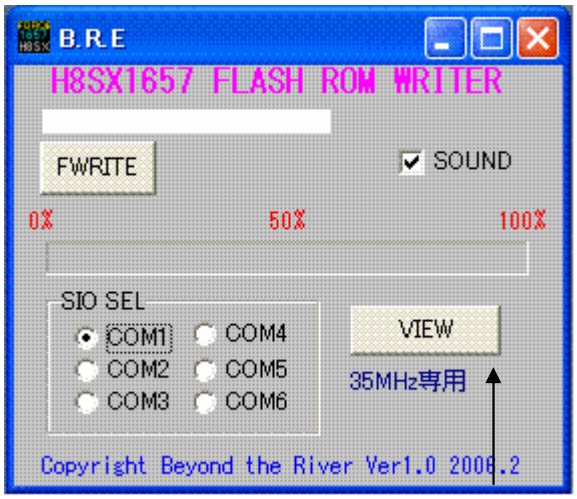
■デバックライブラリの使い方　H8SX／1657編　　2006．2．21

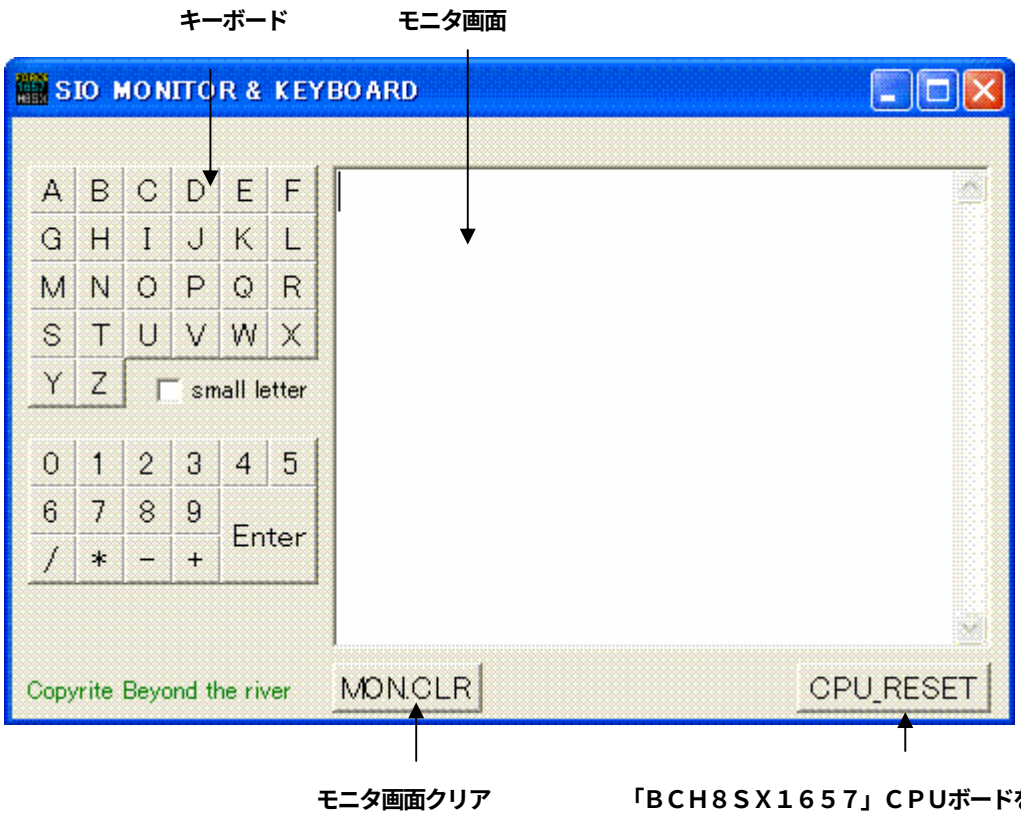
　HEW4で製作したプログラムを弊社CPUボード「BCH8SX1657」に書き込んで動作させることは「フォースライタ」で実現できるわけですが、プログラムデバック中に変数、レジスタの値を見たり、データをセットしたりしてプログラムが思った通りに動作しているかどうか確認したいときがあります。

　高度なデバックはルネサス純正のE－8、E－10等で可能なわけですが、特別なハードウエアを必要としない簡便なデバッカがあると便利です。それを行うのが「フォースライタ」に付属する機能「フォースビュー」です。「フォースビューは」Eケーブルを経由して「BCH8SX1657」CPUボードからパソコンに出力したデータを表示するモニタ画面と、パソコンからCPUボード側にデータを出力するキーボードで構成されます。デバック用ライブラリ「DBH8SX」と併せてスマートなデバック環境を構築できます。



