GEM 通信エンジン・DSHEng5 (GEM+GEM300)

ソフトウェア・パッケージ

DSHEng5 通信エンジン・デモプログラム

説　明　書

装置変数、通信メッセージ、WP 通信シナリオ・シミュレーション

２０１９年6月

株式会社データマップ

文書番号　DSHEng5-19-30502-00

［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】

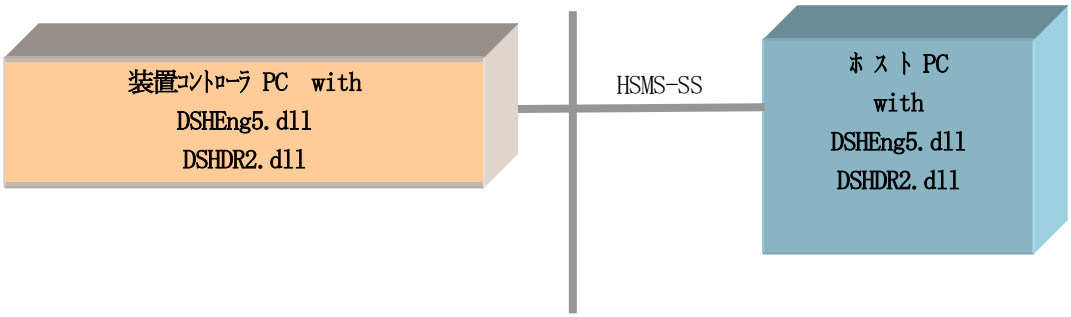
番号	改訂日付	項 目	概 略
1.	2019 年 6 月	初版	
2.			
3.			
4.			

目　次

１．概要

本説明書は株式会社データマップが開発・販売する GEm/GEM300 通信制御用の DSHEng5 通信エンジンならびに DSHEng5 ライブラリ・ソフトウェア・パッケージを評価するために作成されたデモ・プログラムの基本的な仕様と画面を使った操作などについて説明します。

基本仕様は、次のシステム構成を想定し、行うことを前提にしています。



ホスト側 PC には、同じ DSHEng5 通信エンジンのデモプログラムを使用します。

OS は、Windows、プログラム言語は.Net テクノロジによる **C#2013、.Net Vb2013 言語**を使用しています。
デモプログラムに対する操作は、.Net 言語を使って、GUI インタフェース画面を使って行います。

GEM レベルでの通信制御ならびに装置情報の管理には、DSHEng5 クラスのライブラリに提供されるクラス群を利用するプログラミングになっています。

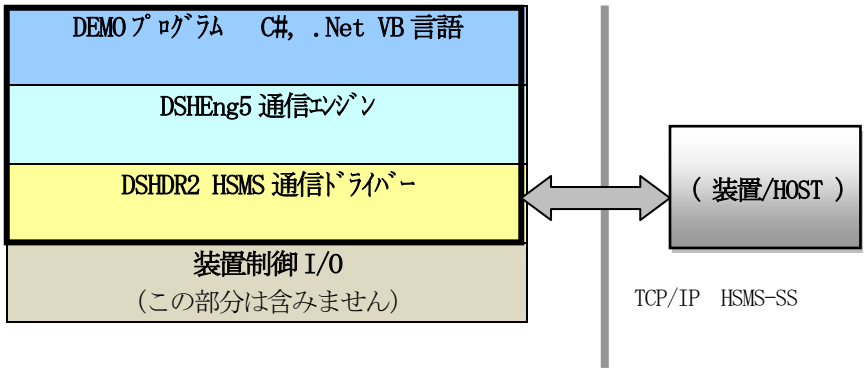
これらによって、装置情報の設定、参照操作、具体的な通信制御機能の実行操作、さらには、装置、ホスト間の連続メッセージ通信機能を確認できます。簡単なウエハー処理 (WP) モデルによるシナリオ処理のデモも含んでいます。

本デモプログラムは、ユーザが DSHEng5 通信エンジンをアプリケーションに組み込む上でのプログラミングの具体的な１つの方法を提供します。ソースファイルがオープンされていますので参照ください。

DSHEng5 ライブラリによるアプリケーションの開発手順については、下記資料を参照ください。

文書番号　DSHEng5-19-30305-00　**「 プログラミングの手引き 」**

デモプログラムを構成する要素は、以下のような階層構造になります。



１．１　DSHEng5 通信制御エンジン関連ドキュメント一覧表

関連ドキュメントとして以下のものがあります。

（1）DSHEng5 ユーザーズ・ガイド、一般関連ドキュメント - ¥DSHEng5¥doc に保存

#	文書番号	文書名	注釈
1	DSHEng5-19-30300-00	DSHEng5 通信制御ｴﾝｼﾞﾝﾗｲﾌﾞﾗﾘ(SECS/HSMS) ﾕｰｻﾞｰｽﾞ･ｶﾞｲﾄﾞ	DSHEng5 の全般的な機能の説明書です。
2	DSHEng5-19-30301-00	DSHEng5 起動ﾌｧｲﾙ定義仕様書	装置別の起動情報の定義方法の説明書です。
3	DSHEng5-19-30302-00	DSHEng5 装置管理情報定義仕様書 （変数、収集ｲﾍﾞﾝﾄ、ｱﾗｰﾑその他）	DSHENG3 と同じ内容です。 定義ﾌｧｲﾙはﾃｷｽﾄﾌｧｲﾙです。
4	DSHEng5-19-30303-00	装置管理情報定義ﾌｧｲﾙｺﾝﾊﾟｲﾗ説明書	DSHENG3 と共通です。
5	DSHEng5-19-30304-00	DSHEng5 への手引き	DSHEng5 導入時に参考にする作業手順書です。
6	DSHEng5-19-30305-00	ｲﾝｽﾄｰﾙと保存ﾌｧｲﾙ	ｲﾝｽﾄｰﾙ手順書です。
7	DSHEng5-19-30308-00	DSHEng5, DSHENG4 起動ﾌｧｲﾙ、装置管理情報ﾌｧｲﾙ設定・編集ﾌﾟﾛｸﾞﾗﾑ説明書	DSHENG4, DSHEng5 共通
8	DSHEng5-19-30310-00	変数リミット監視機能　説明書	ﾘﾐｯﾄ監視の考え方、処理方法の説明書です。
9	DSHEng5-19-30351-00	ﾊﾞｯｸｱｯﾌﾟﾌｧｲﾙ参照ﾌﾟﾛｸﾞﾗﾑ説明書	DOS ｺﾏﾝﾄﾞで List 構造で表示します・

（2）DSHEng5 GEM 通信エンジン・クラス説明書　¥DSHEng5¥doc-class に保存

（APP プログラミングで使用するクラスの説明書です。）

Vol 番号	文書番号	内容
Vol-1	DSHENG5-19-30321-00	DSHEng5 GEM 通信エンジン説明書 Vol－１ エンジン起動・停止、通信確立関連クラス
Vol-2	DSHENG5-19-30322-00	変数情報関連クラス （EC, SV, DVVAL, CE, Report, Alarm ）
Vol-3	DSHENG5-19-30323-00	プロセス情報関連クラス （PP, FPP, RECIPE, PRJ, CJ, CARRIER, SUBSTRATE ）
Vol-4	DSHENG5-19-30324-00	SECS-II メッセージ送信クラス
Vol-5	DSHENG5-19-30325-00	SECS-II 通信メッセージ情報保存クラス
Vol-6	DSHENG5-19-30326-00	SECS-II 通信メッセージエンコード／デコード処理クラス

（3）HSMS 通信ドライバー関連ドキュメント　¥DSHEng5¥doc に保存

#	文書番号	文書名	注釈
1	DSHDR2-06-20000-02	DSHDR2 SECS/HSMS ﾚﾍﾞﾙ-2 通信制御ﾄﾞﾗｲﾊﾞｰ ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ	SECS/HSMS 通信制御ﾄﾞﾗｲﾊﾞｰの説明書です。
2	DSHDR2-06-20040-0	DSHDR2 ﾚﾍﾞﾙ 2 通信ﾄﾞﾗｲﾊﾞｰ通信ﾛｸﾞﾓﾆﾀｰ説明書	ﾘｱﾙﾀｲﾑで通信ﾄﾗﾝｻﾞｸｼｮﾝをﾓﾆﾀｰ画面で見ることができます。

（4）デモプログラム関連ドキュメント　¥DSHEng5¥doc-demo に保存

#	文書番号	文書名	注釈
1	DSHEng5-19-30501-00	ｸﾗｽ・ﾗｲﾌﾞﾗﾘ・ﾃﾞﾓﾌﾟﾛｸﾞﾗﾑ説明書	（本ﾄﾞｷｭﾒﾝﾄです。）
2	DSHEng5-19-30502-00	DSHEng5 ｸﾗｽ･ﾗｲﾌﾞﾗﾘ版　ﾃﾞﾓﾌﾟﾛｸﾞﾗﾑ インストールと保存ファイル	C#, .Net VB ﾃﾞﾓﾌﾟﾛｸﾞﾗﾑです。

１．２　デモ関連ファイルの保存場所

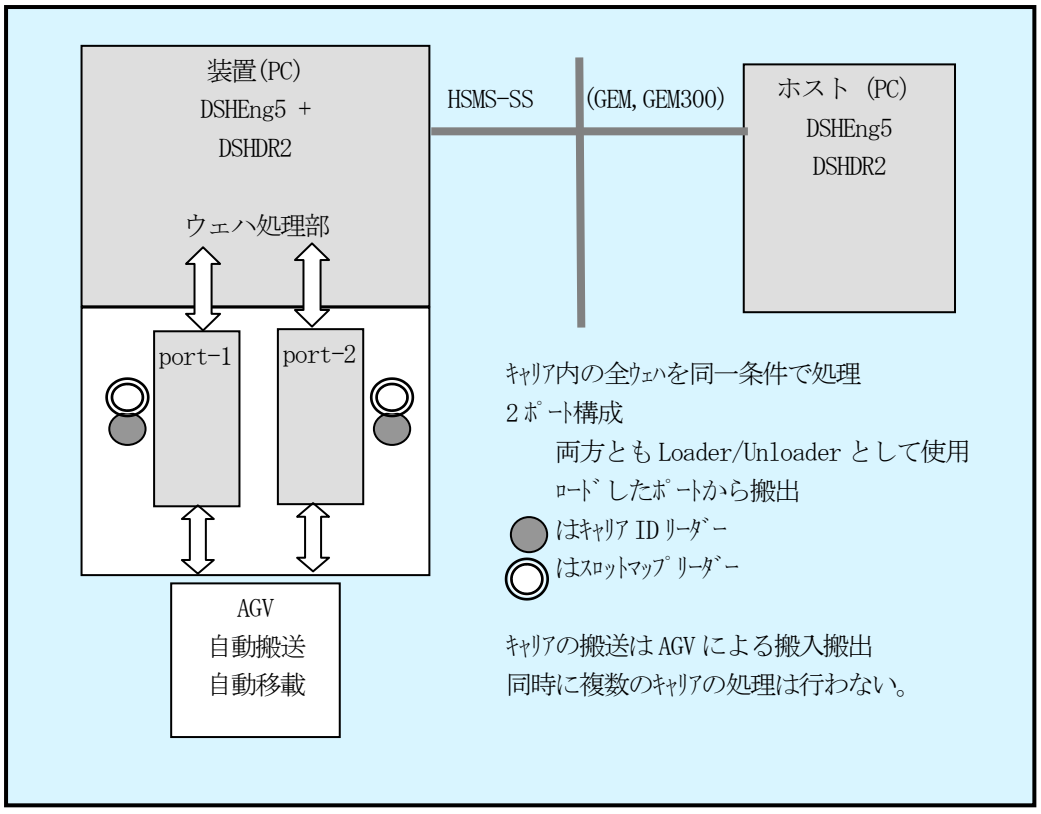
デモプログラムとして提供される setup.exe を実行すると、以下のディレクトリに必要がファイルが保存されます。
ｃ：ドライブの　¥dsheng4 の下に保存されます。

home dir	path	ﾌｧｲﾙ名	種類、機能など
¥dsheng5¥	bin	eng5Appcsdemo.exe dsheng4.dll, eng4class.dll など	実行ﾌﾟﾛｸﾞﾗﾑ・ﾌｧｲﾙ C#の実行ﾌﾟﾛｸﾞﾗﾑ VB の実行ﾌﾟﾛｸﾞﾗﾑ ｴﾝｼﾞﾝ・ﾌﾟﾛｸﾞﾗﾑ ｸﾗｽﾗｲﾌﾞﾗﾘ・ﾌﾟﾛｸﾞﾗﾑ
	cnf	equip.cnf comm_EQ.def host.cnf comm_h.def eqinfo.fil	環境定義ﾌｧｲﾙ 装置起動ﾌｧｲﾙ　　(装置側) DSHDR2 HSMS 通信定義ﾌｧｲﾙ (ﾎｽﾄ側) 装置管理情報定義ﾌｧｲﾙ
	tool	dshgemset5.exe dshcompile.exe seeback.exe	ﾂｰﾙ・ﾌﾟﾛｸﾞﾗﾑ 装置管理情報編集ﾌﾟﾛｸﾞﾗﾑ 同ｺﾝﾊﾟｲﾙ・ﾌﾟﾛｸﾞﾗﾑ 装置情報ﾊﾞｯｸｱｯﾌﾟ参照ﾌﾟﾛｸﾞﾗﾑ
	logmon	logmon.exe	HSMS 通信ﾛｸﾞ・ﾓﾆﾀｰ(ﾘﾓｰﾄ) ﾘｱﾙﾀｲﾑ・HSMS 通信ﾛｸﾞﾓﾆﾀｰ・ﾌﾟﾛｸﾞﾗﾑ
	Eng5AppCsDemo¥	Eng5APPCsDemo.sln	C#ﾃﾞﾓ・ﾌﾟﾛｸﾞﾗﾑ　ｿﾘｭｰｼｮﾝ・ﾌｧｲﾙ
		Eng5AppCsDemo	C#ﾃﾞﾓ・ﾌﾟﾛｸﾞﾗﾑ ｿｰｽﾌｧｲﾙ
	doc		ﾕｰｻﾞｰｽﾞ・ｶﾞｲﾄﾞ 装置起動ﾌｧｲﾙ、装置管理情報定義ﾌｧｲﾙの説明書など
	doc-class		DSHEng5 ｸﾗｽ説明書
	doc-demo		ﾃﾞﾓﾌﾟﾛｸﾞﾗﾑ関連説明書

２．構成

２．１　装置モデルと構成

本デモプログラムの制御対象装置として次のようなモデルを想定します。



本デモ・プログラムは、DSHEng5 ならびに DSHEng5 通信エンジンを使って、GEM に基づく SECS-II メッセージの通信制御と装置情報管理機能を行うことを目的にしています。

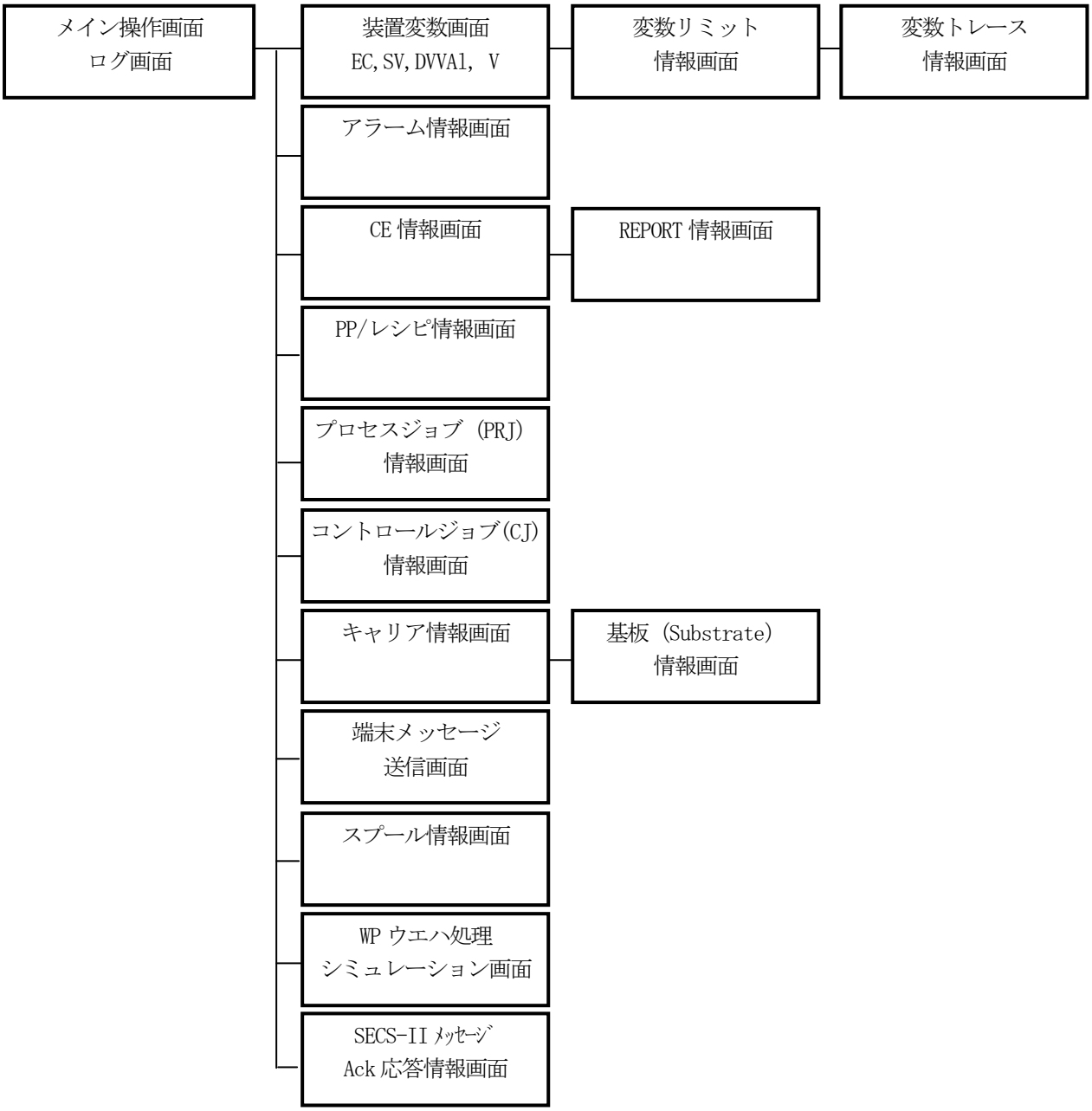
デモプログラムでは、簡単なシナリオモデルにより、WP (Wafer Processing) の連続処理を行う機能も含んでいます。これを WP シミュレーションと呼ぶことにします。

シナリオの中では、装置のポートでのキャリアの Load/Unload は、特にオペレータの操作をしないで自動的に進めることにします。

２．２　デモプログラムの機能構成

２．２．１　操作画面構成

DSHEng5 エンジンのほとんどの機能の確認画面が準備されています。



注)
変数(EC, SV,DVVAL, V), CE, Report, Alarm 情報画面には、Windows >net のｺﾝﾄﾛｰﾙ DataGridView を使用した一覧表の表示例があります。また、HTML ﾌｧｲﾙを作成し、それを IE で表示する機能も含まれます。

