２００９年　４月２４日
ＢＹ　エーワン　Ｒｅｖ１．１４

ルネサス純正Ｃ言語の統合環境ＨＥＷ（Ｖｅｒ３．０１）で H-debugger 用に新ワークスペースおよび新プロジェクト名を登録する方法（ＳＨ－２版）

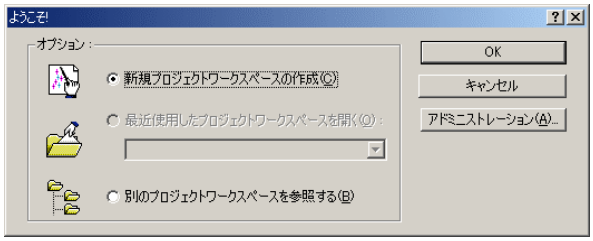
ルネサスＣ言語用統合環境“ＨＥＷ　Ｖｅｒ３．０１（ｒｅｌｅａｓｅ１）”で H-debugger 用に新ワークスペース／プロジェクト名を登録する手順方法を説明します。

説明を明確にするために、名前等を仮に決めて例に沿って説明を進めます。

ワークスペース名：	NewWorkSpace3		
プロジェクト名　：	Project3		
登録モジュール名：	Vector55.asm	ＡＳＭファイル	割り込みベクタテーブル
	Start55.c	Ｃファイル	スタートアップ
	Sh7055.c	Ｃファイル	メインモジュール
ＣＰＵタイプ　：	Sh7055		

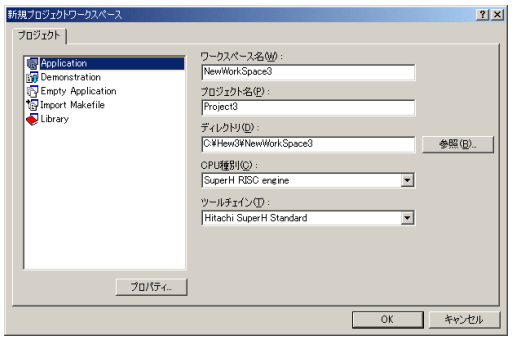
１．新ワークスペースの登録方法

“HEW”起動させます。



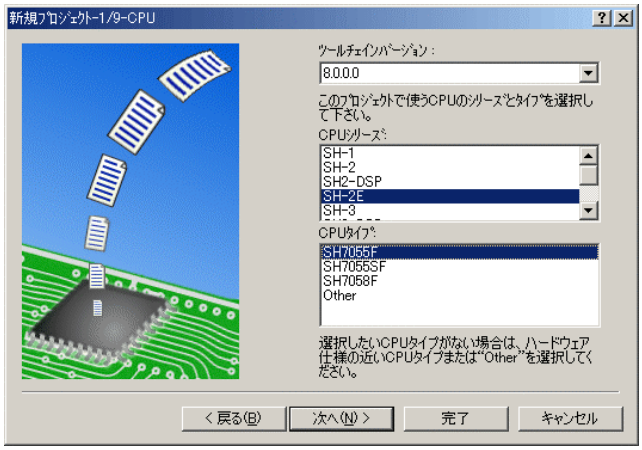
“新規プロジェクトワークスペース”をチェックしてのＯＫをクリックする。

もしくは、キャンセル後に、[ファイル]-[新規ワークスペース]をクリックします。



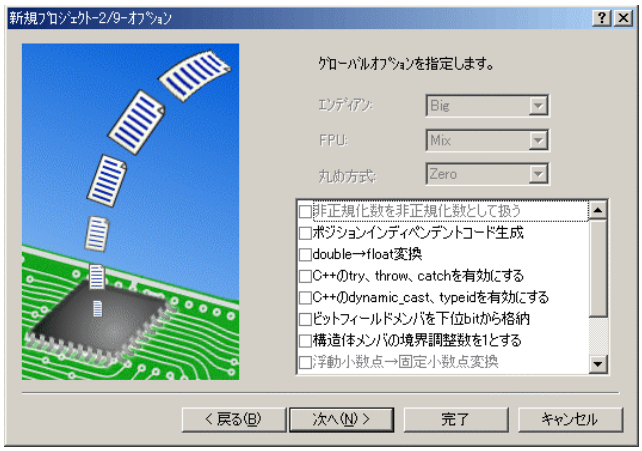
ﾜｰｸｽﾍﾟｰｽ名	“NewWorkSpace3”
ﾌﾟﾛｼﾞｪｸﾄ名	“Project3”
ﾃﾞｨﾚｸﾄﾘ	“C:¥HEW3¥NewWorkSpace3”
CPU 種別	“SuperH RISC engine”
ﾂｰﾙﾁｪｲﾝ	“Hitachi SuperH Standard”
ﾌﾟﾛｼﾞｪｸﾄ	Application

この項目を確認後、ＯＫをクリックして下さい。

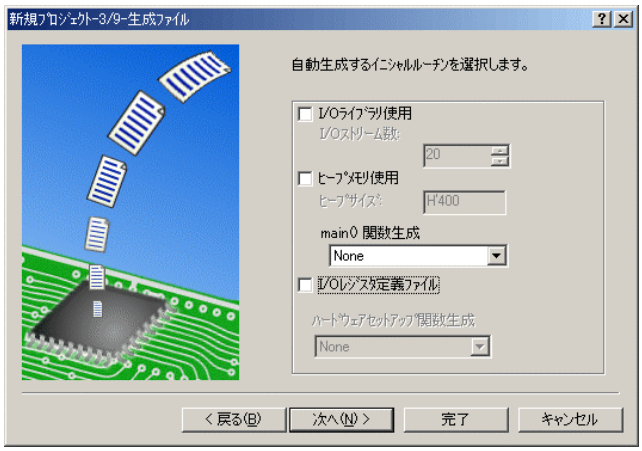


CPU ｼﾘｰｽﾞ を“SH-2E”に選択する。
CPU ﾀｲﾌﾟ を“SH7055F”に選択する。

確認後、次へ>をクリックします。



ＣＰＵスペックを確認後、次へ>をクリックして下さい。



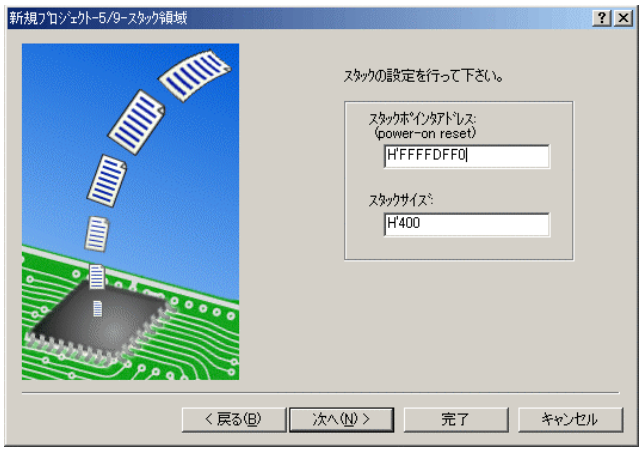
この例では、ＨＥＷの用意したＲＯＭ化支援ソフトは、一切使用しませんので全てのチェックを外して下さい。

