

SECS／HSMS 通 信 シ ミ ュ レ ー タ ( 暫 定 版 )  
( tdISSim )

(Trust Design Simple SECS/HSMS Simulator (Preliminary version))

取 扱 説 明 書

Version 12.121 : 2012.12.12  
Version 14.040 : 2014.04.25  
Version 15.041 : 2015.04.25  
Version 15.050 : 2015.05.08  
Version 15.080 : 2015.08.01  
Version 15.110 : 2015.11.13  
Version 15.111 : 2015.11.27  
Version 16.011 : 2016.01.15  
Version 16.012 : 2016.02.25  
Version 16.040 : 2016.04.05  
Version 16.060 : 2016.06.10  
Version 17.030 : 2017.03.01

合 同 会 社 ト ラ ス ト デ ザ イ ン

長野県 茅野市 中大塩 3 - 4 3

Tel:0266-75-2279 E-mail:info@trust-design.co.jp  
Fax:0266-75-2279 URL:http://www.trust-design.co.jp

## 目

## 次

1. はじめに

2. 操作説明

3. スクリプト言語仕様

## 1. はじめに

本プログラムは、SEMIスタンダード1メッセージトランスファ（SEMI E4/SECS-1）、高速 SECS メッセージサービス シングルセッションモード（SEMI E37.1/HSMS-SS）及び高速 SECS メッセージサービス ジェネラルセッション（SEMI E37.2/HSMS-GS）に準拠した通信をシミュレートし、通信試験等の役に供するものです。

本プログラムは以下の特徴を持ちます。

- ・ ホスト側、装置側、（パッシブ側、アクティブ側）の両方をサポートします。
- ・ メッセージ送信機能、受信機能、受信メッセージに対する返信機能を有します。
- ・ 返信は、受信メッセージに合致した適切な返信メッセージを自動選択して自動返信することも、返信対象となる複数のメッセージから、ユーザが選択して返信することもできます。
- ・ 独自の（超）簡易言語（超は、言語にではなく、簡易にかかります。）で記述したスクリプト・ファイルを使用して、連続した SECS メッセージ通信を自動運転することができます。
- ・ SECS 通信トレースを指定のファイルに指定の個数、容量で保存することができます。
- ・ SML 形式でのメッセージ定義ファイルを使用し、指定メッセージの送受信を行います。
- ・ 送信メッセージ項目のデータ値を、送信状況に合わせて変更設定することが可能です。  
また、大量のデータ値を格納したファイルを用意し、送信メッセージ項目のデータ値として、そのファイル名称を指定することにより、データ値をファイル内容より取得することも可能です。
- ・ 可変長項目を使用することができます。
- ・ 複数レベル、複数個数の不定個数リストを定義することが可能です。  
実行時にリスト個数を確定させ、そのリスト内の各項目値を設定することが可能です。
- ・ 本プログラムは、独自の簡易言語を使用して、連続した SECS メッセージ通信を自動運転することができます。この簡易言語は、以下の機構を持ちます。
  - ・ 複数のシナリオ（実行シーケンス）定義
  - ・ 変数（文字列、整数、実数）
  - ・ 変数の演算（四則演算等）
  - ・ IF 文による条件判定
  - ・ ブロック IF 文、GOTO 文による処理分岐
  - ・ WHILE 文による繰り返し処理
  - ・ CALL 文による関数呼び出し
  - ・ EXEC 文による外部プログラムの起動
  - ・ SECS メッセージ送受信
  - ・ 送信メッセージを構成するデータ項目値を送信時に変更設定
  - ・ 受信メッセージからデータ項目値を抜き出して変数に設定
  - ・ その他 ...
- ・ この簡易言語は、通常の SECS 通信をシミュレートする動作としては、一応（十分）使用可能ではありますが、スクリプト記述手法、動作速度等々の面では、残念ながらイマイチです。  
現在 C言語 をベースにしたスクリプト言語による実装を計画中です。ご期待ください。
- ・ 本プログラムは、「暫定版」のため、エラー処理、処理速度、ヘルプ機能、日本語処理、簡易言語仕様、操作説明等に関して、手を抜いている部分がありますが、・・・ご容赦ください。
- ・ また、同様の理由により、正しい設定、正しい使用方法でお使いいただく、善意のユーザ様を、ご利用いただく対象として想定しております。一部意地悪な使用方法等には対応していない部分もありますが、ご容赦ください。

- ・ SECS/HSMS による通信システムの開発には、弊社 SECS/HSMS 通信パッケージ (Trust Design Simple SECS Communication Library) (使用ライセンス無料) をご利用ください。詳しくは、弊社ホーム・ページをご覧ください。
- ・ SECS-1、HSMS-SS 相互の通信プロトコルを変換するアプリケーションとして、SECS/HSMS プロトコル変換プログラム (Trust Design Simple SECS/HSMS Protocol Converter) (使用ライセンス無料) をご利用ください。詳しくは、弊社ホーム・ページをご覧ください。
- ・ PLC にて制御する装置を、SECS 通信 I/F で上位システムに接続するためのアプリケーションとして、SECS/PLC 通信接続プログラム (Trust Design Simple SECS/PLC Communication Connection) (使用ライセンス無料) を公開しております。詳しくは、弊社ホーム・ページをご覧ください。
- ・ SECS(HSMS) による通信のモニターには、弊社 ネットワーク通信モニター (Trust Design Simple Network Communication Monitor) (使用ライセンス無料) をご利用ください。詳しくは、弊社ホーム・ページをご覧ください。

## 2. 操作説明

### (0) 準備

本プログラムを起動する前に、必ず以下の2つのファイルを正しく設定し、用意する必要があります。

- ・ SECS/HSMS 通信パラメータ設定ファイル ..... (.ini ファイル)
- ・ SECS/HSMS 通信メッセージ構造定義ファイル ..... (.sml or .csv ファイル)

また、通信の自動運転を行う場合は、通信シーケンスを記述した、以下のファイルを正しい設定で、予め作成しておく必要があります。

- ・ シナリオ実行シーケンス定義スクリプト・ファイル ..... (.ssl ファイル)

.ini、.sml、.csv の各ファイルの設定方法は、弊社「SECS/HSMS 通信パッケージ (Trust Design Simple SECS Communication Library) (TDS)」に付属する「プログラマーズ・マニュアル (TDS.pdf)」の該当する部分 (.ini: 2.1(1)、.sml: A (1)、.csv: A (2)) をご参照ください。

SECS/HSMS 通信パッケージ (Free) は、弊社ホーム・ページ (<http://www.trust-design.co.jp/>) より、ダウンロードできます。

.ssl ファイルの記述方法は、第3章をご参照ください。

なお、本パッケージには .ini、.sml、.ssl のサンプル・ファイルが付属しております。まずは、このサンプル・ファイルを基にして、必要に応じて改訂を行い、ご使用することをお勧めします。

(注 1) SECS/HSMS 通信パラメータ設定ファイル (.ini) の設定は、特に以下の項目にご注意ください。  
(詳細は、上述の TDS.pdf 2.1 (1) をご参照ください。)

