DshGemMsgPro　GEM メッセージ・エンコード/デコード

ソフトウェア・ライブラリ

定数・構造体説明書

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

２０１３年９月

株式会社データマップ

文書番号　DshGemMsgPro-13-30311-00

［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.	2013 年 9 月	初版	

目　次

１．概要

本説明書は、SEMI GEM モデルに準拠する SECS-II メッセージのエンコード(Encoding)、デコード(Decoding)を行うために使用する DshGemMsgPro(以下、(GEM-PRO と呼びます) ライブラリが提供する API 関数が使用する定数と構造体について説明します。

プログラム言語として、c, C++, C#, VB.Net に対応しています。

ここで説明する定数と構造体の定義については、各言語別ファイルに記述されています。

c / C++ : DshGemMsgPro.h
C# : DshGemMsgPro.cs
VB.Net : DshGemMSgPro.vb

これらファイルは、DshGemMsgPro デモプログラムのソースファイルとし製品に同梱されます。

本説明書では、C,C++言語のための各構造体、定数について説明します。
他の言語、C#, VB.Net につては、本説明書と、上で述べたファイルをつきあわせて参照ください。

GEM-PRO の関連ドキュメントは次表のとおりです。

GEM-PRO ドキュメント一覧表

	文書番号	タイトル名と内容
1	DshGemMsgPro-13-30321-00 Vol-1	DshGemMsgPro GEM ﾒｯｾｰｼﾞ・ｴﾝｺｰﾄﾞ/ﾃﾞｺｰﾄﾞ API 関数説明書 1. 概要 2. 機能概略 3. API 関数 3.1 GEM-PRO 初期化関数とバージョン取得関数 3.2 S1Fx, S2Fx ﾒｯｾｰｼﾞｴﾝｺｰﾄﾞ・ﾃﾞｺｰﾄﾞ関数
	DshGemMsgPro-13-30322-00 Vol-2	(3.2) S3Fx,S5Fx, S6Fx, S7Fx
	DshGemMsgPro-13-30323-00 Vol-3	(3.2) S10Fx, S14Fx, S15Fx, S16Fx
2	DshGemMsgPro-13-30331-00 Vol-1	DshGemMsgPro GEM ﾒｯｾｰｼﾞ・ｴﾝｺｰﾄﾞ/ﾃﾞｺｰﾄﾞ LIB 関数説明書 ・変数(EC、SV、DVVAL)関連 ・ﾚﾎﾟｰﾄ、収集ｲﾍﾞﾝﾄ(CE)関連 ・ｱﾗｰﾑ関連 ・ﾌﾟﾛｾｽ・ﾌﾟﾛｸﾞﾗﾑ(PP、FPP)関連 ・ﾚｼﾋﾟ関連 ・ﾌﾟﾛｾｽ・ｼﾞｮﾌﾞ関連 ・ｺﾝﾄﾛｰﾙ・ｼﾞｮﾌﾞ関連
	DshGemMsgPro-13-30332-00 Vol-2	・ﾘﾓｰﾄｺﾝﾄﾛｰﾙ、拡張ﾘﾓｰﾄｺﾝﾄﾛｰﾙ関連 ・ｷｬﾘｱｱｸｼｮﾝ、ﾎﾟｰﾄ制御関連 ・端末表示関連 ・ｽﾌﾟｰﾙ関連 ・その他の汎用関数
3	DshGemMsgPro-13-30320-00	DshGemMsgPro GEM ﾒｯｾｰｼﾞ・ｴﾝｺｰﾄﾞ/ﾃﾞｺｰﾄﾞ 定数、構造体説明書
4	DshGemMsgPro-13-30381-00	DshGemMsgPro GEM ﾒｯｾｰｼﾞ・ｴﾝｺｰﾄﾞ/ﾃﾞｺｰﾄﾞ ﾃﾞﾓﾌﾟﾛｸﾞﾗﾑ説明書

GEM-PRO に関する概要、機能については、”GEM-PRO API 関数説明書-VOL-1 “の１，２章をを参照してください。

２．データタイプの定義

２．１　C,C++言語の基本データタイプ

C,　C++言語について、符号付でない(Unsigned)整数を別名で使用したり、変数 ID など整数を変数と分かるような名前で使用できるようにします。

具体的には、typedef　キーワードを使って既存のタイプに別のタイプ名を使って使用します。

（1）標準データタイプ

```
typedef char    CHAR;
typedef unsigned char BYTE;
typedef unsigned char UCHAR;
typedef short   SHORT;
typedef unsigned short USHORT;
typedef int     INT;
typedef unsigned int UINT;
typedef long    LONG;
typedef unsigned long ULONG;
```

（2）装置関連データタイプ

```
typedef ULONG   TVID;               // VID data type
typedef ULONG   TECID;              // ECID
typedef ULONG   TSVID;              // SVID
typedef ULONG   TDVID;              // DVVALID data type

typedef ULONG   TCEID;              // CEID
typedef ULONG   TRPID;              // Report ID

typedef BYTE    TALCD;              // ALCD
typedef ULONG   TALID;              // ALID
typedef ULONG   TDATAID;            // DATAID

typedef ULONG   TDATALENGTH;        // DATALENGTH(mulit-block)
typedef ULONG   TLENGTH;            // LENGTH(mulit-block S7F29)
typedef int     TALED;              // Alarm Enable
typedef int     TPTN;               // Loader Port number
typedef int     TLIMITID;           // VID limit ID

typedef BYTE    *TPPID;             // Process Program ID
typedef BYTE    *TTRID;             // Trace ID
```

