ASL/SX ラッパ Ver.1.0
機能仕様書

第 1.2 版

Xevolver project

2014 年 8 月

改版履歴

版数	作成日	内容
1.0	2014/3/25	新規作成
1.1	2014/4/11	MKL ラッパの一部の仕様を変更 6.3 (4), (5)
1.2	2014/4/28	C インタフェース時の単精度、倍精度の扱いについて更新 6.1 (1), (2), (3), (4) 7.1.1 (1)

目次

1. はじめに

本仕様書は、Linux や Windows システム上（以下 x86 システムと呼ぶ）で利用できる高速フーリエ変換（FFT）ライブラリである FFTW や Intel MKL を用いたプログラムを、SX 上で利用するための ASL ラッパ Ver.1.0 の機能について示したものです。

FFTW や Intel MKL は、x86 システム用にチューニングされたライブラリとして提供され、利用されています。一方 SUPER-UX では、SX 向けに高度にチューニングされたライブラリとして、ASL が提供されています。

x86 システム上で動作しているプログラムを、SX 上で動作させる場合、大きな性能向上が期待できます。しかし、これらのプログラムが SX には提供されていない FFTW や MKL のライブラリを用いている場合、そのまま利用できません。

今回開発する ASL ラッパは、このような移植上の問題を解決し、既存のプログラムを SX 上で容易に実行できるようにし、SX 利用促進に寄与することを目的とします。

2. 対象読者

本仕様書では、読者が、SUPER-UX、Linux、FFTW、Intel MKL についての基本的な知識を有するものと仮定しています。

参考資料は以下のものがあります。

- FFTW：FFTW の公式サイト
  http://www.fftw.org/
- Intel MKL：MKL のドキュメント
  http://software.intel.com/sites/products/documentation/hpc/mkl/mklman/index.htm
- ASL/SX、FORTRAN90/SX、C++/SX：SUPER-UX の製品マニュアル
  - 科学技術計算ライブラリ ASL/SX 利用の手引、ASL C 言語インタフェース利用の手引
  - FORTRAN90/SX プログラミングの手引、言語説明書、並列処理機能利用の手引
  - C++/SX プログラミングの手引

  ※製品マニュアルについては、運用管理者に確認して下さい。

3. 制約

FFTW、MKL、および ASL は、それぞれ独自に開発されたため、機能差が存在します。ラッパは、これらの機能差を全て吸収できるものではありませんが、基本的な機能はプログラムの修正なしに動作できることを目指しています。

本仕様書に記載している内容は、将来、機能強化等のため変更になる可能性があります。

4. 開発対象

ASL ラッパ Ver.1.0 の開発では、FFTW/MKL の関数を呼び出しているプログラムから ASL の関数を呼び出すためのラッパの開発を行います。

5. リリース形態

本ラッパは、SUPER-UX と Linux クロス環境向けにリリースします[1]。

提供形態は、tar 形式であり、コンパイル時にインクルードするヘッダファイルと、リンク時に必要なアーカイブファイルからなります。

利用者は、コンパイル時に-I オプションでインクルードファイルのパスを指定し、リンク時には-L オプションでアーカイブのパス、-l でライブラリ名を指定する必要があります。

インストール方法の詳細は、別紙でリリースメモとして提供します。コンパイルおよびリンク方法の詳細は、本書の「7 利用方法」の節を参照してください。

[1] Ver.1.0 の初期リリースで提供するのは、FFTW→ASL のラッパのみです。MKL→ASL のラッパは別途提供します。

6. ラッパ機能概要

今回提供するラッパは、FFTW の Legacy Fortran インタフェース（Fortran95 以前の Fortran）と C インタフェースです。Modern Fortran インタフェースはサポートしていません。

6.1 FFTW C インタフェース

(1) 提供関数

以下の関数を提供します。提供するのは、単精度と倍精度です。

機能概要	関数名
1 次元 FFT の plan 作成	fftw_plan_dft_1d()
2 次元 FFT の plan 作成	fftw_plan_dft_2d()
3 次元 FFT の plan 作成	fftw_plan_dft_3d()
plan の実行	fftw_execute()
plan の解放	fftw_destroy_plan()
plan の消去	fftw_cleanup()
入出力領域のメモリ割り当て	fftw_malloc()
入出力領域の解放	fftw_free()

