ADVANTEST®

株式会社アドバンテスト

D5112
ISDN プロトコル・アナライザ
（シミュレーション機能）
取扱説明書

MANUAL NUMBER FOJ-8324241C01

適用機種

D5112 にオプション 13 を搭載したもの
D5112B

初版 1993 年 9 月 10 日
Printed in Japan

この取扱説明書の使い方

1. この取扱説明書は、D5112 シリーズのシミュレーション機能について説明しています。D5112 シリーズの他の機能および操作方法については、別冊のD5112 シリーズ取扱説明書を参照して下さい。

2. この取扱説明書は、以下の製品に標準付属品として添付されます。

   - D5112 にオプション13を搭載したもの
   - D5112B

【構成】

1. シミュレーション言語	: シミュレーション言語(PSL51) の機能と仕様について説明します。
2. プログラミング構造	: PSL51 の基本的なプログラミング構造について説明します。
3. 簡単なプログラムの作成	: プログラムを用いて、画面に文字を表示させるまでを順に説明します。
4. プログラムを用いたシミュレーション	: プログラムを用いたシミュレーションの実行手順を説明します。
5. コンパイラによるエラー・コード	: エラー・メッセージとエラー概要を説明します。
6. プログラムの実行	: オブジェクト・プログラムを実行する場合について説明します。
7. メッセージ・ビルダ	: メッセージ・ビルダの作成方法、操作方法を説明します。
8. セーブ／ロード・メニュー	: オブジェクト・プログラムのセーブ／ロードおよびメッセージ・データのセーブ／ロードの手順を説明します。
9. プログラムを用いないでシミュレーション関数を実行する	: プログラムを用いないでシミュレーション関数を実行する方法および使用できるコマンドについて説明します。
10. シミュレーション言語(PSL51)の構文について	: シミュレーション言語(PSL51) の構文について説明します。
11. 機能別関数一覧	: 機能別の関数一覧表です。
12. D チャンネル・シミュレーション	: D チャンネル・シミュレーションの関数について機能別に説明します。
13. B チャンネル・シミュレーション	: B チャンネル・シミュレーションの関数について機能別に説明します。
14. シミュレーションを使いこなすために	: タイマの使用方法、SAPI, TEI 管理などについて説明します。

目次

図一覧

表一覧

1. シミュレーション言語

1.1 PSL51

シミュレーション言語（PSL51 : Protocol Simulation Language for D5112以下PSL51と記します。）には、以下の機能があります。

(1) ユーザーにより作成されたフレームの送出

(2) 受信したフレーム内容の判定

1.2 PSL51 言語仕様

表 1 - 1 PSL51 言語仕様

データ型	整数型	符号付32ビット (-2147483648～+2147483647)
配列	1 次元配列	
演算子	＋ － ＊ ／	加算 減算 乗算 除算
関係演算子	＞ ＜ ＞＝ ＜＝ ＝＝ ！＝	より大きい より小さい より大きいか、等しい より小さいか、等しい 等しい 等しくない
論理演算子	！	否定
ビット毎の 論理演算子	＆ ｜	ビット毎の論理積 ビット毎の論理和
関数	FUNC	整数型の値を持つ
文	代入文 IF 文 FOR 文 WHILE 文 CASE 文 EXIT 文 RETURN文	val = a1 ÷ BB * XYZ1 IF a=1 THEN X=0 ELSE X=1 END FOR i=0 TO 100 DO ..... END WHILE x==1 ..... END CASE XY ÷ 1 OF '1''2' ..... END WHILE 文のループから抜ける 関数からの復帰

2. プログラミング構造

2.1 PSL51 のプログラミング構造例

PSL51 の基本的プログラミング構造例を以下に示します。

```
CHANNEL   D                                  } シミュレーションの宣言文
LAYER     2
SIMMODE   TE
PFEED     OFF

ARRAY  A[8], XYZ[10]                         } 配列の宣言文
ARRAY  X[3] = [0,1,2]
ARRAY  Y[10] = [0,1,2,3[7]]

FUNC  MAIN( )                                } 主プログラム関数
          :
      SUB1( )
          :
RETURN

FUNC  SUB1( )                                } 関数
        :
      RT =SUB2( )
RETURN

FUNC  SUB2( )                                } 関数
      BB = 3
        :
RETURN(BB)
```

プログラムはMAIN（　）という名の主プログラム関数から実行します。従って、プログラム中にMAIN（　）という名の関数が必ず存在しなくてはなりません。

2.2 シミュレーションの宣言文

D チャンネルのシミュレーションを行なう場合、宣言文により、あらかじめ、使用モードを明記しておく必要があります。
宣言文により定義するものは、以下の 7種類です。

(1) CHANNEL D/B

D ：D チャネル上でシミュレーションを行ないます。
B ：B チャネル上でシミュレーションを行ないます。

(2) LAYER 2/3

2 ：トランスペアレント・モードによる実行です。
3 ：レイヤ2 自動実行モードによる実行です。

(3) SIMMODE

TE ：TEモードによる実行です。
NT ：NTモードによる実行です。

（注）(7)の宣言文がU 点インタフェース時は、NTモードのみ有効です。
TEと記述しても、NTにて動作します。

(4) PFEED OFF/NORM/RVS

OFF ：給電なし。
NORM ：ノーマル給電を行ないます。
RVS ：リバース給電を行ないます。

（注）(7)の宣言文が、基本インタフェース時のみ有効で、その他では無効です。
(3)、(4)の宣言文は、D チャネルとB チャネル・プログラムの両方で宣言される場合、D チャネル・プログラムで宣言される方が有効です。
(4)の宣言文は、本器が(3)でNTに宣言されたときのみ有効です。

(5) ADDRESS DTE/DCE

DTE ：自局のアドレスをDTE に指定します。
DCE ：自局のアドレスをDCE に指定します。

(6) MODULO 8/128

8 ：モジュロ8 にてシミュレーションを行ないます。
128 ：モジュロ128 にてシミュレーションを行ないます。

（注）(5)、(6)の宣言文は本器が(1)でB チャネル・シミュレーションに宣言されたときのみ有効です。

(7) INTERFACE BRI/PRI

BRI :基本インタフェースでシミュレーションを行ないます。
PRI :一次群インタフェースでシミュレーションを行ないます。
UI :U 点インタフェースでシミュレーションを行います。

(注) この宣言を省略すると基本インタフェースでのシミュレーションになります。

〈宣言文による宣言例〉

```
INTERFACE   BRI ..... 基本インタフェース
CHANNEL     D   ..... D チャネル・シミュレーション
LAYER       3   ..... レイヤ2 自動モード (レイヤ3 モード)
SIMMODE     NT  ..... NTモード
PFEED       RVS ..... リバース給電
```

2.3 基本、一次群および U点における相違点

基本と一次群のシミュレーション・プログラムでは、以下の相違があります。

(1) 宣言文 INTERFACE

基本I/F	INTERFACE BRI
一次群I/F	INTERFACE PRI
U 点I/F	INTERFACE UI

(2) レイヤ1 の起動関数

基本I/F	PH_ACT()
一次群I/F	−
U 点I/F	PH_ACT()

一次群シミュレーションでは、レイヤ1 は常時起動になっています。従ってレイヤ1 を起動する関数（PH_ACT関数）を実行する必要はありません。

(3) 使用チャネルの指定関数

・D チャネルの指定

基本I/F	−	
一次群I/F	SET_CHANN(24)	(B24チャネルを指定する場合)
U 点I/F	−	

・B チャネルの指定

基本I/F	SET_CHANN(1)	(B1 チャネルを指定する場合)
一次群I/F	SET_CHANN(24)	(B24チャネルを指定する場合)
U 点I/F	SET_CHANN(2)	(B2 チャネルを指定する場合)

基本のD チャネル・シミュレーションでは、チャネルはD チャネル固定ですが、一次群ではB1からB24 の範囲で指定する必要があります。また、B チャネル・シミュレーションでは、必ずチャネル指定が必要です。

(4) レイヤ2 自動モードでTEシミュレーションを行なうときのリンク設定値（初期化）

レイヤ2 リンクを設定する場合LINKON( ) 関数を使用しますが、基本と一次群では設定されている初期値が異なります。

- 基本インタフェース
  TEI 割当手順を起動したあと、網から割り当てられたTEI 値でリンクの設定を行ないます。

- 一次群インタフェース
  TEI 値0 でリンクの設定を行ないます。

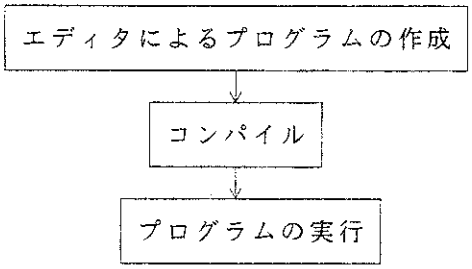
ただし、REG_TEI( ),ACT_TEI( ),REQ_TEI( ),ACT_SAPI( )などの関数を使用することにより、初期値以外のSAPI,TEI値でレイヤ2 リンクの設定をすることができます。また、初期値のSAPI値は0 です。SAPI値を16で使用したい場合は、ACT_SAPI(16)を実行したあと、LINKON( ) 関数を実行します。

3. 簡単なプログラムの作成

3.1 画面に文字表示

プログラムを用いて、画面に文字を表示させるところまでを順を追って説明します。

プログラムは、以下の手順で実行します。



(1) エディタの使用方法

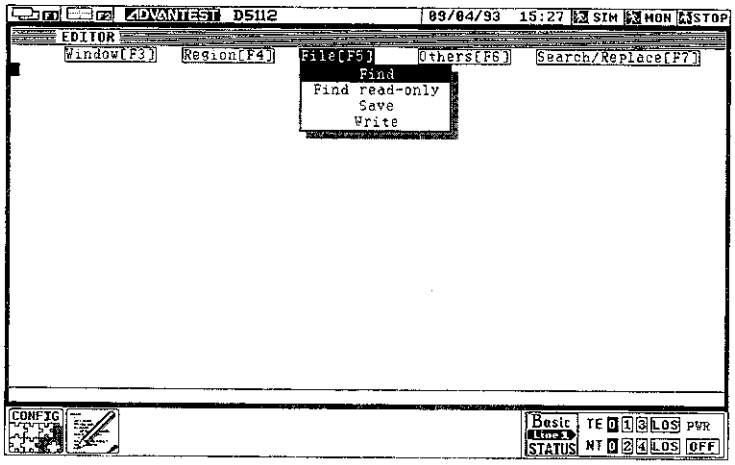


図 3 - 1 エディタの使用方法

① F1キーを押し、ポップアップ・メニューを表示させます。▲▼キーで Editor の位置にカーソルを移動させ、スペース・キーまたはリターン・キーを押して下さい。

② 画面に“ate ”と表示されていることを確認して下さい。

③ F5キーを押し、ポップアップ・メニューを表示させ、スペース・キー（またはリターン・キー）を押して下さい。

④ “Find File”と表示され、ファイル名を聞かれます。“moji.prg”（ 8文字＋ 3文字以下）とファイル名 をタイプして、リターン・キーを押して下さい。

⑤ これにより“moji.prg”のファイルにプログラムを作成できます。以下に示すプログラムをタイプして下さい。

```
FUNC  MAIN ( )
      PRINT ("Welcome to D5112B")
RETURN
```

（命令は大文字で入力して下さい）

(2) コンパイル

① F6キーを押し、ポップアップ・メニューを表示させて下さい。
次に▲▼キーで Execute compile の位置にカーソルを移動させて、スペース・キー（またはリターン・キー）を押して下さい。

② “Compile”のポップアップ画面が表示されます。次にエデットしたファイル名が表示されますので、そのままリターン・キーを押して下さい。

③ “SDO :/PRG/MOJI.PRG 〔y/n 〕？”と作成したファイルをディスクにセーブするか、しないかを聞かれます。“Y ”とタイプし、リターン・キーを押して下さい。

④ 画面が上下に分割され、コンパイル結果が表示されます。
エラーがない場合は、以下のような表示となります。

```
File  name  :  [SDO:/PRG/MOJI.PRG]
Finished
```

⑤ エラー表示が出た場合、▶◀キーで [Window] （ [F 3] キーに対応）の位置にカーソルを移動させてスペース・キーまたはリターン・キーを押して下さい。次に▲▼キーで [Only] の位置にカーソルを移動させ、スペース・キー（またはリターン・キー）を押して下さい。コンパイルが行なわれる前の状態にもどりますので、エラーが発生しているライン番号の内容をエラー・メッセージに従って修正し、再びコンパイルして下さい。

（注）コンパイル実行中に「メモリが足りません。」のエラー・メッセージが表示される場合、[F 2] キーを押して下さい(QUIT の実行）。不要なモジュールをメモリ上から削除して下さい。

(3) プログラムの実行

コンパイルが正常に終了したらプログラムを実行させて下さい。

① [F 1] キーを押し、ポップアップ・メニューを表示させて下さい。
▲▼キーにより [SIMULATOR LAPD] を選択してリターン・キーまたはスペース・キーを押して下さい。

② カーソルを [ ] の位置に移動させ、スペース・キーまたはリターン・キーを押して下さい。

ポップアップ・メニューが表示されますので、▲▼キーにより [SIMULATOR] を選択し、スペース・キーまたはリターン・キーを押して下さい。

③ [F 8] キーを押すとプログラムが実行されます。画面に以下のように表示されることを確認して下さい。

```
Welcome to D5112B
```

[F 10] キーは、プログラムの実行を停止させるためのキーです。
上記のプログラムの場合自動的に停止しますので、[F 10] キーを押す必要はありません。

4. プログラムを用いたシミュレーション

プログラムを用いて、シミュレーションを行なう場合、以下に示すフローに従い実行する必要があります。

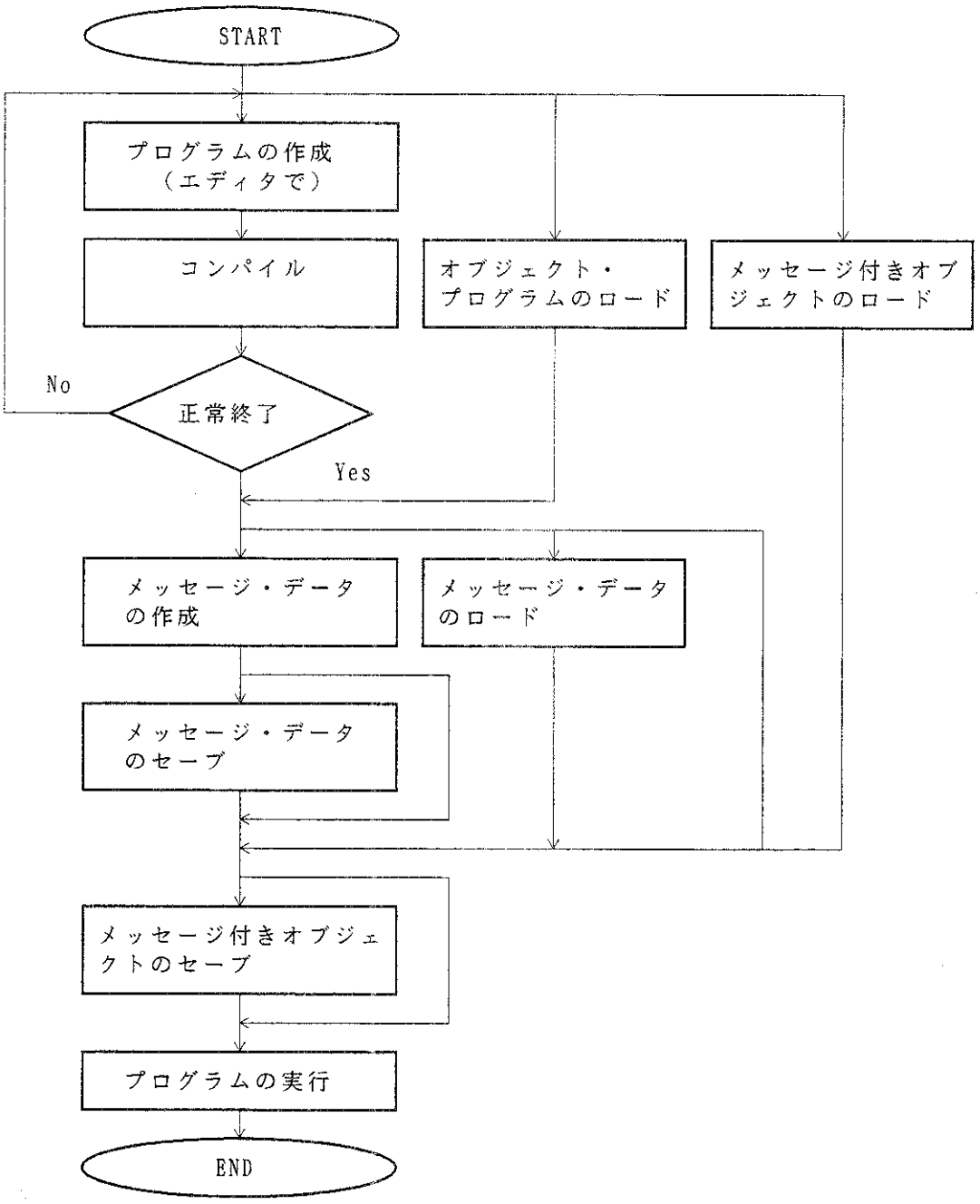


図 4 - 1　シミュレーションの実行手順

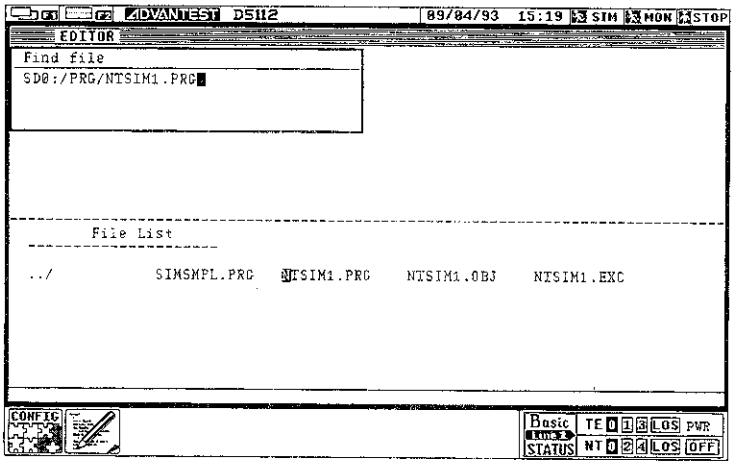


図 4 - 2 プログラムの作成

(1) プログラムの作成

プログラムの作成は、エディタを用いて行ないます。

① F1キーを押し、ポップアップ・メニューを表示させます。▲、▼キーで EDITOR の位置にカーソルを移動し、スペース・キー(またはリターン・キー) を押します。

② F5キーを押し、ポップアップ・メニューを表示させます。▲、▼キーで Find の位置にカーソルを移動し、スペース・キー(またはリターン・キー) を押します。
(C-x C-f でも同じことができます。)
C-x (CTRLキーと Xキーを同時に押します。)
C-f (CTRLキーと Fキーを同時に押します。)

③ ここでファイル名を入力し、リターン・キーを押すと入力された名前のファイルが開きます。すでに入力されているファイルの場合には、▲、▼キーで開きたいファイルの位置にカーソルを移動させ、リターン・キーを押すことにより指定できます。ここで再びリターン・キーを押すと指定したファイルが開きます。

④ プログラムを作成します。

(2) プログラムのコンパイル

プログラムの作成が終了したら、コンパイルを行ないます。

① [F6]キーを押し、ポップアップ・メニューを表示させます。次に[▲]、[▼]キーで Execute Compile の位置にカーソルを移動し、スペース・キー（またはリターン・キー）を押します。（C-x C-e でも同じことができます。）
C-x（[CTRL]キーと[X]キーを同時に押します。）
C-e（[CTRL]キーと[E]キーを同時に押します。）

② ここで、コンパイルを実行するファイル名をタイプします。先にエディタにより、ファイルを編集しているとそのファイル名が表示されます。そのファイル名でよければ、リターン・キーを押します。別なファイルをコンパイルしたい場合、希望のファイル名に直し、リターン・キーを押します。

③ コンパイルするファイルが、エディタにより修正されただけでセーブしていない場合、セーブするかしないかを聞かれます。もし、セーブして良ければ“Y”とタイプして、リターン・キーを押して下さい。（ここで、コンパイルを希望するファイルがセーブされます。）もし、ここで“N”とタイプしてリターン・キーを押すとエディタで修正する前のファイルをコンパイルします。

④ 画面がスプリットされ、コンパイルの結果が表示されます。エラーが出ていないことを確認して下さい。エラーが発生している場合、もう一度エディタにより修正してコンパイルを実行して下さい。
エディタを再び使用したい場合は、[F3]キーによりポップアップ・メニューを出し、Window 項目中の Only を選択し、スペース・キーを押して下さい。元の状態にもどります。（C-x 1 でも同じことができます。）
C-x（[CTRL]キーと[X]キーを同時に押します。）

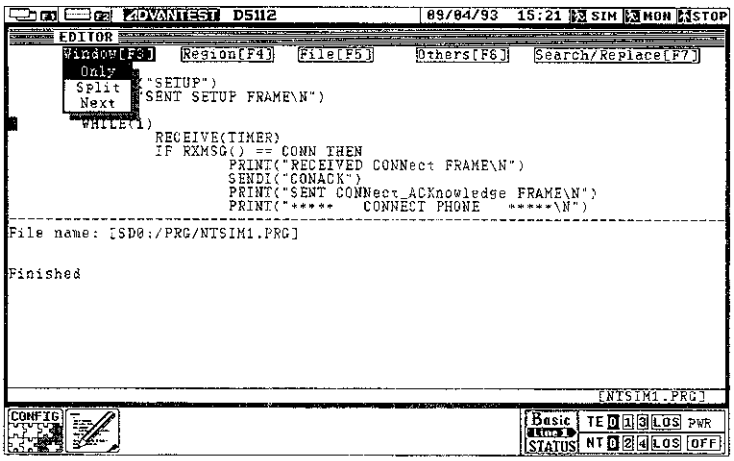


図 4 - 3 プログラムの作成

5. コンパイラによるエラー・コード

PSL51 をコンパイルすると、エラーに対し以下のエラー・コードがディスプレイ上に表示されます。

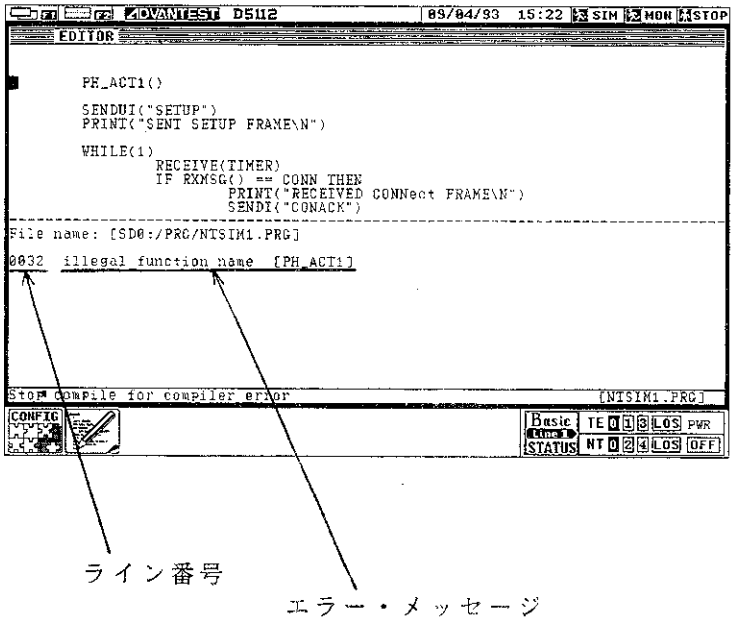


図 5 - 1 エラー・コード

5.1 エラー・メッセージとエラー概要

表 5-1 エラー・メッセージ(1/4)

エラー・メッセージ	概要
missing (,) or illegal character for operator	カッコが、正常なバランスでない。また作用素として不適切な文字〔 〕がある。
illegal use for RESERVED word	予約語〔 〕を名前として使用している。
illegal operator	作用素が連続している。被作用素があるべき位置に作用素〔 〕がある。
illegal function name or label	関数名、名札等を変数名〔 〕として使用している。
PRINT format error	PRINT 文の表記法が不適切である。
SYNTAX error	構文エラー
invalid value	値が不適切である。
illegal function name	関数名として不適切な名前〔 〕を関数呼出しの形で使用している。
redeclared function name	関数名〔 〕は概に宣言されている。
the left side is illegal	代入の左辺が正しくない。
ARRAY needs suffix	配列名に添字がついていない。
illegal ARRAY use	配列名に対して、演算が行なわれている。添字として配列名を使用している。IF文やWHILE 文の条件部分に、配列名だけの式が、使用されている。
illegal ARRAY suffix	配列の添字が正しくない。“)”があるべき位置に〔 〕がある。

(注) 文中の〔 〕の内の文字はディスプレイ上に表示されます。

表 5 - 1 エラー・メッセージ(2/4)

エラー・メッセージ	概要
illegal ARRAY size	配列の大きさの指定が不適切である。
illegal separator	配列の初期値の並びの区切りに記号〔 〕が使用されている。
illegal ARRAY name	名前〔 〕は配列名として不適切である。
needs more initial values	配列の初期値が少なすぎる。
too many initial values	配列の初期値が多すぎる。
illegal arguments	引数の形が正しくない。")"がない。","や")"の代わりに文字〔 〕が使用されている。
illegal separator	引数の区切りに記号〔 〕が使用されている。
illegal argument	名前〔 〕は、引数として正しくない。
illegal function name	名前〔 〕は、関数名として不適切である。
illegal argument number	関数〔 〕の引数の個数が不適切である。
undefined function	関数〔 〕が、未定義である。
SYNTAX error for IF	IFの次の式は、形が不適切である。 THENの位置に〔 〕がある。 IF文の中に構文エラーがある。 END またはELSEの位置に〔 〕がある。
illegal control variable for FOR	FOR 文の制御変数が不適切である。
SYNTAX error for FOR	FOR 文中で、＝の位置が不適切である。＝がない。

(注) 文中の〔 〕の内の文字はディスプレイ上に表示されます。

表 5 - 1 エラー・メッセージ(3/4)

