

## vlbcd\_lib : BCD 演算ライブラリ(10 万桁関数電卓付き)説明書

### (1) 概要

- ・ 32bit マイコン、C 言語用の BCD 演算ライブラリ(C ソース)です。
- ・ 理論上の限界は 7 億桁、実用的には 10 万桁が限度と思います。
- ・ 主な演算機能：
  - 加減乗除、剰余、平方根、立方根、文字列から変換、文字列へ変換、
  - 三角関数(sin,cos,tan およびその逆関数)、
  - 指数・対数関数(exp, loge, exp10, log10, x^y)、
  - 双曲線関数(sinh,cosh,tanh およびその逆関数)
- ・ 応用例として 10 万桁関数電卓(Windows 版)を添付しています。

### (2) インストール

ソース・ライブラリですので適当な場所で解凍してください。

test\_sample のプロジェクト・ファイルは Visual studio Express 2017 用です。

### (3) フォルダ／ファイル構成

vlbcd_lib	
└─ doc	
└─ vlbcd_lib.pdf	本説明書
└─ include	
└─ vlbcd_api.h	アプリケーション側で組み込むヘッダ
└─ lib	
└─ vlbcd_lib.c /.h	ライブラリのソースコードおよびヘッダ
└─ vlbcd_lib_xx.h	関連する数値を格納したヘッダ
└─ test_sample	
└─ TSmain.c	関数電卓メインプログラム
└─ longCALC.vcxproj	プロジェクトファイル
└─ longCALC.exe	ビルド済み実行形式ファイル(1920x1080 モニタ必要)
└─ win32_screen	Windows 用画面インタフェース

#### (4) ライブラリの使用方法

##### ①適用コンパイラの確認

int 型が 32bit のコンパイラに対応しています。

##### ②vlbcd\_api.h のインクルード

BCD 演算関数を呼び出したいアプリケーション・プログラム側でインクルードします。  
以下のオプションがあるので、コンパイル・オプションで指定するか、ソースを書き換えます。

- nBCD\_NUM\_SIGNIF

有効桁数を指定します。12 桁以上 7 億桁以下(実メモリ量により制限)。

速度との兼ね合いで桁数を動的に変更する場合は最大値を指定します。

- BCD\_NUM\_FIXED

有効桁数が固定で良い場合は定義します。オブジェクトが多少小さくなります。

- nBCD\_EXPLIM

数値の指数( $\times 10^n$ )の上限(絶対値)を指定します。デフォルトは 999,999,999 です。

この範囲(+側)を超えると OVERFLOW となって演算が行われなくなります。

-側を下回るとゼロに変換します。

- BCD\_NON\_SECURITY

コンパイラが strcpy\_s0等の安全な関数を提供していない場合は指定します。

- BCD\_COMPACT\_MEM

メモリ(ワーク変数)を節約したい場合は指定します。指定すると乗除算の演算時間が遅くなります。

- BCD\_BASIC\_FUNC

三角関数等を使用せず四則演算等のみに制限する場合に指定します。オブジェクトが小さくでき、ワーク変数も節約できます。

- BCD\_NO\_EVALUATION

評価用機能を使わない場合に指定します。オブジェクトが多少小さくなります。

##### ③BCD 変数の定義と関数の使い方

アプリケーション・プログラム側で、数値(BCD\_REG 型)を格納する変数を定義します。

例 : BCD\_REG x, y, z;

この変数を使って、例えば加算の場合、bcd\_add(z, x, y)とすると、 $z = x + y$  を計算します。結果を格納する変数は右辺の変数と同じでも良いので、bcd\_add(z, z, y)あるいは bcd\_add(z, z, z)と使うことも可能です。

BCD 変数に数値を設定する方法としては、文字列から変換する方法と、二つの int 型を使って  $\pm 2,147,483,647 \times 10^{\pm 999,999,999}$  の範囲の数値を設定する方法があります。また演算結果を文字列に変換する関数も用意してあります。

文字列からの変換例 : `bcd_strto(z, "1.23 e-6");` ;あるいは

```
char m[ ]=" 1.23 e-6";
```

```
bcd_strto(z, m);
```

二つの整数からの設定 : `bcd_ito(z, -5, -1);` // -0.5 が設定される

文字列への変換 : `char m[100];`

```
bcd_tostr(z, m);
```

#### ④ライブラリ・ソースの組み込み

少なくとも、`vlbcd_lib.c`, `vlbcd_lib.h`, `vlbcd_lib_tbl.h` はビルド対象とします。 .

