

統合環境Hew (Ver 4.04) 添付スタートアップ関数を使用した場合の
新ワークスペースおよびプロジェクトを登録する方法
(SH7145 BOOT版)

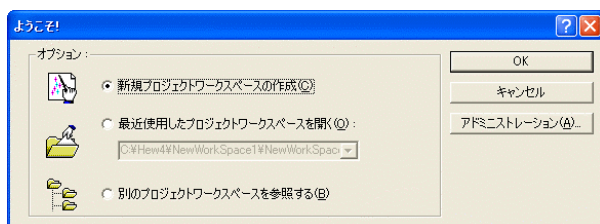
ルネサスC言語用統合環境「Hew Ver 4.04」で H-debugger 用に新ワークスペース／プロジェクトを登録する手順方法を説明します。
説明を明確にするために、名前等を仮に決めて例に沿って説明を進めます。

ワークスペース名	ReneSH7145_BOOT_Hew4		
プロジェクト名	Project		
登録モジュール名	SH7145.c	Cファイル	メインモジュール (アプリ用)
	HewDebugSH2.h	ヘッダファイル	ソフトパーツ用定義ファイル (ソフトパーツを使用しない場合は不要です。)
Hew添付ファイル	Resetprg.c	Cファイル	スタートアップモジュール
	Intprg.c		割込みハンドラモジュール
	Dbsect.c		定数転送用セクション管理宣言
	Vecttbl.c		ベクター定義モジュール
	iodefine.h	ヘッダファイル	I/O 定義ビットフィールド記述用
CPUタイプ	SH7145F		

1. 新ワークスペースの登録方法

“HEW” 起動させます。

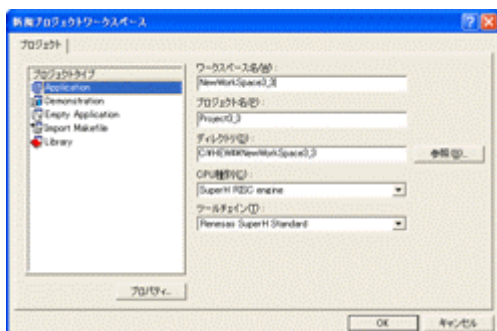
[1-1]



“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。

[1-2]



ワークスペース名	“ReneSH7145_BOOT_Hew4”
プロジェクト名	“Project”
ディレクトリ	“C:\Hew4\Renesas”
CPU 種別	“SuperH RISC engine”
ツールチェーン	“Renesas SuperH Standard”
プロジェクト	Application

この項目を確認後、OKをクリックして下さい。

[1-3]



CPUシリーズを“SH-2”に選択する。
CPUタイプを“SH7145F”に選択する。

確認後、

次へ>をクリックします。

[1-4]



CPUスペックを確認後

次へ>をクリックして下さい。

[1-5]



- ① I/O ライブラリを使用しませんのでチェックを外して下さい。
- ② ヒープメモリを使用しませんのでチェックを外して下さい。
main() 関数生成は“None”に選択する。
- ③ I/O レジスタ定義ファイルは使用しますのでチェックして下さい。
ハードウェアセットアップ関数生成は” None” に選択する。

確認後、次へ>をクリックして下さい。

[1-6]



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。

次へ>をクリックして下さい。

[1-7]



スタックボトムの設定です。Default のままでも構いませんが、RAM 使用の節約の為、「HFFFFFFFC」にする。

スタックサイズはデフォルト値でよい。

次へ>をクリックして下さい。

[1-8]



ここで明示されたHeap作成スタートアップ関数を使用しますので、デフォルトの状態で、

次へ>をクリックして下さい。

[1-9]



シミュレータの設定ですが使用しませんのでチェック無しの状態で、

次へ>をクリックして下さい。

[1-10]

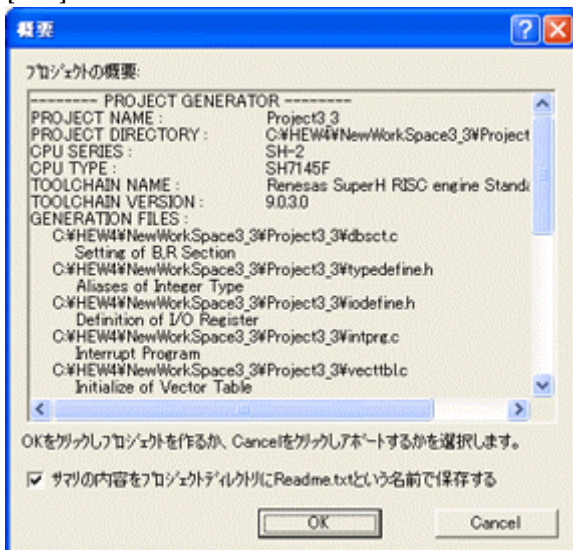


ここで最終になります。

使用するCモジュールを表示します。

この状態で完了をクリックして下さい。

[1-11]



