地球流体電脳ライブラリ

グラフィック関数

(Fortran 90 版)

地球流体電脳倶楽部

2015 年 7 月 11 日

目次

第 I 部

2D グラフィックス

第 1 章

デバイス制御

1.1 ルーチンリスト

DclSelectDevice	グラフィックデバイス番号をコンソールから取得する.
(新規導入)	
DclOpenGraphics	グラフィックデバイスをオープンする.
(GROPN)	
DclNewFrame	新たなフレーム (ページ) を設定する.
(GRFRM)	
DclNewFig	新たな図を設定する.
(GRFIG)	
DclPrintDeviceList	ワークステーション名のリストを表示する.
(SGPWSN)	
DclCloseGraphics	グラフィックデバイスをクローズする.
(GRCLS)	

* 括弧の中は, 対応する f77 インターフェイス名.

1.2 各ルーチンの説明

1.2.1 DclSelectDevice

1. 機能

   使用可能なグラフィックデバイスを表示し, デバイス番号をコンソールから取得する.

2. 書式

   ```
   result=DclSelectDevice()
   ```

3. 引数

   戻り値　<I>　選択されたグラフィックデバイス番号.

4. 備考

   なし.

5. 関連項目

- 関連ルーチン (デバイス)

1.2.2 DclOpenGraphics

1. 機能
   グラフィックデバイスをオープンする.
2. 書式
   ```
   call DclOpenGraphics([ws_id])
   ```
3. 引数
   ws_id <I> デバイス番号を指定する.
   省略時はコマンドラインからデバイス番号を入力する.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (デバイス)

1.2.3 DclNewFrame

1. 機能
   新たなフレーム (ページ) を設定する.
2. 書式
   ```
   call DclNewFrame()
   ```
3. 引数なし.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (デバイス)

1.2.4 DclNewFig

1. 機能
   新たな図を設定する.
2. 書式
   ```
   call DclNewFig()
   ```
3. 引数なし.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (デバイス)

1.2.5 DclPrintDeviceList

1. 機能

   ワークステーション名のリストを表示する.

2. 書式

   ```
   call DclPrintDeviceList()
   ```

3. 引数なし.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (デバイス)

1.2.6 DclCloseGraphics

1. 機能

   グラフィックデバイスをクローズする.

2. 書式

   ```
   call DclCloseGraphics()
   ```

3. 引数なし.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (デバイス)

第 2 章

レイアウト

2.1 ルーチンリスト

2.1.1 主要ルーチン

DclSetFrameSize	第 1 フレームの大きさを設定する.
(SLSIZE,SLFORM)	
DclDivideFrame	フレームを分割する.
(SLDIV)	
DclSetFrameMargin	マージンを設定する.
(SLMGN)	
DclSetAspectRatio	縦横比を設定する.
(SLRAT)	
DclSetFrameTitle	タイトルを設定する.
(SLSTTL)	

2.1.2 設定

DclDrawViewPortFrame	ビューポートの枠を描く.
(SLPVPR)	
DclDrawDeviceWindowFrame	ウインドウの枠を描く.
(SLPWWR)	
DclDrawDeviceViewPortFrame	最大作画領域の枠を描く.
(SLPWVR)	
DclDrawViewPortCorner	ビューポートのコーナーマークを描く.
(SLPVPC)	
DclDrawDeviceWindowCorner	ウインドウのコーナーマークを描く.
(SLPWWC)	
DclDrawDeviceViewPortCorner	デバイスビューポートの
(SLPWVC)	コーナーマークを描く.

* 括弧の中は, 対応する f77 インターフェイス名.

2.2 各ルーチンの説明

2.2.1 DclSetFrameSize

1. 機能

   第 1 フレームの大きさを設定する.

2. 書式

   `call DclSetFrameSize(size)` `call DclSetFrameSize(width,height)`

3. 引数

`size`	`<C3>`	フレームの大きさと向きを指定する. 最初の 2 文字が大きさ, 3 文字目が向きを表す. 指定方法は備考参照.
`width`	`<R>`	フレームの横の実長. 単位は cm.
`height`	`<R>`	フレームの縦の実長. 単位は cm.

4. 備考

   - フレームの大きさは, A 系列または B 系列の規格 ('A4','B5' など) で指定する.
   - フレームの向きとして指定できる文字は以下のとおり.

'P' (Portrait) 'T' (Tate)	縦長
'L' (Landscape) 'Y' (Yoko)	横長
'A' (Auto)	最大作図範囲の縦・横比に合わせる.

5. 関連項目

   - 関連ルーチン (layout)

2.2.2 DclDivideFrame

1. 機能

   フレームを分割する.

2. 書式

   `call DclDivideFrame(direction, x_num, y_num)`

3. 引数

`direction`	`<C1>`	割りつける方向. 縦方向に割り付ける場合, 'T'ate または 'L'engthways, 横方向に割り付ける場合, 'Y'oko または 'S'ideways.
`x_num,`	`<I>`	X 方向, Y 方向の分割数.
`y_num,`		

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (layout)

2.2.3 DclSetFrameMargin

1. 機能

   マージンを設定する.

2. 書式

```
call DclSetFrameMargin(left, right, bottom, top)
```

3. 引数

   `left, right <R>` 左右のマージン. 横方向の幅を 1 とする比率で指定する.

   `top, bottom <R>` 上下のマージン. 縦方向の幅を 1 とする比率で指定する.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (layout)

2.2.4 DclSetAspectRatio

1. 機能

   縦横比を設定する.

2. 書式

```
call DclSetAspectRatio(x, [y])
```

3. 引数

   `x <R>` フレームの x 方向の長さ. 単位は任意.

   `y <R>` フレームの y 方向の長さ. 単位は任意. 省略時は 1.

4. 備考

   - フレームの縦横の比率だけが問題となるので, 単位は任意で良い.
   - X, Y $>$ 0 でなければならない.

5. 関連項目

   - 関連ルーチン (layout)

2.2.5 DclSetFrameTitle

1. 機能

   第 1 レベル目のトップマージンまたはボトムマージンに描くタイトルを設定する.

2. 書式

```
call DclSetFrameTitle(title, side, x_position, y_position, &
& height, [num])
```

3. 引数

title	<C*>	タイトル文字列.
side	<C1>	タイトルを書く場所. 'T' (Top) または 'B' (Bottom).
x_position	<R>	マージン内における文字列の横方向の位置.
		-1. 左寄せ
		0. 中央合わせ
		1. 右寄せ
y_position	<R>	マージン内における文字列の縦方向の位置.
		-1. 下寄せ
		0. 中央合わせ
		1. 上寄せ
height	<R>	文字の高さ.
num	<I>	複数 (最大 5) の文字列を書く時, 何番目の文字列かを指定する. 省略値 1.

4. 備考
   - このルーチンは文字列を設定するだけで, 文字列が描かれるのは "DclNewFrame" が呼ばれた時である.
   - このルーチンは, "DclNewFrame" のあとでも呼べる. これによりページごとに異なるタイトルを書くことができる.
   - 設定した文字列を無効にしたいときは

     ```
     CALL SLDTTL(NT)
     ```

     とすると, 第 NT 番目の設定が無効になる.
   - DCL パラメタ DRAW_PAGE_TITLE を .FALSE. とするとタイトルを描かない (初期値は.TRUE.).
5. 関連項目
   - 関連ルーチン (layout)

2.2.6 DclDrawViewPortFrame

1. 機能

   ビューポートの枠を描く.
2. 書式

   ```
   call DclDrawViewPortFrame(index)
   ```
3. 引数

index	<I>	枠のラインインデクス.
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (layout)

2.2.7 DclDrawDeviceWindowFrame

1. 機能

   ウインドウの枠を描く.

2. 書式

```
call DclDrawDeviceWindowFrame(index)
```

3. 引数

   index <I> 枠のラインインデクス.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (layout)

2.2.8 DclDrawDeviceViewPortFrame

1. 機能

   最大作画領域の枠を描く.

2. 書式

```
call DclDrawDeviceViewPortFrame(index)
```

3. 引数

   index <I> 枠のラインインデクス.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (layout)

2.2.9 DclDrawViewPortCorner

1. 機能

   ビューポートのコーナーマークを描く.

2. 書式

```
call DclDrawViewPortCorner(index, size)
```

3. 引数

   index <I> コーナーマークのラインインデクス.

   size <R> コーナーマークのサイズ. 単位は R 座標系.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (layout)

2.2.10 DclDrawDeviceWindowCorner

1. 機能

   ウインドウのコーナーマークを描く.

2. 書式

   ```
   call DclDrawDeviceWindowCorner(index, size)
   ```

3. 引数

index	<I>	コーナーマークのラインインデクス.
size	<R>	コーナーマークのサイズ. 単位は R 座標系.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (layout)

2.2.11 DclDrawDeviceViewPortCorner

1. 機能

   デバイスビューポートのコーナーマークを描く.

2. 書式

   ```
   call DclDrawDeviceViewPortCorner(index, size)
   ```

3. 引数

index	<I>	コーナーマークのラインインデクス.
size	<R>	コーナーマークのサイズ. 単位は R 座標系.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (layout)

第 3 章

正規変換

3.1 ルーチンリスト

3.1.1 変換パラメタ設定

DclSetTransNumber (GRSTRN)	変換関数番号を設定する.
SetViewPort (GRSVPT)	ビューポートを設定する.
SetWindow (GRSWND)	ウインドウを設定する.
SetSimilarity (GRSSIM)	相似変換を設定する.
SetMapProjectionAngle (GRSMPL)	地図投影の角度を設定する.
SetMapProjectionWindow (GRSTXY)	地図投影ウインドを設定する.

3.1.2 関数確定

DclSetTransFunction (GRSTRF)	変換関数を確定する.

3.1.3 変換パラメタ参照

DclGetTransNumber (SGQTRN)	変換関数番号を参照する.
DclGetViewPort (SGQVPT)	ビューポートの設定値を参照する.
DclGetWindow (SGQWND)	ウィンドウの設定値を参照する.
DclGetSimilarity (SGQSIM)	相似変換の設定値を参照する.
DclGetMapProjectionAngle (SGQMPL)	地図投影の投影角を参照する.

3.1.4 その他

DclGetTransNumber (SGQTRN)	変換関数番号を参照する.
DclTransShortToLong (SGTRSL)	変換関数の略称から名称を求める.
DclTransShortToNum (SGTRSN)	変換関数の略称から関数番号を求める.
DclTransLongToShort (SGTRLS)	変換関数の名称から略称を求める.
DclTransLongToNum (SGTRLN)	変換関数の名称から変換関数番号を求める.
DclTransNumToShort (SGTRNS)	変換関数番号から略称を求める.
DclTransNumToLong (SGTRNL)	変換関数番号から名称を求める.

* 括弧の中は, 対応する f77 インターフェイス名.

3.2 各ルーチンの説明

3.2.1 DclSetTransNumber

1. 機能

   変換関数番号を設定する.

2. 書式

```
call DclSetTransNumber(num)
```

3. 引数

num	<I>	変換関数番号を指定する.

4. 備考
   - ここで設定した値は, DclSetTransFunction を呼ぶことで有効になる.
5. 関連項目
   - 関連ルーチン (controle)

3.2.2 DclSetViewPort

1. 機能
   ビューポートを設定する.
2. 書式
   ```
   call DclSetViewPort([xmin], [xmax], [ymin], [ymax])
   ```
3. 引数
   xmin, xmax <R> ビューポートの x 方向の最大最小値.
   ymin, ymax <R> ビューポートの y 方向の最大最小値.
4. 備考
   - ここで設定した値は, DclSetTransFunction を呼ぶことで有効になる.
5. 関連項目
   - 関連ルーチン (controle)

3.2.3 DclSetWindow

1. 機能
   ウィンドウを設定する.
2. 書式
   ```
   call DclSetWindow([xmin], [xmax], [ymin], [ymax])
   ```
3. 引数
   xmin, xmax <R> ウィンドウの x 方向の最大最小値.
   ymin, ymax <R> ウィンドウの y 方向の最大最小値.
4. 備考
   - ここで設定した値は, DclSetTransFunction を呼ぶことで有効になる.
5. 関連項目
   - 関連ルーチン (controle)

3.2.4 DclSetSimilarity

1. 機能
   相似変換を設定する.
2. 書式
   ```
   call DclSetSimilarity([factor], [xoffset], [yoffset])
   ```
3. 引数

factor	<R>	相似変換のスケーリングファクター.
xoffset, yoffset	<R>	原点のオフセット. これらが 0 のとき，ビューポートの中心に原点が設定される.

4. 備考
   - ここで設定した値は, DclSetTransFunction を呼ぶことで有効になる.
5. 関連項目
   - 関連ルーチン (controle)

3.2.5 DclSetMapProjectionAngle

1. 機能
   地図投影の角度を設定する.
2. 書式
   ```
   call DclSetMapProjectionAngle([longitude], [latitude], [rotation])
   ```
3. 引数

longitude	<R>	投影座標 (TC) の極を置く経度.
latitude	<R>	投影座標 (TC) の極を置く緯度.
rotation	<R>	投影座標の極の周りの回転角. 経度 longitude と投影座標の中央経線のなす角度を指定する.

4. 備考
   - ここで設定した値は, DclSetTransFunction を呼ぶことで有効になる.
5. 関連項目
   - 関連ルーチン (controle)

3.2.6 DclSetMapProjectionWindow

1. 機能
   地図投影ウィンドウを設定する.
2. 書式
   ```
   call DclSetMapProjectionWindow([xmin], [xmax], [ymin], [ymax])
   ```
3. 引数

xmin, xmax	<R>	地図投影のウィンドウ (経度方向).
ymin, ymax	<R>	地図投影のウィンドウ (緯度方向).

4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (controle)

3.2.7 DclSetTransFunction

1. 機能

変換関数を確定する.

2. 書式

```
call DclSetTransFunction()
```

3. 引数なし.
4. 備考

なし.

5. 関連項目
   - 関連ルーチン (controle)

3.2.8 DclGetTransNumber

1. 機能

変換関数番号を参照する.

2. 書式

```
result=DclGetTransNumber()
```

3. 引数

戻り値 <I> 変換関数番号.

4. 備考

なし.

5. 関連項目
   - 関連ルーチン (controle)

3.2.9 DclGetViewPort

1. 機能

ビューポートの設定値を参照する.

