

DSHEng4 装置通信エンジン (GEM+GEM300)
ソフトウェア・パッケージ

APP インタフェース ライブラリ関数説明書

(C, C++, .Net-Vb,C#)

VOL- 5 / 1 5

- 3 . 9 . Spool スプール関連関数
- 3 . 10 . 端末表示関連関数

2 0 0 9 年 7 月

株式会社データマップ



[取り扱い注意]

- ・ この資料ならびにソフトウェアの一部または全部を無断で使用、複製することはできません。
- ・ 本説明書に記述されている内容は予告なしで変更される可能性があります。
- ・ Windows は米国 Microsoft Corporation の登録商標です。
- ・ ユーザーが本ソフトウェアの使用によって生じた遺失履歴、(株)データマップの予見の有無を問わず発生した特別損害、付随的損害、間接損害およびその他の拡大損害に対して責任を負いません。

【改訂履歴】

番号	改訂日付	項目	概略
1.	2009. 7	初版	

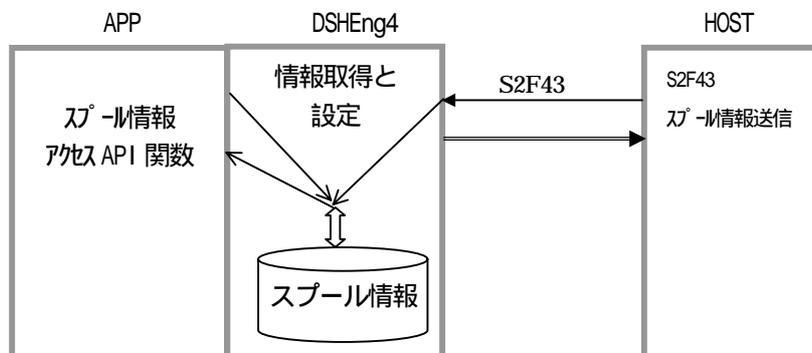
目 次

3.9 Spool スプール関連関数.....	1
3.9.1 使用する情報格納構造体.....	3
3.9.2 Spool スプール情報アクセス関数.....	5
3.9.2.1 EngSetSpoolInfo() - ストリーム指定スプール情報の登録.....	5
3.9.2.2 EngSetAllSpoolInfo() - 全スプール情報の登録.....	6
3.9.2.3 EngGetSpoolInfo() - ストリームのスプール登録情報取得.....	8
3.9.2.4 EngGetAllSpoolInfo() - 全スプール情報の取得.....	9
3.9.2.5 EngResetSpoolInfo() - ストリーム指定のスプール登録解除.....	10
3.9.2.6 EngResetAllSpoolInfo() - 全スプール登録の削除.....	11
3.9.3 Spool スプール関連ライブラリ関数.....	12
3.9.3.1 DshDecodeS2F43() - S2F43 をスプール情報にデコード.....	12
3.9.3.2 DshFreeTSPOOL_INFO() - スプール情報構造体メモリの開放.....	14
3.9.3.3 DshCopyTSPOOL_INFO() - スプール情報構造体メモリのコピー.....	15
3.9.3.4 DshInitTSPOOL_INFO() - スプールS2F43 情報の初期設定.....	16
3.9.3.5 DshPutTSPOOL_INFO() - スプール情報の設定.....	17
3.9.3.6 DshInitTSTRE_INFO() - スプール・ストリーム情報の初期設定.....	18
3.9.3.7 DshPutTSTRE_INFO() - ストリーム情報の設定.....	19
3.9.3.8 DshInitTSPOOL_ERR_INFO() - スプールS2F43 応答情報の初期設定.....	20
3.9.3.9 DshPutTSPOOL_ERR_INFO() - スプールリセットエラー情報の設定.....	21
3.9.3.10 DshFreeTSPOOL_ERR_INFO() - スプールS2F43 応答情報メモリの開放.....	22
3.9.3.11 DshMakeS2F43Response() - S2F43 の応答メッセージの生成.....	23
3.9.4 ユーザ作成ライブラリ関数.....	25
3.9.4.1 DshResponseS2F44() - S2F44 スプール設定応答メッセージ.....	25
3.10 端末サービス情報関連関数.....	27
3.10.1 使用する情報格納構造体.....	28
3.10.2 端末メッセージ送信関数.....	29
3.10.2.1 EngSendS10F10() - 端末要求メッセージ送信関数.....	29
3.10.3 端末表示関連ライブラリ関数.....	31
3.10.3.1 DshDecodeS10F30() - S10F30 端末メッセージのデコード.....	31
3.10.3.2 DshDecodeS10F50() - S10F50 端末情報をTTERMTEXT_INFO構造体にデコード.....	32
3.10.3.3 DshFreeTTERMTEXT_INFO() - 端末表示情報構造体メモリの開放.....	33
3.10.3.4 DshCopyTTERMTEXT_INFO() - 端末表示情報構造体メモリのコピー.....	34
3.10.4 ユーザ作成ライブラリ関数.....	35
3.10.4.1 DshResponseS10F40() - S10F40 端末表示応答メッセージ.....	35
3.10.4.2 DshResponseS10F60() - S10F60 端末表示 (Multi-Block)応答メッセージ.....	36

(VOL - 6 に続く)

3.9 Spool スプール関連関数

ここで述べるスプール関連情報は、DSHEng4 が管理します。従って、APP はこれらの情報をアクセスするために以下の DSHEng4 ライブラリ関数を使用します。



(1) 情報アクセスと送信 API 関数

スプール情報アクセスに関連するサービスのための API 関数名は一覧表のとおりです。

	API 関数名	機能
1	EngPutSpoolInfo()	指定した STREAM のスプール情報を登録または変更します。
2	EngGetSpoolInfo()	指定した STREAM のスプール情報を取得します。
3	EngPutAllSpoolInfo()	全 STREAM のスプール情報を登録または変更します。
4	EngGetAllSpoolInfo()	登録済みの全 STREAM のスプール情報を取得します。
5	EngResetSpoolInfo()	指定した STREAM のスプール登録を解除します。(スプール不許可にします。)
6	EngResetAllSpoolInfo()	全 STREAM のスプール登録を解除します。(スプールを全て不許可にします。)

