===== 必ずお読みください =====

M16C シリーズ,R8C ファミリ用 C コンパイラパッケージ
V.5.45 Release 01
ガイドブック
(第 2 版)

株式会社ルネサス ソリューションズ

2010 年 6 月 1 日

概要

本資料は M16C シリーズ、R8C ファミリ用 C コンパイラパッケージ V.5.45 Release 01 を導入する際の手引きを説明します。インストール、プロジェクトの作成等、このガイドブックをご覧いただきますようお願い申しあげます。

A　Cコンパイラパッケージ V.5.45 Release 01 への移行ガイド

本章では、従来バージョンのコンパイラを使用して作成したプロジェクトを V.5.45 Release 01(へ移行する際の注意事項及び V.5.45 Release 01 で新たにプロジェクトを作成する場合のポイントを説明します。

A.1　V.5.30 Release 02 以前からV.5.45 Release 01 へ移行する場合及びルネサス社製リアルタイムOSと組み合わせて使用する場合のポイント

A.1.1 スタートアップファイルの変更

・_init( )関数

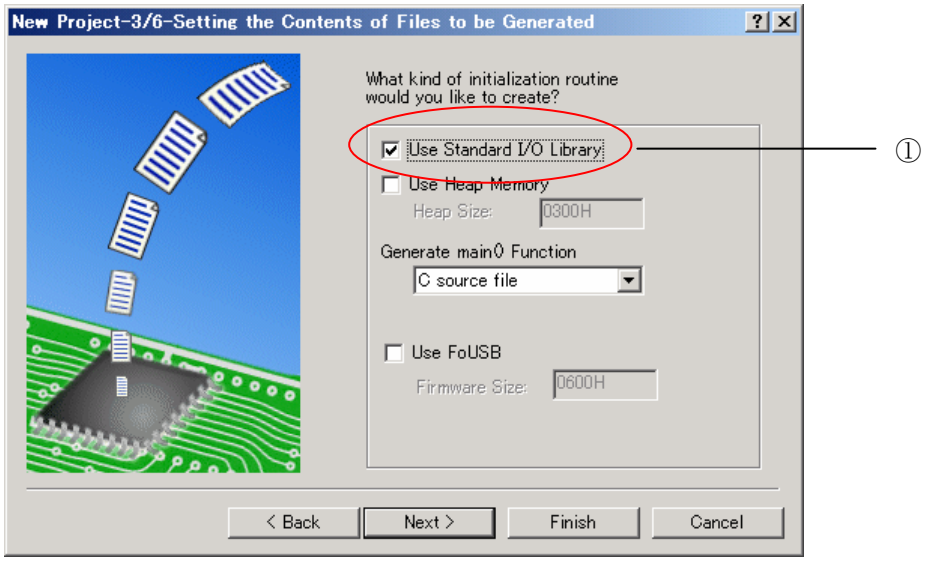
V.5.40 Release 00(A)からライブラリ関数 init( )の名称を_init( )へ変更しています。

そのため、そのままビルドを行うとリンク時に　'_init' value is undefined　が発生する場合があります。

エラーが発生するのは、

- V.5.40Release00 以前のバージョンでプロジェクト作成を行った際に、①の選択を有効にする
- ncrt0.a30 を直接変更して、init 関数の呼び出しを有効にしている

場合です。



本エラーが発生した場合は、ncrt0.a30 の以下の部分を変更してください。

【コンパイラ付属のスタートアップファイル(ncrt0.a30)を使用されている場合】

変更前

```
;=======================================================
; Initialize standard I/O
;-------------------------------------------------------------
.if __STANDARD_IO__ == 1
        .glb    _init
        .call   _init,G
        jsr.a   _init
.endif
```

変更後

```
;=======================================================
; Initialize standard I/O
;-------------------------------------------------------------
.if __STANDARD_IO__ == 1
        .glb    __init
        .call   __init,G
        jsr.a   __init
.endif
;-------------------------------------------------------------
```

※ NC30WA をお使いの場合は、.if～.endif により M16C シリーズ用の処理と R8C ファミリ用の処理を分けています。修正する際にはご使用頂いているマイコン側の記述を修正してください。
もし、修正後エラーが解消できない場合は、M16C 用と R8C 用での修正箇所を間違えている事が考えられますので再度確認してください。

【リアルタイム OS 付属のスタートアップファイル(crt0mr.a30)を使用されている場合】

○M3T-MR308 の場合

変　更

```
; +-------------------------------------------------+
; |    User Initial Routine ( if there are ) |
; +-------------------------------------------------+
; Initialize standard I/O
        .GLB    _init
        JSR.A   _init
```

変　更

```
; +-------------------------------------------------+
; |    User Initial Routine ( if there are ) |
; +-------------------------------------------------+
; Initialize standard I/O
        .GLB    __init
        JSR.A   __init
;-------------------------------------------------
```

○M3T-MR30 の場合
※最新バージョン(V.3.30 Release 2)を含むいくつかのバージョンでは下記ジャンプ処理はコメントアウトされています。

変　更
```
;======================================================================
; Initialize standard I/O
;----------------------------------------------------------------------
        .glb    _init
        jsr.a   _init
```

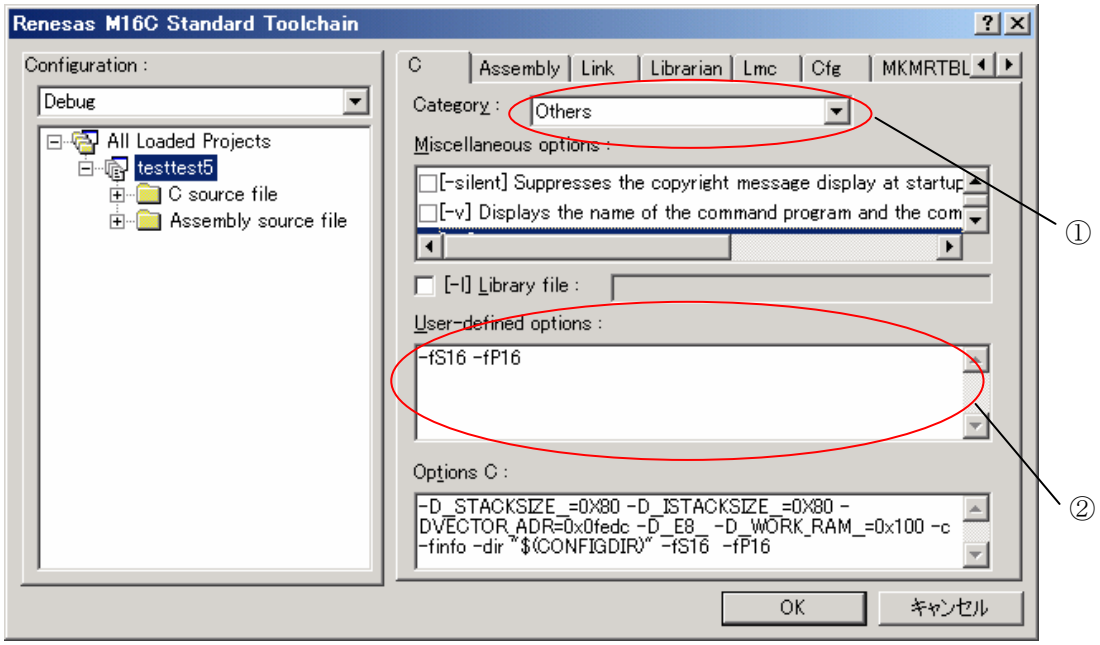
変更後
```
;======================================================================
; Initialize standard I/O
;----------------------------------------------------------------------
        .glb    __init
        jsr.a   __init
```

A.1.2 size_t,ptrdiff_tのサイズ変更

V.5.40 Release00 より size_t と ptrdiff_t のサイズを 16 ビットから 32 ビットへ変更しています。
V.5.30 Release02 以前のバージョンで作成した size_t 及び ptrdiff_t 型を使用したユーザライブラリを利用する場合など size_t,ptrdiff_t を 16bit で使用する必要がある場合は、以下の設定を行ってください。

