

## サンプルプログラムの使い方 Ver 1.00 2006.01.21

対象CPUボード 「BCH83052」

サンプルプログラム内容

以下に示す4つのサンプルプログラムが1つになったソースファイル「test52s.c」、実行ファイル「test52s.mot」がダウンロードできます。フォースライタで「BCH83052」に書き込み、フォースビューで動作確認することができます。

```
test_da()      //D/A出力テスト
test_ad()      //A/D入力テスト
test_io()      //I/O入出力テスト
test_sio()     //SIO入出力テスト
```

開発環境

株式会社ルネサステクノロジ社 HEW4のプロジェクト形式で構成されています。ですが、ソースファイルは(株)秋月電子通商のCbar(GCC on Windows)、有限会社イエローソフト社のYCH8、YLINKでもコンパイル、実行ファイル作成できます。それぞれの環境でファイルに変更等を加える必要はありません。\*1

この3つの環境で作成される実行ファイルはすべて「フォースライタ」で「BCH83052」に書き込みます。

\* 1 (株)秋月電子通商のCbarでは警告が出ますが、動作は問題ありません。気になる場合、void main(void)をmain()に書き換えると消えます。

### 【使い方概要】

1. 初めにダウンロードしてください。「フォースライタ」のダウンロードでH83052ディレクトリの中に添付されてきます。



ダウンロードの仕方は別途ファイル「フォースライタダウンロード方法」をご参照ください。

2. H8/3052用のフォースライタで「test52s.mot」を「BCH83052」書き込みます。

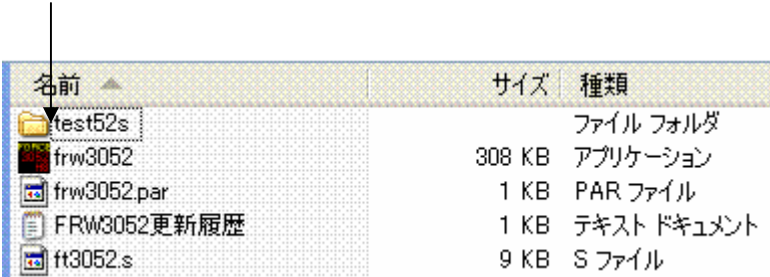
3. フォースビューで0, 1, 2, 3キーをクリックするとそれぞれ

```
0 = test_da() // D / A出力テスト
1 = test_ad() // A / D入力テスト
2 = test_io() // I / O入出力テスト
3 = test_sio() // S I O入出力テスト
```

の動作が確認できます。リセットは「CPU RESET」をクリックしてください。また次の動作を受け付ける待ちに入ります。

### 【使い方詳細】

ダウンロードするとH83052の中にtest52sディレクトリがあります。



名前	サイズ	種類
test52s		ファイル フォルダ
frw3052	308 KB	アプリケーション
frw3052.par	1 KB	PAR ファイル
FRW3052更新履歴	1 KB	テキストドキュメント
ft3052.s	9 KB	S ファイル