確認画面が表示されますので、

OKをクリックして下さい。

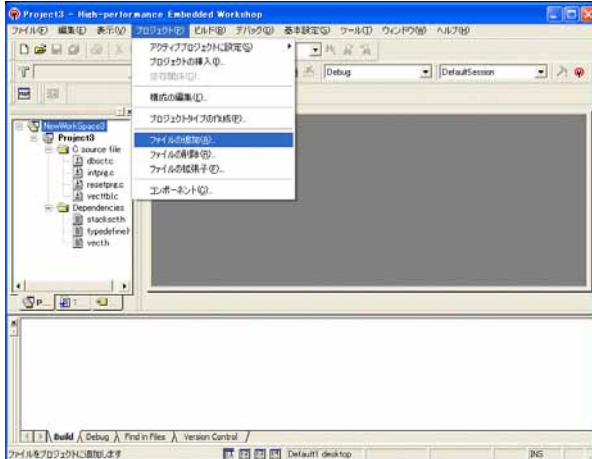
ここまでの操作が新規プロジェクトの登録方法です。

2. 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2ファイルを”C:\Hew4\Renesas\ReneSH7145_BOOT_Hew4\Project”にコピーします。

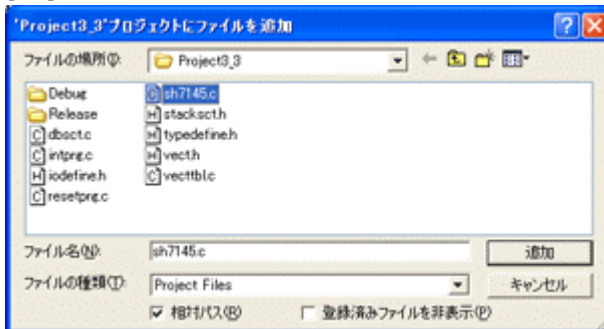
SH7145.c HP よりダウンロードします。(ルネサスC)
HewDebugSH2.h ReneSH7145_BOOT_Hew4.LZH

[2-1]



[プロジェクト]-
[ファイルの追加]をクリックします。

[2-2]



下記1ファイルを指定して下さい。

Sh7145.c

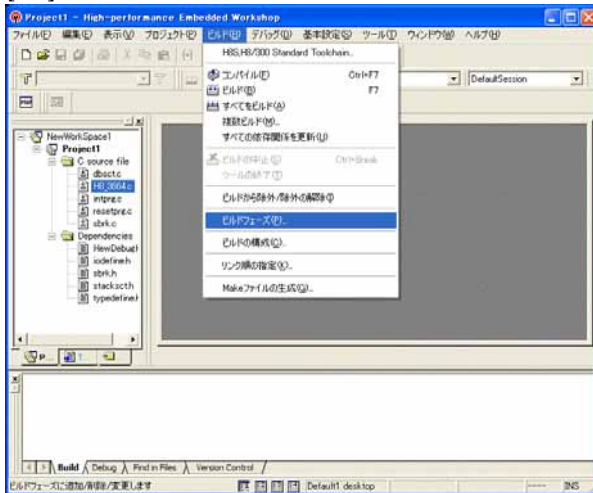
選択後、

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が登録されました。

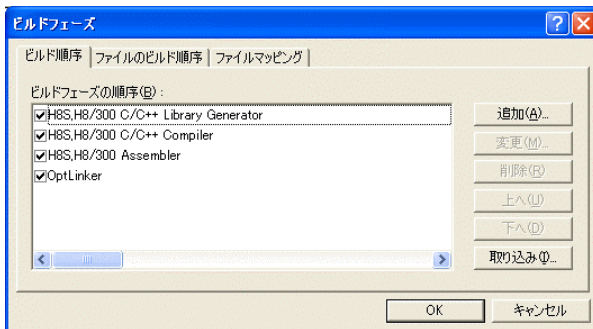
3. シンボルコンバータ「HC symconv」を登録する。

[3-1]