2. 書式

```
call DclGetViewPort([xmin], [xmax], [ymin], [ymax])
```

3. 引数

xmin, xmax <R> ビューポートの x 方向の最大最小値.

ymin, ymax <R> ビューポートの y 方向の最大最小値.

4. 備考

なし.

5. 関連項目
   - 関連ルーチン (controle)

3.2.10 DclGetWindow

1. 機能

ウィンドウの設定値を参照する.

2. 書式

```
call DclGetWindow([xmin], [xmax], [ymin], [ymax])
```

3. 引数

xmin, xmax	<R>	ウィンドウの x 方向の最大最小値.
ymin, ymax	<R>	ウィンドウの y 方向の最大最小値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (controle)

3.2.11 DclGetSimilarity

1. 機能

相似変換の設定値を参照する.

2. 書式

```
call DclGetSimilarity([factor], [xoffset], [yoffset])
```

3. 引数

factor	<R>	相似変換のスケーリングファクター.
xoffset, yoffset	<R>	原点のオフセット. これらが 0 のとき, ビューポートの中心に原点が設定される.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (controle)

3.2.12 DclGetMapProjectionAngle

1. 機能

地図投影の投影角を参照する.

2. 書式

```
call DclGetMapProjectionAngle([longitude], [latitude], [rotation])
```

3. 引数

longitude	<R>	投影座標 (TC) の極を置く経度.
latitude	<R>	投影座標 (TC) の極を置く緯度.
rotation	<R>	投影座標の極の周りの回転角. 経度 longitude と投影座標の中央経線のなす角度を指定する.

4. 備考

5. 関連項目

- 関連ルーチン (controle)

3.2.13 DclTransShortToLong

1. 機能

   変換関数の略称から名称を求める.

2. 書式

   ```
   call DclTransShortToLong(short, long)
   ```

3. 引数

short	<C*>	変換関数の略称.
long	<C*>	変換関数の名称.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (controle)

3.2.14 DclTransShortToNum

1. 機能

   変換関数の略称から関数番号を求める.

2. 書式

   ```
   call DclTransShortToNum(short, num)
   ```

3. 引数

short	<C*>	変換関数の略称.
num	<C*>	変換関数の番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (controle)

3.2.15 DclTransLongToShort

1. 機能

   変換関数の名称から略称を求める.

2. 書式

   ```
   call DclTransLongToShort(long, short)
   ```

3. 引数

long	<C*>	変換関数の名称.
short	<C*>	変換関数の略称.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (controle)

3.2.16 DclTransLongToNum

1. 機能

   変換関数の名称から変換関数番号を求める.

2. 書式

   ```
   call DclTransLongToNum(long, num)
   ```

3. 引数

long	<C*>	変換関数の名称.
num	<I>	変換関数の番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (controle)

3.2.17 DclTransNumToShort

1. 機能

   変換関数番号から略称を求める.

2. 書式

   ```
   call DclTransNumToShort(num, short)
   ```

3. 引数

num	<C*>	変換関数の番号.
short	<C*>	変換関数の略称.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (controle)

3.2.18 DclTransNumToLong

1. 機能

   変換関数番号から名称を求める.

2. 書式

   ```
   call DclTransNumToLong(num, long)
   ```

3. 引数

num	<C*>	変換関数の番号.
long	<C*>	変換関数の名称.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (controle)

第 4 章

ライン

4.1 概要

4.1.1 ラインインデクス

線分の太さと色はラインインデクスと呼ばれる 3 桁の整数 (nnm) で指定される. 線の太さと色のうちどちらか一方しか変えられないようなデバイスに出力する場合でも, ラインインデクスの異なる 2 本の線が識別できるようにするため, ラインインデクスは次のような規則にしたがって太さと色に対応づけられる.

線の太さと色が両方変えられるようなシステムでは, 上位 2 桁 (nn = 0 - 99) が色番号, 下位 1 桁 (m = 0 - 9) が線の太さを表す. 色番号は, 1 から 5 までは標準的に

1: 白または黒 (フォアグラウンド)
2: 赤
3: 緑
4: 青
5: 黄

と決められているが, それより大きな番号に関しては colormap ファイルの定義による. また m : 1 (細) -¿ 9 (太) となっている. 線の太さだけが変えられるような出力装置では, m = 0 のときに限って nn を m として読みかえる. また, 線の色だけがかえられるような出力装置では, nn = 0 のときに限って m を nn として読みかえる. したがって線の太さと色を明示したいとき以外は, 1 桁のラインインデクスを指定しておけば, とりあえず装置に固有な方法によって線分は識別可能となる.

4.1.2 ラインタイプ

ラインタイプとは, 実線, 破線などの線種である. 1 から 4 までの番号にはあらかじめ以下のタイプが決められている.

1: 実線
2: 破線
3: 点線
4: 1 点鎖線

その他の 0 以外の整数は下位 N ビット (N は内部変数 'PATTERN_BIT_LENGTH' で決まる値. 初期値は 16) のビットパターンを用いて線種が設定される. たとえば N = 16 で TYPE = Z'0000F0F0' (16 進定数) のとき, '4bits ON 4bits OFF 4bits ON 4bits OFF' のような破線が設定される. 1 ビット当たりの長さは内部変数 'LINE_BIT_LENGTH' が決める.

4.1.3 ラベル

内部変数 'ENABLE_LINE_LABELING ' を .TRUE. にするとラベル付きの折れ線を描く. ここでいうラベル付き折れ線とは, 描くべき線分のある長さを 1 サイクルとして, その一部分を空白域としその空白部分に指定した文字列を描くものである. ラベルとしてつける文字列は DclSetLineText で指定する. その文字列の高さは DclSetLineTextSize で指定する. サイクルの定義などに関しては折れ線に関する内部変数を参照のこと.

なお, 内部変数 'ENABLE_LINE_LABELING' を .TRUE. とした効果は線分描画ルーチンすべてに及ぶ. したがって, 'ENABLE_LINE_LABELING' を .TRUE. としてラベルつき線分を描いたあとは必ず .FALSE. に戻しておかなくてはいけない.

4.1.4 欠損値処理

内部変数 'INTERPRET_MISSING_VALUE' を .TRUE. にすると欠損値処理をおこなう. つまり欠損値の前後は線で結ばない.

4.2 ルーチンリスト

4.2.1 描画

DclDrawLine (SGPLU,SGPLZU)	ユーザー座標系で折れ線を描く.
DclDrawLineNormalized (SGPLV,SGPLZV)	正規座標系で折れ線を描く.
DclDrawLineProjected (SGPLR,SGPLZR)	透視座標系で折れ線を描く.

4.2.2 設定

DclSetLineType	折れ線のラインタイプを設定する.
(SGSPLT)	
DclSetLineIndex	折れ線のラインインデクスの設定する.
(SGSPLI)	
DclSetLineText	折れ線のラベルの文字列を設定する.
(SGSPLC)	
DclSetLineTextSize	折れ線のラベルの文字高を設定する.
(SGSPLS)	
DclNextLineText	折れ線のラベルの最後の文字番号を増やす.
(SGNPLC)	

4.2.3 参照

DclGetLineType	折れ線のラインタイプを参照する.
(SGQPLT)	
DclGetLineIndex	折れ線のラインインデクスを参照する.
(SGQPLI)	
DclGetLineText	折れ線のラベルの文字列を参照する.
(SGQPLC)	
DclGetLineTextSize	折れ線のラベルの文字高を参照する.
(SGQPLS)	

* 括弧の中は, 対応する f77 インターフェイス名.

4.3 各ルーチンの説明

4.3.1 DclDrawLine

1. 機能

   ユーザー座標系で折れ線を描く.

2. 書式

   ```
   call DclDrawLine(x, y, [type], [index])
   ```

3. 引数

x,y	<R(:)>	折線の座標値 (ユーザー座標系).
type	<I>	折線のラインタイプ.
index	<I>	折線のラインインデクス.

4. 備考

   - type, index を省略したときは, DclSetLineType, DclSetLineIndex によって設定された値が使われる. 初期値はどちらも 1.

- ここで設定した type, index は, DclSetLineType, DclSetLineIndex によって設定された値を変更しない.

5. 関連項目
   - 関連ルーチン (line)

4.3.2 DclDrawLineNormalized

1. 機能

   正規座標系で折れ線を描く.

2. 書式

```
call DclDrawLineNormalized(x, y, [type], [index])
```

3. 引数

x,y	<R(:)>	折線の座標値 (正規座標系).
type	<I>	折線のラインタイプ.
index	<I>	折線のラインインデクス.

4. 備考
   - type, index を省略したときは, DclSetLineType, DclSetLineIndex によって設定された値が使われる. 初期値はどちらも 1.
   - ここで設定した type, index は, DclSetLineType, DclSetLineIndex によって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (line)

4.3.3 DclDrawLineProjected

1. 機能

   透視座標系で折れ線を描く.

2. 書式

```
call DclDrawLineProjected(x, y, [type], [index])
```

3. 引数

x,y	<R(:)>	折線の座標値 (透視座標系).
type	<I>	折線のラインタイプ.
index	<I>	折線のラインインデクス.

4. 備考
   - type, index を省略したときは, DclSetLineType, DclSetLineIndex によって設定された値が使われる. 初期値はどちらも 1.
   - ここで設定した type, index は, DclSetLineType, DclSetLineIndex によって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (line)

4.3.4 DclSetLineType

1. 機能

   折れ線のラインタイプを設定する.

2. 書式

   ```
   call DclSetLineType(type)
   ```

3. 引数

type	<I>	折線のラインタイプ.

4. 備考

   - ここで設定した値は, 折線を描く際のデフォルト値となる.

5. 関連項目

   - 関連ルーチン (line)

4.3.5 DclSetLineIndex

1. 機能

   折れ線のラインインデクスを設定する.

2. 書式

   ```
   call DclSetLineIndex(index)
   ```

3. 引数

index	<I>	折線のラインインデクス.

4. 備考

   - ここで設定した値は, 折線を描く際のデフォルト値となる.

5. 関連項目

   - 関連ルーチン (line)

4.3.6 DclSetLineText

1. 機能

   折れ線のラベルの文字列を設定する.

2. 書式

   ```
   call DclSetLineText(text)
   ```

3. 引数

text	<C*>	ラベルの文字列.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (line)

4.3.7 DclSetLineTextSize

1. 機能

   折れ線のラベルの文字高を設定する.

2. 書式

   ```
   call DclSetLineTextSize(height)
   ```

3. 引数

   `height` `<R>` 説明ラベルの文字高.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (line)

4.3.8 DclGetLineType

1. 機能

   折れ線のラインタイプ (デフォルト値) を参照する.

2. 書式

   ```
   result=DclGetLineType()
   ```

3. 引数

   戻り値 `<I>` ラインタイプ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (line)

4.3.9 DclGetLineIndex

1. 機能

   折れ線のラインインデクス (デフォルト値) を参照する.

2. 書式

   ```
   result=DclGetLineIndex()
   ```

3. 引数

   戻り値 `<I>` ラインインデクス.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (line)

4.3.10 DclGetLineText

1. 機能

   折れ線のラベルの文字列を参照する.

2. 書式

   ```
   call DclGetLineText(text)
   ```

3. 引数

   `text` `<C*>` ラベルの文字列.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (line)

4.3.11 DclGetLineTextSize

1. 機能

   折れ線のラベルの文字高を参照する.

2. 書式

   ```
   result=DclGetLineTextSize()
   ```

3. 引数

   戻り値 `<R>` ラベルの文字高.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (line)

第 5 章

マーカー

5.1 概要

欠損値処理

内部変数 'INTERPRET_MISSING_VALUE ' を .TRUE. にすると欠損値処理をおこなう. つまり欠損値の点にはマーカーを打たない.

また, 内部変数 'POLIMARKER_INTERVAL' を変更することによって何点かに 1 個の割合でマーカーを描くこともできる.

5.2 ルーチンリスト

5.2.1 描画

DclDrawMarker (SGPMU,SGPMZU)	ユーザー座標系でマーカー列を描く.
DclDrawMarkerNormalized (SGPMV,SGPMZV)	正規座標系でマーカー列を描く.
DclDrawMarkerProjected (SGPMR,SGPMZR)	透視座標系でマーカー列を描く.

5.2.2 設定

DclSetMarkerType (SGSPMT)	マーカータイプを設定する.
DclSetMarkerIndex (SGSPMI)	マーカーのラインインデクスを設定する.
DclSetMarkerSize (SGSPMS)	マーカーの大きさを設定する.

5.2.3 参照

DclGetMarkerType	マーカータイプを参照する.
(SGQPMT)	
DclGetMarkerIndex	マーカーのラインインデクスを参照する.
(SGQPMI)	
DclGetMarkerSize	マーカーの大きさを参照する.
(SGQPMS)	

* 括弧の中は, 対応する f77 インターフェイス名.

5.3 各ルーチンの説明

5.3.1 DclDrawMarker

1. 機能

   ユーザー座標系でマーカー列を描く.

2. 書式

   ```
   call DclDrawMarker(x, y, [type], [index], [height])
   ```

3. 引数

x,y	<R(:)>	マーカーの座標値 (ユーザー座標系).
type	<I>	マーカーのタイプ (文字コード).
index	<I>	マーカーのラインタイプ.
height	<I>	マーカーの高さ.

4. 備考

   - type, index, height を省略したときは, DclSetMarkerType, DclSetMarkerIndex, DclSetMarkerSize によって設定された値が使われる. 初期値はそれぞれ 1, 1, 0.01.
   - ここで設定した type, index, height は, DclSetMarkerType, DclSetMarkerIndex, DclSetMarkerSize によって設定された値を変更しない.

5. 関連項目

   - 関連ルーチン (marker)

5.3.2 DclDrawMarkerNormalized

1. 機能

   正規座標系でマーカー列を描く.

2. 書式

   ```
   call DclDrawMarkerNormalized(x, y, [type], [index], [height])
   ```

3. 引数

x,y	<R(:)>	マーカーの座標値 (正規座標系).
type	<I>	マーカーのタイプ (文字コード).
index	<I>	マーカーのラインタイプ.
height	<I>	マーカーの高さ.