ここをクリックしてください。下記の「フォースビュー」画面が現れます。通常、デバック中は両画面共表示して使用します。

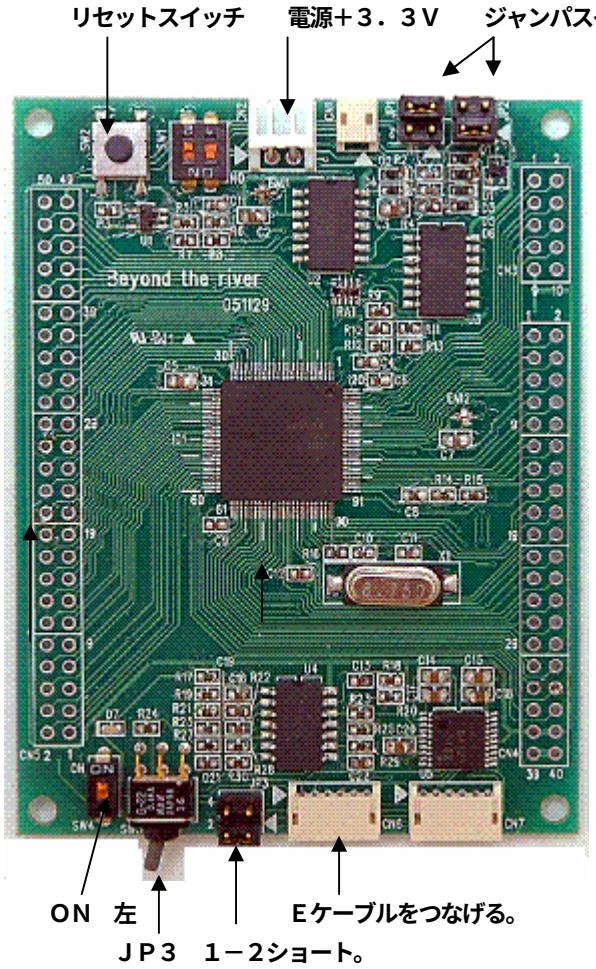
●「フォースビュー」各部の機能と名称



●事前準備

「フォースビュー」機能を使用するためには事前準備が必要です。「フォースライタ」の事前準備と同じです。
ＣＰＵボード「ＢＣＨ８ＳＸ１６５７]、Ｅケーブル、電源等ハードウエアの用意が必要です。

●「ＢＣＨ８ＳＸ１６５７」　ＣＰＵボードの設定　モード６　アドレスバス、データバスアクティブ、外部ＲＡＭ使用の場合。



●パソコンとＣＰＵボード間の接続を確認します。

前提条件として、
１．先に指定したパソコンのＣＯＭポートからＣＰＵボードにＥケーブルが接続されている必要があります。
２．上記までできましたら電源を入れてください。赤いＬＥＤは点灯しません。「フォースライタ　Ｈ８ＳＸ／１６５７」の操作により
フラッシュＲＯＭ書き込み時は点灯、プログラム動作時は消灯します。

●デバックのためのライブラリ「ＤＢＨ８ＳＸ」

デバックのための変数やレジスタ＊１を見るライブラリが弊社より用意されています。「フォースライタ」ダウンロード時に添付してくるＤｅｂｕｇフォルダ中ファイル「ＤＢＨ８ＳＸ．ｏｂｊ」と「ＤＢＨ８ＳＸ．ｈ」です。ＨＥＷ４でプログラム開発を行う時にユーザープログラム中でこれらの中の関数を呼んでいただけば結果がモニタ画面に表示されたり、ポート、データの設定がキーボードからできます。
この関数は従来のよくある方法の欠点＊２を検討して作成されており、シンプルにして強力な関数です。

＊１
ここでいうレジスタとはＨ８ＳＸ　ＣＰＵの内部レジスタ　例Ｒ０Ｈ　という意味でなく、Ｈ８ＳＸ／１６５７内蔵の　例ＡＤＤＲＡＡ／Ｄデータレジスタ Ａという意味です。例えばポートに入出力するのもレジスタの操作です。

＊2

プログラムのデバックでデバック時と完成時にメモリマッピングが異なる方式だと、最終的にＲＯＭ化するときに問題が発生するケースがあります。「フォースビュー」を使ったデバックではメモリマッピングはデバック中も完成時も同じです。

