

SHC/C++ Ver5.0Release2x (PC/Windows) 添付資料  
 SHC/C++ Ver5.0Release2x (SGI/Indy) 添付資料

(株)日立製作所 半導体事業部  
 マイコンシステム設計部

1. SHCV4.1(C コンパイラマニュアル)からの組み込み関数追加一覧を表 1 に示します。

表 1 組み込み関数追加一覧

No.	項目	機能	仕様	説明
1	浮動小数点ユニットシステム/コントロールレジスタ (FPSCR)	浮動小数点ユニットシステム/コントロールレジスタの設定	void set_fpscr( int cr )	浮動小数点ユニットシステム/コントロールレジスタに cr (32 ビット) を設定する。
2		浮動小数点ユニットシステム/コントロールレジスタの参照	int get_fpscr( )	浮動小数点ユニットシステム/コントロールレジスタを参照する。
3	単精度浮動小数点ベクタ演算	FIPR 命令	float fipr( float vec1[4], float vec2[4] )	2 つのベクタの内積を求める。 例 extern float data1[4], data2[4]; この時 fipr(data1, data2) は、ベクタ data1 と data2 の内積を求める。
4		FTRV 命令	void ftrv( float vec1[4], float vec2[4] )	data1 (ベクタ) を tbl (4×4 行列) で変換した結果を data2 (ベクタ) に格納する。ただし、tbl は組み込み関数 ld_ext (No.12) でロードする。 data2 = data1 × tbl 例 extern float tbl[4][4]; extern float data1[4], data2[4]; ld_ext(tbl); この時 ftrv(data1, data2) は、data1 を 4×4 行列 tbl で変換した結果を data2 に格納する。

5	単精度浮動小数点ベクタ演算	4次元ベクタの4×4行列による変換と4次元ベクタとの和	<pre>void ftrvadd(     float vec1[4],     float vec2[4],     float vec3[4] )</pre>	<p>data1(ベクタ)をtbl(4×4行列)で変換した結果とdata2(ベクタ)の和をdata3(ベクタ)に格納する。ただし、tblは組み込み関数ld_ext(No.12)でロードする。</p> <p>data3 = data1 × tbl + data2</p> <p>例</p> <pre>extern float tbl[4][4]; extern float data1[4]; extern float data2[4]; extern float data3[4]; ld_ext(tbl);</pre> <p>この時</p> <p>ftrvadd(data1,data2,data3)は、data1を4×4行列tblで変換した結果とdata2の和をdata3に格納する。</p>
6	単精度浮動小数点ベクタ演算	4次元ベクタの4×4行列による変換と4次元ベクタとの差	<pre>void ftrvsub(     float vec1[4],     float vec2[4],     float vec3[4] )</pre>	<p>data1(ベクタ)をtbl(4×4行列)で変換した結果とdata2(ベクタ)の差をdata3(ベクタ)に格納する。ただし、tblは組み込み関数ld_ext(No.12)でロードする。</p> <p>data3 = data1 × tbl - data2</p> <p>例</p> <pre>extern float tbl[4][4]; extern float data1[4]; extern float data2[4]; extern float data3[4]; ld_ext(tbl);</pre> <p>この時</p> <p>ftrvadd(data1,data2,data3)は、data1を4×4行列tblで変換した結果とdata2の差をdata3に格納する。</p>
7		4次元ベクタの和	<pre>void add4(     float vec1[4],     float vec2[4],     float vec3[4] )</pre>	<p>data1(ベクタ)とdata2(ベクタ)の和をdata3(ベクタ)に格納する。</p> <p>data3 = data1 + data2</p> <p>例</p> <pre>extern float data1[4]; extern float data2[4]; extern float data3[4];</pre> <p>この時</p> <p>add4(data1,data2,data3)はdata1とdata2の和をdata3に格納する。</p>

8	単精度浮動小数点ベクタ演算	4次元ベクタの差	<pre>void sub4(     float vec1[4],     float vec2[4],     float vec3[4] )</pre>	<p>data1(ベクタ)と data2(ベクタ)の差を data3(ベクタ)に格納する。  data3=data1-data2  例  extern float data1[4];  extern float data2[4];  extern float data3[4];  この時  sub4(data1,data2,data3)は、  data1 と data2 の差を data3 に格納する。</p>
9	単精度浮動小数点 4×4 行列の演算	4×4 行列の乗算	<pre>void mtrx4mul(     float mat1[4][4],     float mat2[4][4] )</pre>	<p>tbl1(4×4 行列)を tbl(4×4 行列)で変換した結果 tbl2 に格納する。ただし、tbl は組み込み関数 ld_ext(No.12)でロードする。  tbl2 = tbl1×tbl  例  extern float tbl[4][4];  extern float tbl1[4][4];  extern float tbl2[4][4];  ld_ext(tbl);  この時  mtrx4mul(tbl1,tbl2)は、tbl1 と 4×4 行列 tbl で乗算した結果を tbl2 に格納する。</p>
10		4×4 行列の乗算と和	<pre>void mtrx4muladd(     float mat1[4][4],     float mat2[4][4],     float mat3[4][4] )</pre>	<p>tbl1(4×4 行列)を tbl(4×4 行列)で変換した結果と tbl2(4×4 行列)の和を tbl3(4×4 行列)に格納する。ただし、tbl は組み込み関数 ld_ext(No.12)でロードする。  tbl3=tbl1×tbl+tbl2  例  extern float tbl[4][4];  extern float tbl1[4][4];  extern float tbl2[4][4];  extern float tbl3[4][4];  ld_ext(tbl);  この時  mtrx4muladd(tbl1,tbl2,tbl3)は、tbl1 と 4×4 行列 tbl で乗算した結果と tbl2 との和を tbl3 に格納する。</p>