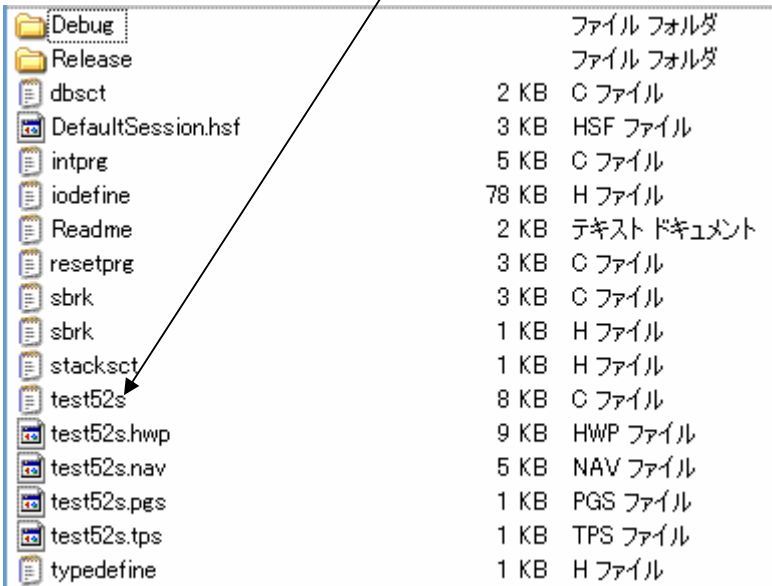
さらに中を見ると



名前	サイズ	種類
test52s		ファイル フォルダ
test52s	1 KB	HEW Workspace File
test52s.tws	1 KB	TWS ファイル

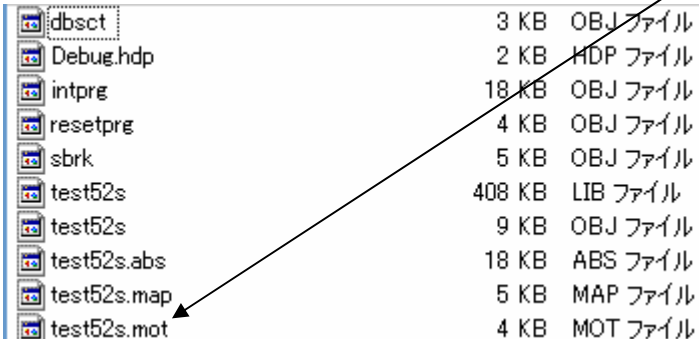
HEW関連のファイルがあります。test52sの中に進みます。

さらに中を見るとCのソースファイルがあります。(株)秋月電子通商のCbar (GCC on Windows) 有限会社イエローソフト社のYCH8、YLINKでコンパイルする場合はこのソースファイルをコピーして使用してください。



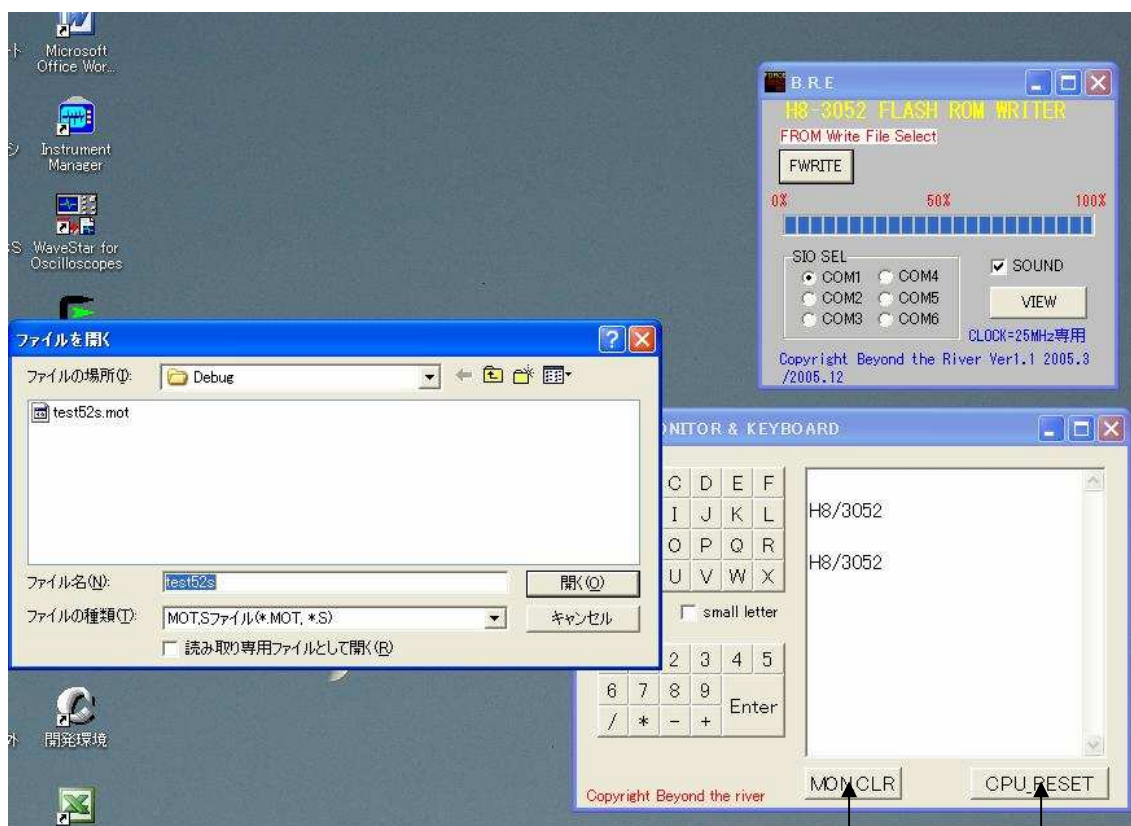
Debug	ファイル フォルダ
Release	ファイル フォルダ
dbstc	2 KB C ファイル
DefaultSession.hsf	3 KB HSF ファイル
intprg	5 KB C ファイル
iodefne	78 KB H ファイル
Readme	2 KB テキスト ドキュメント
resetprg	3 KB C ファイル
sbrk	3 KB C ファイル
sbrk	1 KB H ファイル
stacksct	1 KB H ファイル
test52s	8 KB C ファイル
test52s.hwp	9 KB HWP ファイル
test52s.nav	5 KB NAV ファイル
test52s.pgs	1 KB PGS ファイル
test52s.tps	1 KB TPS ファイル
typedefine	1 KB H ファイル