エラー・メッセージ	概要
SYNTAX error for FOR	FOR 文中で、増文の式が不適切である。TOがない。 FOR 文中で、終値の式が不適切である。DOがない。 FOR 文中で、文の並びが不適切である。
SYNTAX error for WHILE	WHILE の次の式の形が不適切である。 WHILE 文中で、文の並びが不適切である。
illegal label	WHILE 文でないもの、〔 〕に名札がついている。 名札〔 〕が正しく書かれていない。 〔 〕は名札としては不適切である。
redefined label	名札〔 〕が2回以上定義されている。
undefined label	名札〔 〕が未定義である。
needs label after EXIT	EXITの ”(” のあとに名札でない〔 〕がある。
illegal label	EXITの名札〔 〕が正しく書かれていない。
illegal use for CASE	CASE文で、条件〔’ ’〕が不適切である。
SYNTAX error for CASE	CASE文の条件式が不適切である。OFがない。 CASE文で条件が ”’ ” で始まっていない。
illegal RETURN format	RETURN文で、式が不適切である。”) ” がない。
MAIN not found	主プログラム関数MAIN( ) がない。

（注）文中の〔 〕の内の文字はディスプレイ上に表示されます。

表 5 - 1　エラー・メッセージ(4/4)

エラー・メッセージ	概要
variable & array memory overflow	変数および配列が許容容量を超えてしまった。
program memory overflow	プログラム容量がメモリ容量を超えてしまった。
temporary memory overflow	テンポラリー容量が、メモリ容量を超えてしまった。
print statement area overflow	PRINT 文の容量が、メモリ容量を超えてしまった。
parameter not found	引数が見つからない。
missing return	RETURN文が存在しない。

（注）文中の〔 〕の内の文字はディスプレイ上に表示されます。

6.　プログラムの実行

プログラムを実行するためには、オブジェクト・プログラムをディスクからロードしてくるか、ソースプログラムをコンパイルしておく必要があります。
コンパイルしたオブジェクトがD チャネル上で走るかB チャネル上で走るかは、宣言文 CHANNEL でD を指定したかB を指定したかによって決まります。
以下にオブジェクト・プログラムを実行する場合について説明します。

① シミュレーション・アプリケーションを選択します。
D チャネルシミュレーションの場合は“SIMULATOR LAPD”を、B チャネルシミュレーションの場合は“SIMULATOR LAPB”を選択します。選択は、F1キーを押して現れるポップアップ・メニューで行なうか、すでにアプリケーションをロードしてある場合は、ALTキー＋ファンクション・キーで行ないます。

② カーソルが  の位置でスペース・キーを押すと、ポップアップ・メニューが表示されます。▲、▼キーで“SIMULATOR ”を選択し、スペース・キー（またはリターン・キー）を押します。

③ この画面でF8キーを押すと、プログラムが走り始めます。F10キーを押すとプログラムは、停止します。
プログラムでメッセージ・ビルダによるメッセージを必要としている関数(SENDF, SENDI, SENDUI ……）が使用されている場合、メッセージ・ビルダによりメッセージを作成しなければ正常に動作しません。

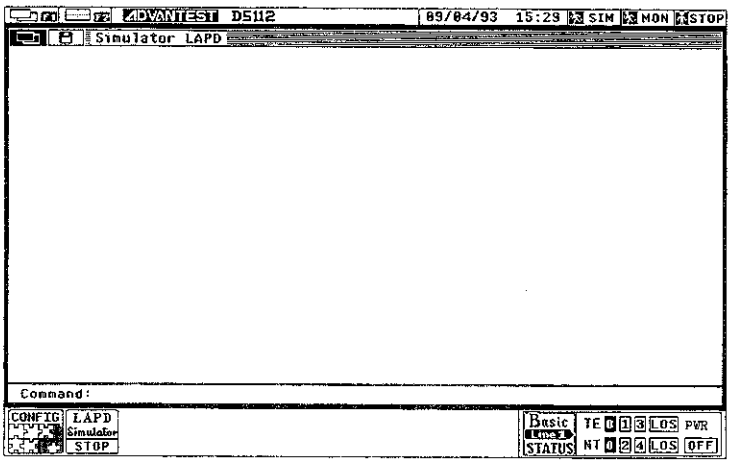


図 6 - 1　シミュレーション画面

7. メッセージ・ビルダ

7.1 メッセージ・ビルダの概略

送信するフレームのデータはメッセージ・ビルダにより作成します。
以下に、シミュレーション用メッセージの作成について説明します。

① シミュレーション・アプリケーションを選択します。
D チャネル・シミュレーションの場合は“SIMULATOR LAPD”を、B チャネル・シミュレーションの場合は“SIMULATOR LAPB”を選択します。選択は、F1キーを押して現れるポップアップ・メニューで行ないます。すでにアプリケーションをロードしてある場合は、Alt キー＋ファンクション・キーで行ないます。

② カーソルが  の位置でスペース・キーを押すと、ポップアップ・メニューが表示されます。▲、▼キーで“MESSAGE ”を選択し、スペース・キー（またはリターン・キー）を押します。

③ メッセージは、No.1～No.100までの100 種類を作成できます。
メッセージ・ビルダを使用する際に注意しなければならないことがあります。まずメッセージ・ビルダで作成した、LAYER2の情報は、レイヤ2 自動モードでは無視されます。また、トランスペアレント・モードで使用される N(S) 、N(R)はプログラムで設定されたV(S)、V(R)値を参照し、自動設定されます。 P／F ビット値は、引数により指定した値が有効です。

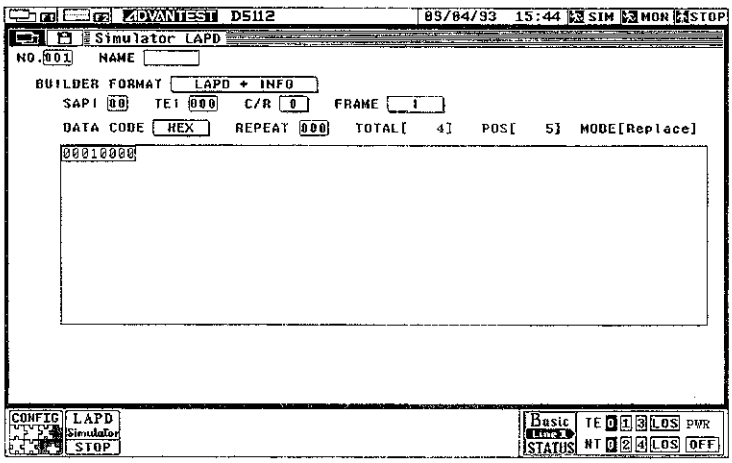
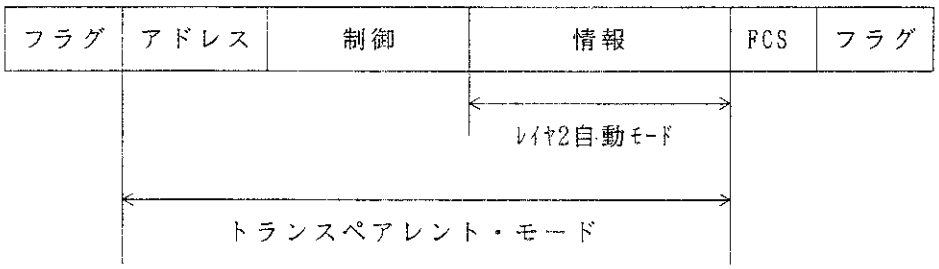


図 7 - 1 メッセージの作成（メッセージ・ビルダ）

7.2 LAPD用メッセージ・ビルダ

メッセージ・ビルダにより下図に示す部分のフレームを作成できます。



メッセージ・ビルダにより、NO.1～NO.100の100 本のフレームを作成できます。
作成したフレームには名前をつける必要があります。名前を付けることにより作成したフレームが有効になります。(6文字まで入力可能)
アドレスと制御フィールドは、SAPI、TEI 、 C／R ビット、フレーム・タイプが設定可能です。N(R)、N(S)値は、メッセージ・ビルダでは設定できません。N(R)、N(S)値は、シミュレーション関数 (INCVS, INCVR, SETVS, SETVR) により設定されたV(S)、V(R)値を参照して自動的に付加されます。また、P/F ビットの設定もできません。これはSENDF関数実行時に引数で与えられる必要があります。
メッセージ作成には、 8通りの方法があります。
作成フォーマットの切り換えは、"BUILDER FORMAT"のポップアップ・メニューで行ないます。"BUILDER FORMAT"の位置でスペース・キーまたは、リターン・キーを押すと以下の 8種類のフォーマットが選択できるポップアップ・メニューが現れます。

(1) LAYER2 INFO : レイヤ2 レベルからの直接入力
(2) LAPD + INFO : レイヤ2 はLAPDメニューによる入力でレイヤ3 は直接入力
(3) LAPD + Q.931 : レイヤ2 はLAPDメニューによる入力でレイヤ3 は Q.931メニューによる入力
(4) LAPD + X.25 : レイヤ2 はLAPDメニューによる入力でレイヤ3 は X.25 メニューによる入力
(5) LAYER3 INFO : レイヤ3 は直接入力
(6) Q.931 : レイヤ3 はQ.931 メニューによる入力
(7) X.25 : レイヤ3 はX.25メニューによる入力
(8) FROM MONITOR : モニタ・モジュールのデータをコピーして使用

希望するフォーマットの位置に▲、▼キーでカーソルを移動させます。リターン・キーを押すことにより、作成フォーマットを変更することができます。
レイヤ2 自動モードでシミュレーションを行なう場合、メッセージのレイヤ2 部分は使われません。そこで、この部分を画面から消去したものが、(5)～(7)のフォーマットです。内部的には、(3)の"LAPD + Q.931"と(6)の"Q.931" のフォーマットは、同じ動作をするので、レイヤ2 自動モードで使用するメッセージを作成するときは、どちらのフォーマットを使用しても構いません。(4)の"LAPD + X.25" と(7)の"X.25"についても同様のことが言えます。

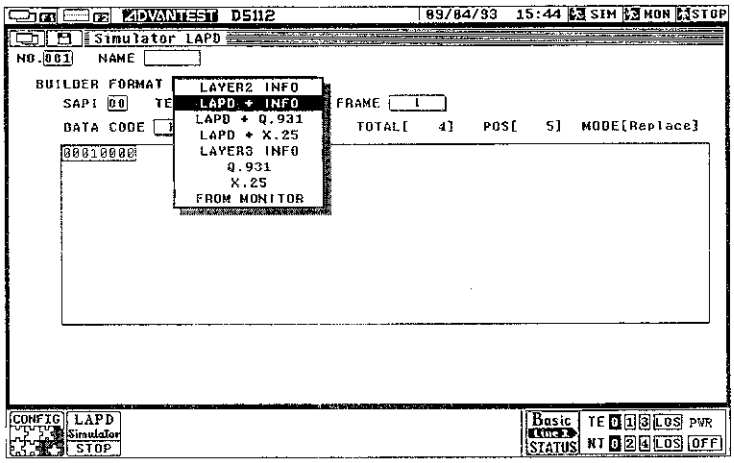


図 7 - 2 Dチャネル用メッセージ・ビルダのフォーマット選択

(1) LAYER2 INFO フォーマット

レイヤ2 レベルからデータを直接入力することによって、メッセージを作成するときに用いるフォーマットです。

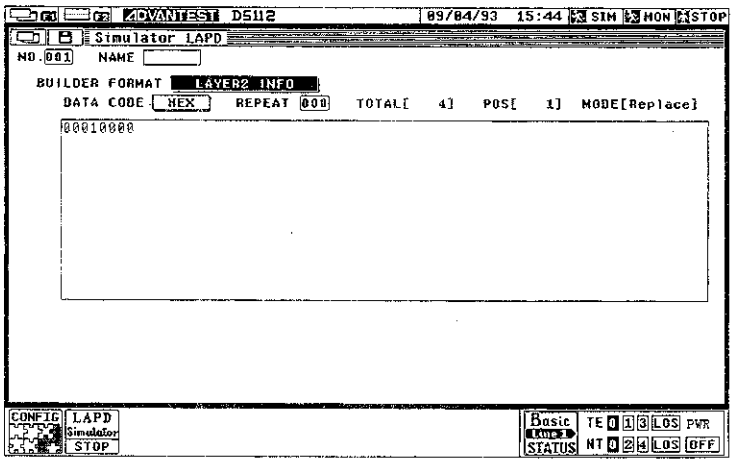


図 7 - 3 LAYER2 INFO フォーマット

(2) LAPD + INFO フォーマット

レイヤ2 はLAPDメニューによりメッセージを作成し、レイヤ3 は直接入力により作成します。

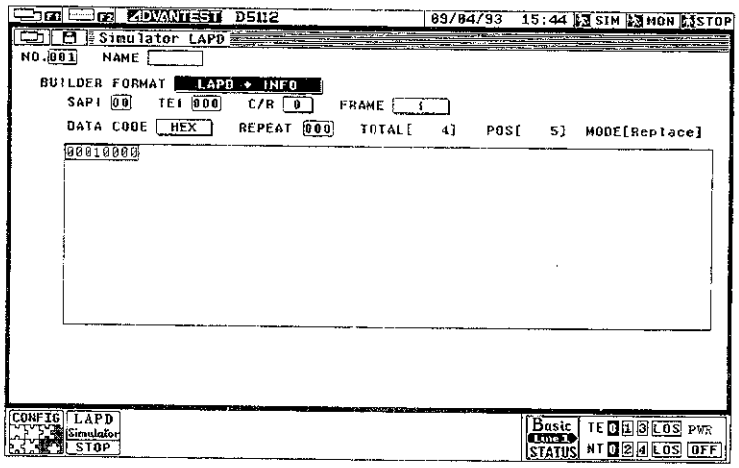


図 7 - 4 LAPD + INFO フォーマット

(3) LAPD + Q.931フォーマット

レイヤ2 はLAPDメニューによりメッセージを作成し、レイヤ3 はQ.931 メニューにより作成します。

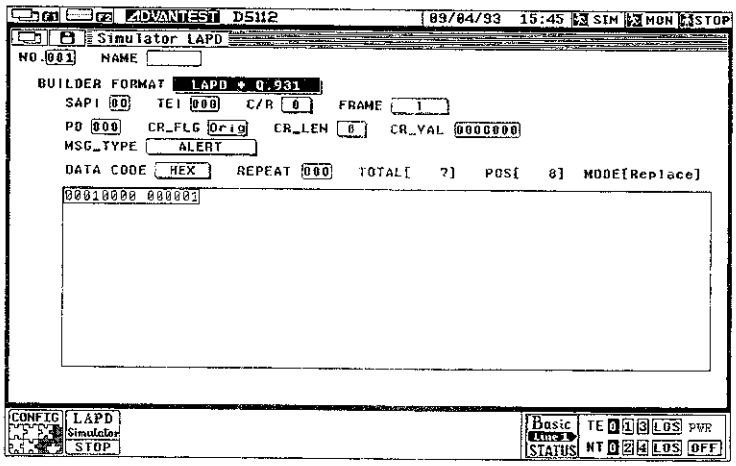


図 7 - 5 LAPD + Q.931フォーマット

(4) LAPD + X.25 フォーマット

レイヤ2 はLAPDメニューによりメッセージを作成し、レイヤ3 はX.25により作成します。

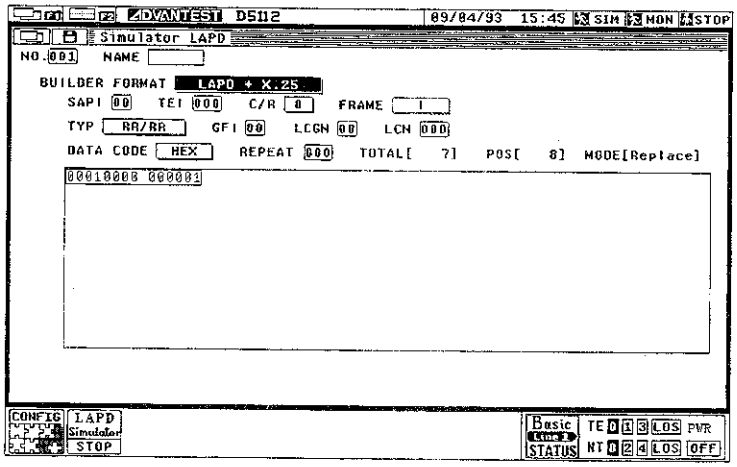


図 7 - 6 LAPD + X.25 フォーマット

(5) LAYER3 INFO フォーマット

レイヤ3 は直接入力によりメッセージを作成します。

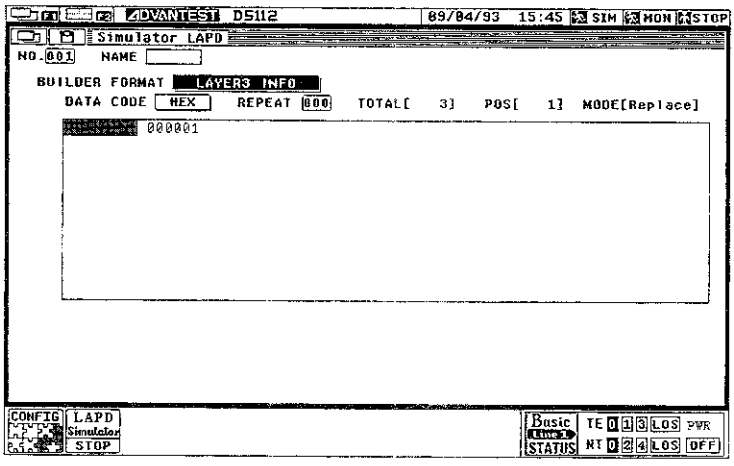


図 7 - 7 LAYER3 INFO フォーマット

(6) Q.931 フォーマット

レイヤ3 はQ.931 メニューによりメッセージを作成します。

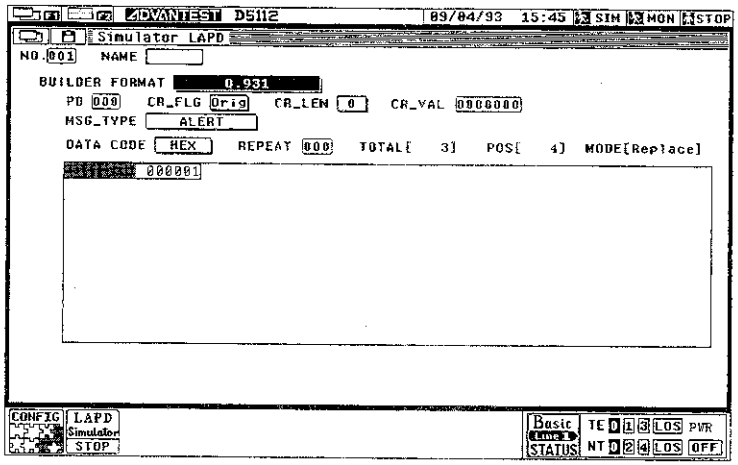


図 7 - 8 Q.931 フォーマット

(7) X.25フォーマット

レイヤ3 はX.25メニューによりメッセージを作成します。

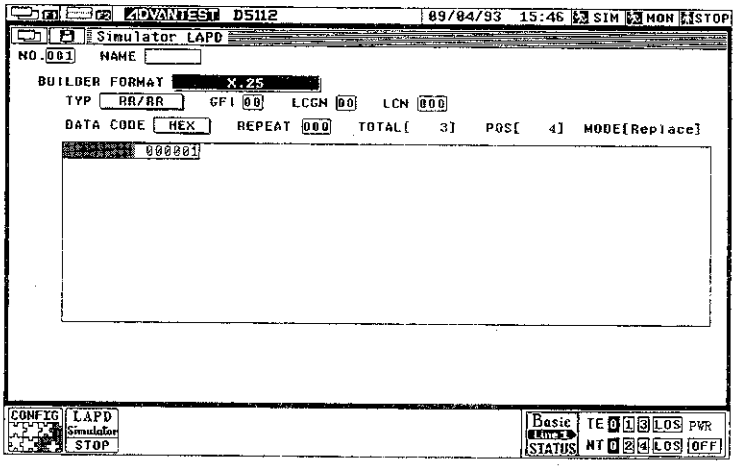


図 7 - 9 X.25フォーマット

(8) FROM MONITORフォーマット

レイヤ2 およびレイヤ3 のメッセージ内容をモニタ・データからコピーします。
[図7-10] では、画面上部にモニタ・モジュール(HDLC MONITOR 2)を表示しています。データを使用するモニタ・モジュールは、MONITOR PORTにカーソルを移動し、スペース・キー（または、リターン・キー）を押して、ポップアップ・メニューを表示させて選択します。（この例では、HDLC MONITOR 2を選択しています。）
スペース・キー（または、リターン・キー）を押して決定します。

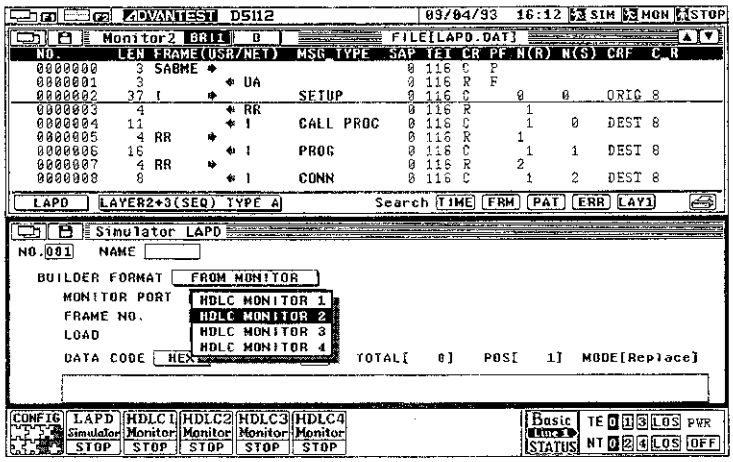


図 7 - 10　モニタ・モジュールの指定

データの指定は、FRAME NO. に数値を入力します。（この例では、カーソル位置のデータの 000002 を指定）カーソルをLOAD領域(Execute) に移動し、スペース・キー（または、リターン・キー）を押すと、データが読み込まれて表示されます。

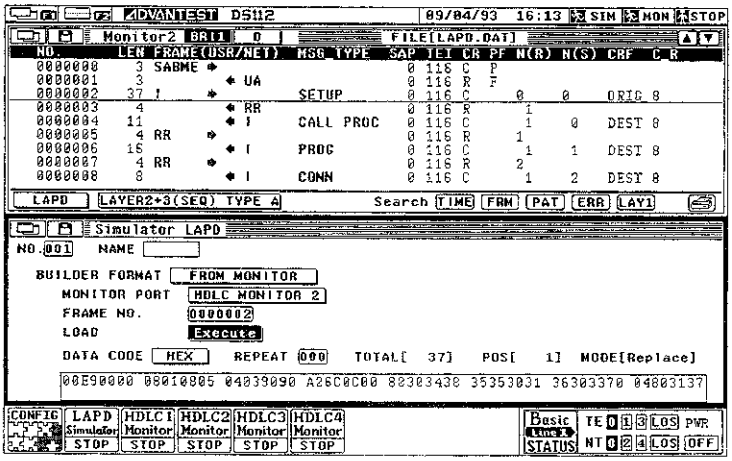
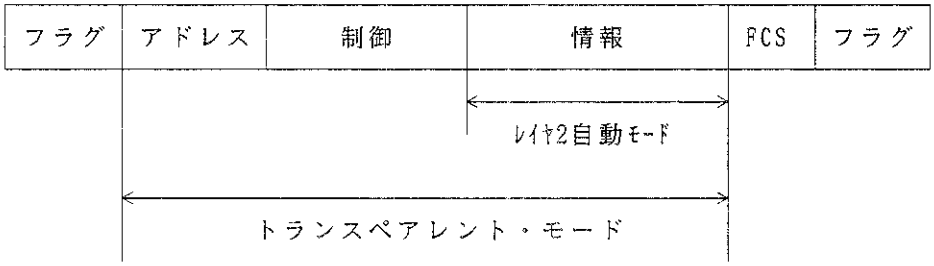


図 7 - 11　データ番号の指定とロード

7.3 LAPB用メッセージ・ビルダ

メッセージ・ビルダにより下図に示す部分のフレームを作成できます。



メッセージ・ビルダにより、NO.1～NO.100の100 本のフレームを作成できます。
作成したフレームには名前をつける必要があります。名前を付けることにより作成したフレームが有効になります。(6文字まで入力可能)
アドレスと制御フィールド内のフレーム・タイプが設定可能です。N(R)、N(S)値は、メッセージ・ビルダでは設定できません。N(R)、N(S)値は、シミュレーション関数(INCVS, INCVR, SETVS, SETVR) により設定されたV(S)、V(S)値を参照してSENDF 関数使用時に自動的に付加されます。また、P/F ビットの設定もできません。これは SENDF関数の引数で与えられます。
メッセージ作成には、 8通りの方法があります。BUILDER FORMATの位置でスペース・キーまたは、リターン・キーを押すと以下の 8種類のフォーマットが選択できるポップ・メニューが現れます。 8種類のフォーマットは以下の通りです。

(1) LAYER2 INFO : レイヤ2 レベルからの直接入力
(2) LAPB8 + INFO : レイヤ2 はLAPB8 メニューによる入力でレイヤ3 は直接入力
(3) LAPB8 + X.25 : レイヤ2 はLAPB8 メニューによる入力でレイヤ3 はX.25メニューによる入力
(4) LAPB128 + INFO : レイヤ2 はLAPB128 メニューによる入力でレイヤ3 は直接入力
(5) LAPB128 + X.25 : レイヤ2 はLAPB128 メニューによる入力でレイヤ3 はX.25による入力
(6) LAYER3 INFO : レイヤ3 は直接入力
(7) X.25 : レイヤ3 はX.25メニューによる入力
(8) FROM MONITOR : モニタ・モジュールからデータをコピーして使用

希望するフォーマットの位置に▲、▼キーでカーソルを移動させます。リターン・キーを押すことにより、作成フォーマットを変更することができます。
レイヤ2 自動モードでシミュレーションを行なう場合、メッセージのレイヤ2 部分は使われません。そこで、この部分を画面から消去したものが、(6)、(7)のフォーマットです。(3)の"LAPB8 + X.25"や(5)の"LAPB128 + X.25"のフォーマットで作成したメッセージと(7)の"X.25"フォーマットで作成したメッセージはレイヤ2 自動モードで使用する場合は同じです。レイヤ2 自動モードの時のレイヤ2 モジュロの選択は、宣言文で行ないます。