実際の内部ＲＯＭ以外に書き込んでデバックする方式ではデバック中と完成時にプログラム動作速度が異なり、問題が発生するケースがあります。「フォースビュー」を使ったデバックでは動作速度はデバック中も完成時もほぼ同じか、まったく同じです。

高機能なデバック環境はそれを使いこなすまでの時間が無視できません。また、高機能なものほどデバック環境自身にバグがあった場合、ユーザーには判断がつきにくいものです。「フォースビュー」を使ったデバックではイニシャル１個と、ｓｅｔ、ｇｅｔ、ｃｏｍの３種類の関数しか使用しません。データは８、１６、３２ビットのアンサイン、サインを扱えます。

●関数

以下は使用できる関数です。

```
void init_DBH8SX_M6(void);                  //イニシャルプログラムです。ｍａｉｎ関数の初めに入れてください。

//データはアンサインがベースです。

void set_b(char [],unsigned char *);        //８ビットの変数、レジスタにキーボードよりデータをセットします。
void get_b(char [],unsigned char *,char );  //８ビットの変数、レジスタの内容をモニタ画面に表示させます。

void set_w(char [],unsigned short *);       //１６ビットの変数、レジスタにキーボードよりデータをセットします。
void get_w(char [],unsigned short *,char ); //１６ビットの変数、レジスタの内容をモニタ画面に表示させます。

void set_l(char [],unsigned long *);        //３２ビットの変数、レジスタにキーボードよりデータをセットします。
void get_l(char [],unsigned long *,char);   //３２ビットの変数、レジスタの内容をモニタ画面に表示させます。

//データでサインを使用したい場合。

void set_bs(char [],char *);
void get_bs(char [],char *,char );

void set_ws(char [],short *);
void get_ws(char [],short *,char );

void set_ls(char [],long *);
void get_ls(char [],long *,char);

//コメントを表示させる。

void  comm（char[]）;                       //コメントを表示させる。
```

■データの設定

ｖｏｉｄ　ｓｅｔ＿ｘ（ｃｈａｒ　［］、ｕｎｓｉｇｎｅｄ　ｃｈａｒ　＊）　ｘ：ｂ：バイト、ｗ：ワード、ｌ：ロングワード

プログラムがこの関数に到達し、
キーボード入力を促すことを示
すためにコメントを６文字まで記入できる

設定したい変数、レジスタ名を入れる。
前に必ず＆を付けてください。

■データの読み込み

```
void get_x(char [ ],unsigned char *,char z)
```

x：b：バイト、w：ワード、l：ロングワード

プログラムがこの関数に到達し、
見たい変数、レジスタ名の表示の前に
コメントを６文字まで記入できる

見たい変数、レジスタ名を入れる
前に必ず＆を付けてください。

表示した後、次に行くか、キーボード入力を待つか
ｚ：０　無条件に次の行にいく
ｚ：１　キーボード入力があるまで待つ

■コメントの表示

コメントをフォースビュー画面に表示させます。最後に改行を入れるとそこまで、いれないと半角換算５０文字まで表示します。
例
```
void comm(”カウンタの値を見る￥ｎ”);
```

