

DSHEng5 装置通信エンジン (GEM+GEM300)

ソフトウェア・パッケージ

装置変数情報定義仕様書

(変数、収集イベント、アラームその他)

2019年6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.	2019.06.06	初版	DSHEng4 版をベースにして、用語などを変更しました。
2.			
3.			
4.			

目次

1. 概要.....	1
1. 1 装置変数情報.....	1
1. 2 装置変数情報定義書式.....	2
1. 3 変数についての制約.....	4
2. 定義仕様.....	5
2. 1 装置定数(EC)の定義.....	5
2. 2 状態変数(SV)の定義.....	6
2. 3 データ変数(DVVAL)の定義.....	7
2. 3. 1 リスト(LIST)変数 (EC, SV, DVVAL) の定義 (DSHENG3-R2 仕様).....	8
2. 4 収集イベント(CE)の定義.....	10
2. 5 レポートIDの定義.....	12
2. 6 アラームIDの定義.....	13
2. 7 スプール対象メッセージの定義.....	14
2. 8 トレース対象状態変数の定義.....	15
2. 9 プロセスプログラム(PP)の定義.....	16
2. 10 書式付きプロセスプログラム(FPP)の定義.....	17
2. 11 レシピ(Recipe)の定義.....	18
3. 装置変数情報のコンパイル.....	19
付録-A 収集イベントレポートのリスト構造の例.....	20

1. 概要

DSHEng5 は、変数を始めとするシステム情報の管理を行います。そして、APP（アプリケーションプログラム）に対しアクセスサービスを提供します。

本仕様書は、ユーザが定義ファイルを使って定義登録できる装置変数情報の仕様について説明します。

本装置起動情報定義ファイルならびに装置変数情報定義ファイルの編集は、DSHGEMSET.EXE 編集プログラムを使って編集、作成することができます。詳しくは、次の資料を参照してください。

DSHEng5-19-30308-00 「DSHEng3 起動ファイル、管理情報ファイル設定・編集プログラム説明書」

また、DSHEng5 製品インストールで、¥DSHEng5¥tool¥ にサンプルファイルが保存されますので参照ください。

ファイル名： EQINFO.TXT （デモプログラムで実際に使用しています。）

1. 1 装置変数情報

表-1 に DSHEng5 が管理する変数情報の一覧が示されていますが、ユーザは定義ファイルのカラムが○になっている情報を定義することができます。

表-1 装置変数情報一覧

定義ファイル	○が定義可能
バックアップ	○がバックアップされる

	情報の種類	定義ファイル	バックアップ	定義コメント	備考
1.	装置変数				
	(1) 装置定数 (EC)	○	○	DEF_EC	
	(2) 装置状態変数 (SV)	○	○	DEF_SV	
	(3) データ変数 (DVVAL)	○	○	DEF_DV	
2.	収集イベント (CE)	○	○	DEF_CE	
3.	レポート ID (RPT)	○	○	DEF_REPORT	
4.	アラーム (ALM)	○	○	DEF_ALARM	
5.	スプール (SPOOL)	○	○	DEF_SPOOL	
6.	トレース (TRACE)	○	○	DEF_TRACE	
7.	プロセスプログラム (PP)	○	○	DEF_PP	PP 情報使用の装置向け
8.	フォーマット付きプロセスプログラム (FPP)	○	○	DEF_FPP	FPP 情報使用の装置向け
9.	レシピ情報 (RCP)	○	○	DEF_RCP	RCP 情報使用の装置向け
10.	キャリア情報 (CAR)		○		
11.	プロセスジョブ (PRJ)		○		
12.	コントロールジョブ (CJ)		○		

(参考) 装置変数情報定義仕様は DSHEng3, DSHEng4, DSHGEMLIB の通信パッケージで互換性があります。

1. 2 装置変数情報定義書式

基本的な定義書式は以下のように行います。

```

<Command> <name>{
    <ID>:    <ID 値>
    <para1>: <P-1 値> [, P-2 値 [, P-3 値] [,....] ]
    <para2>: ....
    .
    <parai>: ....
    .
}

```

- ・ここで、各定義に使用する文字に大文字、小文字の区別はありません。
- ・パラメータの値が数値の場合、頭に **0x** を付けると 16 進表現で定義できます。
- ・パラメータの値を二重引用符(“)囲むこともできます。
- ・”/”(slash が 2 文字)で、その行の”/”以降にコメントを入れることができます。

<Command> - 定義したい情報の種類を表すコマンド
前ページの表-1 の定義コマンド参照。小文字でも可です。

<name> - 情報の固有名（種類の中でユニークでなければならない。）
先頭文字は英文字で始まり、長さは 63 バイト以内です。
name は、C, C++用の include ファイルのマカ用に使用されます。

{ - 情報の内容定義開始記号

<ID>: - 情報識別子の指定を示す。

<ID 値> - 識別子(数値または文字列)

<parai>: - パラメータ-i の定義の指定を示す。

<P-i 値> - パラメータの値、カマ(,)区切りで複数個指定するパラメータもあります。

} - 1 個の情報の定義の終わりを示す記号

なお、変数の値、プロセスプログラム情報のパラメータ値などについて、その値のデータフォーマット（後で定義仕様の説明の中で FORMAT で指定するパラメータ）とそのデータの配列サイズを指定する必要があります。