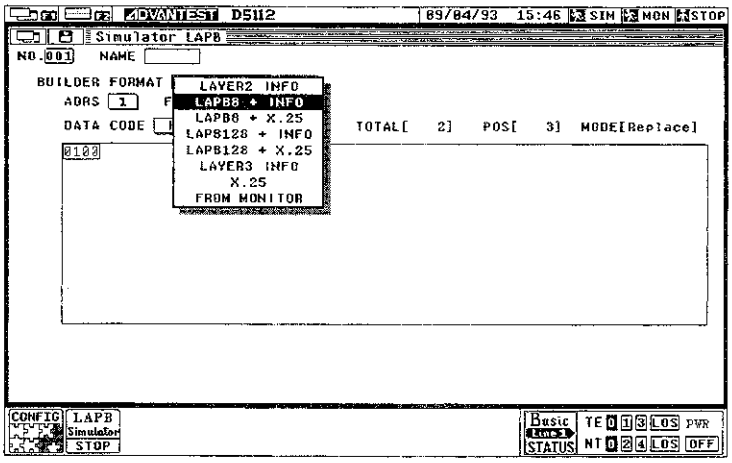


図 7 - 12 Bチャネル用メッセージ・ビルダのフォーマット選択

(1) LAYER2 INFO フォーマット

レイヤ2 からデータを直接入力することにより、メッセージを作成するフォーマットです。

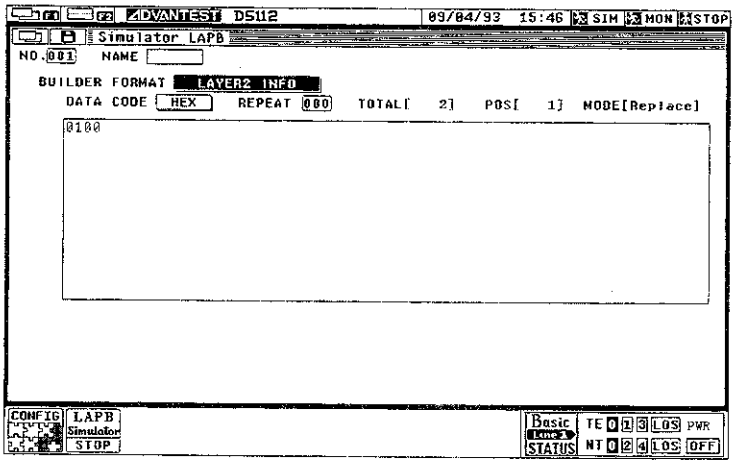


図 7 - 13 LAYER2 INFOフォーマット

(2) LAPB8 + INFOフォーマット

レイヤ2 はLAPBモジュロ8 メニューを使用してメッセージを作成し、レイヤ3 は直接入力することにより作成します。

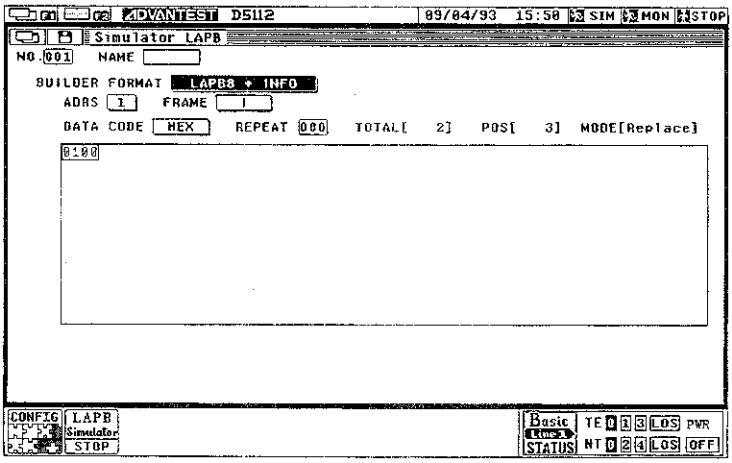


図 7 - 14 LAPB8 + INFO フォーマット

(3) LAPB8 + X.25フォーマット

レイヤ2 はLAPBのモジュロ8 メニューを使用してメッセージを作成し、レイヤ3 はX.25のメニューにより作成します。

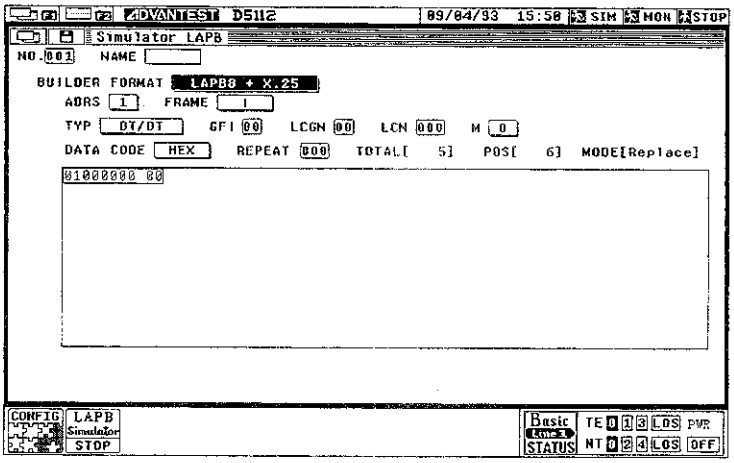


図 7 - 15 LAPB8 + X.25 フォーマット

(4) LAPB128 + INFOフォーマット

レイヤ2 はLAPBのモジュロ128 メニューを使用してメッセージを作成し、レイヤ3 は直接入力することにより作成します。

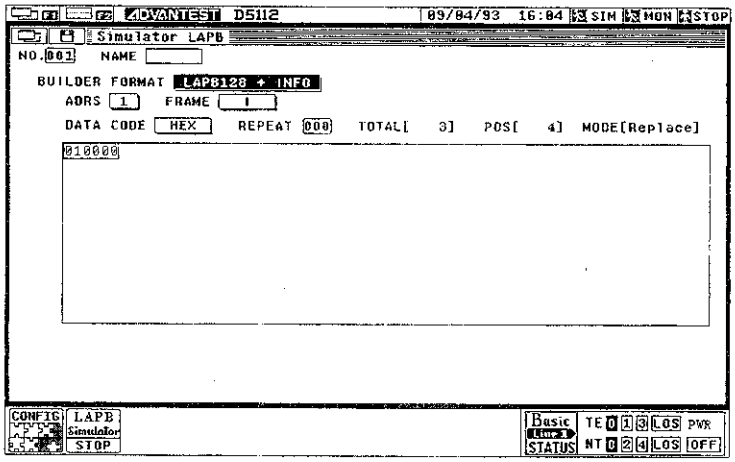


図 7 - 16 LAPB128 + INFO フォーマット

(5) LAPB128 + X.25フォーマット

レイヤ2 はLAPBモジュロ128 のメニューを使用してメッセージを作成し、レイヤ3 はX.25のメニューにより作成します。

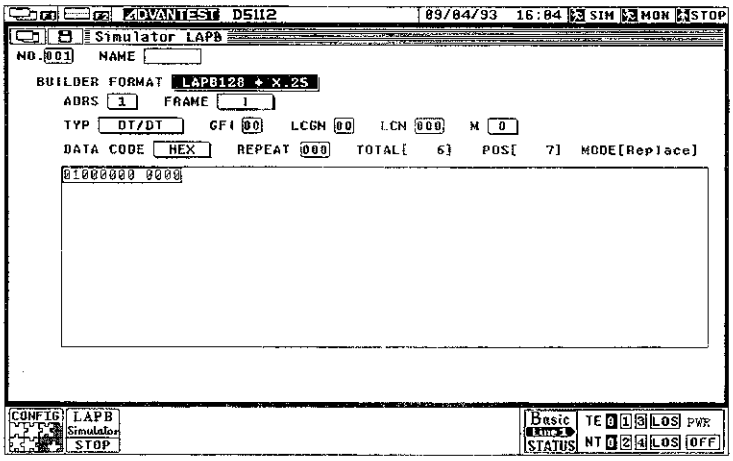


図 7 - 17 LAPB128 + X.25 フォーマット

(6) LAYER3 INFO フォーマット

レイヤ3 は直接入力によりメッセージを作成します。

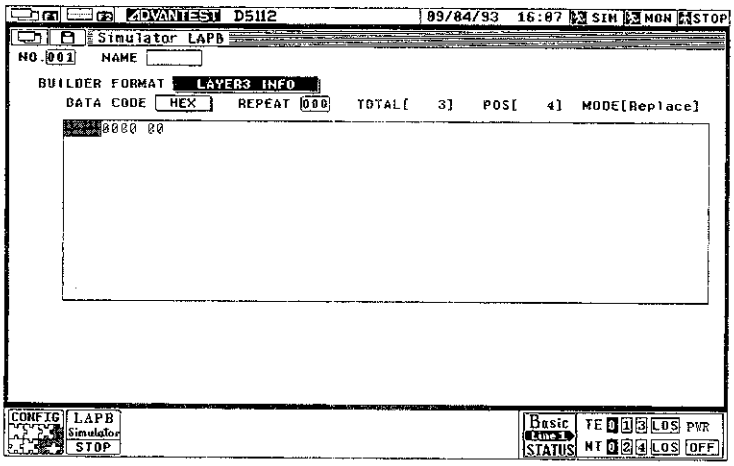


図 7 - 18 LAYER3 INFOフォーマット

(7) X.25フォーマット

レイヤ3 はX.25によりメッセージを作成します。

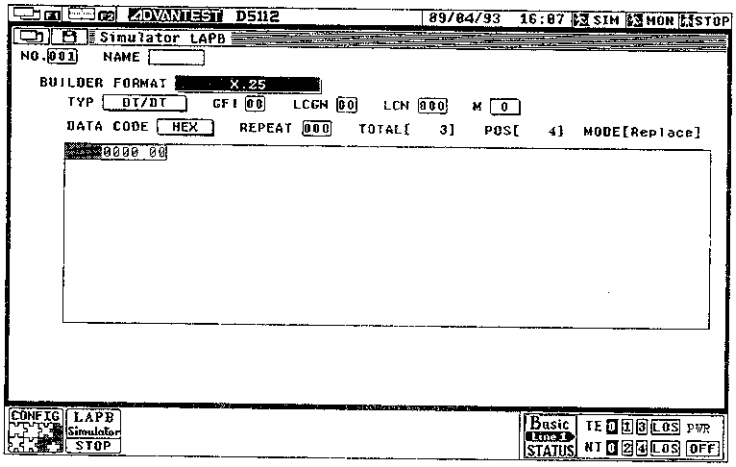


図 7 - 19 X.25 フォーマット

(8) FROM MONITORフォーマット

レイヤ2 およびレイヤ3 のメッセージ内容をモニタ・データからコピーします。[図7-20]では、画面上部にモニタ・モジュール(HDLC MONITOR 3)を表示しています。データを使用するモニタ・モジュールは、MONITOR PORTにカーソルを移動し、スペース・キー（または、リターン・キー）を押して、ポップアップ・メニューを表示させて選択します。（この例では、HDLC MONITOR 3を選択しています。）
スペース・キー（または、リターン・キー）を押して決定します。

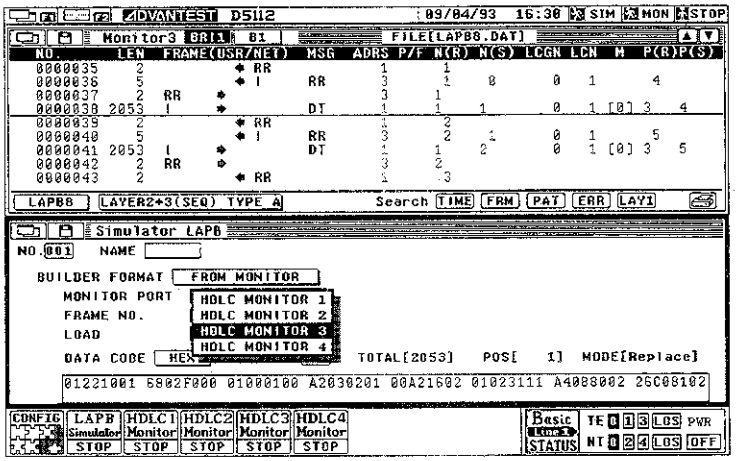


図 7 - 20　モニタ・モジュールの指定

データの指定は、FRAME NO. に数値を入力します。（この例では、カーソル位置のデータの 000038 を指定）カーソルをLOAD領域(Execute) に移動し、スペース・キー（または、リターン・キー）を押すと、データが読み込まれて表示されます。

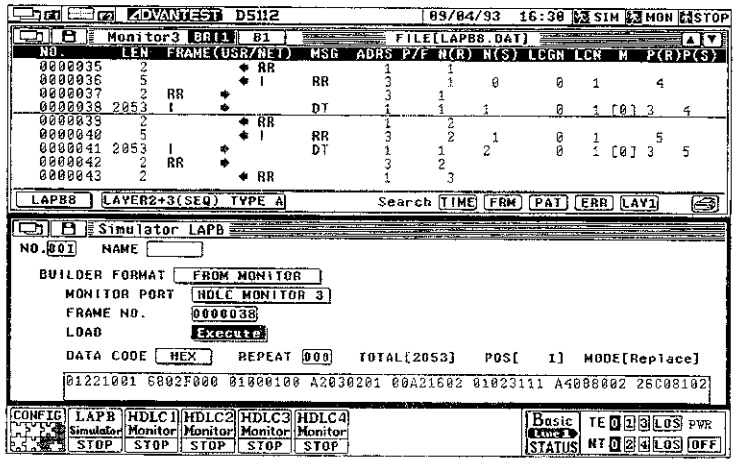


図 7 - 21　データ番号の指定とロード

7.4 メッセージ・ビルダの操作方法

(1) 直接入力部分

メッセージ・データをメニューを使用しないで、直接入力する場合について説明します。

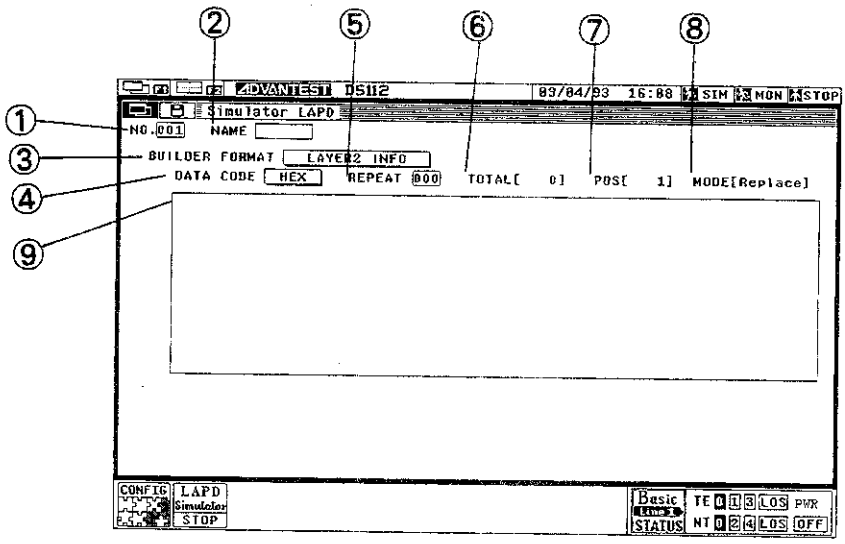


図 7 - 22 データの直接入力

上記の画面についての詳細を以下に示します。

① NO.

フレーム番号を表示します。 1番～100 番号までのメッセージを作成できます。この設定は直接、希望の数字を入力するか、⇧ ⇩のキーで設定した数字の変更をします。

② NAME

フレーム名を入力します。最大 6文字まで入力可能です。

③ BUILDER FORMAT

メッセージ・ビルダの作成フォーマットを指定します。
カーソルを "BUILDER FORMAT" の位置に移動させて、スペース・キーまたはリターン・キーを押します。以下のポップアップ・メニューが出ますので、▲、▼キーで選択し、スペース・キーまたはリターン・キーで決定します。

i) D チャネルの場合

LAYER2 INFO
LAPD + INFO
LAPD + Q.931
LAPD + X.25
LAYER3 INFO
Q.931
X.25
FROM MONITOR

ii) Bチャネルの場合

LAYER2 INFO
LAPB8 + INFO
LAPB8 + X.25
LAPB128 + INFO
LAPB128 + X.25
LAYER3 INFO
X.25
FROM MONITOR

③の"BUILDER FORMAT"を切り換えると、データ直接入力領域の中に下図の様な枠が表示されることがあります。これはメニューにより作成する領域であることを示しています。⑨領域では、この枠の中の数値を変えることはできません。この枠内のメッセージはメニューの値を変えると自動的に作り直されます。

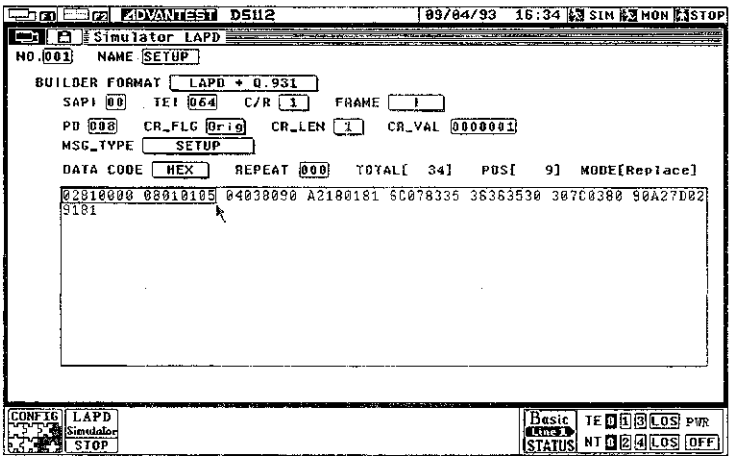


図 7 - 23 データ直接入力領域の枠

また、"BUILDER FORMAT"で"LAYER3 INFO" や" Q.931" "X.25" を選択したとき、下図の様な網掛けが現れます。これは、レイヤ2 自動モードでフレームを送出するときに使われない領域であることを示しています。

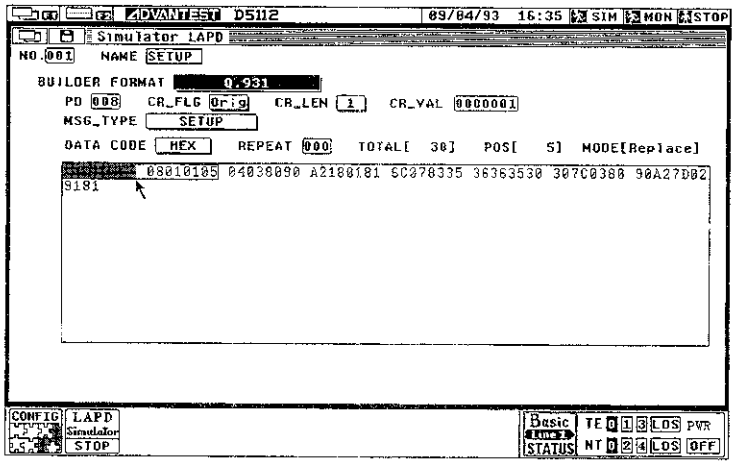


図 7 - 24　レイヤ2 領域の網掛け

④ DATA CODE

⑨の領域に送信データを直接作成しますが、そのときの入力モードを指定します。入力モードは、HEX 、ASCII 、JIS8、EBCDICの4 種類です。入力モードの設定は以下の手順で行ないます。

カーソルを④の位置に移動させます。ここで、スペース・キーを押すと下のポップアップ・メニューが表示されます。

HEX
ASCII
JIS8
EBCDIC

▲、▼キーで、自分の希望する入力モードの位置にカーソルを移動させ、スペース・キーを押します。ポップアップ・メニューが閉じて、④の位置には設定された入力モードが表示されます。

⑤ REPEAT

⑨の領域にリピート回数を設定します。設定されたREPEAT領域をリピート回数だけ繰り返します。0 が設定されるとリピートは行ないません。0 以外が設定された場合は、⑨の領域に設定されたREPEAT領域のフォントが変わります。

⑥ TOTAL

作成したデータからフレームを組み立てた場合の総オクテット数を表示します。ただし、フラグとFCS はカウントされません。また、"BUILDER FORMAT"で"LAYER3 INFO" 、"Q.931" または"X.25"が選択されている場合は、レイヤ2 部分（▒▒部分）もカウントされません。

⑦ POS

カーソルのある位置のオクテットを表示します。
レイヤ2 からメッセージを作成した場合、レイヤ2 の先頭から数えたオクテットを表示します。また、レイヤ3 からメッセージを作成した場合、レイヤ3 の先頭から数えたオクテットを表示します。

⑧ MODE

データの入力モードが何に設定されているか表示します。入力モードの選択は、⑨の領域でC-i （CTRLキーとIキーを同時に押します。）を行なうことによりできます。
〔replace 〕：置換モード
〔insert 〕：挿入モード

⑨ データの直接入力領域

データを直接入力します。入力モードには、HEX 、ASCII 、JIS8、EBCDICの 4種類があります。この設定は④の領域で行なうことができます。また、入力データはリピート領域を設定すれば、その部分のデータがリピート回数だけ繰り返して送信されます。
⑨の領域では、以下のキーを使用することによりデータのエディットができます。

リピート位置のリセット	: CTRL-R
リピート開始位置指定	: CTRL-B
リピート終了位置指定	: CTRL-E
入力モード(replace⇔insert) の切り換え	: CTRL-I
マークのセット	: CTRL-SPC
指定領域の削除	: CTRL-W
カーソルの位置から最後まで削除	: CTRL-K
削除バッファの内容を挿入	: CTRL-Y
データを初期化	: CTRL-C
情報要素識別子の前方検索(Q.931のときのみ有効)	: CTRL-N
情報要素識別子の後方検索(Q.931のときのみ有効)	: CTRL-P

（注）"CTRL-" はコントロール・キーを押しながら"CTRL-" 以降のキーを押すという意味です。
"SPC" はスペース・キーです。

(2) LAPDメニュー

メッセージ・データをLAPDメニューを使用して作成する場合について説明します。

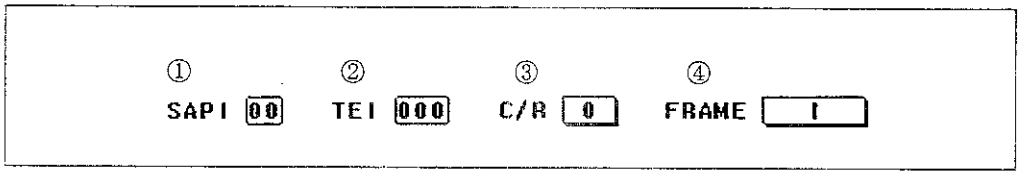


図 7 - 25 LAPD メニュー

① S A P I

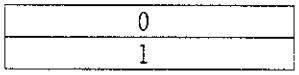
SAPI値を入力します。0 ～ 63 の値が設定可能です。

② T E I

TEI 値を入力します。0 ～ 127の値が設定可能です。

③ C／R

コマンド／レスポンス値を設定します。カーソルをCRの位置に移動させて、スペース・キーまたは、リターン・キーを押します。そして、以下のようなポップアップ・メニューを表示させます。



④ F R A M E

フレームの種類を設定します。カーソルをＦＲＡＭＥの位置に移動させて、スペース・キーまたは、リターン・キーを押します。そして、以下のようなポップアップ・メニューを表示させます。

(3) LAPBメニュー

メッセージ・データをLAPBメニューを使用して作成する場合について説明します。

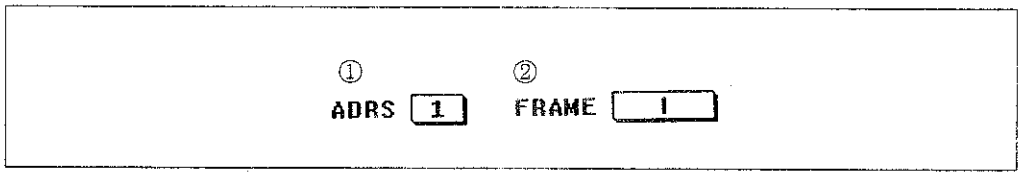


図 7 - 26 LAPB メニュー

① ＡＤＲＳ

アドレスを設定します。
カーソルを "ADRS" の位置に移動させて、スペース・キーまたは、リターン・キーを押すと以下のポップアップ・メニューが表示されます。

1
3

▲、▼キーで希望の値を選択しスペース・キーまたは，リターン・キーで決定します。

② ＦＲＡＭＥ

フレームの種類を設定します。
カーソルを"FRAME" の位置に移動させてスペース・キーまたは、リターン・キーを押すと以下のポップアップ・メニューが表示されます。

i) LAPB8の場合

I	RR	RNR
REJ	SABM	DISC
DM	UA	FRMR

ii) LAPB128 の場合

I	RR	RNR
REJ	SABME	DISC
DM	UA	FRMR

▲、▼、▶、◀キーで希望するフレームを選択しスペース・キーまたは、リターン・キーで決定します。

(4) Q.931 メニュー

メッセージ・データを"Q.931" メニューを使用して作成する場合について説明します。

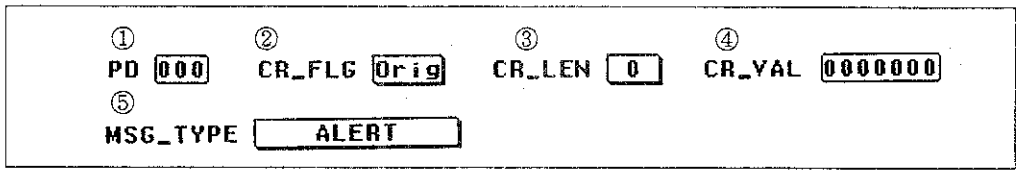


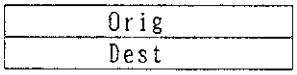
図 7 - 27 Q.931メニュー

① P D

プロトコル識別子を設定します。数字を入力して下さい。"Q.931" プロトコルを使用する場合 "8 " を設定して下さい。

② C R _ F L G

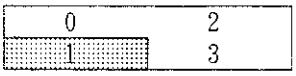
呼番号フラグを設定します。
カーソルをCRF 位置に移動し、スペース・キーまたは、リターン・キーを押すと以下のポップアップ・メニューが表示されます。