(2) ライブラリ関数

他に APP が使用できるスプール情報処理用 API 関数として、以下の関数があります。

	API 関数名	機能
1	DshDecodeS2F43()	S2F43 に含まれるスプール情報を TSPPOOL_INFO 構造体内にデコードします。
2	DshFreeTSPPOOL_INFO()	スプール情報が格納されている TSPPOOL_INFO 構造体と内部で使用されている全メモリを開放するための関数です。
3	DshCopyTSPPOOL_INFO()	TSPPOOL_INFO のスプール情報をコピーします。
4	DshInitTSPPOOL_INFO()	スプール情報構造体 TSPPOOL_INFO を初期設定します。
5	DshPutTSPPOOL_INFO()	TSPPOOL_INFO 構造体内に 1 個の Stream 情報を設定します。
6	DshInitTSTRE_INFO()	1 個のストリーム情報構造体を初期設定します。
7	DshPutTSTRE_INFO()	ストリーム情報構造体に 1 個の Function を設定します。
8	DshInitTSPPOOL_ERR_INFO()	S2F44 メッセージのエンコードに使用される応答情報を TSPPOOL_ERR_INFO 構造体の初期設定を行います。
9	DshPutSpool_ERR_INFO()	TSPPOOL_ERR_INFO 構造体内にエラー情報を設定します。
10	DshFreeTSPPOOL_ERR_INFO()	TSPPOOL_ERR_INFO 構造体内に使用されたメモリを開放します。
11	DshMakeS2F44Response()	S2F43 に対する応答メッセージを TSPPOOL_ERR_INFO 構造体内の情報に基づいて生成します。

(3) ユーザ作成ライブラリ関数

	ライブラリ関数名	機能
1	DshResponseS2F44()	S2F43 スプール (Stream, Function) 設定応答

3.9.1 使用する情報格納構造体

スプール情報を操作する関数は、情報格納のための共通の TSPPOOL_INFO 構造体を使用します。TSPPOOL_INFO と、その内部で使用する他の構造体は下記のとおりです。

(1) TSPPOOL_INFO - Spool Information

```
typedef struct{
    int      s_count;           // # of streams
    TSTRE_INFO **stre_list;    // stream info list
} TSPPOOL_INFO;
```

s_count = 0 の場合、スプール対象 Stream が 1 つもないことを意味します。

s_count > 0 の場合、stre_list[] に stream と function 情報が格納されている TSTRE_INFO のポインタが格納されています。

(2) TSTRE_INFO - Spool Stream Information

```
typedef struct{
    int      stream;           // stream
    int      f_count;         // # of functions
    int      *func_list;      // fuction list
} TSTRE_INFO;
```

stream はスプール指定された stream 値が格納されます。

f_count には、stream に属するメッセージで、スプール対象となる function の数になります。

=0 の場合、当該 stream についてスプール function が無いことを意味します。

func_list[] には f_count の数だけの function 値がリストで格納されます。

(3) TSPPOOL_ERR_INFO S2F44 応答情報

```
typedef struct{
    int      rsack;           // ack for s2f43
    int      err_count;      // # of streams
    TSTRE_ERR_INFO **stre_list; // err stream info list
} TSPPOOL_ERR_INFO;
```

DSHEng4 はこの情報から S2F44 メッセージを組立て戻りに送信します。

rsack は S2F44 の RSACK の値です。

S2F43 の中に指定されたストリーム、ファンクションについて全て OK の場合は、=0 が設定されます。

1 個以上のエラーが見つかった場合、rsack には 0 以外の値が設定され、そのストリームとファンクションが stre_list に設定されます。

err_count は rsack != 0 の時のエラー補足情報の数です。

この数はエラーが見つからなくてもそのストリームとファンクションを含めることができます。この場合、(4) で述べる func_err[i] の中に =0 を設定してください。func_err[i]=0 の場合、正常扱いになります。

stre_list[] は、エラー情報ポインタのリストです。err_count 分のリストです。

(4) TSTRE_ERR_INFO - エラー補足情報

```
typedef struct{                                // S2F43 からの取得情報
    int    strack;                             // ack for stream
    int    stream;                             // stream
    int    f_count;                            // # of functions
    int    *func_list;                         // fuction list
    int    *func_err;                          // func err( 0/1 )
} TSTRE_ERR_INFO;
```

1 個のストリームについて、どの機能がどのエラーであったかを報せます。

strack はそのストリームに対する STRACK です。

stream はストリームです。

f_count は stream 中の機能の数です。

func_list は機能の値のリストです。f_count 分の配列です。

func_err は func_list の配列に対応していて、エラーであったかどうかを示します。

=0 の場合はエラーがなかったことを、!=0 の場合エラーがあったことを意味します。

3.9.2 Spool スプール情報アクセス関数

3.9.2.1 EngSetSpoolInfo() - ストリーム指定スプール情報の登録

(1) 呼出書式

[C, C++]

```
API int APIX EngSetSpoolInfo(
    BYTE    stream,           // スプール登録したいstream
    int     f_count,         // 指定 function の数
    BYTE    *func_list       // f_count 分の function のリスト
);
```

[.NET VB]

```
Function EngSetSpoolInfo (
    ByVal stream As Int32,
    ByVal f_count As Int32,
    ByRef func_list As Int32) As Int32
```

[.NET C#]

```
int EngSetSpoolInfo(
    int stream,
    int f_count,
    int []func_list );
```

(2) 引数

stream

登録したいstream(Sx)の値(2~127)です。

f_count

stream に属する登録したい function(Fy)の数です。

func_list

f_count 分の function の値が格納されている領域のポインタです。

(3) 戻り値

戻り値	意味
0	正常に登録できた。
(-1)	登録できなかった。

(4) 説明

ストリーム単位でそのストリームに属するスプール対象にしたいファンクションを指定して DSHeng4 に登録します。指定ストリームについて既に登録されている情報は一旦解除され新しく設定しなおします。

(5) コーディング例

S6F11, S6F13 の2個をスプール対象にしたい場合、例えば次のようにコーディングします。

```
BYTE    func_list[2];
int     end_status;
func_list[0]=11;   func_list[1]=13;
end_status = EngPutSpoolInfo(6, 2, func_list );
```

3.9.2.2 EngSetAllSpoolInfo() - 全スプール情報の登録

(1) 呼出書式

[C, C++]

```
API int APIX EngSetAllSpoolInfo(
    TSPPOOL_INFO *psinfo           // スプール情報格納構造体のポインタ
);
```

[.NET VB]

```
Function EngSetAllSpoolInfo (
    ByRef ppinfo As dsh_info.TSPPOOL_INFO) As Int32
```

[.NET C#]

```
int EngSetAllSpoolInfo(
    ref TSPPOOL_INFO ppinfo );
```

(2) 引数

psinfo

登録したいスプール情報が格納されている構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常に登録できた。
(-1)	登録できなかった。