データフォーマット表現に使用する記号と配列表現については、次ページの一覧表を参照してください。

配列サイズの表現については以下の制限があります。

- (1) A, J フォーマットについては、可変配列を次のように表現します。
最小の配列サイズと最大の配列サイズの範囲を .. で区切って表現します。
例えば、配列サイズが 4 から 16 まで可能な場合、以下のように表現します。
FORMAT: A[4..16]
- (2) LIST, LEND (リスト表現のためのフォーマット) については配列の師弟は行いません。
- (3) その他のデータフォーマットについては配列サイズは固定の指定になります。

データフォーマット一覧表

記号	フォーマット名	単位サイズ	説 明	表現例
L	List	-	Report にリンクされている VID がネステイニングしているときに Lend との対で使用します。	FORMAT: L
B	Binary	1 (byte)	2進値を持つデータ	FORMAT: B[1]
BOOLEAN	Boolean	1	真理値データ 値=0 が FALSE でその他の値は TRUE を意味します。	FORMAT: BOOLEAN[1]
A J	ASCII JIS8	1	文字列データ 配列サイズはバイト単位で指定します。	FORMAT: A[8] 固定長 FORMAT: A[8..16] 可変長 8~16 文字(バイト)まで。
I1 I2 I4 I8	Integer	1 2 4 8	符号付き整数	FORMAT: I2[1]
U1 U2 U4 U8	Unsigned Integer	1 2 4 8	符号なし整数	FORMAT: U4[1]
F4 F8	4 Bytes 8 Bytes Floating Data	4 8	実数	FORMAT: F4[1] FORMAT: F8[1]
LEND	LIST End	-	L (LIST) の終端を指定するために使用します。	FORMAT: LEND

1. 3 変数についての制約

装置変数には以下の3種類の変数があります。

- (1) 装置定数 (EC)
- (2) 装置状態変数 (SV)
- (3) 装置データ変数 (DVVAL)

これらの変数が SECS-II メッセージの中で以下のように区別されるかについてですが、メッセージの中のデータアイテム名の表示によって以下ようになります。

- ①VID と表示されているものは、装置定数、装置状態変数、装置データ変数が全て対象になります。
- ②ECID と表示されているものは、装置定数だけが対象になります。
- ③SVID と表示されているものは、装置状態変数だけが対象になります。

以上のことから、本 DSHEng5 システムにおいて、変数 ID の値はシステムの中でユニークであることを前提にしています。即ち、同じ ID 値を有する複数の変数定義を行わないことが必要です。

表-1.3 変数とSECS-IIメッセージの関連

#	SECS MSGID	メッセージ名	対象とする変数		
			装置定数	状態変数	データ変数
1.	S1F3, 4,	装置状態要求		○	
2.	S1F11, 12	状態変数一覧要求		○	
3.	S2F13, 14	装置定数要求	○		
4.	S2F15, 16	装置定数変更	○		
5.	S2F23, 24	トレース条件設定		○	
6.	S2F29, 30	装置定数名一覧要求	○		
7.	S2F33, 34	レポート設定	○	○	○
8.	S2F43, 44	変数リット属性定義	○	○	○
9.	S2F45, 46	変数リット属性一覧要求	○	○	○

DSHEng5 においては、変数アクセスのための API 関数はそれぞれの変数の種類に対応して準備されています。

2. 定義仕様

2. 1 装置定数(EC)の定義

(1) 書式

```
DEF_EC <ec_name>{
    ecid: <ec id>
    format: <data item format>
    units: <units>
    min: <min value>
    max: <max value>
    nominal: <default value>
    limit: <limitid>, <lower limit>, <upper limit>
}
```

(2) 書式説明

コメント/サブコメント	パラメータ	パラメータの説明	必須
DEF_EC	<ec_name>	<ec_name> が定数名	○
ecid:	<ec id>	定数 ID (10 進または 16 進で指定)	○
format:	<data item format>	データのタイプ, SECS のデータアイテムのフォーマットで表現	○
units:	<units name>	データの物理単位	
min:	<min value>	値の最小値	
max:	<max value>	値の最大値	
nominal:	<default value>	デフォルト値 (=初期設定値)	
limit:	<limitid>, <low>, <upper>	リミット ID と上下限値の定義	

(補足)

- ①nominal: の指定がなければ、デフォルト値は min の値になる。
min もなければ 数値の場合は 0、テキストの場合は "" になる。
- ②limit は上下限の値の設定に使用される。(DSHEng5 の Limit チェック API 関数による。)
- ③format が L(=ICODE_L) の場合は、イベントポート内の変数リストをリスト構造で表現できる。
この変数に与えられた値がリストに含まれる変数の数になる。
詳しくは、2.3.1 参照してください。

(3) 定義例

```
def_ec EC_Itempl{
    ecid:      0x00000401
    format:    U1[1]
    nominal:   16
    units:     "Degree"
    min:       10
    max:       100
    limit:     10, 100, 200
}
```

