特定モジュールを「__INITSCT 関数」の利用によりＲＡＭ側にコピーさせデバッグする場合の説明

Rev1.20
DEF バージョン 6.30A 仕様より
DEF バージョン 7.10A 仕様より

【対象ＣＰＵ】

1）ルネサス C による H8/300H、H8S シリーズ、H8SX シリーズ、SH-2 シリーズが対象となります。

【機能】

1）プログラムを RAM 側にロケートする方法でなく、プログラム実行時に「_INITSCT」により ROM から RAM にプログラムをコピーさせたのち RAM 側でプログラムを実行させる場合のデバッグ例を示します。
2）BSC（バスステートコントローラ）による拡張 RAM でのデバッグになります。
3）PBC/UBC 無しタイプの CPU 品種でもプログラムメモリが RAM の場合、C ソース/Asm ソース上に直接ソフトブレークが張れます。

【デバッグ開始前の準備】

1）BSC（バスステートコントローラ）設定のスクリプトファイルを作成する。

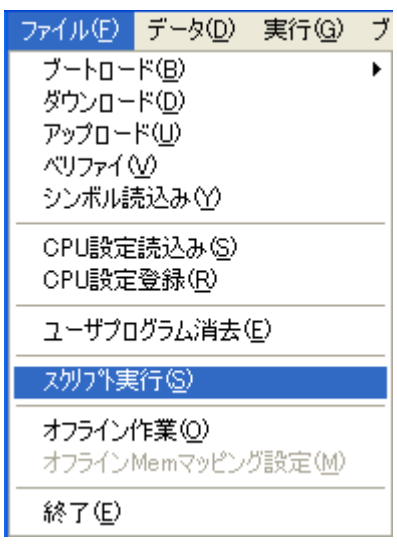
例）ファイル名＜H83069-BSC.log＞

```
//  H8 用(H8/3069F)バスステートコントローラ初期設定
//        エリア 2:SRAM 256Kb 16bit 0x400000
//  コメントは、コマンド実行ラインに記述しないで下さい。

// バス幅コントロールレジスタ  CS2 エリア:16bit
<S  ABWCR  0xfb
// ポート 1 データディレクションレジスタ  A7,A6,A5,A4,A3,A2,A2,A0
<S  P1DDR  0xff
// ポート 2 データディレクションレジスタ  A15,A14,A13,A12,A11,A10,A9,A8
<S  P2DDR  0xff
// ポート 5 データディレクションレジスタ  A19,A18,A17,A16
<S  P5DDR  0xf
// ポート 8 データディレクションレジスタ  CS2 出力端子
<S  P8DDR  0x4
```

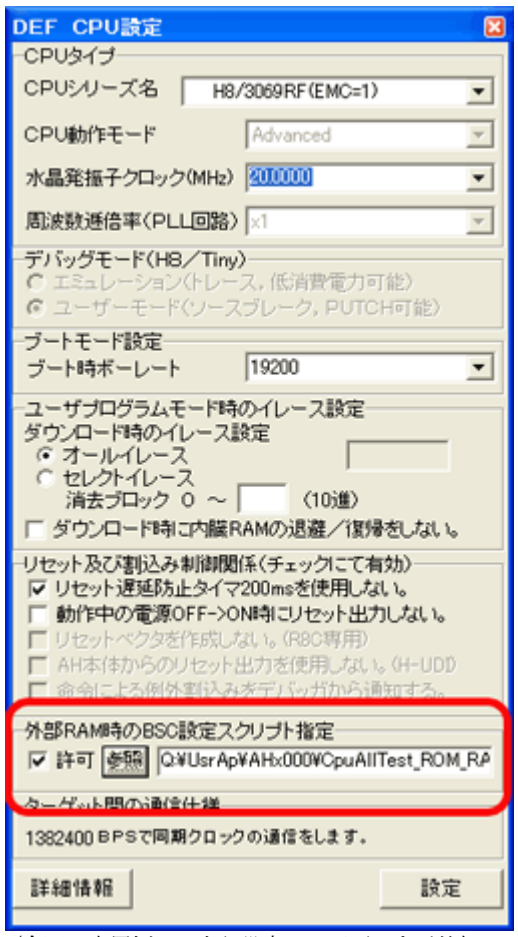
★ＣＰＵ品種用スクリプトファイル例は、ホームページで公開しています。
＜S　｛8 ビットアクセス｝　｛レジスタ名｝　｛データ｝
＜SS　{16 ビットアクセス}
＜SL　{32 ビットアクセス}
〈SQ｛8～32 ビットアクセス｝　＜－　内部登録されているシンボルタイプ（ﾋﾞｯﾄ長）を使用する
//　コメント行
注意　コマンド行には、コメント記述をしないで下さい。