Ver.1.0 では、上記以外の関数はサポートしません。

これらの関数を利用する際は、fftw3.h のインクルードが必要です（既存の FFTW を用いている場合は、すでにインクルードしているはずです）。

※注意※

- **ラッパ関数の引数と変数の大きさ(バイト)の対応は次のとおりでなければなりません。**

	plan 型ポインタ	整数型引数	複素数型
fftw_*	8 バイト	4 バイト	8 バイト(4 バイト×2) または 16 バイト(8 バイト×2)

- **plan 関数で指定する変換サイズを示す要素数は、2 以上でなければなりません。これは、ASL の仕様に基づくものです。**
- **C 版では、単精度版と倍精度版の呼び出し関数名は同じです。単精度と倍精度の切り分けは、FFTW と同様、リンク時に、単精度用または倍精度用のライブラリを指定することで行います。**

以下に、これらの関数の詳細仕様を示します。

(2) fftw_plan_dft_1d

機能

複素 1 次元離散フーリエ変換のプランを生成します。

使用法

```
fftw_plan  fftw_plan_dft_1d(int n, fftw_complex *in, fftw_complex *out,
                  int sign, unsigned flags );
```

引数

n

int 型でなければいけません。変換のサイズを指定します。
入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

in

fftw_complex 型、または fftwf_complex 型のポインタでなければいけません。
入力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

out

fftw_complex 型、または fftwf_complex 型のポインタで、入力配列と同じ型で
なければいけません。出力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

sign

int 型でなければいけません。
値が-1(FFTW_FORWARD)の場合は正変換、それ以外の値は逆変換が実行されます。
誤った引数を指定した場合、結果は保証されません。

flag

unsigned int 型でなければいけません。
FFTW3 と ASL/SX の機能差のため、ラッパでは値に意味は持ちません。このため、
FFTW ラッパは、本引数の値を無視します。

返却値

fftw_plan

fftw_plan 型を返却します。ASL/SX が複素 1 次元離散フーリエ変換を行うために
必要な情報が設定されます。

(3) fftw_plan_dft_2d()

機能

複素 2 次元離散フーリエ変換のプランを生成します。

使用法

```
fftw_plan  fftw_plan_dft_2d(int n1, int n2, fftw_complex *in,
                 fftw_complex *out, int sign, unsigned flags );
```

引数

n1

int 型で、2 以上の値でなければいけません。変換の 1 次元目のサイズを指定します。
n1 と n2 の積が入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

n2

int 型で、2 以上の値でなければいけません。変換の 2 次元目のサイズを指定します。
n1 と n2 の積が入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

in

fftw_complex 型、または fftwf_complex 型のポインタでなければいけません。
入力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

out

fftw_complex 型、または fftwf_complex 型のポインタで、入力配列と同じ型で
なければいけません。出力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

sign

int 型でなければいけません。
値が-1(FFTW_FORWARD)の場合は正変換、それ以外の値は逆変換が実行されます。
誤った引数を指定した場合、結果は保証されません。

flag

unsigned int 型でなければいけません。
FFTW3 と ASL/SX の機能差のため、ラッパでは値に意味は持ちません。このため、
FFTW ラッパは、本引数の値を無視します。

返却値

fftw_plan

fftw_plan 型を返却します。ASL/SX が複素 1 次元離散フーリエ変換を行うために
必要な情報が設定されます。

(4) fftw_plan_dft_3d()

機能

複素 3 次元離散フーリエ変換のプランを生成します。

使用法

```
fftw_plan  fftw_plan_dft_3d(int n1, int n2, int n3, fftw_complex *in,
                 fftw_complex *out, int sign, unsigned flags );
```