(4) 説明

psinfo に格納されているスプール情報を DSHeng4 に登録します。

既に登録されているスプール情報は全て解除された上で新しく情報を設定します。

psinfo 内には、スプールしたい STREAM とその FUNCTION リスト情報を格納することができます。

(5) コーディング例

S5F1, S6F11, S6F13 を登録する場合、例えば、以下のようにコーディングします。

```
TSPPOOL_INFO *sinfo;
sinfo = calloc( sizeof(TSPPOOL_INFO), 1 );
sinfo->s_count = 2;           // S5 と S6
sinfo->stre_list = calloc( sizeof(TSTRE_INFO*) * 2, 1 ); // stre_list ポインタリスト用
sinfo->stre_list[0] = calloc( sizeof(TSTRE_INFO), 1 ); // S5 用
sinfo->stre_list[0]->stream = 5;           // S5
sinfo->stre_list[0]->f_count = 1;         // 1 個
sinfo->stre_list[0]->func_list = malloc( 1 );
sinfo->stre_list[0]->func_list[0] = 1;     // F1

sinfo->stre_list[1] = calloc( sizeof(TSTRE_INFO), 1 ); // S6 用
sinfo->stre_list[1]->stream = 6;           // S6
sinfo->stre_list[1]->f_count = 2;         // 2 個
sinfo->stre_list[1]->func_list = malloc( 2 );
sinfo->stre_list[1]->func_list[0] = 11;    // F11
```

```
sinfo->stre_list[1]->func_list[1] = 13;           // F13  
end_status = EngPutAllSpoolInfo( sinfo );        // 登録  
DshFreeTSPool_INFO( sinfo );                    // sinfo の全メモリを解放
```

.
.

(6) 補足事項

受信した S2F43 メッセージから、それに含まれているスプール情報を TSPool_INFO 構造体を取得するための関数 DshDecodeS4F23() が準備されています。

3.9.2.3 EngGetSpoolInfo() - ストリームのスプール登録情報取得

(1) 呼出書式

[C, C++]

```
API int APIX EngGetSpoolInfo(
    BYTE    stream;           // 取得対象 stream
    BYTE    *func_list       // スプール対象 function の格納リストのポインタ
);
```

[.NET VB]

```
Function EngGetSpoolInfo (
    ByVal stream As Int32,
    ByRef func_list As Int32) As Int32
```

[.NET C#]

```
int EngGetSpoolInfo(
    int stream,
    int []func_list );
```

(2) 引数

stream

取得する対象の stream を指定します。

func_list

取得した function を格納するリストのポインタです。登録されている可能性のある最大数の function を格納するための領域を確保しておく必要があります。

(3) 戻り値

戻り値	意味
>= 0	取得できた function の数
(-1)	stream が正しくなかった。

(4) 説明

stream に指定されたストリームについてスプール登録されている function 情報を取得し、function リストに格納します。

戻り値として 登録されている function 数が返却され、func_list には登録されている function の値を順に格納されます。

3.9.2.4 EngGetAllSpoolInfo() - 全スプール情報の取得

(1) 呼出書式

[C, C++]

```
API int APIX EngGetAllSpoolInfo(
    TSPPOOL_INFO *sinfo           // スプール情報を格納する構造体のポインタ
);
```

[.NET VB]

```
Function EngGetAllSpoolInfo (
    ByRef ppinfo As dsh_info.TSPPOOL_INFO) As Int32
```

[.NET C#]

```
int EngSetAllSpoolInfo(
    ref TSPPOOL_INFO ppinfo );
```

(2) 引数

sinfo

取得したスプール情報を格納するための構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常に取得できた。
(-1)	取得できなかった。

(4) 説明

DSHEng4 に登録されている全スプール情報を取得し、sinfo 構造体領域に情報を格納します。
構造体メンバー内の情報格納用メモリはDSHEng4 側で取得します。

スプール情報のために割付けられたメモリと、その構成メンバーデータ用にメモリが獲得されています。
これらのメモリは、使用后、ユーザがDSHEng4 のAPI 関数を使って次のように開放してください。

```
TSPPOOL_INFO    sinfo;

    if ( EngGetAllSpoolInfo( &sinfo ) == 0 ){
        sinfo の処理
        処理終了後
        DshFreeTSPPOOL_INFO( sinfo );           // sinfo と内部に使用されているメモリの開放
    }
```

sinfo->s_count に、登録されている stream の数が設定されます。この値が =0 の場合は、スプール対象メッセージがないことを意味します。

3.9.2.5 EngResetSpoolInfo() - ストリーム指定のプール登録解除

(1) 呼出書式

[C, C++]

```
API int APIX EngResetSpoolInfo(
    BYTE    stream;           // stream ( 1 < stream < 128 )
);
```

[.NET VB]

```
Function EngResetSpoolInfo (
    ByVal stream As Int32) As Int32
```

[.NET C#]

```
int EngResetSpoolInfo(
    int stream );
```

(2) 引数

stream

プール解除する対象の stream を指定します。

(3) 戻り値

戻り値	意味
0	正常に解除できた。
(-1)	stream の値が間違っている。

(4) 説明

DSHEng4 内部に登録されている stream で指定されたプール情報を全て解除します。

3.9.2.6 EngResetAllSpoolInfo() - 全スプール登録の削除

(1) 呼出書式

[C, C++]

```
API int APIX EngResetAllSpoolInfo( );
```

[.NET VB]

```
Function EngResetSpoolInfo ( ) As Int32
```

[.NET C#]

```
int EngResetSpoolInfo( );
```

(2) 引数

なし。

(3) 戻り値

戻り値	意味
0	正常に削除できた。

(4) 説明

DSHEng4 内に登録されている全てのスプール情報を解除します。

3.9.3 Spool スプール関連ライブラリ関数

3.9.3.1 DshDecodeS2F43() - S2F43 をスプール情報にデコード

(1) 呼出書式

[C, C++]

```
API int APIX DshDecodeS2F43(
    DSHMSG *smsg, // SECS メッセージ 情報構造体のポインタ
    TSPPOOL_INFO *sinfo // デコードした情報を格納する構造体のポインタ
);
```

[.NET VB]

```
Function DshDecodeS2F43 (
    ByRef smsg As dshdr2.DSHMSG,
    ByRef info As dsh_info.TSPPOOL_INFO) As Int32
```

[.NET C#]

```
int DshDecodeS2F43(
    ref DSHMSG smsg,
    ref TSPPOOL_INFO info );
```

(2) 引数

smsg

S2F43 の SECS メッセージ情報構造体のポインタです。

sinfo

デコードした結果のスプール情報を格納する構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常にデコードできた。
(-1)	smsg を正しくデコードできなかった。

(4) 説明

S2F43 メッセージに含まれるスプール情報を、ユーザプログラムが処理しやすい TSPPOOL_INFO 構造体の中にデコードします。

スプール構造体のメンバーに使用するメモリは全て DSHEng4 側が準備します。

従って、情報処理終了後は、sinfo のメモリを DshFreeTSPPOOL_INFO() 関数を使って解放してください。

(DshResponseS7F4() を使う場合、DshResonseS7F4() が開放してくれます。)

smsg S2F43

L,m

L,2

stream

L,n

func1,

func2,

funcn



L,2

.