２．２．２　通信関連機能

１次メッセージの送信処理と１次メッセージの受信処理を行います。

２．２．２．１　１次メッセージの送信

１次メッセージの送信は、DSHEng5 の各メッセージ送信のために設けられたクラスを使って送信します。

例えば、S6F11 メッセージの送信クラスは、class_SendS6F11 になります。
詳しい、説明は以下を参照ください。

文書番号 DSHENG5-19-30324-00 Vol4 SECS-II 「メッセージ送信クラス」
文書番号 DSHEng5-19-30304-00　「DSHEng5 への手引き」 6.2.3 １次メッセージの送信

１次メッセージの送信には、２つのモードがあります。(ブロックモードが追加されました。)

（1）非ブロックモードの送信
送信クラスの send メソッドに callback 関数（イベント・ハンドラー）のポインタと結果情報を受け取る情報格納領域を引数として渡し、応答メッセージの受信によって、callback 関数を呼び出して貰うことによって送信完了を確認することになります。

（2）ブロックモードの送信
class_SendSxFy 送信クラスの SendSxFy_wait()メソッドを使って送信要求をします。要求した後、SendSxFy_wait()を発したプログラム(スレッド)は送信終了までブロックされます。(先に進まない)
SendSxFy_wait()の戻り値が送信結果になります。

２．２．２．２　１次メッセージの受信

相手からの１次メッセージの受信処理は、ユーザが受信処理をしたいとエンジンに登録した１次メッセージの受信処理になります。
受信したい１次メッセージの登録は、EngAPI クラスを使って行います。登録方法については、以下を参照してください。

文書番号 DSHEng5-19-30304-00　「DSHEng5 への手引き」
ユーザプログラム処理対象受信１次メッセージの登録

１次メッセージの受信は、ユーザが EngAPI クラスに受信 Polling を依頼し、依頼時にメッセージ受信イベント・ハンドラーを渡しておきます。EngAPI クラスが１次メッセージを受信すると、受信情報をパラメータにして、非同期にイベント・ハンドラーを呼び出してくれます。

また、処理した後、受信したメッセージに対する２次メッセージの応答送信を行います。
これらについては、以下の説明書を参照ください。

文書番号 DSHEng5-19-30304-00 「DSHEng5 への手引き」6.2.2　受信メッセージの処理

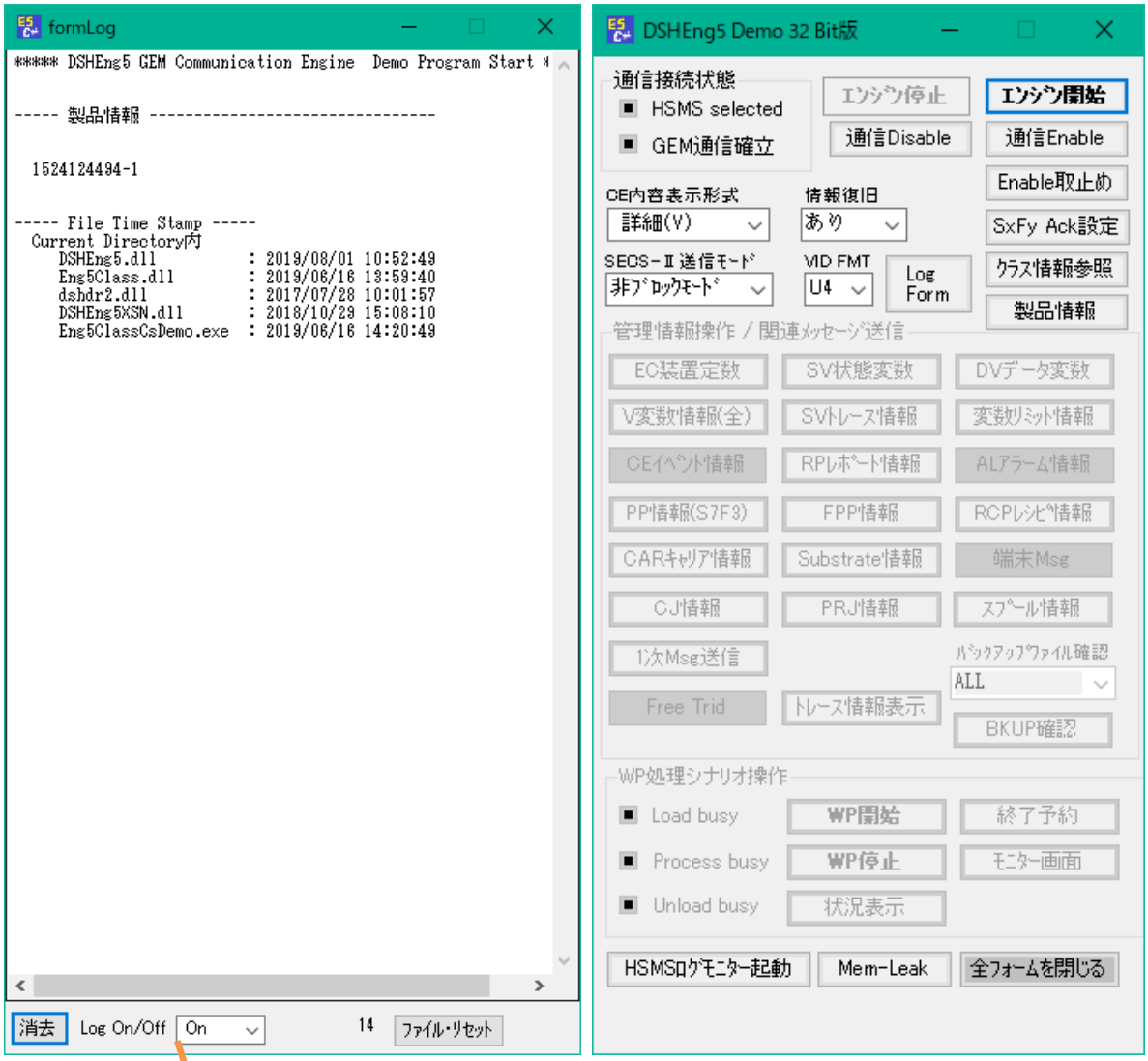
３．画面と操作

３．１　メイン画面

Eng5CsDemoV3.exe または Eng5VbDemoV3.exe デモプログラムの起動によって、デモプログラムのメイン画面が表示されます。

２．２．１の画面構成図のトップレベルの画面です。下に示します。

左側には、処理のログ記録を表示する画面が位置します。（画面下側の Log On/Off の選択で表示出力しないようにできます。）



ﾛｸﾞ表示をする/しないを選択します。
（ﾛｸﾞﾌｧｲﾙには記録されます。）
通信が多くなりｺﾝﾋﾟｭｰﾀに負荷がかかった場合など、Off にしてください。

３．１．１　通信制御の開始と停止操作

３．１．１．１　開始操作

デモ・プログラムが起動から、ｴﾝｼﾞﾝ、装置開始まで流れは次のようになります。

操作(ボタン)	ﾌﾟﾛｸﾞﾗﾑの処理	画面の状態と操作
ﾃﾞﾓﾌﾟﾛｸﾞﾗﾑ起動	ﾒｲﾝ初期画面が表示 される。	ｴﾝｼﾞﾝ開始ﾎﾞﾀﾝが有効になり、 装置変数 ID などのﾌｫｰﾏｯﾄを U2 にしたい場合は、ｺﾝﾎﾞﾎﾞｯｸｽで選択しておく。
通信ｻｲﾄﾞ ﾎｽﾄ/装置選択	通信サイドを選択する。	ﾎｽﾄ / 装置を選択する。 ｴﾝｼﾞﾝ開始前に選択してください。
ｴﾝｼﾞﾝ開始 ｸﾘｯｸ	装置変数 ID(数値 ID)が U2 に選択されていれば、ｴﾝｼﾞﾝに設定する。 EngAPI ｸﾗｽを使って装置制御を開始します。装置 config ﾌｧｲﾙ名をｾｯﾄし、start ﾒｿｯﾄﾞを実行する。 装置開始が正常にできたら、 ・予約変数、ｲﾍﾞﾝﾄ ID の設定をする。 ・ﾕｰｻﾞが処理する受信 1 次ﾒｯｾｰｼﾞを登録する。 ・EngAPI ｸﾗｽに SECS-II 受信ﾒｯｾｰｼﾞのﾎﾟｰﾘﾝｸﾞを開始させる。 ・timer-1 を開始し、HSMS-SS 通信確立と GEM ﾚﾍﾞﾙでの通信確立を監視する。	開始が正常終了ならば ｴﾝｼﾞﾝ開始ﾎﾞﾀﾝ　= 無効 ｴﾝｼﾞﾝ停止ﾎﾞﾀﾝ　= 有効 正常に開始されると、comm.def に指定された PORT を通して相手装置と HSMS-SS ﾌﾟﾛﾄｺﾙ確立のための制御も開始されます。 HSMS-SS のｾﾚｸｼｮﾝ確立したら、通信接続状態の HSMS-Selected ﾗﾝﾌﾟが緑色に点灯します。 同時に、GEM ﾚﾍﾞﾙの通信確立 Enable も行われ、(S1F13, S1F14 のやり取り)通信が確立されれば通信接続状態の GEM 通信確立ﾗﾝﾌﾟが緑色に点灯します。(次ページのメイン画面参照)

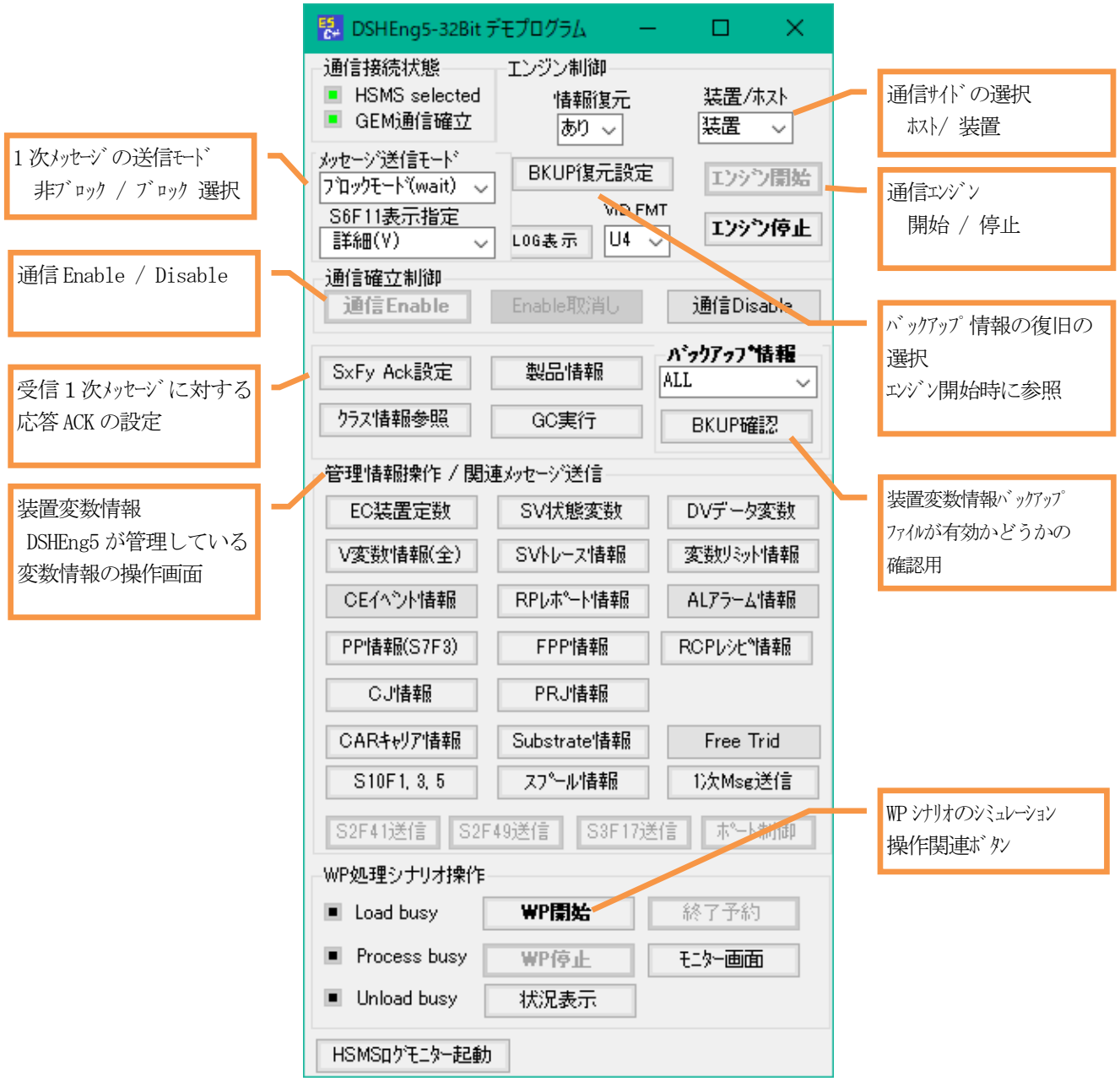
次ページに、エンジン、装置起動後のメイン操作画面表示します。

使用しないボタンは無効になっています。

Eng5CsDemoV3 デモプログラムのメイン画面です。エンジン開始　ボタンクリックによって、通信エンジンが開始された後の画面です。

エンジン開始　ボタンのクリックの後、通信確率のための通信 Enable ボタンは自動的にクリックされ、ホストとの HSMS 通信確立、GEM レベルでの通信確立（S1F13, S1F14 のやり取り）が行われます。

この画面では、HSMS 通信確立、GEM 通信確立状態になっています。画面左上側の通信接続状態 ● ランプが緑色に点灯しています。



３．１．１．２　情報操作、制御操作画面

操作ボタンと各情報操作画面は次のとおりです。

３．１．１．３　停止操作

制御の停止は、エンジン停止ボタンをクリックします。

全て停止したら、エンジン開始ボタンが有効になり、エンジン停止ボタンは、無効になります。

３．１．２　画面内の各種実行ボタンの操作

前述、３．１．１．２に挙げたボタン以外のもので、クリックで直接、機能実行するためのボタン操作について記述します。基本的には、エンジン開始した後に実行できる操作です。

３．１．２．１　通信 Enable / Enable 取消し / 通信 Disable の操作

GEM レベルでの通信確立のための操作です。
class_EnableComm クラスを使ってこれらの機能を実行します。

Enable()メソッドによって、S1F13 の実際のやり取りはエンジンが処理します。

ボタン	ｿｯﾄﾞ	callback のｸﾗｽ	動作
通信 Enable	Enable()	class_CALLBACK.CallbackDefault	S1F13 の送信で通信接続手続きを行う。 終了したら callback で通知される。
Enable 取消し	Cancel()	（なし）	enable()を取りやめる。
通信 Disable	Disable()	class_CALLBACK.CallbackDefault	通信確立状態を未確立状態にする。 終了したら callback で通知される。

Enable()結果のコールバック関数が呼び出され、結果が正常であれば通信確立したことになります。

通信確立後、相手装置との間で、アプリケーションレベルのメッセージの送受信が可能になります。
また、画面の通信接続状態の GEM 通信確立ランプが緑色に点灯します。（●）

３．１．２．２　バックアップ情報の確認

以前に保存した装置管理情報のバックアップファイルの内容が正しい形式で保存されているかどうかの確認を行うための操作です。
エンジン開始前に、この確認をすることができます。

情報全体を一度に確認する方法と、情報を個別に確認する方法があります。この選択はコンボボックスによって指定します。

情報を選択し、BKUP 確認ボタンをクリックすることで、その結果がログ表示されます。

EngAPI クラスのメソッドを使用します。

確認結果 ei とバックアップファイル名は次のように表示されます。

```
check_backup_″ + “<ﾊﾞｯｸｱｯﾌﾟ ﾌｧｲﾙ名>” + ″ : result = ″ + ei.ToString())
```

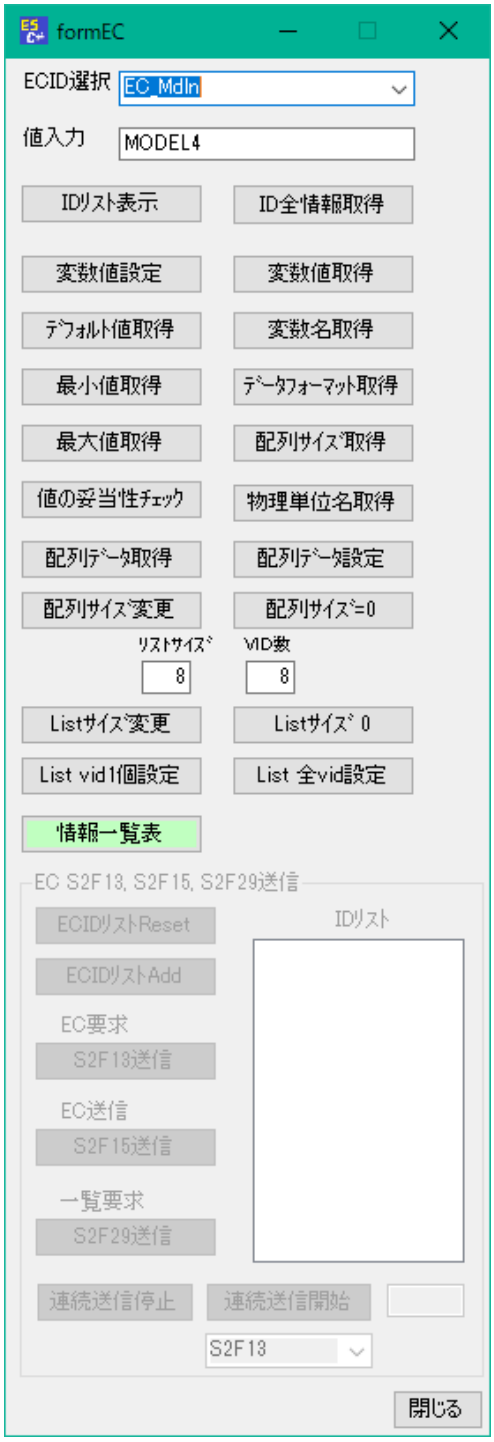
３．２　変数関連情報操作画面

装置変数には、３種類の変数があります。装置定数(EC)、装置状態変数(SV)、データ変数(DVVAL)です。
これら３つの操作画面が独立してあります。また、３種類の変数を一括りにした変数画面があります。

