ｓａｍｐｌｅ８ｐ【　ベータ版　２００６／８／２６　】
**ｐｒｉｎｔｆデバック１：　扱えるデータ型、数値範囲**
**ＢＣＨ８３０４８ＯＮＥ、ＢＣＨ８３０５２用**

有限会社ビーリバーエレクトロニクス

**【　用途　】ＢＣＨ８３０４８ＯＮＥ、ＢＣＨ８３０５２開発セット向け**

**【　プログラム概要　】**

char、int、long、float のデータを、printf コマンドで SIO モニタ（フォースビュー）に表示する。

**【　概要　】**

　ＳＩＯモニタ画面に printf 関数で数値、変数、レジスタ、メモリ、ポートなどの内容をを表示するライブラリ printf_H8.h をリリースしました。

　一般に、ＧＣＣ添付の printf 関数は消費メモリが大きく、組み込み機器にそのまま使用できることは稀でした。そこで、コンパクトで、ある程度機能を満たした printf 関数があれば組み込み機器のデバックもはかどると考えられ、弊社で開発しリリースしたものです。

サイズは６Ｋバイト弱で組み込んでも負担にならないサイズだと思います。

**【　printf_h8.o 仕様　】**

printf 関数は以下のデータ形、フォーマット指定子等が使用できます。

| 扱えるデータ形 | フォーマット指定子 | サイズ（ビット） | 扱える数値範囲<br>１０進数（１６進数） |
|---|---|---|---|
| ｃｈａｒ | ％ｓ　文字列 | ８ | －１２８から１２７<br>（０ｘ８０～０ｘ７ｆ） |
| | ％ｃ　１文字 | | |
| ｓｈｏｒｔ、ｉｎｔ | ％ｄ　１０進数 | １６ | －３２、７６８～３２、７６７<br>（０ｘ８０００～０ｘ７ｆｆｆ） |
| | ％ｘ　１６進数 | １６ | ０～０ｘＦＦＦＦ |
| ｌｏｎｇ | ％ｌｄ　１０進数 | ３２ | －２１４７４８３６４８～<br>２１４７４８３６４７<br>（０ｘ８０００００００～<br>７ｆｆｆｆｆｆｆ） |
| ｆｌｏａｔ | ％ｆ　浮動小数点１０進数 | ３２<br>※１ | ±３２７６６.９９９９程度<br>※２ |

※１　ｄｏｕｂｌｅもｆｌｏａｔと同じ指定子表示できますが、範囲はｆｌｏａｔと同じです。

※２　浮動小数点の演算は仮数のビット長が有限であるため「情報落ち」誤差が生じる可能性があります。また、桁落ち誤差も発生する場合があります。誤差を意識した使用が求められます。

※使用できない表記

０２％ｄのような出力桁指定表記、％ｇべき乗表示、他です。

## 【　printf 関数を使用するにあたり追加すること　】

sample8p 以外で printf 関数を使用する場合、以下の追加が必要です。
①sample8p ディレクトリ内にある printf_h8.h、printf_h8.o ファイルを開発するソースファイルと同じディレクトリに置いてください。
②開発するソースファイルに　#include "printf_H8.h"　と表記してください。
③printf_h8.h 内で定義されている関数名は「print_sio」なので、printf として使用する場合、
#define printf print_sio
と、上記デファイン文で定義してください。後述ご参考。
④xxx.cmd ファイルで以下の２箇所の赤の部分を追加してください。

```
REM Beyond the river 20060420
REM RAM 動作

REM del *.o
del *.mot

h8300-hms-gcc -mh -c start0.s -o start0.o
h8300-hms-gcc -mh -c -g -Wall -O sample8p.c -o sample8p.o
h8300-hms-gcc -mh -g -mint32 -nostartfiles -o sample8p.hms start0.o sample8p.o printf_H8.o
-Th8_3052_rom.x
h8300-hms-objcopy -O srec sample8p.hms sample8p.mot
h8300-hms-objdump -S sample8p.hms > sample8p.lst
h8300-hms-objdump -h sample8p.hms > sample8p.map
```