引数

n1

int 型で、2 以上の値でなければいけません。変換の 1 次元目のサイズを指定します。
n1 と n2 と n3 の積が入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

n2

int 型で、2 以上の値でなければいけません。変換の 2 次元目のサイズを指定します。
n1 と n2 と n3 の積が入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

n3

int 型で、2 以上の値でなければいけません。変換の 3 次元目のサイズを指定します。
n1 と n2 と n3 の積が入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

in

fftw_complex 型、または fftwf_complex 型のポインタでなければいけません。
入力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

out

fftw_complex 型、または fftwf_complex 型のポインタで、入力配列と同じ型で
なければいけません。出力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

sign

int 型でなければいけません。
値が-1(FFTW_FORWARD)の場合は正変換、それ以外の値は逆変換が実行されます。
誤った引数を指定した場合、結果は保証されません。

flag

unsigned int 型でなければいけません。
FFTW3 と ASL/SX の機能差のため、ラッパでは値に意味は持ちません。このため、
FFTW ラッパは、本引数の値を無視します。

返却値

fftw_plan

fftw_plan 型を返却します。ASL/SX が複素 1 次元離散フーリエ変換を行うために
必要な情報が設定されます。

(5) fftw_execute()

機能
事前に生成したプランで入出力配列に対して計算を行います。

使用法
```
void  fftw_execute(const fftw_plan plan);
```

引数
plan
fftw_plan 型でなければいけません。plan は事前に fftw_plan_* で生成する必要があります。
誤った引数を指定した場合、結果は保証されません。

返却値
なし

(6) fftw_destroy_plan()

機能
プランを解放(FREE)します。

使用法
```
void  fftw_destroy_plan(fftw_plan plan);
```

引数
plan
fftw_plan 型でなければいけません。
誤った引数を指定した場合、結果は保証されません。

返却値
なし

(7) fftw_cleanup()

機能
FFTW ラッパでは意味をもちません。
ソース互換維持のために、空の関数として定義します。

使用法
```
void  fftw_cleanup(void);
```

引数
なし

返却値
なし

(8) fftw_malloc()

機能
入出力配列のメモリを確保します。

使用法
```
void  *fftw_malloc(size_t n);
```

引数
n
確保したいサイズをバイト単位で指定します。

返却値
確保したメモリへのポインタを返却します。

(9) fftw_free()

機能
入出力配列のメモリを解放します。

使用法
```
void  fftw_free(void *p);
```

引数
p
解放したいメモリを指すポインタを指定します。fftw_malloc()で確保したメモリでなければいけません。

返却値
なし

6.2 FFTW Legacy Fortran インタフェース

(1) 提供サブルーチン

以下のサブルーチンを提供します。本インタフェースで提供するのは、単精度と倍精度インタフェースです。各サブルーチンは、s で始まる名前は単精度用、d で始まる名前は倍精度であることを示しています。用途に応じて、区別して利用しなければなりません。

機能概要	単精度	倍精度
1 次元 FFT の plan 作成	sfftw_plan_dft_1d()	dfftw_plan_dft_1d()
2 次元 FFT の plan 作成	sfftw_plan_dft_2d()	dfftw_plan_dft_2d()
3 次元 FFT の plan 作成	sfftw_plan_dft_3d()	dfftw_plan_dft_3d()
plan の実行	sfftw_execute()	dfftw_execute()
plan の解放	sfftw_destroy_plan()	dfftw_destroy_plan()
plan の消去	sfftw_cleanup()	dfftw_cleanup()

Ver.1.0 では、上記以外のサブルーチンはサポートしません。また、Legacy Fortran インタフェースの場合、メモリ割り当て・解放するサブルーチンは存在しません。

利用に際しては、fftw3.f のインクルードが必要です（既存の FFTW を用いている場合は、すでにインクルードしているはずです）。

※注意※

- **ラッパ関数の引数と変数の大きさ(バイト)の対応は次のとおりでなければなりません。**

	PLAN 引数	整数型引数	複素数型
sfftw_*	8 バイト	4 バイト	8 バイト(4 バイト×2)
dfftw_*	8 バイト	4 バイト	16 バイト(8 バイト×2)