4. 備考
   - type, index, height を省略したときは, DclSetMarkerType, DclSetMarkerIndex, DclSetMarkerSize によって設定された値が使われる. 初期値はそれぞれ 1, 1, 0.01.
   - ここで設定した type, index, height は, DclSetMarkerType, DclSetMarkerIndex, DclSetMarkerSize によって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (marker)

5.3.3 DclDrawMarkerProjected

1. 機能

   透視座標系でマーカー列を描く.
2. 書式

```
call DclDrawMarkerProjected(x, y, [type], [index], [height])
```

3. 引数

x,y	<R(:)>	マーカーの座標値 (透視座標系).
type	<I>	マーカーのタイプ (文字コード).
index	<I>	マーカーのラインタイプ.
height	<I>	マーカーの高さ.

4. 備考
   - type, index, height を省略したときは, DclSetMarkerType, DclSetMarkerIndex, DclSetMarkerSize によって設定された値が使われる. 初期値はそれぞれ 1, 1, 0.01.
   - ここで設定した type, index, height は, DclSetMarkerType, DclSetMarkerIndex, DclSetMarkerSize によって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (marker)

5.3.4 DclSetMarkerType

1. 機能

   マーカータイプを設定する.
2. 書式

```
call DclSetMarkerType(type)
```

3. 引数

type	<I>	マーカータイプ (文字コード).

4. 備考

なし.

5. 関連項目
   - 関連ルーチン (marker)

5.3.5 DclSetMarkerIndex

1. 機能
   マーカーのラインインデクスを設定する.
2. 書式
   ```
   call DclSetMarkerIndex(index)
   ```
3. 引数
   `index` `<I>` マーカーのラインインデクス.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (marker)

5.3.6 DclSetMarkerSize

1. 機能
   マーカーの大きさを設定する.
2. 書式
   ```
   call DclSetMarkerSize(height)
   ```
3. 引数
   `height` `<R>` マーカーの高さ (文字高).
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (marker)

5.3.7 DclGetMarkerType

1. 機能
   マーカータイプを参照する.
2. 書式
   ```
   result=DclGetMarkerType()
   ```
3. 引数
   戻り値 `<I>` マーカータイプ (文字コード).
4. 備考
   なし.

5. 関連項目
   - 関連ルーチン (marker)

5.3.8 DclGetMarkerIndex

1. 機能

   マーカーのラインインデクスを参照する.
2. 書式

   ```
   result=DclGetMarkerIndex()
   ```
3. 引数

   戻り値 `<I>` マーカーのラインインデクス.
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (marker)

5.3.9 DclGetMarkerSize

1. 機能

   マーカーの大きさを参照する.
2. 書式

   ```
   result=DclGetMarkerSize()
   ```
3. 引数

   戻り値 `<R>` マーカーの大きさ (文字高).
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (marker)

第 6 章

テキスト

6.1 概要

6.1.1 フォント

DCL では現在のところ 2 種類のフォントが使用できる. フォント番号は内部変数 'IFONT' を指定することによって選択できる (初期値は 1). フォントテーブルを参照のこと.

普通, キーボードにある文字はそのまま表示される. キーボードにない文字, あるいはキーボードにあっても正しく表現されない文字は, Fortran90 の関数 ACHAR の引数としてフォントテーブルに示してある番号 (DCL 文字番号) を与えることによって描くことができる.

文字列の有効な長さは関数 LENC を用いて決定される. したがって, 与えた文字列の後方にある NULL 文字あるいは空白文字は無視される.

6.1.2 制御コード

また内部変数 'ENABLE_CONTROL_CHAR' が .TRUE. なら, 制御文字を有効として上付および下付添え字を描くことができる. たとえば, 制御文字の文字番号が初期値から変更されていなければ,

$$(\mathrm{X_i})^2$$

と描くためには

```
'(X_i")|2"'
```

と指定すればよい (いちばん最後の上付および下付添え字のモードの終わりをしめす制御文字は省略できない). 制御文字の指定方法が妥当かどうかは, 文字の長さと高さを求める下位ルーチン (SZQTXW) がチェックする. したがって, 指定方法が妥当でないときエラーメッセージは SZQTXW から出力される.

6.2 ルーチンリスト

6.2.1 描画

DclDrawText	ユーザー座標系で文字列を描く.
(SGTXU,SGTXZU)	
DclDrawTextNormalized	正規座標系で文字列を描く.
(SGTXV,SGTXZV)	
DclDrawTextProjected	透視座標系で文字列を描く.
(SGTXR,SGTXZR)	

6.2.2 設定

DclSetTextHeight	文字の高さを設定する.
(SGSTXS)	
DclSetTextAngle	文字列の角度を設定する.
(SGSTXR)	
DclSetTextIndex	文字列のラインインデクスを設定する.
(SGSTXI)	
DclSetTextPosition	文字列のセンタリングオプションを設定する.
(SGSTXC)	

6.2.3 参照

DclGetTextHeight	文字の高さを参照する.
(SGQTXS)	
DclGetTextAngle	文字列の角度を参照する.
(SGQTXR)	
DclGetTextIndex	文字列のラインインデクスを参照する.
(SGQTXI)	
DclGetTextPosition	文字列のセンタリングオプションを参照する.
(SGQTXC)	

* 括弧の中は, 対応する f77 インターフェイス名.

6.3 各ルーチンの説明

6.3.1 DclDrawText

1. 機能

   ユーザー座標系で文字列を描く.

2. 書式

```
call DclDrawText(x, y, text, [height], [angle], [centering], [index])
```

3. 引数

x,y	<R(:)>	文字を書く位置 (ユーザー座標系).
text	<C*>	テキスト.
height	<R>	文字の高さ.
angle	<R>	角度.
centering	<I>	センタリングオプション.
index	<I>	文字のラインインデクス.

4. 備考
   - height, angle, centering, index を省略したときは, DclSetTextHeight, DclSetTextAngle, DclSetTextPosition, DclSetTextIndex によって設定された値が使われる. 初期値はそれぞれ 0.01, 0., 0, 1.
   - ここで設定した height, angle, centering, index は, DclSetTextHeight, DclSetTextAngle, DclSetTextPosition, DclSetTextIndex によって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (text)

6.3.2 DclDrawTextNormalized

1. 機能

   正規座標系で文字列を描く.
2. 書式

```
call DclDrawTextNormalized(x, y, text, [height], [angle], [centering], [index])
```

3. 引数

x,y	<R(:)>	文字を書く位置 (正規座標系).
text	<C*>	テキスト.
height	<R>	文字の高さ.
angle	<R>	角度.
centering	<I>	センタリングオプション.
index	<I>	文字のラインインデクス.

4. 備考
   - height, angle, centering, index を省略したときは, DclSetTextHeight, DclSetTextAngle, DclSetTextPosition, DclSetTextIndex によって設定された値が使われる. 初期値はそれぞれ 0.01, 0., 0, 1.
   - ここで設定した height, angle, centering, index は, DclSetTextHeight, DclSetTextAngle, DclSetTextPosition, DclSetTextIndex によって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (text)

6.3.3 DclDrawTextProtected

1. 機能

   透視座標系で文字列を描く.

2. 書式

```
call DclDrawTextProtected(x, y, text, [height], [angle], [centering], [index])
```

3. 引数

x,y	<R(:)>	文字を書く位置 (透視座標系).
text	<C*>	テキスト.
height	<R>	文字の高さ.
angle	<R>	角度.
centering	<I>	センタリングオプション.
index	<I>	文字のラインインデクス.

4. 備考

   - height, angle, centering, index を省略したときは, DclSetTextHeight, DclSetTextAngle, DclSetTextPosition, DclSetTextIndex によって設定された値が使われる. 初期値はそれぞれ 0.01, 0., 0, 1.
   - ここで設定した height, angle, centering, index は, DclSetTextHeight, DclSetTextAngle, DclSetTextPosition, DclSetTextIndex によって設定された値を変更しない.

5. 関連項目

   - 関連ルーチン (text)

6.3.4 DclSetTextHeight

1. 機能

   文字の高さを設定する.

2. 書式

```
call DclSetTextHeight(height)
```

3. 引数

height	<R>	文字の高さ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (text)

6.3.5 DclSetTextAngle

1. 機能

   文字列の角度を設定する.

2. 書式

```
call DclSetTextAngle(angle)
```

3. 引数

angle	<R>	文字列の角度.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (text)

6.3.6 DclSetTextIndex

1. 機能

文字列のラインインデクスを設定する.

2. 書式

```
call DclSetTextIndex(index)
```

3. 引数

index	<I>	文字列のラインインデクス.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (text)

6.3.7 DclSetTextPosition

1. 機能

文字列のセンタリングオプションを設定する.

2. 書式

```
call DclSetTextPosition(centering)
```

3. 引数

centering	<I>	文字列のセンタリングオプション.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (text)

6.3.8 DclGetTextHeight

1. 機能

文字の高さを参照する.

2. 書式

```
result=DclGetTextHeight()
```

3. 引数

   戻り値 <R> 文字列の高さ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (text)

6.3.9 DclGetTextAngle

1. 機能

   文字列の角度を参照する.

2. 書式

```
result=DclGetTextAngle()
```

3. 引数

   戻り値 <R> 文字列の角度.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (text)

6.3.10 DclGetTextIndex

1. 機能

   文字列のラインインデクスを参照する.

2. 書式

```
result=DclGetTextIndex()
```

3. 引数

   戻り値 <I> 文字列のラインインデクス.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (text)

6.3.11 DclGetTextPosition

1. 機能

   文字列のセンタリングオプションを参照する.

2. 書式

```
result=DclGetTextPosition()
```

3. 引数

　　戻り値　`<I>`　文字列のセンタリングオプション.

4. 備考

　　なし.

5. 関連項目

　　- 関連ルーチン (text)

第 7 章

トーン

7.1 概要

7.1.1 パターン番号

トーンパターン番号には色の情報と, パターンの情報が両方含まれており, 下位 3 桁がパターン番号, それより上位の桁は色を指定する. 色を指定しない (3 桁の番号だけ指定) 場合は, フォアグラウンド (色番号 1) と解釈される.
パターン番号の最上位桁は, パターンの種類をあらわす.

0: ドット
1: 横線
2: 斜線 (右上がり)
3: 縦線
4: 斜線 (左上がり)
5: 格子 (縦横)
6: 格子 (斜め)

2 桁めは, ドットの大きさや斜線の太さを 0 から 5 の間で指定する. 値が大きくなるにつれてドットは大きくなり斜線は太くなる. 最下位桁はドットや斜線の密度を 0 から 5 の間で指定する. 値が大きくなるにつれてドットや斜線の密度があがる. ただし 0 のときは何も描かれない.

パターン番号として 999 を指定するとべたぬりとなる.

カラーの端末やカラーのプリンターでは, 色を指定したべた塗りによる塗りわけがしばしばおこなわれる. このプログラムをカラーがサポートされない環境で実行しても, それなりな見わけがつくようにドットによる塗りわけとして表現するようなオプションも用意されている.

これらの組合せ以外の整数値はパターンが定義されていない. 実際のパターンについてはトーンパターンテーブル (**??**) を参照のこと.

7.1.2 色番号の読み替え

色を用いたべた塗り (下位 3 桁が 999 であるようなトーンパターン番号) による塗りわけをおこなうとき, 色番号をドットなどのパターン番号として読みかえることができる. 内部変数 'ENABLE_COLOR_SUBSTITUTION' が .TRUE. のときは読みかえる; .FALSE. のときは読みかえない (初期値は .FALSE.). ただし, 色を用いた塗りわけができないような装置では, 'ENABLE_COLOR_SUBSTITUTION' が .FALSE. と設定されていても, パターン番号による読みかえをおこなおうとする.

7.1.3 ソフトフィルとハードフィル

多角形閉領域のぬりつぶし (トーン) を出力する際, 出力装置の能力に応じて, ハードフィルまたはソフトフィルを切替える. ハードフィルでは実際にデバイスに依存した「領域のぬりつぶし」を行なうが, ソフトフィルではそこにドットや線分を描くことによって「ぬりつぶし」を実現する.

内部変数 'ENABLE_SOFTFILL' が .true. のときはソフトフィルをおこなう; .false. のときはハードフィルをおこなう (初期値は .false.). ただし, ハードフィルを指定してもその能力がない場合はソフトフィルとなる.

すでに描かれた図形の上にソフトフィルをおこなってもトーンパターンが上書きされるだけで, それまでに描かれた図形が (べた塗りをしない限り) 消えてしまうことはない. しかしながら, 出力装置によってはハードフィルによるぬりわけによって先に描かれた図形が消えてしまうことがあるので, ほかの図形出力と一緒にトーンを用いるときは 描く順番に注意すること.

7.1.4 頂点座標の指定方法

塗りつぶす多角形を定義する頂点の座標 (PX(i), PY(i)) (i =1,N) に関しては, (PX(1), PY(1)) と (PX(N), PY(N)) は同一な点である必要はない. つまり, (PX(i), PY(i)) と (PX(i+1), PY(i+1)) (i =1,N-1) および (PX(N), PY(N)) と (PX(1), PY(1)) を結んだ閉領域をぬりつぶす.

7.1.5 制約

現在サポートされているサブルーチン群は, 内部的に確保している作業領域の制約から座標値を与える配列の長さ N は 256 以下でなければならない.

7.2 ルーチンリスト

7.2.1 描画

DclDrawShadeRegion (SGTNU,SGTNZU)	ユーザー座標系で多角形領域を塗りつぶす.
DclDrawShadeRegionNormalized (SGTNV,SGTNZV)	正規座標系で多角形領域を塗りつぶす.
DclDrawShadeRegionProjected (SGTNR,SGTNZR)	透視座標系で多角形領域を塗りつぶす.

7.2.2 設定

DclSetShadePattern (SGSTNP)	トーンパターン番号を設定する.

7.2.3 参照

DclGetShadePattern (SGQTNP)	トーンパターン番号を参照する.

* 括弧の中は, 対応する f77 インターフェイス名.

7.3 各ルーチンの説明

7.3.1 DclShadeRegion

1. 機能

   ユーザー座標系で多角形領域を塗りつぶす.

2. 書式

```
call DclShadeRegion(x, y, [pattern])
```

3. 引数

x,y	<R(:)>	多角形の頂点座標 (ユーザー座標系).
pattern	<I>	塗りつぶしのパターン番号.

4. 備考

   - pattern を省略したときは, DclSetShadePattern によって設定された値が使われる. 初期値は 1.
   - ここで設定した pattern は, DclSetShadePattern によって設定された値を変更しない.

5. 関連項目

   - 関連ルーチン (shade)

7.3.2 DclShadeRegionNormalized

1. 機能

   正規座標系で多角形領域を塗りつぶす.