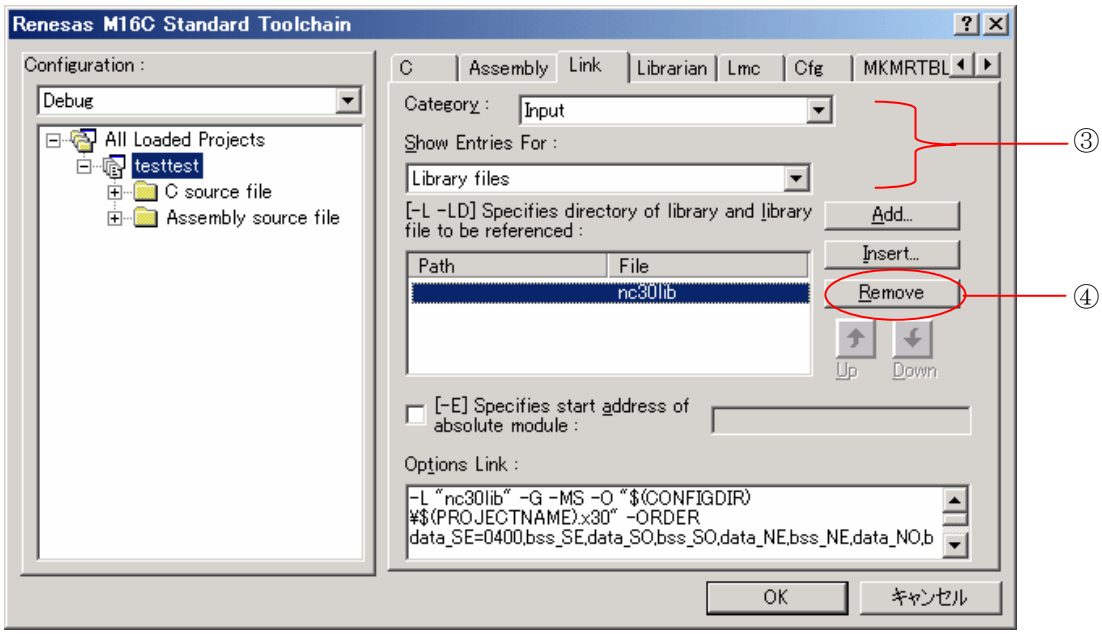
- コンパイルオプション-fsizet_16 (-fS16),-fptrdiff_t (-fP16)を設定する。
- リンクするライブラリを
  NC30WA の場合: nc30lib.lib から nc30s16.lib , r8clib.lib から r8cs16.lib
  NC308WA の場合:nc308lib.lib から nc308_16.lib , nc382lib.lib から nc382_16.lib
  へそれぞれ変更する。

[HEW 上での設定手順]
コンパイルオプション-fsizet_16 (-fS16),-fptrdiff_t (-fP16)を設定する。
HEW の[ビルド]→[Renesas M16C Standard Toolchain…]→C タブを選択する。



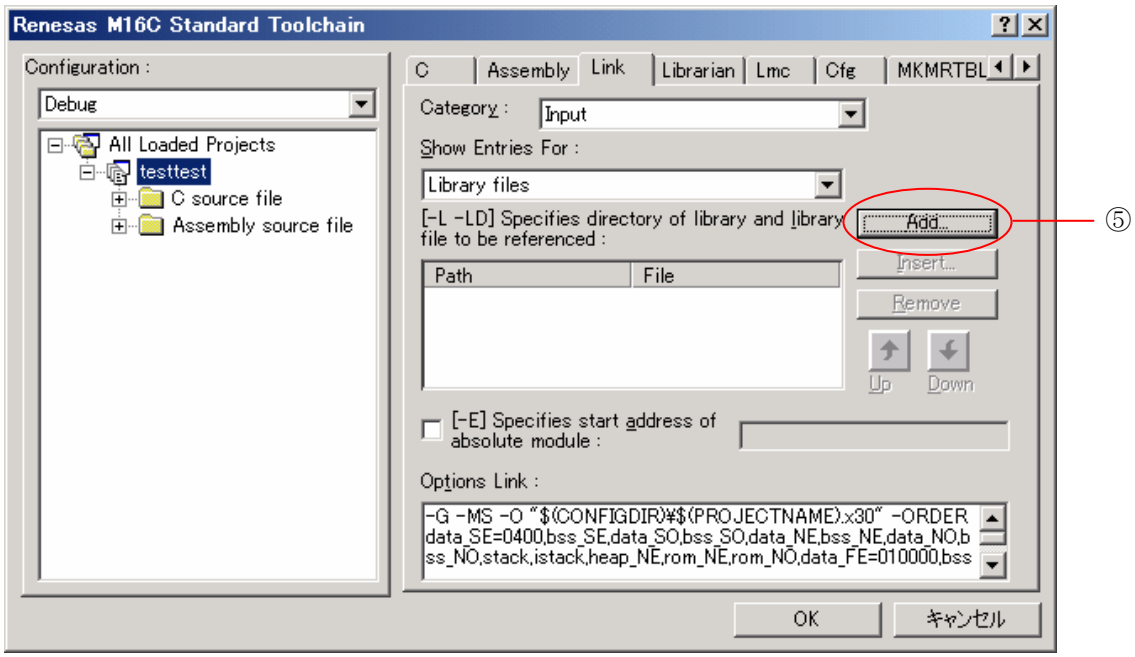
① Category:で Others を選択
② User-defined options:に–fsize_t16 (もしくは、-fS16),-fptrdiff_t16(もしくは、-fP16)を入力

リンクするライブラリを変更する。
HEW の[ビルド]→[Renesas M16C Standard Toolchain…]→Link タブを選択する。



③
Category: Input
Show Entries For : Library files
を選択する

④Remove ボタンを押して一旦 nc30lib.lib を削除する。
R8C の場合は、r8clib.lib
M16C/80 の場合は、nc308lib.lib
M32C/80 の場合は、nc382lib.lib



⑤Add ボタンを押して、[Library file]を選択する。

⑥sise_t,ptrdiff_t のサイズが 16 ビット用のライブラリ名を入力する。
M16C の場合：nc30s16.lib
R8C の場合　：r8cs16.lib
M16C/80 の場合：nc308_16.lib
M32C/80 の場合：nc382_16.lib
OK ボタンを押して終了する。

【リアルタイム OS のコンフィグレータにより生成した makefile を使用している場合】

○M3T-MR308 の場合
makefile 内の以下の部分を修正してください。
以下は、nc308_16.lib をリンク対象とする場合です。

修正前
```
# Use the following macro when you use C-libraries for M32C/80 series.
#NEWLIB          = -l nc382lib
```

修　正
```
# Use the following macro when you use C-libraries for M32C/80 series.
#NEWLIB          = -l nc308_16.lib
```

○M3T-MR30 の場合

makefile 内の"LIBS"マクロは、コンフィグレータによって"nc30lib.lib"に書き換わるため、"LIBS"マクロの修正によって対応することは出来ません。
このため、"$(LINKLST)"生成時の処理を修正することによって対応します。
makefile 内の以下の部分を修正してください。

修正前

```
$(LINKLST): makefile
        @mrecho "-o $(PROGRAM)" $(LINKLST)
        @mrecho -a "-ld $(LIB30)" $(LINKLST)
        @mrecho -a "-l $(LIBS)" $(LINKLST)
```

修　正

```
$(LINKLST): makefile
        @mrecho "-o $(PROGRAM)" $(LINKLST)
        @mrecho -a "-ld $(LIB30)" $(LINKLST)
        @mrecho -a "-l nc30s16.lib -l $(LIBS)" $(LINKLST)
```

A.1.3 割り込みベクタテーブル

V.5.40 Release 00 より、割り込み関数をベクタ番号指定して宣言すると自動的にベクタテーブルを生成します。このため、コンパイルオプション-fmake_vector_table(-fMVT)の選択が不要になりました。また、ベクタテーブルはセクション vector で管理するため、セクション__NC_rvector の設定も不要になりました。
リンク時に"Can’t generate automatically the variable interrupt vector table."が発生した場合は、以下の通りに可変ベクタテーブルの設定を変更してください。

