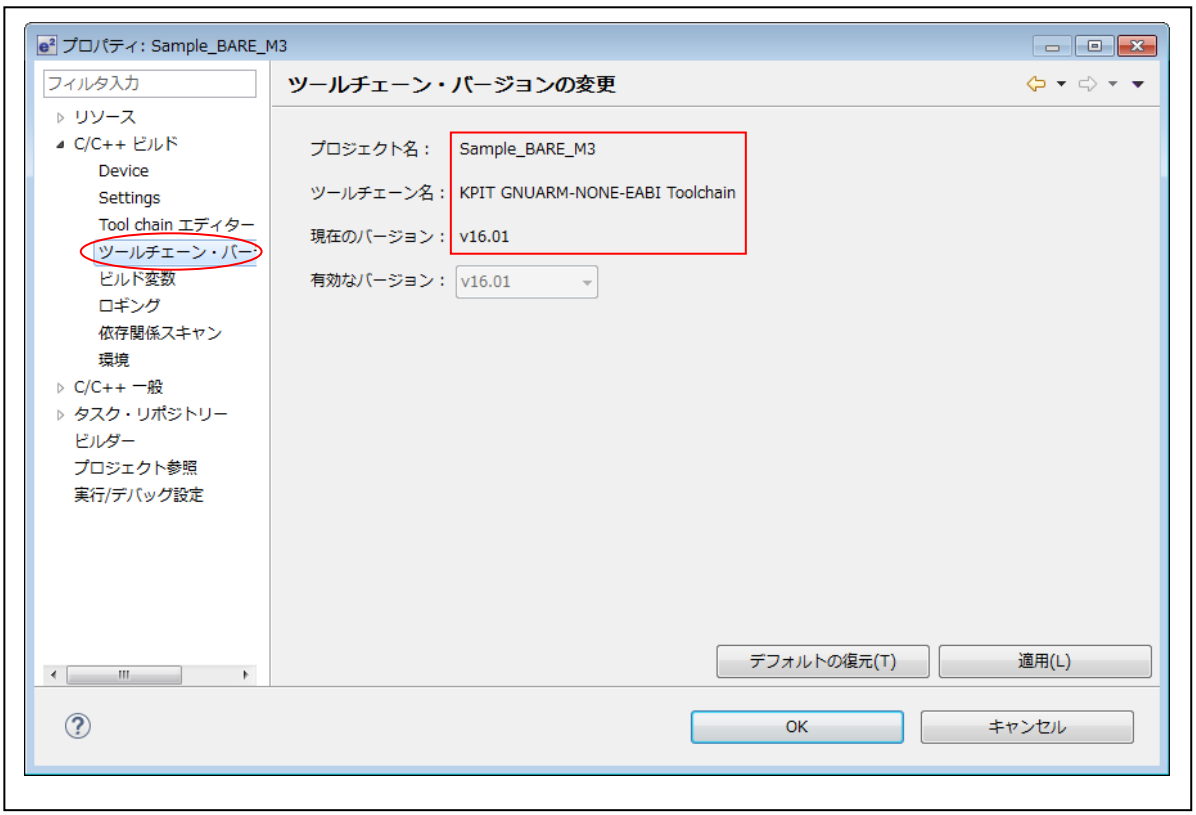
２－２－４－８．エラー・パーサー（デフォルト）



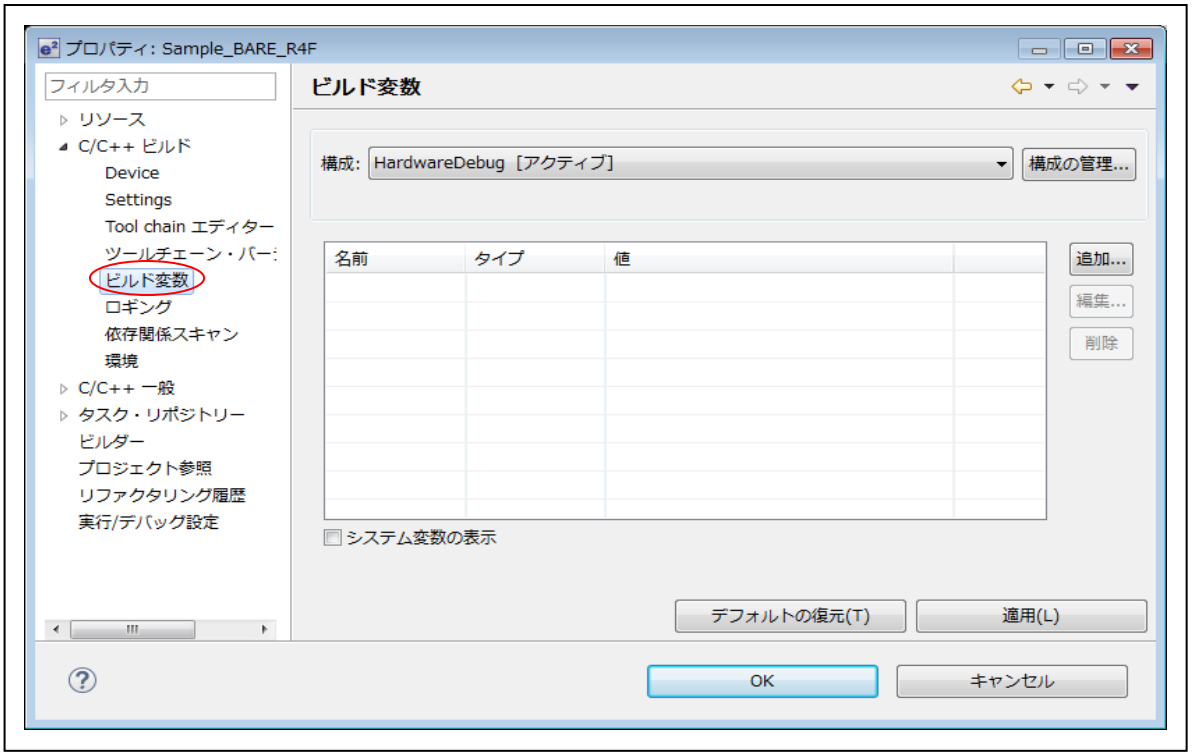
２－２－５．C/C++ビルド（Tool Chain エディター）（デフォルト）



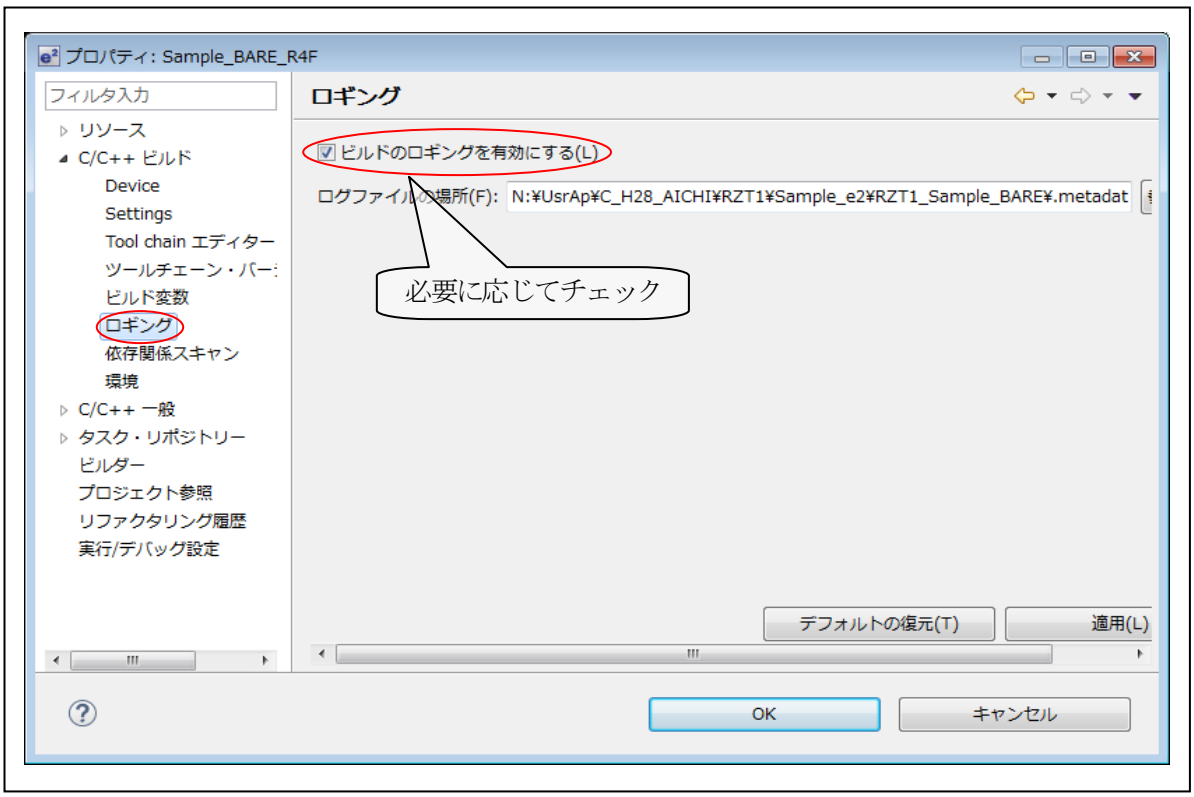
２－２－６．C/C++ビルド（ツールチェーン・バージョン）（デフォルト）



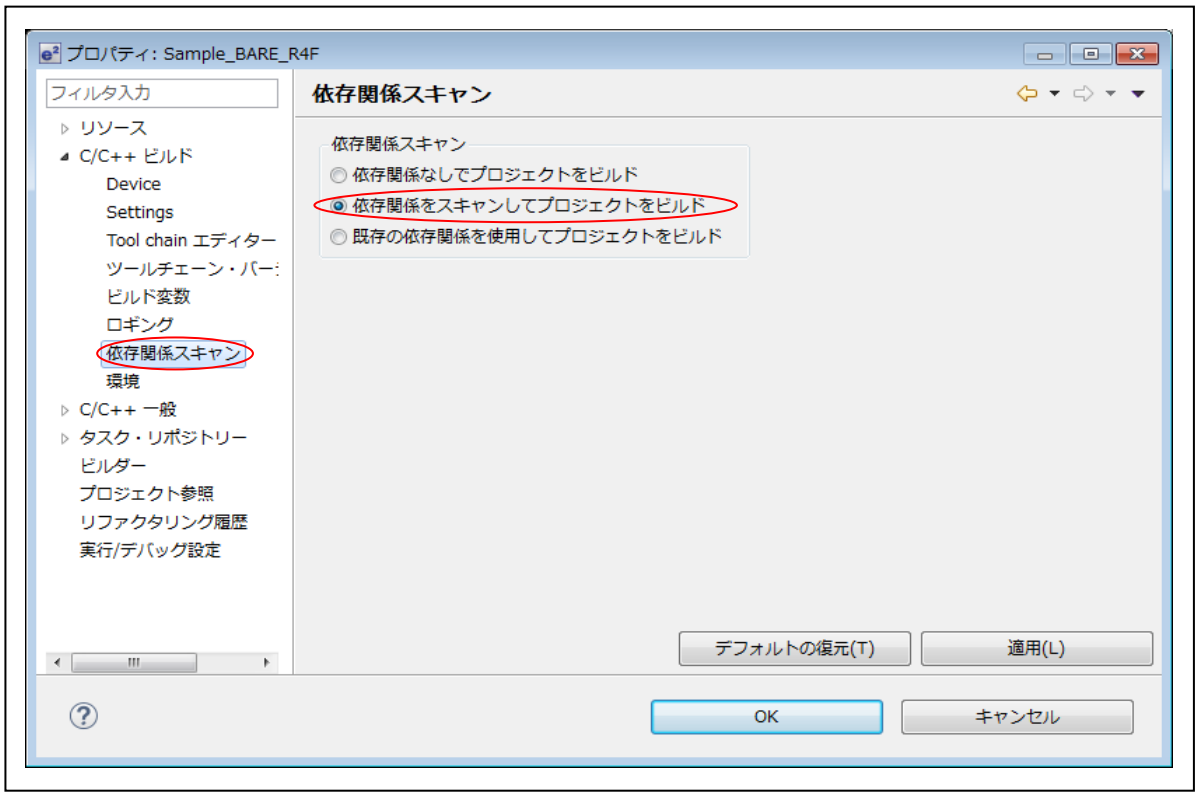
2－2－7．C/C++ビルド（ビルド変数）（デフォルト）



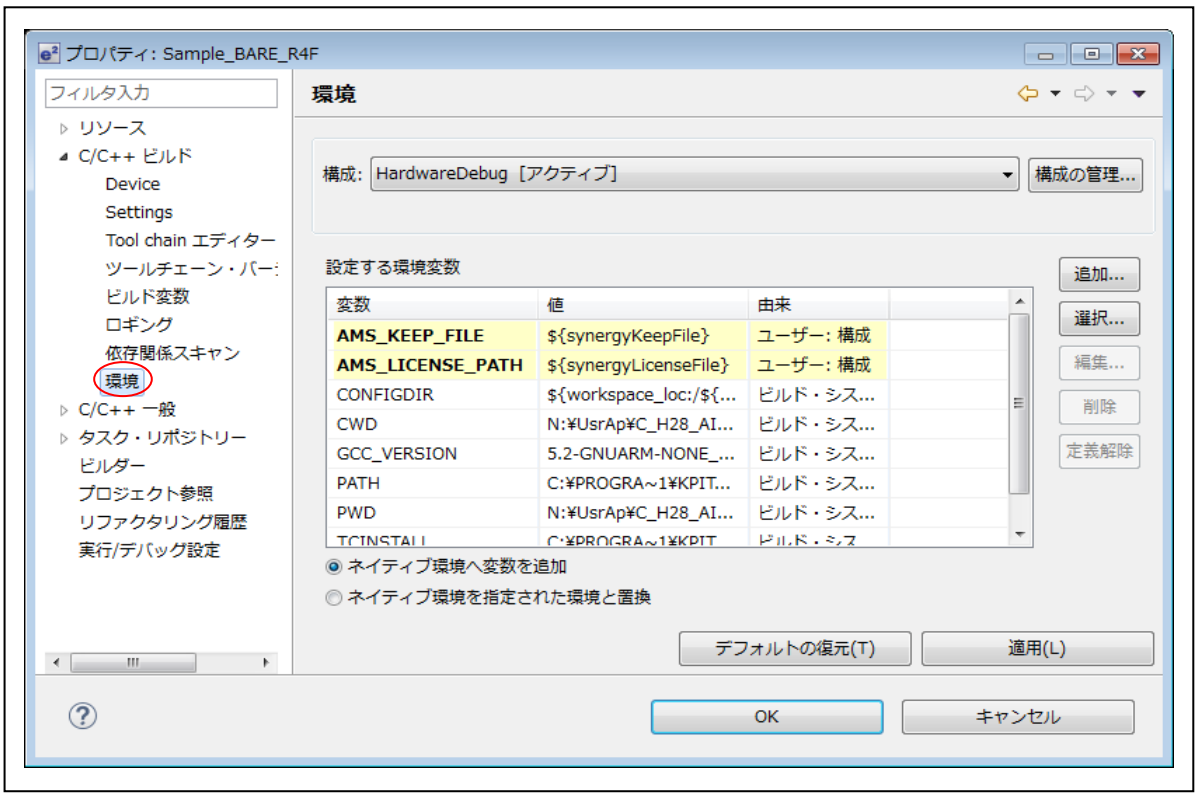
2－2－8．C/C++ビルド（ロギング）（デフォルト）



２－２－９．C/C++ビルド（依存関係スキャン）（デフォルト）



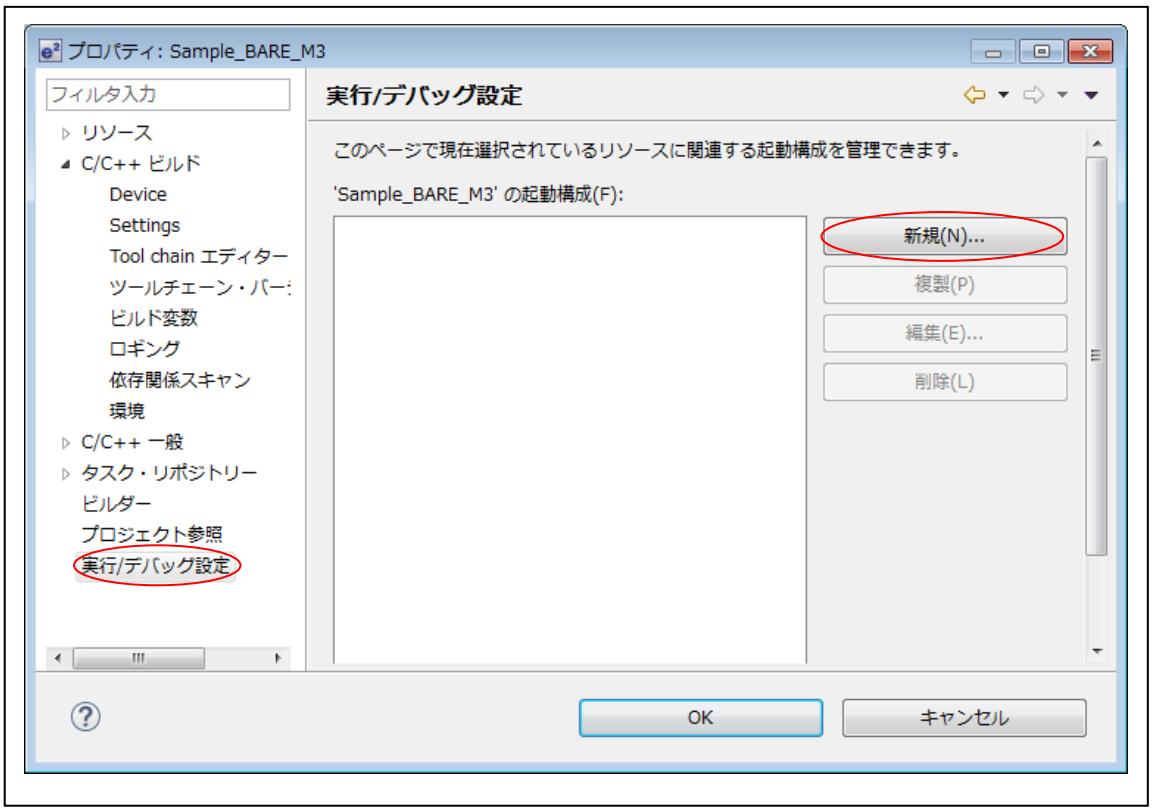
２－２－１０．C/C++ビルド（環境）（デフォルト）



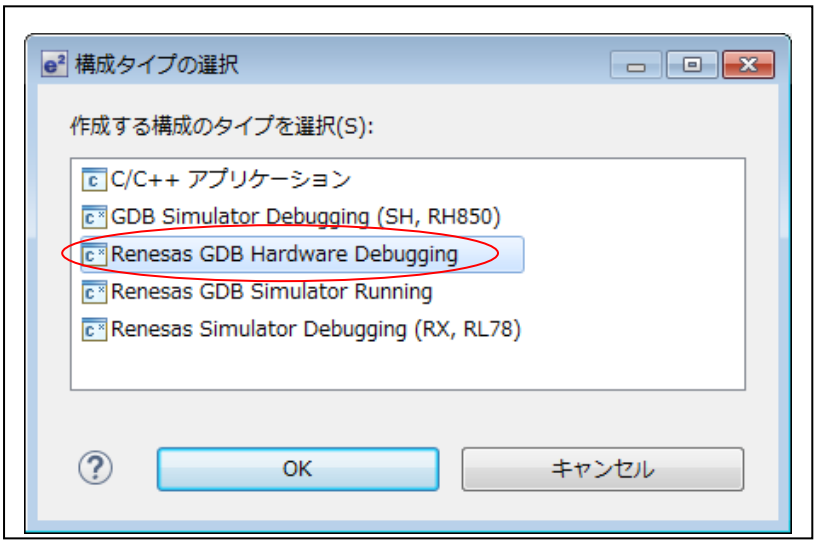
２－２－１１．【説明省略】C/C++　一般（デフォルト）
２－２－１２．【説明省略】タスク・リポジトリー（デフォルト）
２－２－１３．【説明省略】ビルダー（デフォルト）
２－２－１４．【説明省略】プロジェクト参照（デフォルト）
２－２－１５．【説明省略】リファクタリング履歴（デフォルト）