さらにデバックの中を見ます。このtest52s.motファイルをフォースライタで「BCH83052」に書き込み、動作させます。



dbstc	3 KB OBJ ファイル
Debug.hdp	2 KB HDP ファイル
intprg	18 KB OBJ ファイル
resetprg	4 KB OBJ ファイル
sbrk	5 KB OBJ ファイル
test52s	408 KB LIB ファイル
test52s	9 KB OBJ ファイル
test52s.abs	18 KB ABS ファイル
test52s.map	5 KB MAP ファイル
test52s.mot	4 KB MOT ファイル

## 【サンプルプログラムを動作させてみる】



フォースビューモニタ画面のクリア CPUリセット

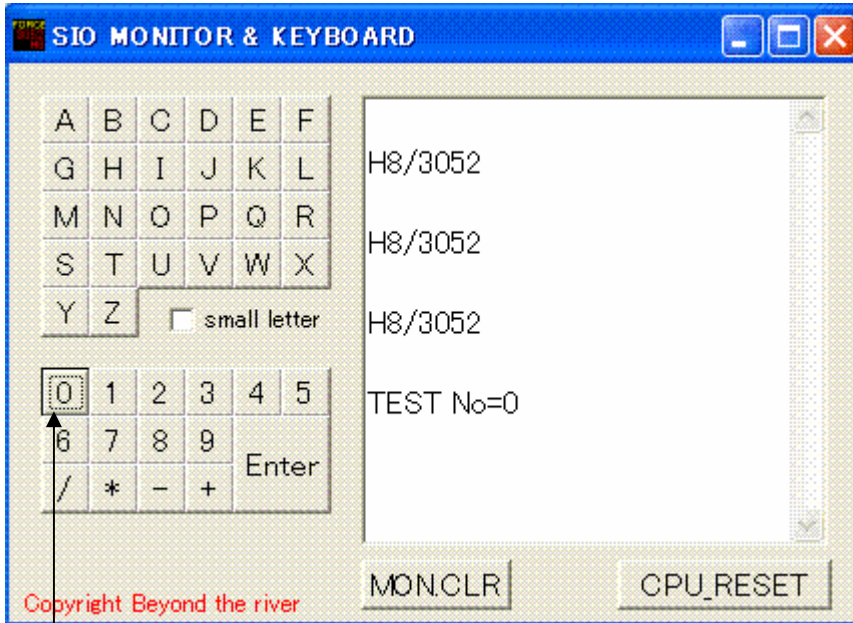
フォースライタのVIEWをクリックして、フォースビューを表示しておきます。test52s.motファイルを書き込みます。書き込み終了とともにフォースビューの画面にはH8/3052と表示されます。