(変更前)

```
        .if         __MVT__==0
        ;------------------------------------------------------------
        ; variable vector section
        ;------------------------------------------------------------
                .section   vector,ROMDATA ; variable vector table
                .org       VECTOR_ADR

        .if         M60TYPE == 1
                .lword     dummy_int                ; vector 0 (BRK)
                .lword     dummy_int                ; vector 1
                        :
                    (省略)
                        :
                .lword     dummy_int                ; vector 62
                .lword     dummy_int                ; vector 63
        .else ; __MVT__

                .section   __NC_rvector,ROMDATA
                .org                VECTOR_ADR

        .endif ; __MVT__
```

(変更後)

```
.section   vector,ROMDATA ; variable vector table
.org       VECTOR_ADR
```

A.1.4 スペシャルページテーブル

V.5.40 Release 00 より、拡張機能#pragma SPECIAL をベクタ番号指定して宣言すると自動的にベクタテーブルを生成します。このため、コンパイルオプション-fmake_special_table(-fMST)の選択が不要になりました。また、スペシャルページテーブルはアセンブル時に自動生成されるセクション svector で管理するため、セクション __NC_svector の設定、およびマクロ定義された SPECIAL 記述によるベクタ番号の設定も不要(設定不可)になりました。
リンク時に"Can’t generate automatically the variable interrupt vector table."が発生した場合は、以下の通りにスペシャルページテーブルの設定を変更してください。

(変更前)

```
.if         __MST__ == 0
;=================================================================
; fixed vector section
;-----------------------------------------------------------
.if          SVECTOR_ADR < 0fffdcH
             .section   svector,ROMDATA                    ; specialpage vector table
             .org                SVECTOR_ADR
.endif
;=================================================================
; special page defination
;-----------------------------------------------------------
;            macro is defined in ncrt0.a30
;            Format: SPECIAL number
;
;-----------------------------------------------------------
;            SPECIAL 255
;            SPECIAL 254
                    :
                (省略)
                    :
;            SPECIAL 19
;            SPECIAL 18
.else
.if          (SVECTOR_ADR < 0fffdcH)
             .section   __NC_svector,ROMDATA
             .org                SVECTOR_ADR
.endif
.endif ; __MST
```

(変更後)

全て削除

A.2　新規プロジェクトを作成する場合のポイント

A.2.1 CPUの選択

新規プロジェクト作成時に CPU Group でマイコンの機種を選択することができます。
マイコン機種選択で有効となるのは、

- sfr ヘッダファイルの登録
- 可変ベクタ割り込みエントリ関数登録ファイル（intprg.c）のワークスペースへの登録
- リンクアドレスの設定

になります。

A.2.2 CPU　Groupに記載のないマイコンで新規ワークスペースを作成する場合

○R8C/Tiny の場合

(1) CPU Series から R8C/Tiny を選択する。
(2) CPU Group から Other を選択する。

V.5.45 Release 01 を使用して新規プロジェクトを作成する場合は、使用する機種の ROM サイズにあったコンパイルオプションとライブラリが設定されているかを HEW の[ビルド]→[Renesas M16C Standard Toolchain…]で確認してください。

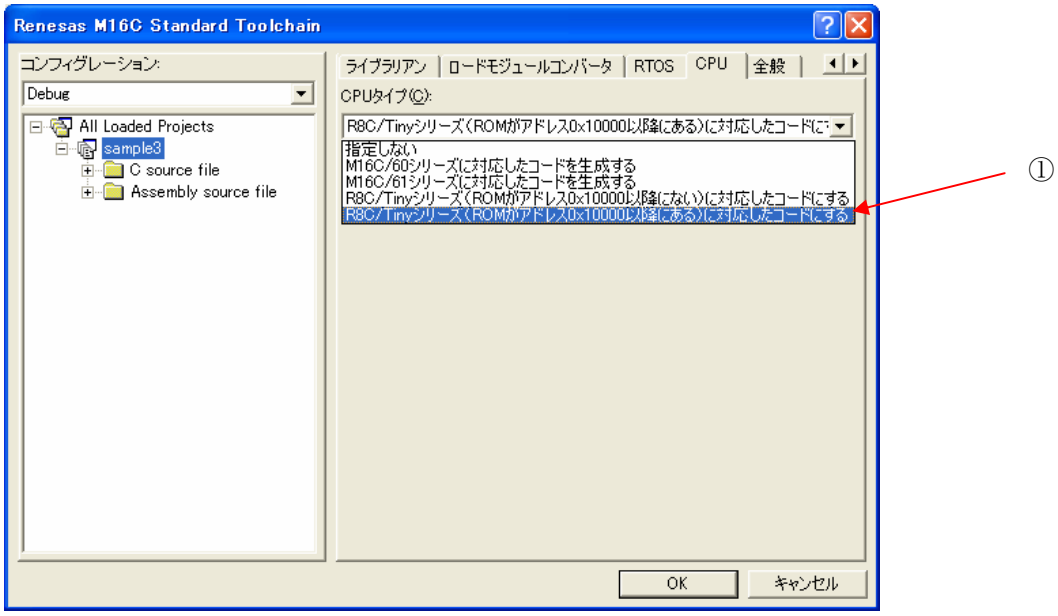
ROM サイズ	コンパイルオプション	ライブラリ
64K バイト未満	-R8C	r8clib.lib
64K バイト以上	-R8CE	nc30lib.lib

V.5.43 Release 00 以前のバージョンを使用して新規プロジェクトを作成する場合は、下記に示す変更を行ってください。

使用する機種の ROM 空間が 64K バイト境界を越える場合
(3)コンパイルオプション-R8C を-R8CE へ変更
(4)リンクするライブラリを r8clib.lib から nc30lib.lib へ変更

コンパイルオプション-R8C を-R8CE へ変更

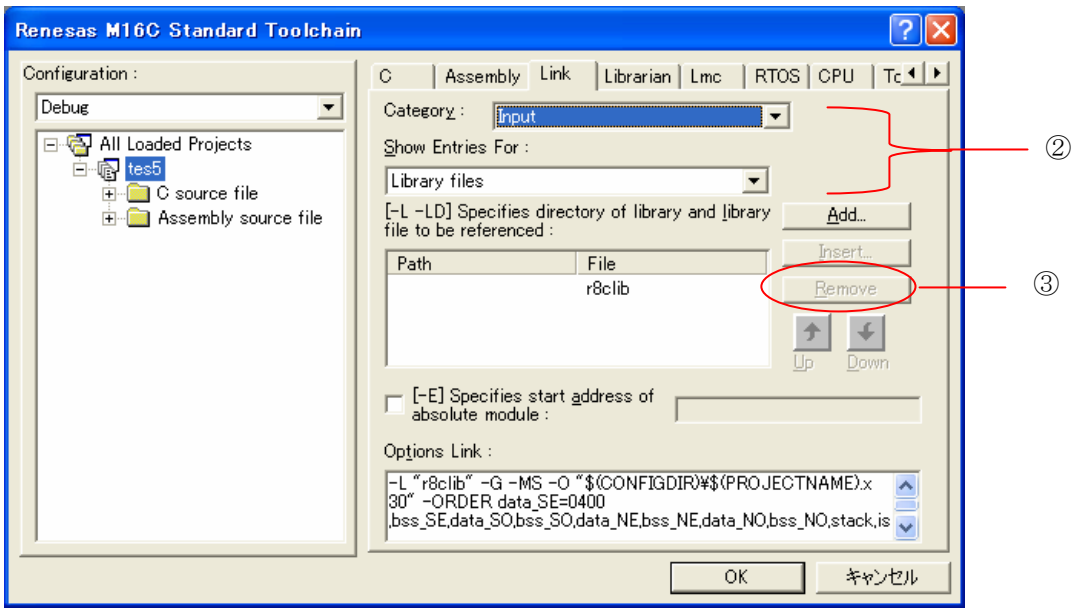
HEW の[ビルド]→[Renesas M16C Standard Toolchain...]→CPU タブを選択する。



①CPU Type:のプルタブメニューから“R8C/Tiny シリーズ(ROM がアドレス 0x10000 番地以降にある)に対応したコードにする“を選択してください。