●使用例１　set_b

1バイトのデータをキーボードより変数ｃｆにセットし、その後のプログラムの動作をみてみます。

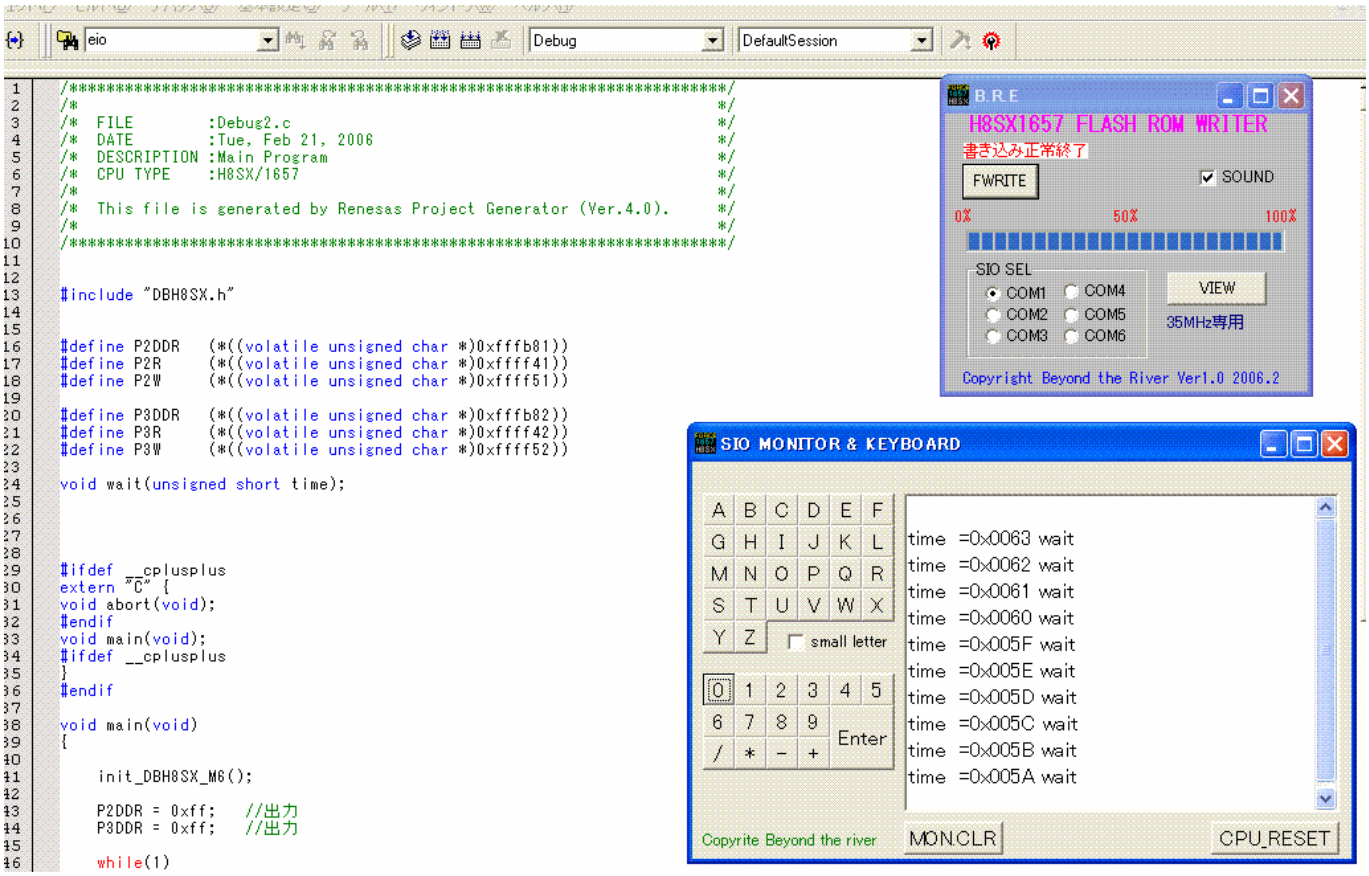
```
void main()
{
unsigned char cf;

        init_DBH8SX_M6();           //イニシャル
        P3DDR  =  0xff;             //ポート3を出力にセット

        comm("P3から設定値を出力します。2桁設定してください\n");     //コメントを表示
        set_b("set_cf",&cf);        //キーボードより2桁入力まち。変数ｃｆにセットされる。
        P3W  = cf;                  //キーボードで設定された数値がｃｆに入りP3から出力される。
```

上記プログラムを「フォースライタ」で書き込み実行すると、「フォースビュー」には次のように表示されます。

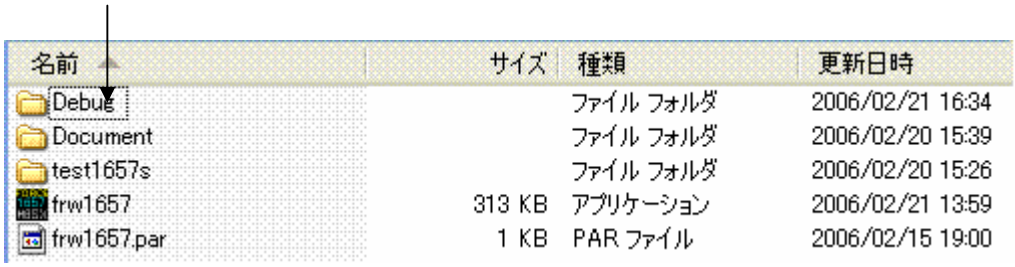
①モニタ画面　コメントＰ３から、、、ｓｅｔ　ｃｆはプログラムがｓｅｔ_ｂ（”ｓｅｔ　ｃｆ”, &ｃｆ）；まで到達したことを示して