▲、▼キーで希望のフラグを選択し、スペース・キーまたは、リターン・キーを押すと、新たに設定されます。

③ C R _ L E N

呼番号長を設定します。
カーソルを"CR_LEN" の位置に移動しスペース・キーまたは、リターン・キーを押すと以下のポップアップ・メニューが表示されます。



▲、▼、▶、◀キーで希望の呼番号長を選択し、スペース・キーまたは、リターン・キーで設定して下さい。

④ C R _ V A L

呼番号を設定します。
設定できるこ番号は③の呼番号長に依存します。
③の呼番号長に"0" を選ぶと、数字が 0以下は入力できません。

⑤ MSG_TYPE

メッセージ・タイプを設定します。カーソルを"MSG_TYPE" の位置に移動し、スペース・キー（またはリターン・キー）を押すとポップアップ・メニューが表示されます。
希望のメッセージの位置にカーソルを移動させ、スペース・キー（またはリターン・キー）を押すとそのメッセージが選択されます。

(5) X.25メニュー

メッセージ・データを"X.25"メニューを使用して作成する場合について説明します。

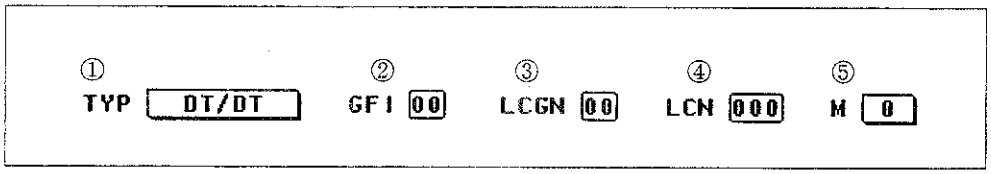


図 7 - 28 X.25 メニュー

① TYP

パケットの種類を設定します。
カーソルを"TYP" の位置に移動し、スペース・キーまたは、リターン・キーを押すとポップアップ・メニューが表示されます。
希望のパケット名の位置にカーソルを移動させ、スペース・キーまたは、リターン・キーを押すとそのパケットが選択されます。

② GFI

ゼネラル・フォーマット識別子を設定します

③ LCGN

論理チャネル・グループ番号を設定します。

④ LCN

論理チャネル番号を設定します。

⑤ M

モア・データ表示値を設定します。
カーソルを "M"の位置にい移動しスペース・キーまたは、リターン・キーを押すと以下のポップアップ・メニューが表示されます。

0
1

▲、▼キーで希望のMビット値を選択し、スペース・キーまたは、リターン・キーを押すと新たに設定されます。
このポップアップ・メニューはパケットの種類が "DT" のときのみ現れます。

8. セーブ／ロード・メニュー

8.1 オブジェクト・プログラムのセーブ／ロード

シミュレーションを実行するためにはソースプログラムをコンパイルするか、ディスクのオブジェクト・プログラムを内部メモリにロードする必要があります。
以下にオブジェクト・プログラムをロードする手順について説明していきます。

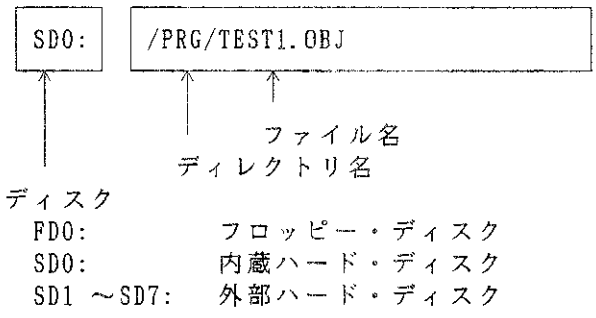
① シミュレーション・アプリケーションを選択します。
Dチャネル・シミュレーションの場合は“SIMULATOR LAPD”を、Bチャネル・シミュレーションは“SIMULATOR LAPB”を選択します。選択はF1キーを押して現れるポップアップ・メニューで行なうか、すでにアプリケーションをロードしてある場合はALTキー＋ファンクション・キーで行ないます。

② カーソルが [💾] の位置で、スペース・キー（またはリターン・キ）を押すとセーブ／ロード画面が表示されます。

③ Data Type の位置にカーソルを移動させ、スペース・キー（またはリターン・キー）を押して下さい。▲、▼キーで、“Object Program”を選択し、スペース・キー（またはリターン・キー）を押して下さい。

④ Operation の位置にカーソルを移動させ、スペース・キー（またはリターン・キー）を押して下さい。“Save”または“Load”のうち希望の位置にカーソルを移動させ、スペース・キーを押して下さい。

⑤ ファイル名を入力する位置にカーソルを移動させ、セーブまたはロードするファイル名をタイプして下さい。ファイル名入力をする位置のフォーマットは以下のようになっています。



⑥ カーソルを [Execute] に移動させ、スペース・キー（またはリターン・キー）を押すと、オブジェクト・プログラムのセーブ／ロードが行なわれます。

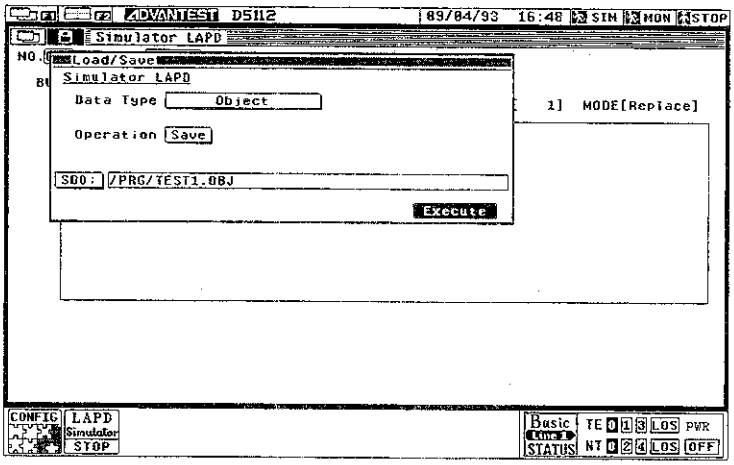


図 8 - 1 オブジェクト・プログラムのセーブ

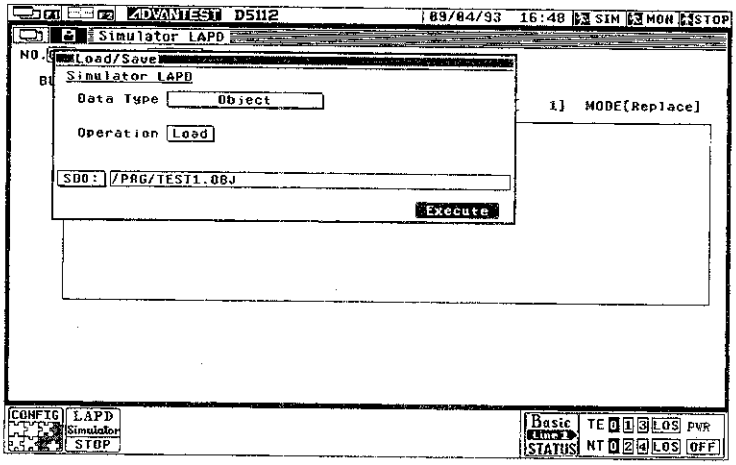


図 8 - 2 オブジェクト・プログラムのロード

8.2 メッセージ・データのセーブ／ロード

メッセージ・データを使用する命令(SENDF、SENDI 、SENDUI...)を含んだプログラムを実行する場合、メッセージ・データが必要です。
メッセージ・データは、メッセージ・ビルダにより、再びデータを作成するか、メッセージ・データをディスクからロードしてくることにより、用意できます。
ここでは、メッセージ・データのセーブ／ロードの手順について以下に示します。

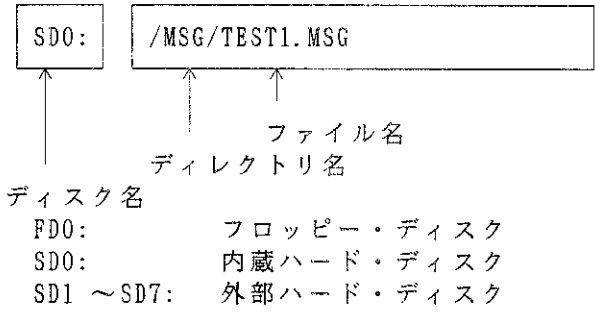
① シミュレーション・アプリケーションを選択します。
D チャネル用メッセージの場合は“SIMULATOR LAPD”を、B チャネル用メッセージは“SIMULATOR LAPB”を選択します。選択はF1キーを押して現れるポップアップ・メニューで行なうか、すでにアプリケーションをロードしてある場合は、ALTキー＋ファンクション・キーで行ないます。

② カーソルが  の位置で、スペース・キー（またはリターン・キー）を押すとセーブ／ロード画面が表示されます。

③ Data Type の位置にカーソルを移動させ、スペース・キー（またはリターン・キー）を押して下さい。▲、▼キーで、“Message Data”を選択して、スペース・キー（またはリターン・キー）を押して下さい。

④ Operation の位置にカーソルを移動させ、スペース・キーを押して下さい。セーブしたい場合は“Save”の位置に、ロードしたい場合は“Load”の位置にカーソルを移動させ、スペース・キーを押して下さい。

⑤ ファイルの位置にカーソルを移動させ、セーブまたはロードするファイル名をタイプして下さい。ファイル名入力をする位置のフォーマットは以下のようになっています。



⑥ カーソルを Execute に移動させ、スペース・キー（またはリターン・キー）を押すとメッセージ・データが、セーブ／ロードされます。

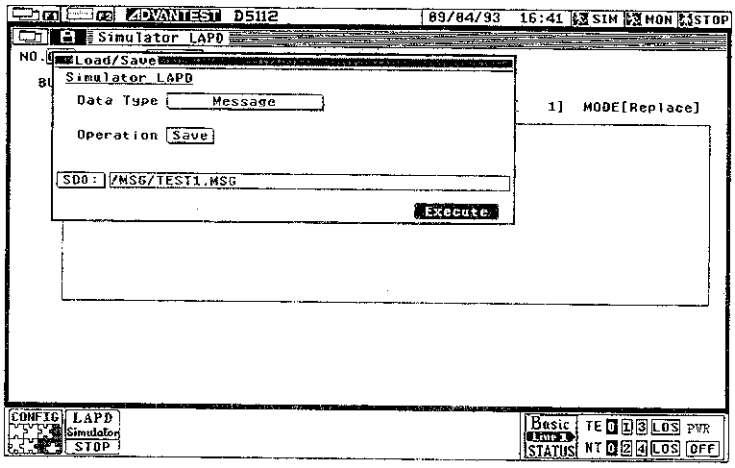


図 8 - 3 メッセージ・データのセーブ

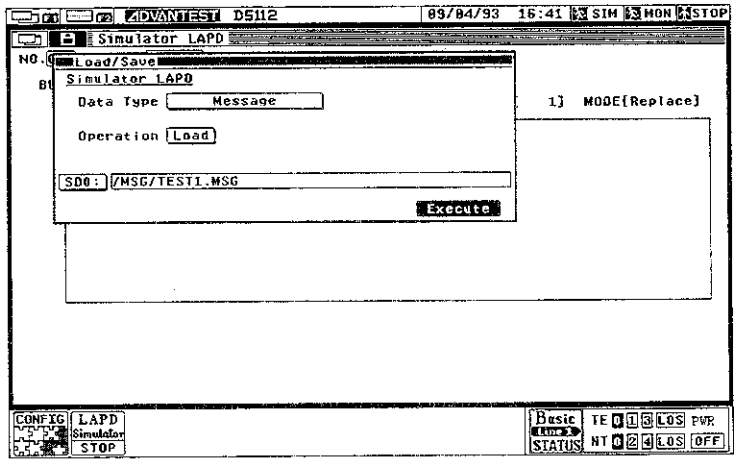


図 8 - 4 メッセージ・データのロード

8.3　メッセージ付きオブジェクト・プログラムのセーブ／ロード

オブジェクト・プログラムとメッセージのセーブ／ロードについて別々に説明してきましたが、これら二つのファイルを一つのファイルとしてセーブ／ロードすることもできます。
このとき、拡張子として".EXC"を用います。
以下にメッセージ付きオブジェクト・プログラムのセーブ、ロードについて説明します。

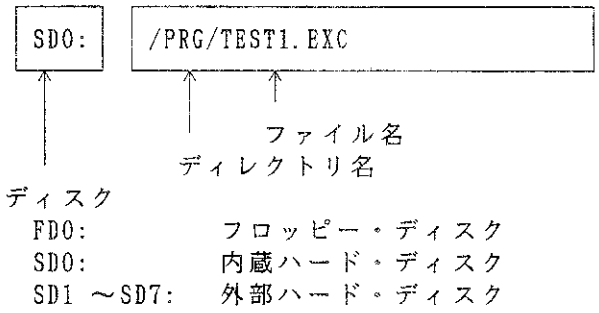
① シミュレーション・アプリケーションを選択します。
Dチャネル・シミュレーションの場合は“SIMULATOR LAPD”を、Bチャネル・シミュレーションは“SIMULATOR LAPB”を選択します。選択はF1キーを押して現れるポップアップ・メニューで行なうか、すでにアプリケーションをロードしてある場合はALTキー＋ファンクション・キーで行ないます。

② カーソルが  の位置で、スペース・キー（またはリターン・キ）を押すとセーブ／ロード画面が表示されます。

③ Data Type の位置にカーソルを移動させ、スペース・キー（またはリターン・キー）を押して下さい。▲、▼キーで、“Message & Object”を選択し、スペース・キー（またはリターン・キー）を押して下さい。

④ Operation の位置にカーソルを移動させ、スペース・キー（またはリターン・キー）を押して下さい。“Save”または“Load”のうち希望の位置にカーソルを移動させ、スペース・キーを押して下さい。

⑤ ファイル名を入力する位置にカーソルを移動させ、セーブまたはロードするファイル名をタイプして下さい。ファイル名入力をする位置のフォーマットは以下のようになっています。



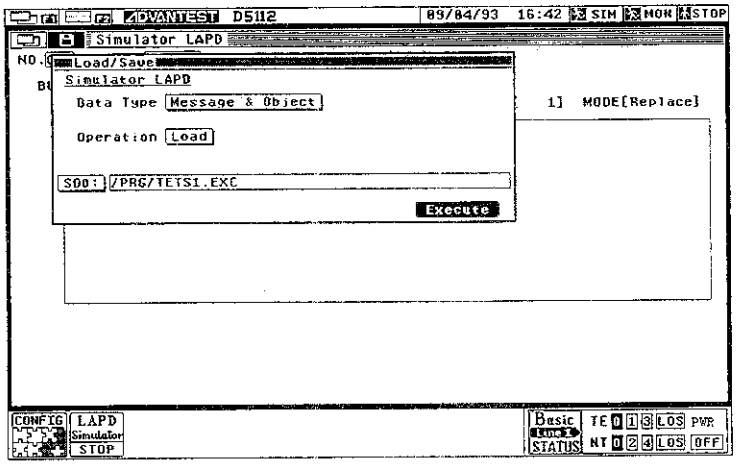


図 8 - 5　メッセージ付きオブジェクトのセーブ

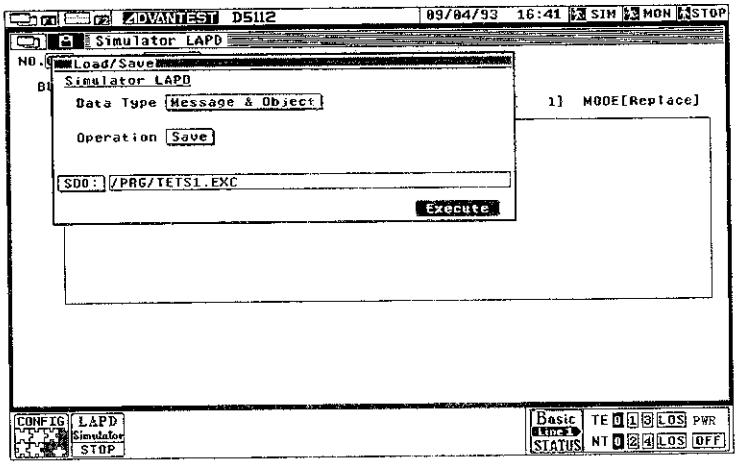


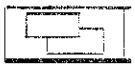
図 8 - 6　メッセージ付きオブジェクトのロード

9. プログラムを用いないでシミュレーション関数を実行する

シミュレーションのCommand 欄でコマンドを打ち込むことにより、シミュレーションを行なうことができます。
だだし、使用できる関数はプログラムを用いて行なう場合に比べて、限定されます。
D チャネル・シミュレーションで使用できるコマンドとB チャネル・シミュレーションで使用できるコマンドについて、それぞれ [表9-1]と [表9-2]に示します。

＜操作手順＞

① シミュレーション・アプリケーションを選択します。
D チャネル・シミュレーションの場合は“SIMULATOR LAPD”を、B チャネル・シミュレーションは“SIMULATOR LAPB”を選択します。選択はF1キーを押して現れるポップアップ・メニューで行なうか、すでにアプリケーションをロードしてある場合はALTキー＋ファンクション・キーで行ないます。

② カーソルを▲、▼キーを用いて  の所へもって行き、スペース・キーまたは、リターン・キーを押してください。

③ ▲、▼キーで、“SIMULATOR”を選択し、スペース・キーまたは、リターン・キーを押して下さい。

④ ▼キーを押し、Command:の欄にカーソルを移動させて下さい。

⑤ 実行したいシミュレーションのコマンド名および引数をタイプして下さい。

表 9－1 Dチャネル用コマンド

コマンド名	引数1	引数2	説明
① INCVS	－	－	V(S)値のインクリメント。
② INCVR	－	－	V(R)値のインクリメント。
③ INTERFACE	BRI	－	基本インタフェースに設定する。
	PRI	－	一次群インタフェースに設定する。
④ SENDF	名前	0/1	フレームを送出する。 （引数2はP/F ビット値）
	フレーム番号	0/1	フレームを送出する。 （引数2はP/F ビット値）
⑤ SENDT	名前	－	フレームをそのまま送出する。
	フレーム番号	－	フレームをそのまま送出する。
⑥ SET_CHANN	1 ～ 24	－	送出チャネルを設定する。 （引数1はチャネル番号）
⑦ SETVS	0 ～127	－	V(S)値を設定する。 （引数1はV(R)値）
⑧ SETVR	0 ～127	－	V(R)値を設定する。 （引数1はV(S)値）
⑨ SIMMODE	TE	－	TEモードに設定する。
	NT		NTモードに設定する。
⑩ SIMSTATUS	－	－	③⑨⑪の設定状態を表示する。
⑪ PFEED	NORM	－	ノーマル給電に設定する。
	RVS	－	リバース給電に設定する。
	OFF	－	給電ＯＦＦ状態にする。
⑫ PH_ACT	－	－	レイヤ1 を起動する。
⑬ PH_DEACT	－	－	レイヤ1 を停止する。
⑭ PH_DEFAULT	－	－	レイヤ1 を停止し、③⑨⑪の設定を解除する。

表 9 - 2 B チャネル用コマンド

コマンド名	引数1	引数2	説明
① INCVS	−	−	V(S)値のインクリメント。
② INCVR	−	−	V(R)値のインクリメント。
③ INTERFACE	BRI	−	基本インタフェースに設定する。
	PRI	−	一次群インタフェースに設定する。
④ MODULO	8	−	モジュロ8 に設定する。
	128	−	モジュロ128 に設定する。
⑤ SENDF	名前	0/1	フレームを送出する。 （引数2 はP/F ビット値）
	フレーム番号	0/1	フレームを送出する。 （引数2 はP/F ビット値）
⑥ SENDT	名前		フレームをそのまま送出する。
	フレーム番号	−	フレームをそのまま送出する。
⑦ SET_CHANN	1 ～ 2(24)		送出Bチャネルを設定する。 （引数1 はチャネル番号）
⑧ SETVS	0 ～ 127(7)	−	V(S)値を設定する。 （引数1 はV(S)値）
⑨ SETVR	0 ～ 127(7)	−	V(R)値を設定する。 （引数1 はV(R)値）
⑩ SIMMODE	TE	−	TEモードに設定する。
	NT	−	NTモードに設定する。
⑪ SIMSTATUS	−	−	③⑩⑫の設定状態を表示する。
⑫ PFEED	NORM	−	ノーマル給電に設定する。
	RVS	−	リバース給電に設定する。
	OFF	−	給電ＯＦＦ状態にする。
⑬ PH_ACT	−	−	レイヤ1 を起動する。
⑭ PH_DEACT	−	−	レイヤ1 を停止する。
⑮ PH_DEFAULT	−	−	レイヤ1 を停止し、③⑩⑫の設定を解除する。

〔例〕本器をNTとして用い、メッセージ・ビルダに登録されているSABME という名のフレームを送信する手順を以下に示します。

＜操作手順＞

① SIMMODE ␣ NT ↵ （NTモードに設定）
↓
② PFEED ␣ RVS ↵ （リバース給電に設定）
↓
③ PH_ACT ↵ （レイヤ1 を起動する）
↓
④ SENDF ␣ SABME␣ 1 ↵ （P/F が1 のSABME という名前のフレームを送信）
⑤ PH_DEFAULT ↵ （シミュレーション関数を実行する以前の状態にもどす）

（注）SENDF コマンドを使用して、フレームを送出するには、メッセージが必要です。
［7.1 メッセージ・ビルダの概略］または［8.2 メッセージ・データのセーブ／ロード］を参照して下さい。

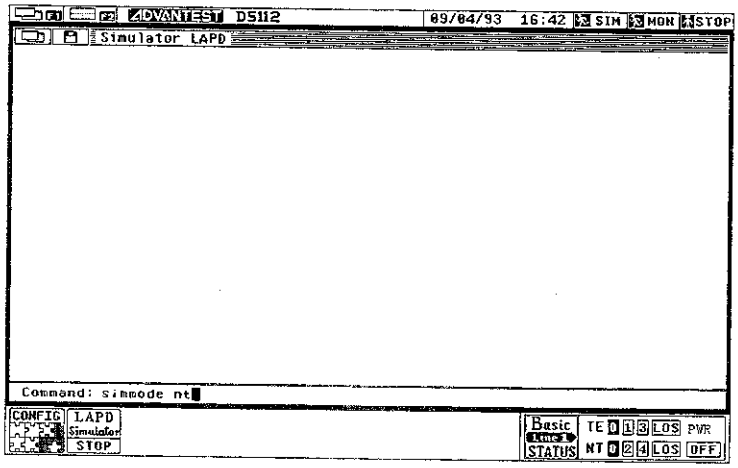
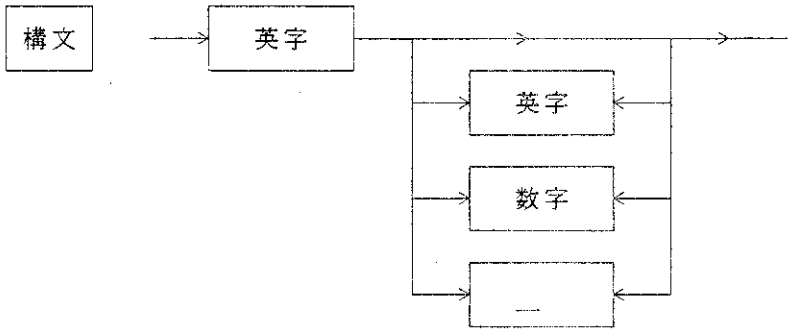


図 9 - 1 シミュレーション・コマンドの入力例

10. シミュレーション言語（PSL51 ）の構文について

(1) 変数

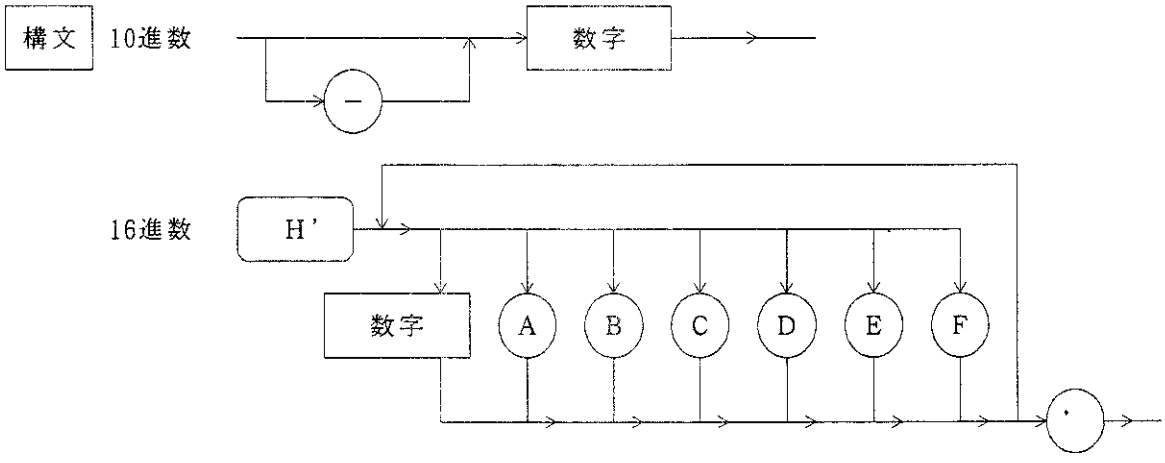


説明 変数は、英文字・数字または '_' で構成されます。また、認識される文字数は10文字までです。

例
```
ABC
X1234
Z_XY
A123456789X      （A123456789 と同じ）
```

※ 変数は、すべて全域的変数として定義されます。

(2) 定数



説明　定数には、10進数と16進数があり、-2147483648 ～2147483647（符号付32ビット）の値をとります。また、16進数は、H'F34'（10進数では3892）のように表現します。

例
```
ABC  =  32767
  X  =  H'F6D'
```

注意

● PSL51 のデータ型は符号付32ビットなので2147483648以上の10進数（16進数ではH'80000000'）は、負の値を示すことになります。

10進数	16 進数	実際の値
2147483648	80000000	2147483648
2147483649	80000001	2147483647
2147483650	80000002	2147483646
・	・	・
・	・	・
・	・	・
4294967294	FFFFFFFE	-2
4294967295	FFFFFFFF	-1

(3) 配列

構文　配列名 → 〔 → 配列の大きさ（数字） → 〕

説明　配列は、1 次元配列のみが使用可能で、変数同様-2147483648 ～2147483647（符号付32ビット）の値をとります。また、配列はプログラムの先頭で宣言することにより使用可能となります。
（〔2.1　PSL51 のプログラミング構造例〕を参照して下さい。）

次の例の様に配列（ARRAY ）をプログラムの先頭で宣言します。また、例では配列XYZ の初期化も以下のように行なっています。

```
XYZ[0]  =  1
XYZ[1]  =  2
XYZ[2]  =  3
XYZ[3]  =  4
XYZ[4]  =  XYZ[5]  =  XYZ[6]  =  5
XYZ[7]  =  XYZ[8]  =  XYZ[9]  =  0
```