2. 書式

```
call DclShadeRegionNormalized(x, y, [pattern])
```

3. 引数

x,y	<R(:)>	多角形の頂点座標 (正規座標系).
pattern	<I>	塗りつぶしのパターン番号.

4. 備考

   - pattern を省略したときは, DclSetShadePattern によって設定された値が使われる. 初期値は 1.
   - ここで設定した pattern は, DclSetShadePattern によって設定された値を変更しない.

5. 関連項目

   - 関連ルーチン (shade)

7.3.3 DclShadeRegionProjected

1. 機能

   透視座標系で多角形領域を塗りつぶす.

2. 書式

```
call DclShadeRegionProjected(x, y, [pattern])
```

3. 引数

x,y	<R(:)>	多角形の頂点座標 (透視座標系).
pattern	<I>	塗りつぶしのパターン番号.

4. 備考

   - pattern を省略したときは, DclSetShadePattern によって設定された値が使われる. 初期値は 1.
   - ここで設定した pattern は, DclSetShadePattern によって設定された値を変更しない.

5. 関連項目

   - 関連ルーチン (shade)

7.3.4 DclSetShadePattern

1. 機能

   トーンパターン番号を設定する.

2. 書式

```
call DclSetShadePattern(pattern)
```

3. 引数

pattern	<I>	トーンパターン番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (shade)

7.3.5 DclGetShadePattern

1. 機能

   トーンパターン番号を参照する.

2. 書式

```
result=DclGetShadePattern()
```

3. 引数

   戻り値 `<I>` トーンパターン番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (shade)

第 8 章

矢印

8.1 ルーチンリスト

8.1.1 描画

DclDrawArrow (SGLAU,SGLAZU)	ユーザー座標系で矢印付き線分を描く.
DclDrawArrowNormalized (SGLAV,SGLAZV)	正規座標系で矢印付き線分を描く.
DclDrawArrowProjected (SGLAR,SGLAZR)	透視座標系で矢印付き線分を描く.

8.1.2 設定

DclSetArrowLineType (SGSLAT)	矢印付き線分のラインタイプを設定する.
DclSetArrowLineIndex (SGSLAI)	矢印付き線分のラインインデクスの設定する.

8.1.3 参照

DclGetArrowLineType (SGQLAT)	矢印付き線分のラインタイプを参照する.
DclGetArrowLineIndex (SGQLAI)	矢印付き線分のラインインデクスを参照する.

* 括弧の中は, 対応する f77 インターフェイス名.

8.2 各ルーチンの説明

8.2.1 DclDrawArrow

1. 機能

   ユーザー座標系で矢印付き線分を描く.

2. 書式

```
call DclDrawArrow(x1, y1, x2, y2, [type], [index])
```

3. 引数

x1,y1	<R>	始点の座標 (ユーザー座標).
x2,y2	<R>	終点の座標 (ユーザー座標).
type	<I>	線分のラインタイプ.
index	<I>	ラインインデクス.

4. 備考
   - type, index を省略したときは, DclSetArrowLineType, DclSetArrowLineIndex によって設定された値が使われる. 初期値は, それぞれ 1.
   - ここで設定した type, index は, DclSetArrowLineType, DclSetArrowLineIndex によって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (arrow)

8.2.2 DclDrawArrowNormalized

1. 機能

   正規座標系で矢印付き線分を描く.

2. 書式

```
call DclDrawArrowNormalized(x1, y1, x2, y2, [type], [index])
```

3. 引数

x1,y1	<R>	始点の座標 (正規座標).
x2,y2	<R>	終点の座標 (正規座標).
type	<I>	線分のラインタイプ.
index	<I>	ラインインデクス.

4. 備考
   - type, index を省略したときは, DclSetArrowLineType, DclSetArrowLineIndex によって設定された値が使われる. 初期値は, それぞれ 1.
   - ここで設定した type, index は, DclSetArrowLineType, DclSetArrowLineIndex によって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (arrow)

8.2.3 DclDrawArrowProjected

1. 機能

   透視座標系で矢印付き線分を描く.

2. 書式

```
call DclDrawArrowProjected(x1, y1, x2, y2, [type], [index])
```

3. 引数

x1,y1 <R> 始点の座標 (透視座標).
x2,y2 <R> 終点の座標 (透視座標).
type <I> 線分のラインタイプ.
index <I> ラインインデクス.

4. 備考
   - type, index を省略したときは, DclSetArrowLineType, DclSetArrowLineIndex によって設定された値が使われる. 初期値は, それぞれ 1.
   - ここで設定した type, index は, DclSetArrowLineType, DclSetArrowLineIndex によって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (arrow)

8.2.4 DclSetArrowLineType

1. 機能
   矢印付き線分のラインタイプを設定する.
2. 書式
   ```
   call DclSetArrowLineType(type)
   ```
3. 引数
   type <I> ラインタイプ.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (arrow)

8.2.5 DclSetArrowLineIndex

1. 機能
   矢印付き線分のラインインデクスを設定する.
2. 書式
   ```
   call DclSetArrowLineIndex(index)
   ```
3. 引数
   index <I> ラインインデクス.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (arrow)

8.2.6 DclGetArrowLineType

1. 機能

   矢印付き線分のラインタイプを参照する.

2. 書式

   ```
   result=DclGetArrowLineType()
   ```

3. 引数

   戻り値 <I> ラインタイプ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (arrow)

8.2.7 DclGetArrowLineIndex

1. 機能

   矢印付き線分のラインインデクスを参照する.

2. 書式

   ```
   result=DclGetArrowLineIndex()
   ```

3. 引数

   戻り値 <I> ラインインデクス.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (arrow)

第 9 章

エラーバー

9.1 ルーチンリスト

9.1.1 描画

DclDrawXErrorBar　x 方向のエラーバーを描く.
(UHERB,UHERBZ)
DclDrawYErrorBar　y 方向のエラーバーを描く.
(UVERB,UVERBZ)

9.1.2 設定

DclSetErrorBarLineType　エラーバーのラインタイプを設定する.
(UUSEBT)
DclSetErrorBarLineIndex　エラーバーのラインインデクスの設定する.
(UUSEBI)
DclSetErrorBarWidth　エラーバーの横幅を設定する.
(UUSEBS)

9.1.3 参照

DclGetErrorBarLineType　エラーバーのラインタイプを参照する.
(UUQEBT)
DclGetErrorBarLineIndex　エラーバーのラインインデクスの参照する.
(UUQEBI)
DclGetErrorBarWidth　エラーバーの横幅を参照する.
(UUQEBS)

* 括弧の中は, 対応する f77 インターフェイス名.

9.2 各ルーチンの説明

9.2.1 DclDrawXErrorBar

1. 機能

   x 方向のエラーバーを描く.

2. 書式

```
call DclDrawXErrorBar(x1, y1, y, [type], [index], [width])
```

3. 引数

x1,y1	<R(:)>	エラーバーの両端の x 座標値 (ユーザー座標).
y	<R(:)>	エラーバーの y 座標値 (ユーザー座標).
type	<I>	エラーバーのラインタイプ.
index	<I>	エラーバーのラインインデクス.
width	<I>	エラーバーの幅 (正規座標系).

4. 備考

   - type, index, width を省略したときは, DclSetErrorBarLineType, DclSetErrorBarLineIndex, DclSetErrorBarWidth によって設定された値が使われる. 初期値はそれぞれ 1, 1, 0.02.
   - ここで指定した type, index, width は, 上記設定ルーチンによって設定された値を変更しない.

5. 関連項目

   - 関連ルーチン (error)

9.2.2 DclDrawYErrorBar

1. 機能

   y 方向のエラーバーを描く.

2. 書式

```
call DclDrawYErrorBar(x, y1, y2, [type], [index], [width])
```

3. 引数

x	<R(:)>	エラーバーの x 座標値 (ユーザー座標).
y1, y2	<R(:)>	エラーバーの両端の y 座標値 (ユーザー座標).
type	<I>	エラーバーのラインタイプ.
index	<I>	エラーバーのラインインデクス.
width	<I>	エラーバーの幅 (正規座標系).

4. 備考

   - type, index, width を省略したときは, DclSetErrorBarLineType, DclSetErrorBarLineIndex, DclSetErrorBarWidth によって設定された値が使われる. 初期値はそれぞれ 1, 1, 0.02.
   - ここで指定した type, index, width は, 上記設定ルーチンによって設定された値を変更しない.

5. 関連項目

- 関連ルーチン (error)

9.2.3 DclSetErrorBarLineType

1. 機能

   エラーバーのラインタイプを設定する.

2. 書式

```
call DclSetErrorBarLineType(type)
```

3. 引数

   `type` `<I>` エラーバーのラインタイプ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (error)

9.2.4 DclSetErrorBarLineIndex

1. 機能

   エラーバーのラインインデクスを設定する.

2. 書式

```
call DclSetErrorBarLineIndex(index)
```

3. 引数

   `index` `<I>` エラーバーのラインインデクス.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (error)

9.2.5 DclSetErrorBarWidth

1. 機能

   エラーバーの横幅を設定する.

2. 書式

```
call DclSetErrorBarWidth(width)
```

3. 引数

   `width` `<R>` エラーバーの横幅 (正規座標系).

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (error)

9.2.6 DclGetErrorBarLineType

1. 機能

   エラーバーのラインタイプを参照する.

2. 書式

   ```
   result=DclGetErrorBarLineType()
   ```

3. 引数

   戻り値 `<I>` エラーバーのラインタイプ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (error)

9.2.7 DclGetErrorBarLineIndex

1. 機能

   エラーバーのラインインデクスを参照する.

2. 書式

   ```
   result=DclGetErrorBarLineIndex()
   ```

3. 引数

   戻り値 `<I>` エラーバーのラインインデクス.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (error)

9.2.8 DclGetErrorBarWidth

1. 機能

   エラーバーの横幅を参照する.

2. 書式

   ```
   result=DclGetErrorBarWidth()
   ```

3. 引数

   戻り値 `<R>` エラーバーの横幅 (正規座標系).

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (error)

第 10 章

棒グラフ

10.1 ルーチンリスト

10.1.1 描画 (x)

DclDrawXBarFrame (UHBRF,UHBRFZ)	x 方向の棒グラフの枠を描く.
DclShadeXBarArea (UHBRA,UHBRAZ)	x 方向の棒グラフの内部に影をつける.
DclDrawXBarLine (UHBRL,UHBRLZ)	x 方向の棒グラフをつなぐ線を描く.

10.1.2 描画 (y)

DclDrawYBarFrame (UVBRF,UVBRFZ)	y 方向の棒グラフの枠を描く.
DclShadeYBarArea (UVBRA,UVBRAZ)	y 方向の棒グラフの内部に影をつける.
DclDrawYBarLine (UVBRL,UVBRLZ)	y 方向の棒グラフをつなぐ線を描く.

10.1.3 設定

DclSetBarWidth (UUSBRS)	棒グラフの横巾を設定する.
DclSetAreaPattern (UUSARP)	内部領域を塗るトーンパターンを設定する.
DclSetFrameType (UUSFRT)	枠のラインタイプを設定する.
DclSetFrameIndex (UUSFRI)	枠のラインインデクスを設定する.

10.1.4 参照

DclGetBarWidth	棒グラフの横巾を参照する.
(UUQBRS)	
DclGetAreaPattern	内部領域を塗るトーンパターンを参照する.
(UUQARP)	
DclGetFrameType	枠のラインタイプを参照する.
(UUQFRT)	
DclGetFrameIndex	枠のラインインデクスを参照する.
(UUQFRI)	

* 括弧の中は, 対応する f77 インターフェイス名.

10.2 各ルーチンの説明

10.2.1 DclDrawXBarFrame

1. 機能

   x 方向の棒グラフの枠を描く.

2. 書式

```
call DclDrawXBarFrame(x1, x2, y, [type], [index], [width])
```

3. 引数

x1,x2	<R(:)>	棒グラフの両端の x 座標値 (ユーザー座標).
y	<R(:)>	棒グラフの中心の y 座標値 (ユーザー座標).
type	<I>	棒グラフのラインタイプ.
index	<I>	棒グラフのラインインデクス.
width	<I>	棒グラフの幅 (正規座標系).

4. 備考

   - type, index, width を省略したときは, DclSetFrameType, DclSetFrameIndex, DclSetFrameWidth によって設定された値が使われる. 初期値はそれぞれ 1, 1, 0.02.
   - ここで指定した type, index, width は, 上記設定ルーチンによって設定された値を変更しない.

5. 関連項目

   - 関連ルーチン (bar)

10.2.2 DclShadeXBarArea

1. 機能

   x 方向の棒グラフの内部に影をつける.

2. 書式

```
call DclShadeXBarArea(x1, x2, y, [pattern1], [pattern2], [width])
```

3. 引数

x1,x2	<R(:)>	棒グラフの両端の x 座標値 (ユーザー座標).
y	<R(:)>	棒グラフの中心の y 座標値 (ユーザー座標).
pattern1	<I>	x1 < x2 の時のパターン番号.
pattern2	<I>	x1 > x2 の時のパターン番号.
width	<R>	棒グラフの幅.

4. 備考
   - pattern1, pattern2, width を省略したときは, DclSetAreaPattern, DclSetFrameWidth によって設定された値が使われる. 初期値はそれぞれ 1, 1, 0.02.
   - ここで指定した pattern1, pattern2, width は, 上記設定ルーチンによって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (bar)

10.2.3 DclDrawXBarLine

1. 機能

   x 方向の棒グラフをつなぐ線を描く.
2. 書式

```
call DclDrawXBarLine(x, y, [type], [index], [width])
```

3. 引数

x	<R(:)>	棒グラフの端の x 座標値 (ユーザー座標).
y	<R(:)>	棒グラフの中心の y 座標値 (ユーザー座標).
type	<I>	ラインタイプ.
index	<I>	ラインインデクス.
width	<R>	棒グラフの幅 (正規座標系).

4. 備考
   - type, index, width を省略したときは, DclSetFrameType, DclSetFrameIndex, DclSetBarWidth によって設定された値が使われる. 初期値はそれぞれ 1, 1, 0.02.
   - ここで設定した type, index, width は, 上記設定ルーチンによって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (bar)

10.2.4 DclDrawYBarFrame

1. 機能

   y 方向の棒グラフの枠を描く.
2. 書式

```
call DclDrawYBarFrame(x1, y1, y2, [type], [index], [width])
```

3. 引数

x	<R(:)>	棒グラフの中心の x 座標値 (ユーザー座標).
y1,y2	<R(:)>	棒グラフの両端の y 座標値 (ユーザー座標).
type	<I>	棒グラフのラインタイプ.
index	<I>	棒グラフのラインインデクス.
width	<I>	棒グラフの幅 (正規座標系).