main() 関数生成を”None”に選択する。

確認後、次へ>クリックして下さい。

C言語ライブラリの選択です。この例では、その他ライブラリを使用しません。

このまま、次へ>クリックして下さい。

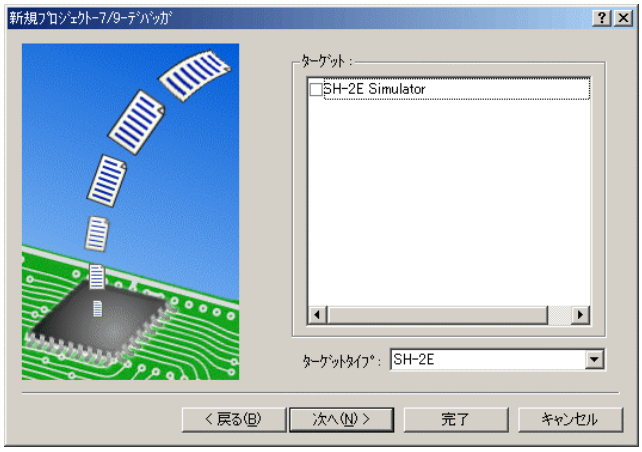


スタックボトムの設定ですが、この情報も使用しません。

このまま、次へ>をクリックして下さい。

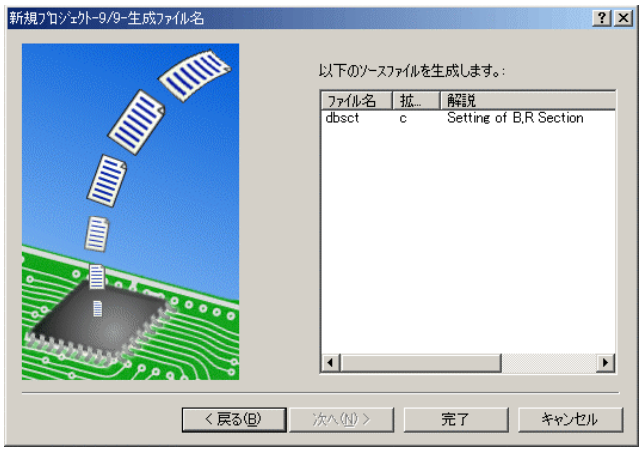
ベクターテーブル設定ファイルの設定ですが、この情報も使用しませんのでチェックを外してください。

確認後、次へ>をクリックして下さい。



シミュレータの設定ですが、この情報も使用しません。

このまま、次へ>クリックして下さい。



ここで最終になります。

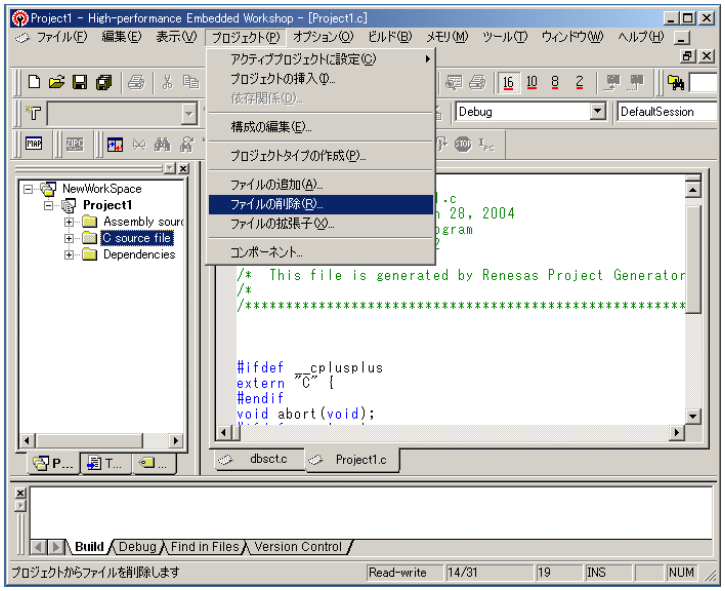
この画面での説明はセグメントを定義するためのＣソースを表示していますが、このソースも使用しませんので、後でプロジェクトからＲｅｍｏｖｅします。

この状態で完了をクリックして下さい。

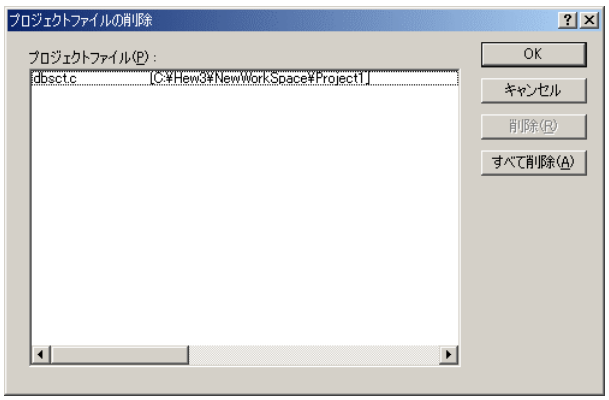
確認画面が表示されますので、OKをクリックして終了です。

次にプロジェクト名”Project3”に希望モジュール（ソースファイル）を登録する前に、不用ソースファイルを削除する方法を説明します。

２．不用なソースファイルを削除する方法



[プロジェクト]-[ファイルの削除]をクリックします。



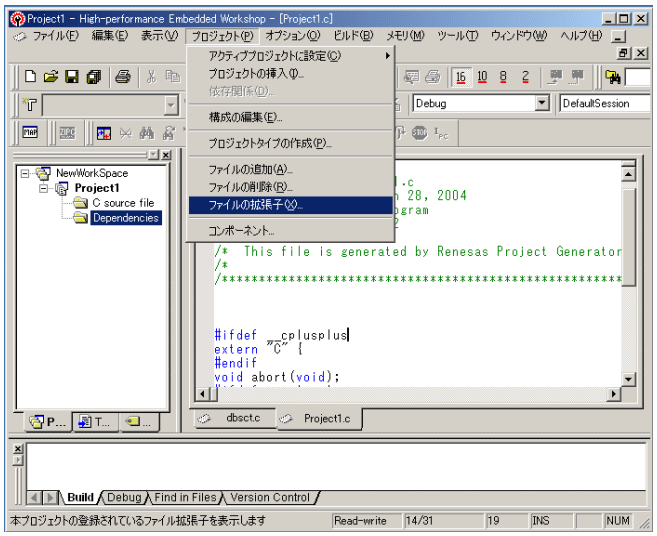
”dbsct.c”をクリックして選択すると”削除”ボタンが使用できるようになります。