例
```
ARRAY  A[10] , B[20] , C[4]
ARRAY  XYZ[10] =  [1, 2, 3, 4, 5 [3], 0 [3] ]
```

注意

- 配列の添字は、0 から始まります。
- 配列の初期化を行なう場合は、全要素の初期化を行なって下さい。
- 全配列の大きさは、変数との総計が32K 以下になるようにして下さい。
  32K以上になると誤動作の原因となる可能性があります。

(4) 演算子

説明 演算子には、加算（＋）・減算（－）・乗算（＊）・除算（／）があります。

例
```
AB1 =  X1(I)     * 6  -  XYZ  +  a  /  ab12
```

注意

- 除算では、結果は整数（-2147483648 ～2147483647）になる様に切り捨てられます。
  （例）
  123 / 12 = 10
  除数が0 のとき、値は（-1）になります。

(5) 関係演算子

説明 関係演算子には、＞，＜，＞＝，＜＝，＝＝，！＝ の6 種類があります。

	関係演算子	機能
(1)	＞	より大きい
(2)	＜	より小さい
(3)	＞＝	より大きいか、等しい
(4)	＜＝	より小さいか、等しい
(5)	＝＝	等しい
(6)	！＝	等しくない

例
```
IF   A＝＝B  THEN   C＝3 ＊A   ELSE   C＝A   END
IF   X！＝Y  THEN   Z＝5 ／W   ELSE   Z＝W   END
```

(6) 論理演算子

説明 論理演算子としては否定、（！）があります。
変数AB1 を例をあげ、論理演算子を使用したときの値を以下に示します。

変数 AB1 の値	! AB1 の値
0	1
0 以外の値	0

例
```
IF (!AB1)  THEN   a = 1  ELSE  a = 0  END
```

(7) ビットごとの論理演算子

説明 ビットごとの論理演算子には、&（ビットごとの論理積）と | （ビットごとの論理和）があります。

例
```
A   =  5
B   =  6
C   =  A   &  B      (C の値は4 )
D   =  A   |  B      (D の値は7 )
```

(8) 関数（FUNCtion）

説明 プログラムは複数の関数と呼ばれる集合体から構成されています。
関数は、次の様な構造をしていなくてはなりません。

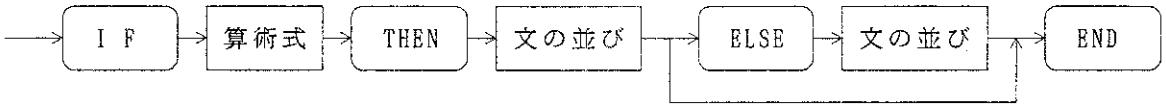
```
FUNC     関数名  (引数1 、引数2 、…  引数N )
    関数内での処理
RETURN (戻り値)
```

また、プログラムは必ずMAIN( ) という名前の関数から実行されるので、
MAIN( ) という関数は必ず作成しなければなりません。
引数や戻り値は省略することもできます。

例
```
FUNC      MAIN( )
   A  =  123
   B  =  256
   C  =  SUB1(A,B)
RETURN
FUNC      SUB1(ARG1, ARG2)
 VAL  =  ARG1+ 3  ×  ARG2
RETURN(VAL )
```

(9) IF文

構文

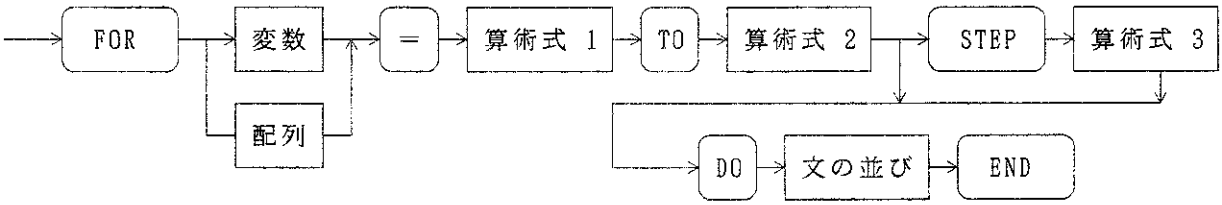


説明 算術式で書かれた条件が真（0 以外）ならば、THEN直後の命令文を実行します。もし偽ならばELSEがある場合は、ELSE直後の文を実行します。
ELSEがない場合は、END 以降の文を実行します。

例
```
IF ( ! check(arg1, arg2))    THEN ok = 0
                             ELSE ok = 1
END
.
.
.
.
FUNC  check(X, Y)
  IF  MODE = = 0    THEN  ret = X+Y
                    ELSE  ret = X-Y
  END
RETURN(ret)
```

(10) FOR 文

構文

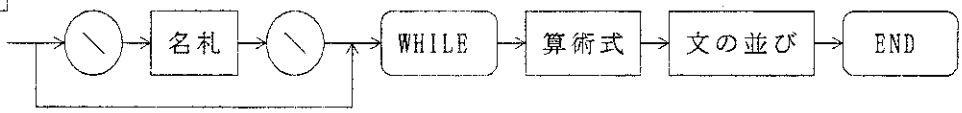


説明 FOR 以下の条件式に従い、DOからEND で囲まれた文の並びを繰り返して実行します。FOR 以下の条件式は、まず、変数あるいは配列に算術式1 の初期値が代入され、DO～END の文の並びを実行するごとに、算術式3 の値だけ、変数あるいは配列が加算されます。
DO～END の文の並びは変数あるいは配列が算術式2 と等しくなるまで繰り返して実行されます。“STEP算術式3 ”は、省略することもでき、省略すると増分は、自動的に1 に設定されます。

例
```
FOR I = 0 TO 13 STEP 1 DO
  A[I] = I+2
  PRINT ("A[%D]=%D ＼N", I, A[I] )
END
```

(11) WHILE 文

構文



説明　算術式が真（0 以外）の間、算術式からEND で囲まれた文の並びを繰り返して実行します。

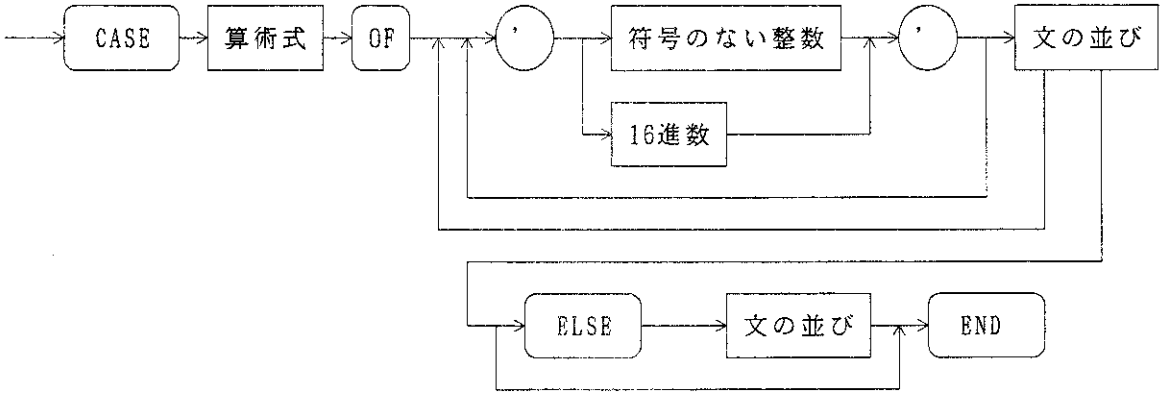
例
```
I = 10
WHILE I
  AB[I] = xyz+ I
  I = I - 1
  PRINT ("AB[%D]=%D＼N", I, AB[I])
END
```

注意

● WHILE 文のループから抜け出すためには、EXIT文及び名札を併用しますが、詳細は、(14)EXIT文の項を参照して下さい。

(12) CASE文

構文



説明 次の例では、変数xyz の値

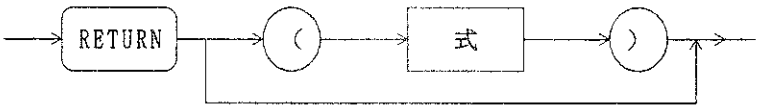
0	または 1	または	3 のとき	関数 proc1 (0)
2	または 4	のとき		関数 proc1 (1)
5	のとき			関数 proc1 (2)
それ以外のときは				関数 proc1 (3)

がそれぞれ実行されます。

例
```
CASE  xyz   OF
  '0' '1'  '3'          proc1 (0 )
  '2' '4'               proc1 (1 )
  '5'                   proc1 (2 )
  ELSE                  proc1 (3 )
END
```

(13) RETURN文

構文



説明 RETURN文は、関数の実行が終了したとき、それを呼び出した関数に制御を戻すための文です。使用方法は、次表の通りです。

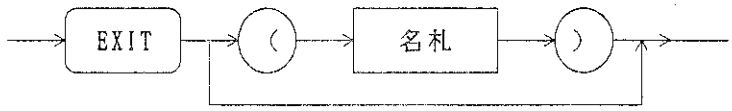
関数終了時 (戻り値がある場合)	RETURN (戻り値)
関数終了時 (戻り値がない場合)	RETURN

例
```
FUNC    func1(arg1, arg2)
  .
  .
  .
  .
  .
RETURN  (xyz + 2)   /* 関数の終了 (戻り値がある場合) */

FUNC    func1(arg1, arg2)
  .
  .
  .
RETURN              /* 関数の終了 (戻り値がない場合) */
```

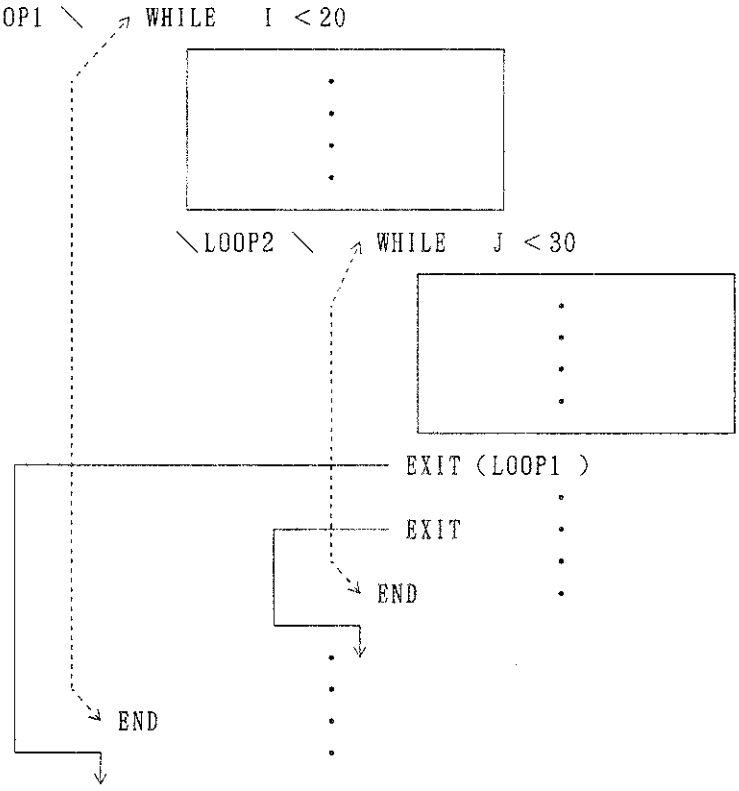
⑭ EXIT文

構文

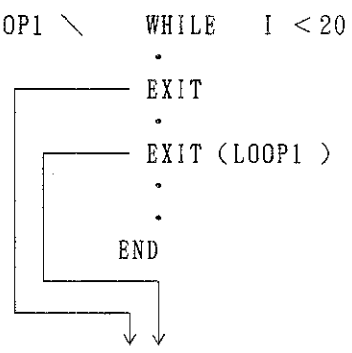


説明 EXIT文は、現在実行されている最も内側のWHILE 文ループを強制的に終了し、そのループを抜け出します。また、名札が指定されているときには、その名札を持つWHILE 文により構成されているWHILE ループを抜け出します。

例 1

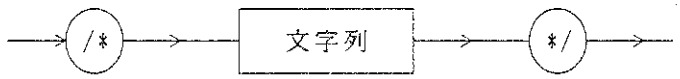


例 2



(15) コメント文（注釈文）

構文

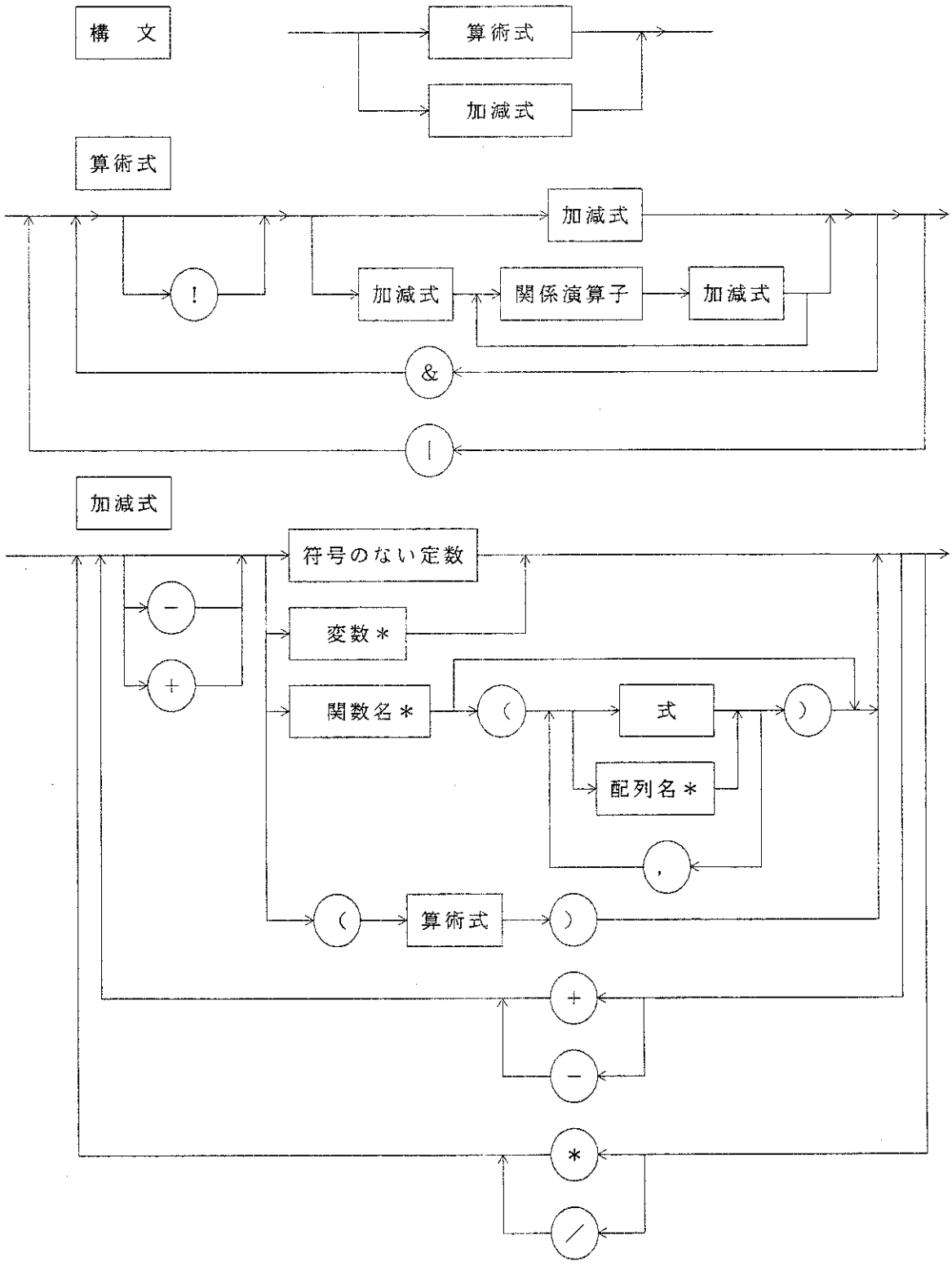


説明 プログラム中において、／＊ と ＊／ で挟まれた文字列は、コメント（注釈）と解釈されコンパイルされません。

例

```
／＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊
    This  is  COMMENT  example  ！！！
＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊／
IF  ab  ＝＝ xy            ／＊ same  value  ？  ＊／
    THEN        PRINT    (" sb  ＝＝ xy ＼ n" )
    ELSE        PRINT    (" sb  ！＝ xy ＼ n" )
END
```

⑯ 式



＊注意を参照して下さい。

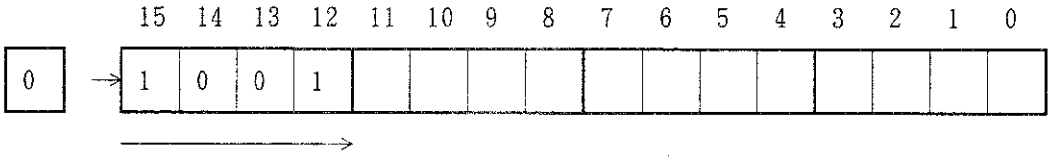
注意

以下の語は予約語として使用されているため、変数・関数名・配列名としては使用できません。

ARRAY	STEP	R_SHIFT
CASE	THEN	L_SHIFT
DO	TO	PRINT
ELSE	WHILE	INPUT
END	LAYER	RND
EXIT	SIMMODE	RANDOMIZE
FOR	PFEED	GET_TIME
FUNC	CHANNEL	
IF	INTERFACE	
OF	ADDRESS	
RETURN	MODULO	

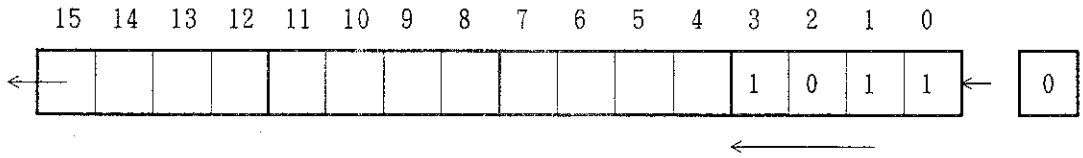
(17) R_SHIFT(val, times)

機能 関数値として、引数val の値を引数times で指定された回数だけ右（Right）シフトした値を返します。ただし、論理シフトが以下の様に実行されます。



(18) L_SHIFT(val, times)

機能 関数値として、引数val の値を引数times で指定された回数だけ左（Left）シフトした値を返します。ただし、論理シフトが以下の様に実行されます。



例
```
IF  data  >=  20
    THEN  data  =  R-SHIFT  (data, 4 )
    ELSE  data  =  L-SHIFT  (data, 4 )

END
```

⑲ PRINT("書式", arg1, arg2, …)

機能1 引数で指定された書式に従って画面に表示します。"書式"以外に引数がない場合は"　　"で囲まれた文字列をそのまま表示します。
arg1, arg2,…には、変数が与えられますが、その表示の指示は、書式内の%で始まる文字コードに従います。
表示形式は、大きく分けて以下の4 種類があります。

① %〔+〕〔0 〕〔n 〕d …10進表示変換
+ :数値が正の場合には"+"を先頭に表示します。
0 :フィールドの先頭に0 でうめます。
n :交換した文字をおさめるフィールド長を桁数で指定します。
d :10進数に変換します。
〔 〕で囲まれた指示は省略することもできます。
〔例〕%D → 123
%5D → 123
%05D → 00123
%+5D → + 123

② %〔0 〕〔n 〕x … 16進表示変換
0 :フィールドの先頭に0 でうめます。
n :交換した文字をおさめるフィールド長を桁数で指定します。
x :16進数に変換します。
〔 〕で囲まれた指示は省略することもできます。
〔例〕%x → 123
%5x → 123
%05x → 00123
%+5x → + 123

③ %〔0 〕〔n 〕u
0 :フィールドの先頭に0 でうめます。
n :交換した文字をおさめるフィールド長を桁数で指定します。
u :符号無し10進数に変換します。
〔 〕で囲まれた指示は省略することもできます。
〔例〕%u → 123
%5u → 123
%05u → 00123
%+5u → + 123

④ %〔n 〕b
n :交換した文字をおさめるフィールド長を桁数で指定します。
b :ブランクを表示します。
〔例〕%5b →

＊書式中で"＼n "または"＼N "を指定すると改行されます。

例

```
A = 123
B = -71
C =  51                    ← 改行指定
PRINT("SAMPLE ")
PRINT("PROGRAM  !!\N")
PRINT("A=%D(%3x)  ",A,A )
PRINT("B=%5D\N",B)
PRINT("C=%U B=%05D\N",C,B)
```

結果

```
SAMPLE PROGRAM  !!
A=123( 7B)  B=  -71
C=51 B=-0071
```

機能 2

PRINT 文を多数回実行するとシミュレーション実行結果表示画面から文字列がスクロール・アウトされてしまいます。が、本体内部のメモリに最大100行が保持されています。シミュレーションの実行停止後に、⇧、⇩キーにより前後スクロールして画面からスクロール・アウトした内容を確認することができます。100 行を越えてPRINT 文が実行されてしまうと、内部メモリ内に保持されないためシミュレーション停止後に内容確認することができません。
この問題を解決するために、本器ではPRINT 文により画面表示される内容を内蔵ハード・ディスク内に、Dチャネル・シミュレーションの場合は"SD0:/LST/SIMLOGD" というファイル名で、Bチャネル・シミュレーションの場合は"SD0:/LST/SIMLOGB" というファイル名で記録しています。
このファイルは、本体のエディタまたは、コンソール機能のTYPEコマンドにより内容確認することができます。また、3.5 インチ・フロッピー・ディスクにコピーすれば、MS-DOSのパソコン上でも内容確認することができます。

*: SD0: は内蔵ハード・ディスクを意味します。

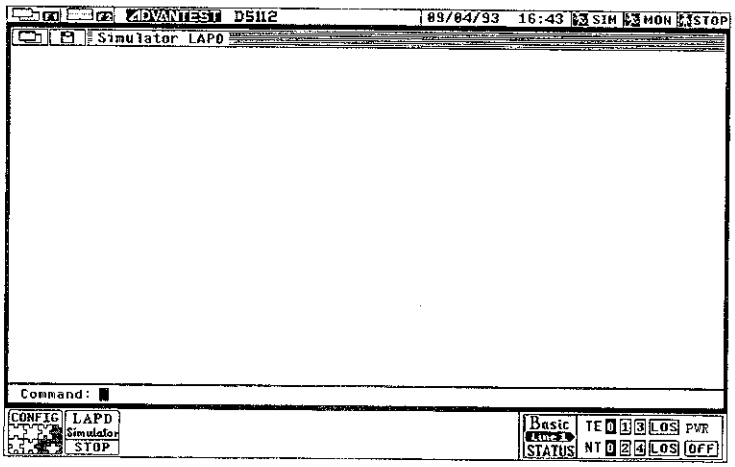
宣言部にLOGFILE ONを定義することにより、PRINT 文実行時の時刻を表示ることができるため、エラーが生じた時刻等をログ・ファイルとして記録しておけます。

注意

- シミュレーションを起動すると、内蔵ハード・ディスク内にログ・ファイルとして
  "SD0:/LST/SIMLOGD" または、"SD0:/LST/SIMLOGB"
  が作成されます。

⑳ INPUT( )

機能 キー入力された値を関数値として返します。
キー入力は、カーソルが Commandの位置（下図参照）にないと実行できません。
キー入力は、10進数入力と16進数入力の2 通りがあります。詳しくは、定数説明を参照して下さい。



例
```
PRINT("INPUT A=")
A=      INPUT( )
PRINT("%D\N",A)
```

(21) RND( )

機能 0 ～255 のランダム値を発生します。
乱数の系列は、本器を立ち上げるごとに異なるように設定されています。乱数の系列を固定で使用する場合は、RANDOMIZE(SEED) 関数により乱数の種を指定して下さい。ただし、内部でもこの関数を使用しているので、プログラム中では、指定した系列どおりに発生しないことがあります。

例
```
B=RND( )
PRINT("RANDOM=%D＼N",B)
```

(22) RANDOMIZE(SEED)

機能 新しい乱数の種(SEED)を指定することにより乱数の系列を変更することができます。ただし0 ×1234567 は、内部で特別な意味を持っているので、使用しないで下さい。また、内部でもRND 関数を(TEI割当手順などで) 使用しているので、SEEDが同じでもユーザ・プログラム中でRND 関数により得られるランダム値は、プログラムを実行させるごとに異なった発生をすることがあります。

例
```
RANDOMIZE(1)
B=RND( )
PRINT("RANDOM=%D＼N",B)
```

(23) GET_TIME( )

機能 内部クロックの現在値を読み出します。（10msec分解能です。）

例
```
TIME=GET_TIME( )
PRINT("TIME=%D＼N",TIME)
```

(24) BEEP( )