2. 2 状態変数(SV)の定義

(1) 書式

```

DEF_SV <sv_name>{
    svid:    <sv id>
    format:  <data item format>
    units:   <units>
    min:     <min value>
    max:     <max value>
    nominal: <default value>
}

```

(2) 書式説明

コマンド / サブコマンド	パラメータ	パラメータの説明	必須
DEF_SV	<sv_name>	<sv_name> が変数名	○
svid:	<sv id>	状態変数 ID (10 進または 16 進で指定)	○
format:	<data item format>	データのタイプ, SECS のデータアイテムのフォーマットで表現	○
units:	<units name>	データの単位	
min:	<min value>	値の最小値	
max:	<max value>	値の最大値	
nominal:	<default value>	デフォルト値 (=初期設定値)	
limit:	<limitid>, <lower>, <upper>	リミット ID, lowerdb, upperdb	
event:	<event name>	CE (収集イベント) 名	

(補足)

- ①nominal: の指定がなければ、デフォルト値は min の値になる。
min もなければ 数値の場合は 0、テキストの場合は “” になる。
- ②limit は上下限の値の設定に使用される。(DSHEng5 の Limit チェック API 関数による。)
- ③event: は、SV 値が変化したときにホストに送信する S6F11 の CEID 名 (DEF_CE で定義される CE)
- ④format が L (=ICODE_L) の場合は、イベントポート内の変数リストをリスト構造で表現できる。
この変数に与えられた値がリストに含まれる変数の数になる。
詳しくは、2.3.1 参照してください。

(3) 定義例

```

def_sv SV_ControlState{
    svid:    0x10001
    format:  U2[1]
    min:     1
    max:     5
    nominal: 1
    units:   "state"
    event:   "CE_ControlState"
}

```

2. 3 データ変数(DVVAL)の定義

(1) 書式

```
DEF_DV <dv_name>{
    dvid:    <dvval id>
    format:  <data item format>
    units:   <units>
    min:     <min value>
    max:     <max value>
    nominal: <default value>
    limit:   <limitid>,<lowerdb>,<upperdb>
}
```

(2) 書式説明

コメント/サブコメント	パラメータ	パラメータの説明	必須
DEF_DV	<dv_name>	<dv_name> が変数名	○
dvid:	<dv id>	変数 ID (10 進または 16 進で指定)	○
format:	<data item format>	データのタイプ, SECS のデータアイテムのフォーマットで表現	○
units:	<units name>	データの単位	
min:	<min value>	値の最小値	
max:	<max value>	値の最大値	
nominal:	<default value>	デフォルト値 (=初期設定値)	
limit:	<limitid>,<lower>,<upper>	リミット ID, lowerdb, upperdb	

(補足)

- ①nominal: の指定がなければ、デフォルト値は min の値になる。
min もなければ 数値の場合は 0、テキストの場合は “” になる。
- ②limit は上下限の値の設定に使用される。(DSHEng5 の Limit チェック API 関数による。)
- ③format が L(=ICODE_L) の場合は、イベントポート内の変数リストをリスト構造で表現できる。
この変数に与えられた値がリストに含まれる変数の数になる。
詳しくは、2.3.1 参照してください。

(3) 定義例

```
def_dv DV_Templ_bin{
    dvid:    0x20003
    format:  U2[1]
    units:   Degree
    min:     10
    max:     100
    limit:   10, 100, 200
}
```

2. 3. 1 リスト (LIST) 変数 (EC, SV, DVVAL) の定義

EC, SV, DVVAL 変数共通の定義方法ですが、format が L[] の変数について、その LIST 変数に含める (LINK したい) 他の変数の ID を指定できるようにします。LIST 変数に含める変数の数は、変数の nominal 値で指定します。

本定義を可能にすることによって、ホストから、レポート ID にリンクしたい変数を、LIST 変数を使って一括りにすることができます。

書式について説明します。例として、SV_ListSample ID=1000、

(1) 書式

```
def_sv <sv_name>{
    svid:    <svid>
    format:  L[1]
    nominal: <変数の数>
    vid:    <変数 id-1>           // nominal で指定した数の変数 ID を並べる。
    vid:    <変数 id-2>
    .
    vid:    <変数 id-n>
}
```

(2) 書式説明

- nominal 値に、List 変数にリンクしたい変数の数を指定します。
- その下に、リンクしたい変数を順番に、変数名で、nominal 値分だけ指定します。
含める変数の種類は、EC, SV, DVVAL のいずれでも構いません。ただし、定義対象となっている変数名を指定することはできません。
- DSHEng3 のアプリケーションプログラムによって、本変数の nominal 値と、リンクされている変数の内容を変更することを禁止します。

(3) 定義例

```
def_sv SV_ListSample{
    svid:    12000
    format:  L[1]
    nominal: 25
    vid:    "SV_Slot00"
    vid:    "SV_Slot01"
    vid:    "SV_Slot02"
    (中略)
    vid:    "SV_Slot24"
}
```

例えば、ホストが S2F33 を使って、SV_Slot00～SV_Slot24 をリンクしたい場合は、SV_ListSample 変数 (ID=12000) 1 個だけ、リンクしたい ID として指定するだけでレポート定義が済みます。