２－２－１６．実行／デバッグ設定

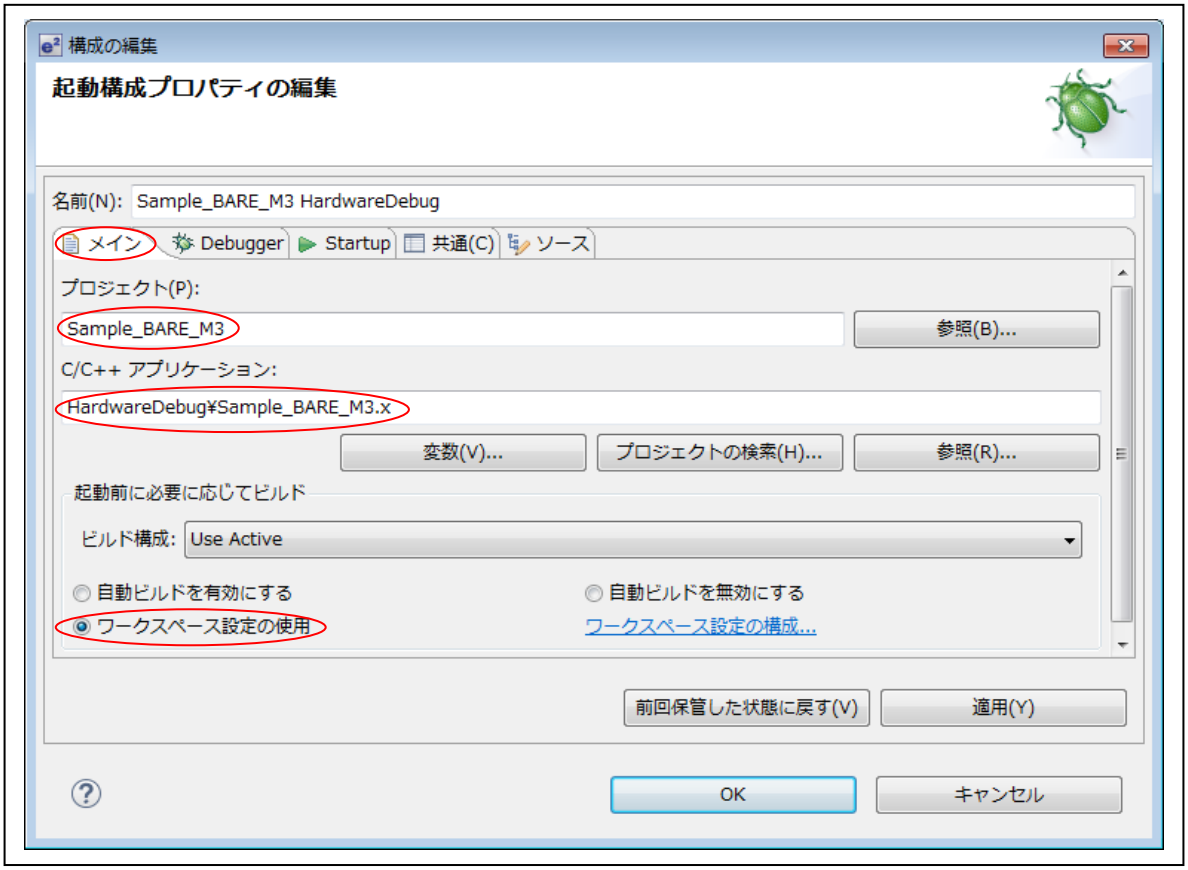
１）新規登録



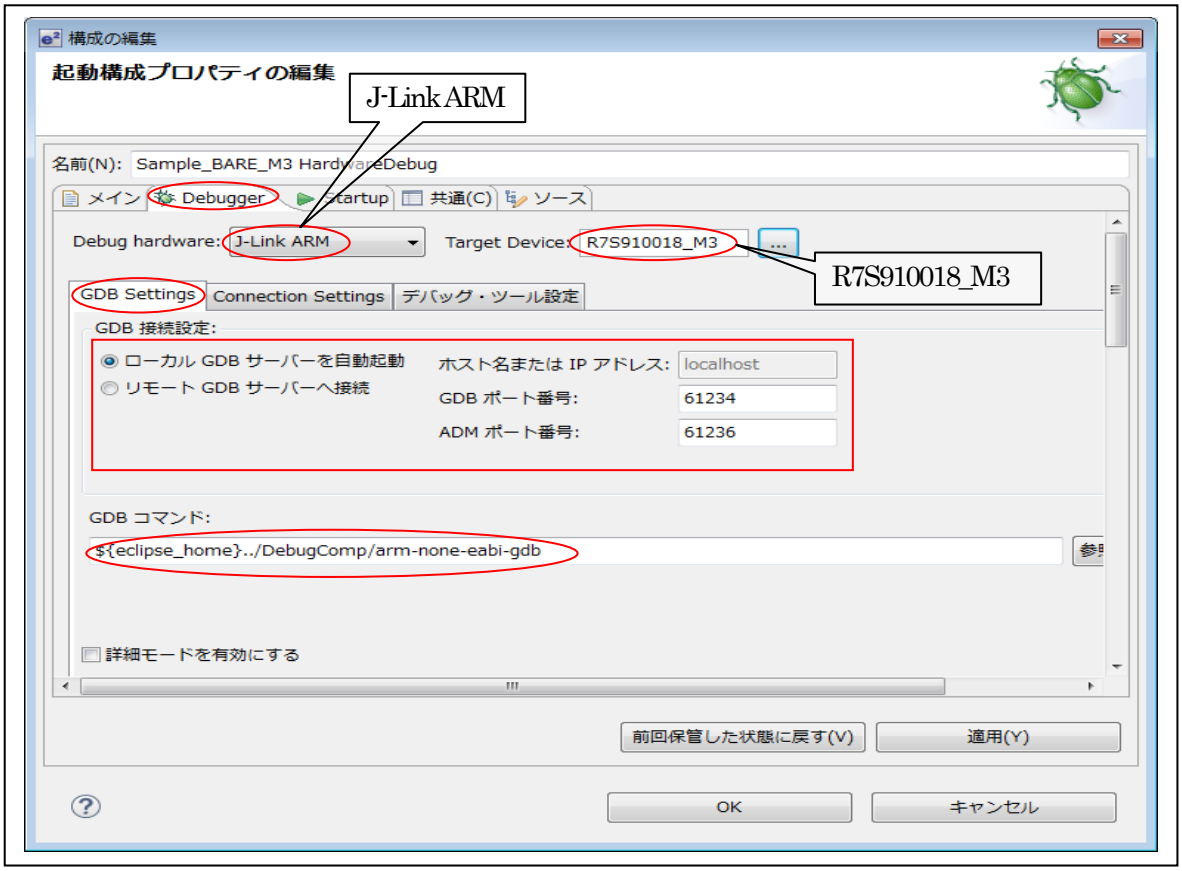
２）構成タイプの選択



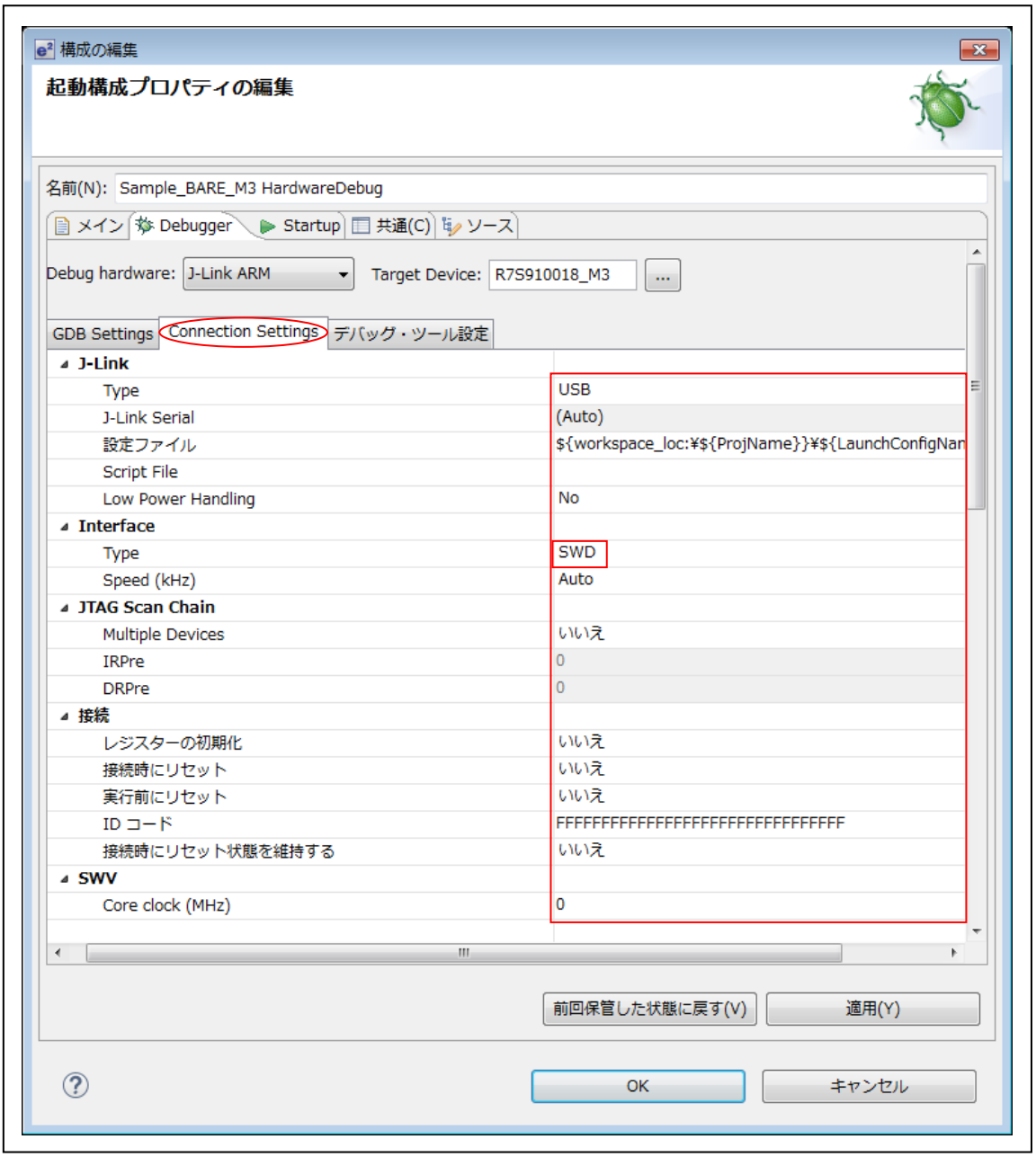
3）起動構成プロパティの編集(メイン)（デフォルト）



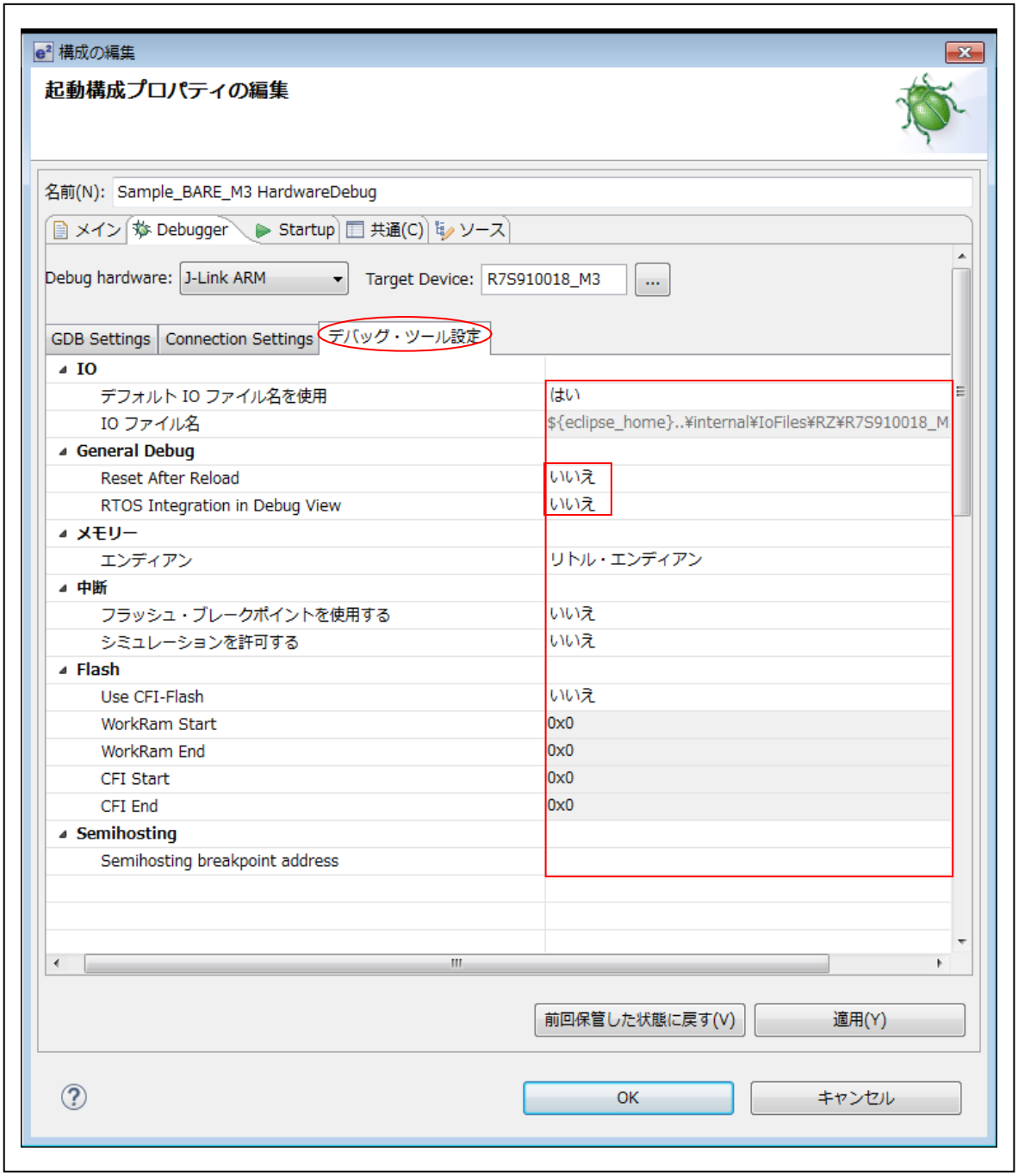