.

(5) 関連関数

DshEncodeSpoolMSG() : TSPool_INFO の内容を S2F43 メッセージにエンコードします。

DshFreeTSPool_INFO() : TSPool_INFO のメモリを解放します。

DshCopyTSPool_INFO() : TSPool_INFO の内容を別の構造体にコピーします。

DshMakeS2F43Response() : TSPool_INFO の ACK 情報から S2F44 応答メッセージを作成します。

3.9.3.2 DshFreeTSPool_INFO() - スプール情報構造体メモリの開放

(1) 呼出書式

[C, C++]

```
API void APIX DshFreeTSPool_INFO(  
    TSPool_INFO *sinfo // メリを開放したいスプール情報構造体の格納ポインタ  
);
```

[.NET VB]

```
Sub DshFreeTSPool_INFO (  
    ByRef info As dsh_info.TSPool_INFO)
```

[.NET C#]

```
void DshFreeTSPool_INFO(  
    ref TSPool_INFO info );
```

(2) 引数

sinfo

メモリを解放したいスプール情報構造体のポインタです。

(3) 戻り値

なし。

(4) 説明

TSPool_INFO 構造体内で情報格納用に使用されているメモリを全て解放します。

開放した後、TSPool_INFO の内容を全て 0 で初期設定します。

sinfo が NULL ならば、何も処理しません。

3.9.3.3 DshCopyTSPPOOL_INFO() - スプール情報構造体メモリのコピー

(1) 呼出書式

[C, C++]

```
int DshCopyTSPPOOL_INFO(
    TSPPOOL_INFO *dinfo,           // 北°-先のポインタ
    TSPPOOL_INFO *sinfo           // 北°-元のポインタ
);
```

[.NET VB]

```
Function DshCopyTSPPOOL_INFO (
    ByRef pinfo As IntPtr,
    ByRef sinfo As dsh_info.TSPPOOL_INFO) As Int32
```

[.NET C#]

```
int DshCopyTSPPOOL_INFO(
    ref TSPPOOL_INFO pinfo,
    ref TSPPOOL_INFO sinfo );
```

(2) 引数

dinfo

スプール情報のコピー先構造体メモリのポインタを格納するポインタです。
構造体用メモリは本関数が取得します。

sinfo

コピー元のスプール情報が格納されている構造体メモリのポインタです。

(3) 戻り値

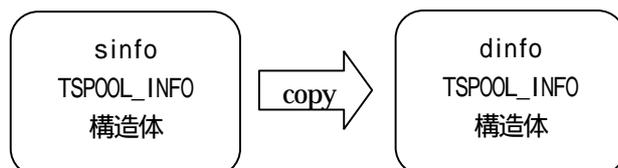
戻り値	意味
0	正常に北°-できた。
(-1)	sinfo または dinfo の値が NULL だったので北°-できなかった。

(4) 説明

sinfo が指す TSPPOOL_INFO 構造体内に格納されているスプール情報を dinfo で指定された TSPPOOL_INFO 構造体にコピーします。

dinfo 内のメンバーで新しいメモリが必要なものは本関数が取得します。

dinfo にコピーされた構造体のメモリはユーザが使用後、DSHFreeTSPPOOL_INFO()関数で解放してください。



3.9.3.4 DshInitTSPPOOL_INFO() スプールS2F43情報の初期設定

(1) 呼出書式

[C, C++]

```
API void APIX DshInitTSPPOOL_INFO(
    TSPPOOL_INFO *info,           // スプール情報構造体の格納ポインタ
    int s_count                   // スプール情報内に設定する TSTRE_INFO 構造体の数
);
```

[.NET VB]

```
Sub DshInitTSPPOOL_INFO (
    ByRef info As dsh_info.TSPPOOL_INFO,
    ByVal s_count As Int32)
```

[.NET C#]

```
void DshInitTSPPOOL_INFO(
    ref TSPPOOL_INFO info,
    int s_count );
```

(2) 引数

info

初期設定する TSPPOOL_INFO スプール情報構造体のポインタです。

s_count

スプール情報構造体に後で設定する Stream 情報の数です。

(3) 戻り値

なし。

(4) 説明

本関数は、TSPPOOL_INFO スプール情報構造体の初期設定を行います。
info で指定された構造体のメンバー s_count の値を設定します。

もし、s_count > 0 の場合は、stre_list に s_count だけの TSTRE_INFO 情報構造体のポインタリストを設けます。

実際に stre_list へのスプール情報 (TSTRE_INFO 構造体への) 設定は DshPutTSTRE_INFO() 関数を使って行ってください。s_count 分だけ順に設定できます。

TSPPOOL_INFO info 使用が済んだら、情報内に確保して使用したメモリは DshFreeTSPPOOL_INFO () 関数を使って開放することができます。

3.9.3.5 DshPutTSPPOOL_INFO () スプール情報の設定

(1) 呼出書式

[C, C++]

```
API void APIX DshPutTSPPOOL_INFO (
    TSPPOOL_INFO *info,           // スプール情報格納構造体のポインタ
    TSTRE_INFO *sinfo            // 1個のStreamの情報のポインタ
);
```

[.NET VB]

```
Function DshPutTSPPOOL_INFO (
    ByRef info As dsh_info.TSPPOOL_INFO,
    ByRef sinfo As dsh_info.TSTRE_INFO) As Int32
```

[.NET C#]

```
int DshPutTSPPOOL_INFO(
    ref TSPPOOL_INFO info,
    ref TSTRE_INFO sinfo );
```

(2) 引数

info

TSPPOOL_INFO スプール情報構造体のポインタです。

sinfo

Stream 1個分のストリーム情報が格納されている TSTRE_INFO 構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常に設定できた。
(-1)	リストが満杯で設定できなかった。

(4) 説明

本関数は、info 内の stre_list リストの先頭から空きリストを探します。

もし、空きリストがなければ、(-1)を返却します。

もし、空きリストがあれば、その空きリストにsinfoを設定し0を返却します。

STRE_INFO 構造体への1個のストリーム情報の設定にはDshInitTSTRE_INFO()、DshPutTSTRE_INFO()関数を使ってください。

3.9.3.6 DshInitTSTRE_INFO() スプール・ストリーム情報の初期設定

(1) 呼出書式

[C, C++]

```
API void APIX DshInitTSTRE_INFO(
    TSTRE_INFO **info,           // ストリーム情報構造体ポインタの格納ポインタ
    int stream,                 // Streamコード (Sx)
    int f_count                 // 設定する Function(Fy)の数
);
```