レポートのリンク定義変更の後、この SV_ListSample がリンクされているレポートのイベントレポートを送信した場合、リンク情報に従って S6F11 メッセージが送信されることになります。

ここでは、ceid に 1 個のレポート ID、 rpid がリンクされ、 rpid には、 1 個の変数 SV_ListSample がリンクされているものとした場合、 S6F11 メッセージは次のようになります。

```
S6F11
  L-3
  DATAID
  ceid
  L, 1 // report 1 個とします。
  L, 2
  rpid
  L, 1
  L, 25 // SV_ListSample
  SV_Slot00 値
  SV_Slot01 値
  SV_Slot02 値
  SV_Slot03 値
  (中略)
  SV_Slot23 値
  SV_Slot24 値
```

2. 4 収集イベント(CE)の定義

(1) 書式

```
DEF_CE <ce_name>{
    ceid:    <ce id>
    enabled: <flag>
    rptname: <report name-1>
    rptname: <report name-2>
    .
    rptname: <report name-n>
}
```

(2) 書式説明

コマンド/サブコマンド	パラメータ	パラメータの説明	必須
DEF_CE	<ce_name>	<ce_name> が収集イベント名	○
ceid:	<ce id>	収集イベント ID (10 進または 16 進で指定)	○
enabled:	<flag>	有効/無効 flag (0=無効, 1=有効)	
rptname:	<report name-i>	当該 CEID に含まれる report 名 (report ID 対応)	

(補足)

- ①enabled コマンドは初期設定値になる。本パラメータの指定がなければ初期値は有効(=1)になる。
- ②rptname は CEID に含めるレポート ID 名で複数個指定可能。無いイベントも可能。
- ③CEID, RPTID, VID の関係

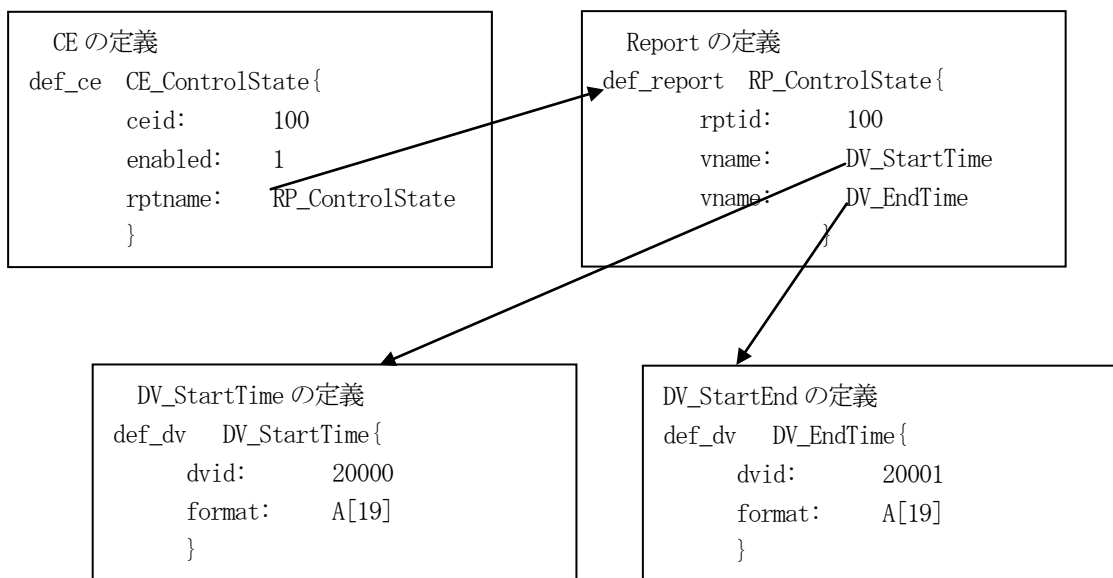
```
CEID --- RPTID-1 --- VID1-1
      |                |--- VID1-2
      |                .
      |                +--- VID1-n
      |
      +--- RPTID-2 --- VID2-1
      |                |--- VID2-2
      |                .
      |                +--- VID2-m
      |
      .
```

(3) 定義例

```
def_ce CE_ControlState{
    ceid:    100
    enabled: 1
    rptname: RP_ControlState
}
```