4. 備考
   - type, index, width を省略したときは, DclSetFrameType, DclSetFrameIndex, DclSetFrameWidth によって設定された値が使われる. 初期値はそれぞれ 1, 1, 0.02.
   - ここで指定した type, index, width は, 上記設定ルーチンによって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (bar)

10.2.5 DclShadeYBarArea

1. 機能

   y 方向の棒グラフの内部に影をつける.
2. 書式

```
call DclShadeYBarArea(x, y1, y2, [pattern1], [pattern2], [width])
```

3. 引数

x	<R(:)>	棒グラフの中心の x 座標値 (ユーザー座標).
y1,y2	<R(:)>	棒グラフの両端の y 座標値 (ユーザー座標).
pattern1	<I>	y1 < y2 の時のパターン番号.
pattern2	<I>	y1 > y2 の時のパターン番号.
width	<R>	棒グラフの幅.

4. 備考
   - pattern1, pattern2, width を省略したときは, DclSetAreaPattern, DclSetFrameWidth によって設定された値が使われる. 初期値はそれぞれ 1, 1, 0.02.
   - ここで指定した pattern1, pattern2, width は, 上記設定ルーチンによって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (bar)

10.2.6 DclDrawYBarLine

1. 機能

   y 方向の棒グラフをつなぐ線を描く.
2. 書式

```
call DclDrawYBarLine(x, y, [type], [index], [width])
```

3. 引数

x	<R(:)>	棒グラフの中心の x 座標値 (ユーザー座標).
y	<R(:)>	棒グラフの端の y 座標値 (ユーザー座標).
type	<I>	棒グラフの枠のラインタイプ.
index	<I>	棒グラフの枠のラインインデクス.
width	<R>	棒グラフの幅 (正規座標系).

4. 備考
   - type, index, width を省略したときは, DclSetFrameType, DclSetFrameIndex, DclSetBarWidth によって設定された値が使われる. 初期値はそれぞれ 1, 1, 0.02.
   - ここで設定した type, index, width は, 上記設定ルーチンによって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (bar)

10.2.7 DclSetBarWidth

1. 機能
   棒グラフの横幅を設定する.
2. 書式
   ```
   call DclSetBarWidth(width)
   ```
3. 引数

width	<R>	棒グラフの横幅.

4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (bar)

10.2.8 DclSetAreaPattern

1. 機能
   内部領域を塗るトーンパターンを設定する.
2. 書式
   ```
   call DclSetAreaPattern(pattern1, pattern2)
   ```
3. 引数

pattern1	<I>	x1 < x2 または y1 < y2 の時のパターン番号.
pattern2	<I>	x1 > x2 または y1 > y2 の時のパターン番号.

4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (bar)

10.2.9 DclSetFrameType

1. 機能

   エラーバーのラインタイプを設定する.

2. 書式

   ```
   call DclSetFrameType(type)
   ```

3. 引数

type	<I>	枠のラインタイプ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (bar)

10.2.10 DclSetFrameIndex

1. 機能

   枠のラインインデクスを設定する.

2. 書式

   ```
   call DclSetFrameIndex(index)
   ```

3. 引数

index	<I>	枠のラインインデクス.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (bar)

10.2.11 DclGetBarWidth

1. 機能

   棒グラフの横幅を参照する.

2. 書式

   ```
   result=DclGetBarWidth()
   ```

3. 引数

戻り値	<R>	棒グラフの横幅.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (bar)

10.2.12 DclGetAreaPattern

1. 機能

   内部領域を塗るトーンパターンを参照する.

2. 書式

   ```
   call DclGetAreaPattern(pattern1, pattern2)
   ```

3. 引数

   pattern1 <I> x1 < x2 または y1 < y2 の時のパターン番号.

   pattern2 <I> x1 > x2 または y1 > y2 の時のパターン番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (bar)

10.2.13 DclGetFrameType

1. 機能

   エラーバーのラインタイプを参照する.

2. 書式

   ```
   result=DclGetFrameType()
   ```

3. 引数

   戻り値 <I> 枠のラインタイプ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (bar)

10.2.14 DclGetFrameIndex

1. 機能

   枠のラインインデクスを参照する.

2. 書式

   ```
   result=DclGetFrameIndex()
   ```

3. 引数

   戻り値 <I> 枠のラインインデクス.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (bar)

第 11 章

箱グラフ

11.1 ルーチンリスト

11.1.1 描画 (x)

DclDrawXBoxFrame　x 方向の箱グラフの枠を描く.
(UHBXF,UHBXFZ)
DclShadeXBoxArea　x 方向の箱グラフの内部に影をつける.
(UHBXA,UHBXAZ)
DclDrawXBoxLine　x 方向の箱グラフをつなぐ線を描く.
(UHBXL,UHBXLZ)

11.1.2 描画 (y)

DclDrawYBoxFrame　y 方向の箱グラフの枠を描く.
(UVBXF,UVBXFZ)
DclShadeYBoxArea　y 方向の箱グラフの内部に影をつける.
(UVBXA,UVBXAZ)
DclDrawYBoxLine　y 方向の箱グラフをつなぐ線を描く.
(UVBXL,UVBXLZ)

11.1.3 設定

DclSetAreaPattern　内部領域を塗るトーンパターンを設定する.
(UUSARP)
DclSetFrameType　枠のラインタイプを設定する.
(UUSFRT)
DclSetFrameIndex　枠のラインインデクスを設定する.
(UUSFRI)

11.1.4 参照

DclGetAreaPattern (UUQARP)	内部領域を塗るトーンパターンを参照する.
DclGetFrameType (UUQFRT)	枠のラインタイプを参照する.
DclGetFrameIndex (UUQFRI)	枠のラインインデクスを参照する.

* 括弧の中は, 対応する f77 インターフェイス名.

11.2 各ルーチンの説明

11.2.1 DclDrawXBoxFrame

1. 機能

   x 方向の箱グラフの枠を描く.

2. 書式

```
call DclDrawXBoxFrame(x1, x2, y, [type], [index])
```

3. 引数

x1,x2	<R(:)>	箱グラフの両端の x 座標値 (ユーザー座標).
y	<R(:)>	箱グラフの中心の y 座標値 (ユーザー座標).
type	<I>	箱グラフの枠のラインタイプ.
index	<I>	箱グラフの枠のラインインデクス.

4. 備考

   - type, index を省略したときは, DclSetFrameType, DclSetFrameIndex によって設定された値が使われる. 初期値はそれぞれ 1, 1.
   - ここで指定した type, index は, 上記設定ルーチンによって設定された値を変更しない.

5. 関連項目

   - 関連ルーチン (box)

11.2.2 DclShadeXBoxArea

1. 機能

   x 方向の箱グラフの内部に影をつける.

2. 書式

```
call DclShadeXBoxArea(x1, x2, y, [pattern1], [pattern2])
```

3. 引数

x1,x2	<R(:)>	箱グラフの両端の x 座標値 (ユーザー座標).
y	<R(:)>	箱グラフの中心の y 座標値 (ユーザー座標).
pattern1	<I>	x1 < x2 の時のパターン番号.
pattern2	<I>	x1 > x2 の時のパターン番号.

4. 備考
   - pattern1, pattern2, width を省略したときは, DclSetAreaPattern によって設定された値が使われる. 初期値はそれぞれ 1, 1.
   - ここで指定した pattern1, pattern2 は, 上記設定ルーチンによって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (box)

11.2.3 DclDrawXBoxLine

1. 機能

   x 方向の箱グラフをつなぐ線を描く.
2. 書式

```
call DclDrawXBoxLine(x, y, [type], [index])
```

3. 引数

x	<R(:)>	箱グラフの端の x 座標値 (ユーザー座標).
y	<R(:)>	箱グラフの中心の y 座標値 (ユーザー座標).
type	<I>	ラインタイプ.
index	<I>	ラインインデクス.

4. 備考
   - type, index を省略したときは, DclSetFrameType, DclSetFrameIndex によって設定された値が使われる. 初期値はそれぞれ 1, 1.
   - ここで設定した type, index は, 上記設定ルーチンによって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (box)

11.2.4 DclDrawYBoxFrame

1. 機能

   y 方向の箱グラフの枠を描く.
2. 書式

```
call DclDrawYBoxFrame(x1, y1, y2, [type], [index])
```

3. 引数

x	<R(:)>	箱グラフの中心の x 座標値 (ユーザー座標).
y1,y2	<R(:)>	箱グラフの両端の y 座標値 (ユーザー座標).
type	<I>	箱グラフのラインタイプ.
index	<I>	箱グラフのラインインデクス.

4. 備考
   - type, index を省略したときは, DclSetFrameType, DclSetFrameIndex によって設定された値が使われる. 初期値はそれぞれ 1, 1.
   - ここで指定した type, index は, 上記設定ルーチンによって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (box)

11.2.5 DclShadeYBoxArea

1. 機能

   y 方向の箱グラフの内部に影をつける.
2. 書式

```
call DclShadeYBoxArea(x, y1, y2, [pattern1], [pattern2])
```

3. 引数

x	<R(:)>	箱グラフの中心の x 座標値 (ユーザー座標).
y1,y2	<R(:)>	箱グラフの両端の y 座標値 (ユーザー座標).
pattern1	<I>	y1 < y2 の時のパターン番号.
pattern2	<I>	y1 > y2 の時のパターン番号.

4. 備考
   - pattern1, pattern2 を省略したときは, DclSetAreaPattern によって設定された値が使われる. 初期値はそれぞれ 1, 1.
   - ここで指定した pattern1, pattern2 は, 上記設定ルーチンによって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (box)

11.2.6 DclDrawYBoxLine

1. 機能

   y 方向の箱グラフをつなぐ線を描く.
2. 書式

```
call DclDrawYBoxLine(x, y, [type], [index])
```

3. 引数

x	<R(:)>	箱グラフの中心の x 座標値 (ユーザー座標).
y	<R(:)>	箱グラフの端の y 座標値 (ユーザー座標).
type	<I>	箱グラフの枠のラインタイプ.
index	<I>	箱グラフの枠のラインインデクス.

4. 備考
   - type, index を省略したときは, DclSetFrameType, DclSetFrameIndex によって設定された値が使われる. 初期値はそれぞれ 1, 1.
   - ここで設定した type, index は, 上記設定ルーチンによって設定された値を変更しない.

5. 関連項目
   - 関連ルーチン (box)

11.2.7 DclSetAreaPattern

1. 機能
   内部領域を塗るトーンパターンを設定する.
2. 書式
   ```
   call DclSetAreaPattern(pattern1, pattern2)
   ```
3. 引数
   `pattern1` `<I>` x1 < x2 または y1 < y2 の時のパターン番号.
   `pattern2` `<I>` x1 > x2 または y1 > y2 の時のパターン番号.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (bar)

11.2.8 DclSetFrameType

1. 機能
   エラーバーのラインタイプを設定する.
2. 書式
   ```
   call DclSetFrameType(type)
   ```
3. 引数
   `type` `<I>` 枠のラインタイプ.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (bar)

11.2.9 DclSetFrameIndex

1. 機能
   枠のラインインデクスを設定する.
2. 書式
   ```
   call DclSetFrameIndex(index)
   ```
3. 引数
   `index` `<I>` 枠のラインインデクス.
4. 備考
   なし.

5. 関連項目
   - 関連ルーチン (bar)

11.2.10 DclGetAreaPattern

1. 機能

   内部領域を塗るトーンパターンを参照する.
2. 書式

   ```
   call DclGetAreaPattern(pattern1, pattern2)
   ```
3. 引数

   pattern1 <I> x1 < x2 または y1 < y2 の時のパターン番号.

   pattern2 <I> x1 > x2 または y1 > y2 の時のパターン番号.
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (bar)

11.2.11 DclGetFrameType

1. 機能

   エラーバーのラインタイプを参照する.
2. 書式

   ```
   result=DclGetFrameType()
   ```
3. 引数

   戻り値 <I> 枠のラインタイプ.
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (bar)

11.2.12 DclGetFrameIndex

1. 機能

   枠のラインインデクスを参照する.
2. 書式

   ```
   result=DclGetFrameIndex()
   ```
3. 引数

   戻り値 <I> 枠のラインインデクス.
4. 備考

   なし.

5. 関連項目
    - 関連ルーチン (bar)

第 12 章

差分

12.1 ルーチンリスト

12.1.1 描画

DclShadeXGap　2 本の折れ線の x 方向の差分を塗りつぶす.
(UHDIF,UHDIFZ)
DclShadeYGap　2 本の折れ線の y 方向の差分を塗りつぶす.
(UVDIF,UVDIFZ)

12.1.2 設定

DclSetAreaPattern　内部領域を塗るトーンパターンを設定する.
(UUSARP)

12.1.3 参照

DclGetAreaPattern　内部領域を塗るトーンパターンを参照する.
(UUQARP)

* 括弧の中は, 対応する f77 インターフェイス名.

12.2 各ルーチンの説明

12.2.1 DclShadeXGap

1. 機能
   2 本の折線の x 方向の差分を塗りつぶす.
2. 書式
   ```
   call DclShadeXGap(x1, x2, y, [pattern1], [pattern2])
   ```
3. 引数

x1,x2	<R(:)>	2 本の折線の x 座標値 (ユーザー座標).
y	<R(:)>	折線の y 座標値 (ユーザー座標).
pattern1	<I>	x1 < x2 の時のパターン番号.
pattern2	<I>	x1 > x2 の時のパターン番号.

4. 備考
   - pattern1, pattern2 を省略したときは, DclSetAreaPattern によって設定された値が使われる. 初期値はそれぞれ 1, 1.
   - ここで指定した pattern1, pattern2 は, 上記設定ルーチンによって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (gap)

12.2.2 DclShadeYGap

1. 機能
   2 本の折線の y 方向の差分を塗りつぶす.
2. 書式
```
call DclShadeYGap(x, y1, y2, [pattern1], [pattern2])
```
3. 引数

x	<R(:)>	折線の x 座標値 (ユーザー座標).
y1,y2	<R(:)>	2 本の折線の y 座標値 (ユーザー座標).
pattern1	<I>	y1 < y2 の時のパターン番号.
pattern2	<I>	y1 > y2 の時のパターン番号.