- **plan サブルーチンで指定する変換サイズを示す要素数は、2 以上でなければなりません。これは、ASL の仕様に基づくものです。**

以下に、個々のサブルーチンの仕様を示します。

(2) sfftw_plan_dft_1d(), dfftw_plan_dft_1d()

機能

複素１次元離散フーリエ変換のプランを生成します。

使用法

```
call sfftw_plan_dft_1d(plan,n,in,out,sign,flag)
call dfftw_plan_dft_1d(plan,n,in,out,sign,flag)
```

引数

plan

8 バイト整数型でなければいけません。ASL/SX が複素 1 次元離散フーリエ変換を行うために必要な情報が設定されます。
誤った引数を指定した場合、結果は保証されません。

n

4 バイト整数型でなければいけません。変換のサイズを指定します。
入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

in

sfftw_plan_dft_1d()の場合は単精度複素数型の 1 次元配列、dfftw_plan_dft_1d()の場合は倍精度複素数型の 1 次元配列でなければいけません。
入力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

out

sfftw_plan_dft_1d()の場合は単精度複素数型、dfftw_plan_dft_1d()の場合は倍精度複素数型の 1 次元配列でなければいけません。
出力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

sign

4 バイト整数型でなければいけません。
値が-1(FFTW_FORWARD)の場合は正変換、それ以外の値は逆変換が実行されます。
誤った引数を指定した場合、結果は保証されません。

flag

4 バイト整数型でなければいけません。
FFTW3 と ASL/SX の機能差のため、ラッパでは値に意味は持ちません。このため、FFTW ラッパは、この値を無視します。

(3) sfftw_plan_dft_2d(),dfftw_plan_dft_2d()

機能

複素 2 次元離散フーリエ変換のプランを生成します。

使用法

```
call sfftw_plan_dft_ 2d(plan,n1,n2,in,out,sign,flag)
call dfftw_plan_dft_ 2d(plan,n1,n2,in,out,sign,flag)
```

引数

plan

8 バイト整数型でなければいけません。ASL/SX が複素 2 次元離散
フーリエ変換を行うために必要な情報が設定されます。
誤った引数を指定した場合、結果は保証されません。

n1

4 バイト整数型で、2 以上の値でなければいけません。1 次元目の変換のサイズを指定します。
n1 と n2 の積が入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

n2

4 バイト整数型で、2 以上の値でなければいけません。2 次元目の変換のサイズを指定します。
n1 と n2 の積が入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

in

sfftw_plan_dft_ 2d の場合は単精度複素数型の 2 次元配列、dfftw_plan_dft_ 2d の場合は
倍精度複素数型の 2 次元配列でなければいけません。
入力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

out

sfftw_plan_dft_ 2d の場合は単精度複素数型の 2 次元配列、dfftw_plan_dft_ 2d の場合は
倍精度複素数型の 2 次元配列でなければいけません。
出力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

sign

4 バイト整数型でなければいけません。
値が-1(FFTW_FORWARD)の場合は正変換、それ以外の値は逆変換が実行されます。
誤った引数を指定した場合、結果は保証されません。

flag

4 バイト整数型でなければいけません。
FFTW3 と ASL/SX の機能差のため、ラッパでは値に意味は持ちません。このため、
FFTW ラッパは、この値を無視します。

(4) sfftw_plan_dft_3d(),dfftw_plan_dft_3d()

機能

複素 3 次元離散フーリエ変換のプランを生成します。

使用法

```
call sfftw_plan_dft_3d(plan,n1,n2,n3,in,out,sign,flag)
call dfftw_plan_dft_3d(plan,n1,n2,n3,in,out,sign,flag)
```