2）作成したスクリプトファイルを実行させ確認をする。＜ファイルメニュ＞



3）スクリプトファイル実行後、拡張 RAM が正しく読み書き出来るか「メモリフィル」等で確認する。

4）作成したスクリプトファイルを CPU 設定に登録する。



（今回の応用例では上記設定は、不要ですが外部 RAM のアクセスが安定し、BSC 設定プログラムの正常動作を確認するまでは設定しておいて下さい）

許可を「チェックレ」後、「参照」PB を押下し、作成したスクリプトファイルを登録する。

5）スクリプトファイル登録による効果

・ユーザプログラムのダウンロード時の、開始と終了後に登録された内容を実行します。（RAM エリアには DL しませんので意味がありません。）

・【RstMon】と【Reset】を実施後、登録された内容を実行します。（BSC 設定プログラムが正常動作したら意味がなくなります。）

6）RAM 側に配置したい特定モジュールにセクション指定をする。

例）ファイル名＜h83069E1.c＞を RAM 側に配置する例です。

```
#pragma section EXT  // <--------  セクション名は、独自だったら何でも良いのですが、仮に「EXT」とする。
//*********************************************************
//      main()
//*********************************************************
main(void)
{
    Ulong    time;

    PBDDR   = 0xff;                         // LED   Port
   .
   .
}
```

7）「_INITSCT」で ROM から RAM にコピーさせる為に定義する。

例）ファイル名＜dbsct.c＞ルネサスＣにより用意されたモジュール

```
#pragma section $DSEC
static const struct {
          _UBYTE *rom_s;       /* Start address of the initialized data section in ROM */
          _UBYTE *rom_e;       /* End address of the initialized data section in ROM   */
          _UBYTE *ram_s;       /* Start address of the initialized data section in RAM */
}DTBL[]= {
          {__sectop("D"), __secend("D"), __sectop("R")},
          {__sectop("PEXT"), __secend("PEXT"), __sectop("PRAM")},    // <----- 追加する。  PEXT->PRAM に Copy する為
          {__sectop("CEXT"), __secend("CEXT"), __sectop("CRAM") }    // <----- 追加する。  CEXT->CRAM に Copy する為
};
```