３．カテゴリ別構造体と定数

３．１　変数（EC,SV）関連構造体

EC, SV, DVVAL 情報に関連する構造体について説明します。
SV トレース、変数リミット関連の構造体についても説明します。

３．１．１　TVID_LIST - 変数 ID リスト構造体 – S1F3, S1F11, S2F13

```
typedef struct{
     int      count;                    // ID 数
     int      max_count;
      TVID    *list;                    // ID ﾘｽﾄ
} TVID_LIST;
```

３．１．２　TV_VALUE, TV_VALUE_LIST　– 変数値格納構造体 – S1F4, S2F13

```
typedef struct{
        TVID        vid;               // 変数 ID
        int         format;            // Format( ICODE_A, B, ...)
        int         asize;             // 配列ｻｲｽﾞ
        void        *value;            // 変数値格納ﾎﾟｲﾝﾀ
} TV_VALUE;

typedef struct{
      int         count;               // 値数
      TV_VALUE    **vv_list;           // 値格納領域ﾎﾟｲﾝﾀ ﾘｽﾄ
} TV_VALUE_LIST;
```

３．１．３　TSV_NAME,　TSV_NAME_LIST – 状態変数名格納構造体 – S1F12

```
typedef struct{
        TSVID       svid;              // SVID
        char        *name;             // 変数名
        char        *units;            // 物理単位
} TSV_NAME;

typedef struct{
        int         count;             // SV 数
        TSV_NAME    **name_list;       // SV 名ﾘｽﾄ
} TSV_NAME_LIST;
```

３．１．４　TEC_NAME,　TEC_NAME_LIST - 装置定数名格納構造体 – S2F30

```
typedef struct{
        TECID       ecid;               // 定数 ID
        char        *name;              // 名前
        int         format;             // 値のﾌｫｰﾏｯﾄ
        int         asize;              // 配列ｻｲｽﾞ
        void        *ecmin;             // 最小値
        void        *ecmax;             // 最大値
        void        *ecdef;             // 初期値
        char        *units;             // 単位名
} TEC_NAME;

typedef struct{
        int         count;              // EC 数
        TEC_NAME    **name_list;        // EC 名ﾘｽﾄ
} TEC_NAME_LIST;
```

３．１．５　TTRACE_INFO　－ SV トレース設定関連　－ S2F23

```
typedef struct{
        char    *name;                  // trace name
        char    *trid;                  // trace id
        int     format;                 // trace id format (=ICODE_A)
        int     asize;                  // trace id array size
        int     max_asize;
        char    *dsper;                 // trace 時間周期
        int     dsper_time;             // trace 時間周期-数値
        int     totsmp;                 // total sample 数
        int     tot_fmt;                        // (未使用)
        int     tot_asize;              // (未使用)
        int     repgsz;                 // report group size
        int     gsz_fmt;                // (未使用)
        int     gsz_asize;              // (未使用)
        int     svid_count;             // svid list の size
        TSVID   *svid_list;             // svid list
} TTRACE_INFO;
```

３．１．６　TTRACE_SV、TTRACE_DATA　－ SV トレースデータ関連　－ S6F1

```
typedef struct{
        int         format;             // data item code
        int         asize;              // array size
        void        *sv;                // status data value
} TTRACE_SV;

//
typedef struct{
        void        *trid;              // trace id
        int         format;             // trace id format
        int         asize;              // trace id array size
        int         smpln;              // sampling no.
        char        *stime;             // start time
        int         count;              // no. of data
        TTRACE_SV   **sv_list;          // sv ptr list
} TTRACE_DATA;
```

3．1．7　TLIMIT_INFO, TLIMIT_LIST 構造体 - S2F45

```
typedef struct{
        int         vid_count;
        TLIMIT_INFO **limit_list;         // list
} TLIMIT_LIST;

typedef struct{
        TDVID       vid;
        int         limit_count;
        TLIMIT_ID_INFO **limitid_list;
        int         format;               // for upperdb & lowerdb の fotmat
        int         asize;                //   〃         〃   〃
} TLIMIT_INFO;

typedef struct{
        TLIMITID    limit_id;             // limit id
        void        *upperdb;             // ﾃﾞｯﾄﾞﾊﾞﾝﾄﾞ上限
        void        *lowerdb;             //   〃         下限
} TLIMIT_ID_INFO;
```

3．1．8　LIMIT_ERR_INFO, TLIMIT_ERR_LIST 構造体 - S2F46

```
typedef struct{
        int         vlaack;
        int         err_count;
        TLIMIT_ERR_INFO **limit_list;
} TLIMIT_ERR_LIST;


typedef struct{
        TDVID       vid;
        int         lvack;
        int         lmt_count;
        TLIMITID    *limit_id;
        int         *limitack;           // B
} TLIMIT_ERR_INFO;
```