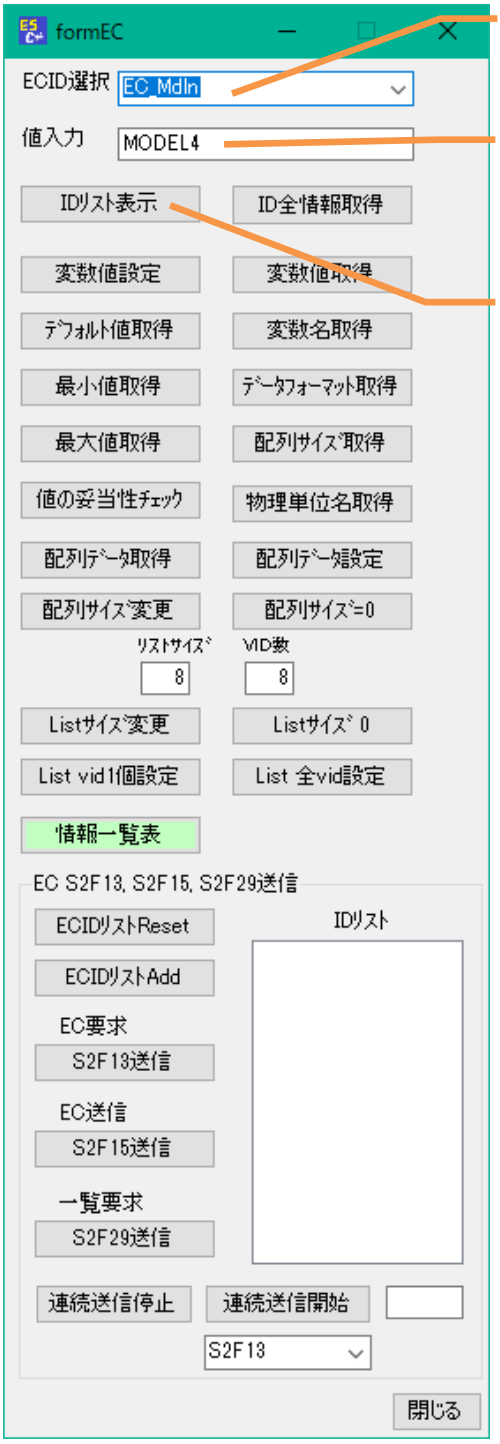
３．２．１　EC 装置定数操作画面

装置定数 EC 操作画面は次のようになります。
それぞれのボタンの機能は、ボタンの表示が示すとおりです。　操作結果は、LOG 画面に表示されます。

装置側選択



ﾎｽﾄ側選択



操作したい ID を選択します。

現在値の表示と
変更したい値入力用

ｴﾝｼﾞﾝに登録されている
全 ID と名前を Log 画面に一覧表示します。

以下のクラスを使用します。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_EC	get_id_list(), get(), set()他

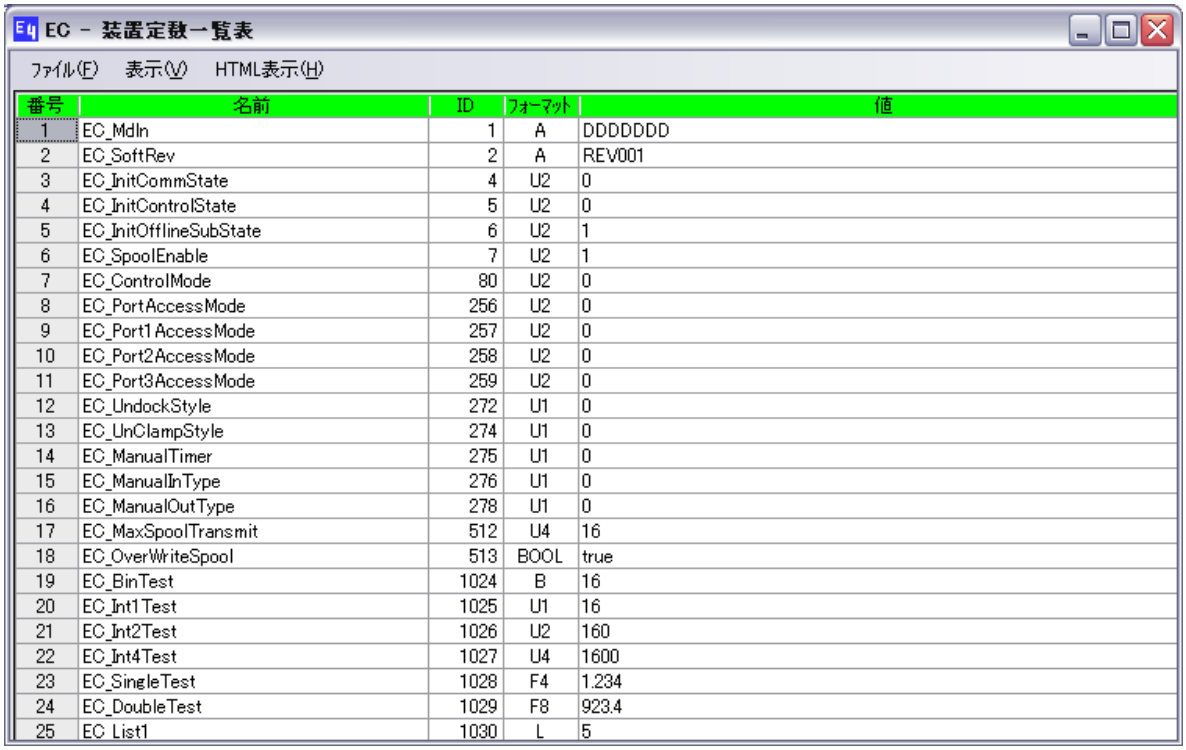
LOG 表示例 － EC_Mdln の変数値取得 ―> Model2 を取得
変数値設定 ―> Model3 に変更

```
** ecid = 1 EC_Mdln
    value    = Model2
** ecid = 1 EC_Mdln
  set OK
```

次ページに情報一覧表ボタンクリックによる操作と機能について説明します。

３．２．１．１　DataGridView コントロールを使った一覧表画面

３．２．１の画面で、　ボタンをクリックすると、下に示す EC 情報一覧表が表示されます。



番号	名前	ID	フォーマット	値
1	EC_Mdln	1	A	DDDDDDD
2	EC_SoftRev	2	A	REV001
3	EC_InitCommState	4	U2	0
4	EC_InitControlState	5	U2	0
5	EC_InitOfflineSubState	6	U2	1
6	EC_SpoolEnable	7	U2	1
7	EC_ControlMode	80	U2	0
8	EC_PortAccessMode	256	U2	0
9	EC_Port1AccessMode	257	U2	0
10	EC_Port2AccessMode	258	U2	0
11	EC_Port3AccessMode	259	U2	0
12	EC_UndockStyle	272	U1	0
13	EC_UnClampStyle	274	U1	0
14	EC_ManualTimer	275	U1	0
15	EC_ManualInType	276	U1	0
16	EC_ManualOutType	278	U1	0
17	EC_MaxSpoolTransmit	512	U4	16
18	EC_OverWriteSpool	513	BOOL	true
19	EC_BinTest	1024	B	16
20	EC_Int1Test	1025	U1	16
21	EC_Int2Test	1026	U2	160
22	EC_Int4Test	1027	U4	1600
23	EC_SingleTest	1028	F4	1.234
24	EC_DoubleTest	1029	F8	923.4
25	EC_List1	1030	L	5

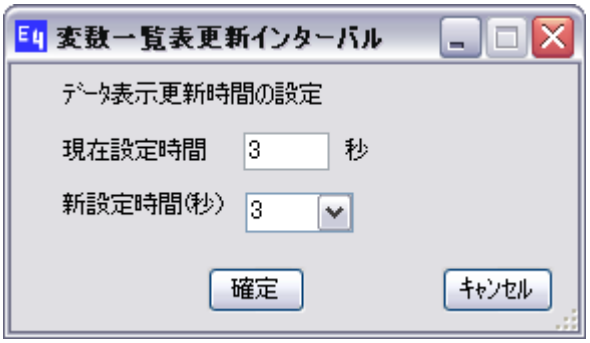
本画面には、任意にメニュータブクリックによる更新と、タイマーを使って一定間隔で画面を更新する２つの方法があります。メニュータブ、表示(V) を使用します。

メニュータブの機能は以下の通りです。

メニュー	メニュー	機能
表示(V)	表示更新(R)	直ちに更新します。
	周期更新 On(N)	一定間隔での更新を可能にします。
	周期更新 Off(F)	一定間隔での更新を禁止にします。
	周期時間設定(P)	一定間隔更新の間隔時間を別画面で設定します。

周期更新 On/Off のタブは、どちらか１つが有効になります。

間隔時間の設定画面は以下の通りです。



時間値を選択し、確定ボタンのクリックで間隔時間が設定されます。

３．２．２　SV 装置状態変数操作画面

装置状態変数 SV 操作画面は次のようになります。
それぞれのボタンの機能は、ボタンの表示名のとおりです。
操作結果は、LOG 画面に表示されます。

装置側選択

ホスト側選択

以下のクラスを使用します。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_SV	get_id_list(), get(), set()他

３．２．２．１　DataGridView コントロールを使った一覧表画面

３．２．１．１　(EC の画面) を参照ください。

３．２．３　DVVAL 装置データ変数操作画面

装置データ変数 DVVAL 操作画面は次のようになります。
それぞれのボタンの機能は、ボタンの表示名のとおりです。
操作結果は、LOG 画面に表示されます。

以下のクラスを使用します。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_DV	get_id_list(), get(), set()他

３．２．３．１　DataGridView コントロールを使った一覧表画面

３．２．１．１　（EC の画面）を参照ください。

３．２．４　変数 装置変数 EC, SV, DVVAL 操作画面

変数(EC, SV, DVVAL)全体の操作画面は次のようになります。

それぞれのボタンの機能は、ボタンの表示名のとおりです。
操作結果は、LOG 画面に表示されます。

以下のクラスを使用します。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_V	get_id_list(), get(), set()他

３．２．４．１　DataGridView コントロールを使った一覧表画面

３．２．１．１　(EC の画面) を参照ください。

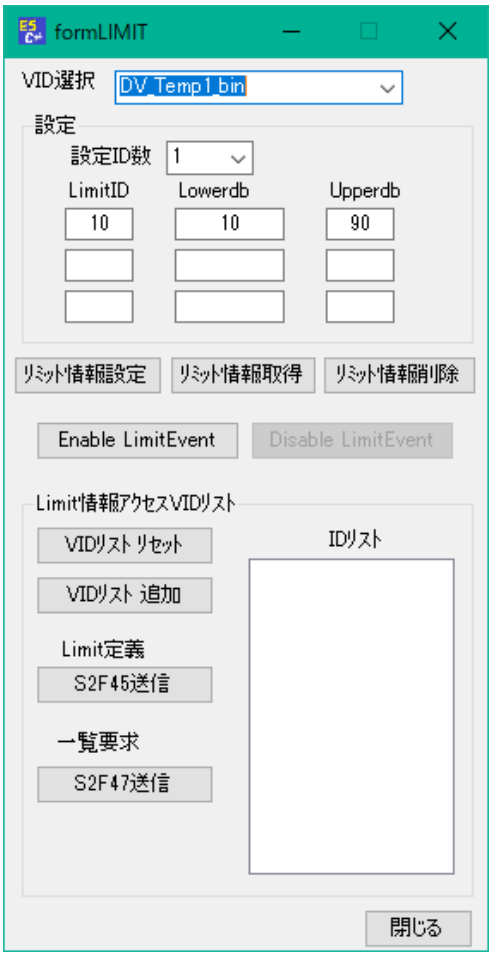
３．２．５　変数リミット操作画面

ホストなしでローカルに変数リミット監視機能をテストできます。
変数リミット操作画面は次のようになります。

それぞれのボタンの機能は、ボタンの表示名のとおりです。
操作結果は、LOG 画面に表示されます。

装置側選択

ﾎｽﾄ側選択



以下のクラスを使用します。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_V	get_id_list()
	TLIMIT_INFO	set_vid(), clear(), set(), get(), add_limitid, delete()
	TLIMIT_LIST	add_limitid()など

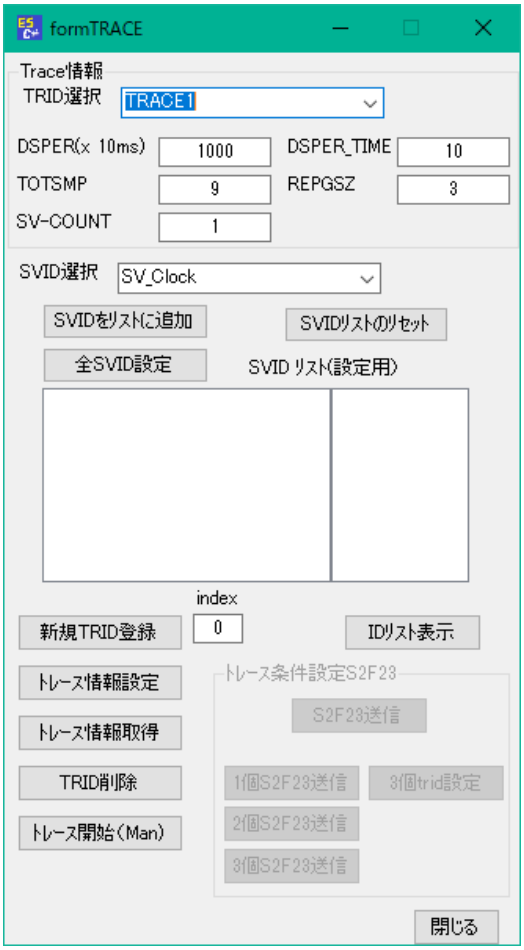
３．２．６　SV トレース操作画面

ホストなしでローカルに SV トレース機能をテストできます。

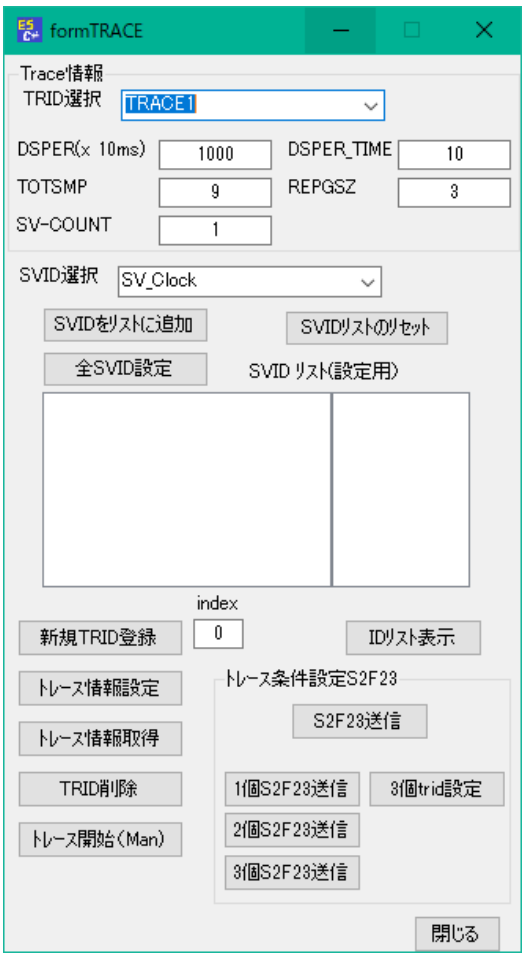
装置状態変数 SV 操作画面は次のようになります。
それぞれのボタンの機能は、ボタンの表示名のとおりです。

操作結果は、LOG 画面に表示されます。

装置側選択



ﾎｽﾄ側選択



以下のクラスを使用します。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_SV	get_id_list(), get(), set()他
	class_TRACE	get(), set()

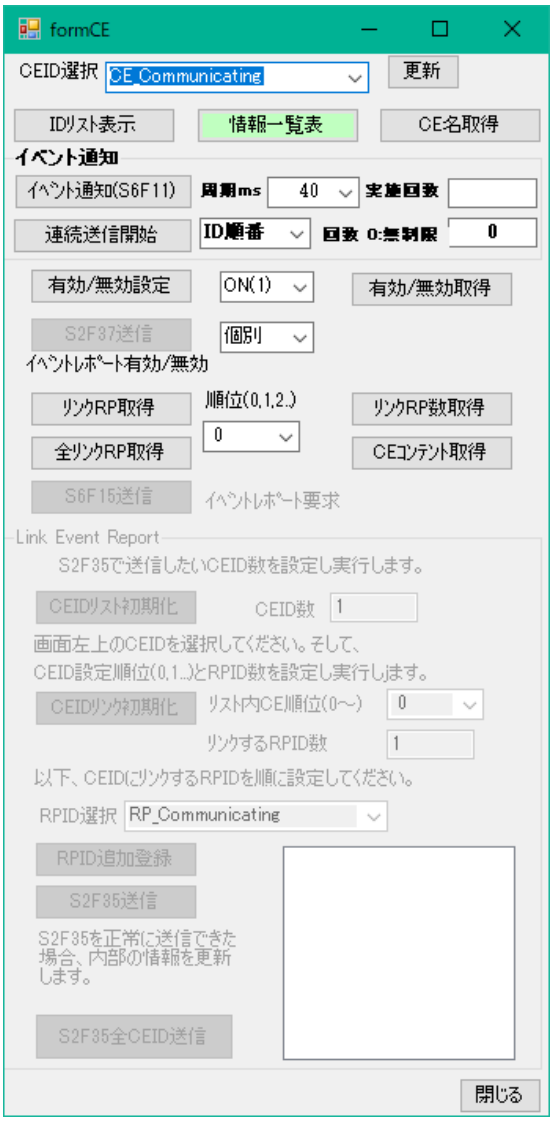
3．3　CE 収集イベントと RP レポート操作画面

CE 収集イベント情報と RP レポート情報の操作のための画面です。

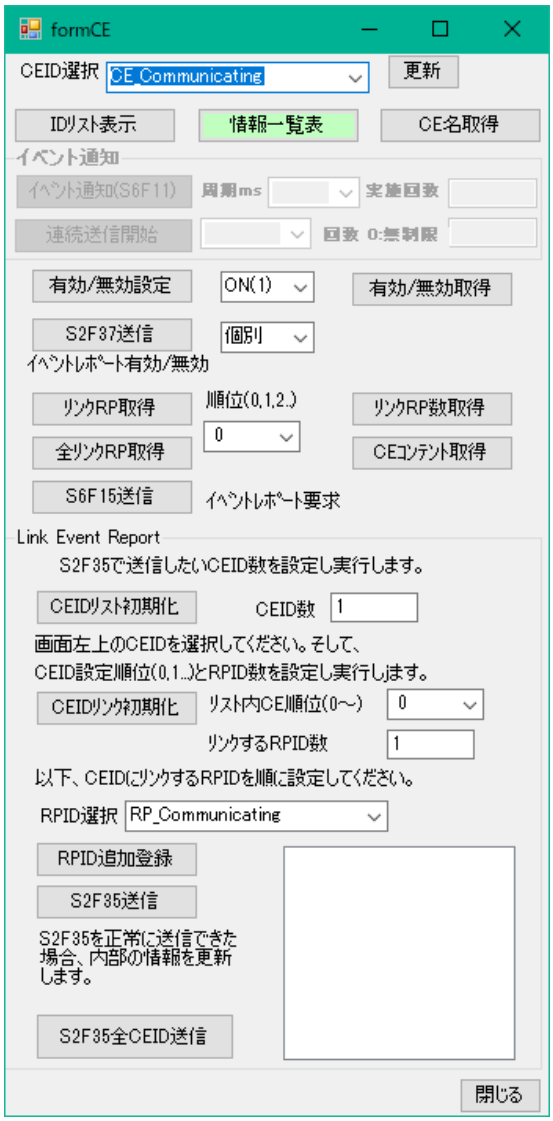
3．3．1　CE 収集イベント情報操作画面

それぞれのボタンの機能は、ボタンの表示名のとおりです。
操作結果は、LOG 画面に表示されます。

装置側選択



ホスト側選択



以下のクラスを使用します。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	c	get_id_list(), set_ceid(), set_enabled(), get_content()他