リンクするライブラリを r8clib.lib から nc30lib.lib へ変更

HEW の[ビルド]→[Renesas M16C Standard Toolchain...]→Link タブを選択する。



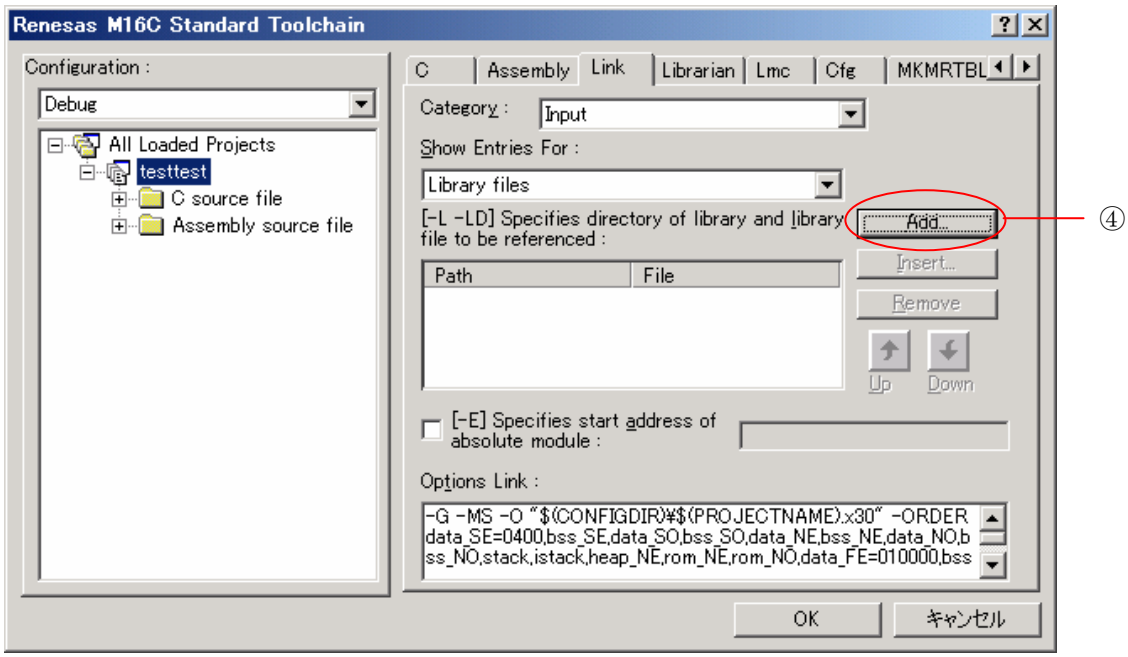
②

Category: Input

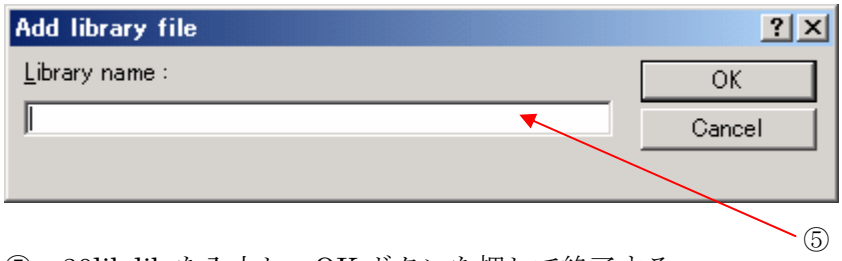
Show Entries For : Library files

を選択する

③Remove ボタンを押して一旦 r8clib.lib を削除する。



④Add ボタンを押して、[Library file]を選択する。



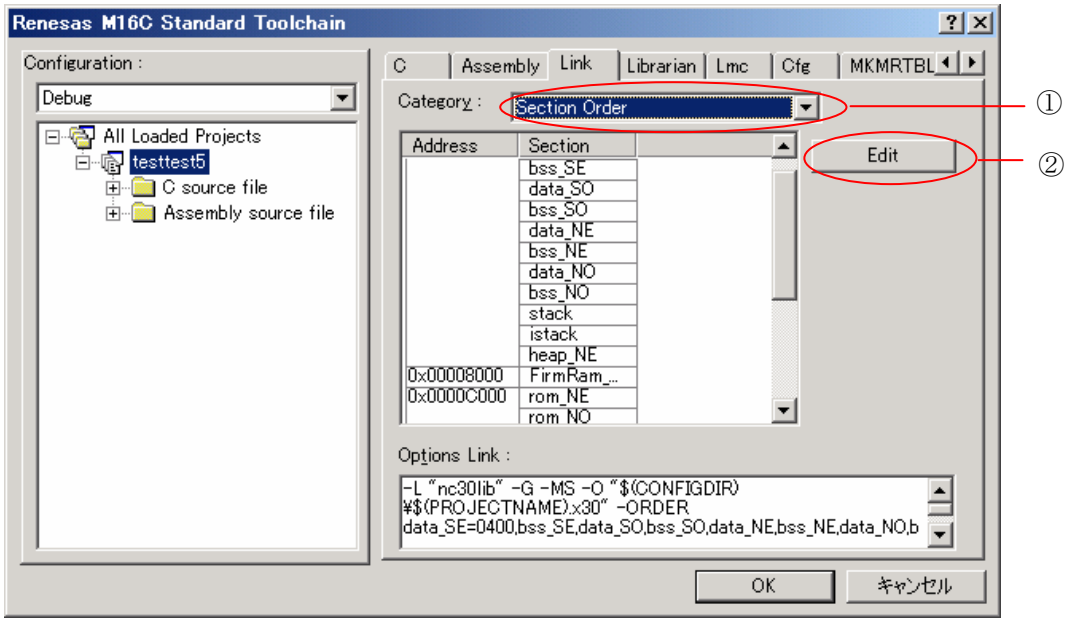
⑤nc30lib.lib を入力し、OK ボタンを押して終了する。

○R8C/Tiny 以外のマイコンを使用して新規ワークスペースを作成する場合

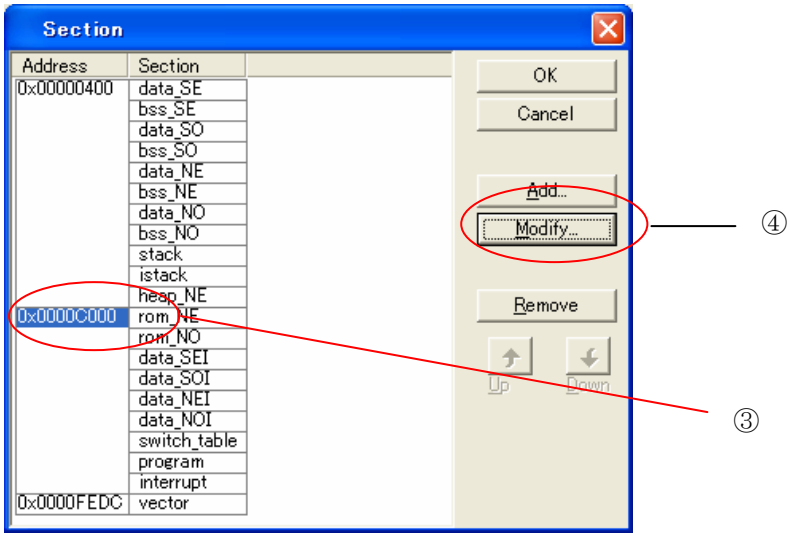
(1)CPU Series から該当する CPU シリーズを選択する。
(2)CPU Group から Other を選択する。

上記それぞれの処理以外に、以下の事に注意する必要があります。

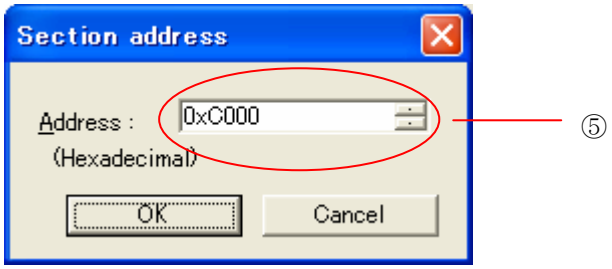
- ｓｆｒヘッダファイルは登録されません。
  使用されるマイコンに対応した sfr ファイルヘッダを WEB から入手するかもしくは、必要とする場合は、作成してください。
- セクションオーダーが正確ではありません。
  使用されるマイコンの ROM/RAM 空間に応じてセクションの各アドレスを変更してください。
- 可変ベクタ割り込みエントリ関数(intprg.c)が登録されません。