11	単精度浮動小数点 4×4 行列の演算	4×4 行列の乗算と差	<pre>void mtrx4sub(     float mat1[4][4],     float mat2[4][4],     float mat3[4][4] )</pre>	<p>tbl1(4×4 行列)を tbl(4×4 行列)で変換した結果と tbl2(4×4 行列)との差を tbl3 に格納する。ただし、tbl は組み込み関数 ld_ext(No.12)でロードする。</p> <p>tbl3=tbl1×tbl - tbl2</p> <p>例</p> <pre>extern float tbl[4][4]; extern float tbl1[4][4]; extern float tbl2[4][4]; extern float tbl3[4][4]; ld_ext(tbl);</pre> <p>この時</p> <p>mtrx4mulsub(tbl1,tbl2,tbl3) は、tbl1 と 4×4 行列 tbl で乗算した結果と tbl2 との差を tbl3 に格納する。</p>
12	裏レジスタへのアクセス	裏レジスタへのロード	<pre>void ld_ext(     float mat[4][4] )</pre>	<p>tbl(4×4 行列)を裏レジスタにロードする。</p> <p>例</p> <pre>extern float tbl[4][4];</pre> <p>この時</p> <p>ld_ext(tbl)は、tbl の内容を裏レジスタにロードする。</p>
13		裏レジスタからのストア	<pre>void st_ext(     float mat[4][4] )</pre>	<p>裏レジスタ の内容を tbl(4×4 行列)にストアする。</p> <p>例</p> <pre>extern float tbl[4][4];</pre> <p>この時</p> <p>st_ext(tbl)は、裏レジスタの内容を tbl にストアする。</p>

2. プライベート命令組み込み関数一覧を表2に示します。本組み込み関数を使用するときは、private.hをインクルードしてください。

表2 プライベート組み込み関数一覧

No.	項目	機能	仕様	説明
1	sin,cos の近似値計算	FSCA 命令	<pre>void fsca(     long data,     float *sinval,     float *cosval )</pre>	<p>data で指定された角度(固定小数点)の sin,cos の値をそれぞれ svalue,cvalue に格納する。</p> <p>例</p> <pre>#include &lt;private.h&gt; extern long data; extern float svalue; extern float cvalue; この時     fsca(data,&amp;svalue,&amp;cvalue) )は、sin 値、cos 値をそれぞれ、svalue,cvalue に格納する。</pre>
2	平方根の逆数の計算	FSRRA 命令	<pre>float fssca(     float val )</pre>	<p>data で指定された値に平方根の逆数を求める。</p> <p>例</p> <pre>#include &lt;private.h&gt; extern float data; この時     fsrra(data)は、data の平方根の逆数を返却する</pre>

3. コンパイルオプションは、cpu=sh4,fpu=single,endian=little を指定してください。コンパイルオプションは、Set4 で SPC=0 となる CPU のバイパス優先判定論理不良を回避するコードを出力するときは、-extra=a=400 を指定してコンパイルしてください。

#### 4. リテラルプールの 32 バイト align

-extra=a=800 指定時に、リテラルプールの開始と終了の割付境界を 32 バイトに調整してオブジェクトを生成します。

#### 5. float 定数除算->float 定数乗算オプション

-aggressive=2(短縮形 -ag=2)指定時、除数データが float 型定数の場合、float 型定数の逆数で乗算する最適化を行います。

#### 6. #pragma aligndata8 (変数名,...)

上記 #pragma で指定した変数を 8 バイト境界に配置します。  
aligndata8 で指定できる変数の型に制限はありません。

注意) この改善により、リンク時に“107 SECTION ATTRIBUTE MISMATCH”のメッセージが出力されますので、この場合、リンク時に -align\_section オプションを指定してください。

7 . SH4 用標準ライブラリは、次の 8 種類があります。cpu オプション、pic オプション、endian オプションおよび fpu オプションの組み合わせにより表に示すライブラリをリンクしてください。

Fpu 指定	Fpu=single			
Endian 指定	Endian = big		Endian = little	
Pic 指定	Pic=0	Pic=1	Pic=0	Pic=1
Cpu=sh4	Sh4nbfzz.lib	Sh4pbfzz.lib	Sh4nlfzz.lib	Sh4plfzz.lib

Fpu 指定	指定なし			
Endian 指定	Endian = big		Endian = little	
Pic 指定	Pic=0	Pic=1	Pic=0	Pic=1
Cpu=sh4	Sh4nbmzz.lib	Sh4pbmzz.lib	Sh4nlmzz.lib	Sh4plmzz.lib

8 . SHC++ Ver5.0R28 の機能は、起動コマンドを除いて SHC Ver.5.0R27 と同様です。  
 起動コマンドを shc (C コンパイラ)から、shcpp(C++コンパイラ)に変更して、  
 C++コンパイラをご使用ください。

以上