います。
②例えばキーボードでＦを２回クリックするとｃｆに０ｘＦＦが設定されます。
③クリックされたキーボードデータの確認のために＝０ｘｎｎと表示されます。この時点でプログラムは次の行に進行しています。
④P3W　= cf とありますので、キーボードで設定した０ｘＦＦがポートより出力されます。
⑤プログラムは再びここに来た場合、同じようにキーボード待ちになります。

●ＨＥＷ４での具体例
　ＨＥＷ４でデバックライブラリ「ＤＢＨ８ＳＸ１６５７」を使ってみます。



　例としてＰ２，３ポートをＯＮ，ＯＦＦさせるプログラムを作成し、ウエイト時間の変数が思い通りに変化するか見てみます。
ダウンロードしたＨ８ＳＸ１６５７の中にＤｅｂｕｇというディレクトリで一式用意されていますので、まるまるコピーしてＨＥＷ４で開くためには￥Ｗｏｒｋ Ｓｐａｃｅ￥等に移動させます。



　ＨＥＷ４を開いて「最近使用したプロジェクトワークスペースを開く」で例として
C:\WorkSpace\Debug\Debug.hws 迄設定し、「ＯＫ」をクリックします。これによりＨＥＷ４で修正、コンパイルできる状態になります。Ｄｅｂｕｇフォルダを WorkSpace 以外に設置しても Debug.hws を設定できればＯＫです。

以下は全プログラムです。赤の部分がユーザーが書いた部分、それ以外はＨＥＷ４が自動生成する部分です。

```
/************************************************************************/
/*                                                                      */
/*  FILE        :Debug.c                                                */
/*  DATE        :Tue, Feb 21, 2006                                      */
/*  DESCRIPTION :Main Program                                           */
/*  CPU TYPE    :H8SX/1657                                              */
/*                                                                      */
/*  This file is generated by Renesas Project Generator (Ver.4.0).      */
/*                                                                      */
/************************************************************************/


#include "DBH8SX.h"


#define P2DDR       (*(volatile unsigned char *)0xfffb81)
#define P2R         (*(volatile unsigned char *)0xffff41)
#define P2W         (*(volatile unsigned char *)0xffff51)

#define P3DDR       (*(volatile unsigned char *)0xfffb82)
#define P3R         (*(volatile unsigned char *)0xffff42)
#define P3W         (*(volatile unsigned char *)0xffff52)

void wait(unsigned short time);

#ifdef __cplusplus
extern "C" {
void abort(void);
#endif
void main(void);
#ifdef __cplusplus
}
#endif

void main(void)
{
        init_DBH8SX_M6();

        P2DDR = 0xff;       //出力
        P3DDR = 0xff;       //出力

        while(1)
        {
                P2W = 0;
                P3W = 0;
                wait(100);
                P2W = 0xff;
                P3W = 0xff;
                wait(100);
```

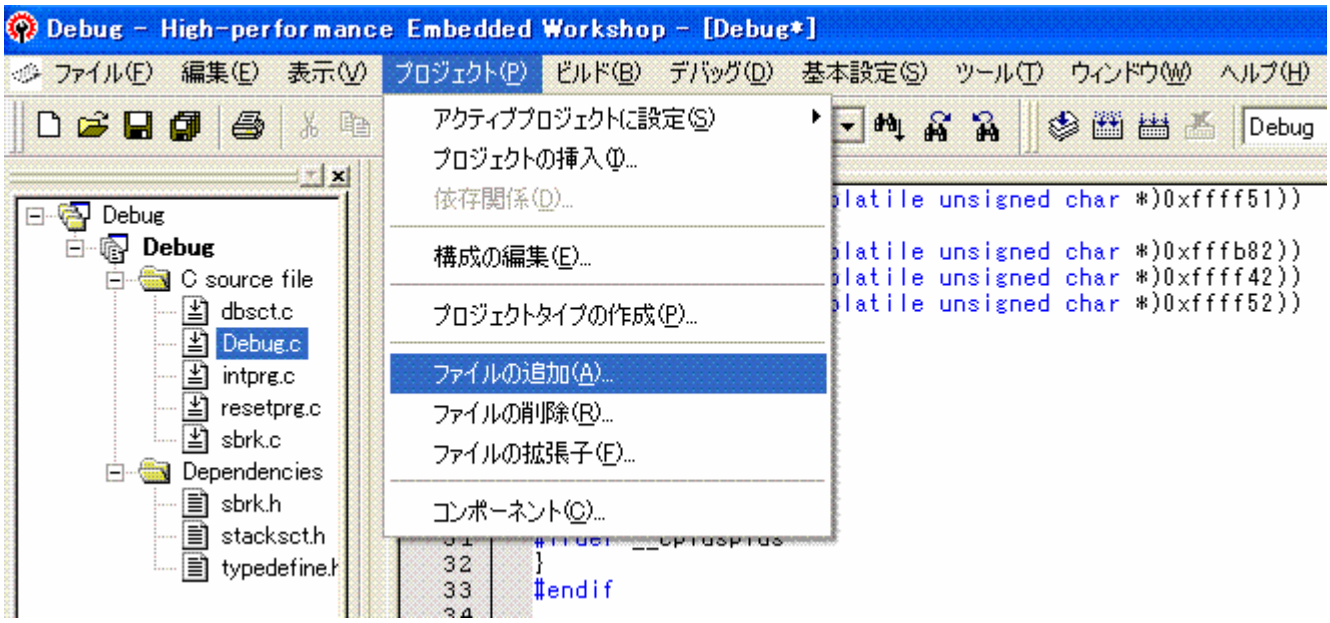
```
        }
}

void wait(unsigned short time)
{
        while(time != 0)
        {
                time--;
                get_w("time",&time,1);
        }

}

#ifdef __cplusplus
void abort(void)
{

}
#endif
```