(4) イベント ID, レポート ID, 変数 ID と S6F11 との関係

次のようにそれぞれの情報が定義されていると仮定します。



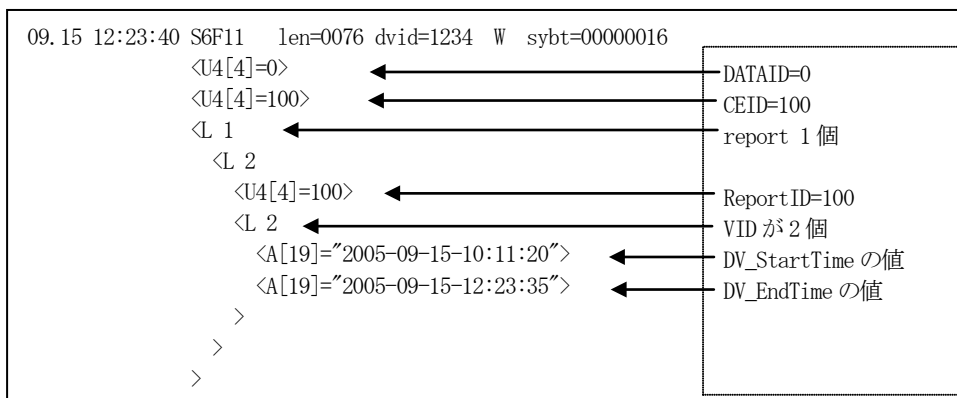
- ①CE_ControlState 収集イベントはCEID=100 であり、リンクしているレポートはRP_ControlState 1 個です
- ②RP_ControlState はRPTID=100 であり、リンクしている装置データ変数は DV_StartTime と DV_EndTime の2つです
- ③DV_StartTime と DV_EndTime 変数はそれぞれDVID が20000, 20001 であり、それぞれフォーマット-A (fmt 10) で最大19 文字の値を持ちます。
それぞれの変数の値が、次のように設定されているとします。
 DV_StartTime = “2005-09-15-10:11:20”
 DV_EndTime - “2005-09-15-12:23:35”

この状態で、APP が収集イベントCE_ControlState を次のAPI 関数を使って送信要求します。

```

EngNotifyEvent( CE_ControlState, evt_callback, 123 );
                (#define CE_ControlState 100 マカ定義されているとします。)
    
```

その結果、DSHEng5 は、上の定義情報と現在値から以下のS6F11 メッセージを組立て、ホストにCEID=100 の収集イベント通知を行うこととなります。



2. 5 レポートIDの定義

(1) 書式

```
DEF_REPORT <report_name>{
    rptid:  <report id>
    vname:  <v name-1>
    vname:  <v name-2>
    .
    vname:  <v name-n>
}
```

(2) 書式説明

コメント/サブコメント	パラメータ	パラメータの説明	必須
DEF_REPORT	<report_name>	<report_name> がレポート名	○
rptid:	<report id>	レポートID (10進または16進で指定)	○
vname:	<variable name-i>	当該レポートIDに含まれる変数名 (EC, SV, VID 対応)	

(補足)

- ①vname: は、当該レポートIDに含まれる変数名。複数個指定可能。
- ②変数名は DEF_EC, DEF_SV または DEF_DV コメントで定義した名前でないといけない。
- ③指定された変数のフォーマットが ICODE_L (L item) の場合、実変数 (ICODE_L, ICODE_END 以外) の変数を指定して、リスト構造で表現することができる。リスト数はこの ICODE_L 変数の値になる。
リストの終了は ICODE_END のフォーマットの変数で行う。
ただし、ICODE_L 変数の値を n とし、ICODE_L と ICODE_END 変数に囲まれる変数の数を p とすると、 $n \leq p$ でなければならない。もし、 $n > p$ の場合は $n=p$ として処理される。
リスト構造は最大8レベルまで可能。

(3) 定義例

```
def_report RP_ControlState{
    rptid:    100
    vname:    DV_StartTime
    vname:    DV_EndTime
}
```

リスト構造の例については **付録-A** を参照のこと。

2. 6 アラームIDの定義

(1) 書式

```
DEF_ALARM <alarm_name>{
    alid:    <alarm id>
    alcd:    <alarm code>
    altx:    <alarm text>
    enabled: <flag>
    ce_on:   <ce-on name>
    ce_off:  <ce-off name>
}
```

(2) 書式説明

コマンド/サブコマンド	パラメータ	パラメータの説明	必須
DEF_ALARM	<alarm_name>	<alarm_name> がアラーム名	○
alid:	<alarm id>	アラームID (10進または16進で指定)	○
alcd:	<alarm code>	アラームコード (10進または16進で指定)	○
altx:	<alarm text>	アラームテキスト、二重引用符(“)で囲むこと。	○
enabled:	<flag>	有効/無効flag (0=無効, 1=有効)	
ce_on:	<ce-on name>	alarm発生時に送信する収集イベント名	
ce_off:	<ce-off name>	alarm復旧時に送信する収集イベント名	

(補足)

①enabledコマンドは初期設定値になる。本コマンドがなければ初期値は有効(=1)になる。

(3) 定義例

```
def_alarm AL_AlarmTempOver{
    alid:    1
    altx:    "Chamber-1 Temperature Over"
    alcd:    "2"
    ce_on:   CE_AlarmOn
    ce_off:  CE_AlarmOff
}
```

2. 7 スプール対象メッセージの定義

(1) 書式

```

DEF_SPOOL <spool_name>{
    stream:  <stream1>
    function: <function-11>
    function: <function-12>
    .
    function: <function-1n>
    stream:  <stream2>
    function: <function-21>
    function: <function-22>
    .
    function: <function-2n>
    .
}

```

(2) 書式説明

コマンド/サブコマンド	パラメータ	パラメータの説明	必須
DEF_SPOOL	<spool_name>	<spool 名	○
stream:	<stream>	対象 stream	○
function:	<function-ij>	指定 stream を持つ function	○

(補足)