機能 BEEP音を発生させます。正常終了時は関数値として0 を返します。

例
```
RET=BEEP( )
PRINT("RETURN VAL=(%D)＼N",RET)
```

11. 機能別関数一覧

11.1 D チャネル・シミュレーション

(1) 共通関数

(2) トランスペアレント・モード用関数

関数名		内容	ページ
(1)	SENDT	フレームを送信する。（ﾒｯｾｰｼﾞ･ﾃﾞｰﾀをそのままま送出）	12-18
(2)	SENDF	フレームを送信する。(N(S),N(R),P/Fを挿入して送出)	12-19
(3)	INCVS	送信状態変数V(S)を＋1 する。	12-20
(4)	INCVR	受信状態変数V(R)を＋1 する。	12-21
(5)	SETVS	送信状態変数V(S)の値を設定する。	12-22
(6)	SETVR	受信状態変数V(R)の値を設定する。	12-23
(7)	INS_SAPI	送信ﾌﾚｰﾑのSAPI値を書き換える。	12-24
(8)	INS_TEI	送信ﾌﾚｰﾑのTEI 値を書き換える。	
(9)	INS_CR	送信ﾌﾚｰﾑのC/R ビット値を書き換える。	
(10)	INS_NR	送信ﾌﾚｰﾑのN(R)値を書き換える。	
(11)	INS_NS	送信ﾌﾚｰﾑのN(S)値を書き換える。	
(12)	INS_PF	送信ﾌﾚｰﾑのP/F ビット値を書き換える。	
(13)	INS_TYPE	送信ﾌﾚｰﾑのフレーム種別を書き換える。	
(14)	INS_CF1	送信ﾌﾚｰﾑ制御ﾌｨｰﾙﾄﾞ の第１ｵｸﾃｯﾄ値を書き換える。	
(15)	INS_CF2	送信ﾌﾚｰﾑ制御ﾌｨｰﾙﾄﾞ の第２ｵｸﾃｯﾄ値を書き換える。	
(16)	INS_FRCF1	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内の第１ ｵｸﾃｯﾄ値を書き換える。	
(17)	INS_FRCF2	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内の第２ ｵｸﾃｯﾄ値を書き換える。	
(18)	INS_FRVS	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内のV(S)値を書き換える。	
(19)	INS_FRVR	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内のV(R)値を書き換える。	
(20)	INS_FRCR	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内のC/R ビット値を書き換える。	
(21)	INS_FRWXYZ	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内のWXYZビット値を書き換える。	12-24
(22)	RXSAPI	受信ﾌﾚｰﾑのSAPI値を読み出す。	12-26
(23)	RXTEI	受信ﾌﾚｰﾑのTEI 値を読み出す。	
(24)	RXCR	受信ﾌﾚｰﾑのC/R ビット値を読み出す。	
(25)	RXNR	受信ﾌﾚｰﾑのN(R)値を読み出す。	
(26)	RXNS	受信ﾌﾚｰﾑのN(S)値を読み出す。	
(27)	RXPF	受信ﾌﾚｰﾑのP/F ビット値を読み出す。	
(28)	RXTYPE	受信ﾌﾚｰﾑのフレーム種別を読み出す。	
(29)	RXCF1	受信ﾌﾚｰﾑ制御ﾌｨｰﾙﾄﾞ 内の第１ ｵｸﾃｯﾄ値を読み出す。	
(30)	RXCF2	受信ﾌﾚｰﾑ制御ﾌｨｰﾙﾄﾞ 内の第２ ｵｸﾃｯﾄ値を読み出す。	
(31)	RXFRCF1	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内の第１ ｵｸﾃｯﾄ値を読み出す。	
(32)	RXFRCF2	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内の第２ ｵｸﾃｯﾄ値を読み出す。	
(33)	RXFRVS	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内のV(S)値を(S) 値を読み出す。	
(34)	RXFRVR	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内のV(R)値を(R) 値を読み出す。	
(35)	RXFRCR	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内の C/Rビット値を読み出す。	
(36)	RXFRWXYZ	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ 内のWXYZビット値を読み出す。	12-26

(3) レイヤ2自動モード用関数

フレーム送信関連		
関数名	内容	ページ
(1) SENDI	Iフレームを送信する。	12-31
(2) SENDUI	UIフレームを送信する。	12-32
(3) SENDXIDC	XIDコマンドを送信する。	12-33
(4) SENDXIDR	XIDレスポンスを送信する。	12-34
(5) SENDPKT	パケットを送信する。	12-35
(6) INCPS	送信順序番号P(S)を+1にする。	12-36
(7) INCPR	受信順序番号P(R)を+1にする。	12-37
(8) SETPS	送信順序番号P(S)の値を設定する。	12-38
(9) SETPR	受信順序番号P(R)の値を設定する。	12-39
(10) INS_PD	送信フレームのプロトコル識別子を書き換える。	12-40
(11) INS_CRL	送信フレームの呼番号長を書き換える。	
(12) INS_CRF	送信フレームの呼番号フラグ値を書き換える。	
(13) INS_CRV	送信フレームの呼番号値を書き換える。	
(14) INS_MSG	送信フレームのメッセージ種別を書き換える。	
(15) INS_INFO	送信フレームの情報要素群フィールドを書き換える。	
(16) INS_CS_VAL	送信フレームの理由表示値を書き換える。	12-40
(17) INS_GFI	送信ﾊﾟｹｯﾄのｾﾈﾗﾙﾌｫｰﾏｯﾄ識別子の値を書き換える。	12-42
(18) INS_Q	送信ﾊﾟｹｯﾄのQビット値を書き換える。	
(19) INS_D	送信ﾊﾟｹｯﾄのDビット値を書き換える。	
(20) INS_LCGN	送信ﾊﾟｹｯﾄの論理チャネルグループ番号を書き換える。	
(21) INS_LCN	送信ﾊﾟｹｯﾄの論理チャネル番号を書き換える。	
(22) INS_TYP	送信ﾊﾟｹｯﾄのパケットタイプ識別子を書き換える。	
(23) INS_PR	送信ﾊﾟｹｯﾄの受信順序番号P(R)を書き換える。	
(24) INS_PS	送信ﾊﾟｹｯﾄの送信順序番号P(S)を書き換える。	
(25) INS_M	送信ﾊﾟｹｯﾄのモアデータ表示値を書き換える。	
(26) INS_CLL	送信ﾊﾟｹｯﾄの発呼ユーザアドレス長を書き換える。	
(27) INS_CDL	送信ﾊﾟｹｯﾄの着呼ユーザアドレス長を書き換える。	
(28) INS_DA	送信ﾊﾟｹｯﾄの着呼ユーザアドレスを書き換える。	
(29) INS_SA	送信ﾊﾟｹｯﾄの発呼ユーザアドレスを書き換える。	
(30) INS_FL	送信ﾊﾟｹｯﾄのファシリティ長を書き換える。	
(31) INS_F	送信ﾊﾟｹｯﾄのファシリティを書き換える。	
(32) INS_CAUSE	送信ﾊﾟｹｯﾄの切断／ﾘｽﾀｰﾄ/ﾘｾｯﾄ原因を書き換える。	
(33) INS_DIAG	送信パケットの診断符号を書き換える。	
(34) SET_DTL	送信パケット内のデータ長を書き換える。	
(35) INS_DATA	送信パケット内のデータを書き換える。	12-42

TEI管理手順関連			
	関数名	内容	ページ
(36)	REQ_TEI	TEI 割当手順を起動する。（TEモード）	12-43
(37)	CHKREQ_TEI	TEI チェック手順を行なう。（NTモード）	12-44
(38)	REMOVE_TEI	TEI 解除手順を行なう。（NTモード）	12-45
(39)	VERIFY_TEI	TEI 検証手順を行なう。（TEモード）	12-46

リンク，SAPI，TEI 管理関連			
	関数名	内容	ページ
(40)	LINKON	リンクの設定を行なう。	12-47
(41)	LINKOFF	リンクの解放を行なう。	12-49
(42)	WAIT_LINK	相手局からのリンク設定待ち状態にする。	12-50
(43)	L_STATUS	リンクが設定されているかを調べる。	12-51
(44)	SET_BUSY	自局をビジー状態にする。	12-52
(45)	REL_BUSY	自局のビジー状態を解除する。	12-53
(46)	PROHIBIT_L	新たなリンクの設定を禁止する。	12-54
(47)	PERMIT_L	リンク設定不許可状態を解除する。	12-55
(48)	REG_TEI	TEI 値を本器に登録し、そのTEI 値のﾌﾚｰﾑを受信可能にする。	12-56
(49)	REL_TEI	TEI 値を解除し、そのTEI 値のﾌﾚｰﾑを受信しないようにする。	12-57
(50)	NEXT_TEI	新たなTEI 値を使用することを宣言する。（TEモード）	12-58
(51)	ACT_SAPI	フレーム送出時に使用されるSAPI値を設定する。	12-59
(52)	ACT_TEI	フレーム送出時に使用されるTEI 値を設定する。	12-60
(53)	ACT_LINK	フレーム送出時に使用されるリンク番号を設定する。	12-61
(54)	LOCK_SAPI	SAPI値が自動で設定変更されるのを禁止する。	12-62
(55)	LOCK_TEI	TEI 値が自動で設定変更されるのを禁止する。	12-63
(56)	LOCK_LINK	リンク番号が自動で設定変更されるのを禁止する。	12-64
(57)	FLEX_SAPI	SAPI値が自動で設定変更されるようにする。	12-65
(58)	FLEX_TEI	TEI 値が自動で設定変更されるようにする。	12-66
(59)	FLEX_LINK	リンク番号が自動で設定変更されるようにする。	12-67
(60)	GET_SAPI	フレーム送出時に使用されるSAPI値を調べる。	12-68
(61)	GET_TEI	フレーム送出時に使用されるTEI 値を調べる。	12-69
(62)	GET_LINK	フレーム送出時に使用されるリンク番号を調べる。	12-70
(63)	SEE_LINK	現在設定されているリンク番号を調べる。	12-71

フレーム受信関連			
	関数名	内容	ページ
(64)	RXSAPI	受信フレームのSAPI値を読み出す。	12-72
(65)	RXTEI	受信フレームのTEI 値を読み出す。	
(66)	RXTYPE	受信フレームのフレーム種別を読み出す。	
(67)	RXPD	受信フレームのプロトコル識別子を読み出す。	
(68)	RXCRL	受信フレームの呼番号長を読み出す。	
(69)	RXCRF	受信フレームの呼番号フラグ値を読み出す。	
(70)	RXCRV	受信フレームの呼番号値を読み出す。	
(71)	RXMSG	受信フレームのメッセージ種別を読み出す。	
(72)	RXINFO_NUM	受信フレームの情報要素数を読み出す。	
(73)	RXINFO_ELM	受信フレームの情報要素識別子を読み出す。	
(74)	RXINFO_LEN	受信フレームの情報要素内容長を読み出す。	12-72, 73
(75)	RXINFO_VAL	受信フレームの情報内容を読み出す。	12-72, 74
(76)	RXINFO_POS	受信フレームの情報要素識別子の位置を調べる。	12-72, 76
(77)	RXCS_VAL	受信フレームの理由表示値を読み出す。	12-72
(78)	RXCHAN_NUM	受信フレームのチャネル番号を読み出す。	12-72, 77
(79)	RXGFI	受信パケットのｾﾈﾗﾙﾌｫｰﾏｯﾄ識別子の値を読み出す。	12-78
(80)	RXQ	受信パケットのQ ビット値を読み出す。	
(81)	RXD	受信パケットのD ビット値を読み出す。	
(82)	RXLCGN	受信パケットの論理ﾁｬﾈﾙｸﾞﾙｰﾌﾟ番号を読み出す。	
(83)	RXLCN	受信パケットの論理チャネル番号を読み出す。	
(84)	RXTYP	受信パケットのパケットタイプ識別子を読み出す。	
(85)	RXPR	受信パケットの受信順序番号 P(R) を読み出す。	
(86)	RXPS	受信パケットの送信順序番号 P(S) を読み出す。	
(87)	RXM	受信パケットのモアデータ表示値を読み出す。	
(88)	RXCLL	受信パケットの発呼ユーザアドレス長を読み出す。	
(89)	RXCDL	受信パケットの着呼ユーザアドレス長を読み出す。	
(90)	RXDA	受信パケットの着呼ユーザアドレスを読み出す。	
(91)	RXSA	受信パケットの発呼ユーザアドレスを読み出す。	
(92)	RXFL	受信パケットのファシリティ長を読み出す。	
(93)	RXF	受信パケットのファシリティを読み出す。	
(94)	RXCAUSE	受信パケットの切断／ﾘｽﾀｰﾄ/ﾘｾｯﾄ原因を読み出す。	
(95)	RXDIAG	受信パケットの診断符号を読み出す。	
(96)	RXDTL	受信パケット内のデータ長を読み出す。	
(97)	RXDATA	受信パケット内のデータを読み出す。	12-78

11.2 B チャネル・シミュレーション

(1) 共通関数

(2) トランスペアレント・モード用関数

フレーム送信関連			
	関数名	内容	ページ
(1)	SENDT	フレームを送信する。（ﾒｯｾｰｼﾞ･ﾃﾞｰﾀ をそのまま送出）	13-17
(2)	SENDF	フレームを送信する。(N(S),N(R),P/Fを挿入して送出)	13-18
(3)	INCVS	送信状態変数 V(S) を＋1 する。	13-19
(4)	INCVR	受信状態変数 V(R) を＋1 する。	13-20
(5)	SETVS	送信状態変数 V(S) の値を設定する。	13-21
(6)	SETVR	受信状態変数 V(R) の値を設定する。	13-22
(7)	INS_ADRS	送信フレームのアドレス値を書き換える。	13-23
(8)	INS_NR	送信フレームのN(R)値を書き換える。	
(9)	INS_NS	送信フレームのN(S)値を書き換える。	
(10)	INS_PF	送信フレームのP/F ビット値を書き換える。	
(11)	INS_TYPE	送信フレームのフレーム種別を書き換える。	
(12)	INS_CF1	送信ﾌﾚｰﾑ制御ﾌｨｰﾙﾄﾞ の第１オクテットの値を書き換える。	
(13)	INS_CF2	送信ﾌﾚｰﾑ制御ﾌｨｰﾙﾄﾞ の第２オクテットの値を書き換える。	
(14)	INS_FRCF1	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内の第１ ｵｸﾃｯﾄ値を書き換える。	
(15)	INS_FRCF2	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内の第２ ｵｸﾃｯﾄ値を書き換える。	
(16)	INS_FRVS	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内のV(S)値を書き換える。	
(17)	INS_FRVR	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内のV(R)値を書き換える。	
(18)	INS_FRCR	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内の C/Rビット値を書き換える。	
(19)	INS_FRWXYZ	送信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内のWXYZビット値を書き換える。	13-23

フレーム受信関連			
	関数名	内容	ページ
(20)	RXADRS	受信フレームのアドレス値を読み出す。	13-25
(21)	RXNR	受信フレームのN(R)値を読み出す。	
(22)	RXNS	受信フレームのN(S)値を読み出す。	
(23)	RXPF	受信フレームのP/F ビット値を読み出す。	
(24)	RXTYPE	受信フレームのフレーム種別を読み出す。	
(25)	RXCF1	受信ﾌﾚｰﾑ制御ﾌｨｰﾙﾄﾞ の第１オクテットの値を読み出す。	
(26)	RXCF2	受信ﾌﾚｰﾑ制御ﾌｨｰﾙﾄﾞ の第２オクテットの値を読み出す。	
(27)	RXFRCF1	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内の第１ ｵｸﾃｯﾄ値を読み出す。	
(28)	RXFRCF2	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内の第２ ｵｸﾃｯﾄ値を読み出す。	
(29)	RXFRVS	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内のV(S)値を読み出す。	
(30)	RXFRVR	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内のV(R)値を読み出す。	
(31)	RXFRCR	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内の C/Rビット値を読み出す。	
(32)	RXFRWXYZ	受信FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内のWXYZビット値を読み出す	13-25

(3) レイヤ2自動モード用関数

フレーム送信関連			
関数名		内容	ページ
(1)	SENDI	Iフレームを送信する。	13-29
(2)	SENDPKT	パケットを送信する。	13-30
(3)	INCPS	送信順序番号 P(S) を+1する。	13-31
(4)	INCPR	受信順序番号 P(R) を+1する。	13-32
(5)	SETPS	送信順序番号 P(S) の値を設定する。	13-33
(6)	SETPR	受信順序番号 P(R) の値を設定する。	13-34
(7)	INS_GFI	送信ﾊﾟｹｯﾄのｾﾈﾗﾙﾌｫｰﾏｯﾄ識別子の値を書き換える。	13-35
(8)	INS_Q	送信ﾊﾟｹｯﾄのQビット値を書き換える。	
(9)	INS_D	送信ﾊﾟｹｯﾄのDビット値を書き換える。	
(10)	INS_LCGN	送信ﾊﾟｹｯﾄの論理チャネルグループ番号を書き換える。	
(11)	INS_LCN	送信ﾊﾟｹｯﾄの論理チャネル番号を書き換える。	
(12)	INS_TYP	送信ﾊﾟｹｯﾄのパケットタイプ識別子を書き換える。	
(13)	INS_PR	送信ﾊﾟｹｯﾄの受信順序番号 P(R) を書き換える。	
(14)	INS_PS	送信ﾊﾟｹｯﾄの送信順序番号 P(S) を書き換える。	
(15)	INS_M	送信ﾊﾟｹｯﾄのモアデータ表示値を書き換える。	
(16)	INS_CLL	送信ﾊﾟｹｯﾄの発呼ユーザアドレス長を書き換える。	
(17)	INS_CDL	送信ﾊﾟｹｯﾄの着呼ユーザアドレス長を書き換える。	
(18)	INS_DA	送信ﾊﾟｹｯﾄの着呼ユーザアドレスを書き換える。	
(19)	INS_SA	送信ﾊﾟｹｯﾄの発呼ユーザアドレスを書き換える。	
(20)	INS_FL	送信ﾊﾟｹｯﾄのファシリティ長を書き換える。	
(21)	INS_F	送信ﾊﾟｹｯﾄのファシリティを書き換える。	
(22)	INS_CAUSE	送信ﾊﾟｹｯﾄの切断／ﾘｽﾀｰﾄ/ﾘｾｯﾄ原因を書き換える。	
(23)	INS_DIAG	送信ﾊﾟｹｯﾄの診断符号を書き換える。	
(24)	SET_DTL	送信ﾊﾟｹｯﾄ内のデータ長を書き換える。	
(25)	INS_DATA	送信ﾊﾟｹｯﾄ内のデータを書き換える。	13-35

リンク関連			
関数名		内容	ページ
(26)	LINKON	リンクの設定を行なう。	13-37
(27)	LINKOFF	リンクの解放を行なう。	13-38
(28)	WAIT_LINK	相手リンク設定待ち状態にする。	13-39
(29)	L_STATUS	リンクが設定されているかを調べる。	13-40
(30)	SET_BUSY	自局をビジー状態にする。	13-41
(31)	REL_BUSY	自局のビジー状態を解除する。	13-42

フレーム受信関連		
関数名	内容	ページ
(32) RXGFI	受信パケットのｾﾈﾗﾙﾌｫｰﾏｯﾄ識別子の値を読み出す。	13-43
(33) RXQ	受信パケットのQ ビット値を読み出す。	
(34) RXD	受信パケットのD ビット値を読み出す。	
(35) RXLCGN	受信パケットの論理チャネルグループ番号を読み出す。	
(36) RXLCN	受信パケットの論理チャネル番号を読み出す。	
(37) RXTYP	受信パケットのパケットタイプ識別子を読み出す。	
(38) RXPR	受信パケットの受信順序番号 P(R) を読み出す。	
(39) RXPS	受信パケットの送信順序番号 P(S) を読み出す。	
(40) RXM	受信パケットのモアデータ表示値を読み出す。	
(41) RXCLL	受信パケットの発呼ユーザアドレス長を読み出す。	
(42) RXCDL	受信パケットの着呼ユーザアドレス長を読み出す。	
(43) RXDA	受信パケットの着呼ユーザアドレスを読み出す。	
(44) RXSA	受信パケットの発呼ユーザアドレスを読み出す。	
(45) RXFL	受信パケットのファシリティ長を読み出す。	
(46) RXF	受信パケットのファシリティを読み出す。	
(47) RXCAUSE	受信パケットの切断／ﾘｽﾀｰﾄ/ﾘｾｯﾄ原因を読み出す。	
(48) RXDIAG	受信パケットの診断符号を読み出す。	
(49) RXDTL	受信パケット内のデータ長を読み出す。	
(50) RXDATA	受信パケット内のデータを読み出す。	13-43

(4) 音声、BERT用関数

関数名	内容	ページ
(1) SOUND_ON	ヘッドホーン、ブザーから音を発生させる。	13-46
(2) SOUND_OFF	ヘッドホーン、ブザーから出ていた音を止める。	13-48
(3) TONE	ブザーから発生される音色を変える。	13-49
(4) VOLUME	ヘッドホーン、ブザーから出ていた音量を変える。	13-50
(5) SET_PRBS	BERTで使用する擬似ランダム・パターンを指定する。	13-51
(6) SET_WORD	BERTで使用するワード・パターンを指定する。	13-53
(7) BERT_ON	エラー・レート試験を行なう。	13-54

12.　Dチャネル・シミュレーション

12.1　共通関数

［表12-1］に共通関数の一覧を示します。

表 12 - 1　共通関数

(1)　INSERT	送信フレームの任意のオクテットの値を変える。
(2)　SET_FRLEN	送信するフレームのフレーム長を変える。
(3)　SET_CHANN	送受信するチャネルを指定する。
(4)　RECEIVE	タイムアウト／フレーム／キー入力／イベント受信待ち状態にする。
(5)　T_START TM_START	タイマを起動する。（分解能 1秒のタイマ起動） タイマを起動する。（分解能100m秒のタイマ起動）
(6)　T_STOP	タイマを停止する。
(7)　READ_TIMER	タイムアウトしたタイマのIDを得る。
(8)　SEND_EVENT	他チャネルにイベントを送る。
(9)　READ_EVENT	他チャネルから送られてきたイベントの内容を読み出す。
(10)　EXTRACT	受信フレームの任意のオクテットの値を読み出す。
(11)　RXFRLEN	受信フレーム・データ長を調べる。
(12)　WAIT	指定の時間プログラム停止する。
(13)　PH_ACT	レイヤ１を起動する。
(14)　PH_DEACT	レイヤ１を停止する。
(15)　READ_VAL	キー入力された数値を読み出す。

次ページ以降に各関数の説明をします。

(1) INSERT

呼出し形式　INSERT("NAME",n,DT)

概　要　送信フレームの任意のオクテット値を書き換えます。

引数の説明
"NAME"	:データの変更を行なうフレームのフレーム名。
n	:データの変更を行なうオクテット値。
DT	:データの変更値。(0～255 の範囲)

関数値
0	:正常終了。
-60	:引数エラー。
-61	:フレーム名エラー。

使用例　INSERT("SABME",2,H'81')

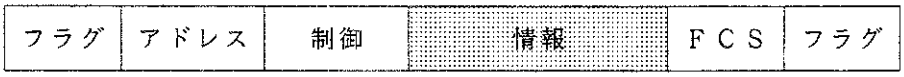
機能説明　引数で指定したフレーム名の任意のオクテットのデータを変更する関数です。

注意事項　一度、変更した値は、シミュレーションをストップするまで有効です。
トランスペアレント・モードとレイヤ2自動モードとでは変更できるデータの領域が違います。(下図参照)
下図で斜線部の領域が変更可能です。
また、変更位置は斜線部の左側から順に、1から数えたオクテットとします。

制限事項　引数で指定したフレーム名のフレームがない場合は、フレーム名エラーになります。
また、データのない場所を変更しようとすると、引数エラーになります。

① トランスペアレント・モード

② レイヤ2自動モード



(2) SET_FRLEN

呼出し形式　SET_FRLEN("NAME",LEN)

概　要　送信フレームのデータ長を変えます。

引数の説明　"NAME" :フレーム名。
LEN :変更するデータ長。
ﾄﾗﾝｽﾍﾟｱﾚﾝﾄ･ﾓｰﾄﾞ 時 : 1～512 。
レイヤ２自動実行モード時: 1～508 または1～509 。
（Max値は、レイヤ２のメッセージに依存する）

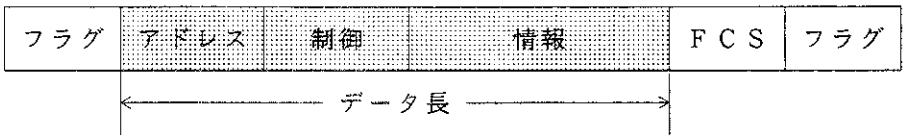
関数値　0 :正常終了。
-60 :引数エラー。
-61 :フレーム名エラー。

使用例　SET_FRLEN("SPL1",10)

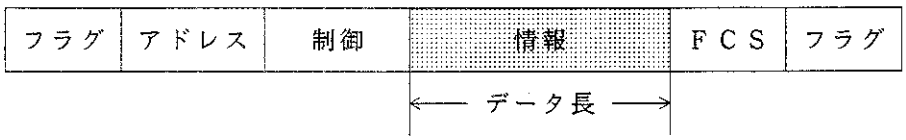
機能説明　引数で指定したフレーム名のデータ長を変更する関数です。

注意事項　一度フレーム長を変えると、それはシミュレーションを停止するまで有効です。トランスペアレントモードとレイヤ２自動実行モードでは、変更できるデータ長の意味が異なります。（下図参照）

① トランスペアレント・モード



② レイヤ２自動実行モード



(3) S E T _ C H A N N

呼出し形式　SET_CHANN(CHANN)

概　要　送受信するチャネルを指定します。

引数の説明　CHANN :チャネル番号
0...チャネルを選択しない状態
1...B1チャネル
2...B2チャネル
・
・
24...B24チャネル

関数値　0 :正常終了。
-57 :一次群インタフェース以外の宣言がされている。
-163 :指定したチャネルが存在しない。

使用例　SET_CHANN(1)

機能説明　使用するチャネルを指定する関数です。この関数を実行すると、引数で指定したチャネルでフレームの送受信を行なうことができます。この関数は、一次群でシミュレーションを行なう場合のみ有効です。基本インタフェースでシミュレーションを行なう場合、D チャネル固定で変更することはできません。
一次群インタフェースでシミュレーションを行なう場合は、B1～B24の任意のチャネルを使用できます。この時、何も指定が無いとB24チャネルを使用してシミュレーションを行ないます。

(4) RECEIVE

呼出し形式　RECEIVE(TIMER_ID)

概　要　タイムアウト／フレーム／イベント受信待ち状態にします。

引数の説明　TIMER_ID :タイマ番号。

関数値
0 :タイムアウトによる終了。
1 :フレーム受信による終了。
2 :キー入力による終了。
10 :D チャネルからのイベント受信による終了。
20 :B チャネルからのイベント受信による終了。

使用例
```
TM_ID=1
TM_SEC=10
T_START(TM_ID,TM_SEC)
RET=RECEIVE(TM_ID)
IF RET==0 THEN
   PRINT("TIME OUT\N")
END
```

機能説明　タイムアウト、フレーム受信、他チャネルからのイベント受信待ち状態になります。
上記のイベントを受信するとこの関数は終了しますが、そのとき関数値で終了理由を返します。
タイマは、本関数とT_START,T_STOP関数を用いて管理します。
タイマの起動は、T_START関数で行ない、タイムアウトしたそのイベントは本関数で受け取ります。そのとき、本関数の引数TINER_ID以外のタイマがタイムアウトした場合でもそれは受信されません。ただし、TINER_IDが0のときは例外で、すべてのタイマIDのタイムアウト・イベントを受信します。このときどのタイマがタイムアウトしたかはREAD_TIMER関数で知ることができます。タイマの停止は、T_STOP関数で行ないます。
すでに、フレームを受信している状態でRECEIVE関数が実行されると関数値として1 を返し終了します。このときRXで始まる受信フレーム読み出し関数が利用できます。
受信した次のフレームの内容を読み出したい場合は、再びRECEIVE関数を実行します。RECEIVE関数を実行するごとに受信したフレームの順に、その内容をRXで始まる関数を使用することにより読み出すことができます。
受信したフレームが無い場合や、次のフレームをまだ受信していない場合は、受信待ち状態になりフレームを受信した後、上記のようにフレーム内容を読み出せるようになります。
INPUT 関数が実行されていないときに、Command ラインから数値が入力されると、関数値2 を返します。この数値は、READ_VAL関数で読み出すことができます。（INPUT 関数参照）
他チャネルからのイベントを受信した場合には、関数値として上記の値を返します。受信した他チャネルからのメッセージはREAD_MESSAGE関数で読むことができます。