4. 備考
   - pattern1, pattern2 を省略したときは, DclSetAreaPattern によって設定された値が使われる. 初期値はそれぞれ 1, 1.
   - ここで指定した pattern1, pattern2 は, 上記設定ルーチンによって設定された値を変更しない.
5. 関連項目
   - 関連ルーチン (gap)

12.2.3 DclSetAreaPattern

1. 機能
   内部領域を塗るトーンパターンを設定する.
2. 書式
```
call DclSetAreaPattern(pattern1, pattern2)
```
3. 引数

pattern1	<I>	x1 < x2 または y1 < y2 の時のパターン番号.
pattern2	<I>	x1 > x2 または y1 > y2 の時のパターン番号.

4. 備考

なし.

5. 関連項目
   - 関連ルーチン (bar)

12.2.4 DclGetAreaPattern

1. 機能

   内部領域を塗るトーンパターンを参照する.

2. 書式

   ```
   call DclGetAreaPattern(pattern1, pattern2)
   ```

3. 引数

   `pattern1` `<I>` x1 < x2 または y1 < y2 の時のパターン番号.

   `pattern2` `<I>` x1 > x2 または y1 > y2 の時のパターン番号.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (bar)

第 13 章

コンター

13.1 ルーチンリスト

13.1.1 描画

DclDrawContour (UDCNTR,UDCNTZ)	2 次元等高線図を描く.

13.1.2 設定

DclSetContourLevel (UDGCLA,UDGCLB)	コンターレベル値を設定する.
DclSetContourLine (UDSCLV)	1 本のコンターレベルを設定する.
DclDelContourLevel (UDDCLV)	1 本のコンターレベルを削除する.
DclClearContourLevel (UDICLV)	コンターレベルを無効にする.
DclSetContourLabelFormat (UDSFMT)	コンターラベルのフォーマットを設定する.

13.1.3 参照

DclGetContourLine (UDQCLV)	コンターレベルの情報を参照する.
DclGetContourLevelNumber (UDQCLN)	現在設定されているコンターレベルの総本数を参照する.
DclClearCounourInterval (RUDLEV)	コンターレベルの間隔を求める.
DclGetContourLabelFormat (UDQFMT)	コンターラベルのフォーマットを参照する.

* 括弧の中は, 対応する f77 インターフェイス名.

13.2 各ルーチンの説明

13.2.1 DclDrawContour

1. 機能

   2 次元等高線図を描く.

2. 書式

   ```
   call DclDrawContour(z)
   ```

3. 引数

z	<R(:,:)>	等高線データ (2 次元).

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (contour)

13.2.2 DclSetContourLevel

1. 機能

   コンターレベルを設定する.

2. 書式

   ```
   call DclSetContourLevel(xmin, xmax, dx)
   call DclSetContourLevel(z,dx)
   ```

3. 引数

xmin, xmax	<R>	コンターレベルの最小値と最大値.
		必ずしもきりのよい値でなくてもよい.
z	<R(:,:)>	等高線データ.
dx	<R>	DX >0 のとき DX をきざみ幅とする.
		DX $=0$ のとき内部変数 'NLEV' を参照して,
		約 NLEV 本の コンターレベルを生成する.
		DX <0 のとき約 INT(ABS(DX)) 本の コンターレベルを生成する.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (contour)

13.2.3 DclSetContourLine

1. 機能

   1 本のコンターレベルを設定する.

2. 書式

```
call DclSetContourLine(level, [index], [type], [label], [height])
```

3. 引数

level	<R>	コンターレベルの値.
index	<I>	コンターのラインインデクス.
type	<I>	コンターのラインタイプ.
label	<C*>	コンターにつけるラベル (8 文字以内).
height	<R>	ラベルの高さ (正規座標系).

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (contour)

13.2.4 DclDelContourLevel

1. 機能

   1 本のコンターレベルを削除する.

2. 書式

```
call DclDelContourLevel(level)
```

3. 引数

level	<R>	削除するコンターのレベル値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (contour)

13.2.5 DclClearContourLevel

1. 機能

   コンターレベルを無効にする.

2. 書式

```
call DclClearContourLevel()
```

3. 引数なし.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (contour)

13.2.6 DclSetContourLabelFormat

1. 機能

コンターラベルのフォーマットを設定する.

2. 書式

```
call DclSetContourLabelFormat(format)
```

3. 引数

format	<C*>	指定するフォーマット (16 文字以下).

4. 備考

なし.

5. 関連項目

- 関連ルーチン (contour)

13.2.7 DclGetContourLine

1. 機能

コンターレベルの情報を参照する.

2. 書式

```
call DclGetContourLine(number, [level], [index], [type], [label], [height])
```

3. 引数

number	<I>	コンターラインの番号 (指定順).
level	<R>	コンターラインのレベル値.
index	<I>	コンターラインのラインインデクス.
type	<I>	コンターラインのラインタイプ.
label	<C*>	コンターラインのラベル.
height	<R>	ラベルの高さ.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (contour)

13.2.8 DclGetContourLevelNumber

1. 機能

現在設定されているコンターレベルの総本数を参照する.

2. 書式

```
result=DclGetContourLevelNumber()
```

3. 引数

戻り値	<I>	設定されているコンターレベルの数.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (contour)

13.2.9 DclGetContourInterval

1. 機能

   コンターレベルの間隔を参照する.

2. 書式

```
result=DclGetContourInterval(nlev)
```

3. 引数

   戻り値 <R> コンターレベルの間隔.

   nlev <I> コンターレベルの順番.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (contour)

13.2.10 DclGetContourLabelFormat

1. 機能

   コンターラベルのフォーマットを参照する.

2. 書式

```
call DclGetContourLabelFormat(format)
```

3. 引数

   format <C*> コンターラベルのフォーマット.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (contour)

第 14 章

ぬりわけ

14.1 ルーチンリスト

14.1.1 描画

DclShadeContour	2 次元等値線図をパターンでぬりわける.
(UETONE)	
DclShadeContourEx	image 機能を使って 2 次元等値線図を色でぬりわける.
(UETONF)	

14.1.2 設定

DclSetShadeLevel	ぬりわけレベルを設定する.
(UEGTLA,UEGTLB,UESTLV,UESTLN)	
DclClearShadeLevel	ぬりわけレベルを無効にする.
(UEITLV)	

14.1.3 参照

DclGetShadeLevel	ぬりわけレベルの情報を参照する.
(UEQTLV)	
DclGetShadeLevelNumber	設定されているぬりわけレベルの数を参照する.
(UEQNTL)	

* 括弧の中は, 対応する f77 インターフェイス名.

14.2 各ルーチンの説明

14.2.1 DclShadeContour

1. 機能

   2 次元等値線図をパターンでぬりわける.

2. 書式

```
call DclShadeContour(z)
```

3. 引数

   z <R(:,:)> ぬり分けたいデータ (2 次元).

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (cont_s)

14.2.2 DclShadeContourEx

1. 機能

   image 機能を使って 2 次元等値線図を色でぬりわける.

2. 書式

```
call DclShadeContourEx(z)
```

3. 引数

   z <R(:,:)> ぬり分けたいデータ (2 次元).

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (cont_s)

14.2.3 DclSetShadeLevel

1. 機能

   ぬりわけレベルを設定する.

2. 書式

```
call DclSetShadeLevel(xmin, xmax, dx)
call DclSetShadeLevel(z,dx)
call DclSetShadeLevel(level1, level2, pattern)
call DclSetShadeLevel(level, pattern)
```

3. 引数

xmin, xmax	<R>	ぬり分けレベルの最小最大値.
z	<R(:,:)>	ぬり分けたいデータ.
dx	<R>	dx >0 のとき dx をきざみ幅とする.
		dx $=0$ のとき内部変数 'NLEV' を参照して,
		約 NLEV 本の トーンレベルを生成する.
		dx <0 のとき約 int(abs(dx)) 本の トーンレベルを生成する.
level	<R(:)>	レベルの配列. 大きさは size(pattern) +1.
pattern	<I(:)>	パターンを指定する配列.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (cont_s)

14.2.4 DclClearShadeLevel

1. 機能
   ぬりわけレベルを無効にする.
2. 書式
   ```
   call DclClearShadeLevel()
   ```
3. 引数なし.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (cont_s)

14.2.5 DclGetShadeLevel

1. 機能
   ぬりわけレベルの情報を参照する.
2. 書式
   ```
   call DclGetShadeLevel(number, level1, level2, pattern)
   ```
3. 引数

number	<I>	ぬり分けレベルの順番.
level1, level2	<R>	ぬり分ける範囲.
pattern	<I>	パターン番号.

4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (cont_s)

14.2.6 DclGetShadeLevelNumber

1. 機能
   設定されているぬりわけレベルの数を参照する.
2. 書式
   ```
   result=DclGetShadeLevelNumber()
   ```
3. 引数

戻り値	<I>	設定されているぬり分けレベルの数.

4. 備考
   なし.

5. 関連項目
    - 関連ルーチン (cont_s)

第 15 章

ベクトル場

15.1 ルーチンリスト

15.1.1 描画

DclDrawVectors　2 次元ベクトル場を描く.
(UGVECT)

15.1.2 設定

DclSetUnitVectorTitle　ユニットベクトルのタイトルを設定する.
(UGSUT)

* 括弧の中は, 対応する f77 インターフェイス名.

15.2 各ルーチンの説明

15.2.1 DclDrawVectors

1. 機能
   2 次元ベクトル図を描く.
2. 書式
   ```
   call DclDrawVectors(u, v)
   ```
3. 引数
   u　<R(:,:)>　ベクトルの x 成分.
   v　<R(:,:)>　ベクトルの y 成分.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (vector)

15.2.2 DclSetUnitVectorTitle

1. 機能

   ユニットベクトルのタイトルを設定する.

2. 書式

   ```
   call DclSetUnitVectorTitle(side, title)
   ```

3. 引数

side	<C1>	タイトルをつける位置を指定する. 'X' (X 方向のベクトル), 'Y' (Y 方向のベクトル) が指定できる.
title	<C*>	つけるタイトル (32 文字以内).

4. 備考

   - 設定できるタイトルの数は最大 10 個, またタイトルの長さは最大 32 文字である.

5. 関連項目

   - 関連ルーチン (vector)

第 16 章

地図投影

16.1 ルーチンリスト

16.1.1 描画 (x)

DclDrawGlobe (UMPGLB)	地図の境界線 (縁) と緯度線, 経度線を描く.
DclDrawGrid (UMPGRD)	緯度線・経度線を描く.
DclDrawLimb (UMPLIM)	地図の境界線 (縁) を描く.
DclDrawMap (UMPMAP)	各種地図情報を描く.

16.1.2 設定

DclSetMapContactPoint (UMSCNT)	投影面の「接点」を設定する.
DclSetCircleWindow (UMSCWD)	円形のウィンドウを設定する.
DclSetMapPoint (UMSPNT)	地図に含める点を設定する.

* 括弧の中は, 対応する f77 インターフェイス名.

16.2 各ルーチンの説明

16.2.1 DclFitMapParm

1. 機能

   地図投影の変換関数のパラメタを適切に決める.

2. 書式

```
call DclFitMapParm()
```

3. 引数なし.
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (map)

16.2.2 DclSetMapContactPoint

1. 機能

   投影面の「接点」を設定する.
2. 書式

```
call DclSetMapContactPoint([lon], [lat], [rot])
```

3. 引数

lon, lat	<R>	接点の緯度・経度.
rot	<R>	接点における経線と、地図投影の中央経線がなす角度.

4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (map)

16.2.3 DclSetCircleWindow

1. 機能

   円形のウィンドウを設定する.
2. 書式

```
call DclSetCircleWindow([lon], [lat], [r])
```

3. 引数

lon, lat	<R>	円形ウィンドウの緯度・経度.
r	<R>	ウィンドウの半径.

4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (map)

16.2.4 DclSetMapPoint

1. 機能

   地図に含める点を設定する.
2. 書式

```
call DclSetMapPoint(lon, lat)
```

3. 引数

`lon, lat <R>` 点の緯度・経度.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (map)

16.2.5 DclDrawGlobe

1. 機能

   地図の境界線（縁）と緯度線, 経度線を描く.

2. 書式

   ```
   call DclDrawGlobe()
   ```

3. 引数なし.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (map)

16.2.6 DclDrawGrid

1. 機能

   緯度線・経度線を描く.

2. 書式

   ```
   call DclDrawGrid()
   ```

3. 引数なし.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (map)

16.2.7 DclDrawLimb

1. 機能

   地図の境界線（縁）を描く.

2. 書式

   ```
   call DclDrawLimb()
   ```

3. 引数なし.

4. 備考

   なし.

5. 関連項目

- 関連ルーチン (map)

16.2.8 DclDrawMap

1. 機能

   各種地図情報を描く.

2. 書式

   ```
   call DclDrawMap(file_name)
   ```

3. 引数

file_name	<C*>	ファイル名.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (map)

第 17 章

座標軸・グラフ

17.1 ルーチンリスト

17.1.1 グラフ

DclDrawScaledGraph	おまかせグラフを描く.
(USGRPH)	

17.1.2 スケーリング

DclFitScalingParm	すでに設定された情報から,
(USPFIT)	適当なスケーリングで正規化変換パラメータを決める.

17.1.3 座標軸

DclDrawScaledAxis	おまかせ座標軸を描く.
(USDAXS,USXAXS,USYAXS)	
DclDrawAxis	目盛間隔, ラベル間隔を指定して
(UXAXDV,UYAXDV)	座標軸を描く.
DclDrawAxisSpecify	目盛とラベルを指定して座標軸を描く.
(UXAXNM,UYAXNM,UXAXLB,UYAXLB)	
DclDrawAxisLog	対数座標軸を描く.
(ULXLOG,ULYLOG)	
DclDrawAxisCalendar	日付座標軸を描く.
(UCXACL,UCYCL,UCXADY,UCYADY,	
UCXANM,UCYANM,UCXAYR,UCYAYR)	

17.1.4 座標軸要素

DclDrawTitle (UXMTTL,UYMTTL, UXSTTL,UYSTTL)	タイトルを描く.
DclDrawAxisLine (UXPAXS,UYPAXS)	座標軸の線を描く.
DclDrawTickmark (UXPTMK,UYPTMK)	目盛を描く.
DclDrawAxisLabel (UXPLBL,UYPLBL, UXPNUM,UYPNUM)	ラベルを描く.
DclShiftAxis (UXSAXS,UYSAXS)	座標軸を外側にシフトする.