プログラムは入力待ちです。フォースビューで0, 1, 2, 3キーをクリックするとそれぞれの動作が選択されます。

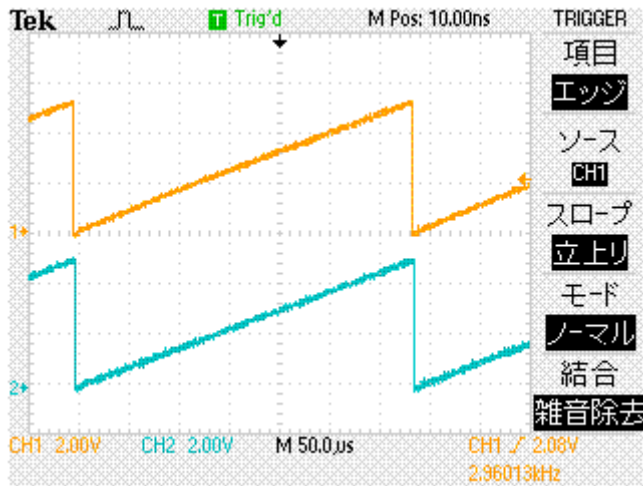
- 0 = test\_da() // D / A 出力テスト
- 1 = test\_ad() // A / D 入力テスト
- 2 = test\_io() // I / O 入出力テスト
- 3 = test\_sio() // S I O 入出力テスト