(5) T_START／TM_START

呼出し形式　T_START(TIMER_ID,SEC)
TM_START(TIMER_ID,SEC)

概　要　タイマを起動します。

引数の説明
TIMER_ID　:起動するタイマのタイマ番号。(1～16777215)
SEC　:タイムアウト値。
関数T_START では、1秒単位　(0～16777215)
関数TM_START では、100m秒単位　(0～16777215)

関数値
0　:正常終了。
-60　:引数エラー。
-120　:最大同時起動タイマ数オーバ。

使用例
```
T_START(201,1)
T202=202
S=2
T_START(T202,S)
```

機能説明　引数で指定したタイマ番号のタイマを起動する関数です。
この関数が実行されたときに、同じタイマ番号のタイマが既に起動されている場合は、そのタイマを停止し、新たにタイマを起動させます。

制限事項　タイムアウト値に 16777215 をこえる値が設定されると引数エラーになります。
最大同時起動タイマ数以上タイマが起動されると最大同時起動タイマ数オーバのエラーになります。
最大同時起動タイマ数は、関数T_START では 5個、関数TM_STARTでは20個まで可能です。

(6) T_STOP

呼出し形式　T_STOP(TIMER_ID)

概　要　タイマを停止します。

引数の説明　TIMER_ID　:停止するタイマのタイマ番号。(0～16777215)

関数値
0	:正常終了。
-60	:引数エラー。
-121	:起動しているタイマがない。

使用例
```
T_STOP(201)
T202=202
T_STOP(T202)
```

機能説明　引数で指定したタイマ番号のタイマを停止する関数です。

制限事項　引数で指定したタイマ番号のタイマが起動していない場合は、エラー・コードを関数値として返すだけで、他に何もしません。

(7) READ_TIMER

呼出し形式　READ_TIMER( )

概　要　タイムアウトしたタイマのIDを得ます。

引数の説明

関数値　0 ～16777215　:タイムアウトしたタイマの番号。
-121　:タイムアウトしたタイマが無い。

使用例

```
T_START(1,5)
T_START(2,10)
RET=RECEIVE(0)
IF RET==0 THEN
   TIMER=READ_TIMER( )
END
PPINT("TIMER=%D＼N",TIMER)
```

機能説明　タイムアウトしたタイマの番号を知ることができます。
タイマの使い方は、RECEIVE関数の説明を参考にして下さい。

(8) SEND_EVENT

呼出し形式　SEND_EVENT(DEST,MSG)

概　要　他チャネルにイベントを送ります。

引数の説明　DEST :10:Dチャネル。
:20:Bチャネル。
MSG :送出するイベント。（0～16777215）

関数値　0 :正常終了。
-60 :引数エラー。

使用例　SEND_EVENT(20,120)
（Bチャネルに120というメッセージを送る）

機能説明　他チャネルにメッセージを送る関数です。
送ったメッセージは、送られたチャネルでRECEIVE関数と
READ_EVENT関数を使用することにより読み出すことができます。

制限事項　送るメッセージは、0～16777215の数です。
それ以外が指定されると引数エラーとなります。

(9) READ_EVENT

呼出し形式　READ_EVENT( )

概　要　他チャネルから送られてきたイベントの内容を読み出します。

引数の説明

関数値　0 ～16777215　:イベント内容。
-156　:イベント未受信。

使用例

```
RET=RECEIVE(0)
IF RET==10 THEN
   MSG=READ_EVENT( )
   PRINT("MESSAGE=[%X]\N",MSG)
END
```

機能説明　他チャネルから送られてきたイベントの内容を知ることができます。
他チャネルからのイベントの送出るがあると、そのイベントの発生をRECEIVE関数により知ることができます。
RECEIVE関数の関数値により、他チャネルからのイベントの受信があったのを確認した後、本関数を実行するとイベント内容を知ることができます。

⑽ EXTRACT

呼出し形式　EXTRACT(n)

概　要　受信フレームの任意のオクテットの値を読み出します。

引数の説明　n　:読み出すデータのオクテット値。

関数値　0 ～255　:読み出したデータ。
-60　:引数エラー。
-80　:フレーム未受信エラー。

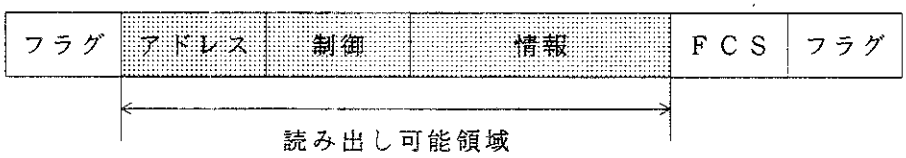
使用例
```
CF1=EXTRACT(3)
AA=4
CF2=EXTRACT(AA)
```

機能説明　受信したフレームの任意のオクテットの値を読み出す関数です。

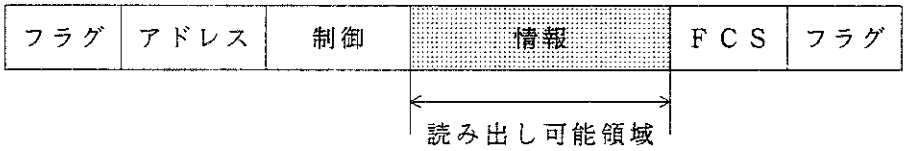
注意事項　トランスペアレント・モードとレイヤ２自動とでは読み出せるデータの領域が違います。（下図参照）
下図で斜線部の領域が読み出し可能領域です。また、読み出す位置の指定は斜線部の左側から順に、１から数えたオクテットで行ないます。

制限事項　引数で設定したオクテットのデータを関数値として返します。引数は、１～受信フレーム長までの値が設定可能です。それ以外を設定すると引数エラーになります。また、フレームを受信していないかRECEIVE関数を実行していない状態でこの関数が実行されるとフレーム未受信エラーになります。

① トランスペアレント・モード



② レイヤ２自動モード



(11) RXFRLEN

呼出し形式　RXFRLEN( )

概　要　受信フレーム・データ長を調べます。

引数の説明

関数値　0 ～65535　:受信フレームのデータ長。
-80　:フレーム未受信エラー。

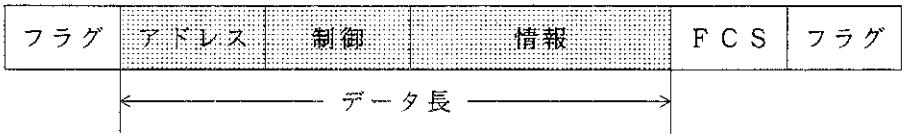
使用例　LENGTH=RXFRLEN( )

機能説明　受信フレームのデータ長を関数値として返します。

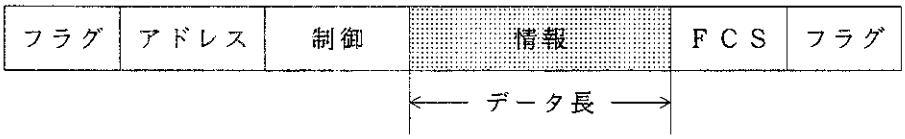
注意事項　トランスペアレント・モードとレイヤ2自動モードでは、データ長の定義が違います。（下図参照）

制限事項　この関数はフレームを受信していないか、またはフレームを受信していてもRECEIVE関数を実行していないと、フレーム未受信エラーとなります。

① トランスペアレント・モード



② レイヤ2自動モード



(12) WAIT

呼出し形式　WAIT(SEC)

概　要　指定の時間プログラムを停止します。

引数の説明　SEC　:ウエイトする時間。
100msec単位で、0～16777215の値が設定できます。

関数値　0　:正常終了。
-60　:引数エラー。

使用例
```
WAIT(10)
SEC=5
WAIT(SEC)
```

機能説明　指定した時間プログラムの実行を停止します。

(13) P H _ A C T

呼出し形式　PH_ACT( )

概　要　レイヤ1を起動させます。

引数の説明

関数値
0　:正常終了。
-56　:基本インタフェース以外の宣言がされている。
-160　:レイヤ1エラー。

使用例　PH_ACT( )

機能説明　レイヤ1の起動を行なう関数です。

注意事項　レイヤ1の起動に失敗すると、レイヤ1エラーを示す関数値が返ります。
この関数は、レイヤ1 を基本インタフェースに宣言しているときのみ有効です。

⑭ PH_DEACT

呼出し形式　PH_DEACT( )

概　要　レイヤ1を停止させます。

引数の説明

関数値
```
0      :正常終了。
-53    :NTモードエラー。
-56    :基本インタフェース以外の宣言がされている。
-160   :レイヤ1エラー。
```

使用例
```
PH_DEACT( )
```

機能説明　レイヤ1を停止する関数です。

制限事項　NTモード以外では、実行できません。
この関数は、レイヤ1 を基本インタフェースに宣言しているときのみ有効です。

⑮ READ_VAL

呼出し形式　READ_VAL( )

概　要　キー入力された数値を読み出します。

引数の説明

関数値
```
0 ～1677215    :キー入力された数値
-175           :キー入力されていない。
```

使用例
```
RET=RECEIVE(0)
IF RET==2 THEN
   VAL=READ_VAL( )
   PRINT("KEY VALUE=%D＼N",VAL)
END
```

機能説明　Command ラインから入力された数値を読み出します。Command ラインから入力された数値を読み出す関数としてINPUT 関数がありますが、INPUT 関数によりキー入力受信待ち状態になっているときは、INPUT 関数が優先されます。従って、RECEIVE 関数にキー入力の情報は渡されません。
INPUT 関数による受信待ち状態か、RECEIVE 関数による受信待ち状態かは、画面下部に表示されているアイコン表示で識別します。
（それぞれ、"INPUT","RCV?"と表示されます。）

12.2 トランスペアレント・モード用関数

[表12-2] にトランスペアレント・モード用関数の一覧を示します。

表 12 - 2 トランスペアレント・モード用関数 (1/2)

フレーム送信関連		
(1)	SENDT	フレームを送信する。（メッセージ・データをそのまま送出）
(2)	SENDF	フレームを送信する。(N(S),N(R),P/Fを挿入して送出)
(3)	INCVS	送信状態変数 V(S) を＋1 する。
(4)	INCVR	受信状態変数 V(R) を＋1 する。
(5)	SETVS	送信状態変数 V(S) の値を設定する。
(6)	SETVR	受信状態変数 V(R) の値を設定する。
(7)	INS_SAPI	送信フレームのSAPI値を書き換える。
(8)	INS_TEI	送信フレームのTEI 値を書き換える。
(9)	INS_CR	送信フレームのC/R ビット値を書き換える。
(10)	INS_NR	送信フレームのN(R)値を書き換える。
(11)	INS_NS	送信フレームのN(S)値を書き換える。
(12)	INS_PF	送信フレームのP/F ビット値を書き換える。
(13)	INS_TYPE	送信フレームのフレーム種別を書き換える。
(14)	INS_CF1	送信フレーム制御フィールドの第１オクテットの値を書き換える。
(15)	INS_CF2	送信フレーム制御フィールドの第２オクテットの値を書き換える。
(16)	INS_FRCF1	送信FRMRフレーム情報フィールド内の第１オクテット値を書き換える。
(17)	INS_FRCF2	送信FRMRフレーム情報フィールド内の第２オクテット値を書き換える。
(18)	INS_FRVS	送信FRMRフレーム情報フィールド内の V(S) 値を書き換える。
(19)	INS_FRVR	送信FRMRフレーム情報フィールド内の V(R) 値を書き換える。
(20)	INS_FRCR	送信FRMRフレーム情報フィールド内のC/R ビット値を書き換える。
(21)	INS_FRWXYZ	送信FRMRフレーム情報フィールド内のWXYZビット値を書き換える。

表 12 - 2 トランスペアレント・モード用関数 (2/2)

フレーム送信関連		
(22)	RXSAPI	受信フレームのSAPI値を読み出す。
(23)	RXTEI	受信フレームのTEI 値を読み出す。
(24)	RXCR	受信フレームのC/R ビット値を読み出す。
(25)	RXNR	受信フレームのN(R)値を読み出す。
(26)	RXNS	受信フレームのN(S)値を読み出す。
(27)	RXPF	受信フレームのP/F ビット値を読み出す。
(28)	RXTYPE	受信フレームのフレーム種別を読み出す。
(29)	RXCF1	受信フレーム制御フィールドの第1オクテットの値を読み出す。
(30)	RXCF2	受信フレーム制御フィールドの第2オクテットの値を読み出す。
(31)	RXFRCF1	受信FRMRフレーム情報フィールド内の第1オクテット値を読み出す。
(32)	RXFRCF2	受信FRMRフレーム情報フィールド内の第2オクテット値を読み出す。
(33)	RXFRVS	受信FRMRフレーム情報フィールド内の V(S) 値を読み出す。
(34)	RXFRVR	受信FRMRフレーム情報フィールド内の V(R) 値を読み出す。
(35)	RXFRCR	受信FRMRフレーム情報フィールド内の C/Rビット値を読み出す。
(36)	RXFRWXYZ	受信FRMRフレーム情報フィールド内の WXYZ ビット値を読み出す。

次ページ以降に各関数の説明をします。

(1) SENDT

呼出し形式　SENDT("NAME")

概　要　フレームを送信します。（メッセージ・データをそのまま送出）

引数の説明　"NAME" :フレーム名。

関数値　0　:正常終了。
-50　:トランスペアレント・モード・エラー。
-61　:フレーム名エラー。

使用例　SENDT("SPL1")

機能説明　トランスペアレント・モードでのフレームの送出を行なう関数です。引数で指定した名前のフレームをそのまま送出します。

注意事項　SENDF関数では、内部の状態変数 V(S),V(R)を N(S),N(R)としてフレーム中に挿入して送出しますが、本関数は一切加工せずに回線上に送出します。

制限事項　レイヤ2自動モードでは、実行できません。

(2) SENDF

呼出し形式　SENDF("NAME",PF)

概　要　フレームを送信します。(N(S),N(R),P/Fを挿入して送出)

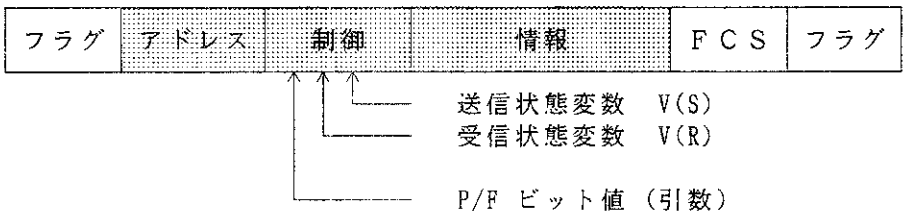
引数の説明　"NAME" :送出するフレーム名。
PF :送出するフレームのP/F ビット値。

関数値　0 :正常終了。
-50 :トランスペアレント・モード・エラー。
-61 :フレーム名エラー。
-110 :送出フレーム長エラー。

使用例
```
SENDF("ID_REQ",0)
PF= 1
SENDF("SABME",1)
```

機能説明　トランスペアレント・モードでのフレームの送出を行なう関数です。
引数で指定されたフレーム名のフレームを送出しますが、その際、引数で指定されたP/F ビット値や、送信シーケンス番号N(S)、受信シーケンス番号N(R)を付加して送信します。
送信状態変数V(S)は、INCVS 関数やSETVS 関数により変更が可能です。
また、受信状態変数V(R)も、同様にINCVR 関数やSETVR 関数により変更が可能です。
SENDF 関数によりフレーム送信する際に付加されるデータを、下図に示します。

制限事項　レイヤ2自動モードでは、実行できません。



(3) INCVS

呼出し形式　INCVS( )

概　要　送信状態変数 V(S) を＋１します。

引数の説明

関数値　0　:正常終了。
-50　:トランスペアレント・モード・エラー。

使用例　INCVS( )

機能説明　送信状態変数V(S)の値を＋１更新する関数です。
この値は、番号制情報フレーム(Iフレーム）を送出する時にのみ
N(S)値としてフレームに付加されます。

制限事項　この関数は、レイヤ２自動モードでは実行できません。

(4) INCVR

呼出し形式　INCVR( )

概　要　受信状態変数 V(R) を＋１します。

引数の説明

関数値　0　:正常終了。
-50　:トランスペアレント・モード・エラー。

使用例　INCVR( )

機能説明　受信状態変数V(R)の値を＋１更新する関数です。
この値は、番号制情報フレーム(Iフレーム）と番号制監視フレーム(Sフレーム）を送出する時にN(R)値として、送信フレームに付加されます。

制限事項　この関数は、レイヤ２自動モードでは実行できません。

(5) SETVS

呼出し形式　SETVS(VS)

概　要　送信状態変数 V(S) の値を設定します。

引数の説明　VS　:設定するV(S)値。
0≦VS≦127

関数値　0　:正常終了。
-50　:トランスペアレント・モード・エラー。
-60　:引数エラー。

使用例
```
SETVS(5)
VS=6
SETVS(VS)
```

機能説明　送信状態変数V(S)の値を設定する関数です。
この値は、番号制情報フレーム(Iフレーム)を送出するときにのみN(S)値として送信フレームに付加されます。

制限事項　この関数は、レイヤ2自動モードでは実行できません。

(6) ＳＥＴＶＲ

呼出し形式　SETVR(VR)

概　要　受信状態変数 V(R) の値を設定します。

引数の説明　VR　:設定するV(R)値。
0≦VR≦127

関数値　0　:正常終了。
-50　:トランスペアレント・モード・エラー。
-60　:引数エラー。

使用例
```
SETVR(5)
VR=6
SETVR(VR)
```

機能説明　受信状態変数V(R)の値を設定する関数です。
この値は、番号制情報フレーム(Iフレーム）と番号制監視フレーム(Sフレーム）を送出するときにのみ、N(R)値として送信フレームに付加されます。

制限事項　この関数は、レイヤ２自動モードでは実行できません。

(7)～(21) LAPDパラメータ挿入関数群

呼出し形式

	[関数名]	[挿入するパラメータ]
(7)	INS_SAPI("NAME",SAPI)	SAPI値
(8)	INS_TEI("NAME",TEI)	TEI 値
(9)	INS_CR("NAME",CR)	C/R ビット値
(10)	INS_NR("NAME",NR)	N(R)
(11)	INS_NS("NAME",NS)	N(S)
(12)	INS_PF("NAME",PF)	P/F ビット値
(13)	INS_TYPE("NAME",TYPE)	フレーム種別
(14)	INS_CF1("NAME",CF1)	制御ﾌｨｰﾙﾄﾞ 第1 ｵｸﾃｯﾄ 値
(15)	INS_CF2("NAME",CF2)	制御ﾌｨｰﾙﾄﾞ 第2 ｵｸﾃｯﾄ 値
(16)	INS_FRCF1("NAME",FRCF1)	FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内第1ｵｸﾃｯﾄ値
(17)	INS_FRCF2("NAME",FRCF2)	FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内第2ｵｸﾃｯﾄ値
(18)	INS_FRVS("NAME",FRVS)	FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内V(S)値
(19)	INS_FRVR("NAME",FRVR)	FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内V(R)値
(20)	INS_FRCR("NAME",FRCR)	FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内C/R ﾋﾞｯﾄ 値
(21)	INS_FRWXYZ("NAME",FRWXYZ)	FRMRﾌﾚｰﾑ情報ﾌｨｰﾙﾄﾞ内WXYZﾋﾞｯﾄ 値

概　要　　送出フレームの任意のLAPDパラメータを書き換えます。

引数の説明

NAME	:フレーム名
SAPI	:SAPI値
TEI	:TEI 値
CR	:C/R ビット値
NR	:N(R)
NS	:N(S)
PF	:P/F ビット値
TYPE	:フレーム種別
CF1	:制御フィールド第1 オクテット値
CF2	:制御フィールド第2 オクテット値
FRCF1	:FRMRフレーム情報フィールド内第1 オクテット値
FRCF2	:FRMRフレーム情報フィールド内第2 オクテット値
FRVS	:FRMRフレーム情報フィールド内V(S)値
FRVR	:FRMRフレーム情報フィールド内V(R)値
FRCR	:FRMRフレーム情報フィールド内C/R ビット値
FRWXYZ	:FRMRフレーム情報フィールド内WXYZビット値

関数値

0	:正常終了
-50	:トランスペアレント・モードエラー
-60	:引数エラー
-61	:フレーム名エラー
-85	:書き換えられるメッセージが不適切

使用例

```
INS_TEI("SABME",64)
SENDF("SABME",1)
```

機能説明

メッセージ中の任意のLAPDパラメータを挿入する関数です。
本関数群は、メッセージの先頭から内容を解釈していき挿入するLAPDパラメータの位置を探していきます。
挿入するLAPDパラメータの位置が発見されるとそこを引数で指定された値に書き換えます。
このとき挿入するLAPDパラメータの位置が発見されない場合には、"書き換えられるメッセージが不適切である"という意味の関数値を返します。

補足説明

INS_TYPE("NAME",TYPE) 関数の引数TYPEとフレーム種別は、以下の対応となっています。

フレーム種別	TYPE		フレーム種別	TYPE	
	10進	16進		10進	16進
I	0	00	UI	3	03
RR	1	01	DISC	67	43
RNR	5	05	UA	99	63
REJ	9	09	FRMR	135	87
SABME	111	6F	XID	175	AF
DM	15	0F			

制限事項

本関数群は、トランスペアレント・モードで実行する関数で、レイヤ2自動モードでは実行できません。

㉒～㊱ LAPDパラメータ読み出し関数群

呼出し形式	[関数名]	[読み出すパラメータ]
	㉒ RXSAPI( )	SAPI値
	㉓ RXTEI( )	TEI 値
	㉔ RXCR( )	C/R ビット値
	㉕ RXNR( )	N(R)値
	㉖ RXNS( )	N(S)値
	㉗ RXPF( )	P/F ビット値
	㉘ RXTYPE( )	フレーム種別
	㉙ RXCF1( )	制御フィールド第1 オクテット値
	㉚ RXCF2( )	制御フィールド第2 オクテット値
	㉛ RXFRCF1( )	FRMRフレーム情報ﾌｨｰﾙﾄﾞ内第1 ｵｸﾃｯﾄ値
	㉜ RXFRCF2( )	FRMRフレーム情報ﾌｨｰﾙﾄﾞ内第2 ｵｸﾃｯﾄ値
	㉝ RXFRVS( )	FRMRフレーム情報ﾌｨｰﾙﾄﾞ内V(S)値
	㉞ RXFRVR( )	FRMRフレーム情報ﾌｨｰﾙﾄﾞ内V(R)値
	㉟ RXFRCR( )	FRMRフレーム情報ﾌｨｰﾙﾄﾞ内C/R ビット値
	㊱ RXFRWXYZ( )	FRMRフレーム情報ﾌｨｰﾙﾄﾞ内WXYZビット値

概　要　　受信フレームの任意のLAPDパラメータを読み出す関数群です。

引数の説明

関数値

0 ～255	:受信フレームのパラメータ。
-50	:トランスペアレント・モード・エラー。
-80	:フレーム未受信エラー。
-81	:受信フレームがFRMRフレームでない。 (㉛～㊱の関数のときのみ)
-82	:読み出すパラメータが存在しない。

使用例

```
RECEIVE(0)
TEI=RXTEI( )
PRINT("TEI=%D ＼N",TEI)
```

機能説明　受信フレームのLAPDパラメータを読み出す関数群です。
受信したフレーム順に読み出すことができます。
受信したフレームを読み出したい場合は、RECEIVE 関数を実行し、受信フレームを内部に取り込んでから本関数群を実行します。次の受信フレームを読み出したい場合は、再びRECEIVE 関数を実行します。このとき受信フレームが無い場合は、受信待ち状態になります。

注意事項　受信したフレームが無い状態やRECEIVE 関数を実行していない状態で本関数が実行されるとフレーム未受信エラーとなります。
また本関数群は、受信したフレームを先頭から解釈していきLAPDパラメータのある位置を探していきます。LAPDパラメータのある位置が発見されるとその値を関数値として返し、終了します。
LAPDパラメータが見つからない場合には、”読み出すパラメータが存在しない”という意味の関数値を返します。

補足説明　㉛～㊱のRXFRで始まるLAPDパラメータ読み出し関数において受信フレームがFRMRフレームでない場合、”受信フレームがFRMRフレームでない”という意味の関数値を返します。
RXTYPE( ) 関数で返される関数値とフレーム種別は、以下の対応となっています。

フレーム種別	関数値		フレーム種別	関数値	
	10進	16進		10進	16進
I	0	00	UI	3	03
RR	1	01	DISC	67	43
RNR	5	05	UA	99	63
REJ	9	09	FRMR	135	87
SABME	111	6F	XID	175	AF
DM	15	0F			

制限事項　本関数はトランスペアレント・モードで実行する関数でレイヤ2 自動モードでは実行できません。

12.3 レイヤ2自動モード用関数

[表12-3] にレイヤ2自動モード用関数の一覧を示します。

表 12 - 3 レイヤ2自動モード用関数 (1/3)