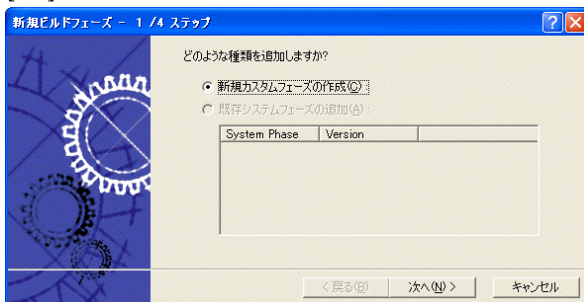
[ビルド]-
[ビルドフェーズ]をクリックします。

[3-2]



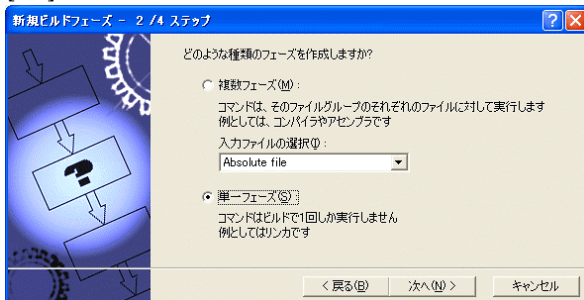
追加をクリックします。

[3-3]



次へ>をクリックします。

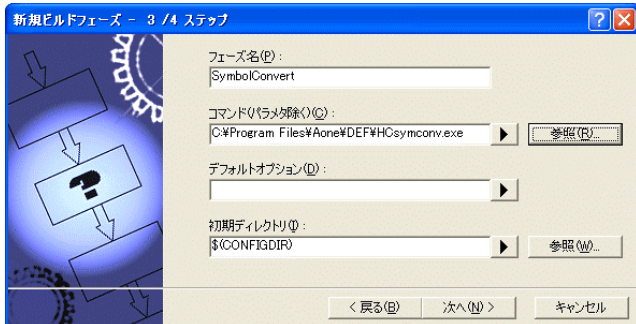
[3-4]



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。

[3-5]



①フェーズ：SymbolConvert

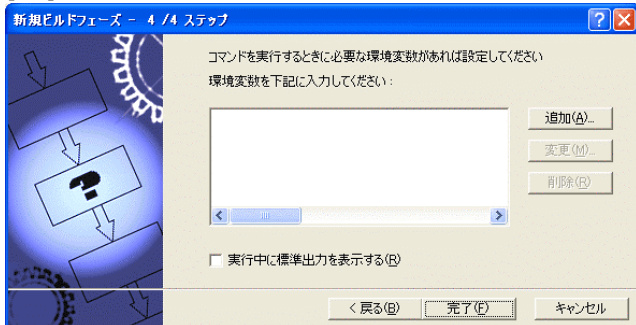
②コマンド：

C:\Program Files\Aone\DEF\HCsymconv.exe

③初期ディレクトリ：\$(CONFIGDIR)

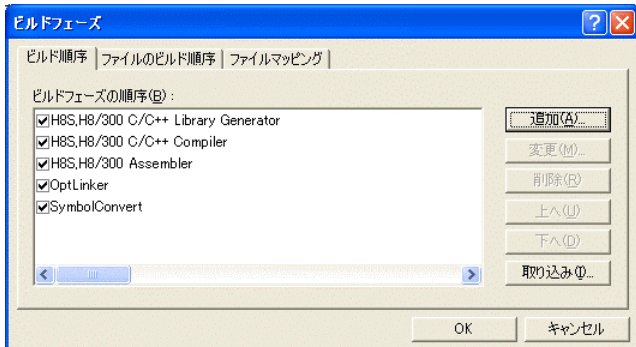
次へ>をクリックします。

[3-6]



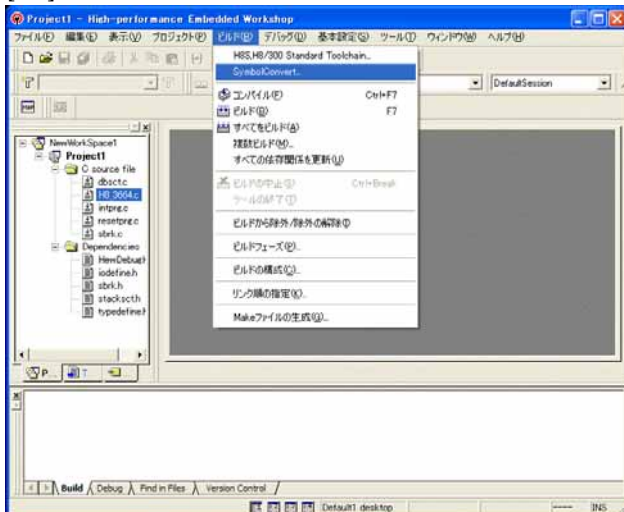
完了をクリックします。

[3-7]