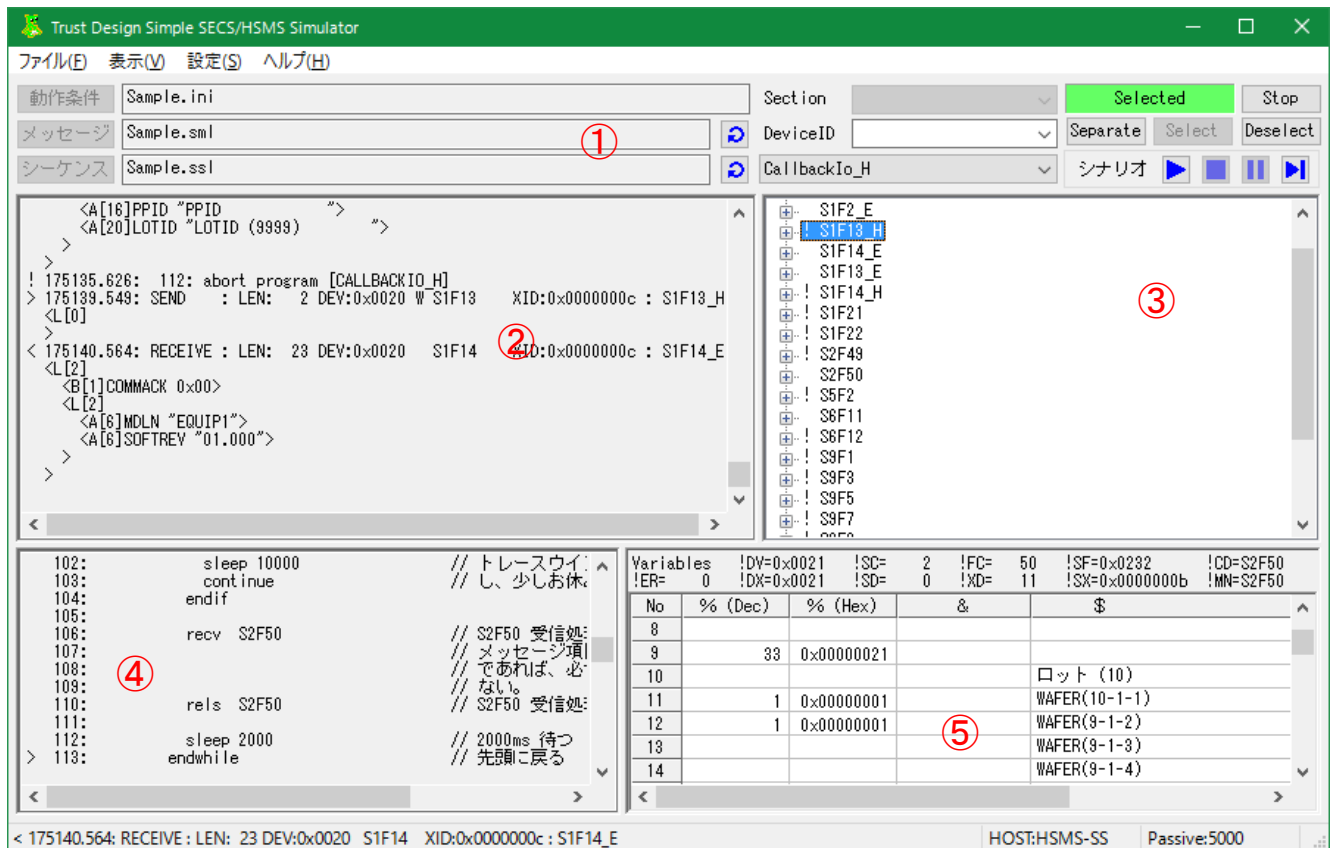
- ・ SECSMODE ..... SECS 通信パラメータ
  - bit#0,1 .... 通信形式 (SECS-1 or HSMS-SS)
  - 4 ..... 装置 or ホスト
  - 5 ..... SECS 時 Master or Slave
  - 6 ..... HSMS 時 Active or Passive
- ・ DEVMODE ..... デバイス制御モード
  - bit#0 ..... デバイス I D チェックの有無
  - 1 ..... 受信待ち状態でない2次メッセージに対する処理
  - 8-12 ... S9Fx、Reject 自動送信の有無
- ・ DEVID ..... 接続デバイス I D
- ・ XDEV ..... 接続デバイス I D の最大個数
- ・ XMSGSSIZE ..... 最大 SECS メッセージ・バイト長  
ある程度余裕を持った数値を指定してください。
- ・ SDEVICE ..... SECS-1 接続時に使用する COM ポート名称 (“COM1” 等)
- ・ HOST ..... HSMS-SS 接続時で Active 接続する場合の接続先ホスト名称 もしくは IP アドレス。
- ・ PORT ..... HSMS-SS 接続に使用する TCP/IP ポート番号
- ・ LINKINT ..... HSMS-SS 接続時のリンクテスト実行間隔  
リンクテストを実行する場合は、実行間隔の秒数を指定してください。
- ・ TRCDIR ..... 通信トレース・ファイル格納フォルダ  
相対パスを指定する場合、本 .ini ファイルの存在するフォルダが基点になります。通信トレース・ファイルのファイル名称に関しては TDS.pdf 2.1 (3) をご参照ください。
- ・ TRCTTYPE ..... 通信トレースへの通信メッセージ出力形式 (TDS.pdf 2.1(2) (c) 注記参照)
- ・ TRCTOUT ..... 通信トレース出力モード
- ・ TRCTLEVEL ..... 通信トレース出力レベル  
SECS-1 接続時、通信制御コードも出力する場合は、6 以上の値を指定してください。HSMS-SS 接続時、リンクテストに関するトレースも出力する場合は、9 or 10 以上の値を指定してください。
- ・ MDMSSG ..... 通信トレース出力に使用するメッセージ定義ファイル (.sml or .csv) を指定してください。
- ・ MDMXITEM ..... データ項目総数の最大個数
- ・ MDMXMSSG ..... 定義するメッセージの最大個数
- ・ MDMXMITEM ..... データ項目総数の最大個数+メッセージ展開時の最大項目数
- ・ MDMXPOOL ..... メッセージ定義 設定データ格納領域サイズ  
これらの項目は、ある程度余裕を持った数値を指定してください。

(1) 起動

インストールしたフォルダにある `tdlSSim.exe`（もしくは `tdlSSim.exe` へのショートカット）をダブル・クリック等により起動します。

[注意] -----+  
| 本プログラムは、ライセンス管理用として、UDP/IP の以下のポートを使用します。また、UDP/Multicast |  
| アドレスとして、以下のクラスDアドレスを使用します。ご使用になるコンピュータのファイアウォール等 |  
| により、これらをブロックしないよう設定してください。 |  
| ・ 36275/udp |  
| ・ 239.254.200.75 |  
| なお、インターネット接続環境を含め、ネットワーク接続ができない状態、NIC が存在しない状態でも使用 |  
| は可能であり、使用に関する機能上の制限等は、同環境がある場合に比して、一切ありません。 |  
+-----

## (2) 画面操作説明



- ① : 全体操作パネル ..... 動作条件の設定。シミュレーションの起動、停止。通信シナリオの実行等の操作を行います。
- ② : 通信トレース表示 ..... 送受信した SECS 通信メッセージのトレース表示を行います。
- ③ : 通信メッセージ編集 ... 指定した「メッセージ定義ファイル」により決定する、送受信対象の SECS 通信メッセージを編集するとともに、送信メッセージを構成するメッセージ項目の編集を行います。
- ④ : 自動実行スクリプト ... 指定した「シーケンス定義ファイル」の内容を表示し、選択したシナリオに対応するスクリプトを確認するとともに、シナリオを自動実行している場合は、実行ステートメントを示します。
- ⑤ : スクリプト変数 ..... 自動実行スクリプトが使用する「スクリプト変数」の内容（値）を表示します。また、値の指定（変更）を行います。

