アブソリュートファイル指定でダウンロードする場合　　ＤＥＦ　Ｖｅｒ６．８０Ａより

２００８年０３月１０日
エーワン　Ｒｅｖ１．００

Ｈｅｗ（Ｖｅｒ４．０４）＋ＫＰＩＴ（ｖ０８０１）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

（ＳＨ－２／７１４５　Ｈ－ＵＤＩ版）

ＫＰＩＴ版の統合環境「Ｈｅｗ　Ｖｅｒ４．００」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	KPIT7145_ABS		
プロジェクト名	Project		
登録モジュール名	SH7145.c	Ｃ	メインモジュール（アプリ用）
KPIT 添付ファイル	start.asm	ＡＳＭ	スタートアップモジュール
	hwinit.c	Ｃ	ハード初期化用モジュール
	vects.c	Ｃ	リセット／割込みベクターテーブル
	inthandler.c	Ｃ	割込みハンドラー用
	iodefine.h	ヘッダ	I/O 定義ビットフィールド記述用
	inthandler.h	ヘッダ	割込みハンドラー用
ＣＰＵタイプ	SH7145F		

 　【ポイント】

「ＳＨ－２　Ｈ－ＵＤＩ版」の場合、Ｈｅｗデフォルト設定から変更に必要な箇所は、

１）コンパイラの「Ｏｂｊｅｃｔ」項目を変更する。

の１点になります。

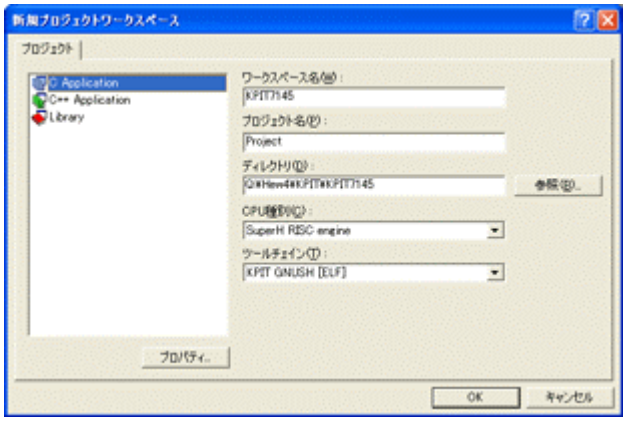
１．新ワークスペースの登録方法　　“HEW”起動させます。

[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。

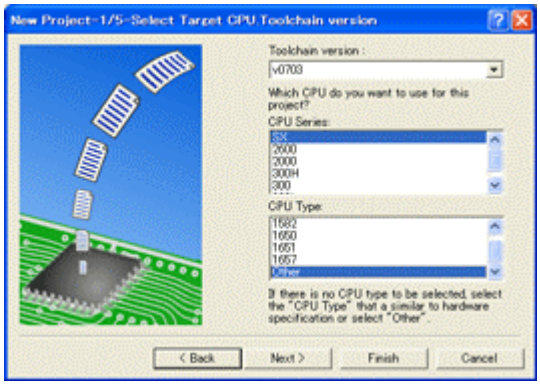
[1-2]



ﾜｰｸｽﾍﾟｰｽ名	KPIT7145_ABS
ﾌﾟﾛｼﾞｪｸﾄ名	Project
ﾃﾞｨﾚｸﾄﾘ	Q:¥Hew4¥KPIT¥KPIT7145
CPU種別	SuperH RISC engine
ﾂｰﾙﾁｪｲﾝ	KPIT GNUSH[ELF]
ﾌﾟﾛｼﾞｪｸﾄ	Application

この項目を設定確認後ＯＫをクリックして下さい。

[1-3]



ＣＰＵスペックを選択します。

①SH2
②SH7145F

Ｎｅｘｔ＞をクリックします。

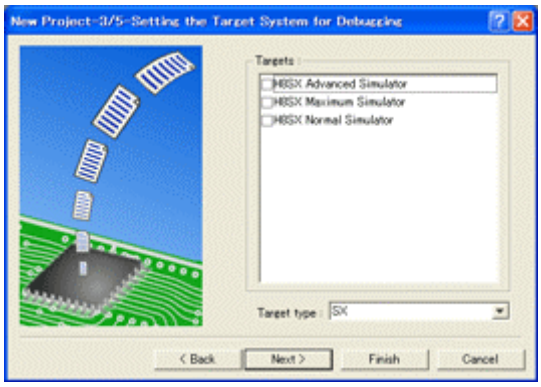
[1-4]

ＣＰＵオプションを選択します。

①「Use Optimized Libraries」
チェックを外します。

Ｎｅｘｔ＞をクリックします。

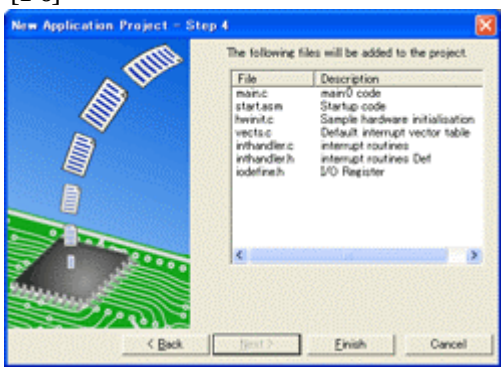
[1-5]



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、

Ｎｅｘｔ＞をクリックします。

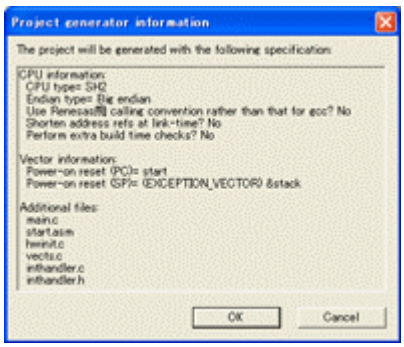
[1-6]