### D/Aテスト

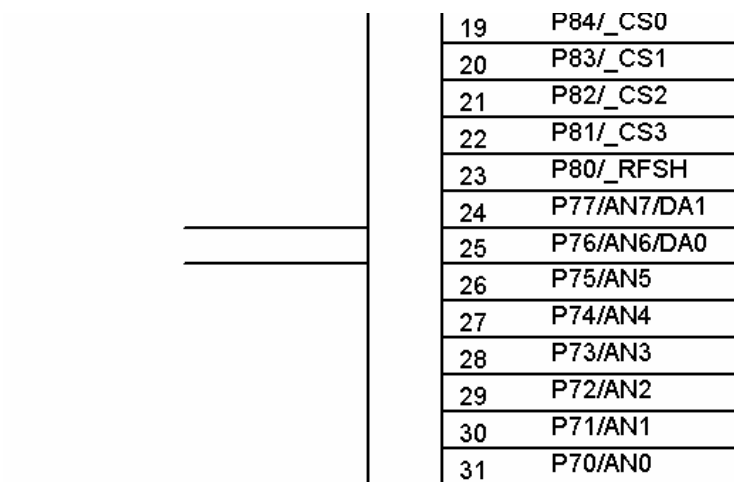
0をクリックし、D/Aコンバータのテストを行います。



ここをマウスでクリックする。TEST No = 0と表示されます。  
D/Aの波形を観測すると以下のように三角形の波形を繰り返します。



BCH83052のCN4の24、25ピンを観測します。



プログラムは以下の通りです。

```
void test_da(void)
{
//DA Test
unsigned char dabuff;

    dabuff = 0;
    DACR = 0xff;    //DA0,1 出力

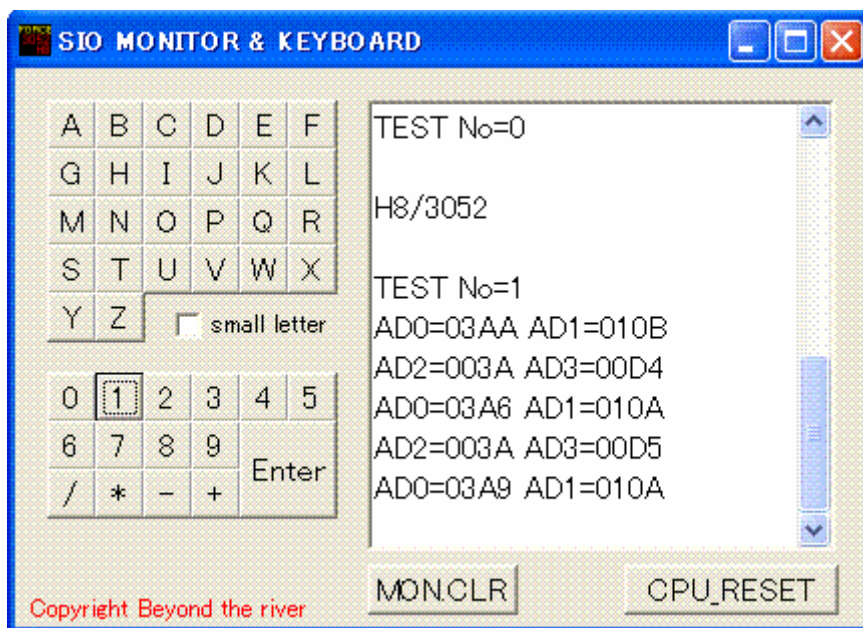
    while(1)
    {
        DADRO = dabuff++;
        DADR1 = dabuff++;
    }
}
```

D/A0,1とも選択し、2つあるD/Aにそれぞれ0~255(0x00~0xFF)までの数をセットして出力電圧を変化させています。dabuffは255までは1ずつ加算され255の次は0になりますので、三角形の波形になります。

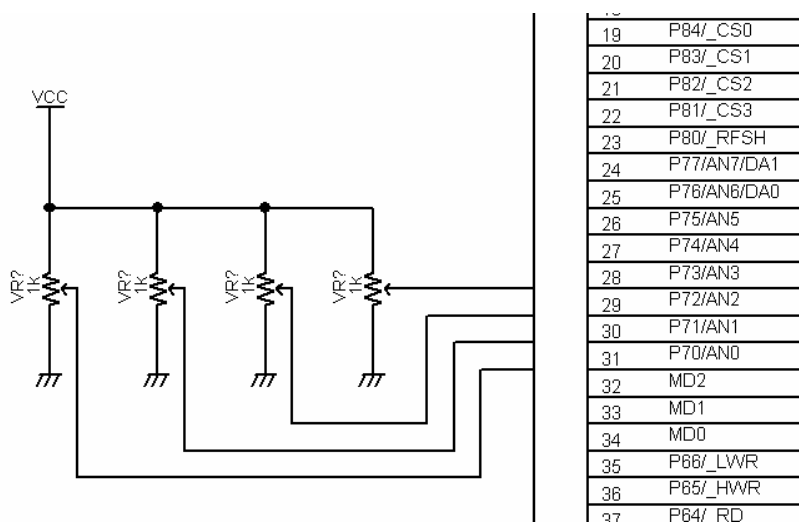
A/Dテスト

CPU\_RESETをクリックし、CPUをリセットさせます。1をクリックし、A/D

テストを選択します。



A / D 0 ~ 3 の 4 つの値が表示されます。それぞれの A / D 端子 ( A N 0 ~ A N 3 ) にポリウム ( 10 K 以下) などを接続し、電圧を可変していくと数値が変わるのが確認できます。0 ~ 5 V で 0 から 0 x 3 F F ( 0 ~ 1 0 2 3 ) まで変わります。