3．1．9 TLIMIT_RSP_INFO, TLIMIT_RSP_LIST, TVLIMIT_EVENT_INFO – S2F48

```
typedef struct{
        TVID        vid;
        char        *units;
        int         format;              // for upperdb & lowerdb
        int         asize;               //   ″         ″   ″
        void        *limit_min;
        void        *limit_max;
        int         limit_count;
        TLIMIT_ID_INFO **limitid_list;    // (2.2.1.6 参照)
} TLIMIT_RSP_INFO;

typedef struct{
        int         vid_count;
        TLIMIT_RSP_INFO **limit_list;
} TLIMIT_RSP_LIST;

typedef struct{
        TVID    vid;                    // vid
        char    *value;                 // value in ascii
        int     limitid;                // limit id
        int     dir;                    // transient direction (0=up, 1=down)
}TVLIMIT_EVENT_INFO;
```

３．２　収集イベント(CE)，　レポート関連構造体

収集イベント、レポートとそれらにリンクする情報に関連する

３．２．１　TCE_INFO　－ CE 情報保存構造体

```
typedef struct{
        TCEID   ceid;
        char    *name;
        int     ceed;           // enable=1, disable=0
        int     rp_count;       // link されている report ID 数
        char    **rpname;
        TRPID   *rpid;          // report ID list
} TCE_INFO;
```

３．２．２　TCE_CONTENT, TRP_CONTENT、TV_CONTENT

```
typedef struct{
        TVID        vid;
        int         format;
        int         asize;
        void        *dptr;                  // fmt L ならば nesting ->TV_CONTENT
} TV_CONTENT;

typedef struct{
        TRPID       rpid;
        int         v_count;
        TV_CONTENT  **v_list;               // RPID にﾘﾝｸされている変数情報のﾘｽﾄ
} TRP_CONTENT;

typedef struct{
        TCEID       ceid;
        int         rp_count;
        TRP_CONTENT **rp_list;              // CEID にリンクされているﾚﾎﾟｰﾄ情報のﾘｽﾄ
        int         next_rp;
        int         next_v;
} TCE_CONTENT;
```

３．２．３　TRP_LINK、TRP_LIST リンク情報　－ S2F33

```
typedef struct{
        TRPID       rpid;               // ﾚﾎﾟｰﾄ ID
        int         count;              // 保存変数 ID の数
        TVID        *vid_list;          // ﾘﾝｸしている変数 ID ﾘｽﾄ
} TRP_LINK;

typedef struct{
        int         count;
        TRP_LINK    **rp_list;
} TRP_LIST;
```

３．２．４　TS6F11_V_INFO,　TS6F11_RP_INFO,　TS6F11_CE_INFO － S6F11, S6F16, S6F20

```
typedef struct{
     TVID      vid;                     // 変数 ID
     int       format;
     int       asize;
     void      *value;                  // 変数値格納ﾎﾟｲﾝﾀ
     void      **link;                  // format-L, LINK TS6F11_V_INFO を指す (Nesting)
} TS6F11_V_INFO;

typedef struct{
     TRPID     rpid;                    // ﾚﾎﾟｰﾄ ID
     int       v_count;                 // ﾘﾝｸされている変数 ID 数
     TS6F11_V_INFO  **v_list;           // 変数情報ﾘｽﾄ
} TS6F11_RP_INFO;

typedef struct{
     TCEID     ceid;                    // CEID
     int       rp_count;                // ﾘﾝｸされているﾚﾎﾟｰﾄ ID 数
     TS6F11_RP_INFO ** rp_list;         // ﾚﾎﾟｰﾄ情報ﾘｽﾄ
} TS6F11_CE_INFO;
```

３．２．５　TCE_LINK, TCE_LIST　－ CE リンク情報 – S2F35

```
typedef struct{
        TCEID       ceid;               // CEID
        int         count;              // ﾘﾝｸされている RPID 数
        TRPID       *rpid_list;         // ﾘﾝｸされているﾚﾎﾟｰﾄ ID ﾘｽﾄ
} TCE_LINK;

typedef struct{
        int         count;              // CEID の数
        TCE_LINK    **ce_list;          // CEID のﾘｽﾄ
} TCE_LIST;
```

３．３　アラーム関連構造体

３．３．１　TAL_S5F1_INFO、,TAL_S5F6_INFO 構造体 – S5F1, S5F6

```
typedef struct{
        int         on_off;             // 発生/復旧 (1/0)
        TALID       alid;               // ALID
        TALCD       alcd;               // ALCD
        char        *altx;              // ALTX
} TAL_S5F1_INFO;

typedef struct{
        int         count;              // ALID の数
        TAL_S5F1_INFO    **al_list;     // ALARM 情報ﾘｽﾄ (on_off は使用しない)
} TAL_S5F6_LIST;
```

３．４　プロセス・プログラム（PP）関連構造体

３．４．１　TPP_INFO PP 情報保存構造体 - S7F3, S7F5

```
typedef struct{
        char        *ppid;
        char        *ppbody;
}TPP_INFO;                              // process program
```

３．４．２　TPINQ_INFO　PP ﾛｰﾄﾞ問合せ情報保存構造体 – S7F1

```
typedef struct{
        char        *ppid;              // ppid
        int         length;             // length
} TPPINQ_INFO;
```