① Category:　Section Order を選択
② Edit ボタンを押す



③修正するアドレスを選択
④Modify ボタンを押す



⑤適切なアドレスへ変更する。

A.2.3 アセンブラ記述スタートアップを使用する場合

C 言語スタートアッププログラムを使用せずに、アセンブラ記述スタートアップ ncrt0.a30，sect30.inc,nc_define.h を使用してプロジェクトを作成する場合、下記に該当する場合には修正が必要です。

・割り込み関数宣言時にベクタ番号指定を行っていない。

修正内容

```
;---------------------------------------------------------------
; variable vector section
;---------------------------------------------------------------
        .section  vector,ROMDATA
        .org      __VECTOR_ADR__
.if   0      <----------------------------------     .if  0 を削除して.lword を有効にする。
        .lword    dummy_int              ; vector 0 (BRK)
        .lword    dummy_int              ; vector 1
        .........
        .lword    dummy_int              ; vector 63
.endif       <----------------------------------     .endif を削除する
```

B　TM→High-performance Embedded Workshop V.4 移行の手引き

本資料は、TM V.2.xx、V.3.xx で作成したプロジェクトを High-performance Embedded Workshop V.4 環境へ移行するための情報を説明します。
なお、本移行の手引きについては、今後 High-performance Embedded Workshop のバージョンアップ等に伴い、記載内容が異なる事が考えられます。
最新の情報につきましては、ルネサス開発環境 HomePage 内の FAQ サイトをご覧ください。

B.1　概要

TM V.2.xx、V.3.xx で作成したプロジェクトを High-performance Embedded Workshop V.4 環境へ移行するには、High-performance Embedded Workshop の Import Makefile 機能を使用します。「Import Makefile」は、指定された makefile に書かれているソースファイルやオプション情報からプロジェクトを作成する機能です。
TM のプロジェクトファイルは、GNU make から実行可能である makefile フォーマットで作成されています。「Import Makefile」にて、TM のプロジェクトファイルを makefile として指定することで、TM のプロジェクトを High-performance Embedded Workshop のプロジェクトへ変換することができます。「Import Makefile」では、TM のプロジェクトファイル以外に hmake、nmake、gmake 用の makefile フォーマットのファイルを High-performance Embedded Workshop のプロジェクトに変換することができます。

B.2　変換手順

以下に、TM のプロジェクトを High-performance Embedded Workshop のプロジェクトへ移行する手順を示します。

1. メニュー［ファイル］→［新規ワークスペース］をクリックします。
2. 新規プロジェクトワークスペースダイアログボックスが表示されます。

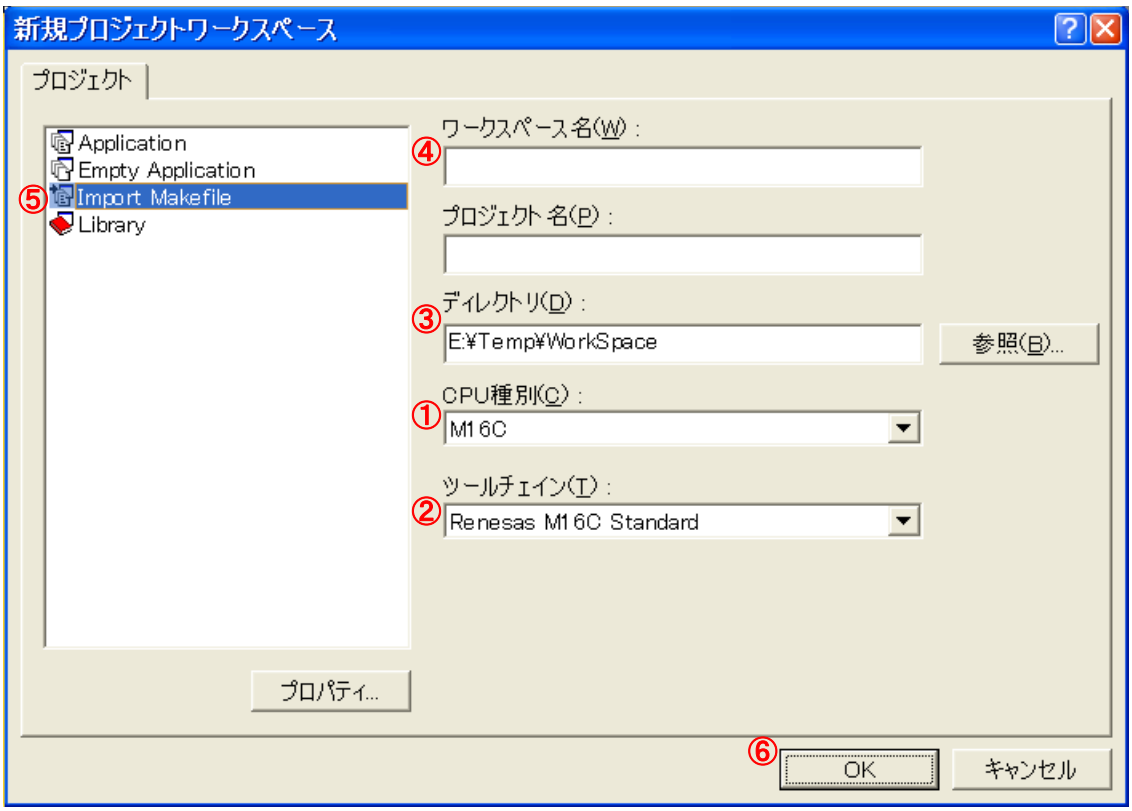


図 1　新規プロジェクトワークスペース

- CPU 種別を選択します。TM のプロジェクトで使用していた CPU 種別を選択してください。

- ツールチェインを選択します。ツールチェイン名とクロスツール名の対応は以下の通りです。TM のプロジェクトで使用していたツールチェイン（クロスツール）を選択してください。

表 1　ツールチェイン名とクロスツール名

ツールチェイン名	クロスツール名
Renesas M16C Standard	NC30WA
Renesas R8C Standard	NC8C
Renesas M32C Standard	NC308WA
Renesas M32R Standard	CC32R

- プロジェクトタイプから Import Makefile を選択します。
- ディレクトリを指定します。
- ワークスペース名を指定します。ワークスペース名を指定すると自動的に（ワークスペースと同名の）プロジェクト名が指定されます。
- OK ボタンをクリックします。