作成されるファイル一覧表示です。

Ｆｉｎｉｓｈ＞をクリックします。

[1-7]



最終確認画面です。

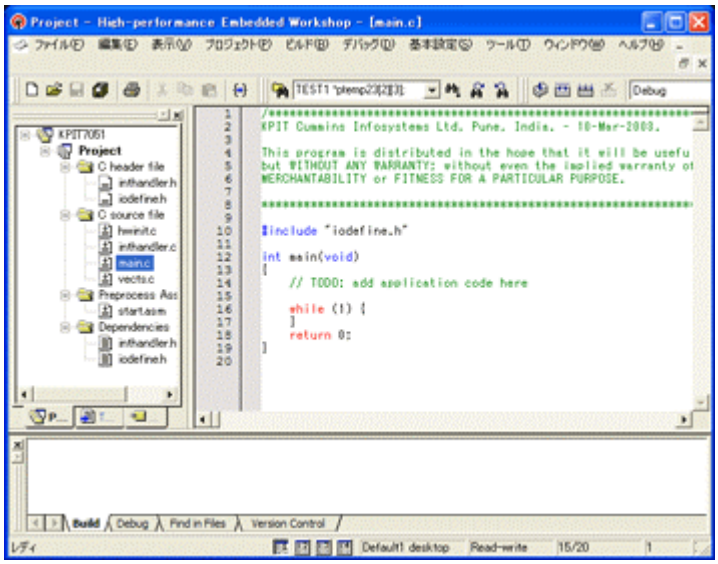
ＯＫをクリックします。

ここまでの操作が新規プロジェクトの登録方法です。

２．プロジェクトから不要モジュール（ソースファイル）を削除します。

目的：　ＫＰＩＴにより準備されたモジュールを使用しない場合に削除しておきます。

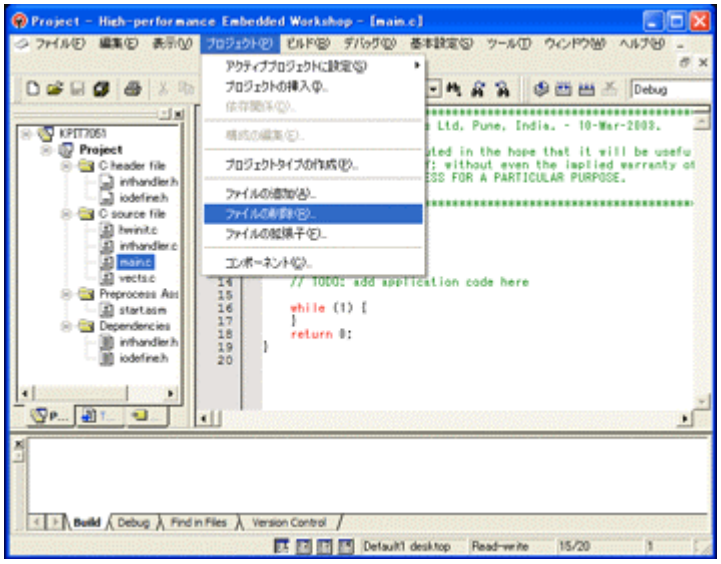
[2-1]



今回の使用例では下記１ファイルを削除します。

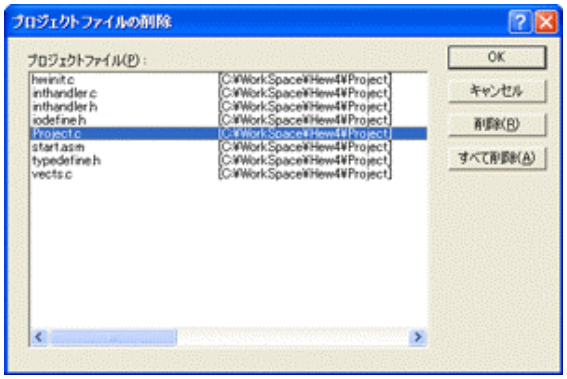
①Project.c

[2-2]



［プロジェクト］－
［ファイルの削除］をクリックします。

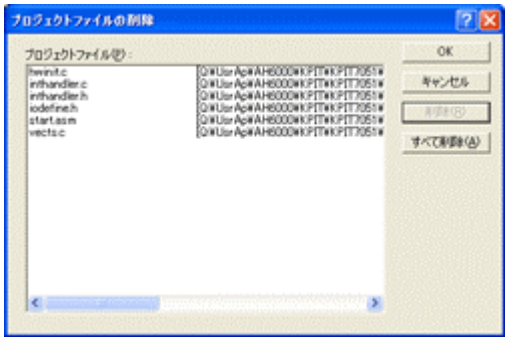
[2-3]



① project.c
の１ファイルを選択する。

削除をクリックします。

[2-4]