①ＤＢＨ８ＳＸ．ｈとＤＢＨ８ＳＸ．ｏｂｊのありか

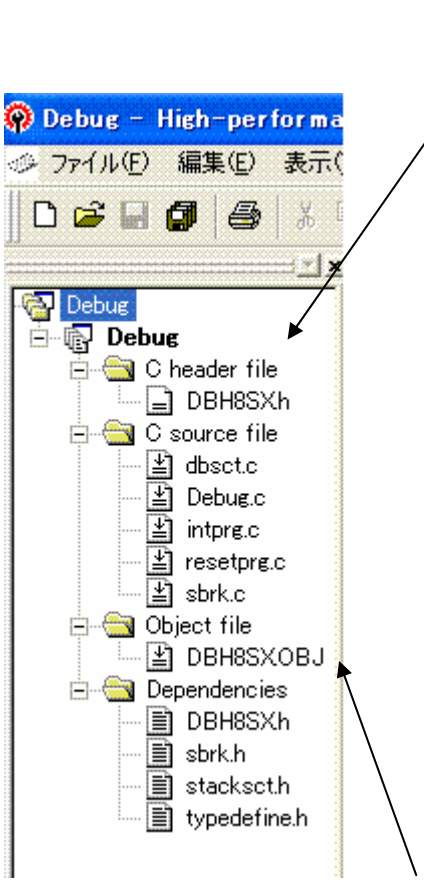
ＤＢＨ８ＳＸ．ｈ　　は右記の場所にあります。C:\WorkSpace\Debug\Debug
ＤＢＨ８ＳＸ．ｏｂｊ は右記の場所にあります。C:\WorkSpace\Debug\Debug\Debug

ご自身で製作されたプロジェクトに上記２点を移す場合も上記場所を参考にしてください。この場合、ディレクトリに移した後、ＨＥＷ４に２つのファイルを追加、認識させる必要があります。