8）スタートアップ関数に「BSC 設定プログラム」を登録する。

例）ファイル名＜resetprg.c＞　ルネサス C

```
#pragma section ResetPRG
void    SoftWait(short ms);
void    Wait1ms(void);
__entry(vect=0) void PowerON_Reset(void)
{
//  set_imask_ccr((_UBYTE)1);
//
//  H8 用(H8/3069F)バスステートコントローラ初期設定
//       エリア 2:SRAM 256Kb 16bit 0x400000
//
    BSC.BCR.BIT.EMC = 1;            // BCR   EMC=1
// バス幅コントロールレジスタ  CS2 エリア:16bit
    BSC.ABWCR.BYTE = 0xfb;
// ポート 1 データディレクションレジスタ  A7,A6,A5,A4,A3,A2,A2,A0
    P1DDR = 0xff;
// ポート 2 データディレクションレジスタ  A15,A14,A13,A12,A11,A10,A9,A8
    P2DDR = 0xff;
// ポート 5 データディレクションレジスタ  A19,A18,A17,A16
    P5DDR = 0xf;
// ポート 8 データディレクションレジスタ  CS2 出力端子
    P8DDR = 0x4;

    _INITSCT();   // <――ポイント１

//  _CALL_INIT();           // Remove the comment when you use global class object
//  _INIT_IOLIB();          // Remove the comment when you use SIM I/O
//  errno=0;               // Remove the comment when you use errno
//  srand((_UINT)1);        // Remove the comment when you use rand()
//  _s1ptr=NULL;            // Remove the comment when you use strtok()
//  HardwareSetup();        // Remove the comment when you use Hardware Setup
//  set_imask_ccr((_UBYTE)0);

    SoftWait(1);  // <――ポイント２ 1ms Wait(ブート I/F の場合必要-Reset 解除時ソフトタイマ推奨タイプ)
    main();

//  _CLOSEALL();            // Remove the comment when you use SIM I/O
//  _CALL_END();            // Remove the comment when you use global class object
    sleep();
}
```

ポイント１　「_INITSCT()」によって「PEXT －＞ PRAM」と「CEXT －＞ CRAM」がコピーされます。
PRAM/CRAM が外部拡張 RAM に配置している場合は、この「_INITSCT()」関数前に BSC（バスステートコントローラ）の初期化が必要です。

ポイント２　ブート I/F の CPU 品種の場合、リセット解除後 NMI を起動するまで CPU は走行します。**「main()」へ飛ぶまでは ROM 走行が必要ですので必ず「SoftWait(n)」**を入れて下さい。リセット解除の遅延はハードに依存しますので、「SoftWait(n)」で調整して下さい。
ダウンロード時も同じようにリセット->NMI のシーケンスを実行します。

【ソフトタイマが必要な理由】

1．ダウンロード前で RAM エリアが不定な場合、RAM エリアに走行してしまい暴走となり、モニタとの通信に必要な I/O 等が書き換わり正常通信ができなくなり二度と立ち上げることができなくなる可能性があるためです。
2．初期段階ではプログラムのバグによりＲＡＭのプログラムエリアが書き換る可能性があるためです。

ポイント３　「_INITSCT()」により、PRAM にプログラムをコピーしますが、「_INITSCT()」の処理前に PRAM 側にソフトブレークを張っていても上書きされることにより無効になりますので注意して下さい。この場合、ハードブレークのある CPU 品種は、ハードブレークを張って下さい。

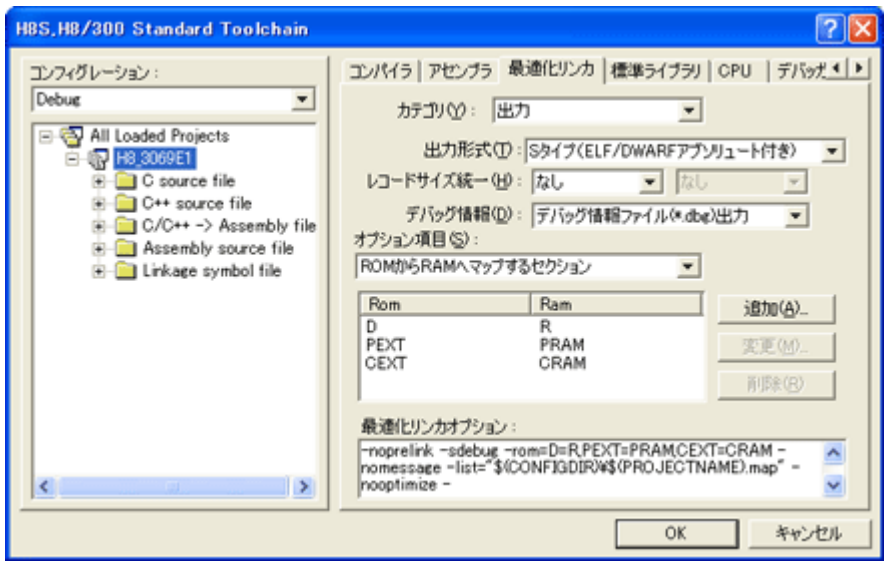
★CPU 品種用スタートアップ関数例は、ホームページで公開しています。

備考

・スタートアップ関数は、ROM 側に配置する必要があります。(SoftWait()関数も必要)
・Hew の場合は、「resetprg.c」内に記述すれば良いかと思います。