確認画面です。

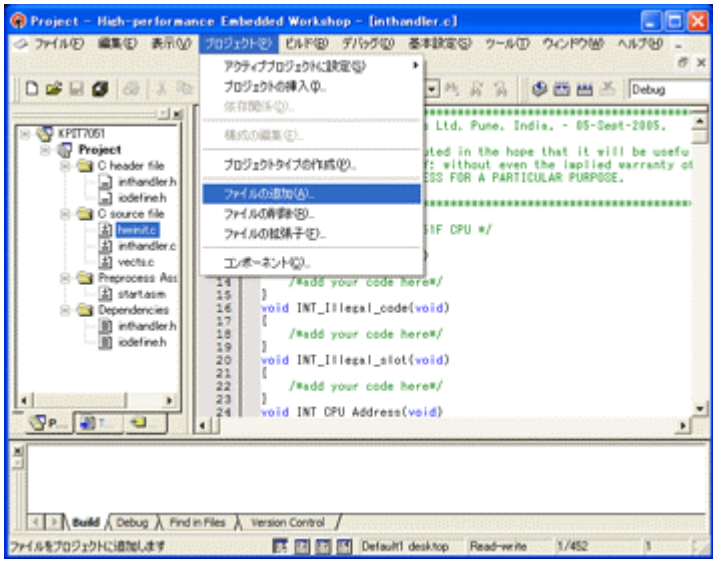
OK をクリックします。

３．プロジェクトに希望モジュール（ソースファイル）を登録します。

準備：　作成済みの１ファイルを”C:¥Hew4¥KPIT¥KPIT7145_ABS¥Project”にコピーして下さい。

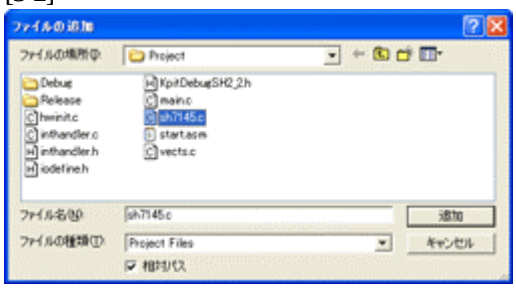
SH7145.ｃ　　HP よりダウンロードします。（ＧＮＵ／ｇｃｃ）
KPIT7145_ABS_v0801.LZH

[3-1]



[プロジェクト]-
[ファイルの追加]をクリックします。

[3-2]



登録ファイルを選択します。

① SH7145.ｃ

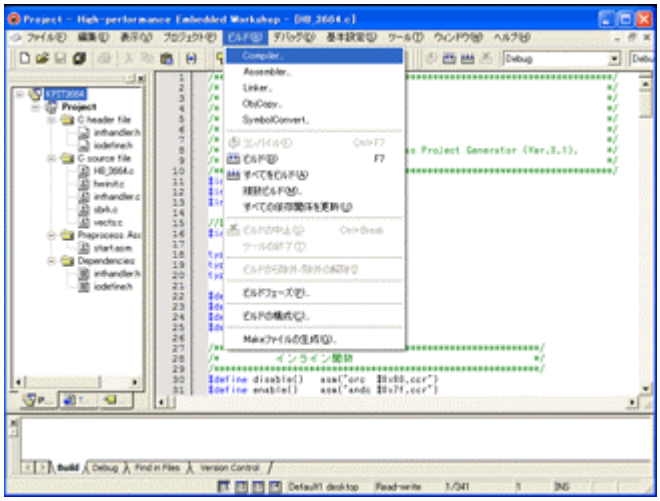
追加をクリックします。

この操作によりプロジェクトにモジュールを登録します。

４．コンパイラーオプションの確認と設定をします。

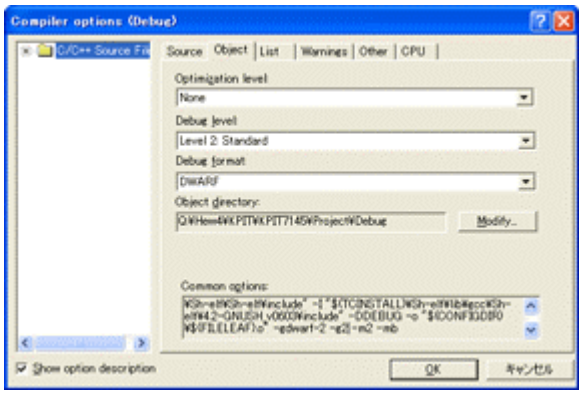
目的：　H-debugger でシンボリックデバッグを可能にする為、コンパイラーオプションの確認と設定をします。

[4-1]



［ビルド］-
［Ｃｏｍｐｉｌｅｒ］をクリックします。

[4-2]



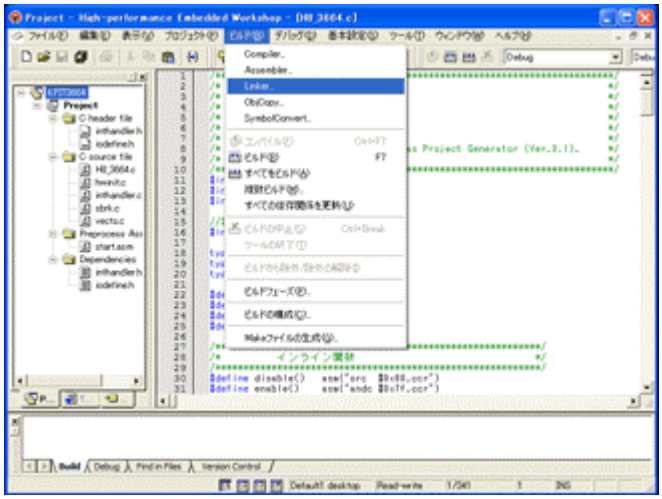
［Ｏｂｊｅｃｔ］タグ
①Optimization : None(Default)
②Debug level :　Level2:Standard(Default)
③Debug format:DWRAF に指定する。
④Object directory : (Default)状態