4）起動構成プロパティの編集(Debugger[GDB Settings])



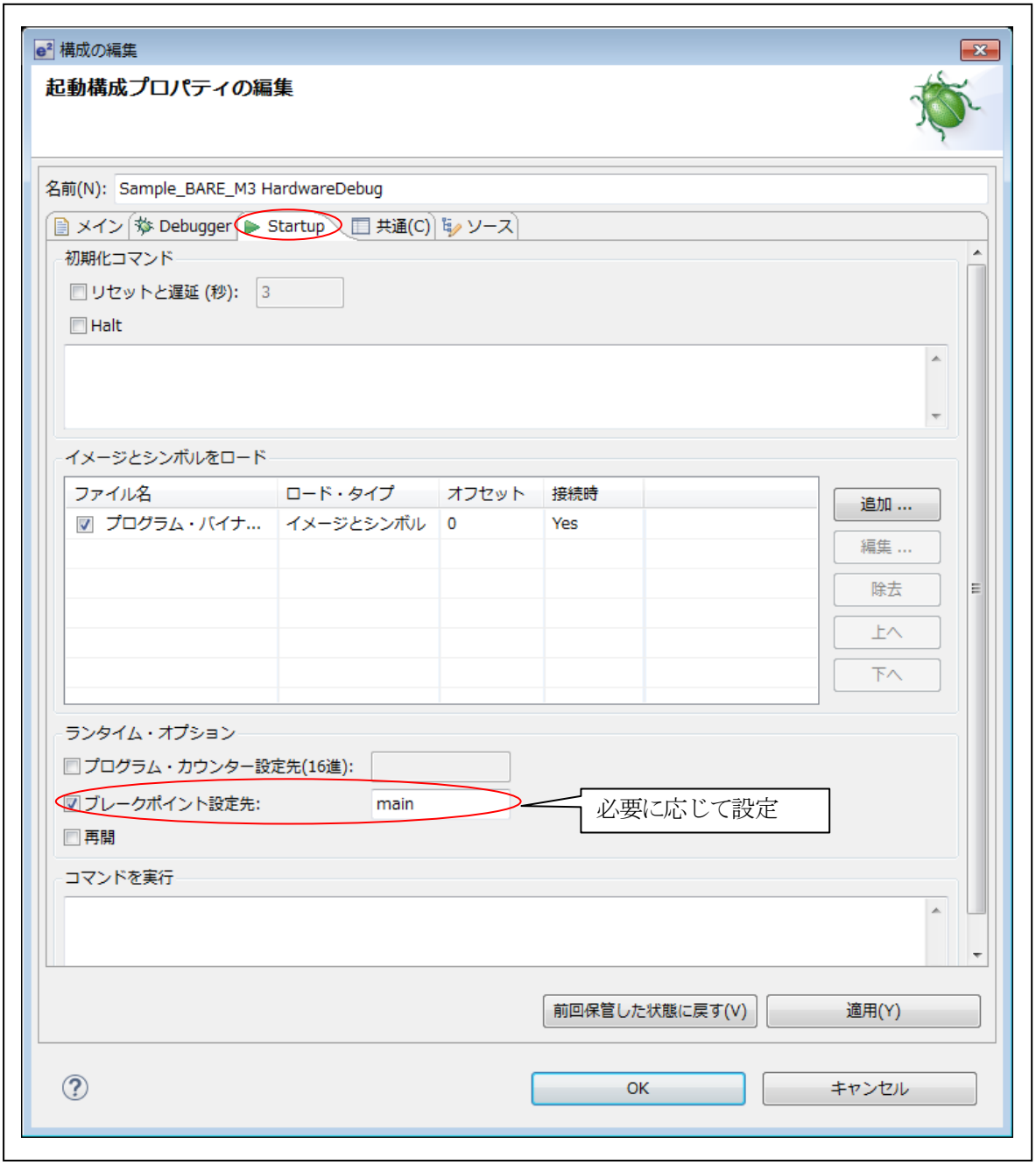
5）起動構成プロパティの編集(Debugger[Connection Settings])



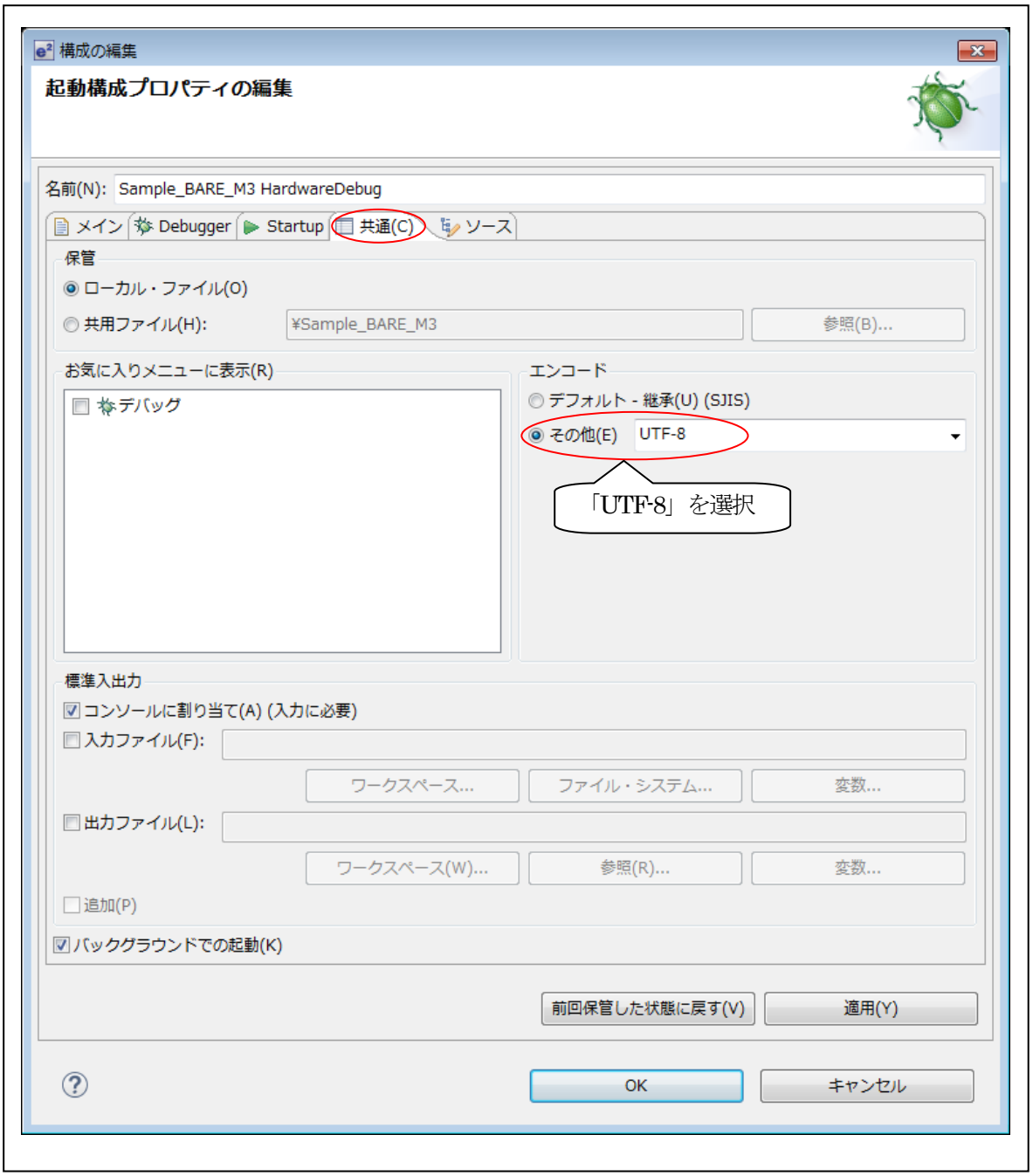
6）起動構成プロパティの編集(Debugger[デバッグ・ツール設定])