＜ 参考 ＞ 一般的な操作手順

1. [動作条件]ボタンにより、SECS/HSMS 通信パラメータを記述した、.ini ファイルを選択します。
2. シナリオの自動実行を行う場合は、[シーケンス]ボタンにより、シナリオ実行シーケンス定義スクリプトを記述した、.ssl ファイルを選択します。
3. [表示]メニューで、通信トレースウインド (②) への表示形態を指定します。
4. [設定]メニューで、「装置」 or 「ホスト」の別を決定します。
5. [Start]ボタンで、シミュレーション処理を開始します。
6. HSMS 接続で「アクティブ」の場合において、[設定]メニューで「自動接続」を指定していない場合は、HSMS-SS の場合は、[Connect]、[Select] ボタンを順に押して、パッシブ側との通信確立を行います。HSMS-GS の場合は、[Connect] でパッシブ側に接続した後、Select 対象の DeviceID を選択し [Select] ボタンにより、対象 DeviceID に関する通信確立を行います。
7. あとは、好きな操作をどうぞ。

(注 1) [Start] ボタン押下時、以下のエラーが発生する場合、指定の .ini の該当箇所の指定値を変更してください。

-941 : データ・アイテム定義テーブルの領域不足 .....	MDMXITEM
-942 : メッセージ定義テーブルの領域不足 .....	MDMXMSG
-943 : メッセージ毎のアイテムを格納するテーブルの領域不足 .....	MDMXITEM
-944 : メッセージ毎のアイテムに設定、チェックするデータ格納領域の不足 .....	MDMXPOOL

その他、エラー番号は「TDS」と共通です。前述の「弊社「SECS/HSMS 通信パッケージ

(Trust Design Simple SECS Communication Library) (TDS)」に付属する「プログラマーズ・マニュアル (TDS.pdf)」の該当箇所をご参照ください。

(注 2) 本プログラムは自動応答モードで動作する場合、基本的には受信する可能性のある全てのメッセージをメッセージ定義ファイル (.sml、.csv) に定義する必要がありますが、以下の方法により、指定の SF-Code の受信に関して、そのメッセージ構造に関わらず、所定の 2 次メッセージ返信を行う事ができます。

- ・ メッセージ定義ファイルにおいて、デフォルトの 2 次メッセージを定義する。詳細は、(注 1) に記載の TDS.pdf の A. (1) (b) をご参照ください。
- ・ シナリオ実行スクリプトを使用する。詳細は第 3 章 (0) の注記をご参照ください。



(a) メニュー

(a-1) [ファイル]

- ・ アプリケーションの終了 ..... tdlSSim を終了します。

(a-2) [表示]

- ・ 通信トレース表示クリア ..... 通信トレースウインド (②) をクリアします。
- ・ 通信トレース最終行表示 ..... 通信トレースウインドのスクロール・バーを、最終行が表示されている状態にします。  
(高速で自動スクロールしている状態等で、通常のスクロール・バーの操作では、なかなか最終行が表示された状態にならない場合に使用します。)
- ・ List 表示 ..... 通信トレースウインドに、送受信 SECS メッセージを、指定の List 形式で表示します。
- ・ Hexa 表示 ..... 通信トレースウインドに、送受信 SECS メッセージを、16 進数形式で表示します。
- ・ スクリプトトレース表示 ..... シナリオの自動実行を行っている場合、現在実行したスクリプト・ステートメントを表示します。
- ・ 位置・サイズ保存 ..... プログラム終了時に、終了時点のウインド位置、サイズ、指定条件等の情報を保存します。次回起動時に、その情報を基に、画面状態を復元します。  
(状態は tdlSSim.exe と同じフォルダの tdlSSimWin.ini に保存します。)
- ・ ステータス・バー ..... ステータス・バーを表示します。

(a-3) [設定]

- ・ 装置 ..... 「装置」側として、シミュレーションを行います。  
[動作条件] において [セクション] を指定しない場合、指定の SECS/HSMS 通信パラメータを記述した .ini ファイルの [EQUIP] セクションを使用します。
- ・ ホスト ..... 「ホスト」側として、シミュレーションを行います。  
[動作条件] において [セクション] を指定しない場合、指定の SECS/HSMS 通信パラメータを記述した .ini ファイルの [HOST] セクションを使用します。
- ・ 自動接続 ..... 指定の SECS/HSMS 通信パラメータを記述した .ini ファイルでの設定が、HSMS アクティブ接続の場合、シミュレーションの開始後、自動的に、パッシブ側への、接続、Select 処理を行います。

(注 1) 自動接続を行うのは、[装置]動作指定の場合ではなく、「アクティブ接続」設定の場合です。  
通常 [装置] 動作の場合は「アクティブ接続」設定である場合が多いと思われますが、あくまで、.ini での設定が「アクティブ接続」の場合です。

- ・ 自動応答 ..... チェックをつけると、相手側から返信が必要な 1 次メッセージを受信した場合、指定のメッセージ定義ファイル (.sml) での設定に従って、最適な 2 次メッセージを自動的に選択し送信します。

(注 2) .sml ファイルに、返信対象の SF-Code を持つ 2 次メッセージが複数設定されている場合、受信 1 次メッセージを解析して決定した 1 次メッセージ定義の後方にある最初の 2 次メッセージを選択します。受信 1 次メッセージ定義の後方に対象となる 2 次メッセージの定義がない場合は、定義ファイルの先頭に戻って検索します。それでも見つからない場合は、S9F3、S9F5 等を応答します。

チェックを外すと、相手側から返信が必要な 1 次メッセージを受信した場合、返信メッセージを選択するダイアログを表示し、送信するメッセージを選択します。

- ・ メッセージ方向評価 ..... メッセージ定義ファイル (.sml) に指定された、「装置」「ホスト」毎のメッセージの有効性を判断し、それぞれの有効送信対象メッセージのみを送信対象として選択可能にします。  
チェックを外した状態では、現在の動作状態が「装置」「ホスト」のどちらでも、全メッセージを送信対象として選択が可能になります。

- ・ 通信試験モード ..... チェックをつけると、簡易的な通信試験を行うモードになります。  
このモードでは、以下が可能です。
  - ・ 送信データの SECS ヘッダ部各項目値の指定
  - ・ 送信データの SECS メッセージ部のバイト長
  - ・ HSMS 接続の場合、送信 TCP/IP パケットのバイト長を指定し分割送信

本モードの場合、送信メッセージを選択すると、以下のダイアログを表示し、上記各パラメータの変更指定が可能となります。

送信メッセージ	
Device ID	0x0020
S-Code	0x82
F-Code	0x31
P-Type Code	0
S-Type Code	0
Source ID	0x0000
Transaction ID	0x0002
Message length	87
Sending size	0
<input type="button" value="送信"/> <input type="button" value="キャンセル"/>	

今度送信する SECS メッセージのヘッダ部各項目値がデフォルト表示されるので、必要に応じて値を修正します。  
16 進数を入力する場合は、接頭句として "0x" を付与してください。

#### Message length

送信メッセージの SECS メッセージ部のバイト長を指定します。

#### Sending size

HSMS 接続時は、送信パケットを分割送信する場合は、1 以上の数値を指定します。0 の場合は、SECS メッセージ部を一括送信します。  
SECS-1 接続時は、マルチ・ブロック送信時の送信バイト長 (1~254) を指定します。0 の場合は、254 バイトとなります。

(注 3) 本モードは、スクリプト実行を行っておらず、自動応答が無効の場合のみ指定可能となります。

(注 4) P-Type、S-Type Code の指定は、通信形態が HSMS の場合のみ指定が可能です。  
SECS-I 接続時のブロック番号の指定はできません。


(注 5) SECS ヘッダ部の各項目値を変更して送信した場合の、相手側動作は予測できません。  
tdlSSim では、「それなり」に処理します。(必ずしも正常な動作になる訳ではありません。)