削除のクリックで、”dbsct.c”を削除することが出来ます。

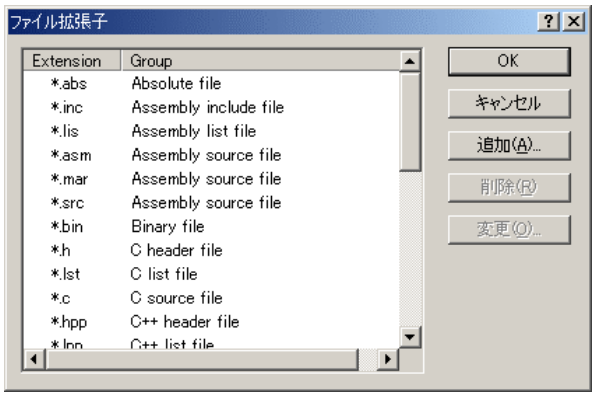
削除したらOKで終了です。

次にプロジェクト名”Poject3”に希望モジュール（ソースファイル）を登録するために、ＨＥＷに拡張子を登録する方法を説明します。

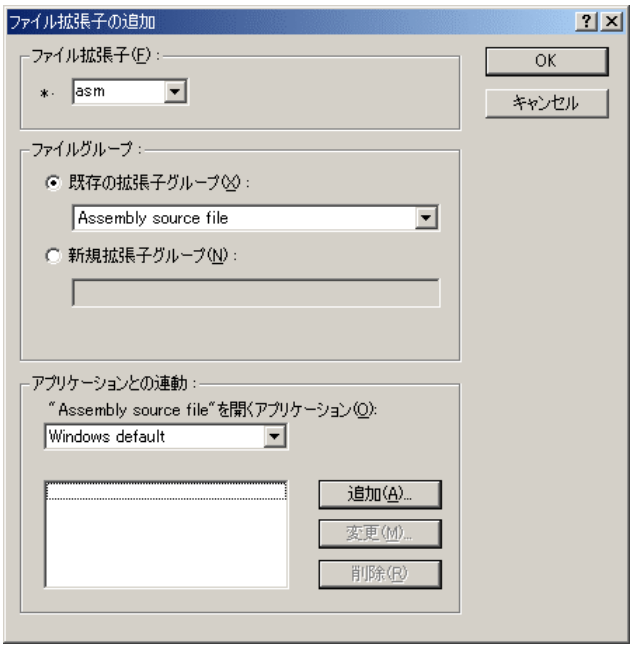
３．拡張子を登録する方法



［プロジェクト］-［ファイルの拡張子］をクリックします。



追加をクリックします。



ファイル拡張子に”asm”を入力する。

既存の拡張子グループ”Assembly source file”を選択する。

OKをクリックします。

この操作により、プロジェクトに拡張子“＊．ａｓｍ”の登録が完了しました。

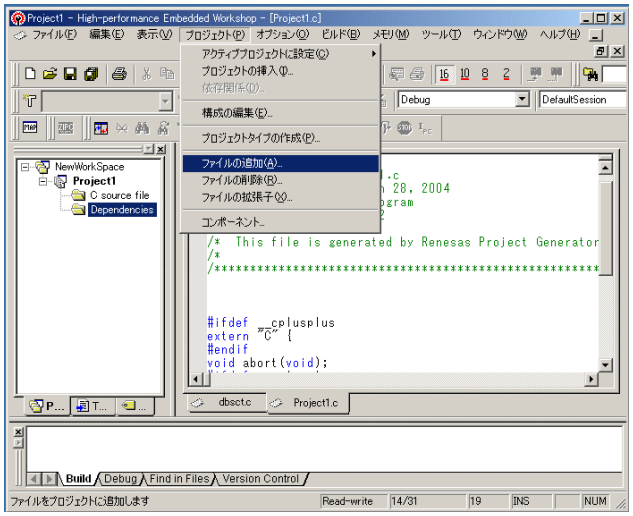
次にプロジェクト名”Project3”に希望モジュール（ソースファイル）を登録します。

４．プロジェクトに希望モジュール（ソースファイル）を登録する方法

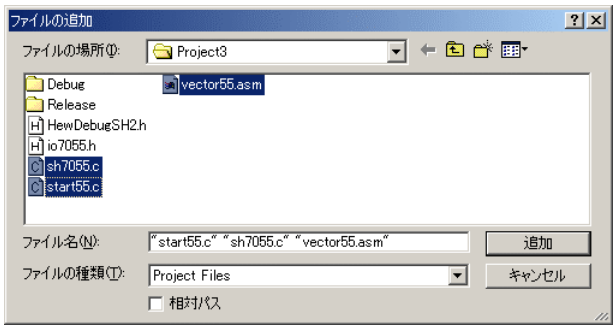
準備：　作成済みのソースファイルを”C:¥Hew3¥NewWorkSpace3¥Project3”にコピーして下さい。

Vector55.asm
Start55.c
Sh7055.c
Io7055.h
HewDebugSH2.h

”¥sample¥HewSH22_0¥SH7055¥”
(左記ソースは、製品 CD の上記フォルダに入っています。)