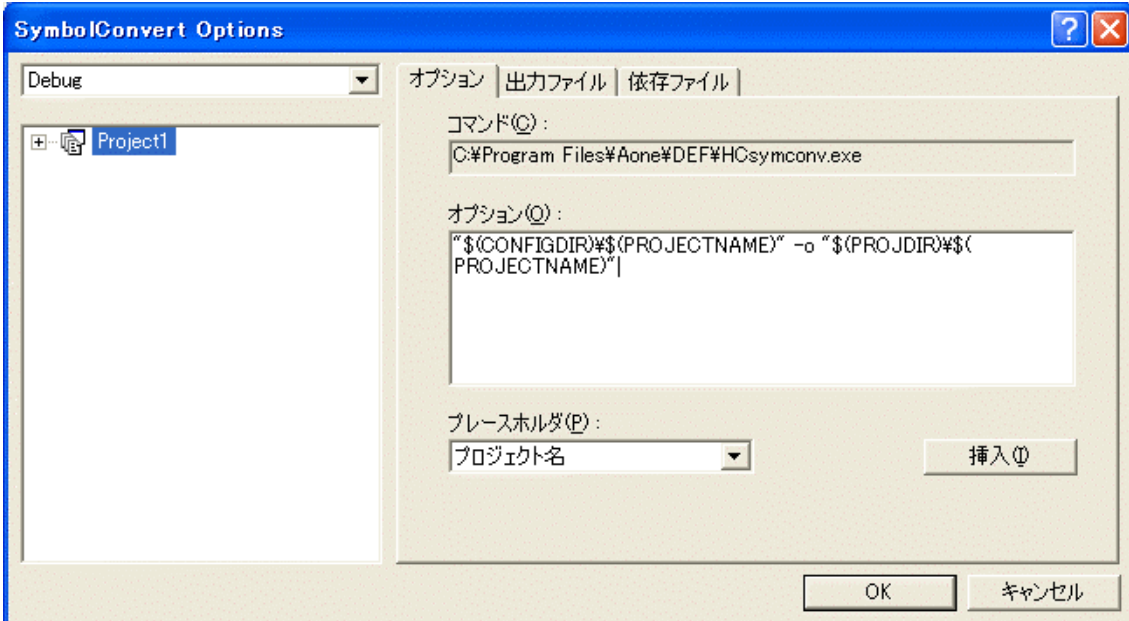
OKをクリックします。

[3-8]



[ビルド]-
[SymbolConvert]をクリック
します。

[3-9]



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 この設定例は、後説明の「*.mot」ファイルの生成されるディレクトリと同じ場所にシンボルコンバータが生成する「*.sym/*.lin」を置く為の指定です。
＜コンフィグレーションDIR＞に生成させたい場合は、
“\$(CONFIGDIR)\\$(PROJECTNAME)”
の指定のみで構いません。
この場合は「*.mot」の生成場所を同じく＜コンフィグレーションDIR＞にして下さい。

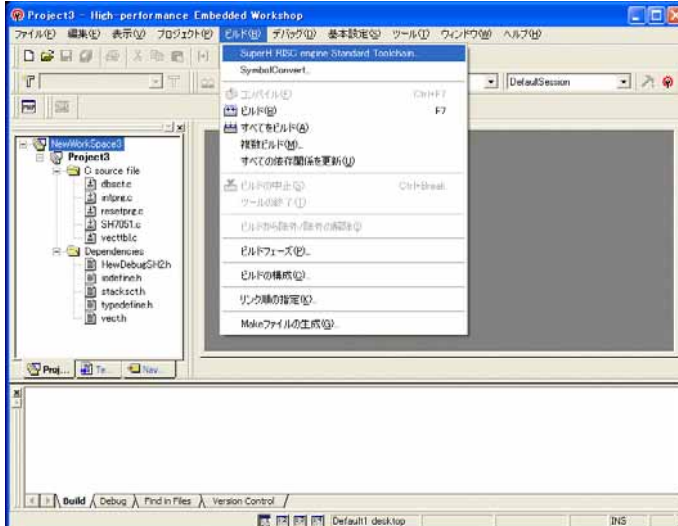
追加事項 (HCSymconv.exe スイッチ説明)