【統合環境Ｈｅｗの設定】

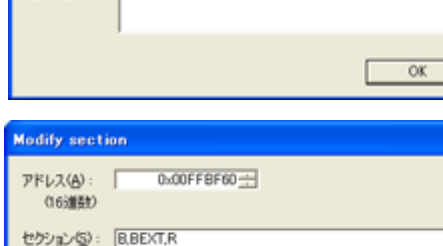
1）Ｈｅｗメニュー＜ビルド＞ー＜.... Standard Toolchain＞の「最適化リンカ」を指定します。



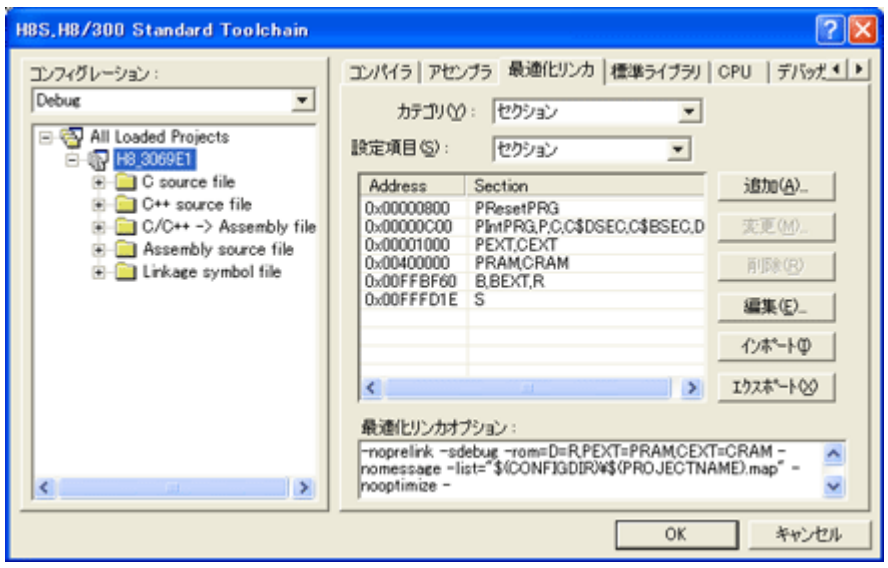
①カテゴリ「出力」を選択する。

②オプション項目「ROM から RAM へマップするセクション」を選択する。

③「追加」クリックにより、
・PEXT(ROM)　から　PRAM(RAM)
・CEXT(ROM)　から　CRAM(RAM)
を追加する。



④左図のように追加します。



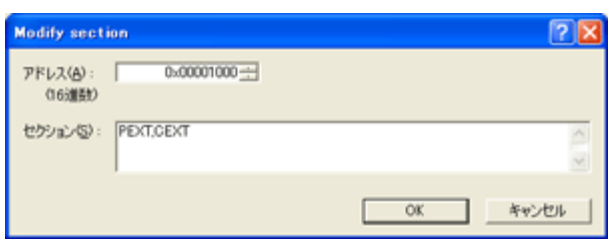
⑤カテゴリ「セクション」を選択する。

⑥「追加」をクリックして、転送元 PEXT/CEXT(ROM)のセクション名とロケーションアドレスを追加します。

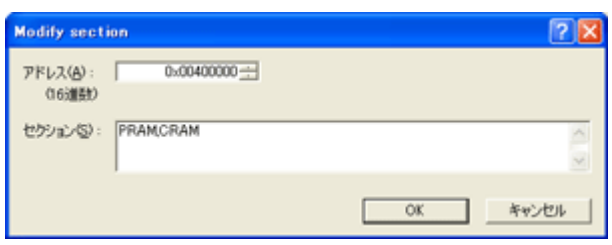
⑦「追加」をクリックして、転送先 PRAM/CRAM(RAM)のセクション名とロケーションアドレスを追加します。