[.NET VB]

```
Sub DshInitTSTRE_INFO (
    ByRef pinfo As IntPtr,
    ByVal stream As Int32,
    ByVal f_count As Int32)
```

[.NET C#]

```
void DshInitTSTRE_INFO(
    ref IntPtr pinfo,
    int stream,
    int f_count );
```

(2) 引数

info

初期設定するスプールストリーム情報構造体のポインタです。

stream

TSTRE_INFO に設定する Stream の値です。

f_count

stream に含まれる function の数です。

(3) 戻り値

なし。

(4) 説明

本関数は、TSPPOOL_INFO 内に設定するストリーム 1 個分の情報を格納するための TSTRE_INFO 構造体の初期設定を行います。

TSTRE_INFO 構造体のメモリは本関数が確保し、そのポインタを info に設定し返します。

そして、構造体のメンバー stream, f_count の値を設定します。

もし、f_count > 0 の場合は、func_list に f_count だけの Function コードを格納するための領域を設けます。

f_list への Function の設定は DshPutTSTRE_INFO() 関数を使って行ってください。f_count 分だけ順に設定できます。

ここで確保されたメモリは DshFreeTSPPOOL_INFO() 関数実行時に開放されます。

3.9.3.7 DshPutTSTRE_INFO () ストリーム情報の設定

(1) 呼出書式

[C, C++]

```
API void APIX DshPutTSTRE_INFO (
    TSTRE_INFO *info,           // ストリーム情報格納構造体のポインタ
    int f_code                  // Functionコード
);
```

[.NET VB]

```
Function DshPutTSTRE_INFO (
    ByRef info As dsh_info.TSTRE_INFO,
    ByVal f_code As Int32) As Int32
```

[.NET C#]

```
int DshPutTSTRE_INFO(
    ref TSTRE_INFO info,
    int f_code );
```

(2) 引数

info

Stream と Function 情報を格納するための TSTRE_INFO 情報構造体のポインタです。

f_code

info 内の func_list に追加する Function コードです。

(3) 戻り値

戻り値	意味
0	正常に設定できた。
(-1)	リストが満杯で設定できなかった。

(4) 説明

本関数は、info 内の func_list リストの先頭から空きリストを探します。

もし、空きリストがなければ、(-1)を返却します。

もし、空きリストがあれば、その空きリストに f_code を設定し 0 を返却します。

3.9.3.8 DshInitTSPool_Err_Info() スプール S2F43 応答情報の初期設定

(1) 呼出書式

[C, C++]

```
API void APIX DshInitTSPool_Err_Info(
    TSPool_Err_Info *erinfo,           // 応答情報構造体の格納ポインタ
    int rsack,                          // S2F44 に設定する rsack
    int err_count                       // エラー情報 TSTRE_Err_Info 構造体の数
);
```

[.NET VB]

```
Sub DshInitTSPool_Err_Info (
    ByRef info As dsh_info.TSPool_Err_Info,
    ByVal rsack As Int32,
    ByVal err_count As Int32)
```

[.NET C#]

```
void DshInitTSPool_Err_Info(
    ref TSPool_Err_Info info,
    int rsack,
    int err_count );
```

(2) 引数

erinfo

初期設定する S2F43 スプール応答情報構造体のポインタです。

rsack

S2F44 の rsack に設定する値です。

err_count

スプール応答情報構造体に後で設定するエラー情報の数です。

(3) 戻り値

なし。

(4) 説明

本関数は、S2F43 スプールリセットの応答メッセージのための TSPool_Err_Info 構造体の初期設定を行います。

erinfo で指定された構造体のメンバー rsack, err_count の値を設定します。

もし、err_count > 0 の場合は、err_list に err_count だけの TSTRE_Err_Info エラー情報構造体のポインタを設けます。

実際に err_list へのエラー情報 (TSTRE_Err_Info 構造体) への設定は DshPutTSTRE_Err_Info() 関数を使って行ってください。err_count 分だけ順に設定できます。

処理が終了したあと、応答情報内に確保して使用したメモリは DshFreeTSPool_Err_Info () 関数を使って開放することができます。

3.9.3.9 DshPutTSPPOOL_ERR_INFO () スプールリセットエラー情報の設定

(1) 呼出書式

[C, C++]

```
API void APIX DshPutTSPPOOL_ERR_INFO (
    TSPPOOL_ERR_INFO *errinfo,           // エラ-情報格納構造体のポインタ
    TSTRE_ERR_INFO *serrinfo            // 1個のStreamのエラ-情報のポインタ
);
```

[.NET VB]

```
Function DshPutTSPPOOL_ERR_INFO (
    ByRef info As dsh_info.TSPPOOL_ERR_INFO,
    ByRef errinfo As dsh_info.TSTRE_ERR_INFO) As Int32
```

[.NET C#]

```
int DshPutTSPPOOL_ERR_INFO(
    ref TSPPOOL_ERR_INFO info,
    ref TSTRE_ERR_INFO errinfo );
```

(2) 引数

errinfo

S2F43 スプールリセット応答メッセージのための TSPPOOL_ERR_INFO 情報構造体のポインタです。

serrinfo

Stream 1 個分のストリームエラー情報が格納されている TSTRE_ERR_INFO 構造体のポインタです。

(3) 戻り値

戻り値	意味
0	正常に設定できた。
(-1)	リストが満杯で設定できなかった。

(4) 説明

本関数は、errinfo 内の stre_list リストの先頭から空きリストを探します。

もし、空きリストがなければ、(-1)を返却します。

もし、空きリストがあれば、その空きリストに serr_info を設定し 0 を返却します。

TSTRE_ERR_INFO 構造体への 1 個分のストリームエラー情報の設定には DshInitTSTRE_ERR_INFO()、

DshPutTSTRE_ERR_INFO()関数を使ってください。

3.9.3.10 DshFreeTSPPOOL_ERR_INFO() スプール S2F43 応答情報メモリの開放

(1) 呼出書式

[C, C++]

```
API void APIX DshFreeTSPPOOL_ERR_INFO(  
    TSPPOOL_ERR_INFO *info // メリを開放したい応答情報構造体の格納ポインタ  
);
```

[.NET VB]

```
Sub DshFreeTSPPOOL_ERR_INFO (  
    ByRef info As dsh_info.TSPPOOL_ERR_INFO)
```