３．４．３　TPPID_LIST PPID リスト保存構造体 – S7F17, S7F20

```
typedef struct{
        int         count;
        char        **ppid_list;
} TPPID_LIST;
```

３．４．４　PP 妥当性確認結果情報構造体 – S7F27

```
typedef struct{
        char        *ppid;
        int         err_count;
        TPP_PVS_INFO **err_list;
} TPP_PVS_LIST;

typedef struct{
        int         ackc7a;
        int         seqnum;
        char        *errw7;
} TPP_PVS_INFO;
```

３．５　書式付プロセス・プログラム（FPP）関連構造体

３．５．１　TS7F23_INFO, TFPP_CCODE, TFPP_PARA - TFPP 情報保存構造体 - S7F23, S7F25

```
typedef struct{
        int         ppara_fmt;
        int         ppara_size;
        int         max_ppara_size;
        void        *ppara;
} TFPP_PARA;

typedef struct{
        int         ccode_fmt;          // CCODE format
        int         ccode_size;
        int         max_ccode_size;
        void        *ccode;             // command code
        int         ppara_count;        // # of para count
        TFPP_PARA   **ppara_list;       // parameter list
} TFPP_CCODE;

typedef struct{
        char        *ppid;
        char        *mdln;              // MDLN
        char        *softrev;           // SOFTREV
        int         ccode_count;        // # of process commands
        TFPP_CCODE  **ccode_list;       // ｺﾏﾝﾄﾞｺｰﾄﾞ情報ﾘｽﾄ
}TS7F23_INFO;                           // formatted process program
```

３．６　レシピ(RECIPE)関連構造体

３．６．１　TRCP_INFO - レシピ情報保存構造体 – S15F13

```
typedef struct{
        char        *rcpparnm;          // para name
        int         par_fmt;
        int         par_size;
        void        *rcpparval;         // para value;
}TRCP_PARA;                             // Recipe Parameter

typedef struct{
        char        *rcpspec;           // ﾚｼﾋﾟ ID
        int         para_count;         // # of pparameter
        TRCP_PARA   **para_list;
        char        *rcpbody;
}TRCP_INFO;                             // Recipe Information
```

３．６．２　TRCP_ERR_INFO　- レシピ応答情報構造体 – S15F4, S15F6, S15F14, S15F18

```
typedef struct{
        int         rmack;              // RMACK (U1)
        int         err_count;          // ｴﾗｰ数
        TERR_INFO   **err_list;         // ｴﾗｰ情報ﾘｽﾄ
} TRCP_ERR_INFO;
```

３．６．３　TRCP_ACT_INFO – レシピ・アクション情報構造体 - S15F3

```
typedef struct{
        char        *rmnsspec;          // rcpid
        int         rmnscmd;            // action command 1=create,5=delete
}TRCP_ACT_INFO;                         // Recipe Action
```

３．６．４　TRCP_RENAME_INFO – レシピ・リネーム情報構造体 – S15F5

```
typedef struct{
        char        *rmnsspec;          // rcpid
        char        *rmnewns;           // new rcpid name
}TRCP_RENAME_INFO;              // Recipe Rename
```

３．６．５　TRCP_S15F8_INFO – レシピスペース応答情報 – S15F8

```
typedef struct{
        ULONG       rmspace;            // ｽﾍﾟｰｽｻｲｽﾞ(ﾊﾞｲﾄ)
        int         rmack;              // U1
        int         err_count;
        TERR_INFO   **err_list;
} TRCP_S15F8_INFO;
```

3．6．6　TRCP_S15F10_INFO – レシピ ステータスデータ情報 – S15F10

```
typedef struct{
        int         rcpstat;            // status U1
        char        *rcpver;            // version A
        int         rmack;              // U1
        int         err_count;
        TERR_INFO   **err_list;
} TRCP_S15F10_INFO;
```

3．6．7　TRCP_RETRIEVE_INFO – レシピ検索要求情報 – S15F17

```
typedef struct{
        char        *rcpspec;           // rcpid
        int         seccode;            // sec code
 }TRCP_RETRIEVE_INFO;
```

3．6．8　TRCP_S15F18_INFO – レシピ検索要求応答情報 – S15F18

```
typedef struct{
        int         q_count;                // ++ 1, 2 or 3
        int         r_count;
        TRCP_SECNM  *m_secnm;               // ++
        char        *rcpbody;
        int         s_count;
        TRCP_SECNM  **secnm_list;
        int         rmack;
        int         err_count;
        TERR_INFO   **err_list;
} TRCP_S15F18_INFO;

typedef struct{
        char        *rcpsecnm;
        int         attr_count;
        TRCP_ATTR   **attr_list;
} TRCP_SECNM;

typedef struct{
        char        *attrid;
        int         format;
        int         asize;
        void        *attrdata;
} TRCP_ATTR;
```