①定義は、stream, function の順に並べます。1 個の複数の function の指定を行うことができます。

(3) 定義例

```

def_spool spool_6{
    stream:    6
    function:  11
}

```

2. 8 トレース対象状態変数の定義

(1) 書式

```

DEF_TRACE <trace_name>{
    trid:    <traceid>
    format:  <data item format>
    dsper:   <time format>
    totsmp:  <total sample 数>
    repgsz:  <report record 数>
    svid:    <svid-1>
    svid:    <svid-2>
    .
    svid:    <svid-n>
}

```

(2) 書式説明

コマンド/サブコマンド	パラメータ	パラメータの説明	必須
DEF_TRACE	<trace_name>	トレース名	○
trid:	<traceid>	トレースID	○
format:	<data item format>	tridのフォーマット	○
dsper:	<time format>	データ収集時間のformat hhmss or hhmmsscc	○
totsmp	<total sample>	合計サンプル数	○
repgsz:	<rep group size>	1個のポートに含めるグループレコードサイズ	○
svid:	<svid>	i番目の対象SVID	○

(補足)

- ①dsper はトレースデータをサブリンクする周期であり、時分秒を文字列で表す。
- ②totsmp は合計のサブグループ数(dsper の間隔で totsmp 数だけ分だけトレースする。)
- ③repgsz は1回のS2F33メッセージで送信するレコード数
- ④svid はトレース対象となる状態変数ID(1個以上を指定する。)
指定された svid が dsper 周期でトレースされ、これが1グループ分のトレースデータになる。

(3) 定義例

```

def_trace trace_1{
    trid:  A[6], "TRACE1"
    dsper:  HHMMSS
    totsmp: 9
    repgsz: 3
    svname: SV_ControlState
    svname: SV_Clock
}

```

2. 9 プロセスプログラム(PP)の定義

(1) 書式

```
DEF_PP <pp_name>{
  ppid:    <format>, <ppid_data>
  ppbody:  <format>, <ppbody_data>
}
```

(2) 書式説明

コマンド/サブコマンド	パラメータ	パラメータの説明	必須
DEF_PP	<pp_name>	pp 名	○
ppid:	<format>, <ppid_data>	PPID のフォーマットと値	○
ppbody:	<format>, <ppid_data>	PP 本体のフォーマットと値	

(補足)

①DEF_PP は、S7F3 メッセージ (プロセスプログラム) を採用するシステム用のコマンドである。

(3) 定義例

```
def_pp pp_1{
  ppid:  A[8..16], "PP-1111"
  ppbody: A[8..80], "PPBODY"
}
```

2. 10 書式付きプロセスプログラム(FPP)の定義

(1) 書式

```

DEF_FPP <fpp_name>{
  fppid:   <fppid_data>
  mdlIn:   <model_name>
  softrev: <soft_rev>
  ccode:   <format>, <cmd_code-1>
  pparam:  <format>, <para-1-1>
  pparam:  <format>, <para-1-2>
  .
  pparam:  <format>, <para-1-i>
  ccode:   <format>, <cmd_code-2>
  pparam:  <format>, <para-2-1>
  .
}

```

(2) 書式説明

コメント/サブコメント	パラメータ	パラメータの説明	必須
DEF_FPP	<fpp_name>	書式付きプロセスプログラム名	○
fppid:	<fppid_data>	書式付きプロセスプログラムID	○
mdlIn:	<model name>	装置型式 (A[6])	
softrev:	<soft_rev>	ソフトウェア版番号 (A[6])	
ccode:	<format>, <ccode_val>	コメントコードのフォーマットと値 ppara が続く)	
ppara:	<format>, <para_val>	ccode に対するパラメータの値フォーマットと値	

(補足)

- ①DEF_FPP は、S7F23 メッセージ (書式付きプロセスプログラム) を採用するシステム用のコメントである。
- ②1 個の ccode の後に、複数のパラメータを指定することができる。

(3) 定義例

```

def_fpp fpp_1{
  ppid:  A[7], "PPID001"
  mdlIn: "WPC-12"
  softrev: "SOFT12"
  ccode:  A[4], "CC01"
  pparam: A[6], "200"
  pparam: A[6], "300"
  pparam: A[6], "44.4"
}

```

2. 11 レシピ(Recipe)の定義

(1) 書式

```
DEF_RCP <rcp_name>{
  rcpid:      <recipe ID>
  rcpparname: <rcpparname-1>
  rcpparval:  <format>, <rcpparval-1>
  rcpparname: <rcpparname-2>
  rcpparval:  <format>, <rcpparval-2>
  .
  .
  rcpbody:    <rcpbody data>
}
```

(2) 書式説明

コマンド/サブコマンド	パラメータ	パラメータの説明	必須
DEF_FPP	<recipe_name>	レシピ名	○
rcp_id:	<recipe id>	レシピ ID (文字列)	○
rcpparname:	<parameter name>	レシピパラメータ名 (文字列)	
rcpparval:	<format>, <para val>	レシピパラメータフォーマットと値	
rcpbody:	<recipe body>	レシピ本体	

(補足)