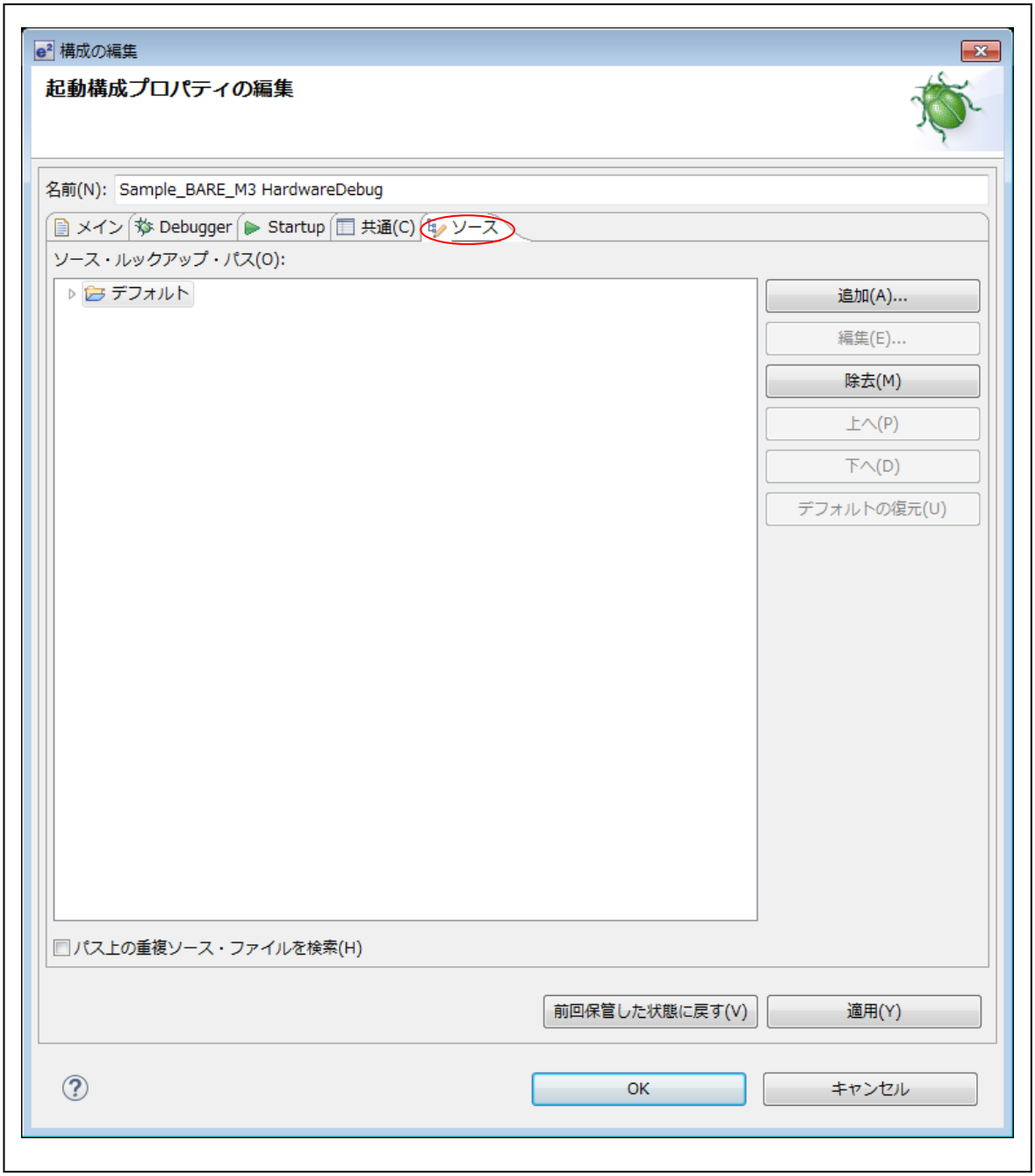
7）起動構成プロパティの編集(Startup)



8）起動構成プロパティの編集(共通)