３．７　キャリア情報関連構造体

キャリア情報は、S16F11、S16F15 で使用される TPRJ_INFO 構造体の中で使用されます。

３．７．１　TCAR_INFO, TSLOT_INFO – キャリア、スロット情報構造体

```
typedef struct{
        int         capacity;
        char        *usage;
        char        *carid;              // ｷｬﾘｱ ID
        int         map_status;
        int         id_status;
        int         acc_status;
        char        *location;
        int         slot_count;          // slot 数
        TSLOT_INFO  **slot_list;         // slot ID list
} TCAR_INFO;

typedef struct{
        int         status;
        int         slotid;            // U1
        char        *mid;
        char        *substid;
        char        *substloc;
} TSLOT_INFO;
```

３．８　コントロール・ジョブ情報関連構造体

S14F9, S14F11 メッセージに関する構造体に説明します。
最初に使用する定数を示します。

（1）オブジェクト・タイプ

```
#define EN_ControlJob          0          // これを使用します
```

（2）コントロール・ジョブの属性インデクス

```
#define EN_ObjID               0
#define EN_CarrierInputSpec    1
#define EN_CurrentPRJob        2
#define EN_DataCollectionPlan  3
#define EN_MtrlOutByStatus     4
#define EN_MtrlOutSpec         5
#define EN_PauseEvent          6
#define EN_ProcessingCtrlSpec  7
#define EN_ProcessingOrderMgmt 8
#define EN_PRJobStatusList     9
#define EN_StartMethod         10
#define EN_State               11
```

３．８．１　TCJ_INFO　－ コントロール・ジョブ情報構造体 － S14F9, S14F11

```
typedef struct{
        int             objspec_flag;     // (内部で使用)
        char            *objspec;         // ｵﾌﾞｼﾞｪｸﾄ・ｽﾍﾟｯｸ
        int             objtype_flag;     // (内部で使用)
        char            *objtype;         // ｵﾌﾞｼﾞｪｸﾄ・ﾀｲﾌﾟ　(="ControlJob")
        char            *objid;           // CJID
        int             attr_count;       // 属性数
        TOBJ_ATTR_INFO  **attr_list;      // 属性情報構造体のﾘｽﾄ
} TCJ_INFO;
```

３．８．２　TOBJ_ATTR_INFO - 属性情報構造体 － S14F9, S14F11

オブジェクト(CJ)の属性情報を保存します。TCJ_INFO の attr_list に含める各属性情報をこの TOBJ_ATTR_INFO に格納します。

```
typedef struct{
        char            *attrid;          // 属性 ID
        int             attrid_index;     // 属性 ID ｲﾝﾃﾞｸｽ
        void            *attrdata;        // 属性ﾃﾞｰﾀまたは構造体のﾎﾟｲﾝﾀ
} TOBJ_ATTR_INFO;
```

（1）３．９-（2）で挙げた属性インデクスに示す分の属性情報があります。

（2）以下の属性 INDEX について、属性情報の構造体を TVOID_LIST 構造体に保存します。
そして、attrdata には、TVOID_LIST 構造体のポインタを設定します。

EN_MtrlOutByStatus
EN_MtrlOutSpec
EN_ProcessingCtrlSpec

以下、各属性情報の格納に使用する構造体について説明します。

3．8．2．1　TVOIFD_LIST

```
typedef struct{
        int         count;
        void        **void_list;
} TVOID_LIST;
```

次の３種類の属性情報のコンテナーとして使用されます。

構造体	属性インデクス
TMTRL_OUT_STAT	EN_MtrlOutByStatus
TMTRL_OUT_SPEC	EN_MtrlOutSpec
TCTRL_SPEC	EN_ProcessingCtrlSpec

3．8．2．2　TMTRL_OUT_STAT - 属性 ID = "MtrlOutByStatus"

```
typedef struct{
        int         mtrl_status;          // U1
        char        *carid;               //
        int         slot_count;
        int         *slotid_list;
} TMTRL_OUT_STAT;
```

3．8．2．3　TMTRL_OUT_SPEC　-　属性 ID = "MtrlOutSpec"

```
typedef struct{
        char        *src_carid;
        int         src_slot_count;
        int         *src_slotid_list;
        char        *dst_carid;
        int         dst_slot_count;
        int         *dst_slotid_list;
} TMTRL_OUT_SPEC;
```

3．8．2．4 TCTRL_SPEC、TCTRL_RULE, TOUT_RULE - 属性 ID = "ProcessingCtrlSpec"

```
typedef struct{
        char        *prjobid;
        int         ctrl_rule_count;
        TCTRL_RULE  **ctrl_rule_list;
        int         out_rule_count;
        TOUT_RULE   **out_rule_list;
} TCTRL_SPEC;

typedef struct{
        char    *name;
        int     fmt;
        int     asize;
        void    *value;
} TCTRL_RULE;

typedef struct{
        int     status;                 // u1
        int     fmt;
        int     asize;
        void    *value;
} TOUT_RULE;
```

3．8．2．5 TPRJ_STATE_LIST - 属性 ID = "PRJobStatusList"

```
typedef struct{
        int     prj_count;
        char    **prj_list;
        int     *state_list;                        // U1
} TPRJ_STATE_LIST;
```

3．8．2．6 TPAUSE_EVETN% - 属性 ID = "PauseEvent"

```
typedef struct{
        int     ce_count;
        int     *ceid_list;
} TPAUSE_EVENT;
```

３．８．２．７　TCJ_TEXT_INFO

```
typedef struct{
        int         text_count;
        char        **text_list;
} TCJ_TEXT_INFO;
```