S6F11 の送信は、ｲﾍﾞﾝﾄ通知のグループボックス内のボタンを使って行います。ceid のコンボボックスで CEID を選択し、イベント通知(S6F11)ボタンのクリックで行います。

送信モードには、非ブロック、ブロックの２つのモードがあります。いずれでも送信できます。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ	callback のｸﾗｽ	動作
ｲﾍﾞﾝﾄ通知(S6F11)	class_SendS6F11	SendS6F11() SendS6F11_wait()	class_CALLBACK. callback_S6F12	選択された CEID の S6F11 を非ﾌﾞﾛｯｸﾓｰﾄﾞで送信する。 同様にﾌﾞﾛｯｸﾓｰﾄﾞで送信する。
連続送信開始 連続送信停止	同上	同上	同上	連続的に S6F11 を送信します。 送信を停止します。

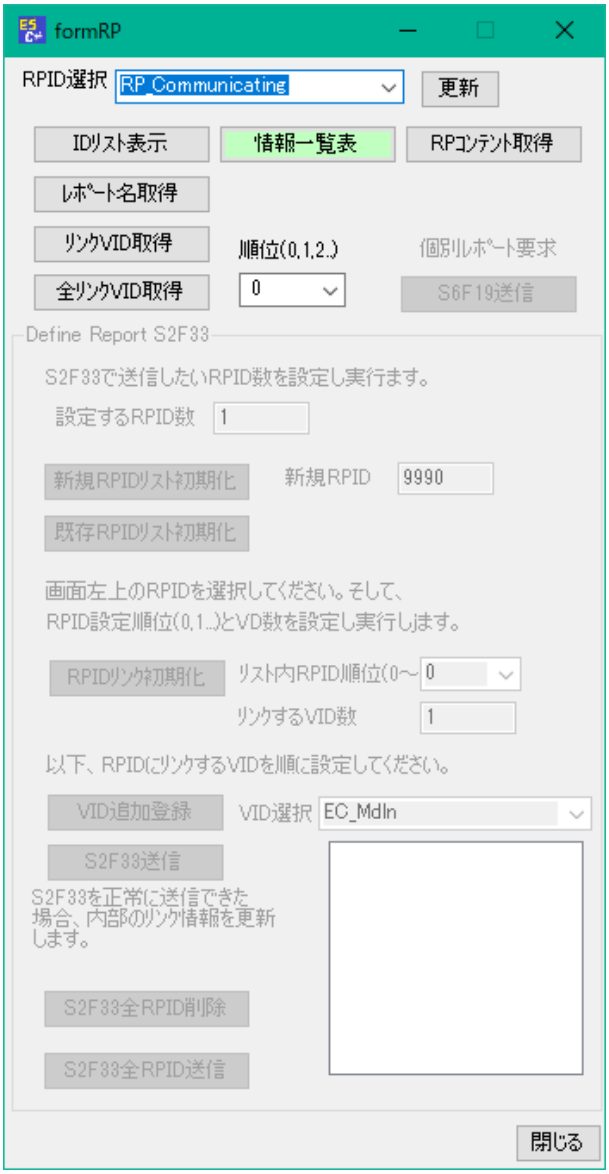
３．３．３．１　DataGridView コントロールを使った一覧表画面

３．２．１．１　(EC の画面) を参照ください。

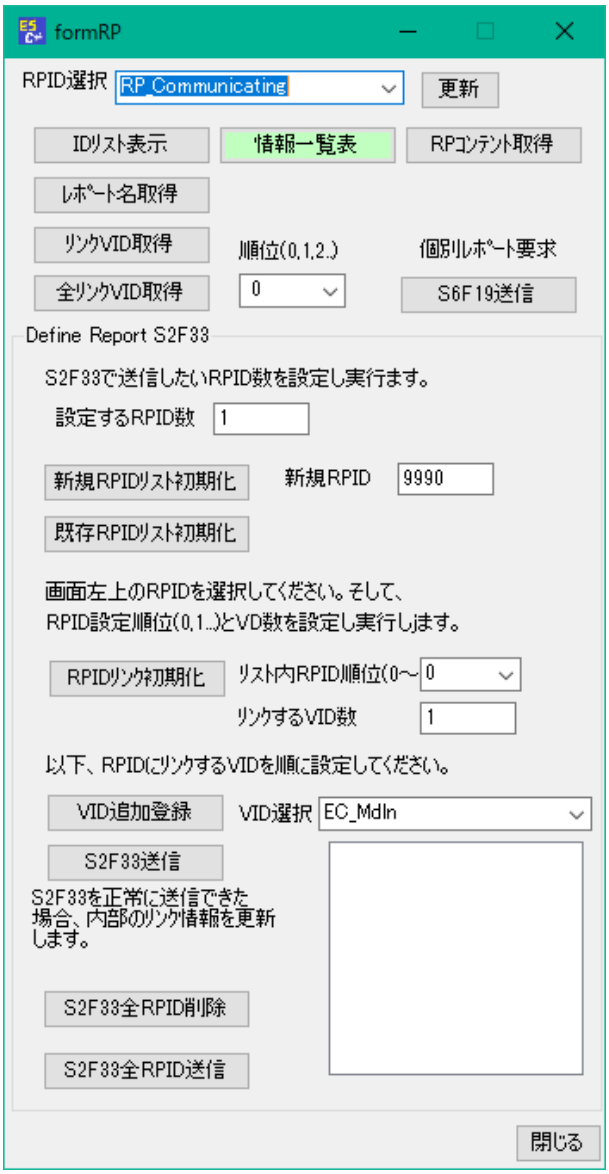
３．３．２　RP レポート情報操作画面

それぞれのボタンの機能は、ボタンの表示名のとおりです。装置側では、S2F33, S6F19 の送信は行いません。
操作結果は、LOG 画面に表示されます。

装置側選択



ホスト側選択



以下のクラスを使用します。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_Report	get_id_list(), set_rpid(), get_content()他

３．３．２．１　DataGridView コントロールを使った一覧表画面

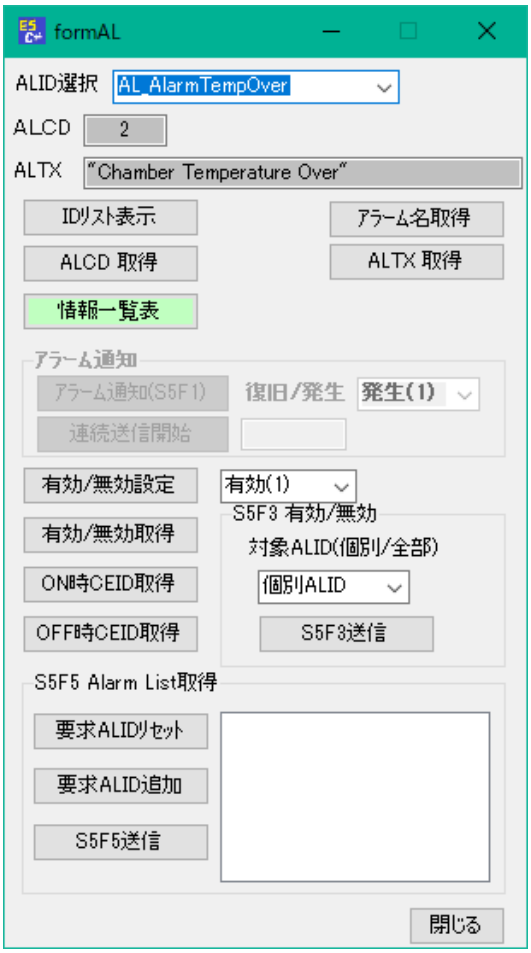
３．２．１．１　(EC の画面) を参照ください。

３．４　アラーム操作画面

AL アラーム通知情報の操作画面です。
それぞれのボタンの機能は、ボタンの表示名のとおりです。
操作結果は、LOG 画面に表示されます。

装置側選択

ホスト側選択



以下のクラスを使用します。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_AL	get_id_list(), set(), get(), set_enabled()他

S5F1 の送信は、ｱﾗｰﾑ知のグループボックス内のボタンを使って行います。alid のコンボボックスで　ALID を選択し、ｱﾗｰﾑ通知(SF1)ボタンのクリックで行います。

送信モードには、非ブロック、ブロックの２つのモードがあります。いずれでも送信できます。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ	callback のｸﾗｽ	動作
ｱﾗｰﾑ通知(S5F1)	class_SendS5F1	SendS5F1()	class_CALLBACK. callback_S5F2	選択された ALD の S5F1 を非ﾌﾞﾛｯｸﾓｰﾄﾞで送信する。
		SendS5F1_wait()	なし	同様にﾌﾞﾛｯｸﾓｰﾄﾞで送信する。
連続送信開始	同上	同上	同上	連続的に S5F1 を送信します。
連続送信停止				送信を停止します。

３．４．１　DataGridView コントロールを使った一覧表画面

３．２．１．１　(EC の画面) を参照ください。

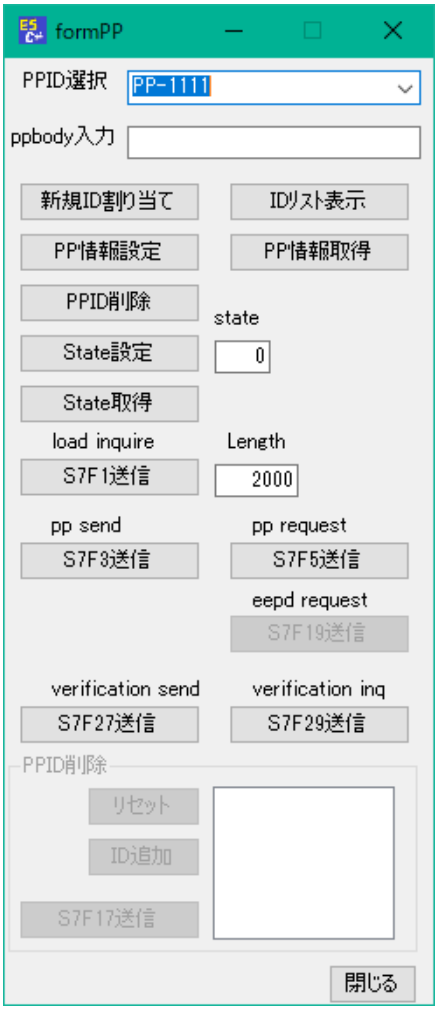
３．５　PP プロセス・プログラムと RCP レシピ情報操作画面

DSHEng5 エンジンでは、プロセス・プログラムとレシピの双方の機能サポートを行います。
それぞれの操作画面について記述します。

３．５．１　PP プロセス・プログラム操作画面

PP プロセス・プログラム情報の操作画面です。新規 ID 登録、指定された ID に対する情報設定/取得アクセスができます。　また、関連するメッセージの送信もあります。
それぞれのボタンの機能は、ボタンの表示名のとおりです。操作結果は、LOG 画面に表示されます。

装置側選択



ﾎｽﾄ側選択

以下のクラスを使用します。

（1）情報アクセス関連

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_PP	get_id_list(), alloc_id(), set_ppbody(), set(), get(), delete()他

（メッセージ送信は次ページ）

（2）メッセージ送信関連

下表のメッセージの送信を行います。
送信モードには、非ブロックとブロックの２つのモードがありますが、ここでは、send()メソッドを使って非ブロックモードでプログラミングしています。

（ブロックモードでは、 send_wait()メソッドを使用します。）

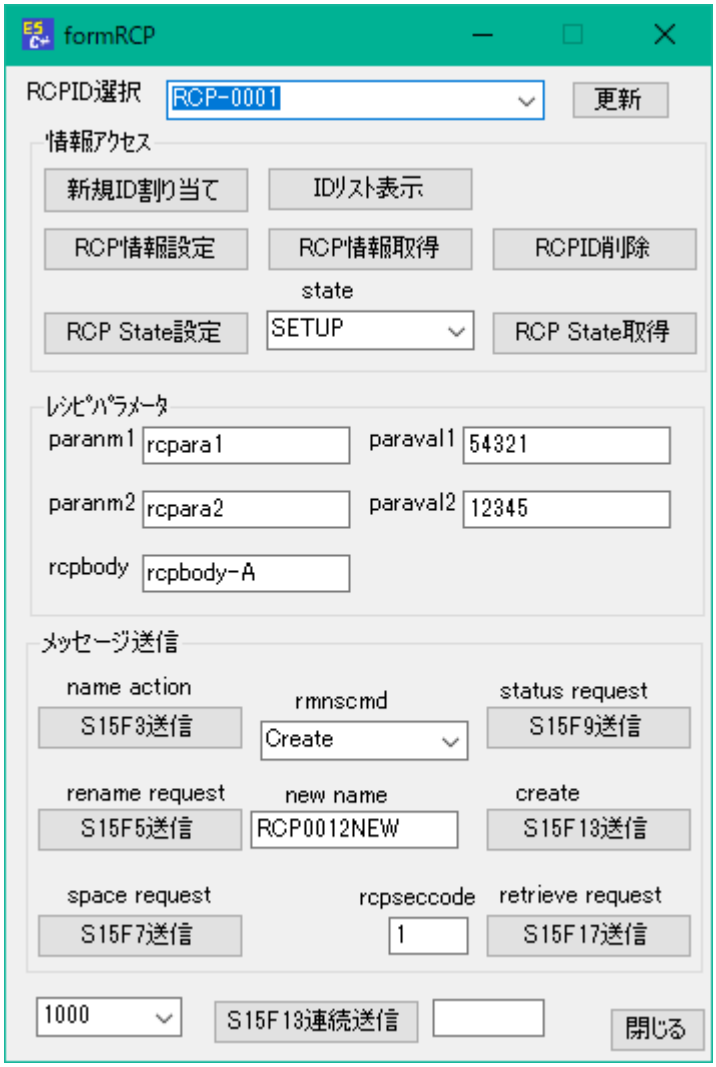
ﾎﾞﾀﾝ	情報ｸﾗｽ	MSG 送信ｸﾗｽ	Msg 受信ｸﾗｽ	送信完了 callback ｸﾗｽ
S7F1 送信	class_SendS7F1	SendS7F1() SendS7F1_wait()	-	class_CALLBACK.callback_S7F2
S7F3 送信	class_SendS7F3	SendS7F3() SendS7F3_wait()	-	class_CALLBACK.callback_S7F4
S7F5 送信	class_SendS7F5	SendS7F5() SendS7F5_wait()	class_PP	class_CALLBACK.callback_S7F6
S7F19 送信	class_SendS7F19	SendS7F19() SendS7F19_wait()	-	class_CALLBACK.callback_S7F20
S7F27 送信	class_SendS7F27	SendS7F27() SendS7F27_wait()	-	class_CALLBACK.callback_S7F28
S7F29 送信	class_SendS7F29	SendS7F29() SendS7F29_wait()	-	class_CALLBACK.callback_S7F30

３．５．２　RCP レシピ操作画面

レシピ情報の操作画面です。新規 ID 登録、指定された ID に対する情報設定/取得アクセスができます。　また、関連するメッセージの送信もあります。

それぞれのボタンの機能は、ボタンの表示名のとおりです。操作結果は、LOG 画面に表示されます。

装置側選択　＆　ﾎｽﾄ側選択



下のクラスを使用します。

（1）情報アクセス関連

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_RCP	get_id_list(), alloc_id(), set_id(), set(), get(), delete()他

（メッセージ送信は次ページ）

（2）メッセージ送信関連

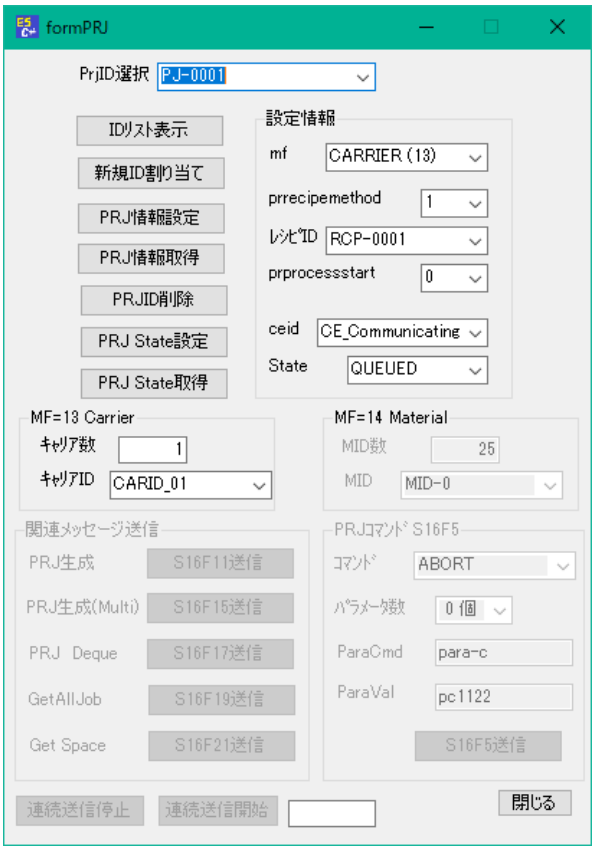
下表のメッセージの送信を行います。
送信モードには、非ブロックとブロックの２つのモードがありますが、ここでは、send()メソッドを使って非ブロックモードでプログラミングしています。

ﾎﾞﾀﾝ	MSG 送信ｸﾗｽ	Msg 受信ｸﾗｽ	送信完了 callback ｸﾗｽ
S15F3 送信	class_SendS15F3	SendS15F3() SendS15F3_wait()	class_CALLBACK.callback_S15F4
S15F5 送信	class_SendS15F5	SendS15F5() SendS15F5_wait()	class_CALLBACK.callback_S15F6
S15F7 送信	class_SendS15F7	SendS15F7() SendS15F7_wait()	class_CALLBACK.callback_S15F8
S15F9 送信	class_SendS15F9	SendS15F9() SendS15F9_wait()	class_CALLBACK.callback_S15F10
S15F13 送信	class_SendS15F13	SendS15F13() SendS15F13_wait()	class_CALLBACK.callback_S15F14
S15F17 送信	class_SendS15F17	SendS15F17() SendS15F17_wait()	class_CALLBACK.callback_S15F18

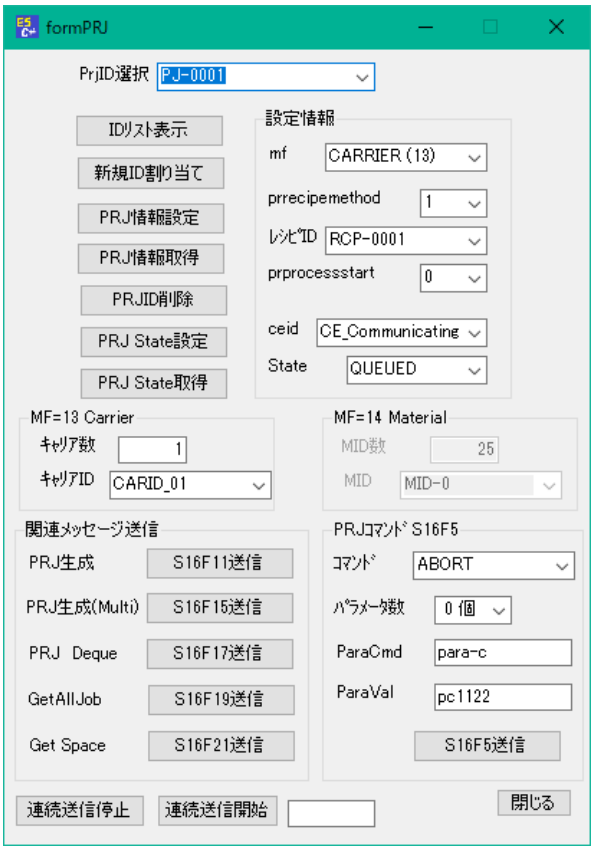
３．６　PRJ プロセス・ジョブ操作画面

PRJ プロセス･ジョブ情報操作画面です｡新規 ID 登録､指定された ID に対する情報設定/取得アクセスができます。
それぞれのボタンの機能は、ボタンの表示名のとおりです。
操作結果は、LOG 画面に表示されます。