以上で、好きなところに printf 関数を記入することができます。sample8p ディレクトリは上記全て実行済ですので、従来通り、sample8p_3048 または sample8p_3052+リターンでコンパイルできます。

## 【　プログラム内容　】

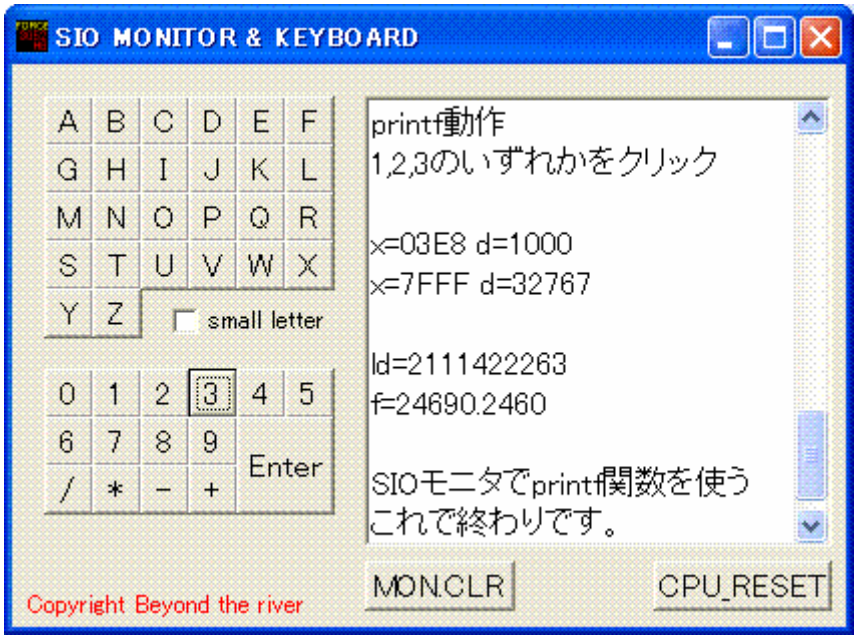
①各データサイズの確認
②d（10 進　16 ﾋﾞｯﾄ）,x(16 進　16 ﾋﾞｯﾄ）,ld（10 進　32 ビット）,f（10 進　浮動小数点）各表示方法、違い
③SIO モニタ表示は全角、漢字表示も対応できます。（全ての漢字が表示できるわけではありません）

## 【　プログラム　】

```
#include "h8_300h.h"
①#include "printf_H8.h"
#include "sio.h"

②#define printf print_sio

int main(void)
{
char key_data;
unsigned short sdata;
int idata;
long ldata;
float fdata;

        //SIO initial

        sio_init();
③       printf("printf 動作\n1,2,3 のいずれかをクリック");

        while(1)
        {
                key_data = char_in1();              //キーデータ読み込み

                switch(key_data)
                {
④                       case '1'://short と int データ

                                //数値セット
                                sdata = 1000;               //x  表示最大 0xFFFF
                                idata = 0x7fff;             //d  表示最大±32767

                                //SIO モニタ
                                printf("\nx=%x d=%d",sdata,sdata);
                                //sdata  hex 表示  dec 表示
                                printf("\nx=%x d=%d",idata,idata);
```

```
                    //idata   hex 表示   dec 表示
                    break;

⑤                   case '2'://long と float データ

                    //数値セット
                    ldata = 1000200030+1111222233;
                    //l 最大±2147483647
                    fdata = 12345.1234+12345.1234;
                    //f 最大±32766.9999 あたりまで扱える

                    //SIO モニタ
                    printf("\nld=%ld",ldata);  //ldata   dec 表示
                    printf("\nf=%f",fdata);     //fdata   hex 表示 dec 表示
                    break;

⑥                   case '3'://char データ

                    //SIO モニタ
                    printf("\nSIO モニタで printf 関数を使う");
                    printf("\n これで終わりです。");       //char 表示
                    break;
            }
        }

}
```