三角関数等を使用する場合は、`vlbcd_lib_pi.h`, `vlbcd_nap_tbl.h`, `vlbcd_ln10_tbl.h` もビルド対象とします。

### (5) 演算速度例

高速乗除算(`BCD_COMPACT_MEM` の指定なし)時の速度例を以下に示します。

#### ①32bit マイコン(ルネサス RH850/F1KM, 120MHz)での速度例

種別	演算内容	桁数	時間
乗算	$9.999 \cdot \cdot 999 \times 9.999 \cdot \cdot 999$	1,000 桁	0.026 秒
除算	$12345678999 \cdot \cdot 99 / 98765432199 \cdot \cdot 99$	1,000 桁	0.025 秒
平方根	$\sqrt{5}$	1,000 桁	0.12 秒
立方根	$\sqrt[3]{5}$	1,000 桁	0.5 秒
sin, cos	$\sin(44.99 \cdot \cdot 99^\circ)$ , $\cos(44.99 \cdot \cdot 99^\circ)$	200 桁	0.13 秒
		1,000 桁	6 秒
exp	$\exp(0.5)$	200 桁	0.1 秒
		1,000 桁	1.8 秒
	$\exp(0.4999 \cdot \cdot 999)$	200 桁	0.24 秒
		1,000 桁	12 秒
loge	$\log_e(e)$	200 桁	0.21 秒
		1,000 桁	12 秒
$\sin^{-1}$	$\sin^{-1}(0.5)$ 注: $30^\circ$ を強制的に出力する機能をオフして計測	200 桁	0.4 秒
		1,000 桁	9 秒
	$\sin^{-1}(0.4999 \cdot \cdot 999)$	200 桁	0.8 秒
		1,000 桁	50 秒

②手持ちのパソコン(Corei7-12700)での速度例

種別	演算内容	桁数	時間
乗算	9.999・・・999 x 9.999・・・999	1 万桁	0.016 秒
		10 万桁	1.6 秒
除算	12345678999・・・99 / 98765432199・・・99	1 万桁	0.015 秒
		10 万桁	1.4 秒
平方根	$\sqrt{5}$	1 万桁	0.073 秒
		10 万桁	7.3 秒
立方根	$\sqrt[3]{5}$	1 万桁	0.34 秒
		10 万桁	34 秒
sin, cos	sin(44.99・・・99°), cos(44.99・・・99°)	1 万桁	23 秒
		10 万桁	4.9 時間
exp	exp(0.5)	1 万桁	0.93 秒
		10 万桁	76 秒
	exp(0.4999・・・999)	1 万桁	49 秒
		7 万桁	3.7 時間
loge	loge(e)	1 万桁	61 秒
		10 万桁	16.4 時間
sin <sup>-1</sup>	sin <sup>-1</sup> (0.5) 注: 30° を強制的に出力する機能をオフして計測	1 万桁	6.5 秒
		5 万桁	4.3 分
	sin <sup>-1</sup> (0.4999・・・999)	1 万桁	4.6 分
		5 万桁	9.3 時間

(6) オブジェクト・サイズ例(ルネサス RH850 の場合)

ROM サイズ(コード+データ)を以下に示します。ライブラリだけでなくテスト用の関数呼び出しの機能(TSmain.c の一部機能)も若干含まれています。

単位 : KB

項目	評価機能あり	評価機能なし
桁数動的変更、全機能、	38.2	36.0
桁数固定(BCD_NUM_FIXED)、全機能	34.6	32.4
桁数固定、四則演算(BCD_BASIC_FUNC)	16.2	14.1
桁数固定、四則演算、低速(BCD_COMPACT_MEM)	(無効)16.2	13.2