- ①DEF_RCP は、S15F13メッセージ (レシピ管理スタンダード RMS) を採用するシステム用のコマンドである。
- ②1 個の rcpparname の後に、1 個の rcpparval を指定することができる。
レシピパラメータは複数定義することができる。
- ③rcpbody は 1 個だけである。

(3) 定義例

```
def_rcp rcp_process_A{
  rcpid:      "RCP100"
  rcpparname: "PARA1"
  rcpparval:  A[80], "20.0"
  rcpparname: "PARA2"
  rcpparval:  A[80], "30.0"
  rcpbody:    "RCP1000500"
}
```

3. 装置変数情報のコンパイル

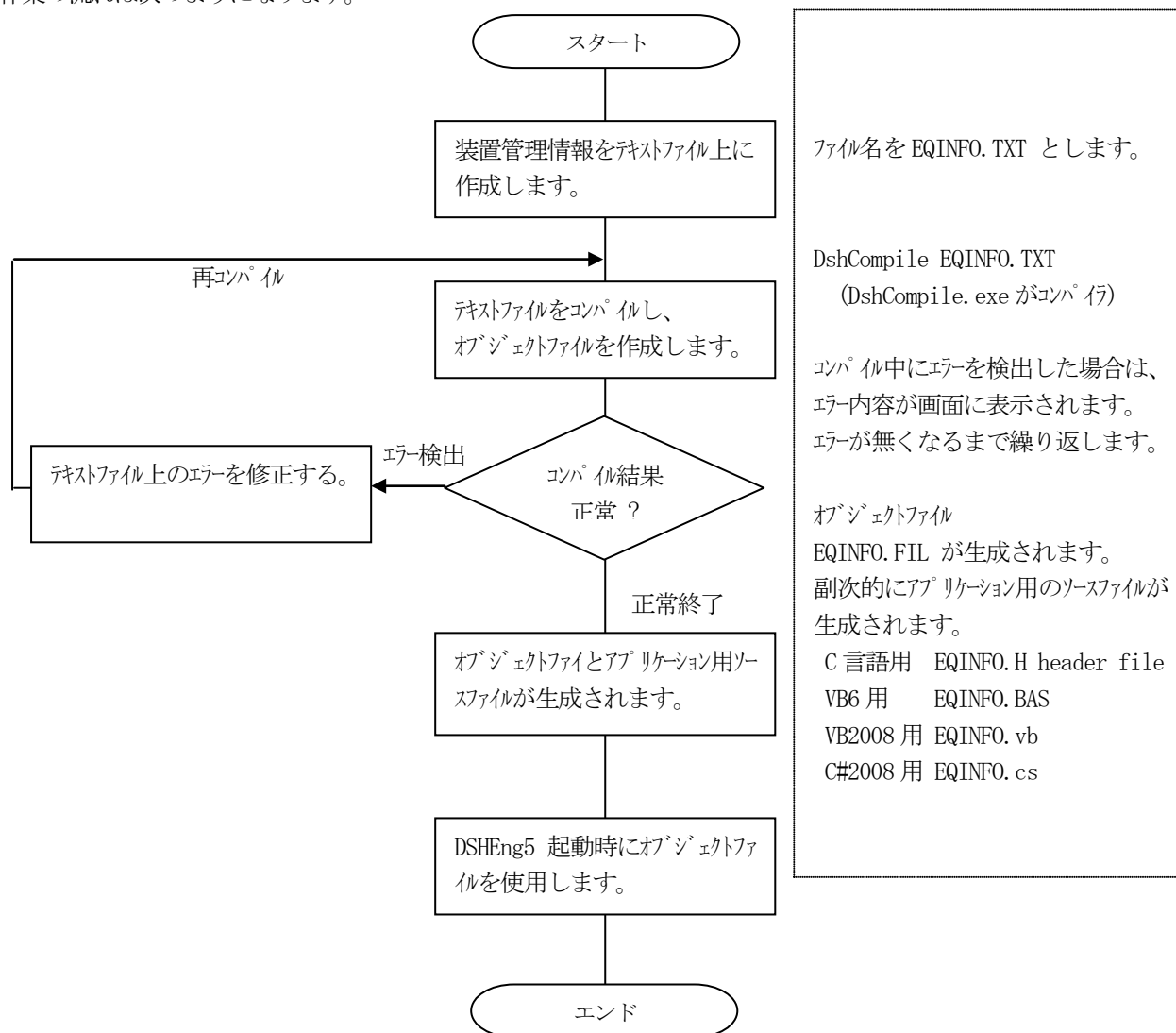
以上述べました装置変数情報は製造装置、ホストコンピュータ間の通信仕様そのほかによって DSHEng5 の登録しなければならない管理情報が決まります。

ユーザは、前述 2. に基づき定義された装置変数情報をテキストファイルに表現し作成します。

ユーザは、作成したこの装置変数情報定義ファイルをコンパイルし、DSHEng5 エンジンが解読できるオブジェクトファイルに変換します。

DSHEng5 エンジンは起動時に作成されたオブジェクトファイルの内容を DSHEng5 内に登録します。そしてユーザが作成するアプリケーションプログラムは DSHEng5 エンジンを通して装置変数情報のアクセスを行うことができます。