(b) 全体操作パネル

- ・ [動作条件] ..... SECS/HSMS 通信パラメータを記述した .ini ファイルを選択します。直接入力することも可能です。


- ・ [メッセージ] ..... 本プログラムが、ウインド上での表示、操作に使用する SECS メッセージ定義ファイル (.sml) を選択します。直接入力することも可能です。相対パスを指定する場合、.ini ファイルが存在するフォルダが基点になります。本項目を指定しない場合、メッセージ定義ファイルとして、SECS/HSMS 通信パラメータ設定ファイル (.ini) の MDMSSG に指定されたファイルを使用します。

(注 1) 本プログラムが指定 (.ini の TRCTOUT、TRCTTYPE 等で指定) により出力する通信トレースファイルに、解析したメッセージ構造の、メッセージ名称、項目名称を表示する場合に使用するのは、あくまで .ini の MDMSSG で指定したファイルです。

- ・  ..... 指定の SECS メッセージ定義ファイル (.sml) を再度読み込みます。シミュレーション処理の開始後、.sml ファイルを変更した場合に使用します。

(注 2) この処理で再読み込みを行うのは、ウインド上での表示、操作に関わるメッセージ定義です。通信トレース・ファイルに出力するメッセージ構造、メッセージ名称、項目名称等は .ini の MDMSSG に指定したメッセージ定義ファイルです。このファイルをシミュレーション実行中に変更した場合、変更後しばらくした後、通信トレース出力に反映します。





- ・ [シーケンス] ..... SECS 通信シナリオの自動実行に使用する、自動実行スクリプトを記述したファイル (.ssl) を選択します。直接入力することも可能です。相対パスを指定する場合、.ini ファイルが存在するフォルダが基点になります。通信シナリオの自動実行を行わない場合は、指定しなくてもかまいません。

- ・  ..... 指定の自動実行スクリプト・ファイル (.ssl) を再度読み込みます。シミュレーション処理の開始後、.ssl ファイルを変更した場合に使用します。

- ・ Section ..... 指定の .ini ファイル内の、使用するセクション名称を選択します。本項目を指定しない場合、「装置」として動作する場合は [EQUIP]、「ホスト」として動作する場合は [HOST] をセクション名として使用します。

- ・ DeviceID ..... 送信する 1 次メッセージに付与する DeviceID (SessionID) を選択、もしくは指定します。(16 進数で指定する場合は、説得句として "0x" を付与します。ドロップダウンリストには、指定の .ini ファイルの DEVID に定義された DeviceID の一覧が最大 64 個表示されます。表示時点で既に Select 状態にある DeviceID (SessionID) には "\*" マークが付与されています。本項目が空欄である場合は、.ini ファイルの DEVID に定義された、最初の値を使用します。HSMS-GS での動作においては、Select、Deselect、Separate 要求の送信時にも、本項目値を使用します。

- ・ シナリオ選択 ..... 自動実行するシナリオを選択します。指定の自動実行スクリプト・ファイル (.ssl) の program 文に指定したプログラム名称の一覧から選択します。

- ・ [Start] / [Stop] ..... Start : シミュレーション処理を開始します。  
    Stop : シミュレーション処理を終了します。
- ・ [Connect] / [Separate] ..... Connect : HSMS 接続時で自動接続でない場合、Passive 側に接続します。  
    Separate : 相手側に “Separate” を要求します。  
    HSMS-SS 動作時は、そのまま切断することになります。  
    HSMS-GS 動作時は、.ini ファイルの設定により動作が異なります。詳細は、2. (0) 記載の TDS.pdf 2.1 (1) (c) の SECSMODE の説明を参照してください。
- ・ [Select] ..... Select : HSMS 接続時で自動接続でない場合、Passive 側に Select を要求します。  
    HSMS-GS 動作時は、指定の DeviceID (SessionID) に対して Select 要求を発行します。
- ・ [Deselect] ..... Deselect : 相手側に「Deselect」を要求します。  
    HSMS-GS 動作時は、指定の DeviceID (SessionID) に対して Deselect 要求を発行します。  
    (注 3) HSMS-SS では、通常 Deselect は使用しません。
- ・  ..... 選択したシナリオの自動実行スクリプトを実行開始します。
- ・  ..... 現在実行している自動実行スクリプトの実行を停止します。
- ・  ..... 現在実行している自動実行スクリプトの実行を一時停止します。
- ・  ..... 現在実行している自動実行スクリプトを 1 行実行して一時停止します。

(c) 通信トレース表示

送受信した SECS 通信メッセージのトレース表示、また指定（[表示]メニューの [スクリプトトレース表示]）により、自動実行スクリプトの実行文の表示を行います。

(注 1) SECS 通信メッセージのリスト形式表示の形態は、.ini ファイルの以下のパラメータで決まります。

- ・ TRCTTYPE ..... 通信トレースへの通信メッセージ出力形式
  - bit#2 ..... 項目データ表示形式
    - =0: 各項目を 1 行のみで表示し、1 行に納まらない場合は、後部を省略する。
    - 1: 各項目を複数行で表示し、数値項目は 1 行に 20 データ、文字列項目は 100Bytes 分を表示する。
  - bit#4, 5, 6 ... リスト出力形式
    - (通常 =2 としてご使用ください。)
    - =0: TDS オリジナル形式
    - 2: SML 形式
  - bit#7 ..... データ項目名表示
    - (通常は =1 としてください。)
  - bit#8, 9 .... メッセージ定義ファイルの形式
    - (通常は =0 として SML 形式のメッセージ定義ファイルをご用意ください。)
    - =0: SML 形式

(注 2) SECS 通信メッセージの 16 進数表示の形態は、.ini ファイルの以下のパラメータで決まります。

- ・ TRCTTYPE ..... 通信トレースへの通信メッセージ出力形式
  - bit#3 ..... 16 進数表示形式
    - =0: 1 行に 16Bytes 表示する。
    - 1: 1 行に 20Bytes 表示する。

(注 3) 送信 1 次メッセージに対して 2 次メッセージの受信がなく、T3 タイムアウトが発生した場合、.ini の設定 ((DEVMODE&0x0100)!=0) により、通信ドライバ (TDS.dll) が自動的に S9F9 を発行しますが、これらの通信ドライバが自動発行する SECS メッセージも、通信トレース表示ウインドに表示します。

(注 4) 例えば、T3 タイムアウトが発生した後に、相手側から対応する 2 次メッセージを受信した場合、.ini の設定が、無効な 2 次メッセージを受信しない設定 ((DEVMODE&0x0002)==0) の場合は、受信した、その時点では無効な 2 次メッセージは、通信トレース表示ウインドには表示しません。これらのメッセージも通信トレース表示ウインドに表示するためには、.ini の DEVMODE の設定を ((DEVMODE&0x0002)!=0) としてください。  
ただし、DEVMODE の設定にかかわらず、通信トレース・ファイルには全ての SECS 通信メッセージを出力します。

(注 5) HSMS-SS 接続時の LinkTest メッセージは、通信トレース表示ウインドには表示しません。  
通信トレース・ファイルには (.ini の TRCTLEVEL>=9 の場合) 出力します。

(注 6) SECS-1 接続時の通信制御コードの送受信は、通信トレース表示ウインドには表示しません。  
通信トレース・ファイルには (.ini の TRCTLEVEL>=6 の場合) 出力します。

(注 7) 通信トレースのファイルへの出力は、.ini の TRCDIR、TRCTTYPE、TRCTOUT、TRCTLEVEL、TRCTSIZに依存します。詳細は、前述の TDS.pdf 2.1 (1) をご参照ください。