OKをクリックします。

５．リンカーオプションの確認と設定をします。

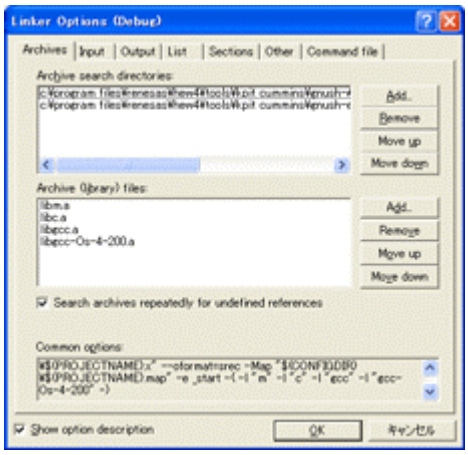
目的：　H-debugger でシンボリックデバッグを可能にする為、リンカーオプションの確認と設定をします。

[5-1]



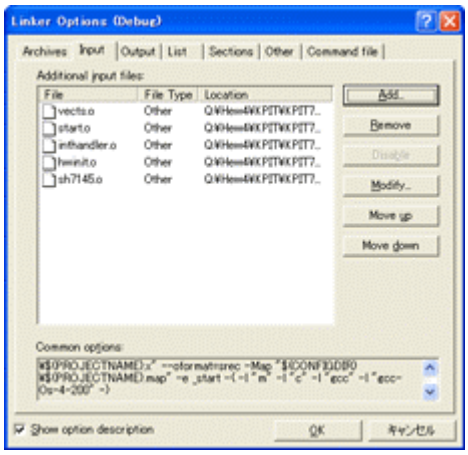
［ビルド］-
［Ｌｉｎｋｅｒ］をクリックします。

[5-2]



［Ａｒｃｈｉｖｅｓ］タグ
デフォルト状態です。(変更の必要なし)

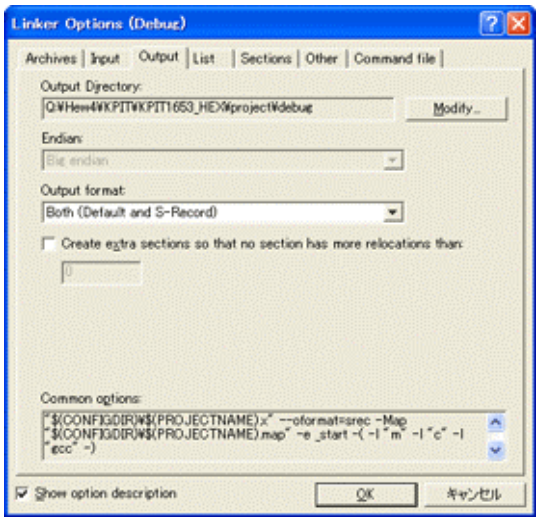
[5-3]



［Ｉｎｐｕｔ］タグ
　基本的には何も設定しなくて良いですが、各モジュールのリンク順番を指定したい場合に全モジュールをここで指定します。

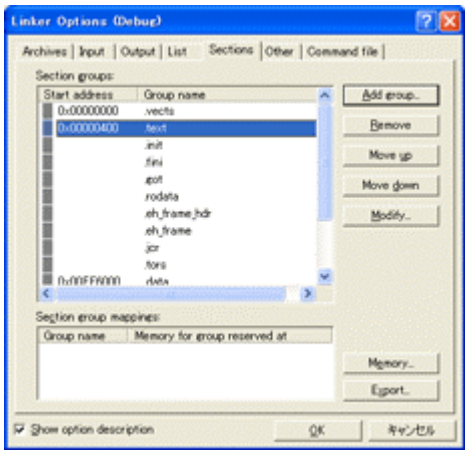
① vects.o
② start.o
③ inthandler.o
④ hwinit.o
⑤ SH7145.o

[5-4]



［Ｏｕｔｐｕｔ］タグ
①Output Directory : (Default)
②Endian : Big endian(Default)
③Output format :Both(Default and S-Record)
すべて、デフォルトです。

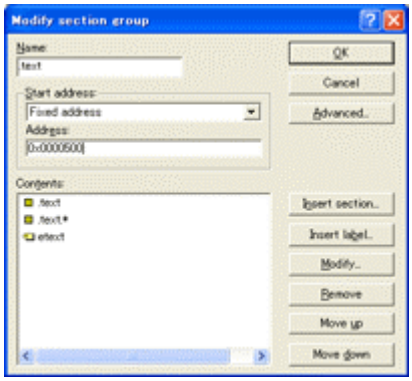
[5-5]



［Ｓｅｃｔｉｏｎｓ］タグ
.text セクションの開始アドレスを変更します。
デフォルトで「0x1000」になっています。
「0x400」番地から可能ですが、DTC ベクターテーブルを考慮して「0x500」番地に変更します。

①.text セクションを選択します。
②【Modify】PB をクリックします。

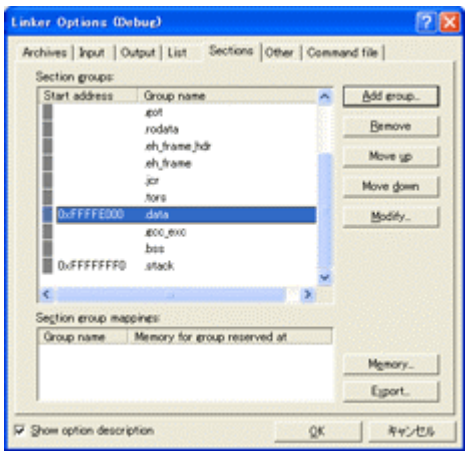
[5-6]



③Address: を「0x500」に変更します。

④ＯＫをクリックします。

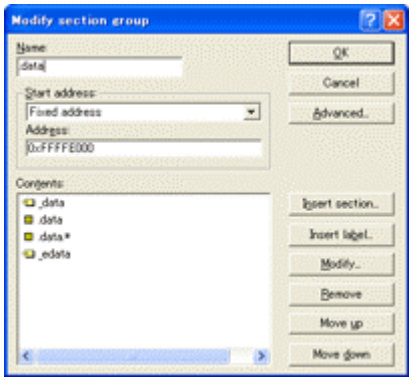
[5-7]