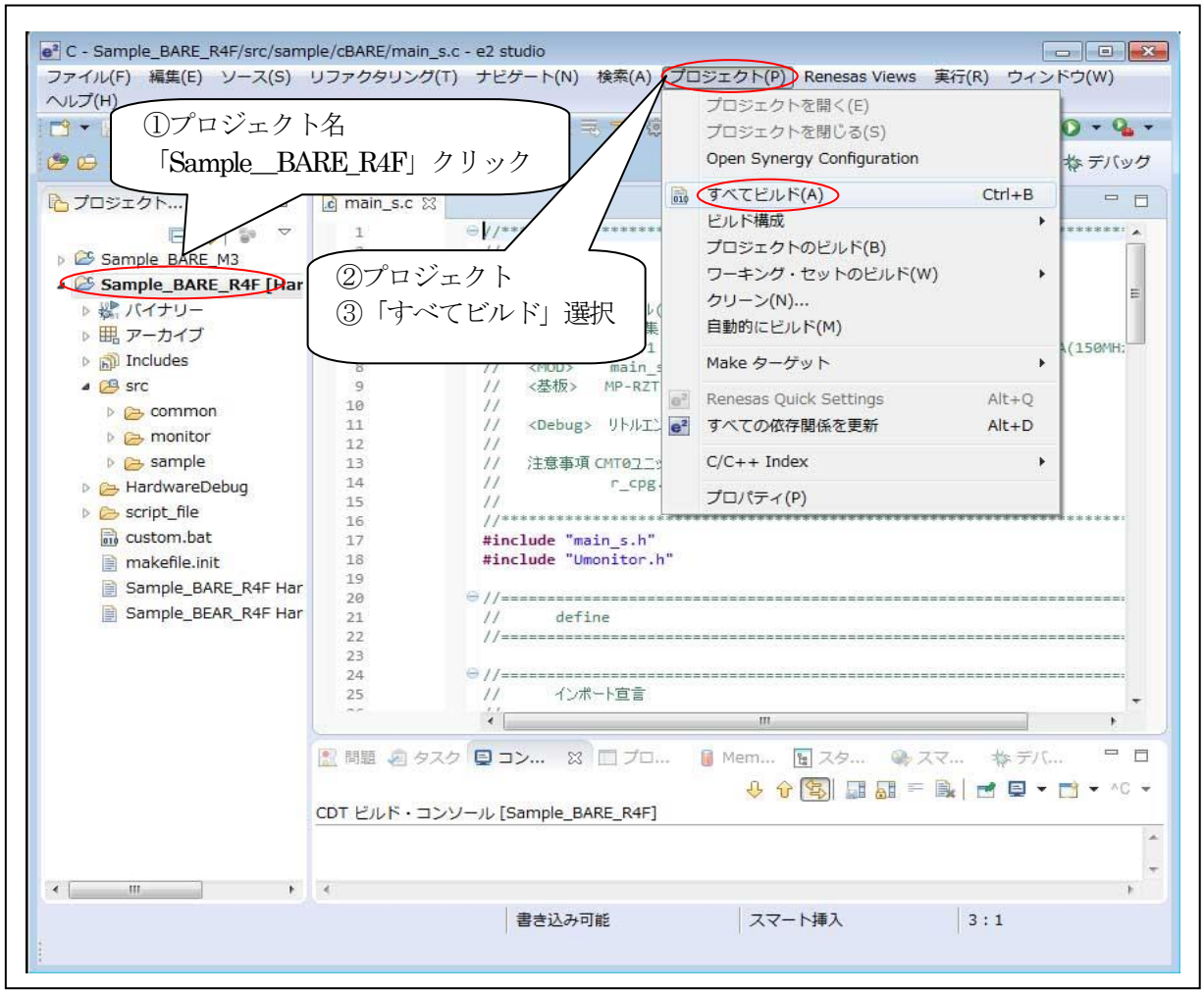
9）起動構成プロパティの編集(ソース)（デフォルト）



3．サンプルプロジェクト「Sample_BARE」をビルドする。

3－1．コア【R4F】側サンプル「Sample_BARE_R4F」をビルドする。

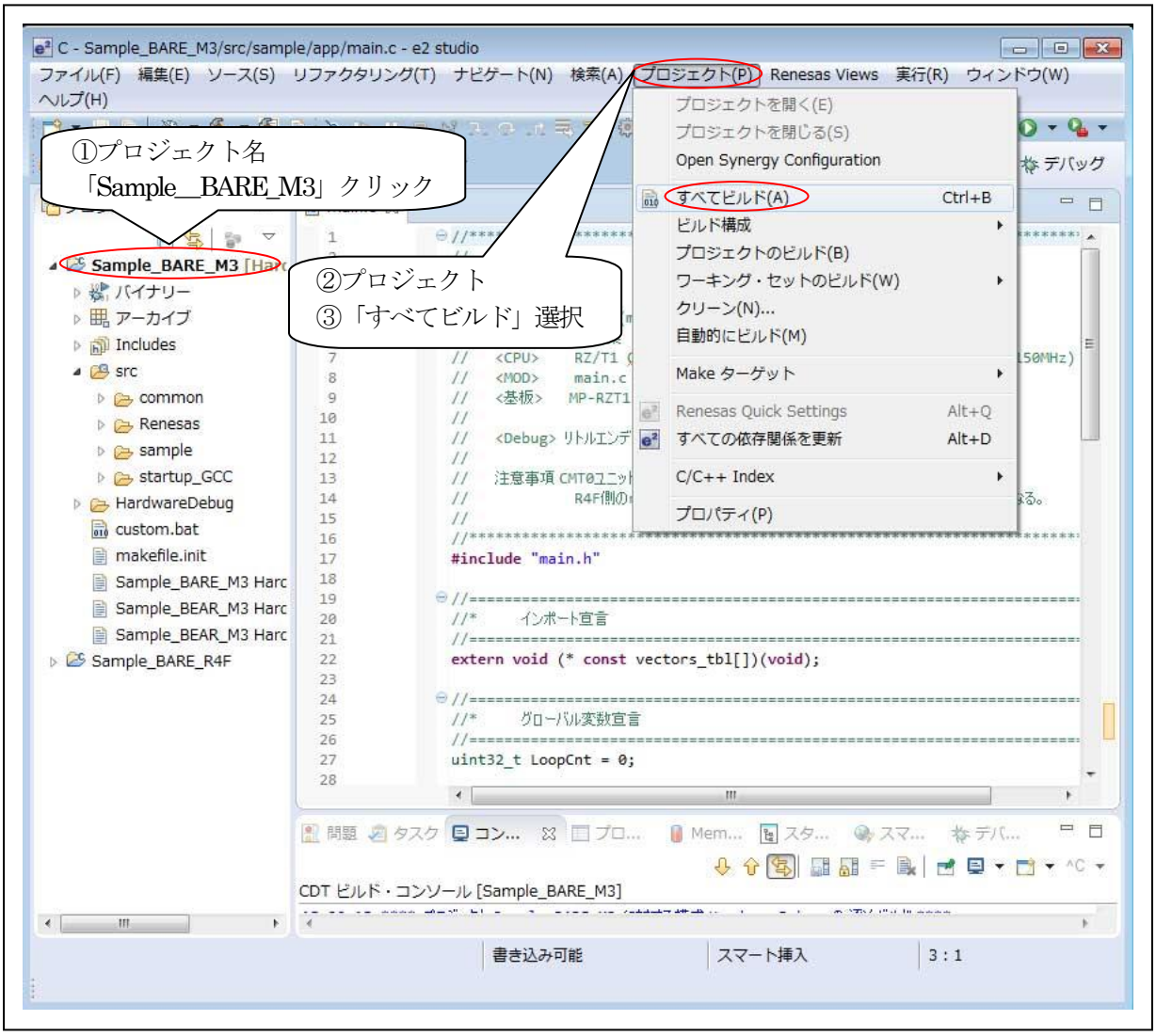
1）すべてビルド



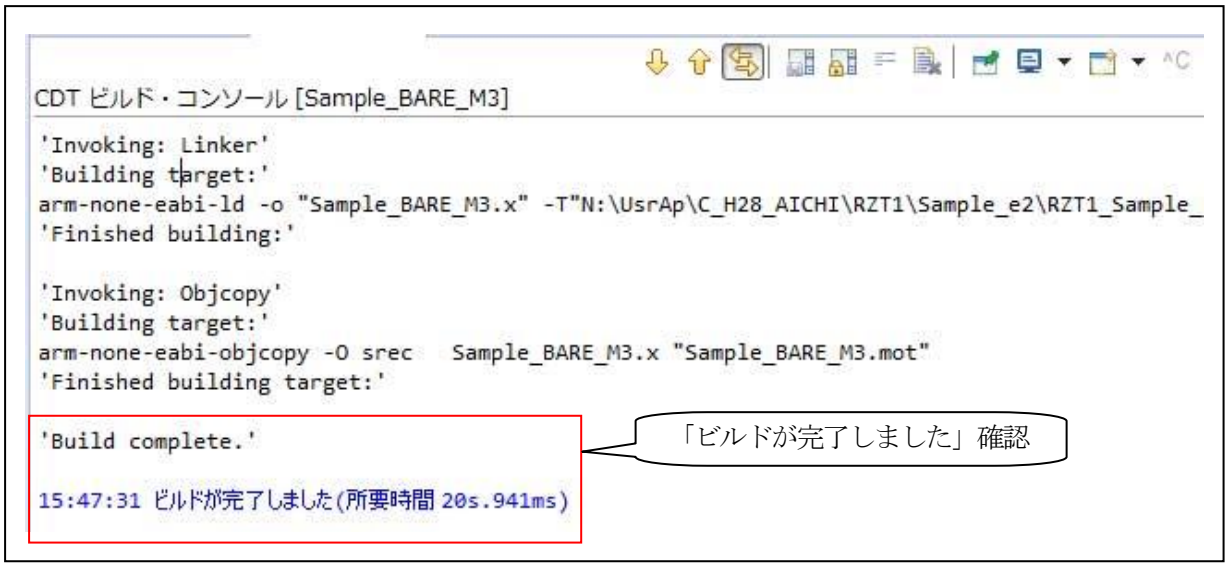
2）ビルド結果

３－２．コア【M3】側サンプル「Sample_BARE_M3」をビルドする。

１）すべてビルド



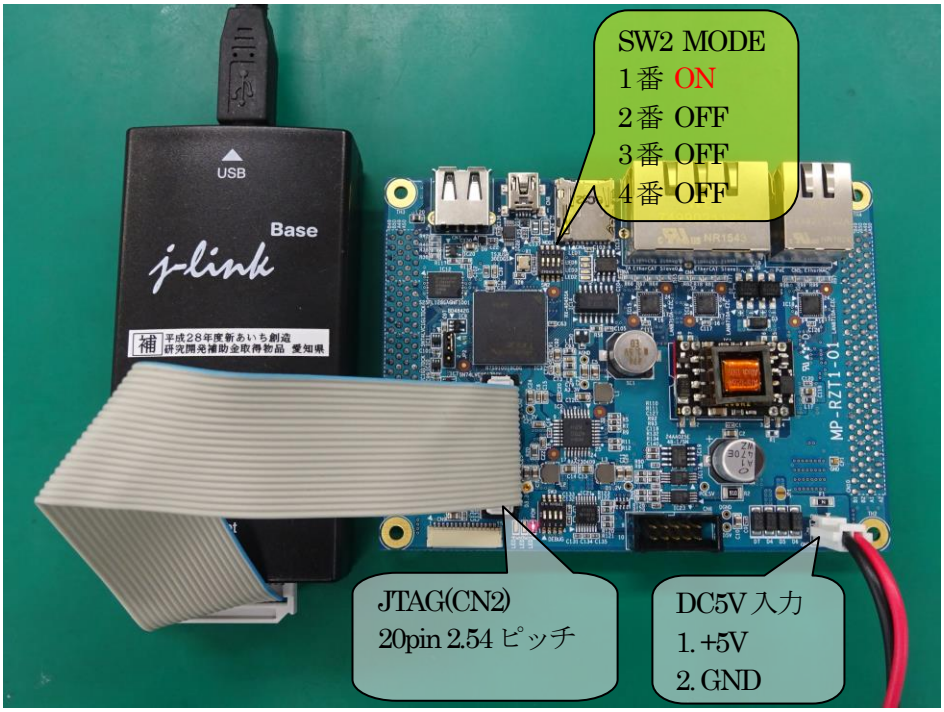
２）ビルド結果



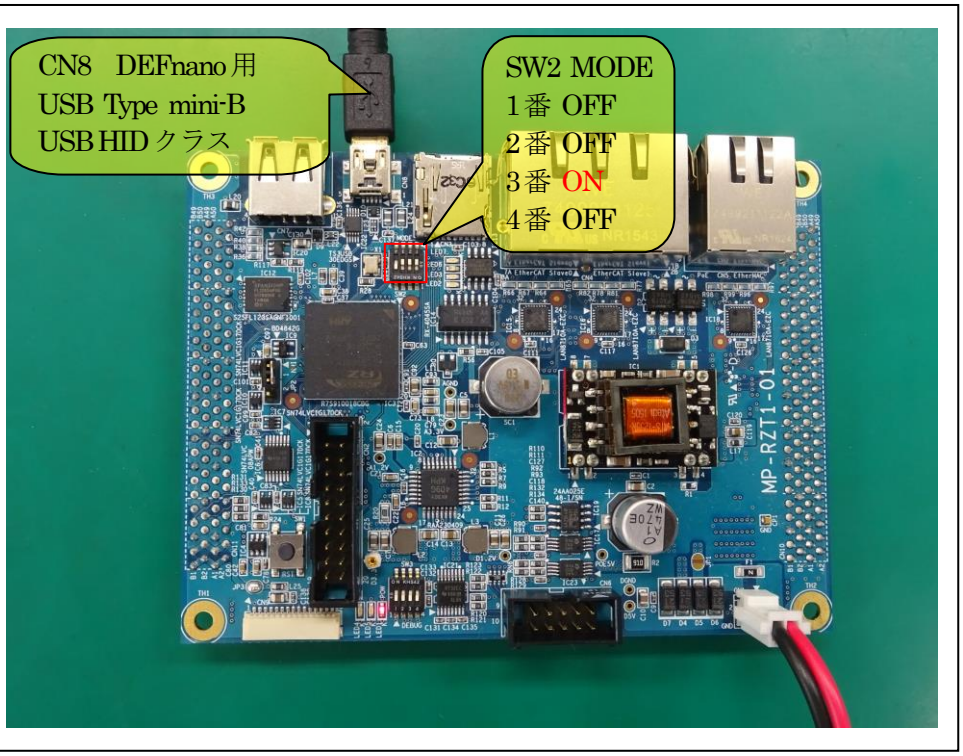
４．サンプルプロジェクト「Sample_BARE」をデバッグする。

４－１．MP-RZT1-01 のデバッグ準備

１）コア【R4F】側とコア【M3】側を J-Link BASE のみ使用してデバッグする場合



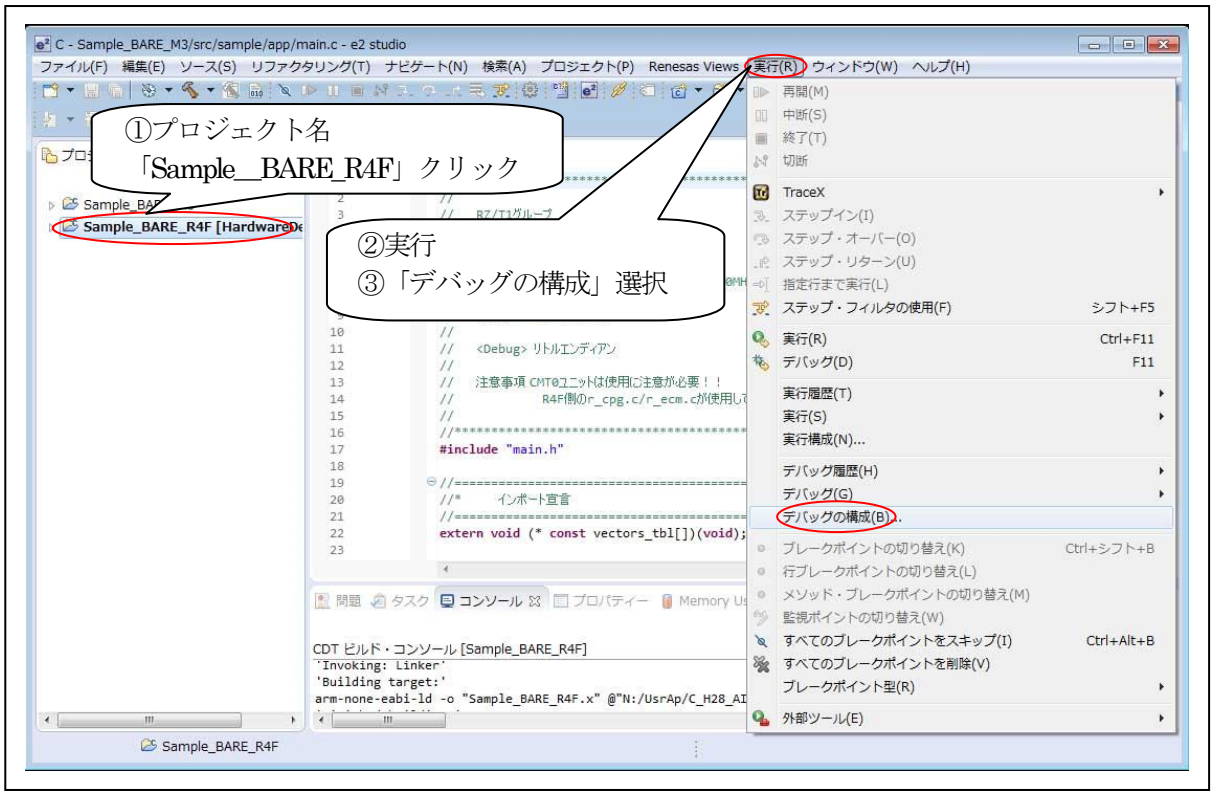
２）コア【R4F】側を DEFnano、コア【M3】側を ROM 化してデバッグする場合