次の２種類の属性情報格納のために使用されます。

属性インデクス	属性 ID
EN_CarrierInput	"CarrierInputSpec"
EN_CurrentPRJob	"CurrentPRJob"

３．８．３　TOBJ_S14_ERR_INFO CJ - 応答情報 - S14F10, S14F12

TOBJ_ERR_INFO と TOBJ_S14_ERR_INFO の２種類ありますが、それぞれ以下のように使用します。

・TOBJ_ERR_INFO　　：　S14F9 を受信した側、S14F10 応答情報を作成するに当たり、TCJ_INFO 構造体と共に応答メッセージを作成するために使用します。

・TOBJ_S14_ERR_INFO　　：　S14F10 を受信した側がデコードする際に使用します。

```
typedef struct{                         // S14F10, 12 を送信する側が使用する。
        int         objack;
        int         err_count;
        TERR_INFO   **err_list;
} TOBJ_ERR_INFO;

typedef struct{                         // S14F10, 12 を受信した側が使用する。
        char        *objspec;           // cjid
        int         attr_count;         // S14F9 or S14F11 で得られた属性数
        TOBJ_ATTR_INFO  **attr_list;    //   〃                    属性
        int         objack;
        int         err_count;
        TERR_INFO   **err_list;
} TOBJ_S14_ERR_INFO;
```

3．8．4　CJ コマンド情報関連構造体

CJ コマンドのコマンド・インデクスとして、以下の定数を使用します。

```
#define CJ_Start            1
#define CJ_Pause            2
#define CJ_Resume           3
#define CJ_Cancel           4
#define CJ_Deselect         5
#define CJ_Stop             6
#define CJ_Abort            7
#define CJ_CJHOQ            8
```

3．8．4．1　TCJ_CMD_INFO　- CJ コマンド情報構造体 - S16F27

```
typedef struct{
        char        *ctljobid;          // CJID
        int         cmd;                // U1
        TCMD_PARA   *cp_info;           // parameter (1 個)
} TCJ_CMD_INFO;

typedef struct{
        char        *cpname;            // cpname
        int         cpval_fmt;          // cpval item fmt
        int         cpval_size;         // cpval data array size
        void        *cpval;             // cpval
}TCMD_PARA;
```

3．8．4．2　TCJ_CMD_ERR_INFO - S16F28

```
typedef struct{
        int         acka;
        TERR_INFO   *err_info;
} TCJ_CMD_ERR_INFO;
```

３．９　プロセス・ジョブ情報関連構造体

定数として、プロセス・ジョブの状態を示す以下のものが定義されています。
これらは、S16F20 などで使用される。

```
[ Process Job State]
        #define PRST_QUEUED             0
        #define PRST_SETTING_UP         1
        #define PRST_WAITING_FOR_START  2
        #define PRST_PROCESSING         3
        #define PRST_PROCESS_COMPLETE   4
        #define PRST_RESERVED           5
        #define PRST_PAUSING            6
        #define PRST_PAUSED             7
        #define PRST_STOPPING           8
        #define PRST_ABORTING           9
```

３．９．１　TPRJ_INFO - プロセス・ジョブ情報構造体 – S16F11, S16F15

```
typedef struct{
        char        *prjobid;           // ﾌﾟﾛｾｽ・ｼﾞｮﾌﾞ ID
        int         mf;
        int         car_count;          // mf=13 のとき
        TCAR_INFO   **car_list;         // ｷｬﾘｱﾘｽﾄ情報
        int         mid_count;          // mf=14 のとき
        char        **mid_list;
        int         prrecipemethod;     // fmt=51(8) U1
        TRCP_INFO   *rcp_info;          // ﾚｼﾋﾟ情報
        int         prprocessstart;     // fmt 11(8) Bool 1=auto,0=man
        int         ceid_count;
        TCEID       *pause_ceid_list;
} TPRJ_INFO;
```

３．９．２　TPRJ_LIST - プロセス・ジョブ情報リスト構造体 – S16F15

```
typedef struct{
        int         prj_count;          // PRJ 数
        TPRJ_INFO   **prj_list;         // PRJ_INFO ﾘｽﾄ
        int         err_count;          // (内部処理用)
        TERR_INFO   **err_list;         // (内部処理用)
} TPRJ_LIST;
```

3．9．3　TPRJ_ERR_INFO - 応答情報 – S16F12, S16F16

```
typedef struct{
        int         prj_count;          // ﾌﾟﾛｾｽ・ｼﾞｮﾌﾞ ID数
        char        **prj_list;         // ﾌﾟﾛｾｽ・ｼﾞｮﾌﾞ IDﾘｽﾄ
        int         acka;               // Boolean
        int         err_count;
        TERR_INFO   **err_list;
} TPRJ_ERR_INFO;
```

3．9．4　TPRJ_CMD_INFO　プロセス・ジョブ・コマンド情報構造体 – S16F5

使用できるコマンドは以下の通りです。

```
        "ABORT",
        "STOP",
        "CANCEL",
        "PAUSE",
        "RESUME"

typedef struct{
        char        *prjobid;           // ﾌﾟﾛｾｽ・ｼﾞｮﾌﾞ ID
        char        *cmd;               // Command
        int         cmd_index;          // (未使用)
        int         cp_count;
        TCMD_PARA   **cp_list;          // (3.8.4.1参照)
} TPRJ_CMD_INFO;
```