## I Oテスト

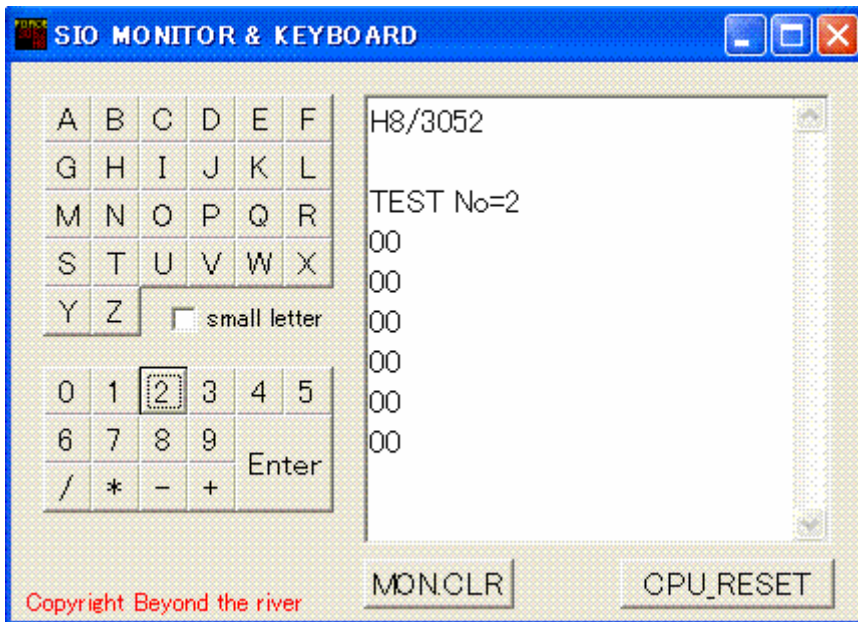
```
void test_io(void)
{
    while(1)
```

```

{
    ascout1(~P4);
    PB = 0xff;
    wait(10000);
    PB = 0x00;
    wait(10000);
}
}

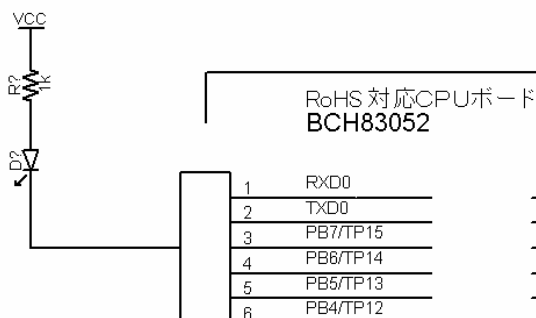
```

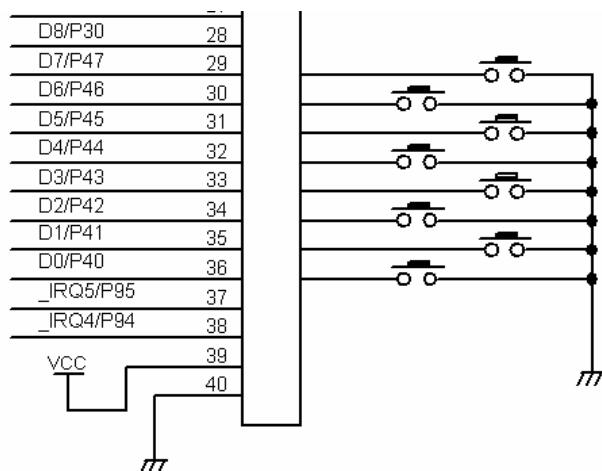
イニシャルでP4は入力、PBは出力に設定しています。関数 test\_io の中ではP4の値の反転がフォースビュー画面に表示され、PBが交互にオン、オフします。