［Ｓｅｃｔｉｏｎｓ］タグ
.data セクションのアドレスを変更します。

①.data セクションを選択します。
②【Modify】PB をクリックします。

[5-8]

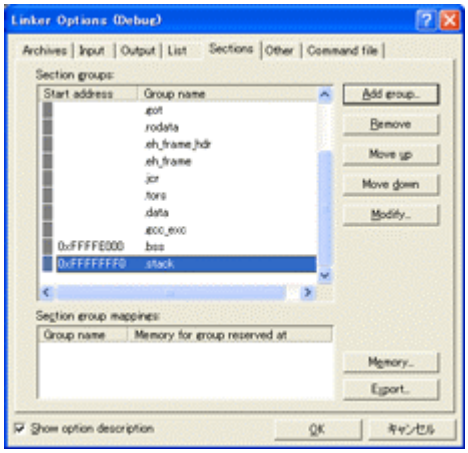


③Start address:【Fixed address】に選択します。
④Address: .bss セクションの先頭アドレスを指定します。
【0xFFFFE000】

ソースブレーク使用の指定の場合でも内臓ＲＡＭは使用しません。

⑤OK をクリックします。

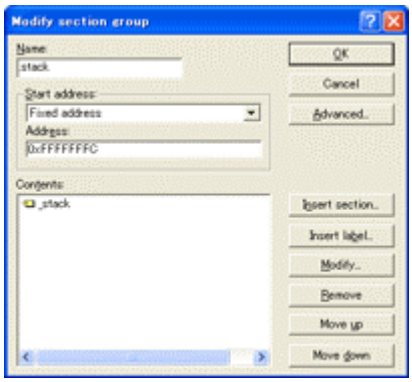
[5-9]



［Ｓｅｃｔｉｏｎｓ］タグ
.stack セクションのアドレスを指定します。
ここでの指定値は、スタックポインタへの初期設定値になります。

①.stack セクションを選択します。
②【Modify】PB をクリックします。

[5-10]



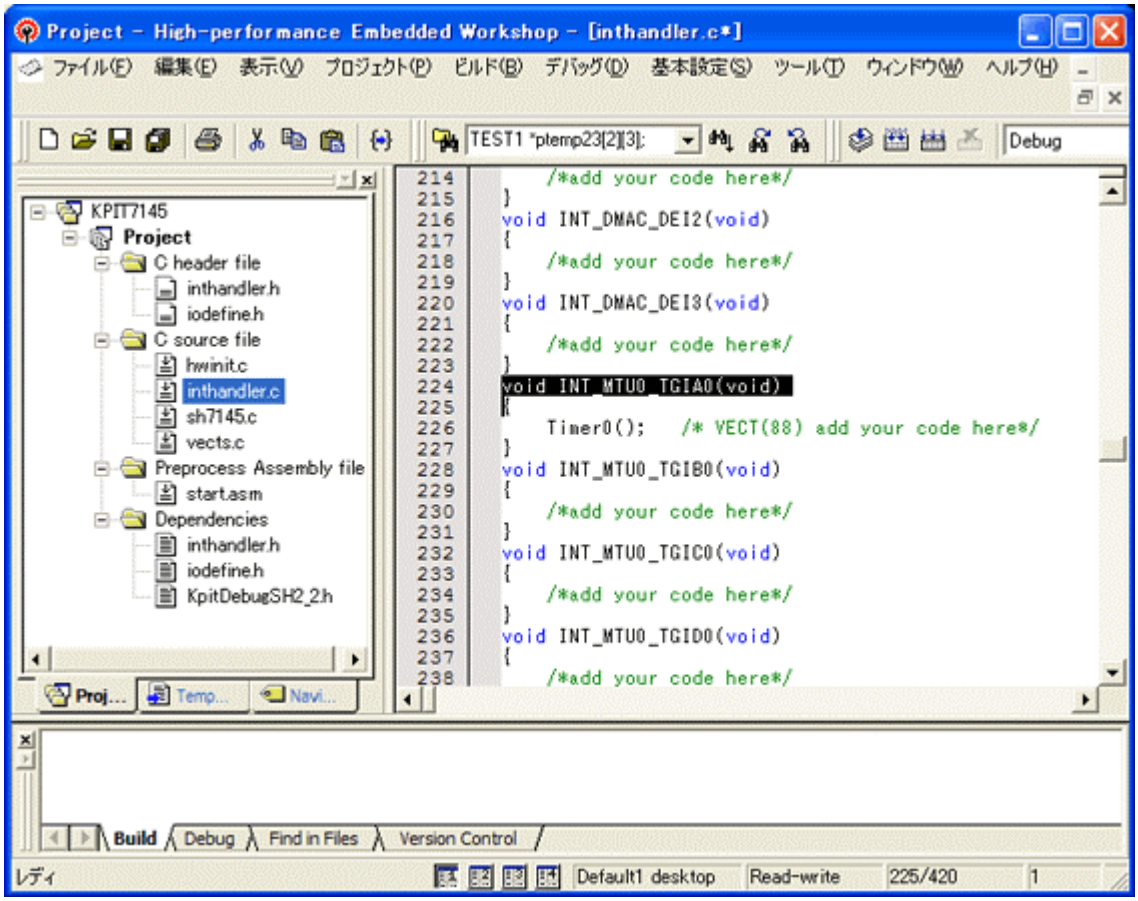
③Address: を「0xFFFFFFFC」に変更します。

④OK をクリックします。

６．割り込みハンドラへ登録します。

目的：　今回説明に使用したモジュール「SH7145.c」は、Timer0（ベクター８８）の割り込みを使用していますので、割り込みハンドラへ登録します。

[6-1]