装置側選択



ホスト側選択



以下のクラスを使用します。

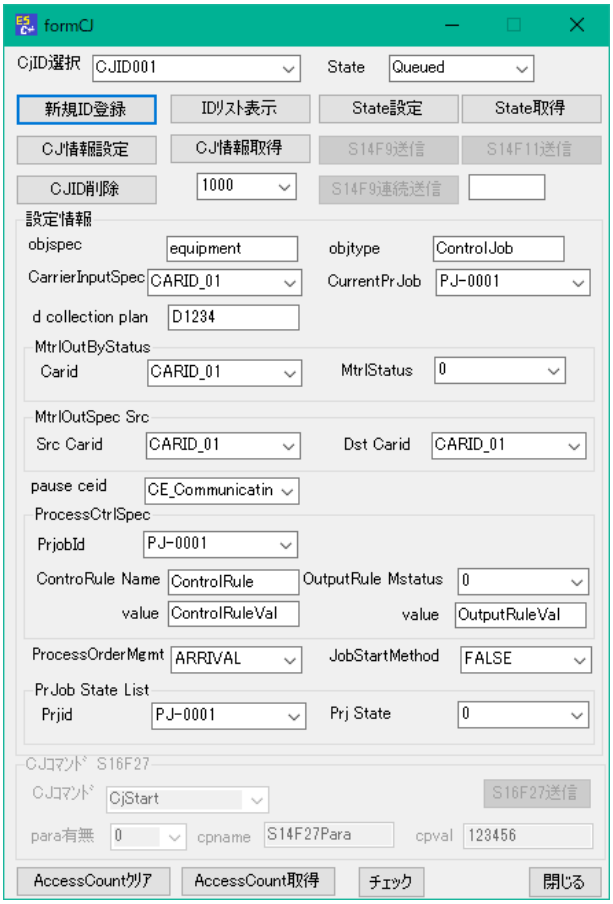
ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_PRJ	get_id_list(), alloc_id(), set_id(), set(), get(), delete()他

3．7　CJ コントロール・ジョブ操作画面

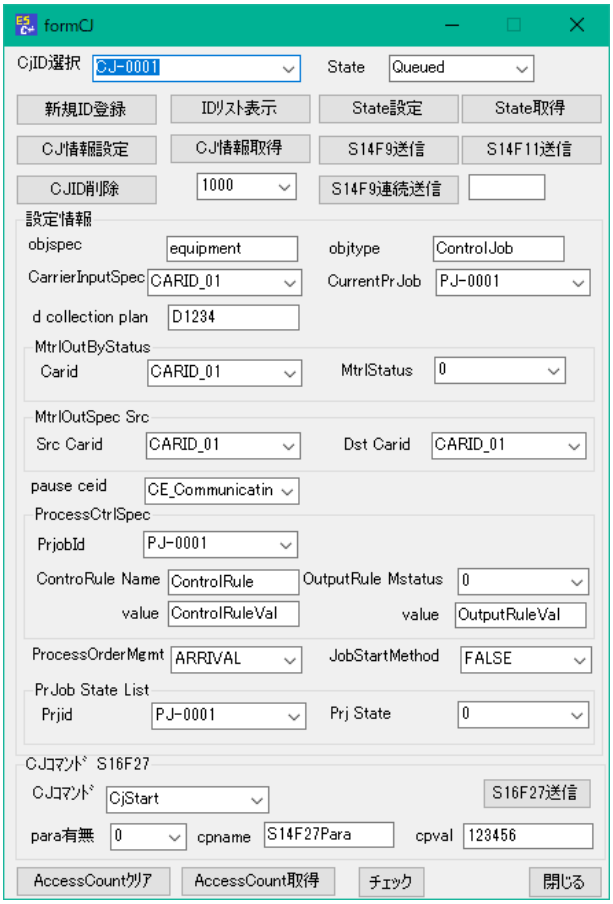
CJ コントロール・ジョブ情報操作画面です。新規 ID 登録、指定された ID に対する情報設定/取得アクセスができます。

それぞれのボタンの機能は、ボタンの表示名のとおりです。操作結果は、LOG 画面に表示されます。

装置側選択



ﾎｽﾄ側選択



以下のクラスを使用します。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	class_CJ	get_id_list(), alloc_id(), set_id(), set(), get(), delete()他

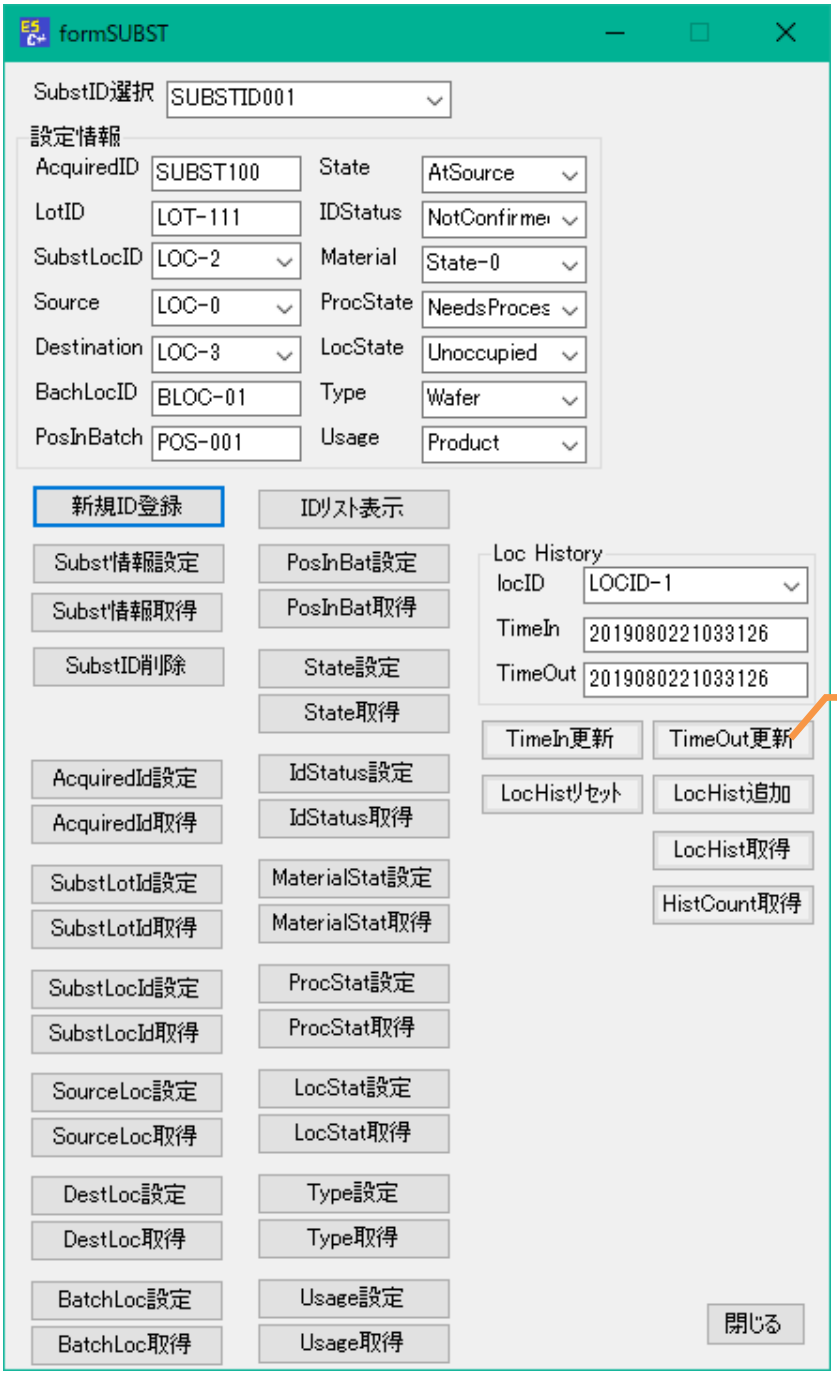
３．８　Carrier キャリア操作画面

Carrier キャリアの操作画面です。新規 ID 登録、指定された ID に対する情報設定/取得アクセスができます。
それぞれのボタンの機能は、ボタンの表示名のとおりです。
操作結果は、LOG 画面に表示されます。

キャリアの管理クラスは、 class_CAR です。

３．９　Substrate 基板操作画面

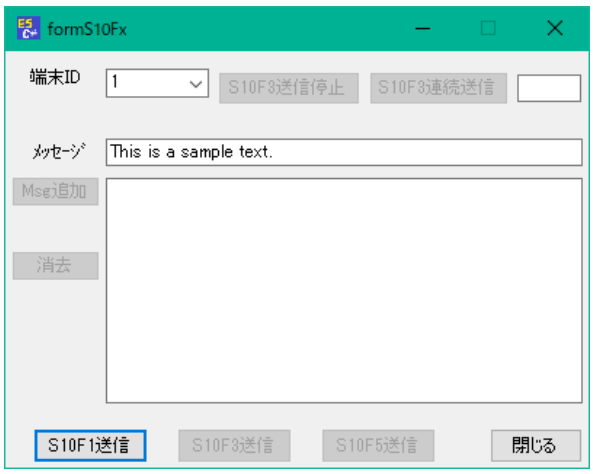
Substrate 基板の操作画面です。新規 ID 登録、指定された ID に対する情報設定/取得アクセスができます。
それぞれのボタンの機能は、ボタンの表示名のとおりです。
操作結果は、LOG 画面に表示されます。

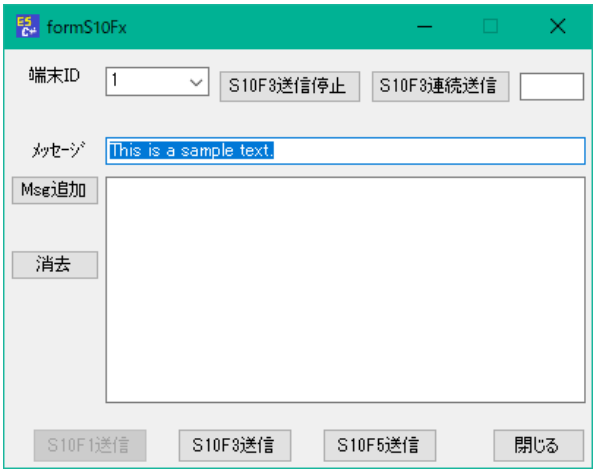


基板の管理クラスは、 class_SUBST です。

3．10　端末メッセージ送信画面

装置側から S10F1，S10F3，S10F5 メッセージを送信するための画面です。





送信モードには、非ブロック、ブロックの２つのモードがあります。いずれでも送信できます。

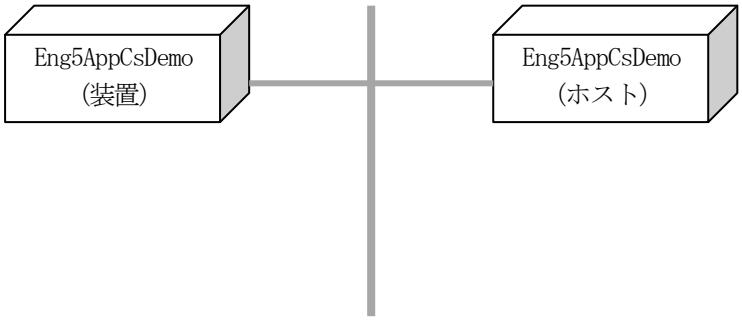
以下のクラスとメソッドを使用します。

ﾒｯｾｰｼﾞ	情報ｸﾗｽ	Msg 送信ｸﾗｽ	送信ﾒｿｯﾄﾞ	送信完了 callback ｸﾗｽ
S10F1	TTERM_INFO	classSendS10F1	SendS10F1() SendS10F1_wait()	class_CALLBACK. callback_S10F2()
S10F3	TTERM_INFO	classSendS10F3	SendS10F3() SendS10F3_wait()	class_CALLBACK. callback_S10F4()
S10F5	TTERM_MULTI_INFO	classSendS10F5	SendS10F5() SendS10F5_wait()	class_CALLBACK. callback_S10F6()

３．11　WP ウェハー・プロセス・シミュレーション操作画面

簡単な WP ウエハー・プロセス・シナリオのモデルを作り、それをホストとの通信で実現します。

本 WP シミュレーションを実現するためには、ホスト側の PC を準備し、その上で、GDSHGemLib 通信エンジンのホスト機能を利用する GemCsDemoV2 デモプログラムが必要になります。また、DSHEng5 エンジンの状態を監視する装置エンジンモニターも使用することができます。(何れも試用版で可能)



ロードからアンロードまでのシナリオの処理内容については、**６．WP ｼﾐｭﾚｰｼｮﾝの処理**　を参照ください。

３．11．１　WP 処理の操作

実際のシミュレーションの開始は、ホストとは GEM レベルでの通信が確立し、ホストが WP シミュレーションの準備ができていることが前提で、メイン画面の下部にある WP 処理ｼﾅﾘｵ操作　内の **WP 開始**ボタンのクリックで行います。

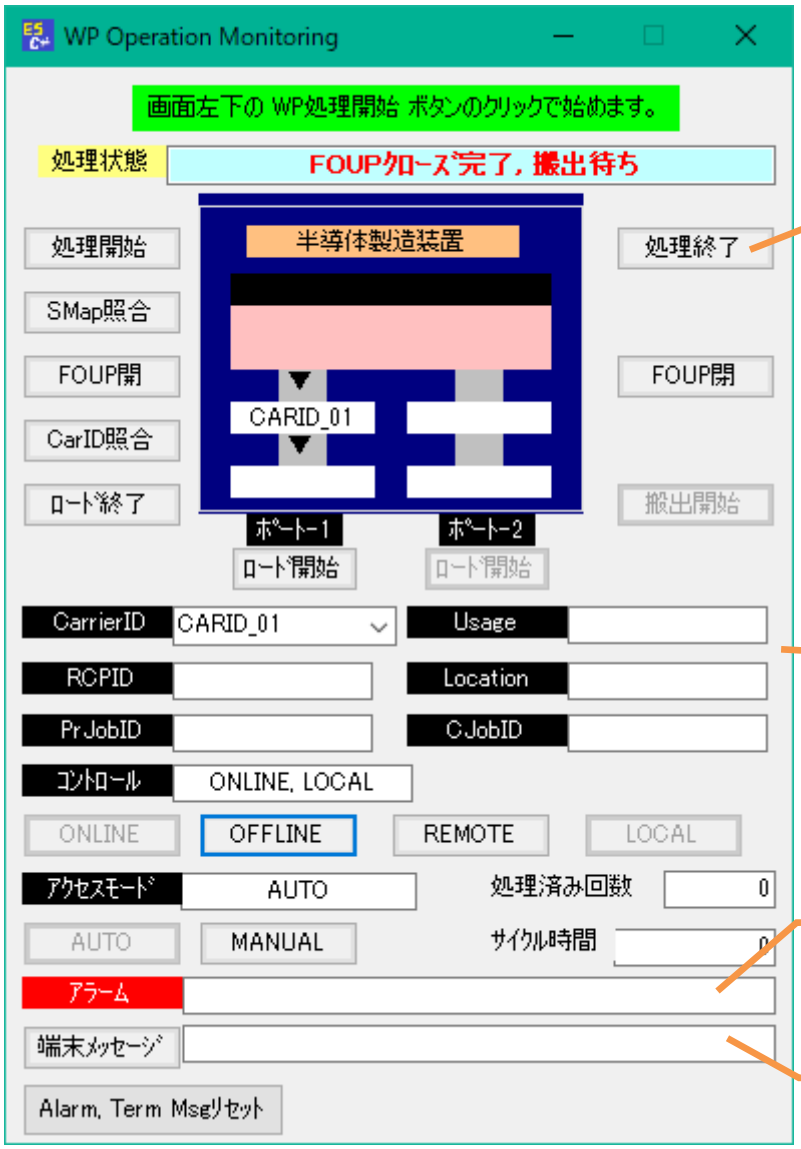
各部の機能は下表のとおりです。

ﾎﾞﾀﾝ/ﾗﾝﾌﾟ	機能	注釈
WP 開始	WP ｼﾐｭﾚｰｼｮﾝを開始する。	処理を開始し、**WP 停止**、**終了予約**を有効にする。 **WP 開始**は無効にする。ﾓﾆﾀｰ画面も表示する。
WP 停止	WP ｼﾐｭﾚｰｼｮﾝを強制停止する。	処理を停止する。 (**WP 開始**ﾎﾞﾀﾝで最初から開始できる)
終了予約	WP ｼﾐｭﾚｰｼｮﾝを、次に搬出完了後、WP 処理を終了する。	
状況表示	Load, Process, Unload の状態値をﾛｸﾞ画面に表示する。	
ﾓﾆﾀｰ画面	ﾓﾆﾀｰ画面を表示する。	次ﾍﾟｰｼﾞに示すﾓﾆﾀｰ画面を表示する。 (一旦消した画面を再表示する)
■Load busy	ｷｬﾘｱ Load 中	Busy になったら緑色になる。
■Process busy	ｷｬﾘｱ処理中	〃
■Unload busy	ｷｬﾘｱ搬出中	〃

3．11．２　WP シミュレーション画面

WP 処理シミュレーションの進行状況と、そのときの処理条件などの情報をモニタリング表示するための画面です。

進行状態は、処理状態、ボタンの有効表示そして、仕掛情報の表示の更新を行います。
処理した回数、１サイクルに要した時間の表示も行います。



ﾛｰﾄﾞ開始->ﾛｰﾄﾞ終了->...
->搬出開始　ﾎﾞﾀﾝが、それぞれの処理が終了したら Enable になります。これらのﾎﾞﾀﾝはｸﾘｯｸしてもなにもしません。

処理中のｷｬﾘｱ ID など処理条件情報が表示される。

Alarm 画面から S5F1 が送信された内容が ALTX が表示される。

端末ﾒｯｾｰｼﾞ画面から S10F1 が送信された Message が表示される。

通信シナリオなどについては、６．で説明します。

３．12　スプール設定操作画面

メッセージのスプールの設定操作画面です。

stream を選び、同じストリームの function を順次リストに追加し、それを設定します。

（本来は、Spool 設定はホストからの S2F43 メッセージによって行われますが、ここでは、装置側で設定する例です。）

以下のクラスを使用します。

ボタン	ｸﾗｽ	ﾒｿｯﾄﾞ
情報ｱｸｾｽ関連	DshSpool	get(), set(), add_func(), reset(), clear_stream()他