作業の流れは次のようになります。



付録-A 収集イベントレポートのリスト構造の例

収集イベントのレポート ID の VID 構成をリスト構造化し、S6F11 を送信する例について記述します。

実際の定義、コンパイルによって生成されるファイルについても説明します。

[前提]

(1) 変数の定義

```
DEF_SV SV_data_1{ // Level-0 1-st item
    svid: 1501
    format: A[0..10]
    nominal: "ITEM-01"
}

DEF_SV SV_data_2{ // Level-1 1-st item
    svid: 1502
    format: A[0..10]
    nominal: "ITEM-11"
}

DEF_SV SV_data_3{ // Level-2 1-st item
    svid: 1503
    format: A[0..10]
    nominal: "ITEM-21"
}

DEF_SV SV_data_4{ // Level-2 2-nd item
    svid: 1504
    format: A[0..10]
    nominal: "ITEM-22"
}

DEF_SV SV_data_5{ // Level-2 3-rd item
    svid: 1505
    format: A[0..10]
    nominal: "ITEM-23"
}

DEF_SV SV_data_6{ // Level-1 final item
    svid: 1506
    format: A[0..10]
    nominal: "ITEM-lend"
}
```



```
DEF_SV SV_list_1{ // level-1 List 開始
    svid: 1511
    format: L
    nominal: 2 // 2 item( 1 List + 1 SVID)
    vid: SV_List_2
    vid: SV_data_2
}

DEF_SV SV_list_2{ // level-2 List 開始
    svid: 1512
    format: L
    nominal: 3 // 3 item ( 3 SVID)
    vid: SV_data_4
    vid: SV_data_5
    vid: SV_data_6
}
```

(2) レポート ID の定義

```
DEF_REPORT RP_List{
    rptid: 1600
    vname: SV_data_1
    vname: SV_List_1 // L nesting (=L-2) level-1 開始
    vname: SV_data_6
}
```

(3) 収集イベント(CE)の定義

```
DEF_CE CE_RList{
    ceid: 1700
    enabled: 1
    rptname: RP_List_1
}
```

[コンパイルと結果]

たとえば、上で定義した情報のソースファイル名、これを 仮に“ENG_ID.TXT”とするとこのファイルの コンパイルによって以下のファイルが生成されます。 コンパイルは 3. で述べた ENG3Conf.exe プログラムを使って行います。

- (1) ENG_ID.FIL - DSHEng5 が実行時に使用されるオブジェクトファイル
- (2) ENG_ID.H - アプリケーションプログラムが使用できる C, C++言語のヘッダーファイル
- (3) ENG_ID.BAS - VB6 言語のための定数ファイル
- (4) ENG_ID.VB - VB2008 言語のための定数ファイル
- (5) ENF_ID.CS - C#2008 言語のための定数ファイル

```

/*-----*/
/*  ENG_ID.H - DshEng3 ID Definition          */
/*-----*/
#define SV_data_1          1501          // 0x000005dd
#define SV_data_2          1502          // 0x000005de
#define SV_data_3          1503          // 0x000005df
#define SV_data_4          1504          // 0x000005e0
#define SV_data_5          1505          // 0x000005e1
#define SV_data_6          1506          // 0x000005e2
#define SV_list_1          1511          // 0x000005e7
#define SV_list_2          1512          // 0x000005e8
#define RP_List            1600          // 0x00000640
#define CE_RList           1700          // 0x000006a4

```

```

Attribute VB_Name = "Eng_Id.bas"
'-----
'  ENG_ID.BAS - DshEng3 ID definition
'-----
Public Const SV_data_1 =          1501          ' 0x000005dd
Public Const SV_data_2 =          1502          ' 0x000005de
Public Const SV_data_3 =          1503          ' 0x000005df
Public Const SV_data_4 =          1504          ' 0x000005e0
Public Const SV_data_5 =          1505          ' 0x000005e1
Public Const SV_data_6 =          1506          ' 0x000005e2
Public Const SV_list_1 =          1511          ' 0x000005e7
Public Const SV_list_2 =          1512          ' 0x000005e8
Public Const RP_List =            1600          ' 0x00000640
Public Const CE_RList =           1700          ' 0x000006a4

```

[イベント通知実行結果]

EngNotifyEvent(CE_Rlist, 0, 111) イベント通信 API 関数を実行すると、以下の S6F11 のメッセージが送信されます。ここで CE_Rlist = 1700 です。

```

CH-1 受 S6F11 len=0096 dvid=1234 W blk=0000 sybt=00000008
<L 3
  <U4[4]=2> // dataid
  <U4[4]=1700> // ceid
  <L 1 // la
    <L 2
      <U4[4]=1600> // report id
      <L 3 // level-0 3 items(list を含め)
        <A[7]="ITEM-01"> // level-0 item-1
        <L 2 // " item-2(=list)
          <A[7]="ITEM-11"> // level-1 item-1
          <L 3 // " item-2(=list)
            <A[7]="ITEM-21"> // level-2 item-1
            <A[7]="ITEM-22"> // " item-2
            <A[7]="ITEM-23"> // " item-3
          >
        >
      <A[9]="ITEM-1end"> // level-0 item-3(=end)
    >
  >
>

```