たとえばP4にはスイッチを付けると押されたスイッチのビットに1が立つのが確認できます。PBにはLEDを付ければLEDが点滅するのが確認できます。

たとえばPB7にLEDを付けると点滅が人の目で確認できる。

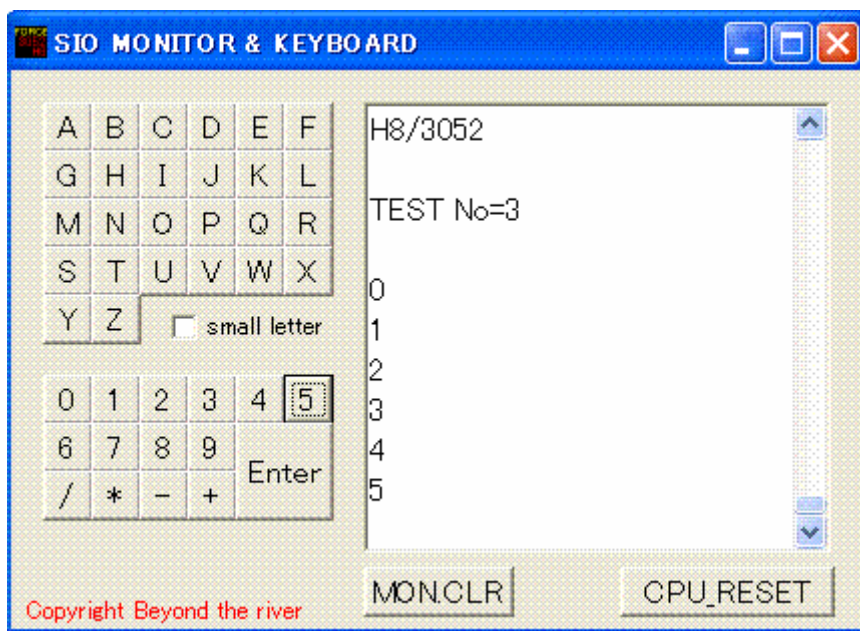




P 4 は内部プルアップ設定されています。

### S I Oテスト

C P U \_ R E S E T をかけ 3 番をクリックします。以降、押したキーボードの値が表示されれば O K です。



フォースビューのキーボードはクリックされた値をパソコンからマイコン側にアスキーコードで出力します。

```
void test_sio(void)
{
    while(1)
    {
```

```
        char_out1(char_in1());  
    }  
}
```

プログラムでは受信したデータをそのままパソコンに返しています。返ってきた値をモニタに表示しています。

#### 余談

今回、同じプログラムを3つの開発環境で評価してみました。波形を観測しながらプログラムを開発していたので、たとえば前述のD/Aテストプログラムで三角形の時間幅がコンパイラにより異なることが目立ちました。ためしに時間を計測してみると遅いものと速いもので三角形の時間幅が3倍も違うのです。残りのものも最速のものに比べちょうど2倍の時間が必要なようです。もちろん、同じハード、同じソースです。

つまり、同じ仕事を行うのに最速なものを1とすると次のものは倍の時間がかかる、最後のものは3倍の時間がかかる、ということになります。各コンパイラオプションはデフォルトです。

もちろん、コンパイラによりデフォルトの考え方、得意、不得意な部分、最適化オプションの設定等々により実行時間は変化してくるでしょうから、このことですぐに良否を決定することはできません。

しかし、単純な代入分と条件分岐、無限ループ、加算等のプログラムで3倍もの時間差が出るとは考えていなかったのが驚きました。CPUのクロックを倍に上げることは容易ではありませんが、コンパイラを変えると3倍の速度で動くこともある、ということは頭に入れておこうと思いました。

#### 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。

CbarはKENCH氏によるフリーソフトです。

GCC(GNU Compiler Collection)CはGNUプロジェクトによるフリーCコンパイラです。

YellowIDE、イエロースコープは有限会社イエローソフト社の登録商標です。

「フォース」機能、「フォースライタ」は弊社の発明です。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](mailto:support@beriver.co.jp)

有限会社ビーリバーエレクトロニクス