[プロジェクト]-[ファイルの追加]をクリックします。



下記３ファイルを指定して下さい。

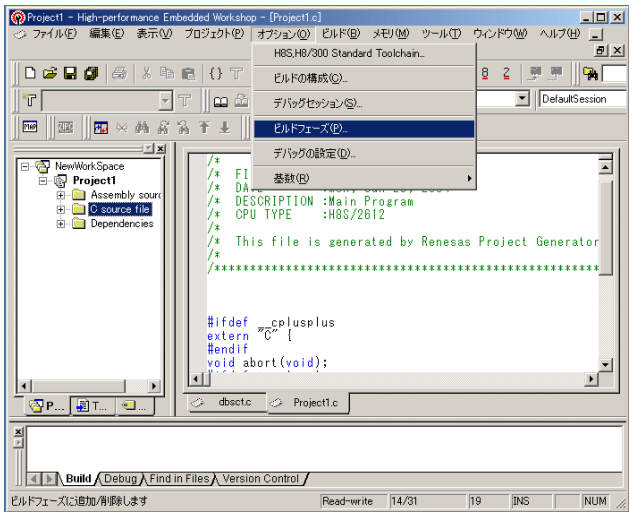
Vector55.asm
Start55.c
Sh7055.c

選択後、追加をクリックします。

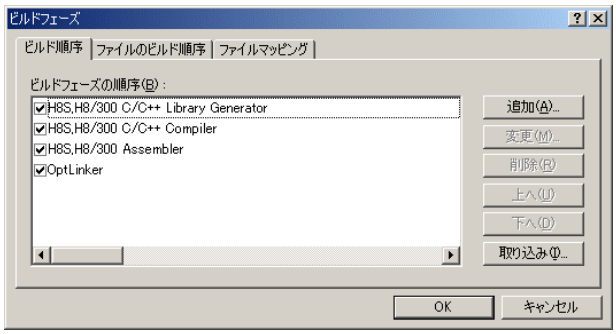
この操作により、プロジェクトにモジュールの登録が完了しました。

次に、ツール関係の設定を説明します。

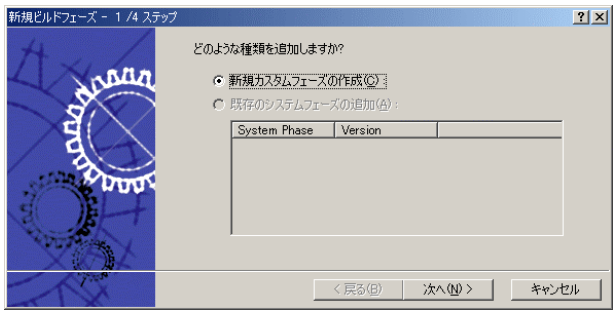
５．シンボルコンバータ「ＨＣｓｙｍｃｏｎｖ」を登録する。



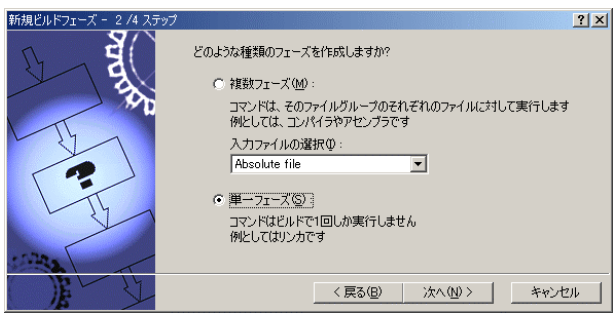
[オプション]-[ビルドフェーズ]をクリックします。



追加をクリックします。

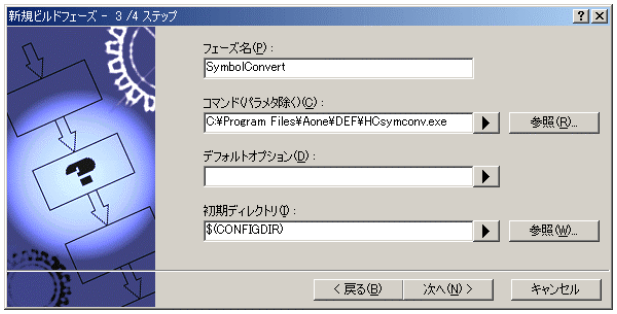


次へ>をクリックします。



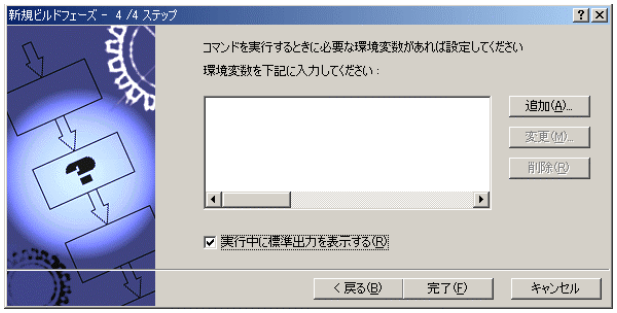
単一フェーズ側にチェックをします。

次へ>をクリックします。

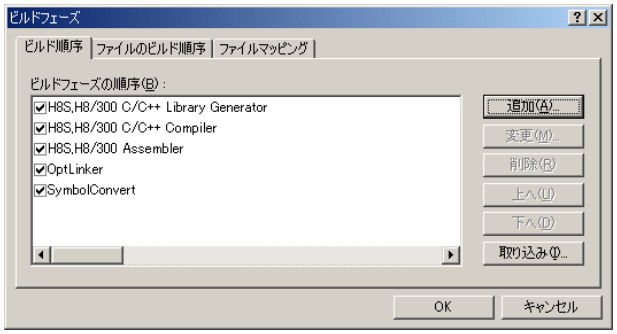


フェーズ：SymbolConvert
コマンド：C:¥ProgramFiles¥Aone¥DEF¥HCsymconv.exe を選択する。
（デフォルト）
初期ディレクトリ：$(CONFIGDIR)

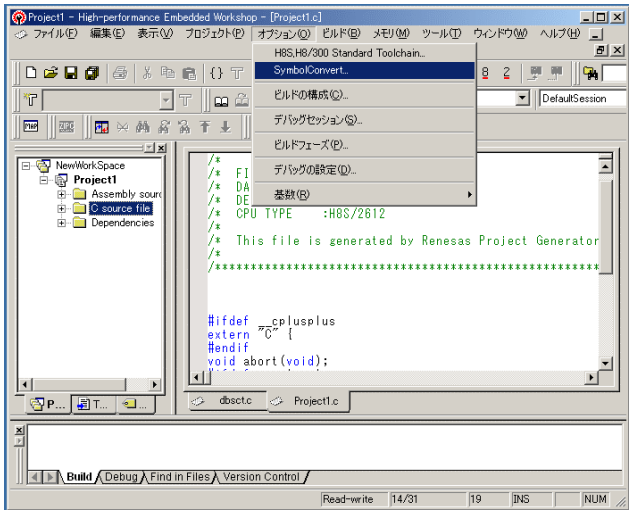
次へ>をクリックします。



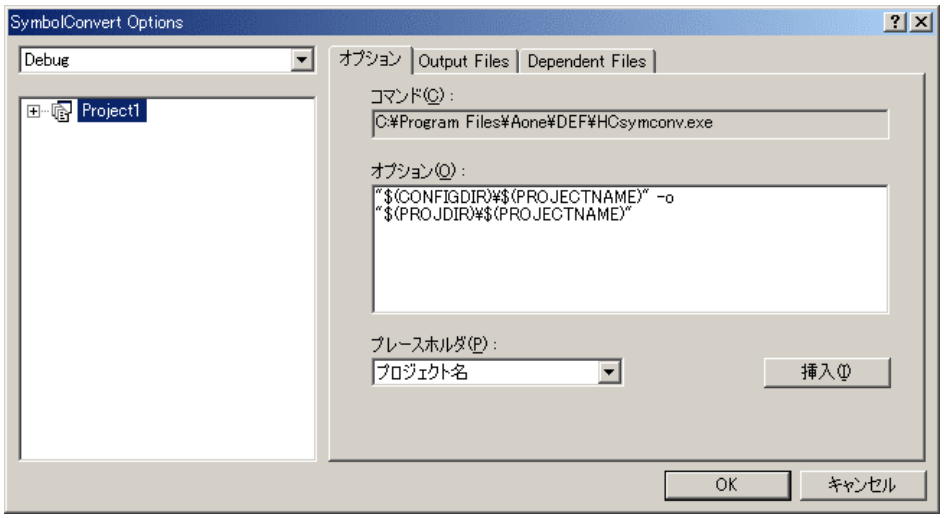
完了をクリックします。



OKをクリックします。



[オプション]-[ＳｙｍｂｏｌＣｏｎｖｅｒｔ]をクリックします。



オプションに下記内容を設定する。

"$(CONFIGDIR)¥$(PROJECTNAME)" -o "$(PROJDIR)¥$(PROJECTNAME)"

（入力ファイル名）　　（出力先名）

注意事項

1）ディレクトリ名に‘　’スペースを使用している場合は、“ダブルクォートで囲んで下さい。
“$(CONFIGDIR)¥$(PROJECTNAME)”-o “$(PROJDIR)¥$(PROJECTNAME)”