[.NET C#]

```
void DshFreeTSPPOOL_ERR_INFO(  
    ref TSPPOOL_ERR_INFO info );
```

(2) 引数

info

メモリを解放したい S2F43 スプール応答情報構造体のポインタです。

(3) 戻り値

なし。

(4) 説明

TSPPOOL_ERR_INFO 構造体内で情報格納用に使用されているメモリを全て解放します。

3.9.3.11 DshMakeS2F43Response() - S2F43 の応答メッセージの生成

(1) 呼出書式

[C, C++]

```
API int APIX DshMakeS2F43Response(
    TSPPOOL_INFO *info,           // スプール情報が格納されている構造体のポインタ
    TSPPOOL_ERR_INFO *erinfo,     // S2F44 応答情報が格納されている構造体ポインタ
    DSHMSG *msg,                 // S2F44 メッセージを格納するメッセージ構造体のポインタ
    BYTE *buff,                 // S2F44 のテキスト格納バッファポインタ
    int buff_size                // buff のバイトサイズ
);
```

[.NET VB]

```
Function DshMakeS2F43Response (
    ByRef info As dsh_info.TSPPOOL_INFO,
    ByRef erinfo As dsh_info.TSPPOOL_ERR_INFO,
    ByRef msg As dshdr2.DSHMSG,
    ByRef buff As Byte,
    ByVal buff_size As Int32) As Int32
```

[.NET C#]

```
int DshMakeS2F43Response(
    ref TSPPOOL_INFO info,
    ref TSPPOOL_ERR_INFO erinfo,
    ref DSHMSG msg,
    byte[] buff,
    int buff_size );
```

(2) 引数

info

スプール情報が格納されている構造体のポインタです。

erinfo

S2F44 応答情報が格納されている構造体のポインタです。

msg

S2F44 応答メッセージ情報を格納するためのメッセージ構造体のポインタです。

buff

S2F44 応答メッセージのテキストを格納するためのバッファポインタです。

buff_size

buff のバイトサイズです。

(3) 戻り値

戻り値	意味
0	正常に生成できた。
(-1)	生成できなかった。(buff 領域不足)

(4) 説明

S2F43 に対する S2F44 応答メッセージを erinfo に含まれるスプール応答情報に従って作成します。
erinfo 内の strack - S2F44 の STRACK として設定します。



以下、erinfo内のerr_count分のエラー情報をメッセージ内にエンコードします。
エラー情報は、stream, strackとfunctionになります。

3.9.4 ユーザ作成ライブラリ関数

3.9.4.1 DshResponseS2F44() S2F44 スプール設定応答メッセージ

(1) 呼出書式

[C, C++]

```
API int APIX DshResponseS2F44(
    ID_TR trid, // DSHDR2 のトランザクション ID
    TSPPOOL_INFO *info, // スプール設定メッセージ情報格納領域のポインタ
    TSPPOOL_ERR_INFO *erinfo // S2F44 応答情報格納用構造体のポインタ
);
```

[.NET VB]

```
Function DshResponseS2F44 (
    ByVal trid As Int32,
    ByRef info As dsh_info.TSPPOOL_INFO,
    ByRef erinfo As dsh_info.TSPPOOL_ERR_INFO) As Int32
```

[.NET C#]

```
int DshResponseS2F44(
    uint trid,
    ref TSPPOOL_INFO info,
    ref TSPPOOL_ERR_INFO erinfo );
```

(2) 引数

trid

S2F43 信時に DSHEng4 から渡される DSHDR2 通信ドライバーのトランザクション管理のための ID です。

info

スプール設定モード情報が格納されている構造体のポインタです。

erinfo

送信する応答メッセージ S2F44 に含まれる情報を格納するための構造体領域のポインタを指定します。

(3) 戻り値

戻り値	意味
0	正常に送信できた。
(-1)	送信できなかった。

(4) 説明

スプール設定メッセージ S2F43 に対する応答メッセージを送信します。

本関数はユーザ作成ライブラリ DLL (dsh_ulib.dll) に含まれる関数ですが、ここでは DSHEng4 パッケージに標準的な関数として付属されているものです。(ユーザ独自による作成も可能です)

引数に指定されている TSPPOOL_ERR_INFO 構造体に含まれている情報から S2F44 メッセージを組み立て、その後、S2F44 メッセージを送信します。

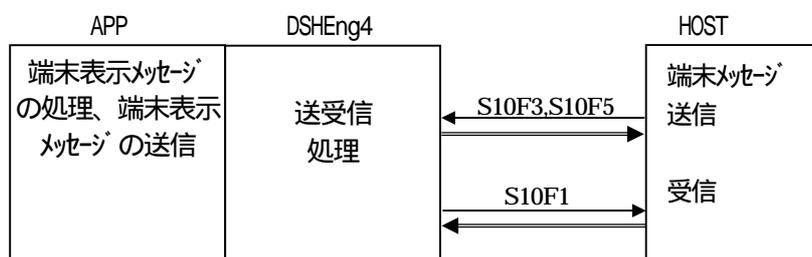


送信が完了したら、TSPool_ERR_INFO の構造体に使用されたメモリを DshFreeTASACCESS_ERR_INFO ()関数を使って開放します。

なお、S2F44 メッセージの組み立てに、DshMakeS2F44Response()関数を使用できます。

3.10 端末サービス情報関連関数

端末サービス関連メッセージとして、S10F1, S10F3, S10F5 があります。APP プログラムが関連する処理は、装置から送信されてくる端末表示メッセージの処理ならびに装置への端末表示要求になります。



(1) 端末メッセージ送信 API 関数

端末サービスのための API 関数名は次の一覧表のとおりです。

	API 関数名	機能
1	EngSendS10F1()	指定された端末要求を S10F1 で送信する。 HOST<-EQ

(2) ライブラリ関数

S10F3, S10F5 受信時に使用するための関数です。

	API 関数名	機能
2	DshDecodeS10F3()	S10F3 メッセージ内の端末テキスト情報をデコードするための関数です。
4	DshDecodeS10F5()	S10F5 メッセージ内の端末情報を TTERMTEXT_INFO 構造体にデコードします。
5	DshFreeTTERMTEXT_INFO()	S10F5 でデコードした情報格納に使用されたメモリを解放します。
6	DshCopyTTERMTEXT_INFO()	TTERMTEXT_INFO 構造体を別の構造体にコピーします。

(3) ユーザ作成ライブラリ関数

	ライブラリ関数名	機能
1	DshResponseS10F4()	S10F3 端末表示応答
2	DshResponseS10F6()	S10F5 端末表示(マルチ)応答