３．13　１次メッセージの送信操作画面

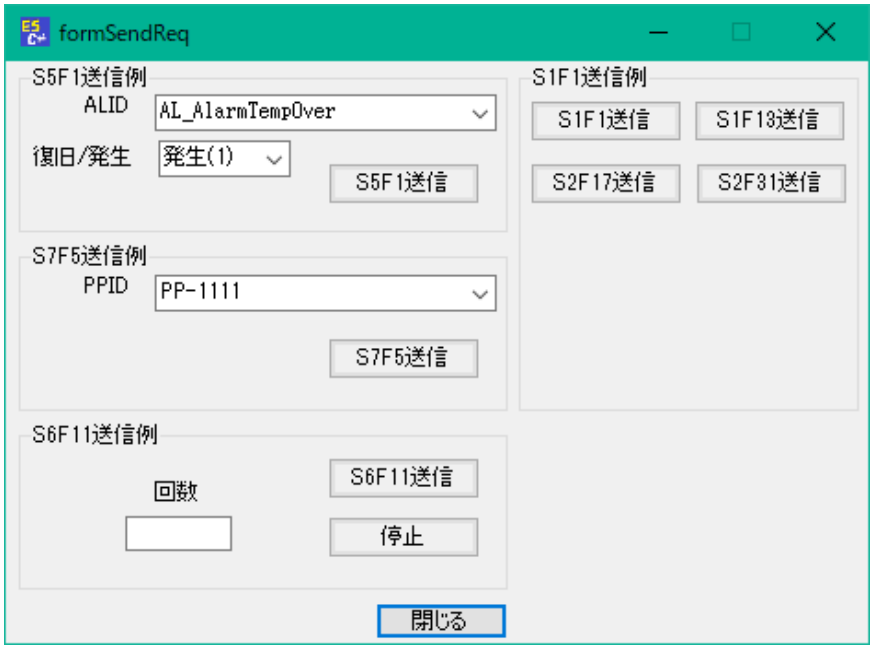
DSHEng5 エンジンで標準としてサポートしていないユーザ固有の１次メッセージを、ユーザが DSHDR2 通信ドライバーのメッセージ組み立て機能を使って準備し、**EngAPI クラスの send_request()または、send_request_wait()メソッド**を使って、その１次メッセージを送信するサンプルを含む画面です。

例として、とりあえず、S5F1, S7F5, S1F1, S1F13, S2F17 の送信例が示されています。

デモプログラム formSendReq フォームには、メッセージ組み立てと送信、そして２次メッセージの受信と処理のコーディングが含まれています。
使用するクラスは次のとおりです。・

ｸﾗｽ	ﾒｿｯﾄﾞ	構造体	備考
HSMS	D_InitItemPut(), D_PutItem()	DSHMSG	1 次 MSG 組立
	D_InitItemGet(), D_GetItem()	DSHMSG	2 次 MSG 解読
EngAPI	send_request() send_request_wait()		1 次 MSG 送信 非ﾌﾞﾛｯｸﾓｰﾄﾞ ﾌﾞﾛｯｸﾓｰﾄﾞ
class_CALLBACK	callback_send_request		send_request()の場合にのみ必要

画面



３．14　１次メッセージに対する応答 ACK 設定画面

ホストから受信する 1 次メッセージに返す応答メッセージに、故意に Ack（ 1 バイト)情報をエラーの値に設定しするための画面です。

本画面に表示されているメッセージを受信した時に、ここで設定した ACK の値が応答メッセージにセットされ、送信されます。

装置側が応答するメッセージに対するものだけです。

操作画面

４．装置情報と通信メッセージ

ここでは、本デモプログラムが、装置ーホスト間の通信確立シナリオならびに、ウェハー処理シナリオ実行時に使用するメッセージならびに変数のデータフォーマットなどについて記述します。

（注） DSHEng5 は下表以外の GEM で使用するメッセージの送受信をサポートしています。

４．１　メッセージ

通信確立、ウエハー処理シナリオなどで使用するメッセージは次の通りです。

メッセージ一覧表

1 次 MSG	2 次 MSG	方向	役　割
S1F13	S1F14	H⟷E	通信確立(ｴﾝｼﾞﾝが実施)
S1F15	S1F16	H→E	ｵﾌﾗｲﾝ要求
S1F17	S1F18	H→E	ｵﾝﾗｲﾝ要求
S3F17	S3F18	H→E	ｷｬﾘｱｱｸｼｮﾝ要求
S3F27	S3F28	H→E	Change Access
S5F1	S5F2	H←E	ｱﾗｰﾑ報告
S6F11	S6F12	H←E	ｲﾍﾞﾝﾄﾚﾎﾟｰﾄ送信
S10F1	S10F2	H←E	端末要求
S10F3	S10F4	H→E	端末表示、ｼﾝｸﾞﾙﾌﾞﾛｯｸ
S10F5	S10F6	H→E	端末表示、ﾏﾙﾁﾌﾞﾛｯｸ
S14F9	S14F10	H→E	Create Object Request(Cj)
S14F11	S14F12	H→E	Delete Object Request(Cj)
S15F13	S15F14	H→E	Recipe Create Request
S16F11	S16F12	H→E	PrJob Create Enh

なお、メッセージに含まれる CEID, RPID, VID のデータフォーマットが符号無し４バイトか、符号無し２バイトのどちらかを選択できます。

４．２　変数一覧

変数（ＥＣ，ＳＶ，ＤＶＶＡＬ）について説明します。

（この定義情報は、EQINFO.TXT ファイルに保存されています。）

デモ用の参考例として定義されているたため、目的、変数値についてあまり意味がないものも含まれます。

４．２．１　ＥＣ - 装置定数

#	NAME	ID	Format	値	UNIT	備考
1	EC_Mdln	1	A[6]			S1F13 など用
2	EC_SoftRev	2	A[6]			S1F13 など用
3	EC_InitCommState	4	U2[1]	0～5		
4	EC_InitControlState	5	U2[1]	0～1		0=offline, 1=online
5	EC_InitOfflineSubState	6	U2[1]	1～3		
6	EC_ControlMode	80	U2[1]	0～2		
7	EC_PortAccessMode	256	U2[1]	0～2		
8	EC_Port1AccessMode	257	U2[1]	0～2		
9	EC_Port2AccessMode	258	U2[1]	0～2		
10	EC_ManualTimer	275	U1[1]	0～255		
11	EC_ManualInType	276	U1[1]			
12	EC_ManualOutType	278	U1[1]			
13	EC_MaxSpoolTransmit	512	U4[1]			
14	EC_OverWriteSpool	513	Boolean[1]			
15	EC_BinTest	1024	B[1]			ﾌｫｰﾏｯﾄﾃｽﾄ用
16	EC_Int1Test	1025	I1[1]			ﾌｫｰﾏｯﾄﾃｽﾄ用
17	EC_Int2Test	1026	I2[1]			ﾌｫｰﾏｯﾄﾃｽﾄ用
18	EC_Int4Test	1027	I4[1]			ﾌｫｰﾏｯﾄﾃｽﾄ用
19	EC_SingleTest	1028	F4[1]			ﾌｫｰﾏｯﾄﾃｽﾄ用
20	EC_DoubleTest	1029	F8[1]			ﾌｫｰﾏｯﾄﾃｽﾄ用

４．２．２　ＳＶ－装置状態変数

#	NAME	ID	Format	値	UNIT	備考
1	SV_Clock	8192	A[16]			YYYYMMDDHHMMSSCC 年　月日時分秒 100ms
2	SV_CommunicationState	8193	U1[1]	0～5		0=disabled 1=enabled 2=not_communicating 3=wait CRA 4=wait delay 5=Communicating
3	SV_ControlState	8194	U2[1]	0～2		0=offline, 1=online local 2=online remote
4	SV_CJExecName1	8195	A[0..80]		CJOB	CJ 名 port-1
5	SV_CJExecName2	8196	A[0..80]		CJOB	CJ 名 port-2
6	SV_ReadyToLoad	8198	A[0..80]			ﾛｰﾄﾞ可能状態文字列
7	SV_LoadPortTransferStatus	8199	U1[1]	0～3		0=out of service 1=blocked 2=ready to load 3=ready to unload
8	SV_LoadPortId	8200	U1[1]	1～2		現ﾛｰﾄﾞﾎﾟｰﾄ番号
9	SV_LoadCarId	8201	A[0..32]			ﾛｰﾄﾞｷｬﾘｱ ID
10	SV_LoadCarIdStatus	8202	U1[1]	0～4		ｷｬﾘｱ ID 読取り状態 0=not read 1=wait for host 2=Id verification OK 3=Id verification faild 4=carid not exists
11	SV_LoadSlotMapStatus	8203	U1[1]	0～3		ｽﾛｯﾄﾏｯﾌﾟ 読取り状態
12	SV_LoadPortAssociationStatus	8204	U1[1]	0～1		0=not associated 1=associated
13	SV_LoadCarAccessStatus	8205	U1[1]	0～4		ｷｬﾘｱ処理ｱｸｾｽ状態 0=procss not accessed 1=process in acess 2=process complete 3=process stopped 4=proc not exists
14	SV_LotId	8207	A[0..32]			ﾛｯﾄ番号
15	SV_SubstID	8208	A[0..32]			基板 ID（未使用）
16	SV_SubstProcState	8209	U1[1]	0～1		基板処理ｱｸｾｽ状態（未使用）
17	SV_SubstState	8210	U1[1]			基板状態(未使用)
18	SV_CarUsage	8211	A[0..32]			ｷｬﾘｱ用途
19	SV_CarLocation	8212	A[0..32]			ｷｬﾘｱ所在位置
20	SV_SubstList	82131	L	1		RPTID link nest 用 list

21	SV_UnloadPortId	8214	U1[1]	1～2		ｱﾝﾛｰﾄﾞﾎﾟｰﾄ
22	SV_UnloadCarId	8215	A[0..32]			ｱﾝﾛｰﾄﾞｷｬﾘｱ ID
23	SV_UnloadCarIdStatus	8216	U1[1]	0～5		ｱﾝﾛｰﾄﾞｷｬﾘｱ状態(未使用)
24	SV_UnloadSlotMapStatus	8217	U1[1]	0～5		ｱﾝﾛｰﾄﾞｷｬﾘｱﾏｯﾌﾟ 状態(未使用)
25	SV_UnloadPortAssociationStatus	8218	U1[1]	0～5		ｱﾝﾛｰﾄﾞｷｬﾘｱｱｿｼｴｰｼｮﾝ状態 (未使用)
26	SV_UnloadCarAccessStatus	8219	U1[1]	0～5		ｱﾝﾛｰﾄﾞｷｬﾘｱｱｸｾｽ状態(未使用)
27	SV_AccessModeStatus	8220	U1[1]			(未使用)
28	SV_Access1ModeStatus	8221	U1[1]			(未使用)
29	SV_Access2ModeStatus	8222	U1[1]			(未使用)
30	SV_Port1ReservedStatus	8224	U1[1]			
31	SV_Port2ReservedStatus	8225	U1[1]			
32	SV_PPID	8226	A[1..80]			Process Program ID (=recipe id) S7F3/S7F23/S15F13 から得る
33	SV_PrjID	8227	A[1..80]			Process Job ID S16F11/S16F15 より得る
34	SV_PrjState	8228	A[1..80]			Process Job の状態 (文字列)
35	SV_CJID	8229	A[1..80]			Control Job ID S14F9 より得る
36	SV_CjState	8230	A[1..80]			Control Job の状態 (文字列)
37	SV_SpoolState	8231	U1[1]			ｽﾌﾟｰﾙの状態 DSHEng5 が内部で使用する
38	SV_SpoolCountActual	8232	U4[1]			ｽﾌﾟｰﾙ数 DSHEng5 が内部で使用する
39	SV_SpoolCountTotal	8233	U4[1]			ｽﾌﾟｰﾙｶｳﾝﾄ合計 DSHEng5 が内部で使用する
40	SV_SpoolFullTime	8234	A[16]			ｽﾌﾟｰﾙ満杯時刻記憶 DSHEng5 が内部で使用する
41	SV_SpoolStartTime	8235	A[16]			ｽﾌﾟｰﾙ開始時刻 DSHEng5 が内部で使用する
42	SV_WaferTransportState	8237	U1[1]			(未使用)
43	SV_WaferProcesStatus	8238	U1[]1			(未使用)
44	SV_WaferLocationStatus	8239	U1[1]			(未使用)
45	SV_List1	254	L[1]	5		CE ｲﾍﾞﾝﾄ内､ﾚﾎﾟｰﾄ ID にﾘﾝｸされる SV ﾘｽﾄ数
46	SV_List1_End	255	LEND[1]	0		同 SV ﾘｽﾄの終わり
47	SV_List2	252	L[1]	3		CE ｲﾍﾞﾝﾄ内､ﾚﾎﾟｰﾄ ID にﾘﾝｸされる SV ﾘｽﾄ数
48	SV_List2_End	253	LEND[1]	0		同 SV ﾘｽﾄの終わり
49	SV_LIST	251	LEND[1]	0		SV ﾘｽﾄの終わり

４．２．３　ＤＶＶＡＬ- 装置データ変数

#	NAME	ID	Format	値	UNIT	備考
1	DV_StartTime	8300	A[19]			処理開始時刻 YYYY-MM-DD-HH:MM:SS
2	DV_EndTime	8301	A[19]			処理終了時刻 YYYY-MM-DD-HH:MM:SS
3	DV_Temp1	8302	A[8]		Degree	温度１（ASCII）
4	DV_Temp1_bin	8303	U2[1]		Degree	温度１（Binary）
5	DV_Temp1	8304	A[8]		Degree	温度２(ASCII)
6	DV_Temp1_bin	8305	U2[1]		Degree	温度２(Binary)

４．３　ＣＥイベントＩＤ

#	NAME	ID	ﾘﾝｸﾚﾎﾟｰﾄ ID	備考
1	CE_Communicating	2	RP_Communicating	通信確立通知
2	CE_SpoolDeactivated	999	(なし)	ｽﾌﾟｰﾙ終了通知
3	CE_ControlState	100	RP_ControlState	
4	CE_AlarmOn	200	(なし)	
5	CE_AlarmOff	201	(なし)	
6	CE_PortAccessMode	2200	RP_PortAccessMode	
7	CE_Port1AccessMode	2201	RP_Port1AccessMode	
8	CE_Port2AccessMode	2202	RP_Port2AccessMode	
9	CE_ReadyToLoad	15600	RP_ReadyToLoad	
10	CE_LoadTransferBlocked	15792	RP_LoadPort	
11	CE_LoadMaterialArrived	15793	RP_LoadPort	
12	CE_LoadWaitingForHost	15794	RP_LoadPort	
13	CE_LoadAssociated	15795	RP_LoadPort	
14	CE_IdVerifyOK	15796	RP_LoadPort	
15	CE_OpenFoup	15797	RP_LoadPort	
16	CE_SlotMapVerifyOK	15798	RP_LoadPort	
17	CE_SubstAtSource	15799	RP_SubstAtSource	
18	CE_SubstOccupied	15800	RP_SubstReport	
19	CE_PrJobPooled	15801	RP_PrJobReport	
20	CE_CJobQueued	15808	RP_CJobReport	
21	CE_CJobSelected	15809	RP_CJobReport	
22	CE_ExecBegan	15810	RP_CJobReport	
23	CE_PrJobSetup	15811	RP_PrJobReport	
24	CE_CarInAccess	15812	RP_LoadPort	
25	CE_SubstAtWork	15813	RP_SubstReport	
26	CE_SubstInProcess	15814	RP_SubstReport	
27	CE_PrJobProcessing	15815	RP_PrJobReport	
28	CE_SubstProcessingComplete	15816	RP_SubstReport	
29	CE_ProcessingComplete	15817	RP_PrJobReport	
30	CE_SubstAtDestination	15824	RP_SubstReport	
31	CE_PrJobComplete	15825	RP_PrJobReport	
32	CE_CJobComplete	15826	RP_CJobReport	
33	CE_CJobDeleted	15827	RP_CJobReport	
34	CE_CloseFoup	15828	RP_LoadPort	
35	CE_CarComplete	15829	RP_LoadPort	
36	CE_ReadyToUnload	15856	RP_LoadPort	
37	CE_UnLoadTransferBlocked	15857	RP_LoadPort	
38	CE_MaterialRemoved	15858	RP_LoadPort	
39	CE_CarDeleted	15859	RP_LoadPort	
40	CE_LoadNotAssociated	15861	RP_LoadPort	
41	CE_TestList	254	RP_LoadPort RP_TestList	複数 RPID のｹｰｽのﾃｽﾄ用

４．４　レポートＩＤ

#	NAME	ID	ﾘﾝｸ VID	備考
1	RP_Communicating	10	SV_Clock	
2	RP_ControlState	100	SV_Clock SV_ControlState	
3	RP_PortAccessMode	1200	EC_PortAccessMode	
4	RP_Port1AccessMode	1201	EC_Port1AccessMode	
5	RP_Port2AccessMode	1202	EC_Port2AccessMode	
6	RP_ReadyToLoad	15600	SV_Clock SV_ReadyToLoad	
7	RP_LoadPort	15792	SV_Clock SV_LoadPortTransferStatus SV_LoadPortId SV_LoadCarId SV_LoadCarIdStatus SV_LoadSlotMapStatus SV_LoadPortAssociationStatus SV_LoadCarAccessStatus	
8	RP_SubstAtSource	15856	SV_Clock SV_LotId SV_SubstList SV_SubstID SV_ListEnd SV_CarLocation SV_SubstProcState SV_SubstState SV_CarUsage	
9	RP_SubstReport	15857	SV_Clock SV_LotId SV_SubstList SV_SubstID SV_ListEnd	
10	RP_PrJobReport	15870	SV_Clock SV_PrjID SV_PrjState SV_PPID	
11	RP_CJobReport	15871	SV_Clock SV_CJID SV_CjState	
12	RP_UnloadPort	15808	SV_Clock SV_LoadPortTransferStatus SV_UnloadPortId SV_UnloadCarId SV_UnloadCarIdStatus SV_UnloadSlotMapStatus SV_UnloadPortAssociationStatus SV_UnloadCarAccessStatus	

13	RP_TestList		SV_Clock SV_UnloadPortId SV_List1 SV_UnloadCarId SV_List2 SV_UnloadCarId SV_UnloadCarIdStatus SV_List2_End SV_List2 SV_UnloadCarId SV_UnloadCarIdStatus SV_List2_End SV_UnloadCarIdStatus SV_List1_End SV_UnloadSlotMapStatus SV_UnloadPortAssociationStatus SV_UnloadCarAccessStatus	Neting 構造 report ﾃｽﾄ用 他に意味はありません

４．５　アラームID

#	NAME	ID	ALCD	ALTX
1	AL_AlarmTempOver	1	2	"Chamber Temperature Over"
2	AL_AlarmPressure_1_Low	101	2	"Chamber Pressure-1 Over"
3	AL_AlarmPressure_2_Low	102	2	"Chamber Pressure-2 Over"
4	AL_AlarmPressure_3_Low	103	3	"Chamber Pressure-3 Over"