評価機能なし : BCD\_NO\_EVALUATION 指定

## (7) サンプル・プロジェクト(Windows 用)

Visual studio Express 2017 で longCALC.vcxproj を開けばビルドできるはずです。

TSmain.c・・・計算する数値をパラメータとして受け取って、BCD 演算関数を呼び出し、結果をテキストで返しています。

画面インタフェースは、win32\_screen フォルダの中のプログラムで行っています。

all_define.h	受け渡しパラメータなど、全般的な定義
all_define.c	ボタンや編集領域などの配置と属性を記述
base_main.c	WinMain などの基本的なプログラム
initialize.c	起動時に.ini ファイルを読み込むプログラム
screen_comd.c	押されたボタンに応じた処理を行うプログラム

## (8) longCALC (Windows 用) の使い方

ビルド済みの longCALC.exe をクリックすれば起動します。

なお、画面の大きさは 1920x1080 ドットのモニタを前提に設計しています。

同じフォルダ内に設定やパラメータを保存する.ini ファイルを作ります。

### ①基本的演算

加減乗除など 2 つの数値の演算の場合は、R1 と R0 に数値を入れて、演算ボタンを押します。例えば減算であれば、R1-R0 を計算して R0 に結果を表示します。元の R1,R0 の内容は 1 つ上に移動します。

平方根や sin など 1 つの数値の演算の場合は、R0 に数値を入れて、演算ボタンを押します。

時間のかかる演算を途中でやめたい場合は、右下にある「中止」ボタンを押します。

### ②有効桁数の設定

左上に設定欄があるので入力します。指定した桁数に対して、内部的には演算誤差を補うための精度拡張などを行います。最終的な桁数は、演算ボタンが押された後に「内部桁数」欄に表示します。

左下の「☐内部桁表示」にチェックすると内部で格納されている数値をすべて表示します。チェックしない場合は有効桁数+1 桁目を四捨五入して有効桁数のみ表示します。

### ③BCD\_COMPACT\_MEM オプションの評価

左下の「☐省メモリ」にチェックすると、BCD\_COMPACT\_MEM オプションを指定した状態となります。

#### ④演算実行時間の測定

「下記を実行」の下に次のコマンドを記述することにより演算実行時間を測定できます。

`loop n c`            `n`:演算繰り返し回数、`c`:演算種別

演算種別は 1 文字で、次の通りです。

<code>+</code> , <code>-</code> , <code>*</code> , <code>/</code> , <code>%</code>	加減乗除と剰余
<code>1</code> , <code>8</code> , <code>9</code>	$x^n$ , 平方根, 立方根
<code>a</code> , <code>b</code> , <code>c</code>	<code>sin</code> , <code>cos</code> , <code>tan</code>
<code>f</code> , <code>g</code> , <code>h</code>	<code>asin</code> , <code>acos</code> , <code>atan</code>
<code>d</code> , <code>e</code> , <code>i</code> , <code>j</code>	<code>exp</code> , <code>exp10</code> , <code>loge</code> , <code>log10</code>

あらかじめ `R1` あるいは `R0` に演算対象となる数値を入力し、「下記を実行」ボタンを押せば実行します。結果は「中止」ボタンの右側に表示します。

#### ⑤三角関数等の精度と演算実行時間の測定

「下記を実行」の下に次のコマンドを記述することにより精度と時間を測定できます。

`prec n c`            `n`:級数展開のループ数、`c`:演算種別

ここで、`n=0` なら `vlbcd_lib_tbl.h` に記載のデフォルトループ数で実行します。

1 以上の数値を入れるとそのループ数で実行します。

あらかじめ、`R6` に演算対象の数値、`R5` に期待値を入力し、「下記を実行」ボタンを押すと、`R1` に演算結果、`R0` に誤差(ppm)を出力します。

#### (9) ライブラリやテストサンプルの利用条件

- ・収益を伴わない趣味や研究であれば利用可能です。
- ・収益を伴う場合は、事前に当方に相談してください。
- ・動作保証や精度の保証は致しかねます。

#### (10) サポート

問い合わせ先: `L00-micon` あとまぐ `memoad.jp` (注: あとまぐを@に変えて下さい)

できるだけタイトル先頭に【サポート依頼】を付けて下さい。

72 時間経過しても何の返事もない場合は、再メールをお願いします。

追加情報がある場合は、[マイコン技研](#) の「技術資料」ページに掲載します。

2025 年 12 月 15 日    マイコン技研    澤田 明