#### (d) 通信メッセージ編集

指定のメッセージ定義ファイルの内容を表示します。

自分側の送信対象メッセージは、メッセージ名称の頭に「!」文字を付与して表示します。

従って、[設定]メニューで、[メッセージ方向評価]をチェックしている場合は、「!」が付いたメッセージのみを相手側に送信することができます。チェックしていない場合は、全てのメッセージを送信することができます。

- ・ メッセージ名称をダブルクリックすると、現在設定されているメッセージを構成する項目の内容で、相手側に送信します。
- ・ メッセージ名称を右クリックすると、以下のメニューを実行することができます。
  - ・ [送信] : メッセージ名称のダブルクリックと同様です。
  - ・ [初期化] : メッセージを構成する各項目を、メッセージ定義ファイルでの定義内容に戻します。
- ・ メッセージ名称を開き、そのメッセージを構成する各データ項目を表示すると、データ項目値を変更することができます。

項目名称をダブルクリックする、もしくは右クリックで表示するメニューの[編集]を実行すると、「メッセージ編集」ダイアログが現れるので、データ項目値を編集します。

  - ・ 可変個数リスト項目は「設定個数」を入力します。
  - ・ 複数個数で構成する数値項目は、','で区切って入力します。数値項目に16進数を指定する場合は、接頭句として0xを付与します。(例:12,0x3a,4321,0xff)
  - ・ 複数個数で構成する項目は、項目値の入力、設定個数の入力とも可能で、後で指示した個数が有効となります。

(注1) メッセージが可変個数リストを持つ場合、まず可変個数リストの個数を確定させる必要があります。また、複数の可変個数リストを持つ場合は、最上位レベルのリスト、同レベルであれば、上方のリストの個数を確定し、全可変個数リスト個数を確定した後、他のデータ項目の設定を行ってください。

(注2) ここで編集したメッセージは、メッセージ名称のダブルクリック（もしくは[送信]）でのメッセージ送信、及び、自動実行シーケンス内でのメッセージ送信に使用します。

(注3) 自動実行スクリプトの send 文により構築した送信メッセージは、このウインドには反映しません。

(注4) [Start]により、シミュレーション処理を開始すると、メッセージ定義を初期化します。仮に前回と同じメッセージ定義ファイルを使用する場合でも、情け容赦なく初期化します。

#### (e) 自動実行スクリプト

自動実行シナリオを選択し、自動実行シーケンスの実行を開始すると、現在の実行状態を表示します。次に実行する行の先頭に「>」を付与して示します。

#### (f) スクリプト変数

自動実行シーケンス中で使用するスクリプト変数、及び特殊変数の値を表示します。

スクリプト変数は、その値を変更することが可能です。

スクリプト変数、特殊変数の詳細は、第3章(5)を参照してください。

% (Dec) 及び % (Hex) は、同一の整数型スクリプト変数です。

& は、実数型スクリプト変数です。

\$ は、文字列型スクリプト変数です。

各スクリプト変数の「セル」をクリックし、直接値をキーインすることで、該当スクリプト変数値を変更することができます。

(注1) 特殊変数、スクリプト変数は、自動実行シナリオの実行開始と共に初期化します。

### 3. シナリオ実行スクリプト言語仕様

本プログラムにおいて、通信処理の自動実行を行うためのシナリオ実行スクリプト・ファイルの記述方法に関して、簡単に記述します。

#### (0) 全体構成

1つのシナリオ実行スクリプト・ファイル（.ssl ファイル）内に、複数の実行単位を記述することが可能です。自動実行時は、選択したスクリプト・ファイル内に記述された実行単位の中から1つのみを選んで、動作させます。動作は、記述したステートメントを連続して動作させることも、1ステップずつ動作させることも可能です。

- ・ 1つの実行単位は、“program” 文から始まり、“end” 文で終了します。
- ・ 行中に “//” 表記があると、それ以降は、コメント扱いとなります。
- ・ 行末が “¥” 表記である場合は、次行へ行が継続するものとして扱います。ただし、継続行を含めた、全体の行の長さは、2000 バイトまでです。

(注 1) 継続記号の前に “//” が存在する場合、コメントは無視して次行に継続します。

例えば・・・

```
%1=123    // コメントです ¥
```

```
%2=456
```

は、“%1=123 %2=456” となります。

(注 2) 本プログラムは、「善意のユーザ」が「正しく」使用することを想定しています。誤った文法等の記述があっても、特にエラー報告等を行わず、それなりに処理してしまいます。

(注 3) 特殊変数 !TS に値を設定する事で、スクリプト各行の実行速度を遅延することができ、スクリプトの処理内容確認を（若干）容易にすることが可能です。詳しくは (5) set をご参照ください。

(注 4) 本プログラムは自動応答モードで動作する場合、基本的には受信する可能性のある全てのメッセージをメッセージ定義ファイル（.sml、.csv）に定義する必要がありますが、シナリオ実行スクリプトを使用し、指定の SF-Code の受信に関して、そのメッセージ構造に関わらず、所定の 2 次メッセージ返信を行う事ができます。これは、シナリオ実行スクリプトの受信処理は、定義済みのメッセージ構造に対して行うのではなく、指定の SF-Code に関して行うことによります。実際のスクリプト記述方法等は、添付のサンプル・スクリプト（Sample.sml）の AnyS5F1\_S6F11 をご参照ください。

本スクリプトは、以下のステートメントで構成します。

- ( 1) program (prog) ..... 実行単位の開始を宣言
- ( 2) function (func) ..... 関数 (サブルーチン) の開始を宣言
- ( 3) end ..... 実行単位 及び 関数の終了を宣言
- ( 4) exit ..... 実行終了
- ( 5) set (省略可) ..... スクリプト変数への代入、演算
- ( 6) if ..... 条件分岐
- ( 7) while ..... 繰り返し制御
- ( 8) break ..... 現在の while ブロックから抜ける
- ( 9) continue ..... 現在の while ブロックの最後の endwhile に移行
- (10) goto ..... 指定のラベル行へ処理を移動
- (11) call ..... 指定の関数 (function 単位) を実行
- (12) return ..... 関数 (function) の実行を終了し、呼出位置に戻る
- (13) exec ..... 外部プログラムの実行
- (14) reserve (resv) ..... スクリプトが処理する対象の受信待ち SF-Code 指定
- (15) cancel ..... 受信待ち SF-Code のキャンセル
- (16) wait ..... reserve にて指定した SF-Code のメッセージの受信待ち
- (17) release (rels) ..... 受信した SECS メッセージ領域の開放
- (18) receive (recv) ..... 受信した SECS メッセージの解析
- (19) send ..... 指定 SECS メッセージの送信
- (20) sleep ..... 指定時間処理をお休み
- (21) display (disp) ..... スクリプト変数を通信トレースに出力
- (22) print ..... 指定文字列を通信トレースに出力
- (23) pause ..... STEP モード ( 1 行ずつ実行) へ移行



- ( 1) program (prog) ..... 実行単位の開始を宣言

```
program XXXXX   もしくは   prog   XXXXXX  
~~~~~
```

XXXXXX : プログラム名称  
ここから、この名称のシナリオ実行シーケンスが始まる。  
スクリプト実行時に、実行するシーケンスのプログラム名称を指定すると、その名称に  
合致するプログラム名称のシーケンスを実行する。  
一連のプログラム実行単位の最後は end 文で終了すること。

- ( 2) function (func) ..... 関数（サブルーチン）の開始を宣言

```
function XXXXX  もしくは   func   XXXXXX  
~~~~~
```

XXXXXX : 関数名称  
ここから、この関数名称のシーケンスが始まる。  
関数は、他から call 文で処理が移行し、function 内の return 文、もしくは最後の  
end 文により、処理が call 文の次に移動する。  
一連の関数実行単位の最後は end 文で終了すること。