プロジェクト（Ｐ）->ファイルの追加（Ａ）で２つのファイルを１つずつクリックします。



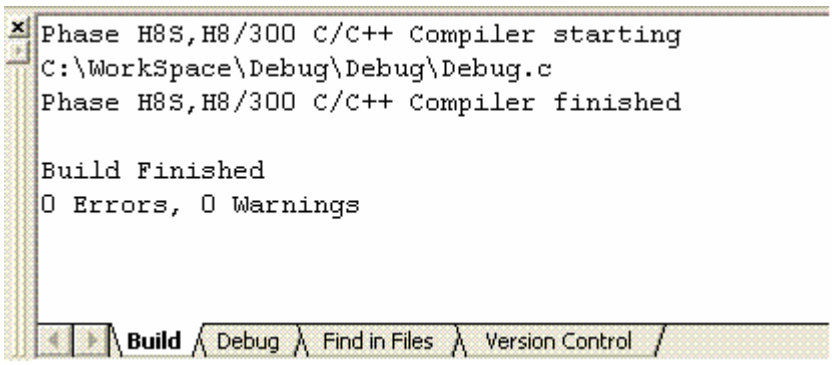
Ｃ　ｈｅａｄｅｒ　ｆｉｌｅ　として「ＤＢＨ８ＳＸ．ｈ」が組み込まれ、



Ｏｂｊｅｃｔ　ｆｉｌｅ　として「ＤＢＨ８ＳＸ．ｏｂｊ」が組み込まれればＯＫです。添付Ｄｅｂｕｇフォルダは上記作業は必要ありません。認識済です。

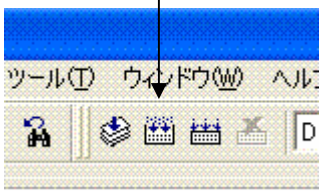
③動作させてみます。

コンパイルします。ここをクリック



エラー、警告とも無い場合、ビルドでＲＯＭ化可能ファイル「Ｄｅｂｕｇ．ｍｏｔ」を作成します。

ここがビルド。

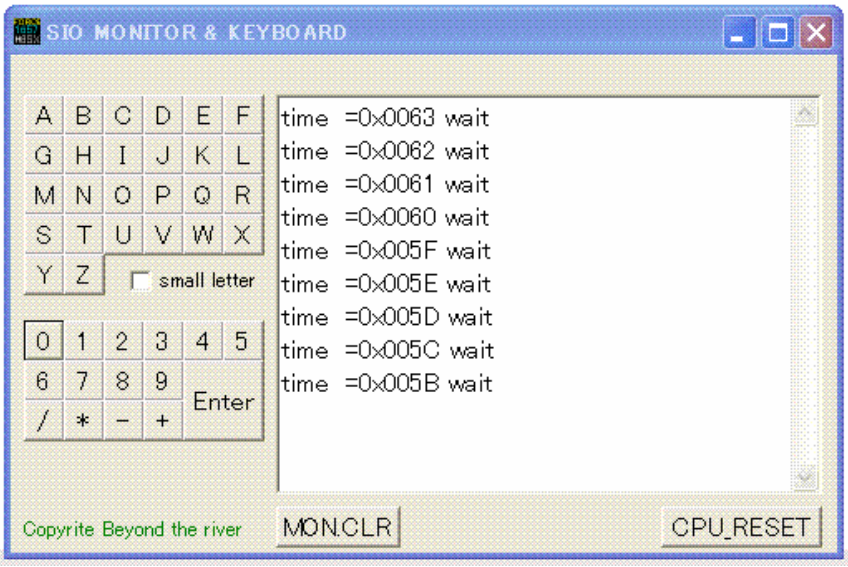


エラーが無ければＯＫです。警告は１つまではＯＫです。１番初めのビルドはパソコンが壊れたかと思うような黒い画面と時間がかかる場合があります。２回目から短縮されますので、数分は待ってみてください。

「ＦＷＲ１６５７」を動作させます。ＨＥＷ４と並動させます。フォースライタはデバック中、閉じないでください。
「ＦＷＲＩＴＥ」をクリックし、C:\WorkSpace\Debug\Debug\Debug\Debug.mot を焼きます。

デバック中は閉じないでください。

```
/************************************************************************/
/*                                                                      */
/*  FILE        :Debug.c                                                */
/*  DATE        :Tue, Feb 21, 2006                                      */
/*  DESCRIPTION :Main Program                                           */
/*  CPU TYPE    :H8SX/1657                                              */
/*                                                                      */
/*  This file is generated by Renesas Project Generator (Ver.4.0        */
/*                                                                      */
/************************************************************************/


#include "DBH8SX.h"


#define P2DDR   (*(volatile unsigned char *)0xfffb81))
#define P2R     (*(volatile unsigned char *)0xffff41))
#define P2W     (*(volatile unsigned char *)0xffff51))

#define P3DDR   (*(volatile unsigned char *)0xfffb82))
#define P3R     (*(volatile unsigned char *)0xffff42))
#define P3W     (*(volatile unsigned char *)0xffff52))

void wait(unsigned short time);

#ifdef __cplusplus
extern "C" {
void abort(void);
#endif
void main(void);
#ifdef __cplusplus
}
#endif

void main(void)
{
    init_DBH8SX_M6();

    P2DDR = 0xff;   //出力
    P3DDR = 0xff;   //出力

    while(1)
    {
        P2W = 0;
        P3W = 0;
        wait(100);
        P2W = 0xff;
        P3W = 0xff;
        wait(100);
```



焼き終わるとＳＩＯ　ＭＯＮＩＴＯＲに「ｔｉｍｅ＝０ｘ００６３」　ｗａｉｔと表示されました。これはプログラムがｇｅｔ_ｗｓの所まで到達し、ｔｉｍｅ＝０ｘ００６３（９９Ｄ）である、ということを示します。ｗａｉｔモードですから、ＫＥＹＢＯＡＲＤでなにか１文字クリックするとプロフラムは次に行きます。それまでは入力待ちです。適当にキーボードをクリックするとｔｉｍｅの値が１つずつ減算されていくのが確認できます。