3．9．5　TPRJ_CMD_ERR_INFO - プロセス・コマンド応答 -　S16F6

```
typedef struct{
        char        *prjobid;           // ﾌﾟﾛｾｽ・ｼﾞｮﾌﾞ ID
        int         acka;               // Boolean
        int         err_count;
        TERR_INFO   **err_list;
} TPRJ_CMD_ERR_INFO;
```

3．9．6　TPRJ_DEQ_INFO　プロセス・ジョブ削除 - S16F17

```
typedef struct{
        int         prj_count;          // 削除 ID 数
        char        **prj_list;         // ﾌﾟﾛｾｽ・ｼﾞｮﾌﾞ ID ﾘｽﾄ
} TPRJ_DEQ_INFO;
```

3．9．7　TPRJ_DEQ_INFO　プロセス・ジョブ削除応答 - S16F18

```
typedef struct{
        int         prj_count;          // # of job dequed
        char        **prj_list;         // ﾌﾟﾛｾｽ・ｼﾞｮﾌﾞ ID ﾘｽﾄ
        int         acka;               // Boolean
        int         err_count;
        TERR_INFO   **err_list;
} TPRJ_DEQ_ERR_INFO;
```

3．9．8　TPRJ_STATE_TAB, TPRJ_STATE - プロセス・ジョブ状態情報 – S16F20

```
typedef struct{
        int     count;                  // PRJ 数
        TPRJ_STATE    **prj_state_list;   // PRJID, STATE list
} TPRJ_STATE_TAB;

typedef struct{
        char    *prjobid;               // ﾌﾟﾛｾｽ・ｼﾞｮﾌﾞ ID
        int     state;                  // 状態
}TPRJ_STATE;
```

3．10　端末表示情報構造体

S10F5 複数行テキストの表示用の構造体があります。

3．10．1　TTERMTEXT_INFO - 複数行テキスト保存構造体 -　S15F5

```
typedef struct{
        int     tid;                    // 端末 ID
        int     text_count;             // # of text
        char    **text_list;            // 表示ﾃｷｽﾄ文字列 text
}TTERMTEXT_INFO;
```

３．11　ホスト・コマンド情報構造体

３．11．１　TRCMD_INFO – ホスト・コマンド情報構造体 – S2F41

```
typedef struct{
        char        *rcmd;              // rcmd
        int         cp_count;           // parameter count
        TRCMD_PARA  **cp_list;          // paramete list
}TRCMD_INFO;

typedef struct{
    char        *cpname;                // cpname
    int         cpval_fmt;              // cpval item fmt
    int         cpval_size;             // cpval data array size
    void        *cpval;                 // cpval
}TRCMD_PARA;
```

３．11．２　TRCMD_HERR_INFO – ホスト・コマンド応答情報構造体 – S2F42

```
typedef struct{
        int         hcack;              // B
        int         err_count;
        char        **cpname_list;      // cpname
        int         *cpack_list;
} TRCMD_HERR_INFO;
```

３．12　拡張リモート・コマンド情報構造体

３．12．１　拡張リモート・コマンド情報構造体 – S2F49

```
typedef struct{
        char        *objspec;           // object spec
        char        *rcmd;              // rcmd
        int         cp_count;           // parameter count
        TERCMD_PARA **cp_list;          // paramete list
}TERCMD_INFO;

typedef struct tercmd_para{
        char        *cpname;            // cpname
        int         cpx_count;          // > 0 cpx_list =NULL ならばnesting なし
        struct tercmd_para **cpx_list;  // nesting  cpx_count > 0 の場合有効
        int         *cpval_fmt;         // cpval item fmt
        int         *cpval_size;        // cpval data array size
        void        **cpval;            // cpval
}TERCMD_PARA;
```

３．12．２　TERCMD_ERR_INFO – 拡張リモート・コマンド応答情報構造体 – S2F50

```
typedef struct{
        int         hcack;              // B
        int         err_count;
        char        **cpname_list;      // cpname
        int         *cepack_list;
} TERCMD_ERR_INFO;
```

３．13　キャリア・アクション

S3F17 メッセージの中に、アクション名と属性名が出てきますが、GEM-PRO のライブラリ関数では、それぞれの名前（文字列）ではなく、対応するインデクス値（整数）を関連ライブラリ関数の引数として渡すようになっています。
実際には、インデクス値として、その記号を使用します。

以下、それぞれのインデクスの記号と値は以下の通りです。

（1）キャリア・アクション名のためのインデクス

```
#define CA_Bind                         0
#define CA_CancelBind                   1
#define CA_CancelCarrier                2
#define CA_CancelCarrierAtPort          3
#define CA_CancelCarrierNotification    4
#define CA_CancelCarrierOut             5
#define CA_CarrierIn                    6
#define CA_CarrierNotification          7
#define CA_CarrierOut                   8
#define CA_CarrierReCreate              9
#define CA_CarrierRelease               10
#define CA_ProceedWithCarrier           11
```