3. New Project-1/4-Import Makefile ウィザードダイアログボックスが表示されます。

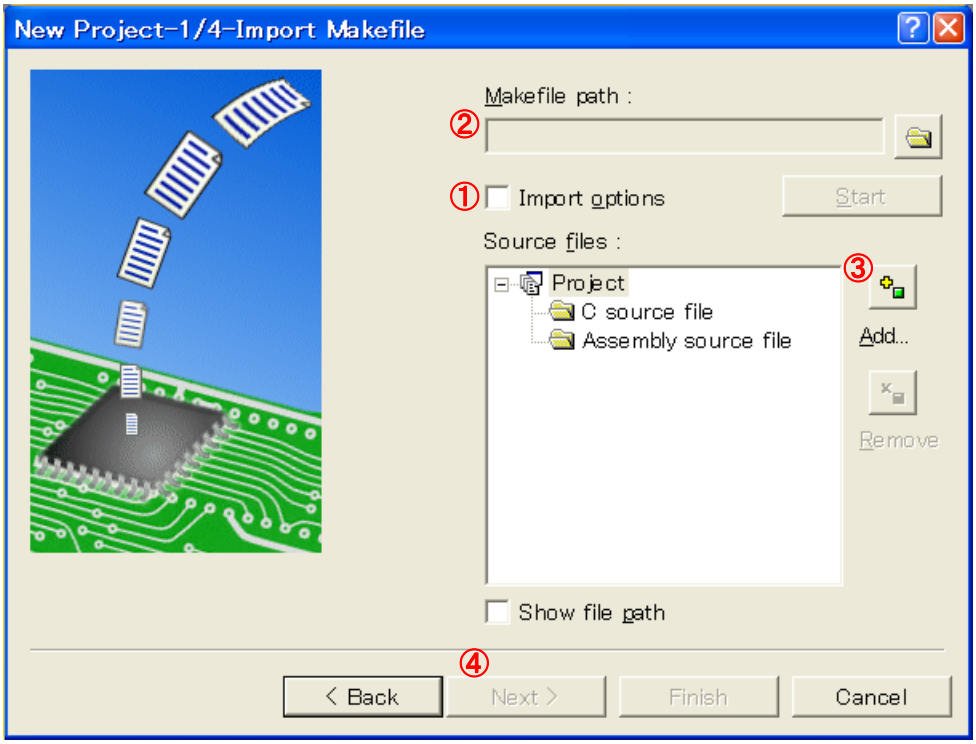


図 2　New Project-1/4-Import Makefile

- 「Import options」をチェックします。
  この項目をチェックすると、ビルド（コンパイラ、アセンブラ etc）オプション情報が High-performance Embedded Workshop プロジェクトへ移行されます。チェックをはずすと、オプション情報は無視されます（High-performance Embedded Workshop プロジェクトへ移行されません）。
- Makefile path に TM のプロジェクトファイル（拡張子が tmk）を指定します。
  Makefile path にファイルが指定されるとすぐに指定ファイルの解析作業が行われます。解析が終了すると解析したソースファイルが Source files ツリーに表示されます。
  「Start」ボタンはクリックすると、再度、指定ファイルの解析作業が行われます。
- 解析結果（Source files ツリー）に誤りがある場合は、Add...、Remove ボタンから Source files ツリーを編集してください。
- Next ボタンをクリックします。

4. 以降はダイアログボックスの指示に従って作業を進めてください。

B.3　注意事項

B.3.1 移行できる情報、できない情報

TM のプロジェクトを High-performance Embedded Workshop 環境へ移行する場合、TM のプロジェクトの全構成を移行できるわけではありません。移行できる情報は、以下の通りです。

- アセンブラソースファイルパス
- C 言語ソースファイルパス
- アセンブラオプション
- C コンパイラオプション
- リンカオプション（リンク順序を除く）

その他の情報は High-performance Embedded Workshop 環境へ移行することができません。移行できない情報は、「Import Makefile」の処理終了後、これ以降に示す注意事項の通りに High-performance Embedded Workshop プロジェクトを編集してください。

B.3.2 クロスツール

「Import Makefile」では、クロスツールのバージョンを High-performance Embedded Workshop プロジェクトへ移行することができません。よって、TM のプロジェクトで使用していたクロスツールのバージョンにかかわらず、High-performance Embedded Workshop プロジェクト移行後に使用可能なクロスツールのバージョンは以下のものになります。

NC30WA	:	V.5.20 Release1 およびそれ以降
NC8C	:	V.5.30 Release1 およびそれ以降
NC308WA	:	V.5.20 Release1 およびそれ以降
CC32R	:	V.4.20 Release1 およびそれ以降

B.3.3 High-performance Embedded Workshopのバージョン

TM のプロジェクトを High-performance Embedded Workshop 環境へ移行する場合、移行先の High-performance Embedded Workshop のバージョンにより移行できる情報が異なります。High-performance Embedded Workshop のバージョンによる移行可能情報は以下の通りです。

表 2　High-performance Embedded Workshop バージョン毎の移行可能情報

		High-performance Embedded Workshop				
		~V.3.01.02	V.3.01.04	V.3.01.05	V.3.01.06	V.4.00
NC30WA	V.5.20 Release1	△	△	△	△	○
	V.5.30 Release1	△	△	△	△	○
	V.5.30 Release 02	---	---	---	---	○
NC8C	V.5.30 Release1	△	△	△	△	△
NC308WA	V.5.20 Release1	△	△	△	△	△
CC32R	V.4.20 Release1	△	△	△	▲	▲
	V.4.20 Release1A	△	△	△	▲	▲
	V.4.30 Release 00(A)	▲	▲	▲	▲	▲

○：アセンブラソースファイル、C 言語ソースファイルおよびアセンブラ、C コンパイラ、リンカのオプション移行可能

▲：アセンブラソースファイル、C 言語ソースファイルおよびアセンブラ、C コンパイラのオプション移行可能

△：アセンブラソースファイル、C 言語ソースファイルのみ移行可能

High-performance Embedded Workshop4 がバンドルされたコンパイラ製品から順次○になります。

B.3.4 ロードモジュールコンバータ

「Import Makefile」では、ロードモジュールコンバータの情報（コマンド実行、オプション情報）を High-performance Embedded Workshop プロジェクトへ移行することができません。TM のプロジェクトでロードモジュールコンバータを使用していた場合は、「Import Makefile」の処理終了後、以下の手順でロードモジュールコンバータの設定を行ってください。

1. メニュー［ビルド］→［ビルドフェーズ］をクリックします。
2. ビルドフェーズダイアログボックスが表示されます。

図 3　ビルドフェーズダイアログボックス

- Mxxx Load Module Converter をチェックします。
- OK ボタンをクリックします。

3. メニュー［ビルド］→［Renesas Mxxx Standard Toolchain…］をクリックします。

4. Renesas Mxxx Standard Toolchain ダイアログボックスが表示されます。

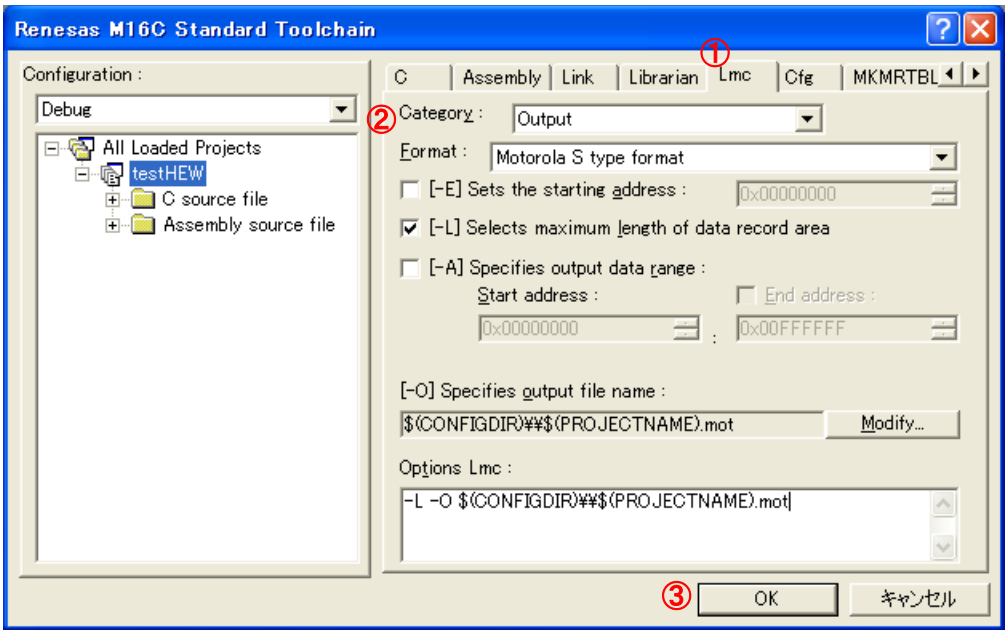


図 4　Renesas Mxxx Standard Toolchain ダイアログボックス

- Lmc タブをクリックします。
- Category を変更してオプションを指定します。
- OK ボタンをクリックします。

B.3.5 外部ツール

「Import Makefile」では、アセンブラ、C コンパイラ、リンカ以外のツールの情報（コマンド実行、オプション情報、依存関係）を High-performance Embedded Workshop プロジェクトへ移行することができません。TM のプロジェクトで、アセンブラ、C コンパイラ、リンカおよびロードモジュールコンバータ以外のツールを使用していた場合は、High-performance Embedded Workshop のカスタムビルドフェーズを作成していただく必要があります。カスタムビルドフェーズは、標準のビルド実行（アセンブラ、C コンパイラ、リンカ）の前後または途中に外部ツールを実行するための独自のビルドフェーズです。
カスタムビルドフェーズ作成手順についての詳細は、High-performance Embedded Workshop4 ユーザーズマニュアル「3.2 カスタムビルドフェーズを作成する」をご覧ください。
ここでは、例としてクロスツールにバンドルされている xrf30 を登録する方法を示します。