3.10.1 使用する情報格納構造体

S10F5 に含まれる複数ブロック端末表示情報をユーザが処理しやすい以下の TTERMTEXT_INFO 構造体の中にデコードします。

TTERMTEXT_INFO と、その内部で使用する他の構造体は下記のとおりです。

```
typedef struct{
    int    tid;                // Terminal ID
    int    text_count;        // # of text line
    char   **text_list;       // text line list
    int    ackc10;            // ackc10用(Work用)
}TTERMTEXT_INFO;           // terminal text
```

3.10.2 端末メッセージ送信関数

3.10.2.1 EngSendS10F1() 端末要求メッセージ送信関数

(1) 呼出書式

[C, C++]

```
API int APIX EngSendS10F1(
    int tid, // 端末 ID
    char *text, // 表示テキストが格納されているポインタ
    int *ackc10 // S10F4 内の ACKC10 格納ポインタ
    int (WINAPI *callback)(), // 実行終了時のコールバック関数
    ULONG upara // callback 時のパラメータ
);
```

[.NET VB]

```
Function EngSendS10F1 (
    ByVal tid As Int32,
    ByVal text As String,
    ByRef ackc10 As Int32,
    ByVal callback As vcallback.Callback_S10F1,
    ByVal upara As Int32) As Int32
```

[.NET C#]

```
int EngSendS10F1(
    int tid,
    byte[] text,
    ref int ackc10,
    CallbackS10F1 TerminalCallback,
    uint upara );
```

(2) 引数

tid

表示する装置の端末 ID です。

text

表示テキストが格納されている領域のポインタです。

ackc10

応答メッセージ S10F4 で得られた ACKC10 を格納するための領域のポインタです。

callback

DSHEng4 によるメッセージ送信処理が終了したときに呼出されるコールバック関数を指定します。

ユーザは任意の関数名を指定できます。

コールバックの指定が=0 の場合はブロックモードになります。

upara

コールバックされたときに引数で渡してもらうためのパラメータです。関数実行終了時にコールバックされた際、何かの判別情報として使用したい値、構造体ポインタなどを設定します。

(3) 戻り値

戻り値	意味
0	(1) ブロックモード : 正常に送信できた。

	ackc10 に応答結果が返却される。 (2) 非ブロック : 要求が受け付けられた。
9	キャンセルされた。
(-1)	送信できなかった。
(-14)	T3タイムアウトを検出した。
その他 >0	ブロックの場合、ackc10 の値が返却されます。

(4) 説明

装置側からホストに対し、端末からのテキスト情報を装置へ送信要求します。S10F1 メッセージで送信します。

端末 ID とテキストメッセージをそれぞれ引数 tid, text で指定します。

要求を受けた DSHEng4 は、端末 ID と表示テキストを S10F1 メッセージにエンコードし、装置に送信します。送信要求から S10F1 応答メッセージ受信までの制御は引数の S10F1 コールバック関数指定の有無によって次のようになります。

Callback の指定	制御の流れ
なし	S10F1 送信後、応答メッセージ S10F2 が受信されるか、またはエラーを検出するまで制御が要求元に戻ってきません。 正常終了であれば、ackc10 に応答結果が返却される。
あり	送信要求後、S10F1 の送信前に制御が戻されます。実行結果はコールバック関数で与えられます。 コールバック関数の end_status が=0 ならば ackc10 に応答結果が返却される。 エラーが検出された場合、(-1) が end_status にセットされます。

(5) コールバック関数

[c,C++]

```
API int APIX callback(
    int end_status,           // 実行結果
    int *ackc10,             // ACKC10 が格納されているポインタ
    ULONG upara              // 呼出時に指定したパラメータ
);
```

[.NET VB]

```
Function Callback_S10F1(ByVal end_status As Integer, ByRef ack10 As Integer, ByVal upara As Integer)
As Integer
```

[.NET C#]

```
int CallbackS10F1(int end_status, int* ack, uint upara);
```

end_status には以下の値が設定されます。

結果	意味
0	正常に送信できた。ackc10 に S10F4 の応答結果が返却される。
(-1)	送信エラーを検出した。
(-14)	T3タイムアウトを検出した。

end_status = 0 のときのみ ackc10 の内容が有効です。

3.10.3 端末表示関連ライブラリ関数

3.10.3.1 DshDecodeS10F3() - S10F3 端末メッセージのデコード

(1) 呼出書式

[C, C++]

```
API int APIX DshDecodeS10F3(
    DSHMSG *smsg,           // SECS メッセージ 情報構造体のポインタ
    int *tid,              // 端末 ID を格納する領域のポインタ
    char *text              // 端末テキスト格納用ポインタ
);
```

[.NET VB]

```
Function DshDecodeS10F3 (
    ByRef smsg As dshdr2.DSHMSG,
    ByRef tid As Int32,
    ByVal text As String) As Int32
```

[.NET C#]

```
int DshDecodeS10F3(
    ref DSHMSG smsg,
    ref int tid,
    byte[] text );
```

(2) 引数

msg

S10F3 の SECS メッセージ 情報が格納されている構造体のポインタです。

tid

S10F3 端末メッセージ情報に含まれる端末 ID を格納するための領域ポインタです。

text

S10F3 端末メッセージに含まれる 1 行分のテキストを格納するための領域ポインタです。
81 バイト以上の領域を指定してください。

(3) 戻り値

戻り値	意味
0	正常にデコードできた。
(-1)	msg を正しくデコードできなかった。

(4) 説明

S10F3 メッセージに含まれる端末メッセージ情報である端末 ID を tid に、そしてテキストを text で指定された領域にデコードして取り込みます。

3.10.3.2 DshDecodeS10F5() - S10F5 端末情報を TTERMTEXT_INFO 構造体にデコード

(1) 呼出書式

[C, C++]

```
API int APIX DshDecodeS10F5(
    DSHMSG *smsg, // SECSメッセージ情報構造体のポインタ
    TTERMTEXT_INFO *info // デコードしたい端末表示情報が格納されている構造体のポインタ
);
```

[.NET VB]

```
Function DshDecodeS10F5 (
    ByRef smsg As dshdr2.DSHMSG,
    ByRef pinfo As dsh_info.TTERMTEXT_INFO) As Int32
```

[.NET C#]

```
int DshDecodeS10F5(
    ref DSHMSG smsg,
    ref TTERMTEXT_INFO pinfo );
```