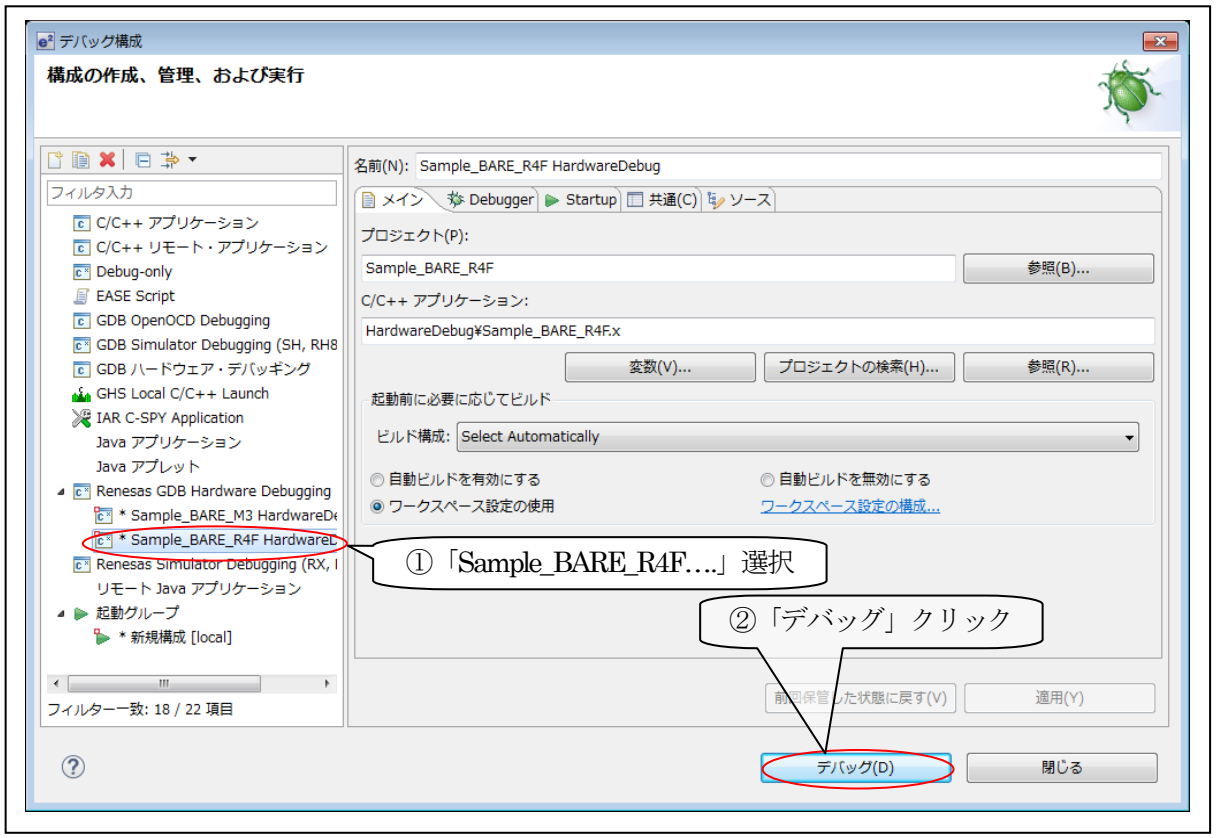


４－２．コア【R4F】側のデバッッガ初期起動

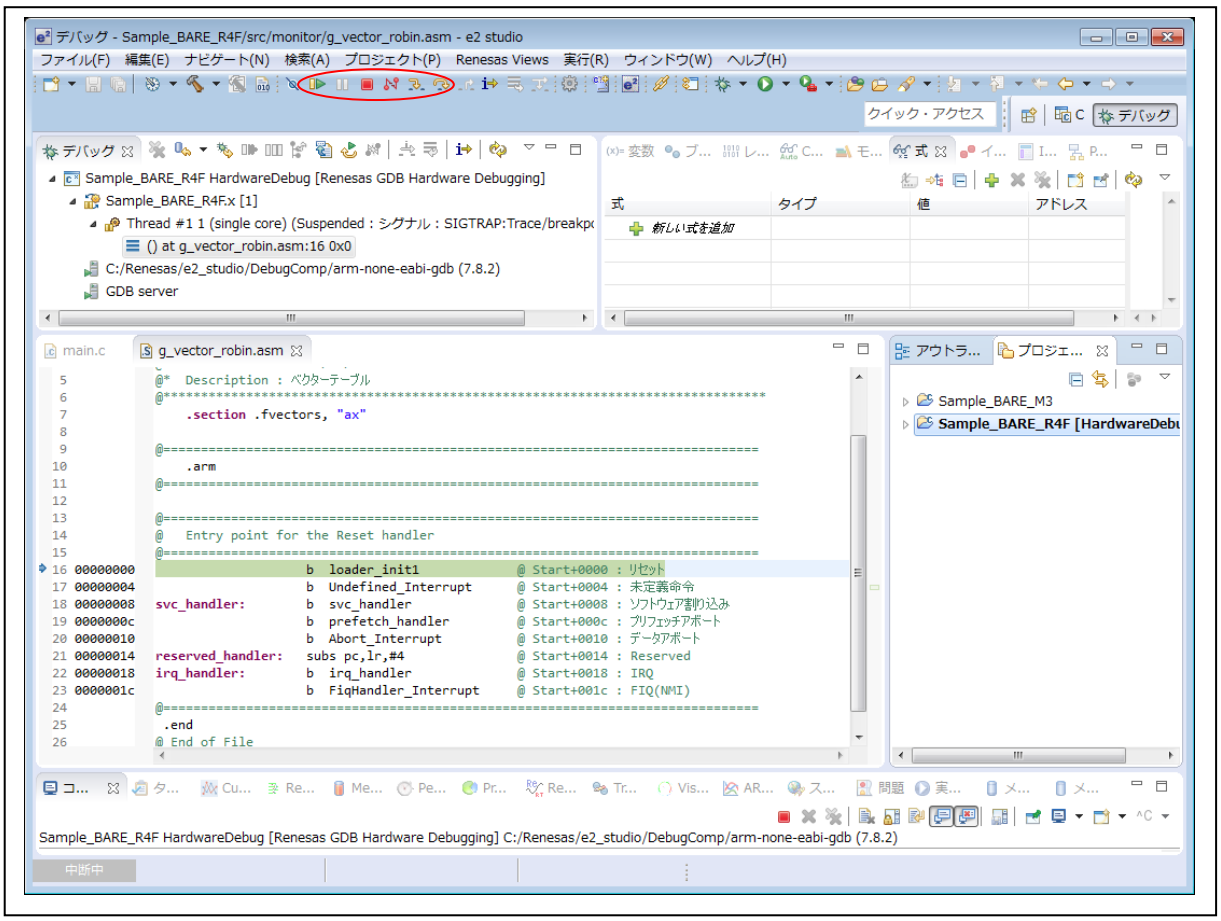
４－２－１．J-Link BASE を使用（詳細なデバッグ操作は SEGGER 提供「J-Link USER Guide」を参照）

1）「デバッグの構成」を選択



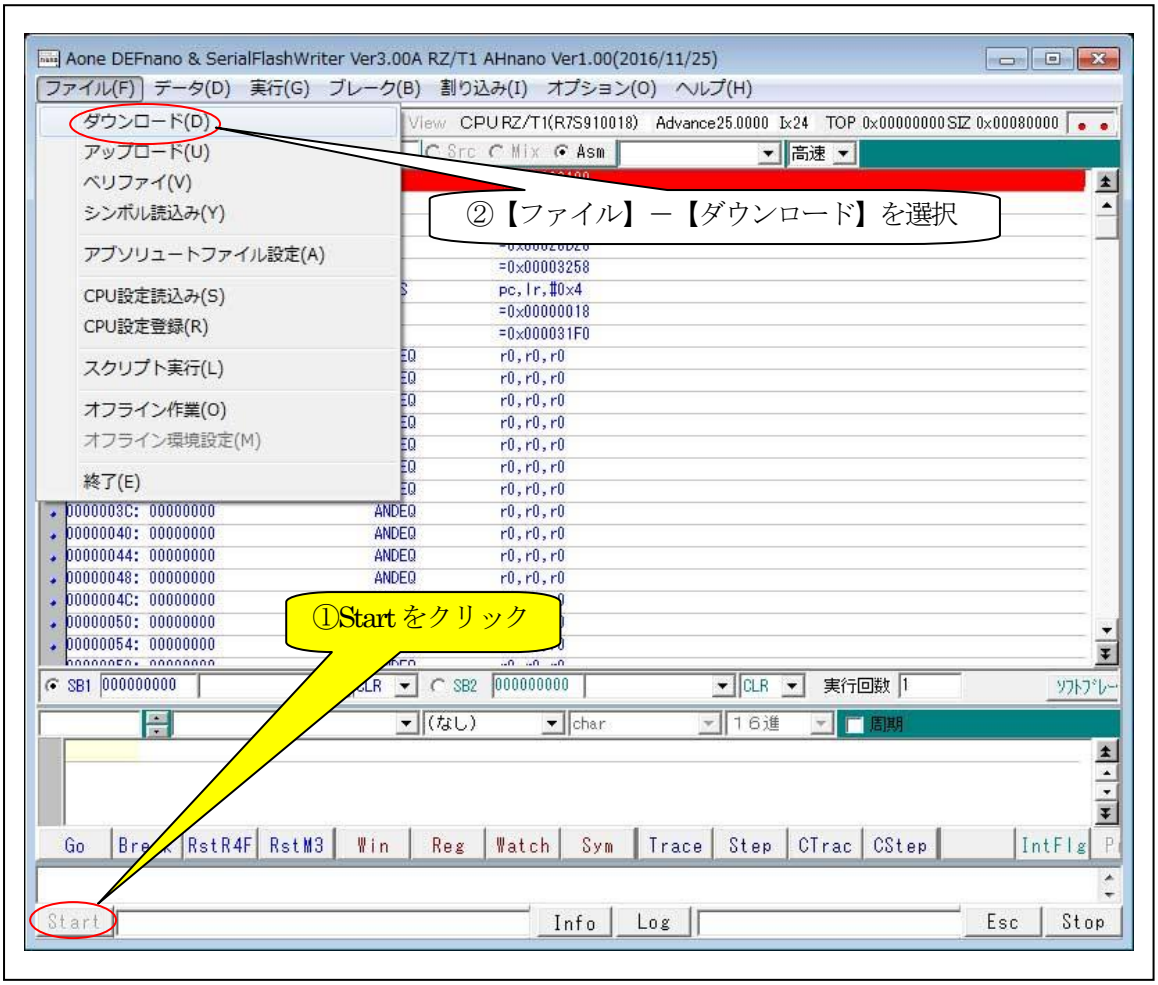


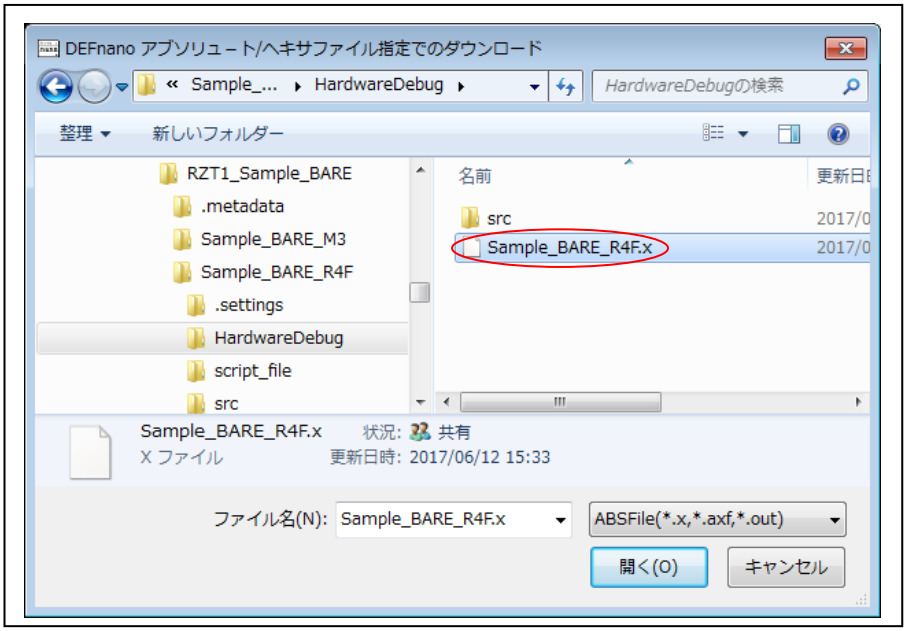
2）デバッガ起動画面


デバッグ - Sample_BARE_R4F/src/monitor/g_vector_robin.asm - e2 studio
ファイル(F) 編集(E) ナビゲート(N) 検索(A) プロジェクト(P) Renesas Views 実行(R) ウィンドウ(W) ヘルプ(H)
クイック・アクセス
デバッグ
Sample_BARE_R4F HardwareDebug [Renesas GDB Hardware Debugging]
Sample_BARE_R4F.x [1]
Thread #1 1 (single core) (Suspended : シグナル : SIGTRAP:Trace/breakpo
() at g_vector_robin.asm:16 0x0
C:/Renesas/e2_studio/DebugComp/arm-none-eabi-gdb (7.8.2)
GDB server
式
タイプ
値
アドレス
新しい式を追加
main.c
g_vector_robin.asm
@* Description : ベクターテーブル
.section .fvectors, "ax"
.arm
@ Entry point for the Reset handler
00000000 b loader_init1 @ Start+0000 : リセット
00000004 b Undefined_Interrupt @ Start+0004 : 未定義命令
svc_handler: 00000008 b svc_handler @ Start+0008 : ソフトウェア割り込み
0000000c b prefetch_handler @ Start+000c : プリフェッチアボート
00000010 b Abort_Interrupt @ Start+0010 : データアボート
reserved_handler: 00000014 subs pc,lr,#4 @ Start+0014 : Reserved
irq_handler: 00000018 b irq_handler @ Start+0018 : IRQ
0000001c b FiqHandler_Interrupt @ Start+001c : FIQ(NMI)
.end
@ End of File
アウトラ...
プロジェ...
Sample_BARE_M3
Sample_BARE_R4F [HardwareDebu
Sample_BARE_R4F HardwareDebug [Renesas GDB Hardware Debugging] C:/Renesas/e2_studio/DebugComp/arm-none-eabi-gdb (7.8.2)
中断中

４－２－２．DEFnano を使用（詳細なデバッグ操作は Aone 提供「DEFnano_Vx_xx.pdf」を参照）

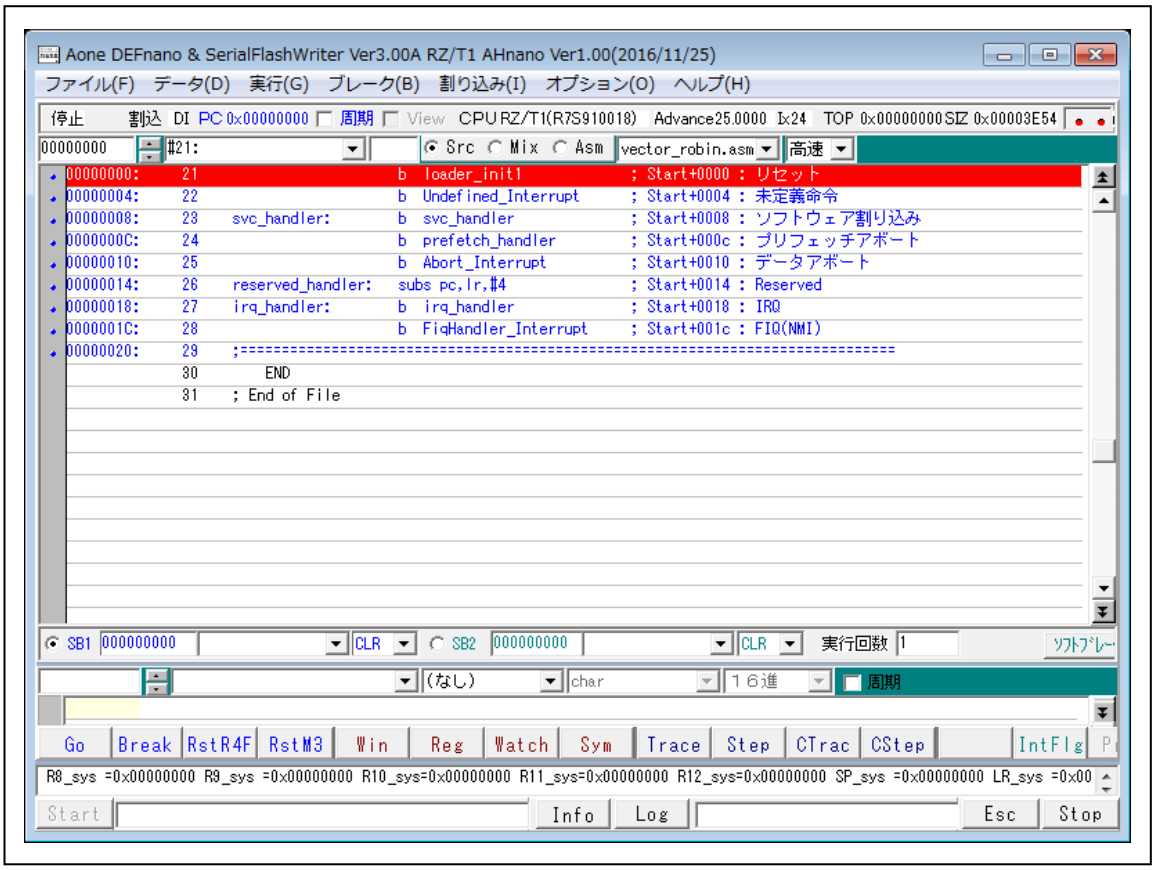
１）サンプル「Sample_BARE_R4F」をダウンロードする。





<RZT1_Sample_BARE><Sample_BARE_R4F><HardeareDebug>[Sample_BARE_R4F.x]を選択後、「開く」をクリック