2）$(PROJECTNAME)の先頭に「¥」記号を挿入して下さい。（手入力）

3）オプションＳＷ「－ｏ」の両端には、スペースを入れてください。（手入力）

4）Ｃソースファイルの有る場所にシンボルコンバート後のファイルを置くのが目的ですので必ず指定して下さい。

追加事項（ＨＣｓｙｍｃｏｎｖ．ｅｘｅ　スイッチ説明）

1）［－ｏ］（省略可）出力ファイル名を指定

2）［－ｒ］（省略可）モジュール毎のディレクトリ情報を作成しない。ELF 専用(Ｖｅｒ３．２ｘｘから)

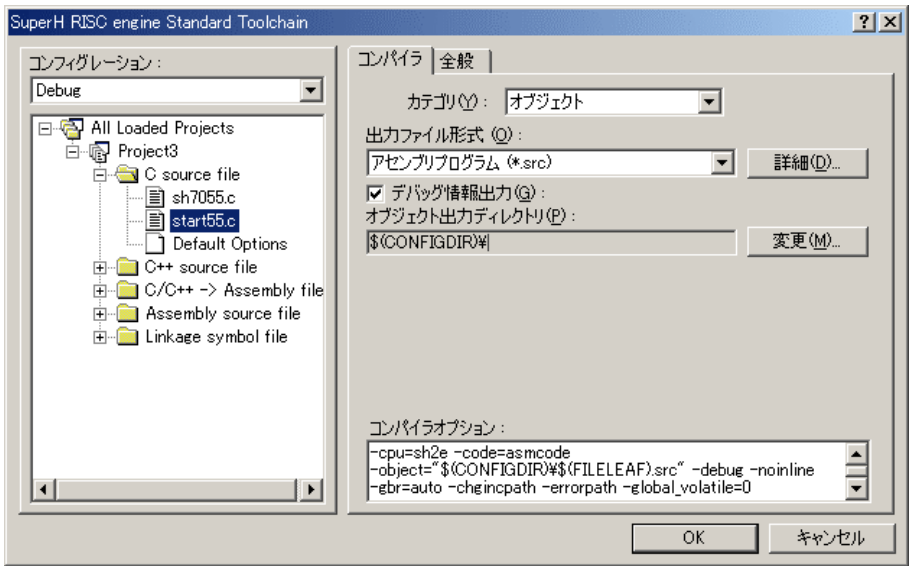
3）［－ｓ］（省略可）ラインシンボル情報をソート（アドレス順）しない。(Ｖｅｒ３．２ｘｘから)

4）［－ｉ］（省略可）不整合なＩｎｌｉｎｅ情報を削除する。(Ｖｅｒ３．３ｘｘから)

5）［－ｇ］（省略可）スタティック変数をグローバル化する。(Ｖｅｒ３．６ｘｘから)

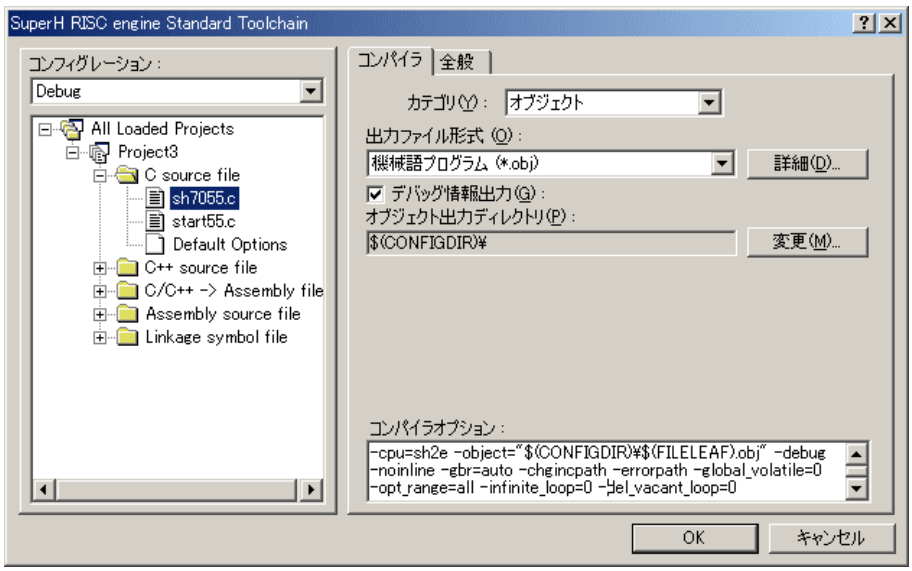
６．Cコンパイラとアセンブラの設定

Cソース内に「#pragma asm」のインラインアセンブラ記述がある場合



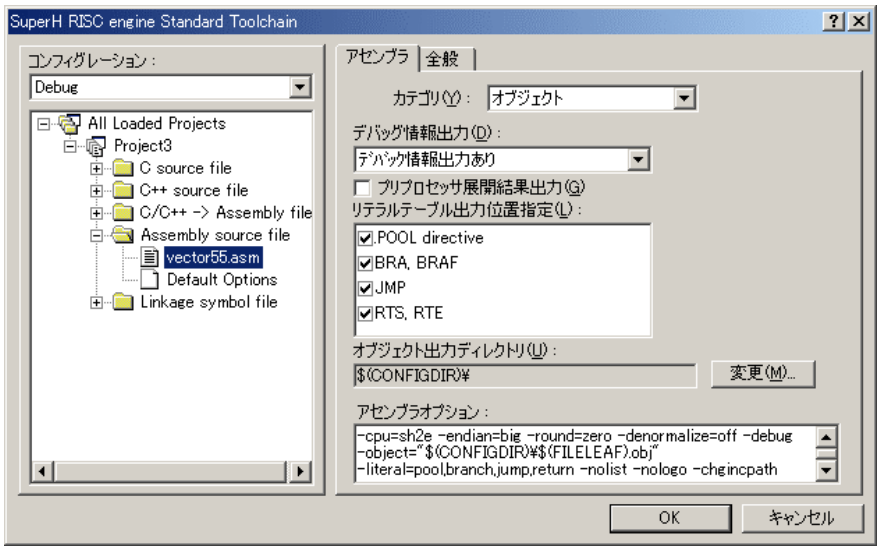
・カテゴリ「**オブジェクト**」の出力ファイル形式を「**アセンブリプログラム(*.src)**」に設定する。
・この設定にしますとシンボル情報のタイプは全てラベルになります。

全てC言語記述の場合



・カテゴリ「**オブジェクト**」の出力ファイル形式を「**機械語プログラム(*.obj)**」に設定する。
・この設定にしますと詳細なシンボル情報を得ることなります。