引数

plan

8 バイト整数型でなければいけません。ASL/SX が複素 3 次元離散
フーリエ変換を行うために必要な情報が設定されます。
誤った引数を指定した場合、結果は保証されません。

n1

4 バイト整数型で、2 以上の値でなければいけません。1 次元目の変換のサイズを指定します。
n1 と n2 と n3 の積が入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

n2

4 バイト整数型で、2 以上の値でなければいけません。2 次元目の変換のサイズを指定します。
n1 と n2 と n3 の積が入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

n3

4 バイト整数型で、2 以上の値でなければいけません。3 次元目の変換のサイズを指定します。
n1 と n2 と n3 の積が入出力配列のサイズでなければいけません。
誤った引数を指定した場合、結果は保証されません。

in

sfftw_plan_dft_3d の場合単精度複素数型の 3 次元配列、dfftw_plan_dft_3d の場合
倍精度複素数型の 3 次元配列でなければいけません。
入力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

out

sfftw_plan_dft_3d の場合単精度複素数型の 3 次元配列、dfftw_plan_dft_3d の場合
倍精度複素数型の 3 次元配列でなければいけません。
出力配列を指定します。
誤った引数を指定した場合、結果は保証されません。

sign

4 バイト整数型でなければいけません。
値が-1(FFTW_FORWARD)の場合は正変換、それ以外の値は逆変換が実行されます。
誤った引数を指定した場合、結果は保証されません。

flag

4 バイト整数型でなければいけません。
FFTW3 と ASL/SX の機能差のため、ラッパでは値に意味は持ちません。このため、
FFTW ラッパは、この値を無視します。

(5) sfftw_execute(),dfftw_execute()

機能
事前に生成したプランで入出力配列に対して計算を行います。

使用法
```
call sfftw_execute(plan)
call dfftw_execute(plan)
```

引数
plan
8 バイト整数型でなければいけません。plan は事前に sfftw_plan_*、または dfftw_plan_*で生成する必要があります。
誤ったプランを指定した場合、結果は保証されません。

(6) sfftw_destroy_plan(),dfftw_destroy_plan()

機能
プランを解放(FREE)します。

使用法
```
call sfftw_destory(plan)
call dfftw_destory(plan)
```

引数
plan
8 バイト整数型でなければいけません。

(7) sfftw_cleanup(),dfftw_cleanup ()

機能
FFTW ラッパでは意味をもちません。
ソース互換維持のために、空のサブルーチンとして定義します。

使用法
```
call sfftw_cleanup()
call dfftw_cleanup()
```

引数
なし

(8) その他

C インタフェースで提供している、malloc および free 各関数に対応するサブルーチンは、Legacy Fortran 版での提供はありません(FFTW の仕様です)。

6.3 Intel MKL インタフェース

(1) 提供インタフェース

以下のインタフェースを提供します。

	C 関数名または Fortran サブルーチン名
FFT 実行に必要な情報の生成	DftiCreateDescriptor
Descriptor の commit を行う	DftiCommitDescriptor
順方向の変換を行う	DftiComputeForward
逆方向の変換を行う	DftiComputeBackward
Descriptor 領域を解放する	DftiFreeDescriptor

Ver.1.0 でこれらの機能を利用する場合は、C、Fortran インタフェースそれぞれで、mkl_dfti.h、mkl_dfti.f90 のインクルードが必要です。

以下に、個々の関数・サブルーチンの仕様を示します。

なお、引数の型の詳細については、MKL のドキュメントを参照してください。

(2) DftiCreateDescriptor

機能

プランを解放(FREE)する ASL/SX が離散フーリエ変換を行うために必要な情報を設定します。

使用法

```
[Fortran]
DftiCreateDescriptor( desc_handle, precision, forward_domain,
                      dimension, length )

[C]
DftiCreateDescriptor(&desc_handle, precision, forward_domain,
                      dimension, length );
```