⑧「追加」をクリックして、特定モジュールで派生したＢセクション BEXT を RAM 側のセクションに追加します。

⑨「OK」をクリックします。



PEXT, CEXT の追加例



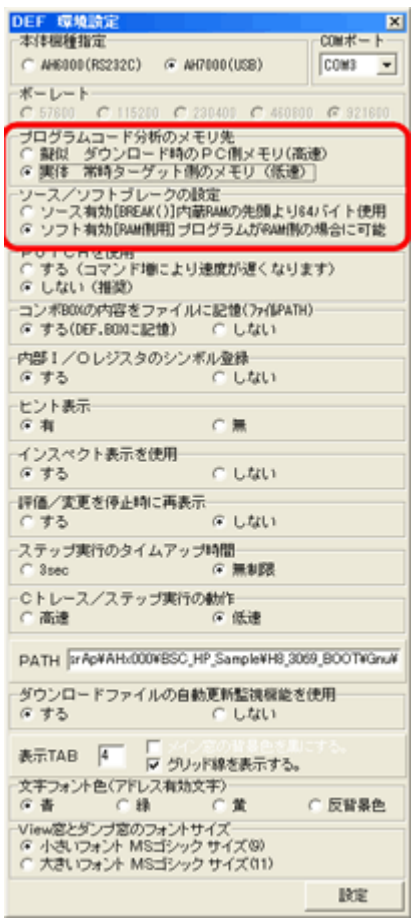
PRAM, CRAM の追加例

BEXT の追加例

【ＨｏｗＴｏ】

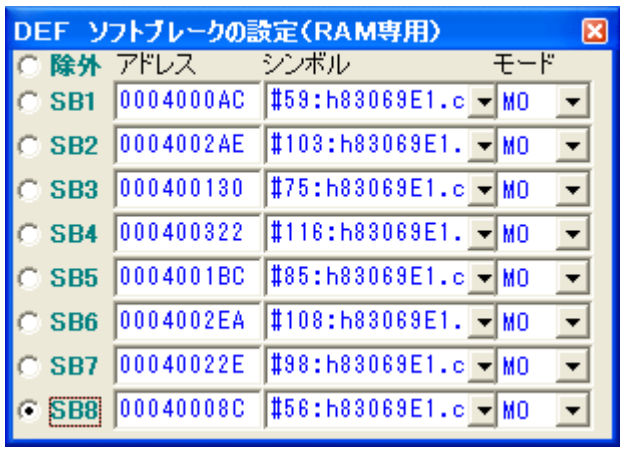
1）プログラムデバッグの初期段階で暴走等の原因により、プログラムのRAMエリアを書き換えてしまうバグが潜んでいる可能性がある場合は【環境設定】の「プログラムコード分析のメモリ先」を「実体」側に指定して下さい。(安定するまでです。)