- 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(Ver 3. 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。(Ver 3. 2xxから)
- 4) [-i] (省略可) 重複モジュール情報を削除する。(Ver 3. 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。(Ver 3. 6xxから)
- 6) [-m] (省略可) 重複モジュール情報をCソースにマージする。(Ver 3. 80Bから)
- 7) [-f] (省略可) 使用インクルードファイルをCViewに登録する。(Ver 3. 80Bから)

4 . 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。
ライブラリを作成および設定の確認をします。

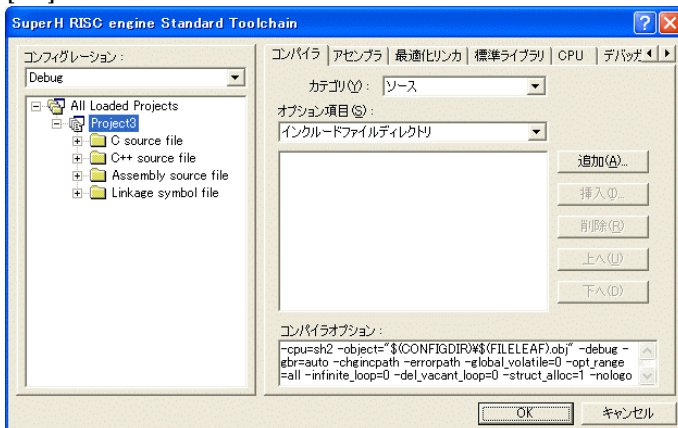
[4-1]



[ビルド]-

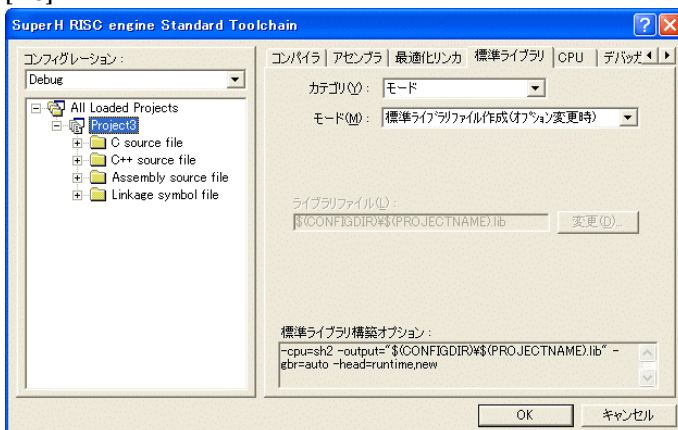
[SuperH RISC engine Standard Toolchain]をクリックします。

[4-2]



「標準ライブラリ」タグをクリックする。

[4-3]



カテゴリのモードが

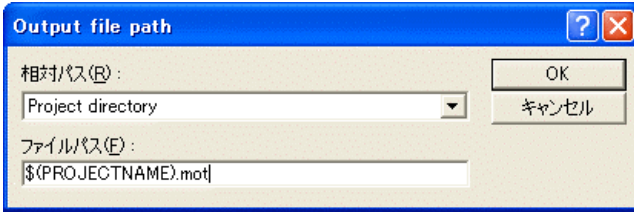
「ライブラリファイル作成 (オプション変更時)」指定になっている事を確認します。

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。

デフォルトのまま

OKをクリックする。

[5-4]



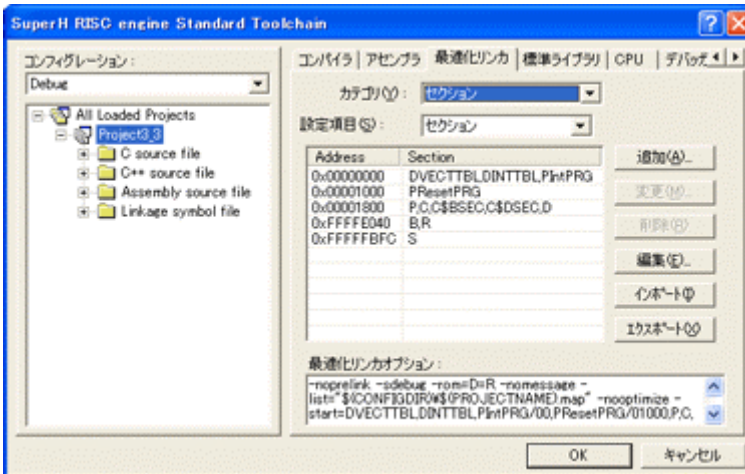
相対パス「Project directory」に設定します。

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の設定です。絶対条件として、「*.mot/*.sym/*.lin」は、同じ場所に置く必要があります。