- ( 3) end ..... 実行単位 及び 関数の終了を宣言

```
end  
~~~
```

- ( 4) exit ..... 実行終了

```
exit  
~~~~
```

( 5) set (省略可) ..... スクリプト変数への代入、演算

```
[set] par=exp [par=exp [ ... ]]  
~~~~~
```

(注 1) キーワードの “set” は省略可能

par=exp : スクリプト変数、スクリプト定数 及び 一部の特殊変数に値を設定する。

スクリプト変数には以下があり、いずれの項目にもいずれの変数も指定できる。

\$00 .. \$99 : 文字列変数

%00 .. %99 : 整数変数

&00 .. &99 : 実数変数

%[0], \$[%2] といった表記も可能であり、 %[0] は %00 と同一、 \$[%2] は、評価時点での %2 値を \$変数の添字値とする。(注) []内に指定可能な変数は %変数のみ。

スクリプト定数として以下を指定できる。スクリプト定数は、単純代入のみ可能 (演算代入は不可)。

!TS : 各スクリプト実行行の実行遅延時間 (m秒)。デバッグ時等に >0 値を指定するとスクリプトの実行速度が遅くなり、動作を追いやすくなる。

!TW : SECS メッセージ受信後、スクリプトが受信処理を行い、Release 文を発行するまでのタイムアウト時間 (m秒で指定するが、分解能は秒単位)。!TW への代入は、次の SECS メッセージ受信から有効になる。

以下の特殊変数を par に指定できる。特殊変数は、単純代入のみ可能 (演算代入は不可)。

!DX : 次回送信 1 次メッセージに付与するデバイス I D

exp として、以下を指定できる

定数 : “文字列”、数値

文字列変数 (\$xx) に代入する “文字列” 中に、スクリプト変数を埋め込むことができる。スクリプト変数、特殊変数自体 (例えば \$09、!ER) を記述する場合は、スクリプト変数等の前に、同一文字を置きエスケープする。(例えば \$\$09、!!ER と記述する。) また、“文字列” 中に “ を含める場合は ¥ でエスケープすること。

(例えば \$1=“¥“ABC¥” - ¥“XYZ¥”” と記述すると \$1 は “ABC” - “XYZ” となる。)

16 進数は 0xff 形式で指定する。

変数 : \$9、%9、&9

特殊変数 : !TM :	現在時刻	(通算秒)
!YY :	現在年下 2 桁	(YY : 0 - 99)
!MD :	現在月	(MM : 1 - 12)
!DD :	現在日	(DD : 1 - 31)
!HH :	現在時	(HH : 0 - 23)
!MM :	現在分	(MM : 0 - 59)
!SS :	現在秒	(SS : 0 - 59)
!T3 :	T3 Timeout 値	(m秒)
!TS :	スクリプト実行遅延時間	(m秒)
!TW :	スクリプト受信時処理 Timeout 値	(m秒)
!ER :	SECS 送受信実行エラー・コード	
	=0:OK 1:Receive error 2:Send error 3:Wait Timeout	
!DX :	次回送信 1 次メッセージに付与するデバイス I D	
!DV :	最終受信 デバイス I D	
!SC :	最終受信 S-Code	(8bit)
!FC :	最終受信 F-Code	(8bit)
!SF :	最終受信 SF-Code	(16bit S-Code + F-Code)
!CD :	最終受信 SF-Code	(S99F99 形式文字列)
!MN :	最終受信 メッセージ名称	(メッセージ定義ファイルでの名称)
!SD :	最終受信 Source-ID	(16bit)
!XD :	最終受信 Transaction-ID	(16bit)
!SX :	最終受信 System-byte	(32bit) (!SD*0x00010000+!XD)

<< 次ページに続く >>

<< 前ページから続く >>

式           : 変数 1 演算子 変数 2  
              変数 1、変数 2 は定数でもよい  
              変数 1 は 特殊変数 (整数型変数) でもよい  
              変数 1 は par と同じ型でなければならない。  
              演算子は par の種別毎に以下の何れかとする  
文字列変数 : +  
整数変数   : +、-、\*、/、^、%、&、|、<<、>>  
実数変数   : +、-、\*、/、^

(注 2) par=exp 中には、演算子の前後を含めて、空白を置くことはできない。

即ち %1=%2+3 は O K。%1 = %2+3 は N G。

(注 3) 文字列定数に日本語を含めると、正しく解釈しない場合がある。

( 6) if ..... 条件分岐

```
if exp [then] statement          // then は省略可能
~~~~~                          // (注 1) この場合、次に elseif、else を置いて
                                //      ブロック if を構成することはできない。
```

もしくは

```
if exp0 then
  exp0 が true の場合の statement ブロック
elseif exp1 then                // elseif ブロックは省略可能
  exp1 が true の場合の statement ブロック
else                             // else ブロックは省略可能
  全ての if、elseif の条件式が false の場合の statement ブロック
endif
~~~~~
```

exp : 比較演算式 もしくは 定数 (true もしくは false)

式 1 比較演算子 式 2

式 1 として、以下を指定できる

変数 : \$99、%99、&99

特殊変数 : !TM : 現在時刻 (通算秒)  
 !YY : 現在年下 2 桁 (YY : 0 - 99)  
 !MD : 現在月 (MM : 1 - 12)  
 !DD : 現在日 (DD : 1 - 31)  
 !HH : 現在時 (HH : 0 - 23)  
 !MM : 現在分 (MM : 0 - 59)  
 !SS : 現在秒 (SS : 0 - 59)  
 !T3 : T3 Timeout 値 (m秒)  
 !TS : スクリプト実行遅延時間 (m秒)  
 !TW : スクリプト受信時処理 Timeout 値 (m秒)  
 !ER : SECS 送受信実行エラー・コード  
       =0:OK 1:Receive error 2:Send error 3:Wait Timeout  
 !DX : 次回送信 1 次メッセージに付与するデバイス I D  
 !DV : 最終受信 デバイス I D  
 !SC : 最終受信 S-Code (8bit)  
 !FC : 最終受信 F-Code (8bit)  
 !SF : 最終受信 SF-Code (16bit S-Code + F-Code)  
 !CD : 最終受信 SF-Code (S99F99 形式文字列)  
 !MN : 最終受信 メッセージ名称 (メッセージ定義ファイルでの名称)  
 !SD : 最終受信 Source-ID (16bit)  
 !XD : 最終受信 Transaction-ID (16bit)  
 !SX : 最終受信 System-byte (32bit) (SD+XD)

式 2 として、以下を指定できる

定数 : "文字列"、数値 (16 進数は 0xff 形式)

変数 : \$99、%99、&99

比較演算子として、式 1 の形態別に以下を指定できる

文字列 : ==、!=、<、<=、>=、>

整数 : ==、!=、<、<=、>=、>

実数 : ==、!=、<、<=、>=、>

<< 次ページに続く >>

<< 前ページから続く >>

(注 1) exp 中には、演算子の前後を含めて、空白を置くことはできない。  
即ち %1==%2 は OK。%1 == %2 は NG。

(注 2) 文字列定数に日本語を含めると、正しく解釈しない場合がある。  
(ちょっと手抜きです・・・)

label       : ジャンプ先のラベル (goto 参照)

statement : 実行文

(注 2) ブロック if 文を記述する場合、statement ブロック中に、if をネストすることが可能。  
(注 3) elseif 文の else と if の間、endif 文の end と if の間に空白を置くことはできない。

( 7) while ..... 繰り返し制御