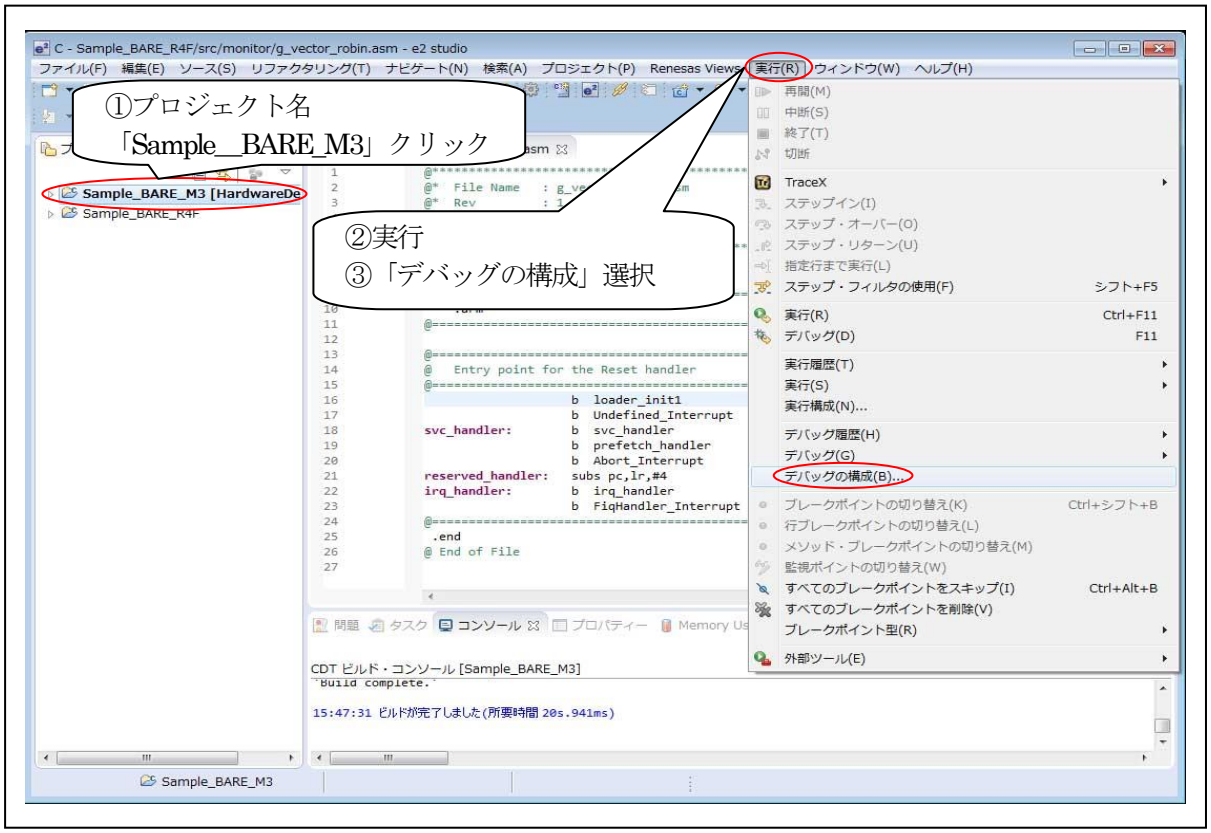
2）デバッガ起動画面

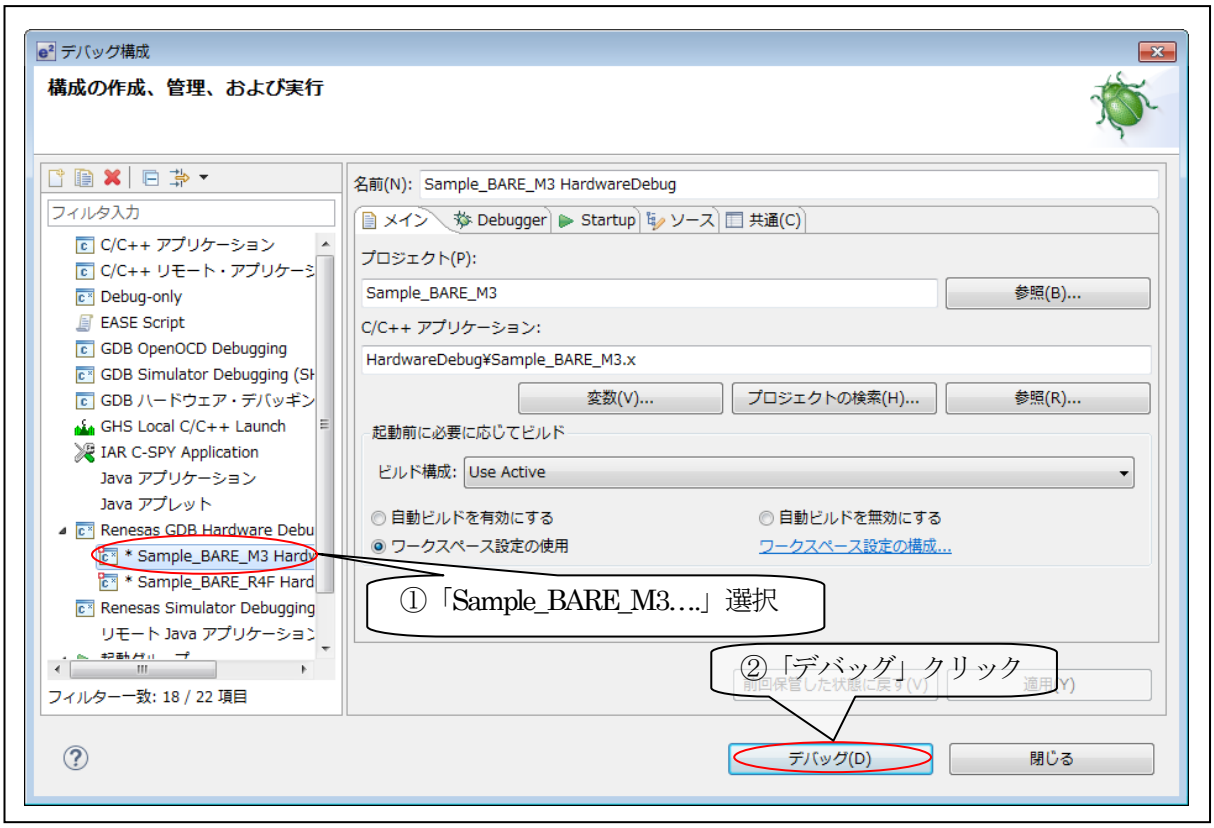

Aone DEFnano & SerialFlashWriter Ver3.00A RZ/T1 AHnano Ver1.00(2016/11/25)
ファイル(F) データ(D) 実行(G) ブレーク(B) 割り込み(I) オプション(O) ヘルプ(H)
停止 割込 DI PC 0x00000000 周期 View CPU RZ/T1(R7S910018) Advance25.0000 Ix24 TOP 0x00000000 SIZ 0x00003E54
00000000: 21 b loader_init1 ; Start+0000 : リセット
00000004: 22 b Undefined_Interrupt ; Start+0004 : 未定義命令
00000008: 23 svc_handler: b svc_handler ; Start+0008 : ソフトウェア割り込み
0000000C: 24 b prefetch_handler ; Start+000c : プリフェッチアボート
00000010: 25 b Abort_Interrupt ; Start+0010 : データアボート
00000014: 26 reserved_handler: subs pc,lr,#4 ; Start+0014 : Reserved
00000018: 27 irq_handler: b irq_handler ; Start+0018 : IRQ
0000001C: 28 b FiqHandler_Interrupt ; Start+001c : FIQ(NMI)
00000020: 29
30 END
31 ; End of File
Go Break RstR4F RstM3 Win Reg Watch Sym Trace Step CTrac CStep IntFlg
Start Info Log Esc Stop

４－３．コア【M3】側のデバッッガ初期起動

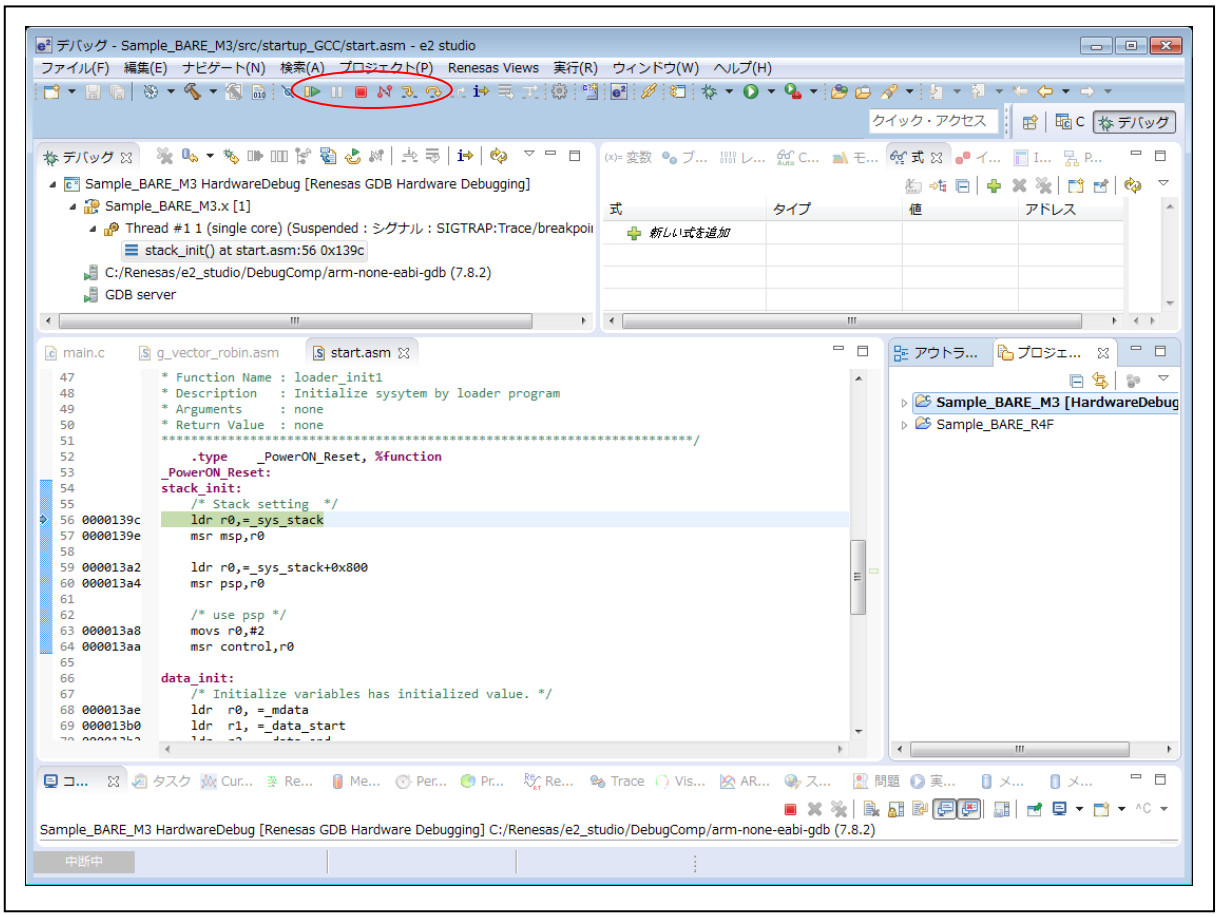
４－３－１．J-Link BASE を使用（詳細なデバッグ操作は SEGGER 提供「J-Link USER Guide」を参照）

1）「デバッグの構成」を選択





2）デバッガ起動画面


デバッグ - Sample_BARE_M3/src/startup_GCC/start.asm - e2 studio
ファイル(F) 編集(E) ナビゲート(N) 検索(A) プロジェクト(P) Renesas Views 実行(R) ウィンドウ(W) ヘルプ(H)
クイック・アクセス
デバッグ
Sample_BARE_M3 HardwareDebug [Renesas GDB Hardware Debugging]
Sample_BARE_M3.x [1]
Thread #1 1 (single core) (Suspended : シグナル : SIGTRAP:Trace/breakpoi
stack_init() at start.asm:56 0x139c
C:/Renesas/e2_studio/DebugComp/arm-none-eabi-gdb (7.8.2)
GDB server
式
タイプ
値
アドレス
新しい式を追加
main.c
g_vector_robin.asm
start.asm
* Function Name : loader_init1
* Description : Initialize sysytem by loader program
* Arguments : none
* Return Value : none
.type _PowerON_Reset, %function
_PowerON_Reset:
stack_init:
/* Stack setting */
0000139c ldr r0,=_sys_stack
0000139e msr msp,r0
000013a2 ldr r0,=_sys_stack+0x800
000013a4 msr psp,r0
/* use psp */
000013a8 movs r0,#2
000013aa msr control,r0
data_init:
/* Initialize variables has initialized value. */
000013ae ldr r0, =_mdata
000013b0 ldr r1, =_data_start
アウトラ...
プロジェ...
Sample_BARE_M3 [HardwareDebug
Sample_BARE_R4F
Sample_BARE_M3 HardwareDebug [Renesas GDB Hardware Debugging] C:/Renesas/e2_studio/DebugComp/arm-none-eabi-gdb (7.8.2)
中断中