2）ソフトブレークを有効にする場合は【環境設定】の「ソース／ソフトブレーク設定」を「ソフト有効」側に指定して下さい。



【ソフトブレーク設定】

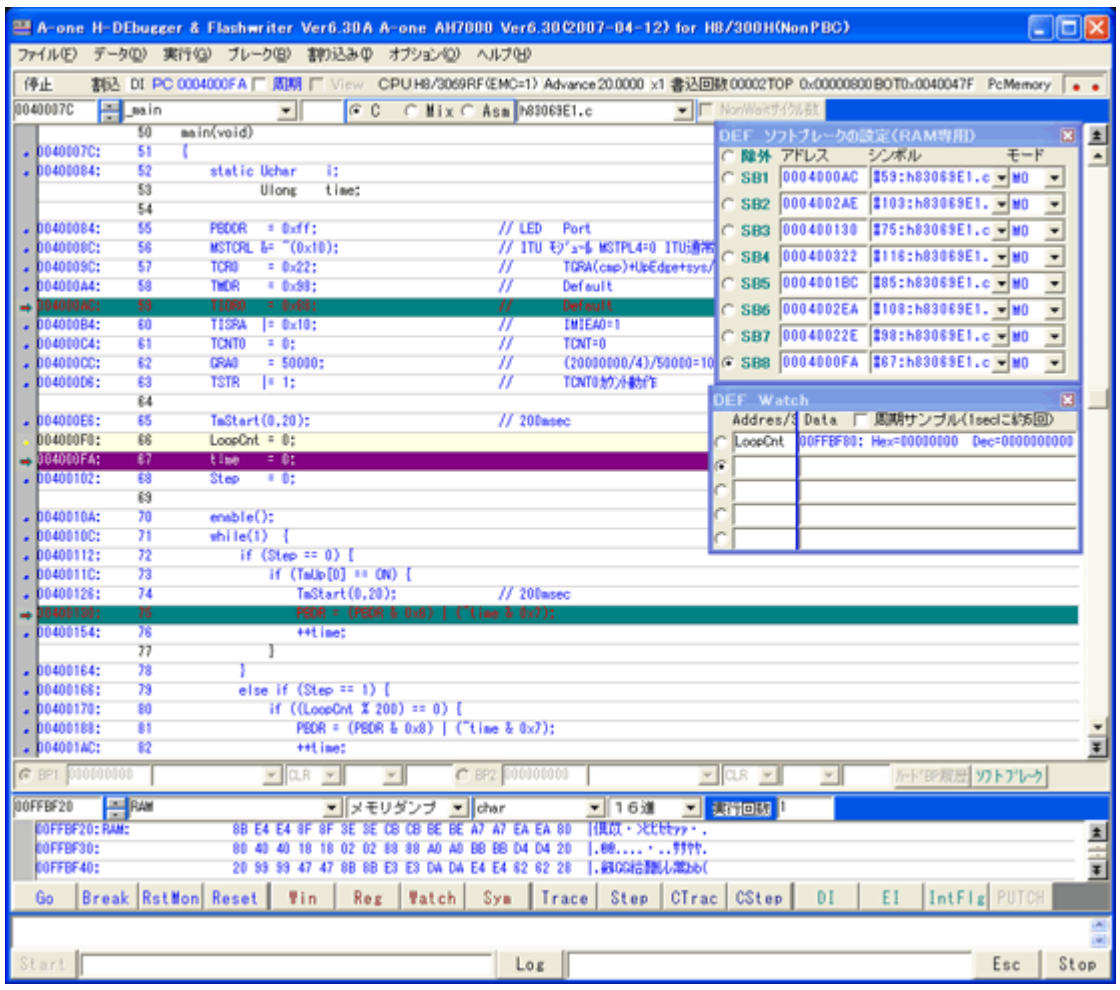
1）RAMに配置したプログラムのみにソフトブレークを直接設定が出来ます。



・「SB1->SB8」を選択してから、CView画面上で「ダブルクリック」しますとソフトブレークの設定が出来ます。
・「除外」を選択しますと、CView画面上で「ダブルクリック」がソフトブレーク設定から除外されます。ハードブレークを設定する場合に選択して下さい。

【ソフトブレーク設定画面例】

1）DEF にて「ソフトブレーク設定」をした画面です。



備考

・PBC 無しタイプの CPU の場合でも、プログラムが RAM での実行時は【Trace/Step】が可能になります。

・プログラムを内蔵 RAM 側に配置した場合でも同じく機能します。（この場合は BSC 設定は不要です）

以上