17.1.5 設定

DclScalingPoint (USSPNT)	グラフの中に含めたい座標点を設定する.
DclSetTitle (USSTTL)	座標軸のタイトルを設定する.
DclSetAxisFactor (UZFACT)	文字, 目盛等の大きさを変える.

* 括弧の中は, 対応する f77 インターフェイス名.

17.2 各ルーチンの説明

17.2.1 DclDrawScaledGraph

1. 機能

   おまかせグラフを描く.

2. 書式

   ```
   call DclDrawScaledGraph(x, y, [type], [index])
   ```

3. 引数

x,y	<R(:)>	折線の座標値 (ユーザー座標系).
type	<I>	折線グラフのラインタイプ.
index	<I>	折線グラフのラインインデクス.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (graph)

17.2.2 DclFitScalingParm

1. 機能

   地図投影の変換関数のパラメタを適切に決める.

2. 書式

   ```
   call DclFitScalingParm()
   ```

3. 引数なし.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (graph)

17.2.3 DclDrawScaledAxis

1. 機能

   おまかせ座標軸を描く.

2. 書式

   ```
   call DclDrawScaledAxis([side], [section])
   ```

3. 引数

side	<C*>	描く座標軸の位置. 省略値は'tbrl'.
section	<R>	side='h' または 'v' の時の切片.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (graph)

17.2.4 DclDrawAxis

1. 機能

   目盛間隔, ラベル間隔を指定して座標軸を描く.

2. 書式

   ```
   call DclDrawAxis(side, dlabel, dtick, [title], [unit], [factor], [offset],
   [section])
   ```

3. 引数

side	<C*>	座標軸の位置.
dlabel	<R>	ラベルの位置.
dtick	<R>	目盛の間隔.
title	<C*>	タイトル. 省略値はブランク.
unit	<C*>	単位. 省略値はブランク.
factor	<R>	座標軸の係数. 省略値は 1.
offset	<R>	座標軸のオフセット. 省略値は 0.
section	<R>	side='h' または 'v' の時の切片.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (graph)

17.2.5 DclDrawAxisSpecify

1. 機能

   目盛とラベルを指定して座標軸を描く.

2. 書式

```
call DclDrawAxisSpecify(side, label_pos, [tick_pos], [label], [title], [unit],
[factor], [offset], [section])
```

3. 引数

side	<C*>	座標軸の位置.
label_pos	<R(:)>	ラベルの位置.
tick_pos	<R(:)>	目盛の位置.
label	<C*(:)>	ラベルの文字列.
title	<C*>	タイトル. 省略時はブランク.
unit	<C*>	単位. 省略時はブランク.
factor	<R>	座標軸の係数. 省略時は 1.
offset	<R>	座標軸のオフセット. 省略時は 0.
section	<R>	side='h' または 'v' の時の切片. 省略時は 0.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (graph)

17.2.6 DclDrawAxisLog

1. 機能

   対数座標軸を描く.

2. 書式

```
call DclDrawAxisLog(side, nlabel, nticks, [title], [unit], [format], [factor],
[section], [label_pos])
```

3. 引数

side	<C*>	座標軸の位置.
nlabel	<I>	1 桁の範囲に描くラベルの数. 1 から 4 までの整数値を指定する.
		1: 1×10^n のみにラベルを描く
		2: 1×10^n, 2×10^n にラベルを描く
		3: 1×10^n, 2×10^n, 5×10^n にラベルを描く
		4: ユーザーが指定した場所にラベルを描く
nticks	<I>	1 桁の範囲に描く目盛の数. 1 から 9 の範囲で指定する.
		9 より小さな値を指定した場合には, 目盛間隔の狭いところから省かれる.
title	<C*>	タイトル. 省略値はブランク.
unit	<C*>	単位. 省略値はブランク.
format	<C*>	ラベル位置の座標値を文字化するフォーマット.
factor	<R>	座標軸の係数. 省略値は 1.
section	<R>	side='h' または 'v' の時の切片.
label_pos	<R(:)>	1 桁の範囲に描くラベルの位置. 各要素の値は 1 以上で 10 より小さくなければならない.
		また label_pos(1) は 1 でなければならない.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (graph)

17.2.7 DclDrawAxisCalendar

1. 機能

日付座標軸を描く.

2. 書式

```
call DclDrawAxisCalendar(side, first_day, [title], [unit], [type], [nd])
```

3. 引数

side	<C*>	座標軸の位置.
first_day	<dcl_date>	座標軸の始点の日付.
title	<C*>	タイトル. 省略値はブランク.
unit	<C*>	単位. 省略値はブランク.
type	<C*>	座標軸のタイプ. 省略値は'DMY'.
nd	<I>	日付座標軸の長さ (日数).

4. 備考

なし.

5. 関連項目
   - 関連ルーチン (graph)

17.2.8 DclDrawTitle

1. 機能
   タイトルを描く.
2. 書式
   ```
   call DclDrawTitle(side, title, [position], [sw])
   ```
3. 引数

side	<C*>	座標軸の位置.
title	<C*>	タイトル.
position	<R>	-1 から +1 の間の実数値で タイトルを描く位置を指定する. -1 のとき左よせ, 0 のときセンタリング, +1 のとき右よせする. 省略値 0.
sw	<I>	大きさのスイッチ. 1 のとき小さめ, 2 の時大きめのタイトルを描く. 省略値 1.

4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (graph)

17.2.9 DclDrawAxisLine

1. 機能
   座標軸の線を描く.
2. 書式
   ```
   call DclDrawAxisLine(side, [sw])
   ```
3. 引数

side	<C*>	座標軸の位置.
sw	<I>	大きさのスイッチ. 1 の時細めの線, 2 の時太めの線を描く. 省略値 1.

4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (graph)

17.2.10 DclDrawTickmark

1. 機能
   目盛を描く.

2. 書式

```
call DclDrawTickmark(side, position, [sw])
```

3. 引数

side	<C*>	座標軸の位置.
position	<R(:)>	目盛の位置 (ユーザー座標系).
sw	<I>	大きさのスイッチ.
		1 の時小さめの目盛, 2 の時大きめの目盛を描く. 省略値 1.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (graph)

17.2.11 DclDrawAxisLabel

1. 機能

ラベルを描く.

2. 書式

```
call DclDrawAxisLabel(side, position, [label], [sw])
```

3. 引数

side	<C*>	座標軸の位置.
position	<R(:)>	ラベルの位置 (ユーザー座標系).
label	<C*(:)>	ラベルの文字列. 省略すると, ラベル位置の座標値を描く.
sw	<I>	大きさのスイッチ.
		1 の時小さめ, 2 の時大きめのラベルを描く. 省略値 1.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (graph)

17.2.12 DclShiftAxis

1. 機能

座標軸を外側にシフトする.

2. 書式

```
call DclShiftAxis(side)
```

3. 引数

side	<C*>	座標軸の位置.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (graph)

17.2.13 DclScalingPoint

1. 機能

   グラフの中に含めたい座標点を設定する.

2. 書式

```
call DclScalingPoint([x], [y])
```

3. 引数

   x,y <R(:)> 点の座標値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (graph)

17.2.14 DclSetTitle

1. 機能

   座標軸のタイトルを設定する.

2. 書式

```
call DclSetTitle([xtitle], [ytitle], [xunit], [yunit])
```

3. 引数

   xtitle <C*> x 軸のタイトル.

   ytitle <C*> y 軸のタイトル.

   xunit <C*> x 軸の単位.

   xunit <C*> y 軸の単位.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (graph)

17.2.15 DclSetAxisFactor

1. 機能

   文字, 目盛等の大きさを変える.

2. 書式

```
call DclSetAxisFactor(fact)
```

3. 引数

   fact <R> 倍率.

4. 備考

   なし.

5. 関連項目

- 関連ルーチン (graph)

第 18 章

グリッド

18.1 ルーチンリスト

18.1.1 設定

DclSetXGrid (UWSGXA)	格子点配列の格子点 x 座標を各座標値で設定する.
DclSetYGrid (UWSGYA)	格子点配列の格子点 y 座標を各座標値で設定する.
DclSetXEvenGrid (UWSGXB)	格子点 x 座標を等間隔に設定する.
DclSetYEvenGrid (UWSGYB)	格子点 y 座標を等間隔に設定する.

18.1.2 参照

DclGetXGrid	現在設定されている格子点の x 座標値を参照する.
(UWQGXA)	
DclGetYGrid	現在設定されている格子点の y 座標値を参照する.
(UWQGYA)	
DclGetXEvenGrid	格子点 x 座標の最小値, 最大値, 格子点数を参照する.
(UWQGXB)	
DclGetYEvenGrid	格子点 y 座標の最小値, 最大値, 格子点数を参照する.
(UWQGYB)	
DclGetXGridValue	格子点の x 座標値を参照する.
(RUWGX)	
DclGetYGridValue	格子点の y 座標値を参照する.
(RUWGY)	
DclGetXGridNumber	x 座標値に最も近い格子番号を参照する.
(IUWGX)	
DclGetYGridNumber	y 座標値に最も近い格子番号を参照する.
(IUWGY)	

* 括弧の中は, 対応する f77 インターフェイス名.

18.2 各ルーチンの説明

18.2.1 DclSetXGrid

1. 機能

   格子点配列の格子点 X 座標を各座標値で設定する.

2. 書式

   ```
   call DclSetXGrid(x)
   ```

3. 引数

   x <R(:)> グリッドの x 座標値.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (grid)

18.2.2 DclSetYGrid

1. 機能

   格子点配列の格子点 Y 座標を各座標値で設定する.

2. 書式

```
call DclSetYGrid(y)
```

3. 引数

y	<R(:)>	グリッドの y 座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (grid)

18.2.3 DclSetXEvenGrid

1. 機能

格子点 X 座標を等間隔に設定する.

2. 書式

```
call DclSetXEvenGrid(xmin, xmax, n)
```

3. 引数

xmin, xmax	<R>	x 座標の最小, 最大値.
n	<I>	格子点数. xmin と xmax の間を n-1 等分する.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (grid)

18.2.4 DclSetYEvenGrid

1. 機能

格子点 Y 座標を等間隔に設定する.

2. 書式

```
call DclSetYEvenGrid(ymin, ymax, n)
```

3. 引数

ymin, ymax	<R>	y 座標の最小, 最大値.
n	<I>	格子点数. ymin と ymax の間を n-1 等分する.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (grid)

18.2.5 DclGetXGrid

1. 機能

現在設定されている格子点の X 座標値を参照する.

2. 書式

```
call DclGetXGrid(x)
```

3. 引数

x	<R(:)>	格子点の x 座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (grid)

18.2.6 DclGetYGrid

1. 機能

現在設定されている格子点の Y 座標値を参照する.

2. 書式

```
call DclGetYGrid(y)
```

3. 引数

y	<R(:)>	格子点の y 座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (grid)

18.2.7 DclGetXEvenGrid

1. 機能

格子点 X 座標の最小値, 最大値, 格子点数を参照する.

2. 書式

```
call DclGetXEvenGrid(xmin, xmax, n)
```

3. 引数

xmin, xmax	<R>	x 座標の最小, 最大値.
n	<I>	格子点数.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (grid)

18.2.8 DclGetYEvenGrid

1. 機能

格子点 Y 座標の最小値, 最大値, 格子点数を参照する.

2. 書式

```
call DclGetYEvenGrid(ymin, ymax, n)
```

3. 引数

ymin, ymax	<R>	y 座標の最小, 最大値.
n	<I>	格子点数.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (grid)

18.2.9 DclGetXGridValue

1. 機能

格子点の X 座標値を参照する.

2. 書式

```
result=DclGetXGridValue(ix)
```

3. 引数

戻り値	<R>	x 座標値.
ix	<I>	格子点番号.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (grid)

18.2.10 DclGetYGridValue

1. 機能

格子点の Y 座標値を参照する.

2. 書式

```
result=DclGetYGridValue(iy)
```

3. 引数

戻り値	<R>	y 座標値.
iy	<I>	格子点番号.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (grid)

18.2.11 DclGetXGridNumber

1. 機能

X 座標値に最も近い格子番号を参照する.

2. 書式

```
result=DclGetXGridNumber(x)
```

3. 引数

戻り値	<I>	格子点番号.
x	<R>	x 座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (grid)

18.2.12 DclGetYGridNumber

1. 機能

Y 座標値に最も近い格子番号を参照する.

2. 書式

```
result=DclGetYGridNumber(y)
```

3. 引数

戻り値	<I>	格子点番号.
y	<R>	y 座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (grid)

第 II 部

3D グラフィックス

第 19 章

3D コントロール

19.1 ルーチンリスト

DclSet3DTransFunction　3 次元正規変換の変換関数を確定する.
(SCSTRF)

19.1.1 設定

Set3DViewPort　3 次元正規変換のビューポートを設定する.
(SCSVPT)
Set3DWindow　3 次元正規変換のウインドウを設定する.
(SCSWND)
Set3DLogAxis　3 次元座標の対数軸を設定する.
(SCSLOG)
Set3DOrigin　3 次元正規変換のスケーリングファクターと原点を設定する.
(SCSOG)
Set3DTransNumber　3 次元正規変換の変換関数番号を設定する.
(SCSTRN)

19.1.2 参照

Get3DViewPort　3 次元正規変換のビューポートを参照する.
(SCQVPT)
Get3DWindow　3 次元正規変換のウインドウを参照する.
(SCQWND)
Get3DLogAxis　3 次元座標の対数軸を参照する.
(SCQLOG)
Get3DOrigin　3 次元正規変換のスケーリングファクターと原点を参照する.
(SCQOG)
Get3DTransNumber　3 次元正規変換の変換関数番号を参照する.
(SCQTRN)

* 括弧の中は, 対応する f77 インターフェイス名.