４－４．デバッグに関する特記事項

１）J-Link BASE を使用した場合、コア【R4F】・【M3】を同時にデュアルコアデバッグすることはできません。
必ず、片側のコアを ROM 化してデバッグ作業を進めて下さい。

デバッグ手順の一例として、
①R4F 側で使用する I/O の初期化のみを実施する「初期化プログラム」を作成する。
（M3 側のリセット解除処理を含む）
②上記の「初期化プログラム」を ROM 化して、コア【M3】側のデバッグを進め完成させる。
③上記の「コア【M3】側プログラム」を ROM 化して、コア【R4F】側のデバッグを進め完成させる。
④総合検証で「②<－>③」のデバッグ作業を進めて完成品にする。

２）I-jet や DEFnano を使用してのデバッグに関する共通事項は、デバッグ時はプログラムコードを RZ/T1 の内蔵 RAM にダウンロードしてデバッグ作業を進めます。
特に JTAG デバッガを使用したサンプル例ですとダウンロード時にシリアル FROM に書き込み後、アプリ側でロードさせるコードが含まれていますが、MP-RZT1-01 を使用する場合は不要です。

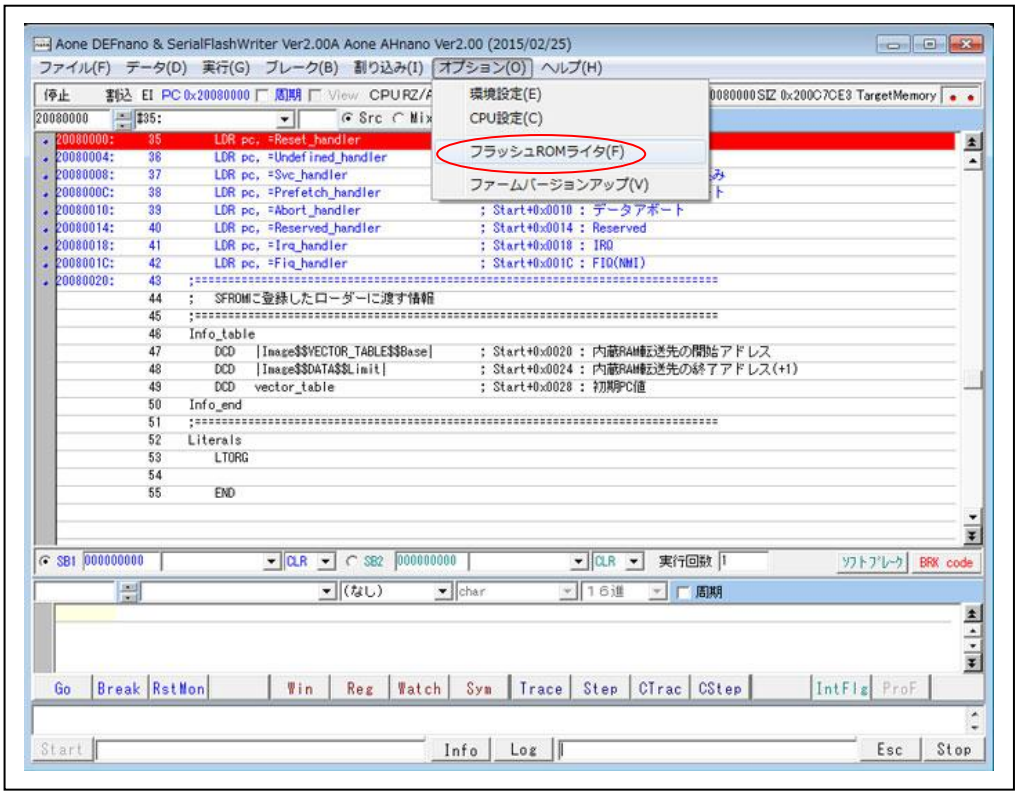
３）MP-RZT1-01 基板は、オンボードのシリアル FROM にローダーとデバッガ用ファームが書き込み済みで出荷しています。

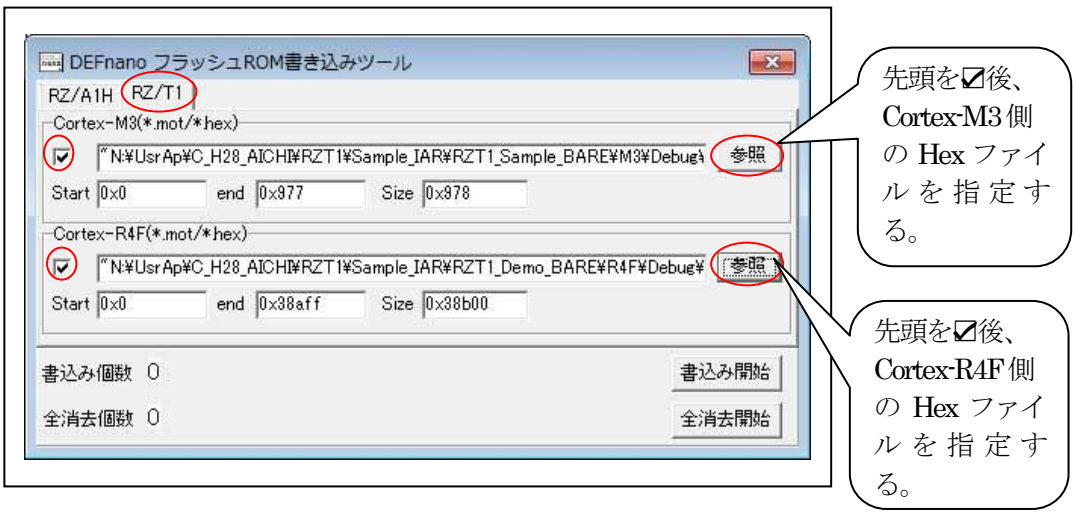
４）ROM 書き込みに関しては、5 項をご覧下さい。

５．ROM 書き込み

５－１．DEFnano を使用して ROM 書き込みする。

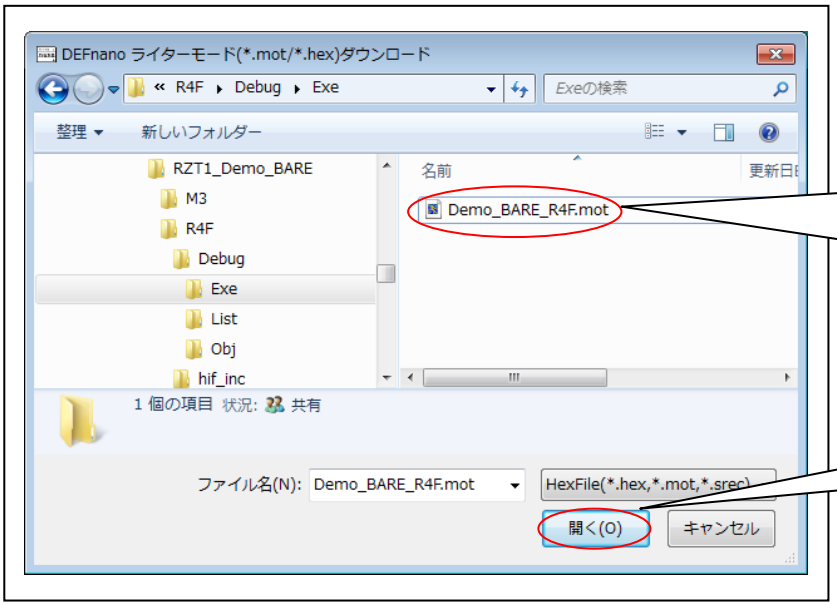
【オプション】－【フラッシュ ROM ライタ】を選択





先頭を☑後、Cortex-M3 側の Hex ファイルを指定する。

先頭を☑後、Cortex-R4F 側の Hex ファイルを指定する。



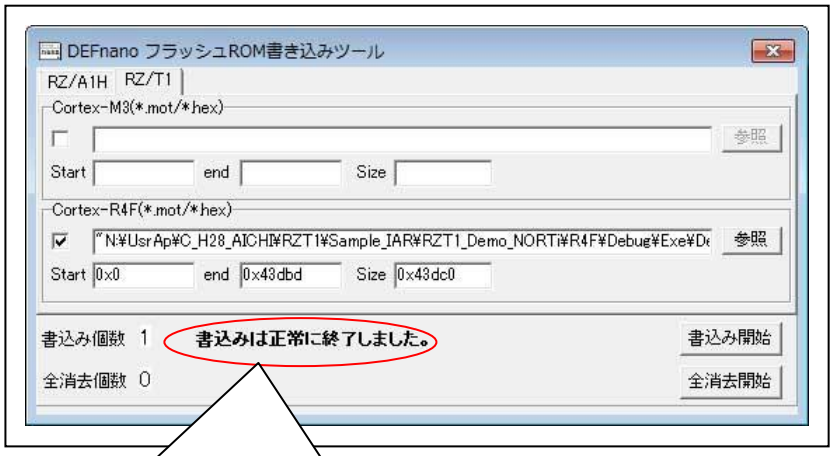
ライターモードでは、「*.mot」の Hex ファイルを指定する。

「開く」をクリックする。

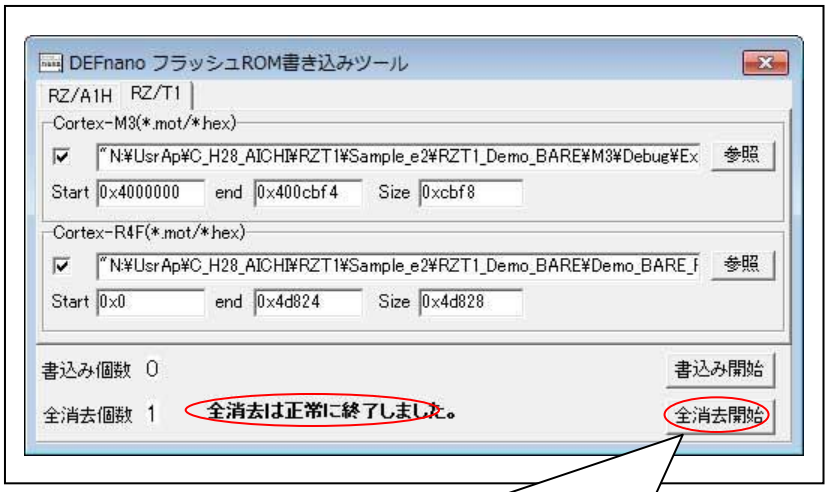
「書込み開始」をクリックする。
☑選択されたファイルを書き込みます。

【Cortex-M3 側】
内部処理に応じてインジケータ表示
①RAM にダウンロード中
②セクターイレーズ
③シリアルフラッシュに書込み中
④ベリファイ中

【Cortex-R4F 側】
内部処理に応じてインジケータ表示
①RAM にダウンロード中
②セクターイレーズ
③シリアルフラッシュに書込み中
④ベリファイ中



正常終了しますと、「書込みは正常に終了しました。」と表示され、個数がカウントアップします。個数は、この窓を閉じますとゼロになります。
この例は、R4F 側のみを Write した時の画面です。



「全消去開始」をクリックしますと、
アプリケーションエリア(セクター２～３１)を全消去します。M3/R4F 側の個別指定はできません。

シリアルフラッシュ ROM マップ(MP-RZT1-01)	
ローダ・デバッグ用ファーム	０～１セクター 0x0 ～0x1_FFFF
アプリケーションエリア Cortex-M3	２～１５セクター 0x2_0000～0x9_FFFF 0xA_000～0xF_FFFF（予備）
アプリケーションエリア Cortex-R4F	１６～３１セクター 0x10_0000～0x17_FFFF 0x18_0000～0x1F_FFFF（予備）

6．注意事項

・本文書の著作権は、エーワン（株）が保有します。
・本文書を無断での転載は一切禁止します。
・本文書に記載されている内容についての質問やサポートはお受けすることが出来ません。
・本文章に関して、ARM 社およびルネサス　エレクトロニクス社および SEGGER 社およびエンビテック社への問い合わせは御遠慮願います。
・本文書の内容に従い、使用した結果、損害が発生しても、弊社では一切の責任は負わないものとします。
・本文書の内容に関して、万全を期して作成しましたが、ご不審な点、誤りなどの点がありましたら弊社までご連絡くだされば幸いです。
・本文書の内容は、予告なしに変更されることがあります。

7．商標

・J-Link BASE は、SEGGER 社の登録商標、または商品名称です。
・ ARM Cortex, Thumb および ARM Cortex-M3/R4F は ARM Limited の EU およびその他の国における商標および登録商標です
・RZ/T1 は、ルネサス　エレクトロニクス株式会社の登録商標、または商品名です。
・その他の会社名、製品名は、各社の登録商標または商標です。

8．参考文献

・「RZ/T1 グループ　ユーザーズマニュアル　ハードウェア編」
ルネサス　エレクトロニクス株式会社
・ルネサス　エレクトロニクス株式会社提供のサンプル集
・「J-Link　User Guide」　SEGGER 社
・「J-Flash　User Guide」　SEGGER 社
・「Flasher　User Guide」　SEGGER 社
・その他

〒486-0852
愛知県春日井市下市場町 6-9-20
エーワン株式会社
http://www.robin-w.com