1. メニュー［ビルド］→［ビルドフェーズ］をクリックします。
2. ビルドフェーズダイアログボックスが表示されます。追加ボタンをクリックします。

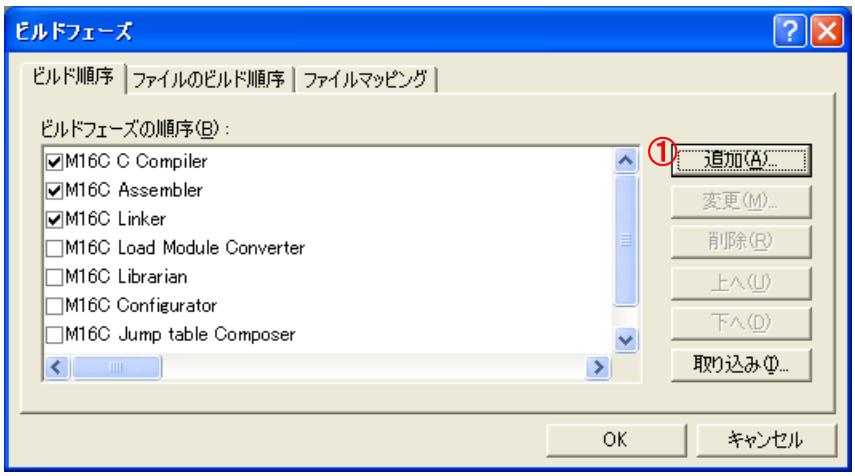


図 5　ビルドフェーズダイアログボックス

3.　新規ビルドフェーズダイアログボックスが表示されます。ウィザードに従ってツールを登録します。

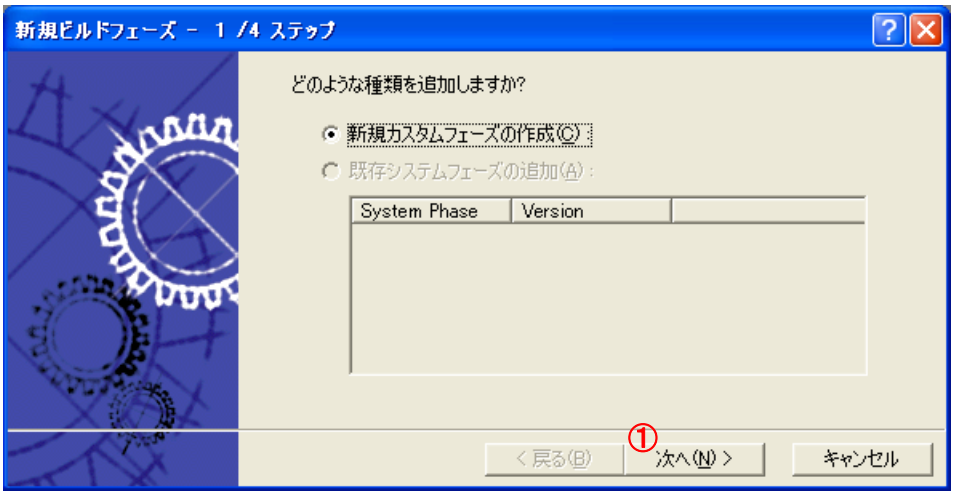


図 6　新規ビルドフェーズ-1/4 ステップ

- ［1/4 ステップ］次へボタンをクリックします。

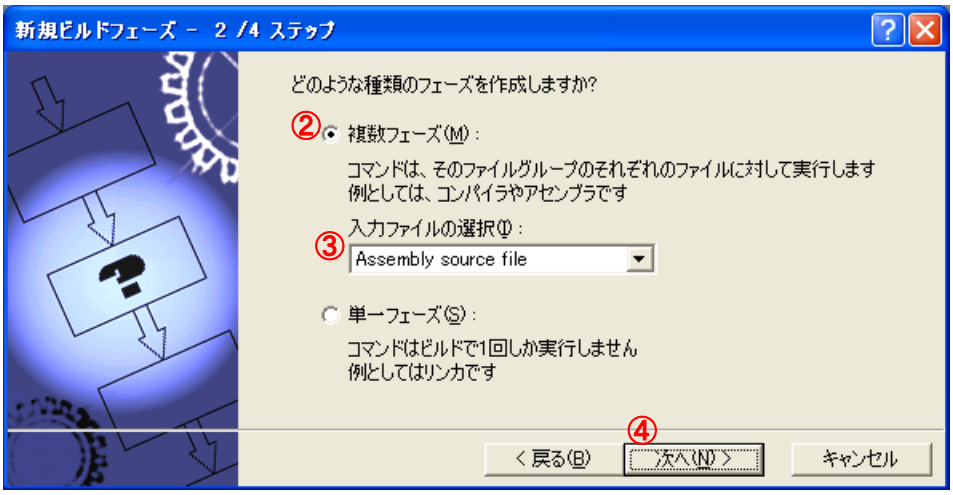


図 7　新規ビルドフェーズ-2/4 ステップ

- ［2/4 ステップ］複数フェーズを選択します。
- ［2/4 ステップ］入力ファイルの選択から Assembly source file を選択します。
- ［2/4 ステップ］次へボタンをクリックします。

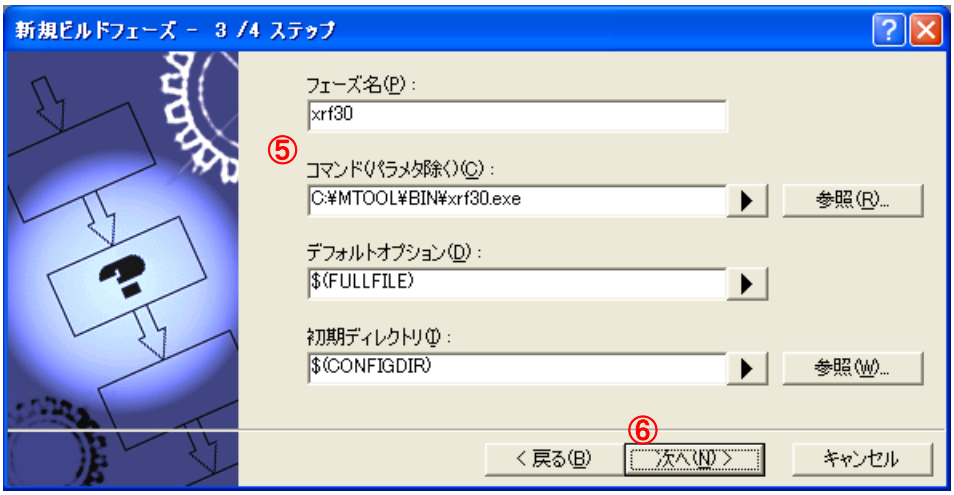


図 8　新規ビルドフェーズ-3/4 ステップ

- ［3/4 ステップ］フェーズ名に xrf30 を、コマンドに xrf30 のフルパスを指定します。
- ［3/4 ステップ］次へボタンをクリックします。

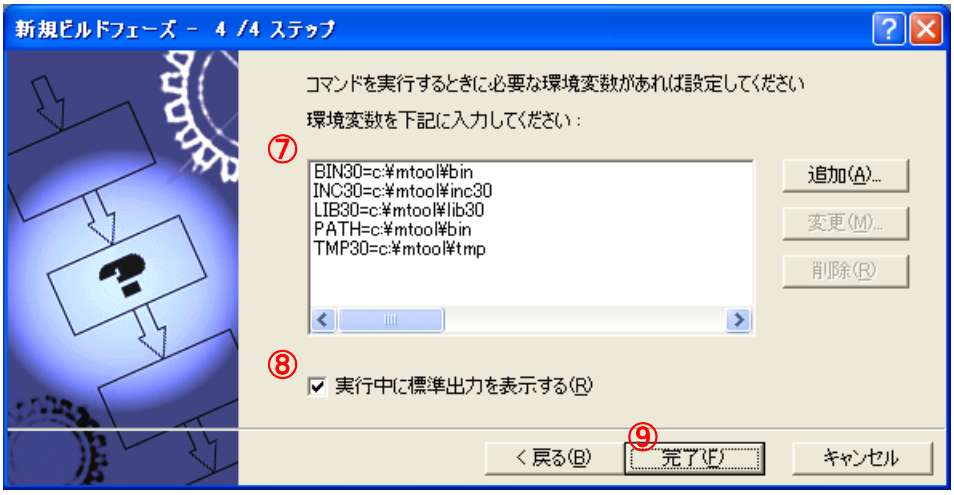


図 9　新規ビルドフェーズ-4/4 ステップ

- ［4/4 ステップ］環境変数を指定します。
- ［4/4 ステップ］実行中に標準出力を表示するをチェックします。
- ［4/4 ステップ］完了ボタンをクリックします。