４．６　予約 CEID と変数

DSHEng5 が内部で自動処理するために必要な予約収集イベント ID と予約変数 ID の登録には、以下のものを使用します。

予約変数の設定は、DSH_ENG.class_Reserved_V クラスの set_reserved__CE、set_reserved__EC, set_reserved__SV メソッドを使用します。

装置開始時に登録します。

CE/変数	予約ｲﾝﾃﾞｸｽ	ID 名	備考
CE 収集ｲﾍﾞﾝﾄ	CEX_RSV_COMMUNICATING	CE_Communicating	
	CEX_RSV_SPOOL_END	CE_SpoolDeactivated	
EC 装置定数	ECX_RSV_MDLN	EC_Mdln	
	ECX_RSV_SOFTREV	EC_SoftRev	
	ECX_RSV_SPOOL_MAX	EC_MaxSpoolTransmit	
	ECX_RSV_SPOOL_OVERWRITE	EC_OverWriteSpool	
	ECX_RSV_INIT_COMMSTATE	EC_InitCommState	
SV 状態変数	SVX_RSV_CLOCK	SV_Clock	
	SVX_RSV_COMMUNICATING	SV_CommunicationState	
	SVX_RSV_SPOOL_STATE	SV_SpoolState	
	SVX_RSV_SPOOL_TOTAL	SV_SpoolCountTotal	
	SVX_RSV_SPOOL_ACTUAL	SV_SpoolCountActual	
	SVX_RSV_SPOOL_STIME	SV_SpoolStartTime	
	SVX_RSV_SPOOL_FTIME	SV_SpoolFullTime	

これらの設定は、formMain フォームで、ｴﾝｼﾞﾝ開始ボタンがクリックされたときの処理の中にあります。

setup_eq_start()の中で、たとえば、以下のように設定します。

```
DSH_ENG.class_Reserved_V.set_reserved_CEID(class_const.CEX_RSV_COMMUNICATING,eng_id.CE_Communicating);
DSH_ENG.class_Reserved_V.set_reserved_ECID(class_const.ECX_RSV_MDLN, eng_id.EC_Mdln);
```

５．ジョブ定義ファイル

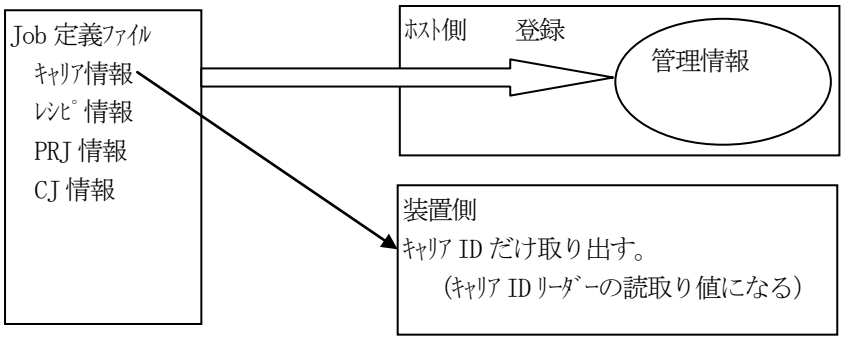
WP シミュレーションでホスト側の APP が使用するファイルであり、本デモプログラムのためだけのファイルです。

基本的には、ジョブ定義ファイルは、ホスト側で使用する情報です。

ただし、キャリア関連については、装置側で使用します。

ウェハ処理を行うキャリアとそれに対する処理情報（レシピ、プロセスジョブ、コントロールジョブ）をテキストファイル内にジョブ単位で定義します。そして、実際のウェハ処理シミュレーション時に、そのファイルを指定し、そこで定義されている順にジョブ単位の処理を行います。このファイルには、１個以上のジョブを登録することになります。

ジョブ定義ファイルで定義された情報は、全てホスト側のデモプログラムで管理情報として登録されます。また、装置側では、ポートへのロード時にキャリア ID だけを取り出して使用します。



（1）定義コマンド

#	ﾒｲﾝｺﾏﾝﾄﾞ	ｻﾌﾞｺﾏﾝﾄﾞ	意味
1	$START		１個のｼﾞｮﾌﾞの開始を示す。
2	$END		１個のｼﾞｮﾌﾞの終了を示す。
3	CARRIER		ｼﾞｮﾌﾞで処理されるｷｬﾘｱを定義します。
		PORT	ﾛｰﾄﾞﾎﾟｰﾄを指定します。
		ID	ｷｬﾘｱ ID を指定します。
		USAGE	usage を指定します。
		SLOT	ｽﾛｯﾄ情報を指定します。（ｽﾛｯﾄ ID, LOTID など）
4	RECIPE		ﾚｼﾋﾟ（またはﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ）を定義します。
		ID	ﾚｼﾋﾟ ID または PPID を指定します。
		BODY	RCPBODY(S15F13)または PPBODY （S7F3） を指定します。
		PARA	ﾚｼﾋﾟ(S15F13)または FPP(S7F23)のﾊﾟﾗﾒｰﾀを定義します。
		F_CCODE	FPP(S7F23)のｺﾏﾝﾄﾞｺｰﾄﾞを指定します。
5	PROCESS_JOOB		ﾌﾟﾛｾｽｼﾞｮﾌﾞを定義します。
		ID	ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID を指定します。
6	CONTROL_JOOB		ｺﾝﾄﾛｰﾙｼﾞｮﾌﾞを定義します。
		ID	ｺﾝﾄﾛｰﾙｼﾞｮﾌﾞ ID を指定します。

コマンドの文字は大文字でも小文字でも構いません。

（2）コマンド一般形式

```
$START
  〈ﾒｲﾝｺﾏﾝﾄﾞ〉{
      〈ｻﾌﾞｺﾏﾝﾄﾞ〉: 〈ﾊﾟﾗﾒｰﾀ1〉 [,〈ﾊﾟﾗﾒｰﾀ2〉[,.],....]
          (必要なだけ)
  }
  〈ﾒｲﾝｺﾏﾝﾄﾞ〉{
      〈ｻﾌﾞｺﾏﾝﾄﾞ〉: 〈ﾊﾟﾗﾒｰﾀ1〉 [,〈ﾊﾟﾗﾒｰﾀ2〉[,.],....]
          (必要なだけ)
  }
 .
 .
$END
```

①メインコマンドは、 コマンド文字列の後、”{“ 文字と ”}” 文字で囲みます。
②サブコマンドは、その後に、”:” 文字を付け、その後、１個以上のパラメータを続けます。
パラメータ間は “,” で区切ります。

（3）サブコマンド個別の形式

ﾒｲﾝｺﾏﾝﾄﾞ	ｻﾌﾞｺﾏﾝﾄﾞ	ｻﾌﾞｺﾏﾝﾄﾞﾊﾟﾗﾒｰﾀ
CARRIER	ID	〈id〉 ｷｬﾘｱ ID 文字列
	PORT	〈port no.〉 1 または 2
	USAGE	〈usage〉 “PRODUCT”、”MONITOR”, “DUMMY”などの文字列
	SLOT	〈slot#〉, 〈slotid〉, 〈ﾛｯﾄ id〉, 〈基板 id〉, 〈ﾛｹｰｼｮﾝ〉 〈slot#〉 はｽﾛｯﾄ①順位(0,1,..25) 〈slotid〉 は 2 桁の数値 〈ﾛｯﾄ id〉 は文字列 〈基板 ID〉 は文字列 〈ﾛｹｰｼｮﾝ〉 は文字列
RECIPE	ID	〈id〉 ﾚｼﾋﾟ またはﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ ID で、文字列
	BODY	〈body〉 文字列
	PARA	〈name〉, 〈value〉 〈name〉はﾊﾟﾗﾒｰﾀ名で文字列 〈value〉はﾊﾟﾗﾒｰﾀ値で、文字列
	f_ccode	〈name〉, 〈value1〉 [, 〈value2〉 [, 〈value3〉,]..] 〈name〉はｺﾏﾝﾄﾞｺｰﾄﾞ名で文字列 〈value〉はｺﾏﾝﾄﾞ値で、文字列
PROCESS_JOB	ID	〈id〉 ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID 文字列
CONTROL_JOB	ID	〈id〉 ｺﾝﾄﾛｰﾙｼﾞｮﾌﾞ ID 文字列

７． ４に１個のジョブ情報のサンプルを示します。

６．実現するシナリオ

本デモプログラムでは、シナリオ処理例として、次の２つのシナリオを実現します。

（1）通信確立シナリオ
（2）ウェハ処理シナリオ（正常ケース）

なお、レシピ（プロセスプログラム）ならびに、プロセスジョブ送信に使用する SECS-II メッセージを次のメッセージを使用します。

レシピ　　：　S15F13
プロセスジョブ　：　S16F11

ウェハ処理シナリオ実行プログラムの中では、制御モードはオンライン・リモート、ポートアクセスモードはオートモードで行います。（デモプログラム内では、特にチェックしていません。）

また、ウェハー処理シナリオについては単なるサンプルですので、流れ、処理ステップ定義についても大雑把に決められています。

エラーに対する細かな処理も行っていませんのでご了承ください。

６．１　通信確立シナリオ

	装置側デモプログラム	通信 MSG	ホスト側プログラム
1.	装置主導 Establish Communication Request CE_Communicating	S1F13 ---> <=== S1F14 S6F11 --->| <=== S6F12 <--- S2F31 S2F32 ===>	 Establish Communication Request 通信確立通知ｲﾍﾞﾝﾄ Date and Time Set Request
2.	ホスト主導	<--- S1F13 S1F14 ===> S6F11 --->| <=== S6F12 <--- S2F31 S2F32 ===>	Establish Communication Request 通信確立通知ｲﾍﾞﾝﾄ Date and Time Set Request

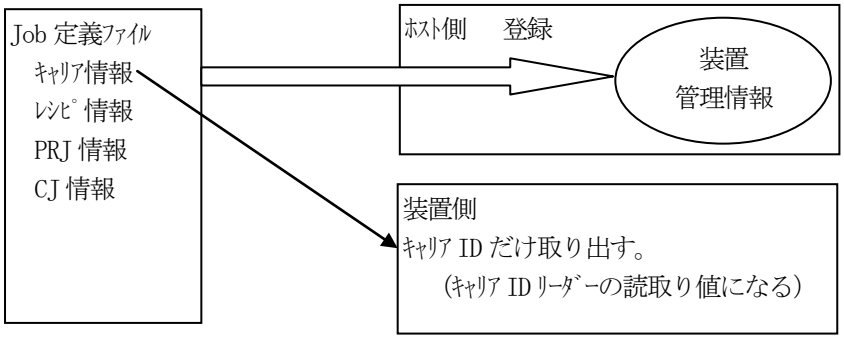
通信確立要求には、S1F13, S6F11 通信のやり取りは、GemEnable() API 関数が全てを処理してくれます。

６．２　ウェハ処理シナリオ（正常シナリオ）

６．２．１　ジョブ定義ファイル

ウェハ処理を行うキャリアとそれに対する処理情報（レシピ、プロセスジョブ、コントロールジョブ）をテキストファイル内にジョブ単位で定義します。そして、実際のウェハ処理シミュレーション時に、そのファイルを指定し、そこで定義されている順にジョブ単位の処理を行います。このファイルには、１個以上のジョブを登録することになります。

ジョブ定義ファイルで定義された情報は、全てホスト側のデモプログラムで管理情報として登録されます。また、装置側では、ポートへのロード時にキャリア ID だけを取り出して使用します。



（1）定義コマンド

#	ﾒｲﾝｺﾏﾝﾄﾞ	ｻﾌﾞｺﾏﾝﾄﾞ	意味
1	$START		１個のｼﾞｮﾌﾞの開始を示す。
2	$END		１個のｼﾞｮﾌﾞの終了を示す。
3	CARRIER		ｼﾞｮﾌﾞで処理されるｷｬﾘｱを定義します。
		PORT	ﾛｰﾄﾞﾎﾟｰﾄを指定します。
		ID	ｷｬﾘｱ ID を指定します。
		USAGE	usage を指定します。
		SLOT	ｽﾛｯﾄ情報を指定します。（ｽﾛｯﾄ ID, LOTID など）
4	RECIPE		ﾚｼﾋﾟ（またはﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ）を定義します。
		ID	ﾚｼﾋﾟ ID または PPID を指定します。
		BODY	RCPBODY（S15F13）または PPBODY （S7F3） を指定します。
		PARA	ﾚｼﾋﾟ（S15F13）または FPP（S7F23）のﾊﾟﾗﾒｰﾀを定義します。
		F_CCODE	FPP（S7F23）のｺﾏﾝﾄﾞｺｰﾄﾞを指定します。
5	PROCESS_JOB		ﾌﾟﾛｾｽｼﾞｮﾌﾞを定義します。
		ID	ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID を指定します。
6	CONTROL_JOB		ｺﾝﾄﾛｰﾙｼﾞｮﾌﾞを定義します。
		ID	ｺﾝﾄﾛｰﾙｼﾞｮﾌﾞ ID を指定します。

コマンドの文字は大文字でも小文字でも構いません。

（2）コマンド一般形式

```
$START
  〈ﾒｲﾝｺﾏﾝﾄﾞ〉{
      〈ｻﾌﾞｺﾏﾝﾄﾞ〉: 〈ﾊﾟﾗﾒｰﾀ1〉 [,〈ﾊﾟﾗﾒｰﾀ2〉[,.],....]
          (必要なだけ)
  }
  〈ﾒｲﾝｺﾏﾝﾄﾞ〉{
      〈ｻﾌﾞｺﾏﾝﾄﾞ〉: 〈ﾊﾟﾗﾒｰﾀ1〉 [,〈ﾊﾟﾗﾒｰﾀ2〉[,.],....]
          (必要なだけ)
  }
 .
 .
$END
```

①メインコマンドは、 コマンド文字列の後、"{" 文字と "}" 文字で囲みます。
②サブコマンドは、その後に、":" 文字を付け、その後、１個以上のパラメータを続けます。
パラメータ間は "," で区切ります。

（3）サブコマンド個別の形式

ﾒｲﾝｺﾏﾝﾄﾞ	ｻﾌﾞｺﾏﾝﾄﾞ	ｻﾌﾞｺﾏﾝﾄﾞﾊﾟﾗﾒｰﾀ
CARRIER	ID	〈id〉 ｷｬﾘｱ ID 文字列
	PORT	〈port no.〉 1 または 2
	USAGE	〈usage〉 "PRODUCT"、"MONITOR", "DUMMY"などの文字列
	SLOT	〈slot#〉, 〈slotid〉, 〈ﾛｯﾄ id〉, 〈基板 id〉, 〈ﾛｹｰｼｮﾝ〉 〈slot#〉 はｽﾛｯﾄ①順位(0,1,..25) 〈slotid〉 は 2 桁の数値 〈ﾛｯﾄ id〉 は文字列 〈基板 ID〉 は文字列 〈ﾛｹｰｼｮﾝ〉 は文字列
RECIPE	ID	〈id〉 ﾚｼﾋﾟ またはﾌﾟﾛｾｽﾌﾟﾛｸﾞﾗﾑ ID で、文字列
	BODY	〈body〉 文字列
	PARA	〈name〉, 〈value〉 〈name〉はﾊﾟﾗﾒｰﾀ名で文字列 〈value〉はﾊﾟﾗﾒｰﾀ値で、文字列
	f_ccode	〈name〉, 〈value1〉 [, 〈value2〉 [, 〈value3〉,]..] 〈name〉はｺﾏﾝﾄﾞｺｰﾄﾞ名で文字列 〈value〉はｺﾏﾝﾄﾞ値で、文字列
PROCESS_JOB	ID	〈id〉 ﾌﾟﾛｾｽｼﾞｮﾌﾞ ID 文字列
CONTROL_JOB	ID	〈id〉 ｺﾝﾄﾛｰﾙｼﾞｮﾌﾞ ID 文字列

次ページに１個のジョブ情報のサンプルを示します。

（4）ジョブファイルサンプル（JobSche.txt）

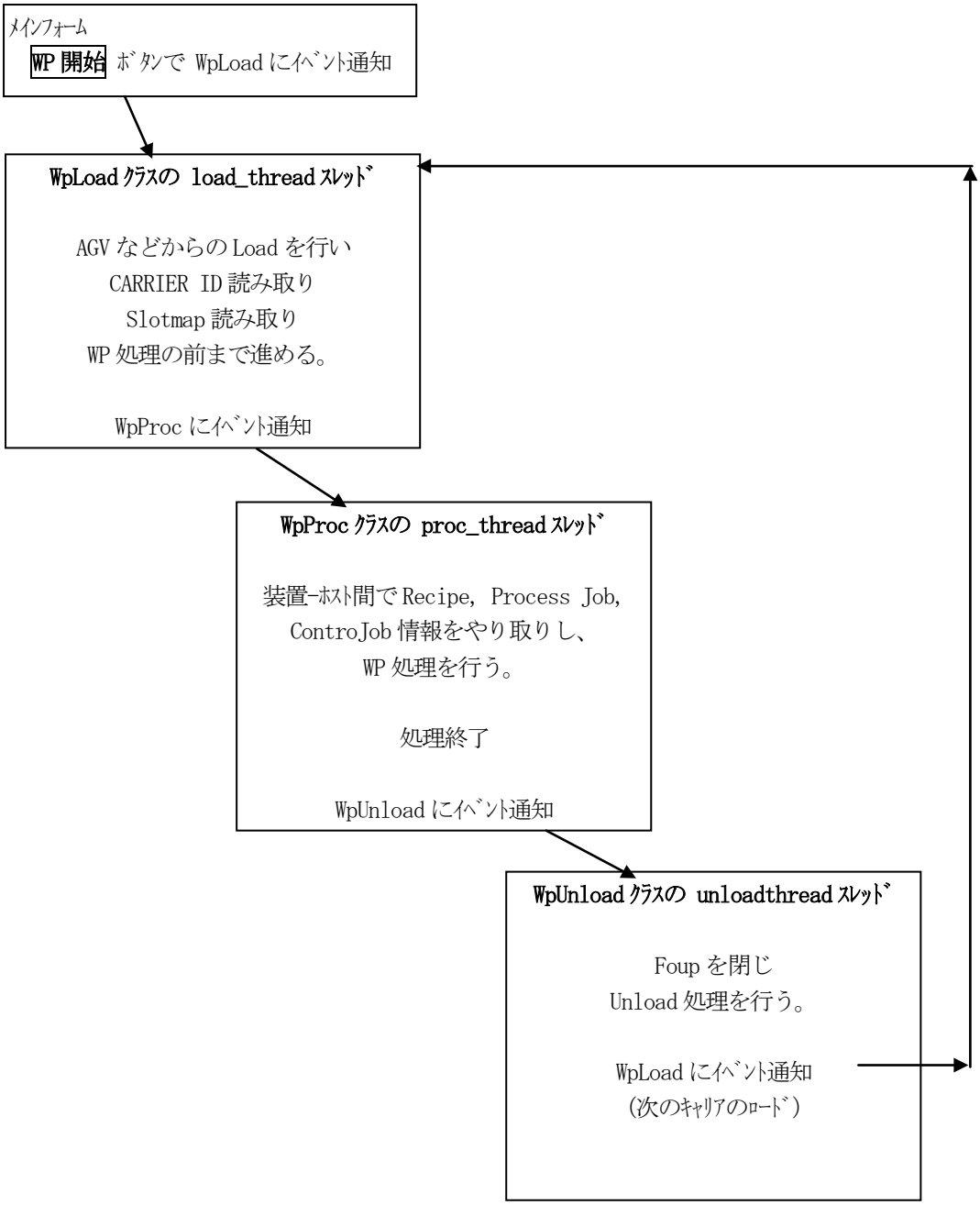
```
$START
//-------------------------------------------------------
carrier{
        port:   1
        id:     "CARID_01"
        usage:  "PRODUCT"
        slot:   0,10,"LOT_000","SUBST_01_00","LOC_A"     // 0
        slot:   1,11,"LOT_001","SUBST_01_01","LOC_A"     // 1
        slot:   2,12,"LOT_001","SUBST_01_02","LOC_A"     // 2
        slot:   3,13,"LOT_001","SUBST_01_03","LOC_A"     // 3
        slot:   4,14,"LOT_001","SUBST_01_04","LOC_A"     // 4
        slot:   5,15,"LOT_001","SUBST_01_05","LOC_A"     // 5
        slot:   6,16,"LOT_001","SUBST_01_06","LOC_A"     // 6
        slot:   7,17,"LOT_001","SUBST_01_07","LOC_A"     // 7
        slot:   8,18,"LOT_001","SUBST_01_08","LOC_A"     // 8
        slot:   9,19,"LOT_001","SUBST_01_09","LOC_A"     // 9
        slot:  10,20,"LOT_001","SUBST_01_10","LOC_A"     // 10
        slot:  11,21,"LOT_001","SUBST_01_11","LOC_A"     // 11
        slot:  12,22,"LOT_001","SUBST_01_12","LOC_A"     // 12
        slot:  13,23,"LOT_001","SUBST_01_13","LOC_A"     // 13
        slot:  14,24,"LOT_001","SUBST_01_14","LOC_A"     // 14
        slot:  15,25,"LOT_001","SUBST_01_15","LOC_A"     // 15
        slot:  16,26,"LOT_001","SUBST_01_16","LOC_A"     // 16
        slot:  17,27,"LOT_001","SUBST_01_17","LOC_A"     // 17
        slot:  18,28,"LOT_001","SUBST_01_18","LOC_A"     // 18
        slot:  19,29,"LOT_001","SUBST_01_19","LOC_A"     // 19
        slot:  20,30,"LOT_001","SUBST_01_20","LOC_A"     // 20
        slot:  21,31,"LOT_001","SUBST_01_21","LOC_A"     // 21
        slot:  22,32,"LOT_001","SUBST_01_22","LOC_A"     // 22
        slot:  23,33,"LOT_001","SUBST_01_23","LOC_A"     // 23
        slot:  24,34,"LOT_001","SUBST_01_24","LOC_A"     // 24
    }
//-------------------------------------------------------
recipe{
        id:     "RCP-0001"
        body:   "RCPBODY-001-001-001-A"
        para:   "PARA10","100.00"
        para:   "PARA11","101.01"
        f_ccode: "fppcmd1","1.00","1.01","1.02"
}
//-------------------------------------------------------
process_job{
        id:     "PJ-0001"
}
//-------------------------------------------------------
control_job{
        id:     "CJ-0001"
}
$END
```