【　解説　】




①#include "printf_H8.h"

printf_h8.h が関数のヘッダファイル、printf_h8.o がオブジェクトファイルです。必ずインクルードしてください。

②#define printf print_sio

printf_h8.h 内で定義されている関数名は「print_sio」なので、printf として使用する場合、このデファイン文で定義してください。もちろん、print_sio のままでも使えます。

③　　printf("printf 動作\n1,2,3 のいずれかをクリック");

「ＫＥＹＢＯＡＲＤ」　１，２，３いずれかのクリックでそれぞれ種類の違う表示をします。

④「１」のクリックで short データと int データ（このコンパイラでは同じ１６ビット）のデータを１６進数（％ｘ）、１０進数（％ｄ）で表示します。unsigned データも表示できますが、その場合、％ｘ表示（０～ＦＦＦＦ）を使用してください。％ｄですと０ｘ７ＦＦＦ以上はマイナス表示になってしまいます。

```
case '1'://short と int データ

    //数値セット
    sdata = 1000;           //x  表示最大 0xFFFF
    idata = 0x7fff;         //d  表示最大±32767

    //SIO モニタ
    printf("\nx=%x d=%d",sdata,sdata);
    //sdata  hex 表示  dec 表示
```

```
                    printf("\nx=%x d=%d",idata,idata);
                    //idata   hex 表示   dec 表示
                    break;
```

⑤「２」のクリックで long データと float データ（どちらも３２ビット）を表示します。

```
            case '2'://long と float データ

                    //数値セット
                    ldata = 1000200030+1111222233;
                    //l 最大±2147483647
                    fdata = 12345.1234+12345.1234;
                    //f 最大±32766.9999 あたりまで扱える

                    //SIO モニタ
                    printf("\nld=%ld",ldata);  //ldata   dec 表示
                    printf("\nf=%f",fdata);     //fdata   hex 表示 dec 表示
                    break;
```

SIO モニタ画面で表示されているように long の整数　ldata＝10002000030+1111222233＝2111422263 と誤差はありません。浮動小数点の fdata＝12345.1234+12345.1234＝24690.2468 となってほしいところですが、24690.2460 と表示されています。小数点以下４桁目の数値が違います。これが先に述べた誤差です。コンパイラによらず浮動小数点演算に付き物の誤差です。全体からみれば 0.00000、、、%の誤差です。用途によってはどの程度の誤差が生じているのか、この printf 関数を用いて確認するといいでしょう。

⑥文字列の表示です。漢字も表示できます。（全てではありません。文字化けする文字もあります）

```
            case '3'://char データ

                    //SIO モニタ
                    printf("\nSIO モニタで printf 関数を使う");
                    printf("\n これで終わりです。");      //char 表示
                    break;
```

## 【　printf 関数使用上の注意点　】

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。

NG な例１

```
ｗｈｉｌｅ（1）
 {
        printf("time  %d",time);     //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「SIO モニタ」は消えます。理由は受信バッファオーバーフローです。プログラム中の問題のコマンドを削除するか、以下の対策を行いコンパイルします。再び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。

対策例１

表示後、「KEYBORAD」からの 1 文字入力を待つ

```
ｗｈｉｌｅ（1）
 {
        printf("time  %d",time);     //ｔｉｍｅが連続してパソコン側に出力される
        cf=char_in1();
 }
```

対策例 2

表示後、適当なウエイトを置く

```
ｗｈｉｌｅ（1）
 {
        printf("time  %d",time);     //ｔｉｍｅが連続してパソコン側に出力される
        wait(1000);
 }
```

対策例 3

表示後、無限ループにする。確認後、コメントとする。

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

## 【　お問い合わせ　】

〒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