引数

desc_handle

ディスクリプタ・データ構造を指定します。

precision

精度を指定します。

指定できる値は、DFTI_SINGLE または DFTI_DOUBLE のいずれかです。

forward_domain

順方向領域の型を指定します。

本ラッパでは、指定して意味のある値は、DFTI_COMPLEX のみです。このため、

他の値を指定しても意味がありません。

(3) DftiCommitDescriptor

機能
本関数/サブルーチンは、ラッパで何もしません。MKL とのソース互換のために存在します。

使用法
[Fortran]
```
DftiCommitDescriptor( desc_Handle )
```

[C]
```
DftiCommitDescriptor( desc_handle );
```

引数
desc_handle
ディスクリプタ・データ構造を指定します。
ラッパではこの引数の妥当性を検査しません。

(4) DftiComputeForward

機能
複素フーリエ順変換を行います。

使用法
[Fortran]
```
DftiComputeForward( desc_handle, x_inout )
```

[C]
```
DftiComputeForward(desc_handle, x_inout);
```

引数
desc_handle
ディスクリプタ・データ構造を指定します
x_in
入力配列を指定します。
複素数型の配列でなければなりません。精度はディスクリプタ
で設定した精度と一致する必要があります。
Fortran では擬寸法階数 1 の配列 DIMENSION(0:*)として宣言されます。
y_out
出力配列を指定します。
複素数型の配列でなければなりません。精度はディスクリプタ
で設定した精度と一致する必要があります。
Fortran では擬寸法階数 1 の配列 DIMENSION(0:*)として宣言されます。

(注意)Ver.1 では、以下の関数・サブルーチンは提供しません。
DftiComputeForward(desc_handle, x_in,y_out)
DftiComputeForward(desc_handle, xre_inout, xim_inout)
DftiComputeForward(desc_handle, xre_in, xim_in, yre_out, yim_out)

(5) DftiComputeBackward

機能
複素フーリエ順変換を行います。

使用法
[Fortran]
```
DftiComputeBackward( desc_handle, x_inout )
```

[C]
```
DftiComputeBackward(desc_handle, x_inout);
```

引数
desc_handle
ディスクリプタ・データ構造を指定します

x_in
入力配列を指定します。
複素数型の配列でなければなりません。精度はディスクリプタ
で設定した精度と一致する必要があります。
Fortran では擬寸法階数 1 の配列 DIMENSION(0:*)として宣言されます。

y_out
出力配列を指定します。
複素数型の配列でなければなりません。精度はディスクリプタ
で設定した精度と一致する必要があります。
Fortran では擬寸法階数 1 の配列 DIMENSION(0:*)として宣言されます。

（注意）Ver.1 では、以下の関数・サブルーチンは提供しません。
```
DftiComputeBackward( desc_handle, x_in,y_out )
DftiComputeBackward( desc_handle, xre_inout, xim_inout )
DftiComputeBackward( desc_handle, xre_in, xim_in, yre_out, yim_out )
```

(6) DftiFreeDescriptor

機能
ディスクリプタに割り当てられていたメモリを解放します。

使用法
[Fortran]
```
DftiFreeDescriptor( desc_Handle )
```

[C]
```
DftiFreeDescriptor( &desc_handle );
```

引数
desc_handle
ディスクリプタ・データ構造を指定します

7. 利用方法

7.1 FFTW ラッパ

7.1.1 コンパイルおよびリンク時オプション

(1) C インタフェース

クロス環境では、sxc++コマンド、SUPER-UX 上では c++コマンドを用います。コンパイルおよびリンク時には、以下のオプションを指定しなければなりません。

オプション	説明
-I ディレクトリ名	ヘッダファイル fftw3.h があるディレクトリを指定します
-L ディレクトリ名	ラッパのアーカイブファイル(libfftw2asl.a)があるディレクトリを指定します
-f90lib	FORTRAN90/SX のライブラリを利用することを指定します(ASL/SX が使用する)
-lfftw2aslf または -lfftw2asl	単精度用ラッパのアーカイブファイル(fftw2aslf)、または倍精度用ラッパのアーカイブファイル(fftw2asl)を指定します
-laslcint	ASL C インタフェース用のライブラリを指定します
-lasl	ASL/SX のライブラリを指定します

※注意※

- **単精度と倍精度の場合で、リンクするライブラリが異なります。ライブラリ名を誤って指定した場合、実行結果は不正になります。**
- **リンク時に指定するライブラリのリンク順は重要です。リンク順を誤って指定すると、リンク時にシンボル未定義のエラーとなるか、もしリンクが成功しても実行結果は不正になります。**
- **コンパイル時に、-size_t64 を指定してはいけません。指定した場合の動作は保証しません。**