19.2 各ルーチンの説明

19.2.1 DclSet3DTransFunction

1. 機能
   3 次元正規変換の変換関数を確定する.
2. 書式
   ```
   call DclSet3DTransFunction()
   ```
3. 引数なし.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (3d_cntrl)

19.2.2 DclGet3DTransNumber

1. 機能
   3 次元正規変換の変換関数番号を参照する.
2. 書式
   ```
   result=DclGet3DTransNumber()
   ```
3. 引数
   戻り値 <I> 変換関数番号.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (3d_cntrl)

19.2.3 DclGet3DViewPort

1. 機能
   3 次元正規変換のビューポートを参照する.
2. 書式
   ```
   call DclGet3DViewPort(xmin, xmax, ymin, ymax, zmin, zmax)
   ```
3. 引数
   xmin, xmax <R> x 座標の最大最小値.
   ymin, ymax <R> y 座標の最大最小値.
   zmin, zmax <R> z 座標の最大最小値.
4. 備考
   なし.

5. 関連項目
   - 関連ルーチン (3d_cntrl)

19.2.4 DclGet3DWindow

1. 機能

   3 次元正規変換のウィンドウを参照する.

2. 書式

```
call DclGet3DWindow(xmin, xmax, ymin, ymax, zmin, zmax)
```

3. 引数

   xmin, xmax <R> x 座標の最大最小値.
   ymin, ymax <R> y 座標の最大最小値.
   zmin, zmax <R> z 座標の最大最小値.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (3d_cntrl)

19.2.5 DclGet3DLogAxis

1. 機能

   3 次元座標の対数軸を設定する.

2. 書式

```
call DclGet3DLogAxis(log_x, log_y, log_z)
```

3. 引数

   log_x <L> x 軸の対数フラグ. .true. の時, 対数軸にする.
   log_y <L> y 軸の対数フラグ. .true. の時, 対数軸にする.
   log_z <L> z 軸の対数フラグ. .true. の時, 対数軸にする.

4. 備考

   なし.

5. 関連項目
   - 関連ルーチン (3d_cntrl)

19.2.6 DclGet3DOrigin

1. 機能

   3 次元正規変換のスケーリングファクターと原点を参照する.

2. 書式

```
call DclGet3DOrigin(factor, x, y, z)
```

3. 引数

factor <R> スケーリングの係数.

x, y, z <R> 原点の座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_cntrl)

19.2.7 DclSet3DTransNumber

1. 機能

3 次元正規変換の変換関数番号を設定する.

2. 書式

```
call DclSet3DTransNumber(num)
```

3. 引数

num <I> 変換関数番号.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_cntrl)

19.2.8 DclSet3DViewPort

1. 機能

3 次元正規変換のビューポートを設定する.

2. 書式

```
call DclSet3DViewPort(xmin, xmax, ymin, ymax, zmin, zmax)
```

3. 引数

xmin, xmax <R> x 座標の最大最小値.

ymin, ymax <R> y 座標の最大最小値.

zmin, zmax <R> z 座標の最大最小値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_cntrl)

19.2.9 DclSet3DWindow

1. 機能

3 次元正規変換のウィンドウを設定する.

2. 書式

```
call DclSet3DWindow(xmin, xmax, ymin, ymax, zmin, zmax)
```

3. 引数

xmin, xmax	<R>	x 座標の最大最小値.
ymin, ymax	<R>	y 座標の最大最小値.
zmin, zmax	<R>	z 座標の最大最小値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_cntrl)

19.2.10 DclSet3DLogAxis

1. 機能

3 次元座標の対数軸を設定する.

2. 書式

```
call DclSet3DLogAxis(log_x, log_y, log_z)
```

3. 引数

log_x	<L>	x 軸の対数フラグ. .true. の時, 対数軸にする.
log_y	<L>	y 軸の対数フラグ. .true. の時, 対数軸にする.
log_z	<L>	z 軸の対数フラグ. .true. の時, 対数軸にする.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_cntrl)

19.2.11 DclSet3DOrigin

1. 機能

3 次元正規変換のスケーリングファクターと原点を設定する.

2. 書式

```
call DclSet3DOrigin(factor, x, y, z)
```

3. 引数

factor	<R>	スケーリングの係数.
x, y, z	<R>	原点の座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_cntrl)

第 20 章

透視変換

20.1 ルーチンリスト

20.1.1 確定

DclSet3DProjection (SCSPRJ)	透視変換を確定する.

20.1.2 設定

DclSet3DEyePoint (SCSEYE)	透視変換の視点を設定する.
DclSet3DObjectPoint (SCSOBJ)	透視変換の焦点を設定する.
DclSet2DPlane (SCSPLN)	2 次元平面を 3 次元空間に割付る.

20.1.3 参照

DclGet3DEyePoint (SCQEYE)	透視変換の視点を参照する.
DclGet3DObjectPoint (SCQOBJ)	透視変換の焦点を参照する.
DclGet2DPlane (SCQPLN)	割りつけられた 2 次元平面の情報を参照する.

* 括弧の中は, 対応する f77 インターフェイス名.

20.2 各ルーチンの説明

20.2.1 DclSet3DProjection

1. 機能

   透視変換を確定する.

2. 書式

```
call DclSet3DProjection()
```

3. 引数なし.
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (3d_prjct)

20.2.2 DclSet3DEyePoint

1. 機能

   透視変換の視点を設定する.
2. 書式

```
call DclSet3DEyePoint(x, y, z)
```

3. 引数

   `x, y, z <R>` 視点の座標値.
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (3d_prjct)

20.2.3 DclSet3DObjectPoint

1. 機能

   透視変換の焦点を設定する.
2. 書式

```
call DclSet3DObjectPoint(x, y, z)
```

3. 引数

   `x, y, z <R>` 焦点の座標値.
4. 備考

   なし.
5. 関連項目
   - 関連ルーチン (3d_prjct)

20.2.4 DclSet2DPlane

1. 機能

   2 次元平面を 3 次元空間に割付る.
2. 書式

```
call DclSet2DPlane(x_dir, y_dir, section)
```

3. 引数

x_dir, y_dir	<I>	2 次元平面の x 座標、y 座標に対応する 3 次元座標を以下の数字で指定する. 1: x 座標 2: y 座標 3: z 座標 これらの数値に負の値を指定すると, 正負の方向を逆に割り当てる.
section	<R>	2 次元平面が切る座標軸の切片.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_prjct)

20.2.5 DclGet3DEyePoint

1. 機能

透視変換の視点を参照する.

2. 書式

```
call DclGet3DEyePoint(x, y, z)
```

3. 引数

x, y, z	<R>	視点の座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_prjct)

20.2.6 DclGet3DObjectPoint

1. 機能

透視変換の焦点を参照する.

2. 書式

```
call DclGet3DObjectPoint(x, y, z)
```

3. 引数

x, y, z	<R>	焦点の座標値.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_prjct)

20.2.7 DclGet2DPlane

1. 機能

   割りつけられた 2 次元平面の情報を参照する.

2. 書式

   ```
   call DclGet2DPlane(x_dir, y_dir, section)
   ```

3. 引数

x_dir, y_dir	<I>	2 次元平面の x 座標、y 座標に対応する 3 次元座標軸. 数字の意味については、DclSet3DPlane を参照.
section	<R>	2 次元平面が切る座標軸の切片.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (3d_prjct)

第 21 章

3D ライン

21.1 ルーチンリスト

21.1.1 描画

DclDraw3DLine	3 次元ユーザー座標系で折れ線を描く.
(SCPLU,SCPLZU)	
DclDraw3DLineNormalized	3 次元正規座標系で折れ線を描く.
(SCPLV,SCPLZV)	

21.1.2 設定

DclSet3DLineIndex	折れ線のラインインデクスを設定する.
(SCSPLI)	

21.1.3 参照

DclGet3DLineIndex	折れ線のラインインデクスを参照する.
(SCQPLI)	

* 括弧の中は, 対応する f77 インターフェイス名.

21.2 各ルーチンの説明

21.2.1 DclDraw3DLine

1. 機能

   3 次元ユーザー座標系で折れ線を描く.

2. 書式

   ```
   call DclDraw3DLine(x, y, z, [index])
   ```

3. 引数

x, y, z	<R(:)>	折線の座標値.
index	<I>	折線のラインインデクス. 省略値は 1.

4. 備考

なし.

5. 関連項目
   - 関連ルーチン (3d_line)

21.2.2 DclDraw3DLineNormalized

1. 機能
   3 次元正規座標系で折れ線を描く.
2. 書式
   ```
   call DclDraw3DLineNormalized(x, y, z, [index])
   ```
3. 引数
x, y, z	<R(:)>	折線の座標値.
index	<I>	折線のラインインデクス. 省略値は 1.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (3d_line)

21.2.3 DclSet3DLineIndex

1. 機能
   折れ線のラインインデクスを設定する.
2. 書式
   ```
   call DclSet3DLineIndex(index)
   ```
3. 引数
index	<I>	折線のラインインデクス.
4. 備考
   なし.
5. 関連項目
   - 関連ルーチン (3d_line)

21.2.4 DclGet3DLineIndex

1. 機能
   折れ線のラインインデクスを設定する.
2. 書式
   ```
   result=DclGet3DLineIndex()
   ```
3. 引数
戻り値	<I>	ラインインデクス.
4. 備考

なし.

5. 関連項目

   - 関連ルーチン (3d_line)

第 22 章

3D マーカー

22.1 ルーチンリスト

22.1.1 描画

DclDraw3DMarker (SCPMU,SCPMZU)	3 次元ユーザー座標系でマーカー列を描く.
DclDraw3DMarkerNormalized (SCPMV,SCPMZV)	3 次元正規座標系でマーカー列を描く.

22.1.2 設定

DclSet3DMarkerType (SCSPMT)	マーカータイプを設定する.
DclSet3DMarkerIndex (SCSPMI)	マーカーのラインインデクスを設定する.
DclSet3DMarkerSize (SCSPMS)	マーカーの大きさを設定する.

22.1.3 参照

DclGet3DMarkerType (SCQPMT)	マーカータイプを参照する.
DclGet3DMarkerIndex (SCQPMI)	マーカーのラインインデクスを参照する.
DclGet3DMarkerSize (SCQPMS)	マーカーの大きさを参照する.

* 括弧の中は, 対応する f77 インターフェイス名.

22.2 各ルーチンの説明

22.2.1 DclDraw3DMarker

1. 機能

   3 次元ユーザー座標系でマーカー列を描く.

2. 書式

```
call DclDraw3DMarker(x, y, z, [type], [index], [height])
```

3. 引数

x, y, z	<R(:)>	マーカー列の座標値.
type	<I>	マーカータイプ.
index	<I>	マーカーのラインインデクス.
height	<R>	マーカーの高さ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (3d_mark)

22.2.2 DclDraw3DMarkerNormalized

1. 機能

   3 次元正規座標系でマーカー列を描く.

2. 書式

```
call DclDraw3DMarkerNormalized(x, y, z, [type], [index], [height])
```

3. 引数

x, y, z	<R(:)>	マーカー列の座標値.
type	<I>	マーカータイプ.
index	<I>	マーカーのラインインデクス.
height	<R>	マーカーの高さ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (3d_mark)

22.2.3 DclSet3DMarkerType

1. 機能

   マーカータイプを設定する.

2. 書式

```
call DclSet3DMarkerType(type)
```

3. 引数

   type <I> マーカータイプ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (3d_mark)

22.2.4 DclSet3DMarkerIndex

1. 機能

   マーカーのラインインデクスを設定する.

2. 書式

```
call DclSet3DMarkerIndex(index)
```

3. 引数

   index <I> マーカーのラインインデクス.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (3d_mark)

22.2.5 DclSet3DMarkerSize

1. 機能

   マーカーの大きさを設定する.

2. 書式

```
call DclSet3DMarkerSize(height)
```

3. 引数

   height <R> マーカーの大きさ.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (3d_mark)

22.2.6 DclGet3DMarkerType

1. 機能

   マーカータイプを参照する.

2. 書式

```
result=DclGet3DMarkerType()
```

3. 引数

戻り値 <I> マーカータイプ.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_mark)

22.2.7 DclGet3DMarkerIndex

1. 機能

マーカーのラインインデクスを参照する.

2. 書式

```
result=DclGet3DMarkerIndex()
```

3. 引数

戻り値 <I> マーカーのラインインデクス.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_mark)

22.2.8 DclGet3DMarkerSize

1. 機能

マーカーの大きさを参照する.

2. 書式

```
result=DclGet3DMarkerSize()
```

3. 引数

戻り値 <R> マーカーの大きさ.

4. 備考

なし.

5. 関連項目

- 関連ルーチン (3d_mark)

第 23 章

3D トーン

23.1 ルーチンリスト

23.1.1 描画

DclDraw3DHatch	3 次元ユーザー座標系でトーンを描く.
(SCTNU,SCTNZU)	
DclDraw3DHatchNormalized	3 次元正規座標系でトーンを描く.
(SCTNV,SCTNZV)	

23.1.2 設定

DclSet3DHatchPattern	トーンパターン番号を設定する.
(SCSTNP)	

23.1.3 参照

DclGet3DHatchPattern	トーンパターン番号を参照する.
(SCQTNP)	

* 括弧の中は, 対応する f77 インターフェイス名.

23.2 各ルーチンの説明

23.2.1 DclDraw3DHatch

1. 機能

   3 次元ユーザー座標系でトーンを描く.

2. 書式

   ```
   call DclDraw3DHatch(x, y, z, [pattern1], [pattern2])
   ```

3. 引数

x, y, z	<R(:)>	塗りつぶす多角形の頂点座標.
pattern1	<I>	表のパターン番号.
pattern2	<I>	裏のパターン番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (3d_tone)

23.2.2 DclDraw3DHatchNormalized

1. 機能

   3 次元正規座標系でトーンを描く.

2. 書式

```
call DclDraw3DHatchNormalized(x, y, z, [pattern1], [pattern2])
```

3. 引数

x, y, z	<R(:)>	塗りつぶす多角形の頂点座標.
pattern1	<I>	表のパターン番号.
pattern2	<I>	裏のパターン番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (3d_tone)

23.2.3 DclSet3DHatchPattern

1. 機能

   トーンパターン番号を設定する.

2. 書式

```
call DclSet3DHatchPattern([pattern1], [pattern2])
```

3. 引数

pattern1	<I>	表のパターン番号.
pattern2	<I>	裏のパターン番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (3d_tone)

23.2.4 DclGet3DHatchPattern

1. 機能

   トーンパターン番号を参照する.

2. 書式

```
call DclGet3DHatchPattern(pattern1, pattern2)
```

3. 引数

`pattern1` `<I>` 表のパターン番号.

`pattern2` `<I>` 裏のパターン番号.

4. 備考

   なし.

5. 関連項目

   - 関連ルーチン (3d_tone)