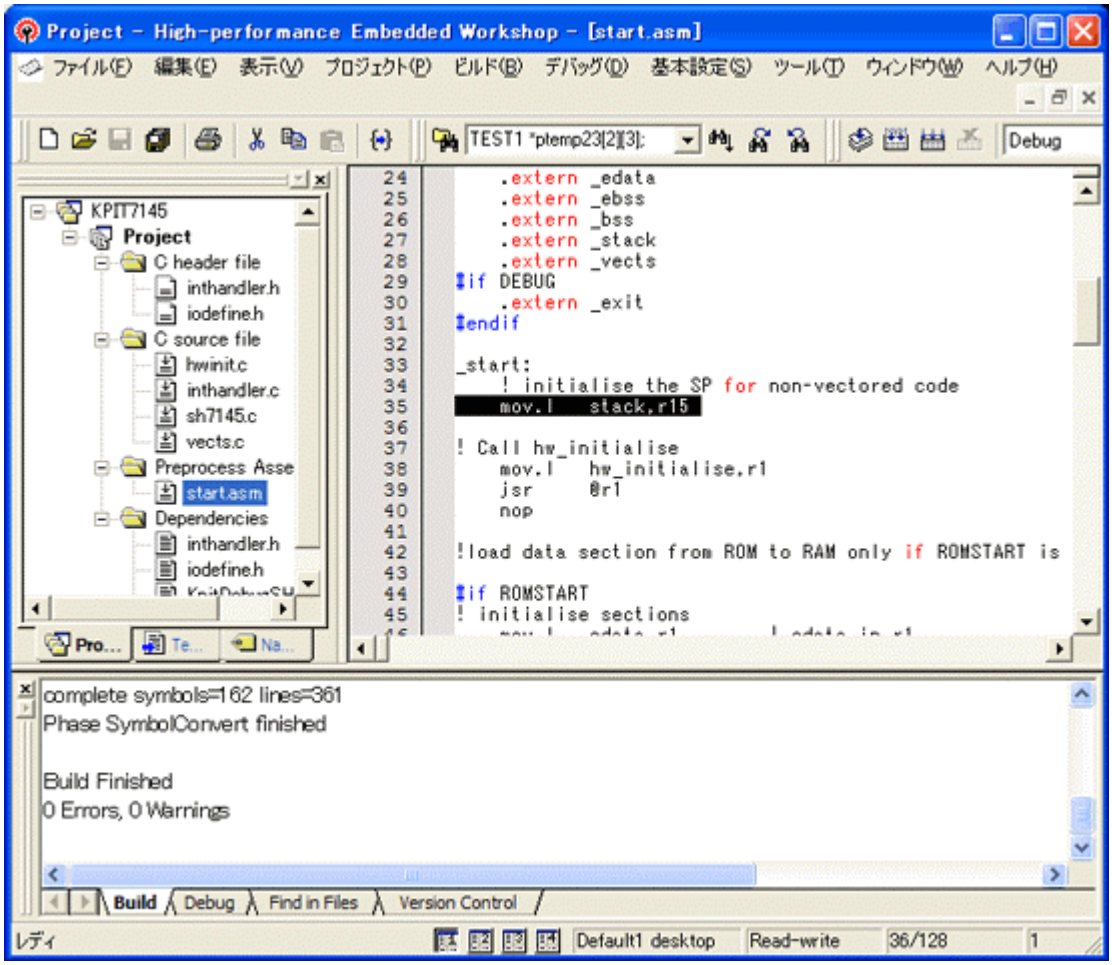
①inthandler.c を選択します。
②void INT_MTU0_TGIA0(void){ Timer0(); }の関数を記述します。

【注意】
①「vects.c」の２５６ベクターに「INT_Dummy」が登録されています。これは 0x400 番地を超えますし、意味の無い登録ですので削除します。（このサンプルでは、削除済みです。）

７．スタートアップ「start.asm」の説明です。

目的：　スタートアップ「start.asm」に実際は不要なソースコードがありますが、
その補足説明です。

[7-1]

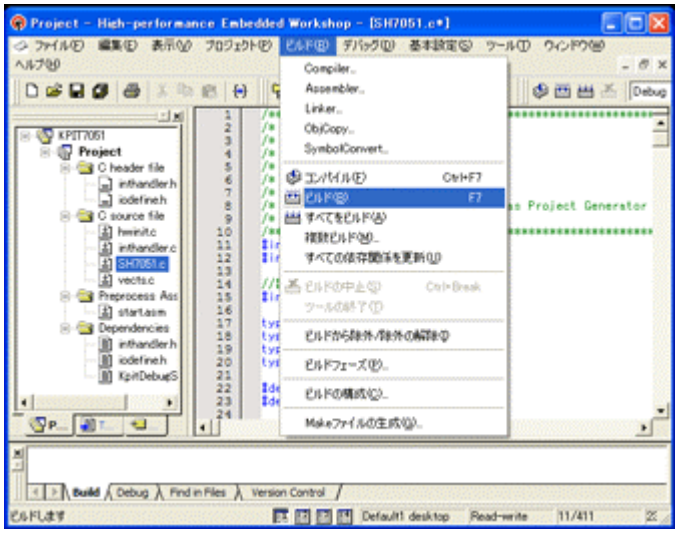


①35行目に「ｍｏｖ．ｌ　ｓｔａｃｋ，ｒ１５」と記述してあります。本来SH-2の場合は、ベクタ１にスタックポインタ値が格納されており、リセット解除時にハード側にて設定されます。
デバッグ途中（スタックポインタが変化後）で、「start.asm」からプログラムを走らせたくなった場合、このソース行があればハードリセットしなくてもスタックポインタは初期化されますので、記述しておいても良いでしょう。
デバッグ終了時でも、このソース行を残しておいても問題ありません。