4.　ビルドフェーズダイアログボックスに戻ります。
　登録したビルドフェーズ（xrf30）がリストの最後に追加されます。

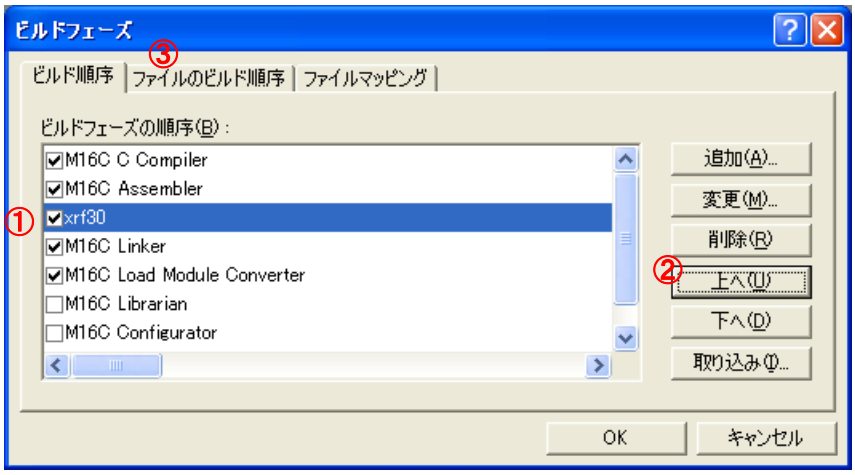


図 10　ビルドフェーズダイアログボックス

- xrf30 を選択します。
- 上へボタンクリックします。Assembler の下まで xrf30 を移動します。
- ファイルのビルド順序タブをクリックします。

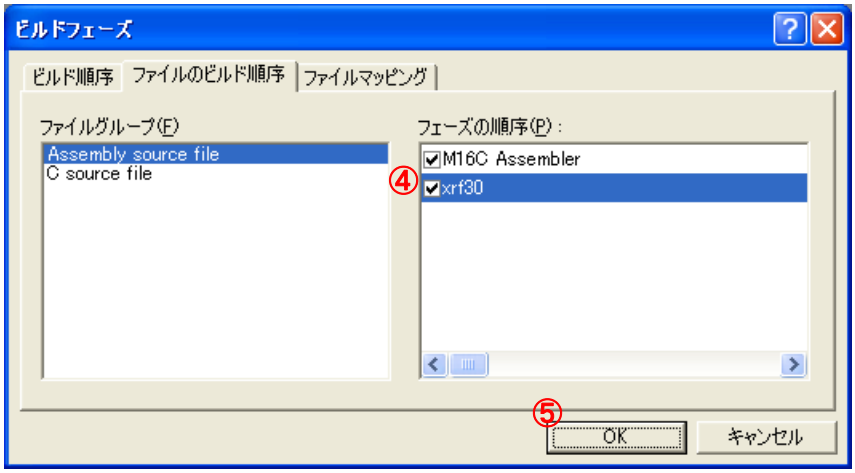


図 11　ビルドフェーズダイアログボックス

- xrf30 をチェックします。
- OK ボタンをクリックします。

5.　メニュー［ビルド］→［xrf30］をクリックします。

6.　xrf30 Options ダイアログボックスが表示されますので、必要に応じてオプションなどを設定してください。この設定を行うと、ビルド時のアセンブラ実行後（リンカの実行前）に xrf30 がすべてのアセンブラソースファイルに対し実行されるようになります。

B.3.6 リンク順序

「Import Makefile」では、リンク順序の情報を High-performance Embedded Workshop プロジェクトへ移行することができません。リンク順序は、アルファベット順となります。リンク順序を変更する場合は、以下の手順で設定してください。

本機能は、High-performance Embedded Workshop4 がバンドルされたコンパイラ製品からの対応となります。

1. メニュー［ビルド］→［リンク順の指定］をクリックします。
2. リンク順序のカスタマイズダイアログボックスが表示されます。

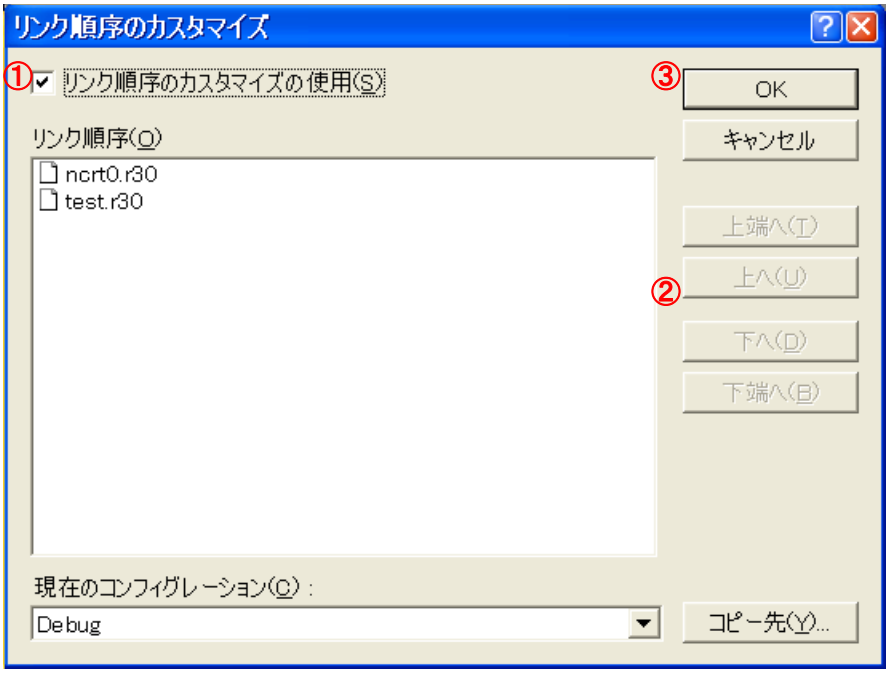


図 12　リンク順序のカスタマイズダイアログボックス

- リンク順序のカスタマイズの使用をチェックします。
- リンク順序リストからファイルを選択して、上へまたは下へボタンをクリックします。
- OK ボタンをクリックします。

B.3.7 スタートアッププログラムの先頭リンク

「Import Makefile」では、リンク順序の情報を High-performance Embedded Workshop プロジェクトへ移行することができません。リンク順序は、アルファベット順となります。そのためスタートアップが先頭にリンクされないことがあります。スタートアッププログラムを先頭にリンクするには、「C.3.6 リンク順序」の項の設定を行ってください。

「C.3.6 リンク順序」に対応していないバージョンのコンパイラは、以下の手順で設定してください。

- 1. メニュー「ビルド」→「Renesas XXX Standard Toolchain...」をクリックします。("Renesas XXX Standard Toolchain"ダイアログボックスが開きます)
- 2. 「link」タブをクリックします。
- 3. [Category:]~、「Input」を選択します。
- 4. 「Show entries for:」から、「Relocatable files」を選択します。
- 5. 「Add」ボタンをクリックします。(「Add Relocatable files」ダイアログボックスが開きます)
- 6. 「Relative to:」から、「Configuration directory」を選択します。
- 7. 「File path:」にスタートアッププログラム名（例：NC30WA の場合、ncrt0.r30）を入力します。
- 8. 「Add Relocatable files」ダイアログボックスの[OK]ボタンをクリックします。
- 9. 「Renesas XXX Standard Toolchain」ダイアログの[OK]ボタンをクリックします。