```
while exp do
  exp が true の間、繰り返し実行する statement ブロック
endwhile
~~~~~
```

exp : 比較演算式 ((6) if 文参照)  
exp として true を指定すると、無限ループとなる。

(注 1) while ブロックのネストを構成することは、実行プログラム全体 (call 文による関数内も含めて) 最大 30 レベルまで可能。

(注 2) if -- endif ブロックと、while -- endwhile ブロックがかぶってしまう、などというケースは想定しない。誤ったブロック構成であったとしても、なんらエラー表示のようなものは発生しない。

(例) 以下は誤った構成。でもそれなりに実行する。

```
if true then
  %2=0
  while %2==0 then
    なんとか、なんとか
  endif
endwhile
```

( 8) break ..... 現在の while ブロックから抜ける

```
break
~~~~~
```

( 9) continue ..... 現在の while ブロックの最後の endwhile に移動

```
continue
~~~~~
```

(10) goto ..... 指定のラベル行へ処理を移動

```
goto label
~~~~~
```

label : ジャンプ先のラベル  
ラベルは "XXXXX:" と記述し、ラベル行には実行文を置いてはいけない。  
ラベル行は、現在実行している処理単位 (program ~ end もしくは function ~ end) 内に無くてはならない。

- (11) call ..... 指定の関数 (function 単位) を実行

```
call func_name  
~~~~~
```

func\_name : 呼出先関数の名称

- (12) return ..... 関数 (function) の実行を終了し、呼出位置に戻る

```
return  
~~~~~
```

- (13) exec ..... 外部プログラムの実行

```
exec [+w] "program [parameters...]"  
~~~~~
```

+w : 本オプションを指定すると、指定のプログラムを、ウインドを表示した状態で起動する。  
指定しない場合、指定プログラムはウインドを持たない。

program : 起動する外部プログラム (.exe のパス名称)

parameters : prog に渡す引数。print 文と同様、スクリプト変数、特殊変数を含むことが可能。  
また、最後 (閉じ " の前) に '&' を付与すると、指定プログラムの終了を待たずに、  
処理を継続する。

(注 1) 標準入出力のリダイレクションは使用できない。

(参考) exec 文で起動したプログラムで、起動引数に応じた SECS メッセージ項目の内容を作成し  
その結果を、例えば "secs\_item.dat" と云ったファイルに記録し、引き続く send 文で、SECS  
メッセージ項目の指定に "send SxFx\_xx @file=secs\_item.dat,section" として、送信する SECS  
メッセージを作成することができる。send 文を参照すること。

(14) reserve (resv) ..... スクリプトが処理する対象の受信待ち SF-Code 指定

```
reserve SxFx [SxFx] ...   もしくは   resv SxFx [SxFx] ...  
~~~~~
```

SxFx : 受信予約対象の SF コード (注) メッセージ名称ではない  
SOF0 と記述すると、全てのメッセージを待つ。

(説明) スクリプトで処理する受信メッセージを予約する。指定した SF-Code は、既に予約済みの他の SF-Code に加えて予約状態となる。指定の SF-Code が既に予約済みの場合は、何も行わない。受信予約を行わない SF を受信した場合、1 次メッセージの場合は、自動応答により 2 次メッセージを返信し、2 次メッセージの場合は、そのまま受け流す。

受信予約した SF を受信した場合は、一旦 WAIT 文により、メッセージ受信を待ち、その後、receive 文にて受信メッセージを解析処理した後、release 文にて、受信メッセージ領域を「必ず」開放処理する必要がある。

release 文を実行した時点で、その受信メッセージに関する処理が完了し、reserve 文での予約 SF-Code を解除する。従って、同一の SF-Code で再度受信待ちをする場合は、再度 reserve 文を実行しなければならない。

(注意) 例えば、S1F13 受信、S1F14 送信、S1F1 受信、S1F2 送信、といったシーケンスの場合、S1F14 の送信後、相手側の S1F1 送信が素早い場合、こちらの S1F2 受信準備ができていないと、スクリプト内で S1F2 受信ができない可能性がある。従って、こちらの送信に対する、相手側の受送信が素早い場合、次の受信メッセージの reserve 処理を、送信処理よりも先に行っておく必要がある。即ち、前記シーケンスの場合は、以下のようになる。

```
reserve S1F13           // S1F13 を受信予約  
wait                   // S1F13 受信待ち  
receive S1F13           // 必ずしも必要はない  
release S1F13           // S1F13 受信情報を解放  
  
reserve S1F1            // S1F14 の send の前に、次の S1F1 の reserve を行う  
send    S1F14_XX        // S1F14 返信  
wait                   // S1F1 受信待ち  
receive S1F1            // 必ずしも必要はない  
release S1F1            // S1F1 受信情報を解放  
send    S1F2_XX         // S1F2 返信
```



(15) cancel ..... 受信待ち SF-Code のキャンセル

```
cancel SxFx [SxFx] ... or cancel all
~~~~~
```

SxFx : キャンセル対象の、受信予約 SF コード ((注) メッセージ名称ではない)  
all : 全ての受信予約 SF コードをキャンセルする。

(説明) スクリプトで処理する受信メッセージの予約を取り消す。

指定の SF-Code を予約済みのリストから削除する。受信したメッセージに対して release 文を実行すると、そのメッセージの SF-Code は自動的に予約リストから削除する。従って、その SF-Code に対して cancel 文を実行する必要はない。

同一の SF-Code で再度受信待ちをする場合は、再度 reserve 文を実行しなければならない。

(16) wait ..... reserve にて指定した SF-Code のメッセージの受信待ち

```
wait [msec]
~~~~~
```

msec : wait 状態を解除するタイムアウト値 (m秒)  
省略した場合 及び msec=0 の場合は、タイムアウトしない。  
スクリプト変数、特殊変数 (!T3) を使用することが可能。

(説明) reserve 文により受信予約した SF コードのメッセージを受信するまで待つ。

wait 文にて、受信したメッセージは「必ず」release 文で受信処理の完了を宣言する必要がある。

wait 文で受信した SECS メッセージの情報は、この時点で特殊変数に格納される。

タイムアウトが発生すると !ER==3 となる。

(17) release (rels) ..... 受信した SECS メッセージ領域の開放

```
release SxFx      もしくは      rels SxFx
~~~~~          ~~~~~
```

SxFx : 処理対象の SF コード ((注) メッセージ名称ではない。reserve 文で指定したもの)

(説明) wait 文で受信したメッセージの receive 文による解析を終了し、受信データ領域を開放する。  
receive 文は必ずしも実行する必要はない。

(注 1) 受信した SF-Code を引き続き受信対象とする場合は、release 文を使用せずに、再度 reserve 文にて、その SF-Code を指定する。この時点で受信対象とはしない場合は、必ず release 文にて受信 SF-Code に関する開放処理を行わなければならない。

特に Multi Open Transaction が発生するケースにおいて、同一 SF-Code を連続して受信処理する場合は、release 文を記述せず、reserve 文を記述すること。

下記 (例 1)、(9) reserve 文、(10) wait 文参照

(注 1) wait 文から SECS メッセージ受信により処理がスクリプトに戻ってから、release 文により、当該 SECS メッセージの受信処理を完了するまでは 10 秒 (set !TW=20000 等により、変更が可能。set を参照すること) 以内でなければならない。