(2) 引数

smsg

デコードする S10F5 メッセージが格納されているメッセージ情報構造体のポインタです。

info

デコードした端末メッセージ情報を格納する構造体のポインタです。

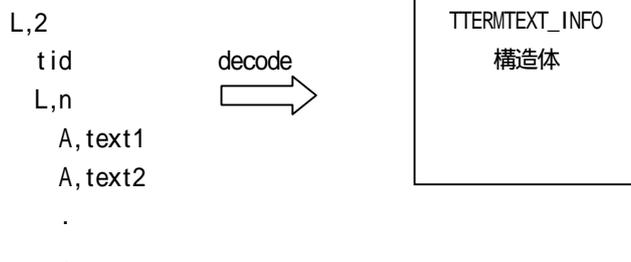
(3) 戻り値

戻り値	意味
0	正常にデコードできた。
(-1)	smsg を正しくデコードできなかった。

(4) 説明

smsg に格納されている S10F5 メッセージをデコードし、info で指定された TTERMTEXT_INFO 構造体領域に端末メッセージ情報を格納します。。

smsg S10F5



デコード後、info の処理が終了したら、DshFreeTTERMTEXT_INFO()関数を使って構造体内で使用されたメモリを開放してください。

3.10.3.3 DshFreeTTERMTEXT_INFO() - 端末表示情報構造体メモリの開放

(1) 呼出書式

[C, C++]

```
API void APIX DshFreeTTERMTEXT_INFO(  
    TTERMTEXT_INFO *info    // メモリを開放したい端末表示情報が格納されている構造体のポインタ  
);
```

[.NET VB]

```
Sub DshFreeTTERMTEXT_INFO (  
    ByRef info As dsh_info.TTERMTEXT_INFO)
```

[.NET C#]

```
void DshFreeTTERMTEXT_INFO(  
    ref TTERMTEXT_INFO info );
```

(2) 引数

pinfo

メモリを解放したい端末表示情報構造体のポインタです。

(3) 戻り値

なし。

(4) 説明

TTERMTEXT_INFO 構造体内で情報格納用に使用されているメモリを全て解放します。
開放した後、TTERMTEXT_INFO の内容を全て 0 で初期設定します。
info が NULL ならば、何も処理しません。

3.10.3.4 DshCopyTTERMTEXT_INFO() - 端末表示情報構造体メモリのコピー

(1) 呼出書式

[C, C++]

```
API int APIX DshCopyTTERMTEXT_INFO(
    TTERMTEXT_INFO *dinfo,           // 北°-先のポインタ
    TTERMTEXT_INFO *sinfo           // 北°-元のポインタ
);
```

[.NET VB]

```
Sub DshCopyTTERMTEXT_INFO (
    ByRef pinfo As IntPtr,
    ByRef info As dsh_info.TTERMTEXT_INFO)
```

[.NET C#]

```
void DshCopyTTERMTEXT_INFO(
    ref TTERMTEXT_INFO pinfo,
    ref TTERMTEXT_INFO info );
```

(2) 引数

pinfo

端末表示情報のコピー先構造体メモリのポインタです。

sinfo

コピー元の端末表示情報が格納されている構造体メモリのポインタです。

(3) 戻り値

戻り値	意味
0	正常に北°-できた。
(-1)	sinfo または dinfo の値が NULL だったので北°-できなかった。

(4) 説明

sinfo が指す TTERMTEXT_INFO 構造体内に格納されている端末表示情報を dinfo に指定された TTERMTEXT_INFO 構造体にコピーします。

dinfo 内のメンバーで新しいメモリが必要なものは本関数が取得します。

dinfo 内メンバーで確保されたメモリは、使用后、DshFreeTTERM_TEXT_INFO()関数を使って開放してください。

3.10.4 ユーザ作成ライブラリ関数

3.10.4.1 DshResponseS10F4() S10F4 端末表示応答メッセージ

(1) 呼出書式

[C, C++]

```
API int APIX DshResponseS10F4(
    ID_TR trid,           // DSHDR2 のトランザクション ID
    char *text,          // 端末表示メッセージ 格納ポインタ
    int ackc10           // S10F4 応答情報格納用構造体のポインタ
);
```

[.NET VB]

```
Function DshResponseS10F4 (
    ByVal trid As Int32,
    ByVal text As String,
    ByVal ackc7 As Int32) As Int32
```

[.NET C#]

```
int DshResponseS10F4(
    uint trid,
    byte[] text,
    int ackc7 );
```

(2) 引数

trid

S10F3 信時に DSHEng4 から渡される DSHDR2 通信ドライバーのトランザクション管理のための ID です。

text

端末表示の表示メッセージ (文字列) が格納されているポインタです。

ackc10

送信する応答メッセージ ack の値です。

(3) 戻り値

戻り値	意味
0	正常に送信できた。
(-1)	送信できなかった。

(4) 説明

端末表示メッセージ S10F3 に対する応答メッセージを送信します。

本関数はユーザ作成ライブラリ DLL(dsh_ulib.dll)に含まれる関数ですが、ここでは DSHEng4 パッケージに標準的な関数として付属されているものです。(ユーザ独自による作成も可能です)

引数に指定されている ackc10 情報から S10F4 メッセージを組み立て、その後、S10F4 メッセージを送信します。

3.10.4.2 DshResponseS10F6() S10F6 端末表示 (Multi-Block) 応答メッセージ

(1) 呼出書式

[C, C++]

```
API int APIX DshResponseS10F6(
    ID_TR trid,                // DSHDR2 のトランザクション ID
    TTERMTEXT_INFO *info,     // 端末表示 (Multi-Block) メッセージ 格納ポインタ
    int ackc10                 // S10F6 応答情報格納用構造体のポインタ
);
```

[.NET VB]

```
Function DshResponseS10F6 (
    ByVal trid As Int32,
    ByRef info As dsh_info.TTERMTEXT_INFO,
    ByVal ackc10 As Int32) As Int32
```

[.NET C#]

```
int DshResponseS10F6(
    uint trid,
    ref TTERMTEXT_INFO info,
    int ackc10 );
```

(2) 引数

trid

S10F5 信時に DSHEng4 から渡される DSHDR2 通信ドライバーのトランザクション管理のための ID です。

info

端末表示 (Multi-Block) の表示メッセージ情報が格納されているポインタです。

ackc10

送信する応答メッセージ ack の値です。

(3) 戻り値

戻り値	意味
0	正常に送信できた。
(-1)	送信できなかった。

(4) 説明

端末表示 (Multi-Block) メッセージ S10F5 に対する応答メッセージを送信します。

本関数はユーザ作成ライブラリ DLL (dsh_ulib.dll) に含まれる関数ですが、ここでは DSHEng4 パッケージに標準的な関数として付属されているものです。(ユーザ独自による作成も可能です)

引数に指定されている ackc10 情報から S10F6 メッセージを組み立て、その後、S10F6 メッセージを送信します。