以下に指定形式と例を示します。なお、ここで、ASL/SX は、通常のインストール手順によりインストールされ、利用できる環境が設定されているものとします。

- 指定形式

```
sxc++ -I$(INCDIR) [他オプション] ソースファイル
                -L$(LIBDIR) -lf90lib -l{fftw2aslf|fftw2asl} -laslcint -lasl
```

※$(INCDIR)、$(LIBDIR)はそれぞれ、ラッパのヘッダファイル、アーカイブファイルがあるディレクトリを示します。

※セルフコンパイラ環境では、コマンド名は c++となります。

- 使用例

FFTW を用いた倍精度の利用者プログラム example.c を、クロス環境でコンパイル、リンクする場合の例を示します。

```
sxc++ -I/opt/fftw2asl/include example.c
```

```
-L/opt/fftw2asl/lib -lf90lib -lfftw2asl -laslcint -lasl
```

その他のオプションは、C++/SX、ASL/SX、および ASL C インタフェースの説明書を参照してください。

(2) Legacy Fortran インタフェース

クロス環境では、sxf90 コマンド、SUPER-UX 上では f90 コマンドを用います。コンパイルおよびリンク時には、以下のオプションを指定しなければなりません。

オプション	説明
-Ep	コンパイル時に C プリプロセッサを起動します
-I ディレクトリ名	ヘッダファイル fftw3.f があるディレクトリを指定します
-L ディレクトリ名	アーカイブファイル libfftw2asl.a があるディレクトリを指定します
-lfftw2asl	ライブラリファイルを指定します(Legacy Fortran 版では、単精度と倍精度でライブラリを切り分ける必要はありません)
-lasl	ASL/SX のライブラリファイルを指定します

その他のオプションは、FORTRAN90/SX の説明書を参照してください。

以下に指定形式と例を示します。なお、ここで、ASL/SX は、通常のインストール手順によりインストールされ、利用できる環境が設定されているものとします。

- 指定形式

```
sxf90 -Ep -I$(INCDIR) [他オプション] ソースファイル -L$(LIBDIR) -lfftw2asl -lasl
```

※$(INCDIR)、$(LIBDIR)はそれぞれ、ラッパのヘッダファイル、アーカイブファイルがあるディレクトリを示します。

※セルフコンパイラ環境では、コマンド名は f90 となります。

- 使用例

FFTW を用いた利用者プログラム example.f を、クロス環境でコンパイル、リンクする場合の例を示します。

```
sxf90 -Ep -I/opt/fftw2asl/include example.f -L/opt/fftw2asl/lib -lfftw2asl -lasl
```

その他のオプションは、FORTRAN90/SX、および ASL/SX の説明書を参照してください。

7.1.2 実行時オプション

実行時には、ラッパとして指定すべきオプションはありません。

C++/SX、FORTRAN90/SX で必要なオプションがあれば、それを指定してください。

7.2 MKL ラッパ

7.2.1 コンパイルおよびリンク時オプション

(1) C インタフェース

クロス環境では、sxc++コマンド、SUPER-UX 上では c++コマンドを用います。コンパイルおよびリンク時には、以下のオプションを指定しなければなりません。

オプション	説明
-I ディレクトリ名	ヘッダファイル mkl_dfti.h があるディレクトリを指定します
-L ディレクトリ名	ラッパのアーカイブファイル(libmkl2asl.a)があるディレクトリを指定します
-lf90lib	ASL/SX が使用する FORTRAN90/SX のライブラリを指定します
-lmkl2asl	ラッパのアーカイブファイルを指定します
-laslcint	ASL C インタフェース用のライブラリを指定します
-lasl	ASL/SX のライブラリを指定します

※注意※

- **リンク時に指定するライブラリのリンク順は重要です。リンク順を誤って指定すると、リンク時にシンボル未定義のエラーとなるか、もしリンクが成功しても実行結果は不正になります。**
- **コンパイル時に、-size_t64 を指定してはいけません。指定した場合の動作は保証しません。**

以下に指定形式と例を示します。なお、ここで、ASL/SX は、通常のインストール手順によりインストールされ、利用できる環境が設定されているものとします。

- 指定形式

```
sxc++ -I$(INCDIR) [他オプション] ソースファイル
                -L$(LIBDIR) -lf90lib -lmkl2asl -laslcint -lasl
```