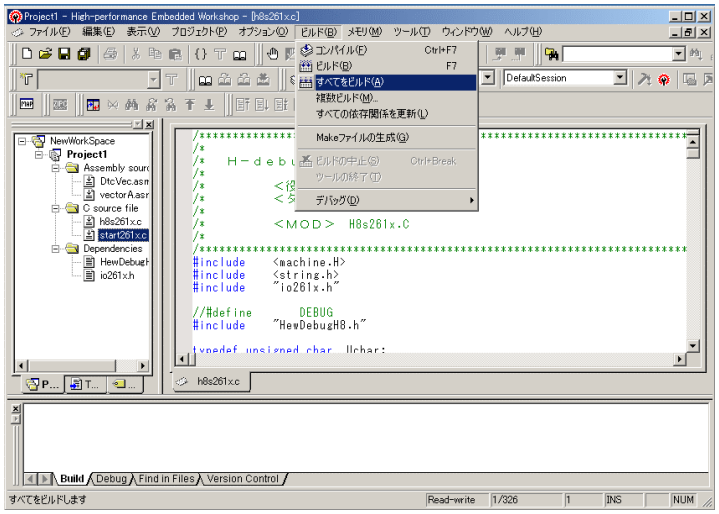
全てアセンブラ記述の場合



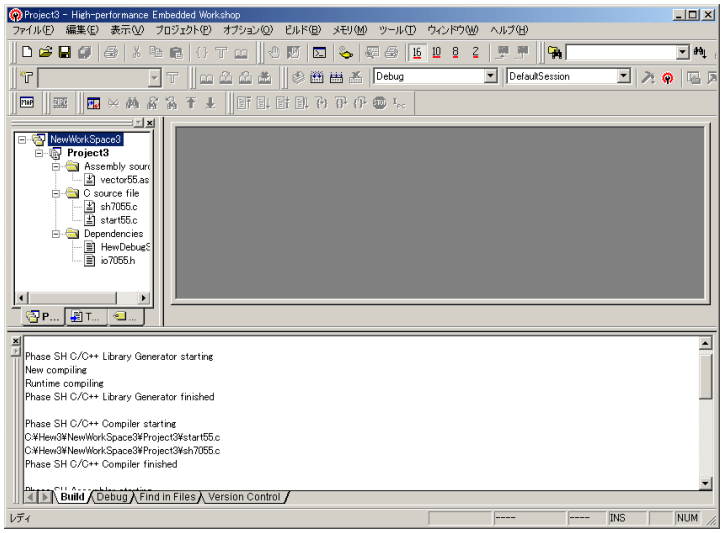
・カテゴリ**「オブジェクト」**のデバッグ情報を**「デバッグ情報出力」**に設定する。

7．ツール（ライブラリ）の設定

HEWは、プロジェクトごとにC言語用ライブラリを作成する仕様になっています。
ここで「すべてをビルド」してライブラリを作成および設定の確認をします。

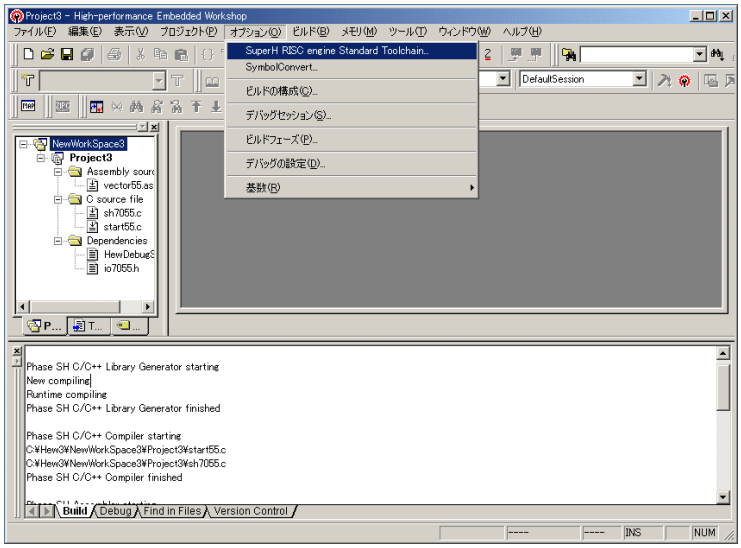


[ビルド]-[すべてをビルド]をクリックします。

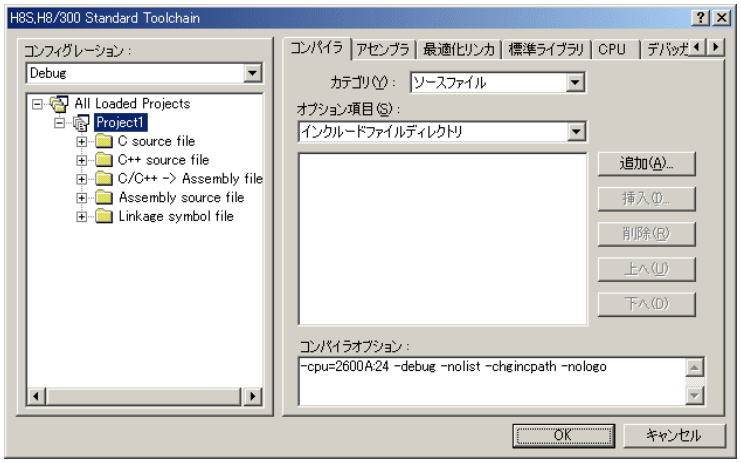


ライブライの作成を始めます。

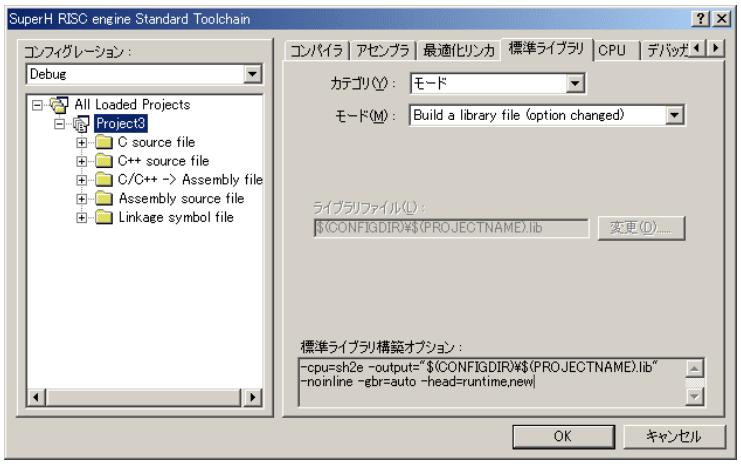
終了後プロジェクトのファイルをコンパイルしてリンクまで実行します。この時点ではまだエラーもしくはワーニングが出ますが無視してください。



[オプション]-[SuperH RISC engine Standard Toolchain]をクリックします。



[標準ライブラリ]タグをクリックする。



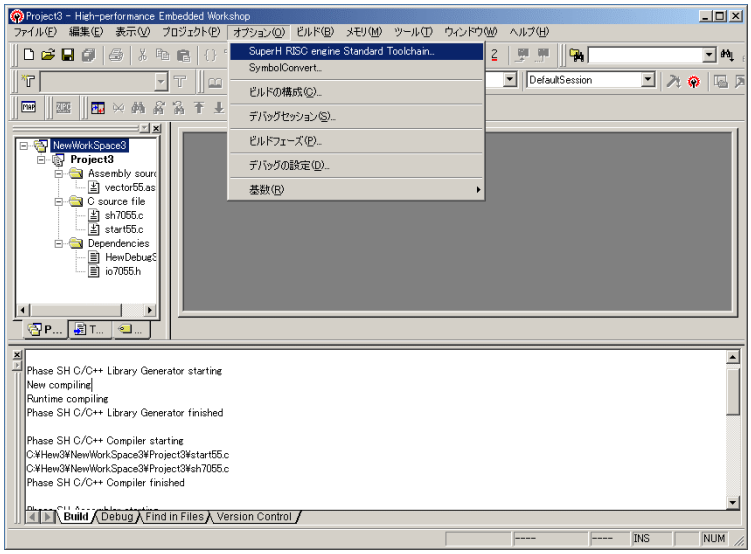
カテゴリのモードが「Build a library file(option changed)」指定になっている事を確認します。