HCsymconvで出力ファイルを「Configuration directory」にした場合は、上記の指定も「Configuration directory」にして下さい。今回の使用例は、「Project directory」になっています。

[5-5]



カテゴリの「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。

(モニタエリア確保のため)

OKをクリックする。

0x00000000	DVECTTBL, DINTTBL, PIntPRG
0x00001000	PRresetPRG
0x00001800	P, C, C\$BSEC, C\$DSEC, D
0xFFFFE040	B, R
0xFFFFFBFC	S

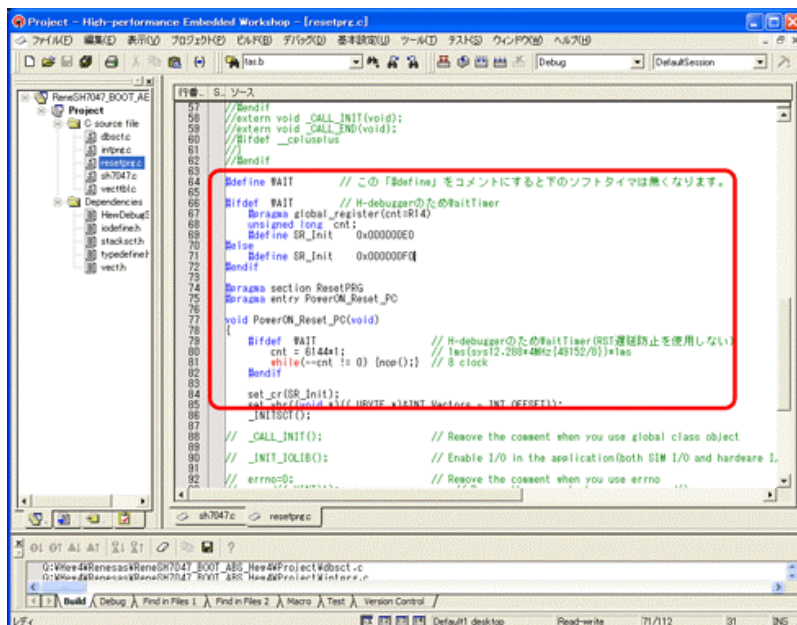
(重要) 「Bセクション」0xFFFFE040（内臓RAM先頭 0xFFFFE000）は、ソースブレイク使用する場合の例です。

DEFバージョン6.50xから、ソースブレイクを使用する場合は、モニタワーク方式をスタック方式に選択して下さい。

6 . スタートアップおよびベクターの変更と追加

1) <resetprg. c>の変更

[6-1]



①左図のようなソース行を追加しますとリセット時ソフトタイマ挿入になります。

このソースを追加することにより④項のメリットがあります。

②デバッグ中は、割り込みマスクレベルを14にします。「2) 項参照」

③ソフトタイマが不必要な場合は「#define WAIT」をコメントにしてください。

④ソフトタイマによるメリット

- ・DEFの「RstMon」操作時「main ()」プログラムまで走行しないため初期設定等による内部レジスタが汚れない。
この例ですと **1ms** ですがリセット遅延が無い場合は **CPU** 設定でリセット遅延なしの指定をすれば「**20us**」タイマ値で機能します。

2) ブレークおよびトレース/ステップ実行について (マスクレベルを14にする理由)

UBCブレーク割り込みのプライオリティは、15になっていますのでCPUのSRレジスタの割り込みマスクレベルを14以下に設定する必要があります。下記方法のどちらかを都合に応じて実施して下さい。