８．ビルドを実行します。

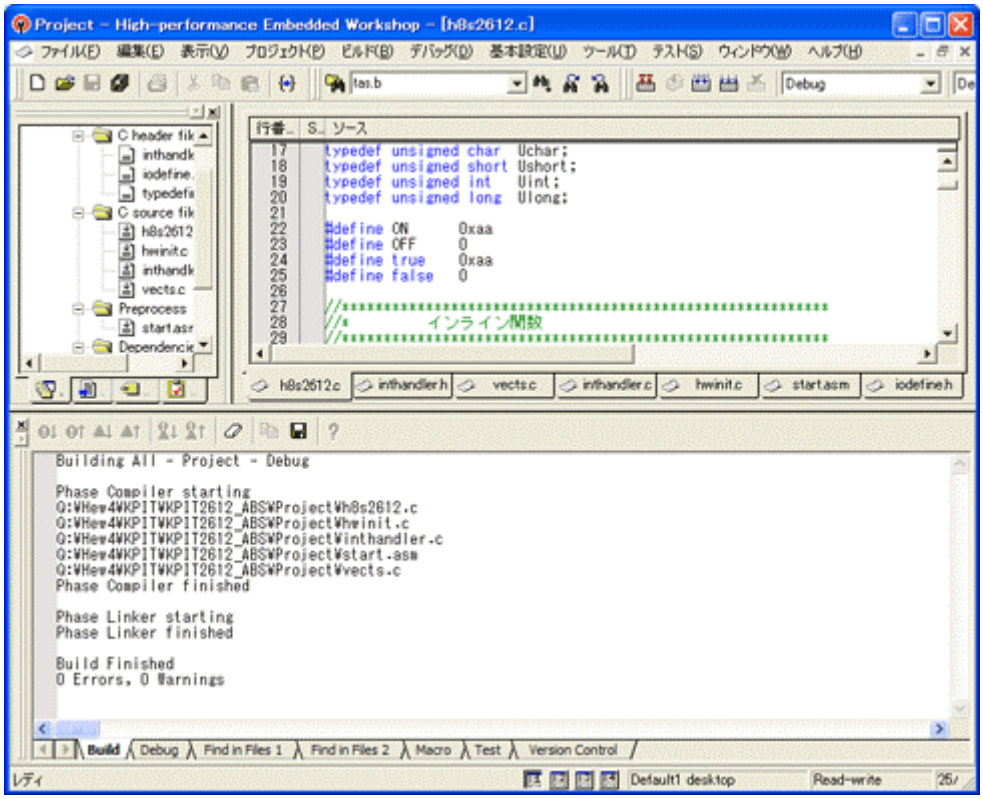
目的：　コンパイル／アセンブリ／リンクロケートを実行させる為、
ビルドを実行します。

[8-1]



[ビルド]-
[ビルド]をクリックします。

[8-2]

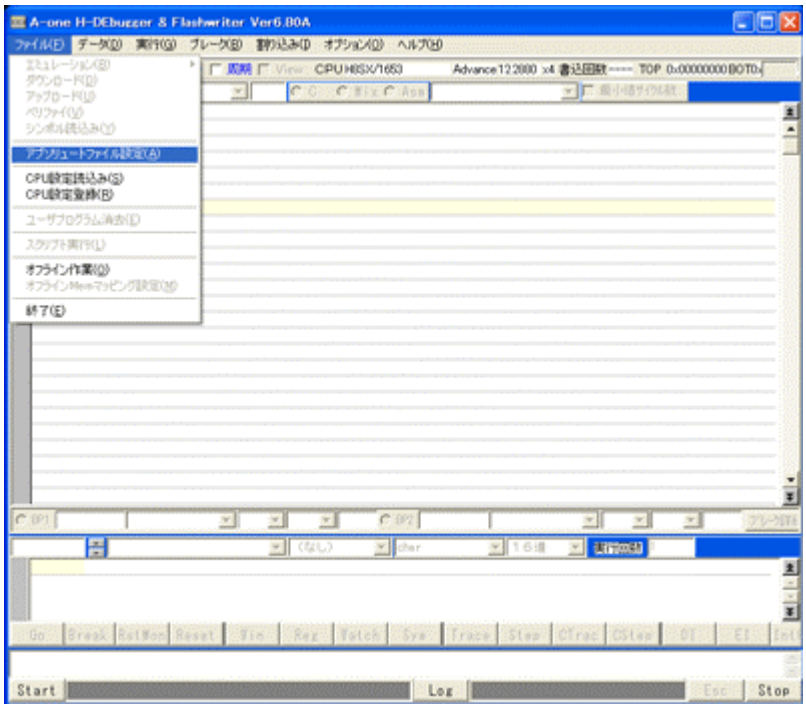


↑のように「0 Errors, 0 Warnings」になれば成功です。

９．ＤＥＦでの確認

１）アブソリュートファイル指定でのダウンロードを指定する。

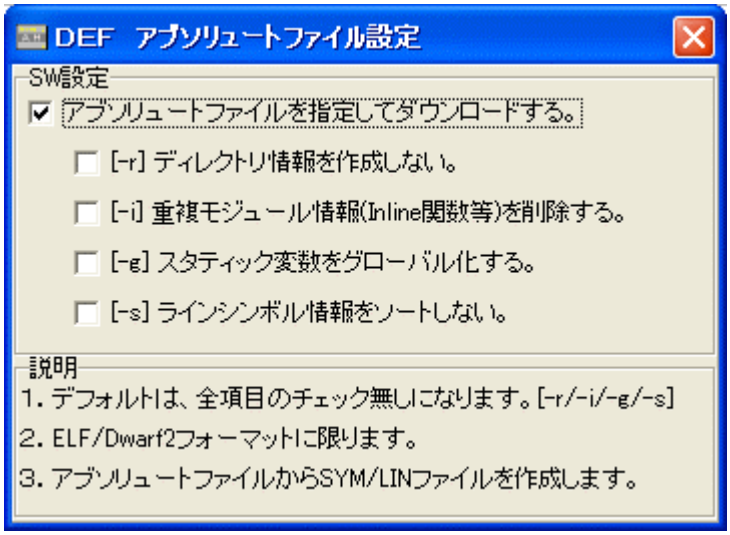
[9-1]