メインプログラムは　ｗａｉｔ（１００）をはさんだＰ３Ｗ、Ｐ２Ｗ＝０と０ｘｆｆの無限ループです。

```
        while(1)
        {
                P2W = 0;
                P3W = 0;
                wait(100);
                P2W = 0xff;
                P3W = 0xff;
                wait(100);

        }
}

void wait(unsigned short time)
{
        while(time != 0)
        {
                time--;
                get_w("time",&time,1);
        }

}
```

ｗａｉｔ(ｕｎｓｉｇｎｅｄ　ｓｈｏｒｔ　ｔｉｍｅ)のｔｉｍｅは１００、９９、９８、９７、、と減算されていくはずです。ＫＥＹＢＯＡＲＤでなにか１文字クリックするとプロフラムは次に行き、ＳＩＯ　ＭＯＮＩＴＯＲ　にｇｅｔ＿ｗのところにくるたびにｔｉｍｅのデータを表示していきます。０ｘ００６３、０ｘ００６２、０ｘ００６１、、、と順次減算されていくのが分かります。ｔｉｍｅ＝０になるとＰ２、Ｐ３のレベルが変化し、再び０ｘ００６３、、、と減算していきます。

例ではｔｉｍｅの数値でしたが、同様にポートのデータの入出力、レジスタの読み込み、書き込みがｓｅｔ、ｇｅｔ関数を使って可能です。例えばＡＤＤＲＡを読めばＡ／ＤデータＡＮ０の値が読めます。（イニシャルは必要）

以上が弊社が提供している簡易デバックプログラム「ＤＢＨ８ＳＸ」の機能と使い方です。考え方はパソコンのプログラム開発でよく行われるｐｒｉｎｔｆデバックに限りなく近いです。加えてデータ設定ができるところが便利です。

●諸注意

①無限ループの中で短時間でパソコン側に出力を繰り返すようなプログラムは対応できません。理由は常に　受信データ数　＞　表示数の関係では受信バッファが増加していって最後に設定数をオーバーフローしてしまうためです。

例１

```
while (1)
 {
        get_ws("time  ",&time,0);       //ｔｉｍｅが連続してパソコン側に出力される
 }
```

もし仮に上記のようなプログラムを書いて実行しますと、たいがい「ＦＷＲ３６７２」は消えます。何回立ち上げても消える場合、受信バッファオーバーフローです。ＣＰＵボードの電源を切り、プログラム中の問題のコマンドを削除するか、ｗａｉｔ付に変更してコンパイルします。再び電源投入、書き込みで直ります。プログラムを書き換えない限り直りません。

②ｓｍａｌｌ　ｌｅｔｔｅｒをチェックするとキーボードは小文字を出力します。但し、ｓｅｔ関数は小文字には対応していません。データ設定は大文字を使用してください。例　ｆはＮＧ　ＦはＯＫ

③データ設定時、０～９、Ａ～Ｆ以外のデータをクリックした場合、結果は不定です。

④デバック終了時はコメント文にして影響させないようにします。　例　// get_b("\ncf　",&cf,0);

⑤デバック中「フォースライタ」は常時動作させてください。フォースライタを終了しますとＣＰＵボードも動作停止します。

⑥「フォースライタ」は動作中、ＣＯＭポートを占有します。他のアプリケーションソフトが同一のＣＯＭポートを使用することはできません。必要であれば別ファイル「ＣＯＭポートについて」等をご参考に増設してください。

ご注意

■ＨＥＷⓇは株式会社ルネサステクノロジ社の登録商標です。

■ＷｉｎｄｏｗｓⓇは米国マイクロソフト社の登録商標です。

■「フォースⓇ」機能、「フォースⓇライタ」「フォースⓇビュー」は弊社で知的財産権申請、登録されています。

１．本文章に記載された内容は弊社有限会社ビーリバーエレクトロニクスの調査結果です。

２．本文章に記載された情報の内容、使用結果に対して弊社はいかなる責任も負いません。

３．本文章に記載された情報に誤記等問題がありましたらご一報いただけますと幸いです。

４．本文章は許可なく転載、複製することを堅くお 断りいたします。

〒350-1213　埼玉県日高市高萩 1141-1

TEL　042（985）6982　FAX　042（985）6720
Homepage：http//beriver.co.jp
e-mail：support@beriver.co.jp
有限会社ビーリバーエレクトロニクス