６．２．２　シナリオ詳細

以下のシーケンスで行う。本シナリオは、WpLoad, WpProc, WpUnload クラス内のスレッドで実行することにします。
進行状況は WP オペレーションモニター画面に表示します。

シーケンスは、あくまでデモ用のものです。従って、通信の流れを簡単に実現するために細かいステップでの処理項目を省略しています。
シナリオは、ポートー１，２から交互に搬入し、処理を行い、搬入と同じポートから搬出します。

処理は、Load, Processing, Unload の３つのクラスのスレッドによって行われます。



６．２．２．１　Load 処理 -　WpLoad クラス

	装置側デモプログラム	通信 MSG	ホスト側プログラム	状態
	(Load ｲﾍﾞﾝﾄ受信)		(Load ｲﾍﾞﾝﾄ受信)	SOPE_LOAD_IDLE
1	搬送開始 - port-1			
2	CE_ReadyToLoad	S6F11 --->		
		<=== S6F12		SOPE_LOAD_START
3	CE_LoadTransferBlocked	S6F11 --->	状態更新	
	(ｲﾍﾞﾝﾄ通知)	<=== S6F12		SOPE_LOAD_END
	(ｷｬﾘｱ到着)			
4	CE_LoadMaterialArrived	S6F11 --->		
		<=== S6F12		SOPE_READ_CARID
	(ｷｬﾘｱ ID 読取り、登録)		ｷｬﾘｱ ID 取り込み	ｷｬﾘｱ ID 登録
5	CE_LoadWaitingForHost	S6F11 --->		
	(ｷｬﾘｱ ID 付き)	<=== S6F12		
6	ｷｬﾘｱｺﾝﾃﾝﾄ登録	<--- S3F17	ｷｬﾘｱｱｸｼｮﾝ CONTENT	
		S3F18 ===>	ProceedWithCarrier	SOPE_PORT_ASSOCIATED
7	CE_LoadAssociated	S6F11 --->	ID 確認 OK	
		<=== S6F12		SOPE_ID_VERIFY_END
8	(ｷｬﾘｱ情報格納)		移動	
9	CE_IdVerifyOK	S6F11 --->		
		<=== S6F12		
10	CE_SubstAtSource	S6F11 --->		
		<=== S6F12		
11	CE_SubstOccupied	S6F11 --->		
		<=== S6F12		SOPE_OPEN_FOUP
	(Foup Open)			
12	CE_OpenFoup			SOPE_READ_SLOTMAP
	(ｽﾛｯﾄﾏｯﾌﾟ 照合)	S6F11 --->		
		<=== S6F12		
13	CE_LoadWaitingForHost	S6F11 --->		SOPE_SLOTMAP_VERIFY_END
		<=== S6F12		
		<--- S3F17	ｷｬﾘｱｱｸｼｮﾝ	
		S3F18 ===>	ProceedWithCarrier	
14	CE_SlotMapVerifyOK	S6F11 --->		
		<=== S6F12		SOPE_LOAD_IDLE(終了)
				WpProc ｽﾚｯﾄﾞｲﾍﾞﾝﾄ通知

６．２．２．２　ウエハー処理 - WpProcess クラス

	装置側デモプログラム	通信 MSG	ホスト側プログラム	状態
1	(WpLoad からｲﾍﾞﾝﾄ)		(WpLoad からｲﾍﾞﾝﾄ)	SOPE_PROC_IDLE
2		<--- S15F13 S15F14 ===>	ﾚｼﾋﾟ送信 3 つのうち 1 個	SOPE_PPID_READY ﾚｼﾋﾟ情報登録
3		<--- S16F11 S16F12 ===>	ﾌﾟﾛｾｽｼﾞｮﾌﾞ送信 （または S16F15）	SOPE_PRJ_READY PRJ 情報登録
4	CE_PrJobPooled	S6F11 --->| <=== S6F12		
5		<--- S14F9 S14F10 ===>		SOPE_CJ_READY CJ 情報登録
6	CE_CJobQueued	S6F11 --->| <=== S6F12		
7	CE_CJobSelected) (処理開始)	S6F11 --->| <=== S6F12		
8	CE_ExecBegan (処理中)	S6F11 --->| <=== S6F12		SOPE_PROC_START
9	CE_PrJobSetup	S6F11 --->| <=== S6F12		
10	CE_CarInAccess	S6F11 --->| <=== S6F12		
11	CE_SubstAtWork	S6F11 --->| <=== S6F12		
12	CE_SubstInProcess	S6F11 --->| <=== S6F12		
13	CE_PrJobProcessing	S6F11 --->| <=== S6F12		
	(処理終了)			
14	CE_SubstProcessingComplete	S6F11 --->| <=== S6F12		SOPE_PROC_END

15	CE_ProcessingComplete	S6F11 ---> <=== S6F12		
16	CE_SubstAtDestination	S6F11 ---> <=== S6F12		
17	CE_PrJobComplete ﾌﾟﾛｾｽｼﾞｮﾌﾞ削除	S6F11 ---> <=== S6F12		SOPE_PRJ_COMPLETE
18	CE_CJobComplete	S6F11 ---> <=== S6F12		SOPE_CJ_COMPLETE
19	ｺﾝﾄﾛｰﾙｼﾞｮﾌﾞ削除	<--- S14F11 S14F12 ===>	ｺﾝﾄﾛｰﾙｼﾞｮﾌﾞ delete	
20	CE_CJobDeleted	S6F11 ---> <=== S6F12		SOPE_CJ_DELETE

６．２．２．３　Unload 処理 -　WpUnload クラス

	装置側デモプログラム	通信 MSG	ホスト側プログラム	状態
	(WpProc からｲﾍﾞﾝﾄ)		(WpProc からｲﾍﾞﾝﾄ)	SOPE_UNLOAD_IDLE
1	(Foup 閉)			SOPE_CLOSE_FOUP
2	CE_CloseFoup	S6F11 ---> <=== S6F12		
3	CE_CarComplete	S6F11 ---> <=== S6F12		
4	CE_ReadyToUnload	S6F11 ---> <=== S6F12		SOPE_UNLOAD_START
5	CE_LoadTransferBlocked	S6F11 ---> <=== S6F12		
6	CE_MaterialRemoved	S6F11 ---> <=== S6F12		
7	CE_LoadNotAssociated	S6F11 ---> <=== S6F12		(SOPE_UNLOAD_END)
8	CE_ReadyToLoad (次のｷｬﾘｱ受入れ可)	S6F11 ---> <=== S6F12		SOPE_UNLOAD_IDLE WpLoad にｲﾍﾞﾝﾄ (次のｷｬﾘｱ Load)

７．各種定義ファイルについて

DSHEng5 では、以下の定義ファイルを準備します。

（1）通信環境定義ファイル
（2）装置起動ファイル
（3）装置管理情報定義ファイル

７．１　ＤＳＨＤＲ２通信環境定義ファイル

DSHEng5 の HSMS 通信制御ドライバーDSHDR2.DLL に必要なファイルです。

装置側のチャンネル－１用のファイル例を示します。

```
#-------------------------------------------------------------
# DSHDR2 環境ファイルのサンプル - HSMS-SS & SECS
#-------------------------------------------------------------
START DSH
    MAX_MSG_SIZE = x100040      # max msg size = 1M + x40
    MAX_TRANSACTION = 512
    LOG_LINE = 100000
    LOG_FILE = c:¥DSHENG4¥log¥DSHDR2.LOG
    LOG_TYPE = LIST             # List structure
    LOG_MODE = 0                # save old log file
    MON_PORT = 9999             # Log Monitoring TCP Port
END

#-------------------------------------------------------------
START PORT
    PORT = 1                            # HSMS
    PROTOCOL = HSMS             # SS
    PORT_MODE = ACTIVE
    TCP_PORT = 5001
    IP = 192.168.1.21           # remote passive ip addr
    T3 = 45
    T5 = 10
    T6 = 5
    T7 = 10
    T8 = 5
    LINKTEST = 60
END

#-------------------------------------------------------------
START DEVICE
    DEVICE = 1
    DVID = x1234
    PORT = 1                    #     using port-1
END
```

７．２　装置起動ファイル

装置別に必要な起動条件ファイルです。

例を次に示します。

```
//----- equip.cnf - 装置起動ファイル ----------------------

LOG_PATH          = c:¥dsheng4¥log
LOG_MODE          = daily
LOG_LIFE           = 6

//LOG_MODE          = 0
//LOG_FILE          = equip.log
//LOG_SIZE          = 100000

INFO_FILE         = c:¥dsheng4¥cnf¥eqinfo.fil
BKUP_PATH         = c:¥dsheng4¥backup
INFO_BACKUP       = 1
PP_COUNT          = 100
FPP_COUNT         = 64
RCP_COUNT         = 200
CAR_COUNT         = 80
CAR_CAPACITY      = 80
SUBST_COUNT       = 250
CJ_COUNT          = 32
PRJ_COUNT         = 30
TRACE_COUNT       = 28
SPOOL_PATH        = c:¥dsheng4¥spool
COMM_PORT         = 1
COMM_DEVICE       = 1
S1F13_SEND        = 2
```

７．３　装置管理情報定義ファイル

装置の管理情報（装置変数、収集イベント、レポート、アラームなど）を定義するファイルです。

Eng3Conf.exe　プログラムでコンパイルした上で DSHEng5 で使用します。

次に、例を示します。（一部抜粋したものです。）

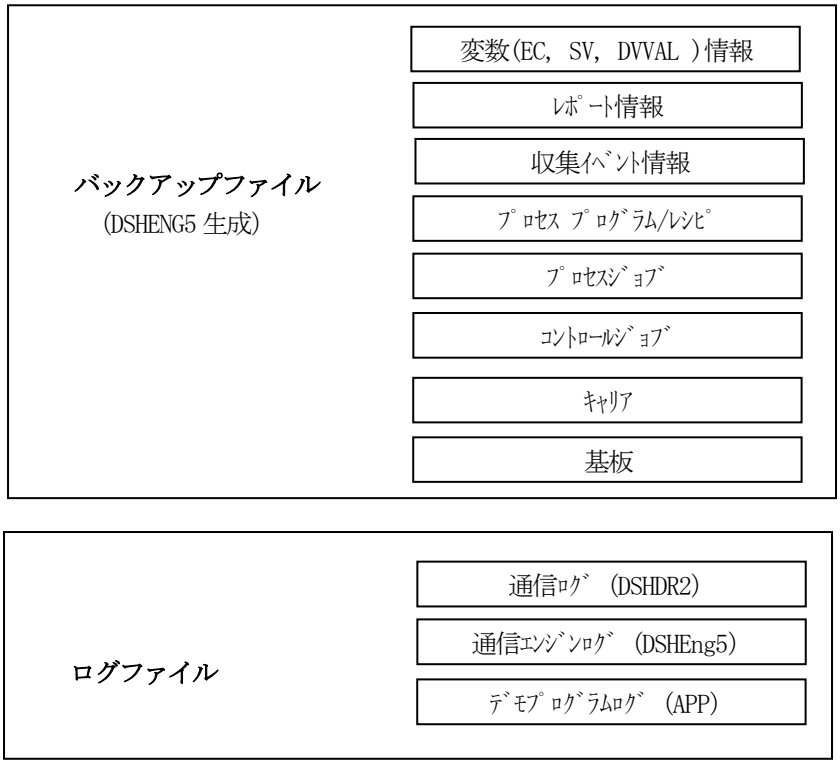
```
//----- C:¥DSHENG4¥bin¥EQINFO.TXT ----------------------------------


//----- EC - Equipment Constant -------------------------------------------
def_ec  EC_Mdln{
        ecid:     1                             // =0x00000001
        format:   A[6..6]
        nominal:  "DSH100"
        }

def_ec  EC_SoftRev{
        ecid:     2                             // =0x00000002
        format:   A[6..6]
        nominal:  "REV001"
        }

def_ec  EC_InitCommState{
        ecid:     4                             // =0x00000004
        format:   U2[1]
        min:      "0"
        max:      "1"
        nominal:  "0"
        }

def_ec  EC_InitControlState{
        ecid:     5                             // =0x00000005
        format:   U2[1]
        min:      "0"
        max:      "1"
        nominal:  "0"
        }

def_ec  EC_InitOfflineSubState{
        ecid:     6                             // =0x00000006
        format:   U2[1]
        min:      "1"
        max:      "3"
        nominal:  "1"
        }

(以下省略)
```

7．4　ジョブファイルサンプル (¥DSHEng5¥cnf¥JobSche.txt)

```
$START
//-------------------------------------------------------------
carrier{
        port:   1
        id:     "CARID_01"
        usage:  "PRODUCT"
        slot:   0,10,"LOT_000","SUBST_01_00","LOC_A"     // 0
        slot:   1,11,"LOT_001","SUBST_01_01","LOC_A"     // 1
        slot:   2,12,"LOT_001","SUBST_01_02","LOC_A"     // 2
        slot:   3,13,"LOT_001","SUBST_01_03","LOC_A"     // 3
        slot:   4,14,"LOT_001","SUBST_01_04","LOC_A"     // 4
        slot:   5,15,"LOT_001","SUBST_01_05","LOC_A"     // 5
        slot:   6,16,"LOT_001","SUBST_01_06","LOC_A"     // 6
        slot:   7,17,"LOT_001","SUBST_01_07","LOC_A"     // 7
        slot:   8,18,"LOT_001","SUBST_01_08","LOC_A"     // 8
        slot:   9,19,"LOT_001","SUBST_01_09","LOC_A"     // 9
        slot:  10,20,"LOT_001","SUBST_01_10","LOC_A"     // 10
        slot:  11,21,"LOT_001","SUBST_01_11","LOC_A"     // 11
        slot:  12,22,"LOT_001","SUBST_01_12","LOC_A"     // 12
        slot:  13,23,"LOT_001","SUBST_01_13","LOC_A"     // 13
        slot:  14,24,"LOT_001","SUBST_01_14","LOC_A"     // 14
        slot:  15,25,"LOT_001","SUBST_01_15","LOC_A"     // 15
        slot:  16,26,"LOT_001","SUBST_01_16","LOC_A"     // 16
        slot:  17,27,"LOT_001","SUBST_01_17","LOC_A"     // 17
        slot:  18,28,"LOT_001","SUBST_01_18","LOC_A"     // 18
        slot:  19,29,"LOT_001","SUBST_01_19","LOC_A"     // 19
        slot:  20,30,"LOT_001","SUBST_01_20","LOC_A"     // 20
        slot:  21,31,"LOT_001","SUBST_01_21","LOC_A"     // 21
        slot:  22,32,"LOT_001","SUBST_01_22","LOC_A"     // 22
        slot:  23,33,"LOT_001","SUBST_01_23","LOC_A"     // 23
        slot:  24,34,"LOT_001","SUBST_01_24","LOC_A"     // 24
    }
//-------------------------------------------------------------
recipe{
        id:     "RCP-0001"
        body:   "RCPBODY-001-001-001-A"
        para:   "PARA10","100.00"
        para:   "PARA11","101.01"
        f_ccode: "fppcmd1","1.00","1.01","1.02"
}
//-------------------------------------------------------------
process_job{
        id:     "PJ-0001"
}
//-------------------------------------------------------------
control_job{
        id:     "CJ-0001"
}
$END
```

８．バックアップファイルとログファイル

管理情報バックアップファイル、各種ログファイルが生成されます。



（1）管理情報バックアップファイル
DSHEng5 が管理する情報のバックアップファイルです。
変数 (EC, SV, DVVAL)、プロセスプログラム(PP)/レシピ、プロセスジョブ(PRJ)、コントロールジョブ(CJ)
キャリア、基板情報が対象です。装置起動ファイル内に指定されるディレクトリにバックアップされます。
バイナリ形式のファイルです。

（2）通信ログファイル
DSHDR2 通信ドライバーが相手装置と行った通信の SECS-II レベルのメッセージがリスト構造で記録されます。日付別にファイルを作成することができます。

（3）DSHEng5 通信エンジンログファイル
装置固有でない、DSHEng5 のログ情報が記録されます。内部プログラムの開始、終了情報も含まれます。
日付別にファイルを作成指定することができます。

（4）デモプログラムログファイル
デモプログラムがアプリケーションレベルのログを記録します。