この指定によりオプション変更時のみライブラリを作成する事になります。

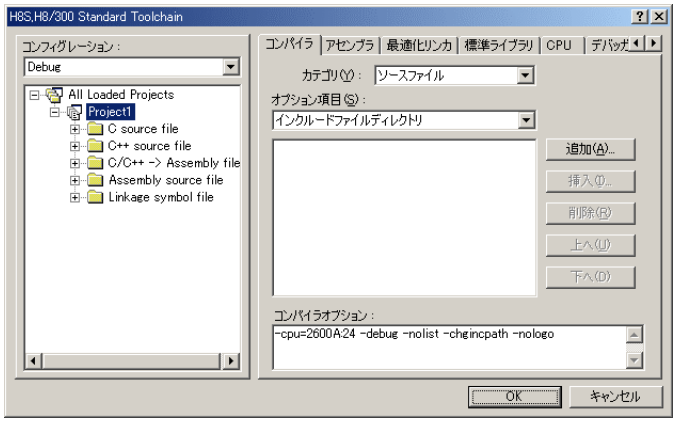
OKをクリックする。

次に、リンカの設定を説明します。

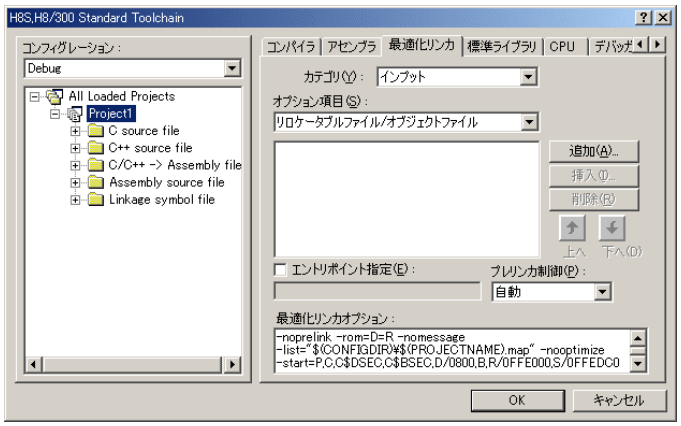
８．ツール（リンカ）の設定



[オプション]-[SuperH RISC engine Standard Toolchain]をクリックします。



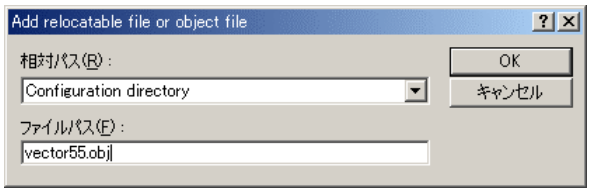
「最適化リンカ」タグをクリックする。



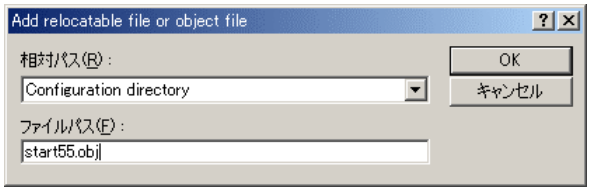
カテゴリの「インプット」を選択する。

オプション項目の「ﾘﾛｹｰﾀﾌﾞﾙﾌｧｲﾙ/ｵﾌﾞｼﾞｪｸﾄﾌｧｲﾙ」を選択する。

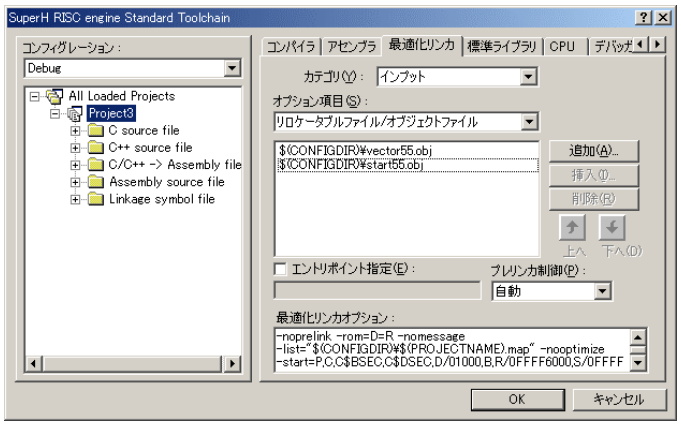
追加をクリックする。



相対パス「Configuration Directory」に選択します。
ファイルパス「vector55.obj」と入力します。
入力を確認して[OK]をクリックします。



相対パス「Configuration Directory」に選択します。
ファイルパス「start55.obj」と入力します。
入力を確認して[OK]をクリックします。