（2）同属性名のためのインデクス

```
#define CA_ObjType                      0
#define CA_ObjId                        1
#define CA_Capacity                     2
#define CA_CarrierAccessingStatus       3
#define CA_CarrierIDStatus              4
#define CA_ContentMap                   5
#define CA_LocationID                   6
#define CA_SlotMap                      7
#define CA_SlotMapStatus                8
#define CA_SubStrateCount               9
#define CA_Usage                        10
```

3．13. 1　TCACT_INFO、TCACT_PARA　情報構造体 – S3F17

```
typedef struct{
        TDATAID     dataid;             // (未使用)
        char        *caction;           // action ｺﾏﾝﾄﾞ名
        int         action_index;
        char        *carspec;           // carrier spec ( carid )
        int         ptn;                // port no.
        int         cp_count;           // parameter count
        TCACT_PARA  **cp_list;          // paramete list
}TCACT_INFO;

typedef struct{
        char        *cattrid;           // attrid
        int         attr_index;
        void        *cattrdata;         // attrdata
}TCACT_PARA;
```

3．13. 2　TCACT_SLOT_INFO - スロット情報構造体 – 属性名　= "SlotMap"

```
typedef struct{
        int         count;
        int         *slot_list;
} TCACT_SLOT_INFO;
```

3．13. 3　TCACT_CONTENT - キャリア内容情報

```
typedef struct{
        int         count;              // lot or substrate count
        char        **lotid;            // lotid
        char        **substid;          // substrate id
}TCACT_CONTENT;
```

3．13. 4　TCACT_ERR_INFO - キャリア・アクション応答情報構造体　– S3F18, S3F23, S3F25

```
typedef struct{
        int         caack;
        int         err_count;
        TERR_INFO   **err_list;
} TCACT_ERR_INFO;
```

３．14　ポート・アクションとアクセス・モード

３．14．１　TPORTG_INFO、TPORTG_PARA - ポート・グループ・アクション情報構造体 – S3F23

```
typedef struct{
        char        *portgrpaction;     // group action
        char        *portgrpname;       // port group name
        int         pn_count;           // parameter count
        TPORTG_PARA **pn_list;          // paramete list
}TPORTG_INFO;

typedef struct{
        char        *paramname;         // paramname
        int         pval_fmt;           // paramval  item fmt
        int         pval_size;          // paramval  data array size
        void        *paramval;          // paramval data
}TPORTG_PARA;
```

３．14．２　TPORT_INFO、TPORT_PARA - ポート・アクション情報構造体 – S3F25

```
typedef struct{
        char        *portaction;        // port action
        int         ptn;
        int         pn_count;           // parameter count
        TPORT_PARA  **pn_list;          // paramete list
}TPORT_INFO;

typedef struct{
        char        *paramname;         // paramname
        int         pval_fmt;           // paramval  item fmt
        int         pval_size;          // paramval  data array size
        void        *paramval;          // paramval data
}TPORT_PARA;
```

３．14．３　TACCESS_INFO - ポート・アクセス変更情報構造体 – S3F27

```
typedef struct{
        int         accessmode;         // access mode 0/1
        int         port_count;         // no. of port
        int         *port_list;         // port no. list
}TACCESS_INFO;
```

３．14．４　TACCESS_ERR_INFO, TACCESS_ERR_PORT -アクセス変更応答情報構造体 – S3F28

```
typedef struct{
        int         caack;
        int         err_count;
        TACCESS_ERR_PORT **err_list;
}TACCESS_ERR_INFO;

typedef struct{
        int         port;               // port no.
        int         errcode;            // ok/ng - port
        char        *errtext;           // error text - port
}TACCESS_ERR_PORT;
```

３．15　スプール情報

３．15．１　TSPOOL_INFO - スプール設定情報構造体 – S2F43

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

typedef struct{                         // S2F43 からの取得情報
        int     stream;                 // stream
        int     f_count;                // # of functions
        int     *func_list;             // fuction list
} TSTRE_INFO;
```

３．15．２　TSPOOL_ERR_INFO - スプール設定応答情報構造体 - S2F44

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

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;
```

３．16　オブジェクト・エラー情報

オブジェクト情報関連メッセージの応答メッセージのエラー情報を設定する際、TERR_INFO を使用します。

TERR_INFO は、エラーコードとエラーテキストから成っています。

エラーコードは、以下のように定義されています。

```
#define ERC_OK                  0
#define ERC_OBJ_ERROR           1
#define ERC_TARGET_OBJ_ERROR    2
#define ERC_OBJ_INSTANCE_ERROR  3
#define ERC_OBJ_ATTR_ERROR      4
#define ERC_OBJ_RO_ERROR        5       // Read only
#define ERC_OBJ_TYPE_ERROR      6
#define ERC_OBJ_ATTRVAL_ERROR   7
#define ERC_SYNTAX_ERROR        8
#define ERC_OBJ_CHECK_ERROR     9
#define ERC_INVALID             10
#define ERC_OBJ_BUSY_ERROR      11
```

３．16．１　TERR_INFO

オブジェクトタイプ関連メッセージの応答情報の中で、エラー情報として使用する構造体です。

```
typedef struct{
        int     errcode;
        char    *errtext;
} TERR_INFO;
```