特に STEP 実行中等は要注意。

(注 2) release 文を実行すると、それ以降、特殊変数 (!ER 等) の内容は保証されない。

即ち、wait 文以降、例えば !CD の値により処理を分岐する等の処理を行うことができるが、一旦 release 文を実行すると、それ以降、それらの判定等の処理を行うことができなくなる。

release 文以降の、例えば send 文等で使用する場合には、release 文実行前に、特殊変数の値を一旦スクリプト変数に代入しておくこと。

(例 1) 例えば、S6F11 を連続受信し、S6F12 を返信するといったシーケンスの場合、以下ようになる。

```
reserve S6F11      // S6F11 を受信予約
while true do      // 無限ループ
  wait             // S6F11 受信待ち
  receive S6F11     // 必ずしも必要はない
//release S6F11    // 引き続き S6F11 を reserve する場合、release 文を使用しない
  reserve S6F11     // S6F12 の send の前に、次の S6F11 の reserve を行う
  send S6F12_XX     // S6F12 返信
endwhile
```

(18) receive (recv) ..... 受信した SECS メッセージの解析

receive SxFx [scv=item [scv=item ... ]]   もしくは   recv SxFx [scv=item [scv=item ... ]]  
~~~~~

SxFx       : 処理対象の SF コード                   ((注) メッセージ名称ではない。reserve 文で指定したもの)

scv=item : 受信メッセージからデータ値を取得し、スクリプト変数への代入を指示する。

scv (スクリプト変数) には、以下があり、いずれの項目にもいずれの変数にも、全ての型のメッセージ項目を指定可能。

\$00 .. \$99 : 文字列変数

%00 .. %99 : 整数変数

&00 .. &99 : 実数変数

item は、メッセージ定義ファイルに設定した項目名であり、不定個数項目の場合は "item:9" (9 は 1 以上の項目番号、':' はコロン) といった、項目番号を付与して指定する。

また、数値項目の場合、複数の項目値がある場合は、先頭項目でない場合は、

"item:9" (9 は 0 以上の項目内添字値、';' はセミコロン) 形式で指定する。

例えば、以下の様に記述する。記述が長くなる場合は、行末に '¥' を置くと次行に継続する。また、複数行に分けて記述することも可能。

receive S2F49 \$1=RCMD \$2=LOTID %1=STKD %2=IDATA:1;7 ¥  
                  &2=RDATA:3

もしくは

receive S2F49 \$1=RCMD \$2=LOTID  
receive S2F49 %1=STKD %2=IDATA:1;7 &2=RDATA:3

(注 1) 最後に受信したメッセージが指定の SF の場合に、その SF コードのメッセージに対する処理を行う。指定 SF コード以外のメッセージに対しては、何も処理しない。

(注 2) receive 文の前に、wait 文を実行しなければならない。

receive 文は必ずしも実行する必要はない。

**release 文 (場合により reserve 文で代用可) は必ず必要。**

(14) reserve 文、(16) wait 文参照

(注 3) 1 回の wait でのメッセージ受信に対応する receive 文は、複数回実行することができる。

処理を行うメッセージ項目が多い場合等、複数回に分けて実行すると良い。

(注 4) receive 文による受信メッセージの解析処理が完了したら、「必ず」release 文 (もしくは、場合により reserve 文) を実行すること。

(19) send ..... 指定 SECS メッセージの送信

```
send SxFx_xx [item=value [item=value ... [@file="data_file.dat[, sect]] ...]]
~~~~~
```

SxFx\_xx : メッセージ定義ファイルに設定したメッセージ名称 (注) S Fコードではない)

item=value : 指定メッセージを構成する項目に対して、値を設定する。  
使用するメッセージ・データは、現在設定されている内容に、このオプション指定での値を上書きしたものを使用する。ここで設定した内容は、元のメッセージ・データには反映しない。

不定個数リスト項目 : item=99  
文字列項目 : item="xxxxxxxxx"  
数値項目 : item=99 or =0xff or =99.999 or =9.999e999

また、スクリプト変数、特殊変数を value として指定できる。  
スクリプト変数、特殊変数に関しては set 文を参照すること。いずれの項目にもいずれの変数も指定できる。  
文字列項目に文字列定数を設定する場合は、文字列定数中に、スクリプト変数、特殊変数を埋め込むことができる。例えば、文字列項目に対して "ABC %1 XYZ !!TM=!TM" といった指定が可能。(set 文参照)

例えば、以下の様に記述する。記述が長くなる場合は、行末に '¥' を置くと次行に継続する。  
send s2f49\_hload RCMD="LOADDATA" LOTID="LOTID です" ¥  
PORT=2 MTKD=%5 ¥  
PDATA="%12,%13,%14,%15" RECIPE=\$1 CASSETE=\$3

(注 1) 指定のメッセージに不定個数リストが存在する場合は、全ての不定個数リストのリスト個数を、上位のリスト項目から順番に指定した後、通常項目の指定を行うこと。リスト個数を確定すると、そのリスト内の各項目名称は、ルールに従って変化する。不定個数リスト確定時の項目名称のルールに関しては、第 1 章記載の TDS.pdf を参照すること。(\_TDSMDMssgBuild() の注記、及び 例示を参照すること。)

(注 2) 複数個数の項目値を指定する場合は、value の全体を " で括り  
item="1, 2, %12, %13, 99" のように指定すること。

(注 3) 項目値の一部のみを指定することも可能。従って receive 文でのような形式を用いて  
"item;3=12, 13, %5, %6" といった指定が可能。連続していないデータを設定する場合は同一項目を複数回指定する事ができるが、連続して指定しなければならない。  
例えば、10 個のデータ値を持つ項目 ITEM の、[3]=3、[5]=5、[6]=6 とする場合は、  
ITEM;3=3 ITEM;5="5, 6" と連続して指定する。

```
@file="data_file.dat,section"
```

： 指定メッセージを構成する項目の値を、ファイル（項目値設定ファイル）で指定する。

data\_file.dat : 項目値を指定したファイル・パス名称

section : 指定ファイル内の使用するセクション名称

項目値指定ファイルは、以下の .ini ファイル構成とする。

```
// の後ろは、行頭、行中にかかわらず、コメントとする
```

```
// まず、セクションを指定しない場合に使用する項目値を指定する。
```

```
RCMD = "LOADDATA" // データ種別
```

```
PORT = 3 // ポート番号
```

```
// 次に、セクションを指定した場合の項目値を指定する。
```

```
[SECT01]
```

```
RCMD = "LOAD#1" // 1 番目のデータ種別
```

```
PORT = 2 // 1 番目のポート番号
```

```
// 以下、同様に記述する。
```

```
// 例えば、SECT01 の設定を使用する場合は、このファイルが "sample.dat" だとすると
```

```
// send s2f49_hload @file="sample.dat, SECT01" と記述する。
```

(注 4) 送信 1 次メッセージに付与するデバイス I D は !DX に設定された値を使用する。

!DX 値が =0 の場合は、設定ファイル (.ini) の DEVID に指定した先頭のデバイス I D を使用する。

送信 2 次メッセージに付与するデバイス I D は、直前に受信した 1 次メッセージに付与されていたデバイス I D (!DV) を使用する。

(注 5) 送信異常が発生すると !ER==2 となる。

- (20) sleep ..... 指定時間処理をお休み

```
sleep msec  
~~~~~
```

msec : 待ち時間 (m秒)  
スクリプト変数を使用することが可能。

- (21) display (disp) ..... スクリプト変数を通信トレースに出力

```
display [scp]   もしくは   disp [scp]  
~~~~~          ~~~~~
```

scp : スクリプト変数  
省略した場合は、全スクリプト変数 及び 特殊変数を表示する。

- (22) print ..... 指定文字列を通信トレースに出力

```
print format  
~~~~~
```

format : プリント出力書式  
書式中に \$99, %99, &99, !ER 等を含めることができる。  
(注) 日本語処理を省略している。(日本語が使えないわけではない。)

- (23) pause ..... STEP モード (1行ずつ実行) へ移行

```
pause  
~~~~~
```

会話 (STEP) モードに移行する。