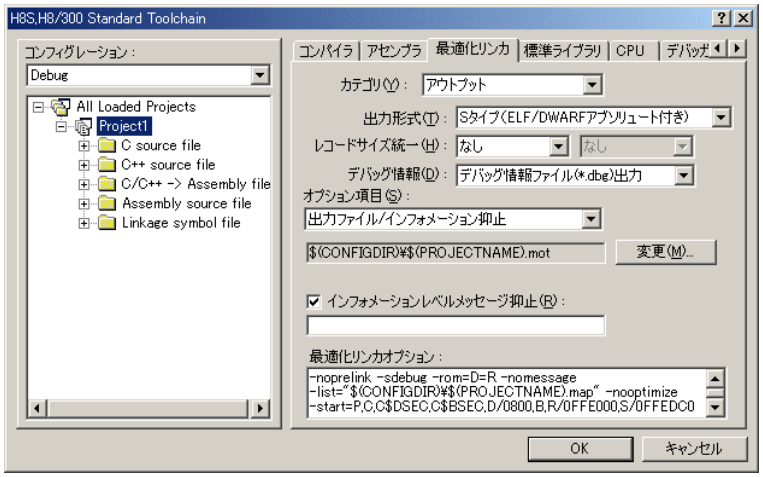


ファイルがセットされたのを確認します。

この指定は、“最初にこのリスト順にモジュールをリンクしなさい”との指示になります。

（DEF の C View にてアドレス順に他モジュールを表示させたい場合は、追加設定してください。）

（重要）　特に「start55.obj」は、スタートアップ関数ですので０ｘ８００番地にする為に必要な設定です。



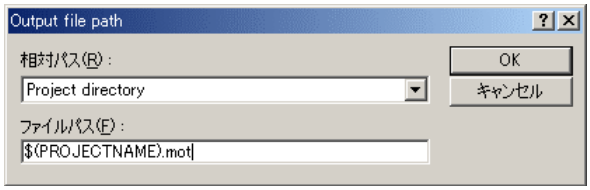
カテゴリの「アウトプット」を選択する。

出力形式の「S ﾀｲﾌﾟ(ELF/DWARF ｱﾌﾞｿﾘｭｰﾄ付)」を選択する。

デバッグ情報の「ﾃﾞﾊﾞｯｸﾞ情報ﾌｧｲﾙ(*.dbg)出力」を選択する。

オプション項目の「出力ﾌｧｲﾙ/ｲﾝﾌｫﾒｰｼｮﾝ抑止」の変更をクリックします。

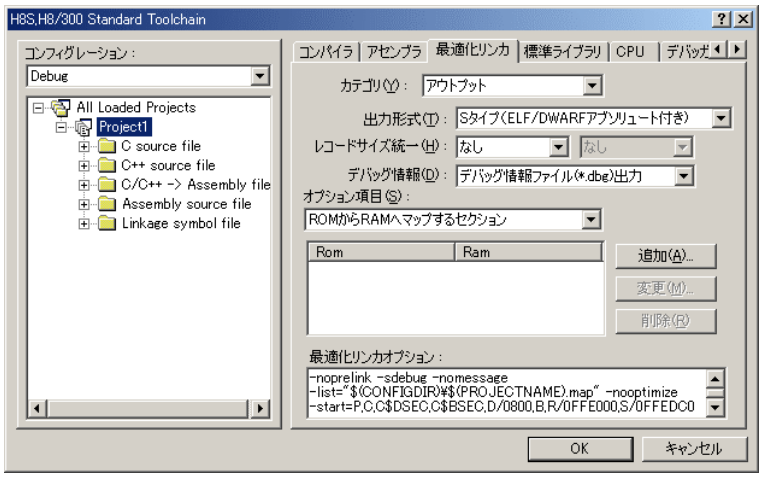
(重要)　シンボリックデバッグを可能にするために必要な設定です。



相対パスを「Project directry」に設定します。

OKをクリックする。

(重要)　この指定は、ＨＥＸファイルをＣソースファイルのある同じディレクトリに置くために必要です。

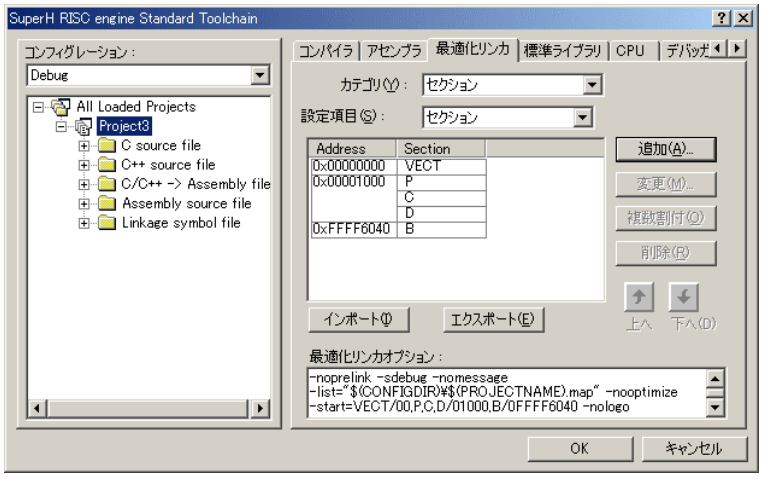


オプション項目の「ROMからRAMへのマップするセクション」を選択する。

Rom/Ram の D/R を選択クリックする。

削除をクリックする。

これで、未使用セクション名が Remove されます。



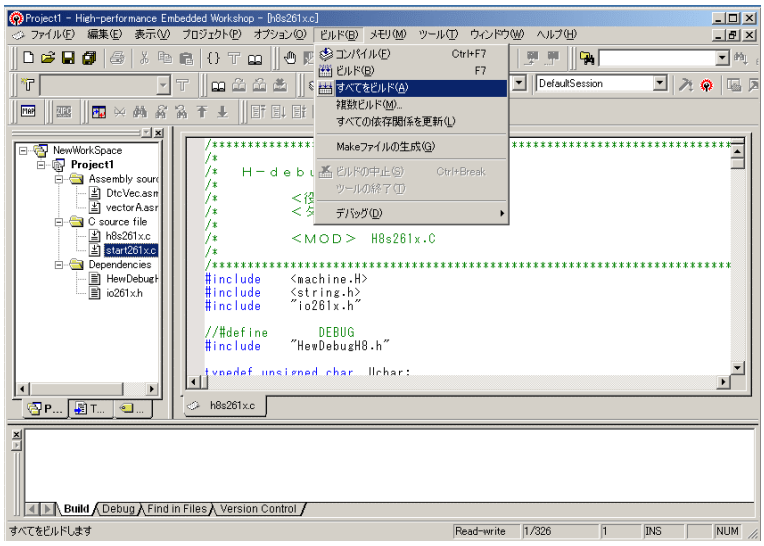
「セクション」を選択する。

下図のようにセクション指定をする。

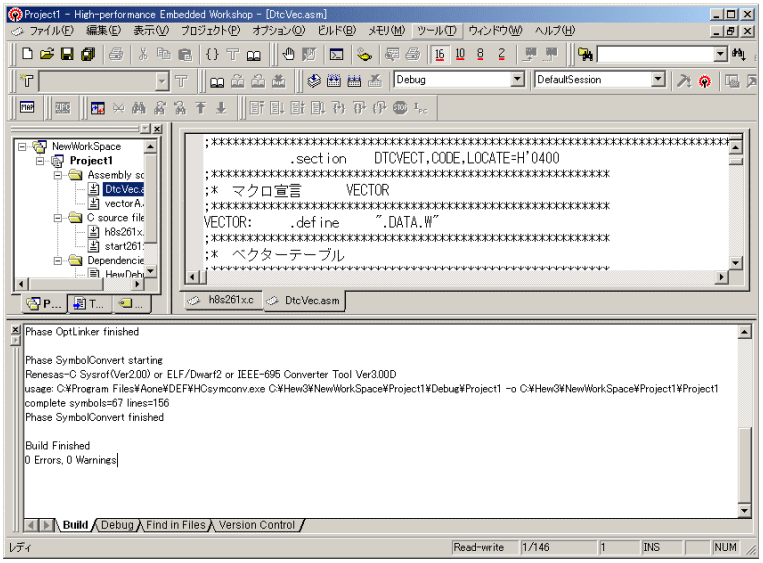
0x00000000	VECT
0x00001000	P
	C
	D
0xFFFF6040	B

OKをクリックする。

(重要)　セクション名：VECTは、「vector55.asm」で宣言しています。
「Bセクション」0xＦＦＦＦ６０４０（0xＦＦＦＦ６０００）は、ソースブレークを使用する場合の例です。
ＤＥＦバージョン６．５０ｘから、ソースブレーク使用する場合は、モニタワーク方式をスタック方式に選択する必要があります。



［ビルド］-[すべてをビルド]をクリックします。



「0 Error 0 Warnings」になり作業終了です。

以上