※$(INCDIR)、$(LIBDIR)はそれぞれ、ラッパのヘッダファイル、アーカイブファイルがあるディレクトリを示します。

※セルフコンパイラ環境では、コマンド名は c++となります。

- 使用例

FFTW を用いた利用者プログラム example.c を、クロス環境でコンパイル、リンクする場合の例を示します。

```
sxc++ -I/opt/fftw2asl/include example.c
          -L/opt/fftw2asl/lib -lf90lib -lmkl2asl -laslcint -lasl
```

その他のオプションは、C++/SX、ASL/SX、および ASL C インタフェースの説明書を参照してください。

(2) Legacy Fortran インタフェース

クロス環境では、sxf90 コマンド、SUPER-UX 上では f90 コマンドを用います。コンパイルおよびリンク時には、以下のオプションを指定しなければなりません。

オプション	説明
-Ep	コンパイル時に C プリプロセッサを起動します
-I ディレクトリ名	ヘッダファイル mkl_dfti.f90 があるディレクトリを指定します
-L ディレクトリ名	アーカイブファイル libmkl2asl.a があるディレクトリを指定します
-lmkl2asl	ライブラリファイルを指定します
-lasl	ASL/SX のライブラリファイルを指定します

その他のオプションは、FORTRAN90/SX の説明書を参照してください。

以下に指定形式と例を示します。なお、ここで、ASL/SX は、通常のインストール手順によりインストールされ、利用できる環境が設定されているものします。

- 指定形式

```
sxf90 -I$(INCDIR) [他オプション] ソースファイル -L$(LIBDIR) -lmkl2asl -lasl
```

※$(INCDIR)、$(LIBDIR)はそれぞれ、ラッパのヘッダファイル、アーカイブファイルがあるディレクトリを示します。

※セルフコンパイラ環境では、コマンド名は f90 となります。

- 使用例

FFTW を用いた利用者プログラム example.f を、クロス環境でコンパイル、リンクする場合の例を示します。ここでは、ヘッダファイルが/opt/mkl2asl/include に、ライブラリが/opt/mkl2asl/lib 下に格納されていることを前提としています。

```
sxf90 -I/opt/mkl2asl/include example.f -L/opt/mkl2asl/lib -lmkl2asl -lasl
```

その他のオプションは、FORTRAN90/SX、および ASL/SX の説明書を参照してください。

7.2.2 実行時オプション

実行時には、ラッパとして指定すべきオプションはありません。

C++/SX、FORTRAN90/SX で必要なオプションがあれば、それを指定してください。

8. エラーメッセージ

本節では、ラッパが出力するメッセージについて説明します。

メッセージ	説明
Illegal PLAN	実行しようとしたplanが誤っています。 planを正しく生成していないか、planを生成しないまま execute を呼び出している可能性があります。
Insufficient memory.	入出力領域を確保できません。サイズを小さくするか、他で使用している領域を削減できないか確かめてください。
The size of the transform is invalid	変換サイズが誤っています。 ASLの仕様上、2次元以上の入出力データは、各次元のサイズが2以上でなければなりません。
Internal Error : 詳細メッセージ	ラッパシステムの予期しない動作時に出力します。このメッセージが生成された場合、ラッパの運用責任者に連絡して下さい。

9. 注意制限事項

注意事項は、前節までの説明に記述していますが、本節で再掲します。

項番	注意事項	関係する節
(1)	ラッパの plan 関数/サブルーチンでは、指定する変換サイズを示す要素数は、2以上でなければなりません	6.1、6.2
(2)	リンク時に指定するライブラリ順は、本書に記載どおりに指定してください。	7.1、7.2
(3)	C インタフェースの場合、-size_64t オプションの指定はできません。	7.1、7.2

10.おわりに

「0

はじめに」でも述べたように、x86 システムで流通しているアプリケーションを、SX 上で利用することで、利用者の研究開発の効率化が期待できます。

本機能により、SX システムの利用促進が向上することが期待できます。

以上