＜ファイル＞ー
＜アブソリュート設定＞
を指定します。

２）設定します。

[9-2]



左画面のように
「チェック　レ」を入れて下さい。

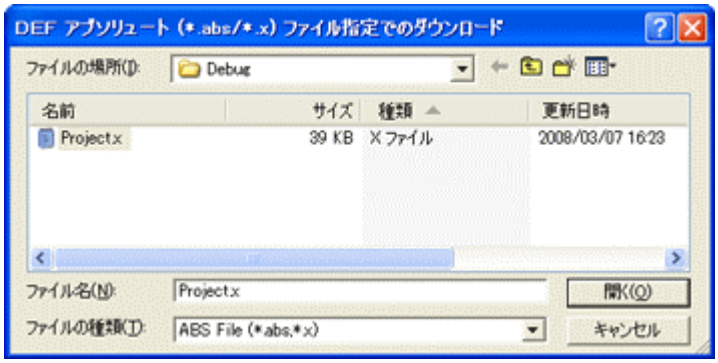
Ｈｅｗデフォルト設定の状態で使用する場合は、
[-r]をチェックしないで下さい。

アブソリュートファイルから直接「*.SYM/*.LIN」ファイルを作成します・

この設定は記憶します。

３）ダウンロードします。

[9-3]



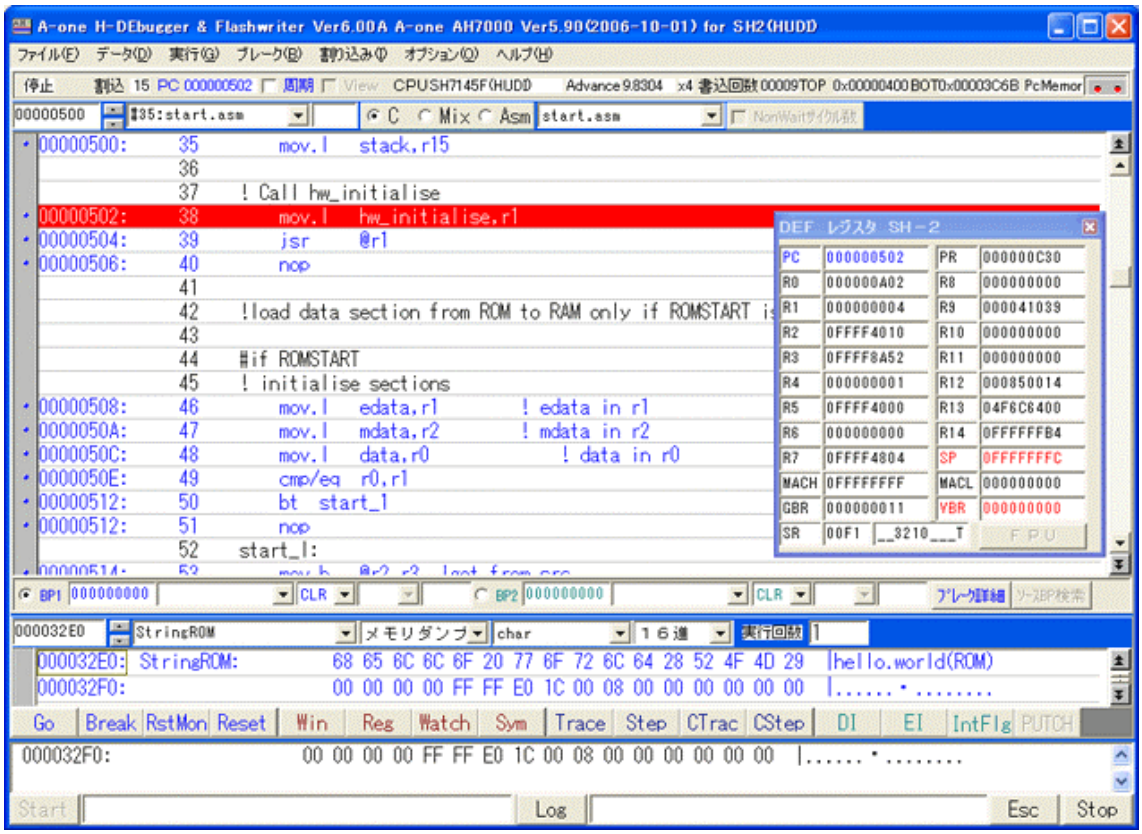
＜ファイル＞ー
＜ダウンロード＞
を指定します。

左画面の通り、ファイル指定後「開く」をクリックしますとダウンロードを開始します。

＜Debug＞ホルダーがデフォルト位置になります。

４）　＜ｓｔａｒｔ．ａｓｍ＞の確認

[9-4]



①５００Ｈ番地にスタックポインタの設定コードがあります。
②ＳＰ値が「0xFFFFFFFC」値になっているのが確認できます。

これで「Ｈ－ｄｅｂｕｇｇｅｒ」用の設定作業が終了です。

以上