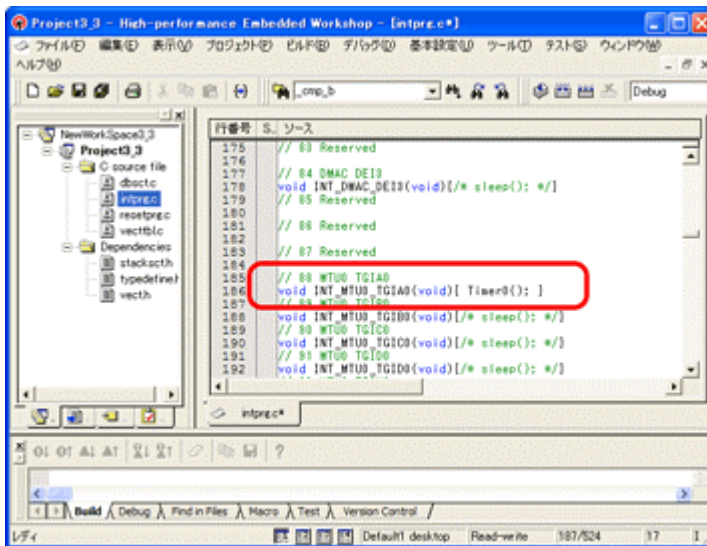
なお、トレース/ステップはUBCを利用したソフト判定での実施となります。

【方法①】 「set_imask(14)」を記述する。

【方法②】 DEF操作でのショートPB「DI」をクリックする。

3) <intprg.c>の変更

[6-3]

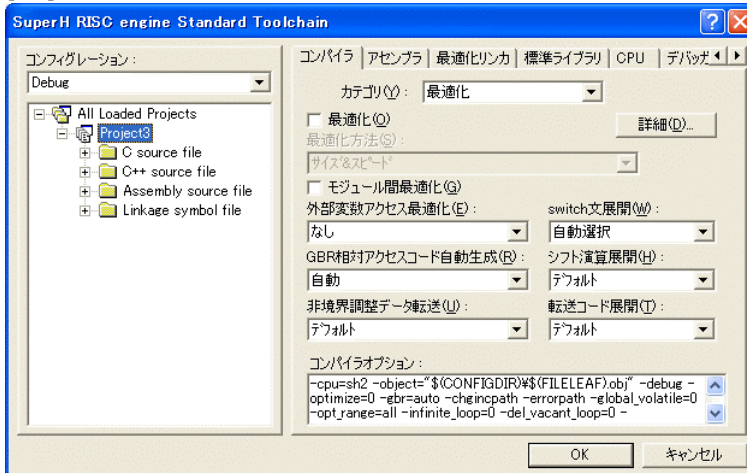


今回使用の「SH7144.C」は、MTU0のTGIA0割り込みを使用した例ですのでベクターを設定します。

- ①「vector 88」に「Timer0();」関数を登録します。

4) コンパイラの「最適化」を外す

[6-4]



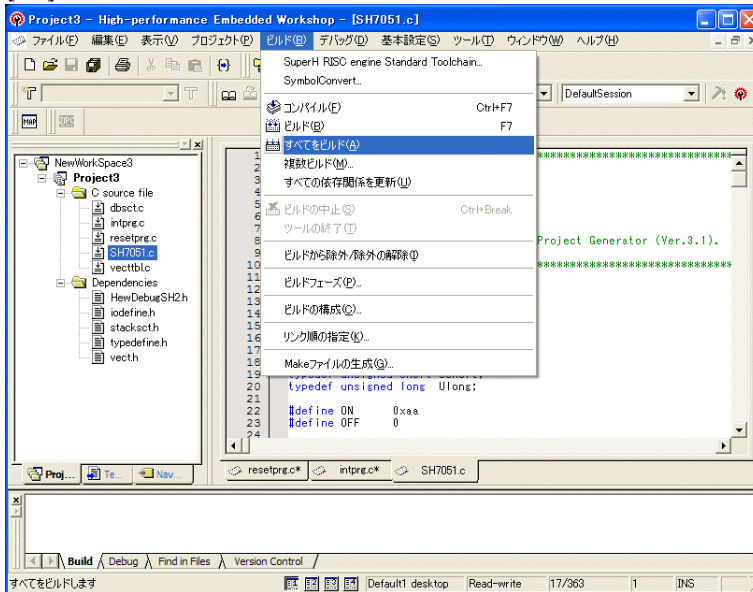
- ①[ビルド] - [SuperH RISC engine Standard Toolchain]をクリックします。

- ②「コンパイラ」を選択
- ③カテゴリ「最適化」を選択
- ④「最適化」のチェックを外す。

- ⑤OKをクリックする。

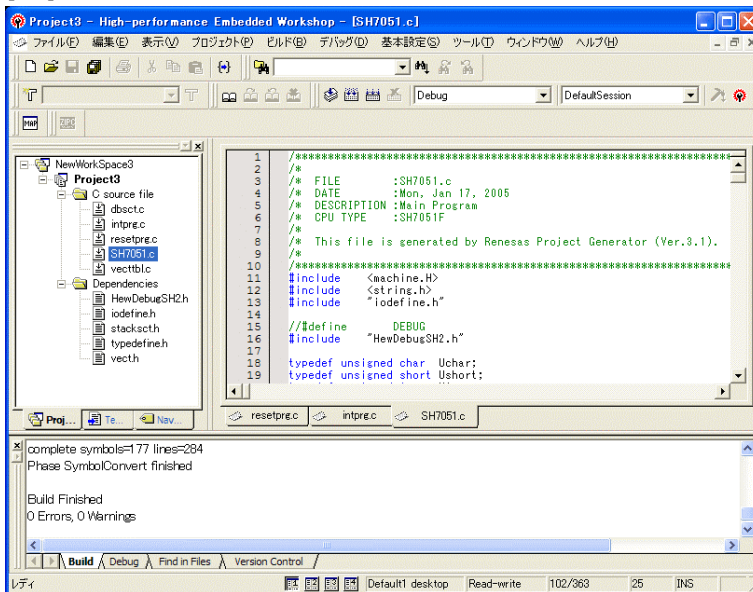
7. ビルドの実行

[7-1]



[ビルド] -
[すべてをビルド]をクリック
します。

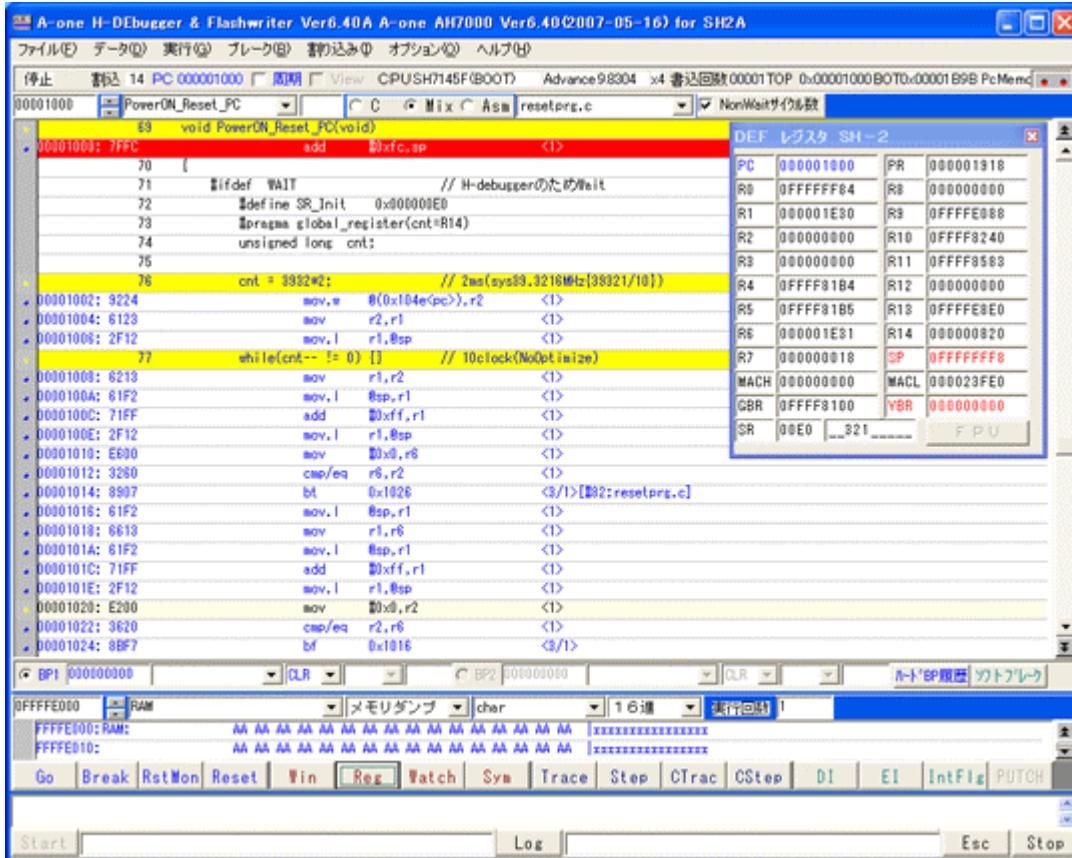
[7-2]



「0 Error 0 Warnings」にな
り作業終了です。

8. DEFでの確認

[8-1]



- ①0番地のリセットベクター値「0x1000」が確認できます。
- ②ソフトタイマの逆アセンブラ表示です。(参考まで)
- ③SR値およびSP値がリセット状態であることが確認できます。

これで「H-Debugger」用の設定作業が終了です。

以上