フレーム送信関連	
(1) SENDI	Iフレームを送信する。
(2) SENDUI	UIフレームを送信する。
(3) SENDXIDC	XID コマンドを送信する。
(4) SENDXIDR	XID レスポンスを送信する。
(5) SENDPKT	パケットを送信する。
(6) INCPS	送信順序番号 P(S) を+1にする。
(7) INCPR	受信順序番号 P(R) を+1にする。
(8) SETPS	送信順序番号 P(S) の値を設定する。
(9) SETPR	受信順序番号 P(R) の値を設定する。
(10) INS_PD	送信フレームのプロトコル識別子を書き換える。
(11) INS_CRL	送信フレームの呼番号長を書き換える。
(12) INS_CRF	送信フレームの呼番号フラグ値を書き換える。
(13) INS_CRV	送信フレームの呼番号値を書き換える。
(14) INS_MSG	送信フレームのメッセージ種別を書き換える。
(15) INS_INFO	送信フレームの情報要素群フィールドを書き換える。
(16) INS_CS_VAL	送信フレームの理由表示値を書き換える。
(17) INS_GFI	送信パケットのゼネラルフォーマット識別子の値を書き換える。
(18) INS_Q	送信パケットのQ ビット値を書き換える。
(19) INS_D	送信パケットのD ビット値を書き換える。
(20) INS_LCGN	送信パケットの論理チャネルグループ番号を書き換える。
(21) INS_LCN	送信パケットの論理チャネル番号を書き換える。
(22) INS_TYP	送信パケットのパケットタイプ識別子を書き換える。
(23) INS_PR	送信パケットの受信順序番号 P(R) を書き換える。
(24) INS_PS	送信パケットの送信順序番号 P(S) を書き換える。
(25) INS_M	送信パケットのモアデータ表示値を書き換える。
(26) INS_CLL	送信パケットの発呼ユーザアドレス長を書き換える。
(27) INS_CDL	送信パケットの着呼ユーザアドレス長を書き換える。
(28) INS_DA	送信パケットの着呼ユーザアドレスを書き換える。
(29) INS_SA	送信パケットの発呼ユーザアドレスを書き換える。
(30) INS_FL	送信パケットのファシリティ長を書き換える。
(31) INS_F	送信パケットのファシリティを書き換える。
(32) INS_CAUSE	送信パケットの切断／リスタート／リセット原因を書き換える。
(33) INS_DIAG	送信パケットの診断符号を書き換える。
(34) SETDTL	送信パケット内のデータ長を書き換える。
(35) INS_DATA	送信パケット内のデータを書き換える。

表 12 - 3 レイヤ2自動モード用関数 (2/3)

TEI 管理手順関連		
(36)	REQ_TEI	TEI 割当手順を起動する。 (TEモード)
(37)	CHKREQ_TEI	TEI チェック手順を行なう。 (NTモード)
(38)	REMOVE_TEI	TEI 解除手順を行なう。 (NTモード)
(39)	VERIFY_TEI	TEI 検証手順を行なう。 (TEモード)
リンク,SAPI,TEI 管理関連		
(40)	LINKON	リンクの設定を行なう。
(41)	LINKOFF	リンクの解放を行なう。
(42)	WAIT_LINK	相手局からのリンク設定待ち状態にする。
(43)	L_STATUS	リンクが設定されているかを調べる。
(44)	SET_BUSY	自局をビジー状態にする。
(45)	REL_BUSY	自局のビジー状態を解除する。
(46)	PROHIBIT_L	新たなリンクの設定を禁止する。
(47)	PERMIT_L	リンク設定不許可状態を解除する。
(48)	REG_TEI	TEI 値を本器に登録し、そのTEI 値のフレームを受信可能にする。
(49)	REL_TEI	TEI 値を解除し、そのTEI 値のフレームを受信しないようにする。
(50)	NEXT_TEI	新たなTEI 値を使用することを宣言する。 (TEモード)
(51)	ACT_SAPI	フレーム送出時に使用されるSAPI値を設定する。
(52)	ACT_TEI	フレーム送出時に使用されるTEI 値を設定する。
(53)	ACT_TEI	フレーム送出時に使用されるリンク番号を設定する。
(54)	LOCK_SAPI	SAPI値が自動で設定変更されるのを禁止する。
(55)	LOCK_TEI	TEI 値が自動で設定変更されるのを禁止する。
(56)	LOCK_LINK	リンク番号が自動で設定変更されるのを禁止する。
(57)	FLEX_SAPI	SAPI値が自動で設定変更されるようにする。
(58)	FLEX_TEI	TEI 値が自動で設定変更されるようにする。
(59)	FLEX_LINK	リンク番号が自動で設定変更されるようにする。
(60)	GET_SAPI	フレーム送出時に使用されるSAPI値を調べる。
(61)	GET_TEI	フレーム送出時に使用されるTEI 値を調べる。
(62)	GET_LINK	フレーム送出時に使用されるリンク番号を調べる。
(63)	SEE_LINK	現在設定されているリンク番号を調べる。

表 12 - 3 レイヤ2自動モード用関数 (3/3)

フレーム受信関連	
(64) RXSAPI	受信フレームのSAPI値を読み出す。
(65) RXTEI	受信フレームのTEI 値を読み出す。
(66) RXTYPE	受信フレームのフレーム種別を読み出す。
(67) RXPD	受信フレームのプロトコル識別子を読み出す。
(68) RXCRL	受信フレームの呼番号長を読み出す。
(69) RXCRF	受信フレームの呼番号フラグ値を読み出す。
(70) RXCRV	受信フレームの呼番号値を読み出す。
(71) RXMSG	受信フレームのメッセージ種別を読み出す。
(72) RXINFO_NUM	受信フレームの情報要素数を読み出す。
(73) RXINFO_ELM	受信フレームの情報要素識別子を読み出す。
(74) RXINFO_LEN	受信フレームの情報要素内容長を読み出す。
(75) RXINFO_VAL	受信フレームの情報内容を読み出す。
(76) RXINFO_POS	受信フレームの情報要素識別子の位置を調べる。
(77) RXCS_VAL	受信フレームの理由表示値を読み出す。
(78) RXCHAN_NUM	受信フレームのチャネル番号を読み出す。
(79) RXGFI	受信パケットのゼネラルフォーマット識別子の値を読み出す。
(80) RXQ	受信パケットのQ ビット値を読み出す。
(81) RXD	受信パケットのD ビット値を読み出す。
(82) RXLCGN	受信パケットの論理チャネルグループ番号を読み出す。
(83) RXLCN	受信パケットの論理チャネル番号を読み出す。
(84) RXTYP	受信パケットのパケットタイプ識別子を読み出す。
(85) RXPR	受信パケットの受信順序番号 P(R) を読み出す。
(86) RXPS	受信パケットの送信順序番号 P(S) を読み出す。
(87) RXM	受信パケットのモアデータ表示値を読み出す。
(88) RXCLL	受信パケットの発呼ユーザアドレス長を読み出す。
(89) RXCDL	受信パケットの着呼ユーザアドレス長を読み出す。
(90) RXDA	受信パケットの着呼ユーザアドレスを読み出す。
(91) RXSA	受信パケットの発呼ユーザアドレスを読み出す。
(92) RXFL	受信パケットのファシリティ長を読み出す。
(93) RXF	受信パケットのファシリティを読み出す。
(94) RXCAUSE	受信パケットの切断／リスタート／リセット原因を読み出す。
(95) RXDIAG	受信パケットの診断符号を読み出す。
(96) RXDTL	受信パケット内のデータ長を読み出す。
(97) RXDATA	受信パケット内のデータを読み出す。

次ページ以降に各関数の説明をします。

(1) SENDI

呼出し形式　SENDI("NAME")

概　要　Iフレームを送信します。

引数の説明　"NAME" :送出するフレーム名。

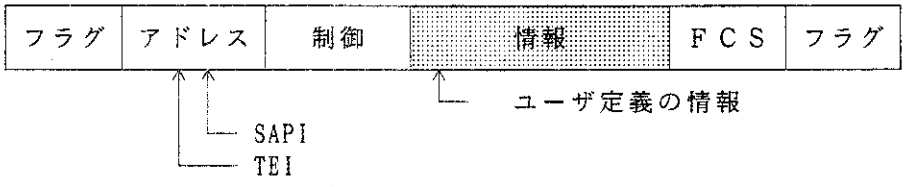
関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-61　:フレーム名エラー。
-70　:SAPIエラー。
-71　:TEI エラー。
-72　:リンク未設定エラー。
-110　:送出フレーム長エラー。
-200　:その他のエラー。

使用例　SENDI("SETUP")

機能説明　レイヤ2自動モードで、Iフレームの送出を行なう関数です。
引数で指定されたフレーム名のIフレームを送出しますが、その際、アドレスには、あらかじめ設定されたSAPI値とTEI 値を使います。（下図参照）

注意事項　Iフレームを送出するためには、まずリンクの設定を行なわなければなりません。
リンクの設定は LINKON 関数を用いて行なうか、相手局からのリンク設定要求を受け付けて行ないます。
リンク設定が行なわれていないSAPI, TEI ペアを用いてこの関数が実行されると、リンク未設定エラーになります。
Iフレームは一番最近リンクが設定されたSAPI, TEI ペアを用いて通常は送出されますが、一度設定したSAPI, TEI ペアで固定にして変更したくない場合は、それぞれLOCK_SAPI, LOCK_TEI 関数を用います。

制限事項　また、以前に設定したリンクについてのフレームを送出したい場合には、ACT_SAPI, ACT_TEI 関数を用いて正しく設定して下さい。
SAPI値に0、16、63以外の値が設定された状態で、この関数が実行されるとSAPIエラーになります。
TEI 値に未登録なTEI が設定された状態で、この関数が実行されるとTEI エラーになります。
TEI 値の登録はREG_TEI 関数を用いて行なうことができます。
引数で指定されたフレーム名のフレームがない場合や、送信フレームのフレーム長が短い場合にはエラーになります。
この関数は、トランスペアレント・モードでは使用できません。



(2) SENDUI

呼出し形式　SENDUI("NAME")

概　要　UIフレームを送信します。

引数の説明　"NAME" :送出するフレーム名。

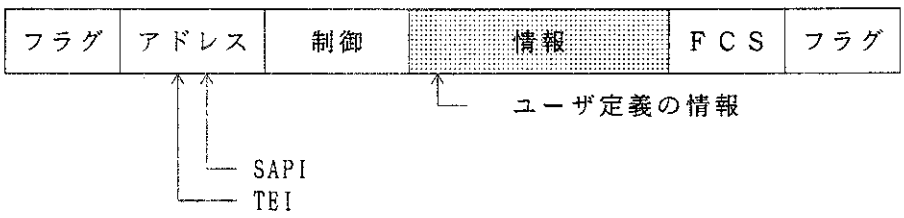
関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-61　:フレーム名エラー。
-70　:SAPIエラー。
-71　:TEI エラー。
-110　:送出フレーム長エラー。

使用例　SENDUI ("IDCHK")

機能説明　レイヤ2自動モードで、UIフレームの送出を行なう関数です。
引数で指定されたフレーム名のUIフレームを送出しますが、その際、アドレスにはあらかじめ設定されたSAPI値とTEI 値を使います。
（下図参照）

注意事項　SAPI値とTEI 値の初期値は、それぞれ0 、127 です。
SAPI値とTEI 値の設定には、それぞれACT_SAPI, ACT_TEI 関数を用いて行ないます。
ただし、TEI 値の設定は、TEI の割当手順が行なわれた場合や、リンクの設定が行なわれた場合にも生じます。
また、SAPI値の設定もリンクの設定が行なわれた場合に生じます。
新たなSAPI、TEI の設定を禁止したい場合は、それぞれLOCK_SAPI, LOCK_TEI 関数を用いて行ないます。

制限事項　SAPI値に0 、16、63以外の値が設定された状態で、この関数が実行されるとSAPIエラーになります。
TEI 値に未登録なTEI が設定された状態で、この関数が実行されるとTEI エラーになります。
TEI 値の登録はREG_TEI 関数を用いて行なうことができます。
引数で指定されたフレーム名のフレームがない場合や、送信フレームのフレーム長が短い場合にはエラーになります。
この関数は、トランスペアレント・モードでは使用できません。



(3) SENDXIDC

呼出し形式　SENDXIDC("NAME")

概　要　XID コマンドを送信します。

引数の説明　"NAME" :送出するフレーム名。

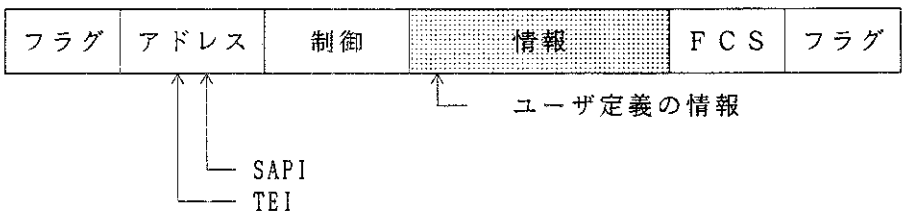
関数値
0	:正常終了。
-51	:レイヤ2自動モード・エラー。
-61	:フレーム名エラー。
-70	:SAPIエラー。
-71	:TEI エラー。
-110	:送出フレーム長エラー。

使用例　SENDXIDC("XID1")

機能説明　レイヤ2自動モードで、XID コマンドの送出を行なう関数です。
引数で指定されたフレーム名のXID コマンドを送出しますが、その際、アドレスにはあらかじめ設定されたSAPI値とTEI 値を使います。（下図参照）

注意事項　SAPI値とTEI 値の初期値は、それぞれ0 、127 です。
SAPI値とTEI 値の設定には、それぞれACT_SAPI, ACI_TEI 関数を用いて行ないます。
ただし、TEI 値の設定は、TEI の割当手順が行なわれた場合や、リンクの設定が行なわれた場合にも生じます。
また、SAPI値の設定もリンクの設定が行なわれた場合に生じます。
新たなSAPI, TEI の設定を禁止したい場合は、それぞれLOCK_SAPI, LOCK_TEI 関数を用いて行ないます。

制限事項　SAPI値に0 、16、63以外の値が設定された状態で、この関数が実行されるとSAPIエラーになります。
TEI 値に未登録なTEI が設定された状態で、この関数が実行されるとTEI エラーになります。
TEI 値の登録はREG_TEI 関数を用いて行なうことができます。
引数で指定されたフレーム名のフレームがない場合や、送信フレームのフレーム長が短い場合にはエラーになります。
この関数は、トランスペアレント・モードでは使用できません。



(4) SENDXIDR

呼出し形式　SENDXIDR("NAME")

概　要　XID レスポンス送信します。

引数の説明　"NAME"　:送出するフレーム名。

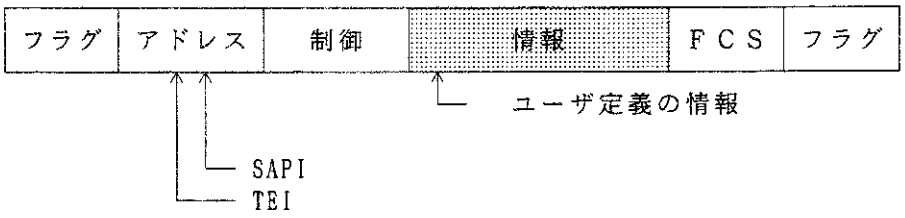
関数値
0	:正常終了。
-51	:レイヤ2自動モード・エラー。
-61	:フレーム名エラー。
-70	:SAPIエラー。
-71	:TEI エラー。
-110	:送出フレーム長エラー。

使用例　SENDXIDR("XID2")

機能説明　レイヤ2自動モードで、XIR レスポンスの送出を行なう関数です。引数で指定されたフレーム名のXID レスポンスを送出しますが、その際、アドレスにはあらかじめ設定されたSAPI値とTEI 値を使います。（下図参照）

注意事項　SAPI値とTEI 値の初期値は、それぞれ0 、127 です。
SAPI値とTEI 値の設定には、それぞれACT_SAPI, ACT_TEI 関数を用いて行ないます。
ただし、TEI 値の設定は、TEI の割当手順が行なわれた場合や、リンクの設定が行なわれた場合にも生じます。
また、SAPI値の設定もリンクの設定が行なわれた場合に生じます。
新たなSAPI, TEI の設定を禁止したい場合は、それぞれLOCK_SAPI, LOCK_TEI 関数を用いて行ないます。

制限事項　SAPI値に0 、16、63以外の値が設定された状態で、この関数が実行されるとSAPIエラーになります。
TEI 値に未登録なTEI が設定された状態で、この関数が実行されるとTEI エラーになります。
TEI 値の登録はREG_TEI 関数を用いて行なうことができます。
引数で指定されたフレーム名のフレームがない場合や、送信フレームのフレーム長が短い場合にはエラーになります。
この関数は、トランスペアレント・モードでは使用できません。



(5) SENDPKT

呼出し形式　SENDPKT("NAME",LCGN,LCN)

概　要　パケットを送信します。

引数の説明
"NAME" :フレーム名。
LCGN :論理チャネルグループ番号。(0～15)
LCN :論理チャネル番号。(0～255)

関数値
0 :正常終了。
-51 :レイヤ2自動モード・エラー。
-60 :引数エラー。
-61 :フレーム名エラー。
-70 :SAPIエラー。
-71 :TEI エラー。
-72 :リンク未設定エラー。
-110 :送出フレーム長エラー。

使用例
```
PH_ACT( )
LINKON( )
LCGN=0
LCN=1
SENDPKT("CR",LCGN,LCN)
```

機能説明　レイヤ2自動モードで、パケットフレームを送出する関数です。
引数で指定した名前のメッセージをIフレームとして送出します。
このとき引数で指定したLCGNとLCN を送出するメッセージに挿入します。また、送出するメッセージのタイプがDTパケットのときは、引数LCGN,LCNで指定された論理チャネルのP(R),P(S) が挿入されます。同様、RR,RNRパケットの時も指定された論理チャネルのP(R)がメッセージ中に挿入されます。
メッセージは、あらかじめメッセージ・ビルダで作成しておく必要があります。作成したメッセージに名前は必ず付けて下さい。
本関数でフレームを送出するためにはレイヤ2リンクが設定されていなければなりません。リンクの設定はLINKON( ) 関数を用いて行なうか、相手局からのリンク設定要求を受け付けて行ないます。

注意事項　本関数でフレームが送出されない場合以下の点をチェックして下さい。
1. レイヤ1は起動しているか。
2. レイヤ2リンクは設定されているか。
3. メッセージ・ビルダで引数で指定した名前のメッセージを作成してあるか。

(6) I N C P S

呼出し形式　INCPS(LCGN,LCN)

概　要　送信順序番号 P(S) を＋1 します。

引数の説明　LCGN :論理チャネルグループ番号。(0～15)
LCN :論理チャネル番号。(0～255)

関数値　0 :正常終了。
-51 :レイヤ2自動モード・エラー。
-60 :引数エラー。

使用例
```
INCPS(0,1)
SENDPKT("DT",0,1)
```

機能説明　送信順序番号P(S)を＋1 更新する関数でする。
引数で指定した論理チャネル(LCGN とLCN)のP(S)値を変更します。
この値は、DTパケットを送出するときにメッセージ中に挿入されます。

制限事項　本関数は、トランスペアレント・モードでは実行できません。

(7) INCPR

呼出し形式　INCPR(LCGN,LCN)

概　要　受信順序番号 P(R) を+1 します。

引数の説明　LCGN :論理チャネルグループ番号。(0～15)
LCN :論理チャネル番号。(0～255)

関数値　0 :正常終了。
-51 :レイヤ2自動モード・エラー。
-60 :引数エラー。

使用例
```
INCPR(0,1)
SENDPKT("DT",0,1)
```

機能説明　受信順序番号P(R)を+1 更新する関数です。
引数で指定した論理チャネル（LCGNとLCN)のP(R)値を変更します。
この値は、DT,RR,RNR パケットを送出するときにメッセージ中に挿入されます。

制限事項　本関数は、トランスペアレント・モードでは実行できません。

(8) SETPS

呼出し形式　SETPS(LCGN, LCN, PS)

概　要　送信順序番号 P(S) の値を設定します。

引数の説明
LCGN :論理チャネルグループ番号。(0～15)
LCN :論理チャネル番号。(0～255)
PS :送信順序番号。(0～127)

関数値
0 :正常終了。
-51 :レイヤ2自動モード・エラー。
-60 :引数エラー。

使用例
```
SETPS(0,1,7)
SENDPKT("DT",0,1)
```

機能説明　送信順序番号P(S)の値を設定する関数です。
引数で指定した論理チャネル（LCGNとLCN)のP(S)値を変更します。
この値は、DTパケットを送出するときのみメッセージ中に挿入されます。

制限事項　本関数は、トランスペアレント・モードでは実行できません。

(9) SETPR

呼出し形式　SETPR(LCGN,LCN,PR)

概　要　受信順序番号 P(R) の値を設定します。

引数の説明

LCGN	:論理チャネルグループ番号。(0～15)
LCN	:論理チャネル番号。(0～255)
PR	:受信順序番号。(0～127)

関数値

0	:正常終了。
-51	:レイヤ2自動モード・エラー。
-60	:引数エラー。

使用例

```
SETPR(0,1,3)
SENDPKT("DT",0,1)
```

機能説明　受信順序番号P(R)の値を設定する関数です。
引数で指定した論理チャネル（LCGNとLCN)のP(R)値を変更します。
この値は、DT,RR,RNR パケットを送出するときのみメッセージ中に挿入されます。

制限事項　本関数は、トランスペアレント・モードでは実行できません。

(10)～(16) Q931パラメータ挿入関数群

呼出し形式

[関数名]	[挿入するパラメータ]
(10) INS_PD("NAME",PD)	プロトコル識別子
(11) INS_CRL("NAME",CRL)	呼番号長
(12) INS_CRF("NAME",CRF)	呼番号フラグ
(13) INS_CRV("NAME",CRV)	呼番号
(14) INS_MSG("NAME",MSG)	メッセージ種別
(15) INS_INFO("NAME",N,INFO)	情報要素群フィールド
(16) INS_CS_VAL("NAME",CAUSE)	理由表示値

概　要　　送出フレームの任意のQ931パラメータを書き換えます。

引数の説明

NAME	:フレーム名
PD	:プロトコル識別子
CRL	:呼番号長
CRF	:呼番号フラグ
CRV	:呼番号
MSG	:メッセージ種別
N	:N 番目を指定
INFO	:情報要素群フィールドの値
CAUSE	:理由表示値

関数値

0	:正常終了
-60	:引数エラー
-61	:フレーム名エラー
-85	:書き換えられるメッセージが不適切

使用例

```
INS_CRV("SETUP",5)
SENDI("SETUP")
```

機能説明　　メッセージ中の任意のパラメータを書き変える関数群です。
本関数群は、メッセージの先頭から内容を解釈していき挿入するパラメータの位置を探していきます。
挿入するパラメータの位置が発見されるとそこを引数で指定された値に書き換えます。
このとき該当するパラメータが複数の位置にわたって存在する場合、引数N でその位置を確定させる必要があります。(INS_INFO関数がこれに該当します。)
また、メッセージを解釈した結果、挿入するパラメータの位置が存在しない場合には、" 書き換えられるメッセージが不適切""という意味の関数値を返します。

補足説明　INS_INFO関数は、メッセージ種別以降のオクテットを書き換えたい場合に使用します。
下図にINS_INFOの引数の指定方法を説明します。

① INS_INFO("NAME",N,INFO)

プロトコル識別子	
呼番号長	
メッセージ種別	
1オクテット目	N=1
2オクテット目	N=2
⋮	⋮
nオクテット目	N=n

(17)～(35) X.25パラメータ挿入関数群

呼出し形式

	[関数名]	[挿入するパラメータ]
(17)	INS_GFI("NAME",GFI)	ゼネラルフォーマット識別子(GFI)
(18)	INS_Q("NAME",Q)	Q ビット
(19)	INS_D("NAME",D)	D ビット
(20)	INS_LCGN("NAME",LCGN)	論理チャネルグループ番号(LCGN)
(21)	INS_LCN("NAME",LCN)	論理チャネル番号(LCN)
(22)	INS_TYP("NAME",TYP)	パケットタイプ識別子(TYP)
(23)	INS_PR("NAME",PR)	受信順序番号 (P(R))
(24)	INS_PS("NAME",PS)	送信順序番号 (P(S))
(25)	INS_M("NAME",M)	M ビット
(26)	INS_CLL("NAME",CLL)	発呼ユーザアドレス長
(27)	INS_CDL("NAME",CDL)	着呼ユーザアドレス長
(28)	INS_DA("NAME",N,DA)	着呼ユーザアドレス(N番目)
(29)	INS_SA("NAME",N,SA)	発呼ユーザアドレス(N番目)
(30)	INS_FL("NAME",FL)	ファシリティ長
(31)	INS_F("NAME",N,F)	ファシリティ(Nオクテット目)
(32)	INS_CAUSE("NAME",CAUSE)	原因
(33)	INS_DIAG("NAME",DIAG)	診断符号
(34)	SET_DTL("NAME",LEN)	データ長
(35)	INS_DATA("NAME",N,DATA)	データ(Nオクテット目)

概　要　　送出フレームの任意のパラメータを書き換えます。

引数の説明

NAME	:フレーム名
GFI	:ゼネラルフォーマット識別子(GFI)
Q	:Q ビット
D	:D ビット
LCGN	:論理チャネルグループ番号(LCGN)
LCN	:論理チャネル番号(LCN)
TYP	:パケットタイプ識別子
PR	:受信順序番号
PS	:送信順序番号
M	:M ビット
CLL	:発呼ユーザアドレス長
CDL	:着呼ユーザアドレス長
N	:N 番目を指定
DA	:着呼ユーザアドレス
SA	:発呼ユーザアドレス
FL	:ファシリティ長
F	:ファシリティ
CAUSE	:原因
DIAG	:診断符号
LEN	:データ長
DATA	:データ

(36) REQ_TEI

呼出し形式　REQ_TEI( )

概　要　TEI 割当手順を起動します。(TEモード)

引数の説明

関数値　0 ～ 126 :割当られたTEI 値。
-51 :レイヤ2自動モード・エラー。
-52 :TEモード・エラー。
-90 :TEI 割当エラー。

使用例　TEI=REQ_TEI( )

機能説明　レイヤ2自動モードで、TEI 割当手順を起動する関数です。
この関数は、本器がTEモードのときに使用します。
TEI 割当が正常に終了すると、割り当てられたTEI 値を関数値として返します。
また、このとき割り当てられたTEI 値の登録も行ないます。

制限事項　この関数は、トランスペアレント・モードやNTモードでは使用できません。

(37) CHKREQ_TEI

呼出し形式　CHKREQ_TEI(TEI)

概　要　TEI チェック手順を行ないます。(NT モード)

引数の説明　TEI　:IDチェックを行なうTEI 値。

関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-53　:NTモード・エラー。
-60　:引数エラー。

使用例
```
CHKREQ_TEI(64)
TEI=65
CHKREQ_TEI(TEI)
```

機能説明　レイヤ2自動モードで、TEI チェック手順を行なう関数です。
この関数は、本器がNTモードのときに使用します。
引数で指定されたTEI 値について、TEI チェック手順が行なわれます。引数が127 の場合はすべてのTEI 値についてのTEI チェック手順が行なわれます。

制限事項　引数が、0 ～127 以外の場合は引数エラーになります。
また、モードがトランスペアレント・モードやTEモードになっていると実行できません。

(38) REMOVE_TEI

呼出し形式　REMOVE_TEI(TEI)

概　要　TEI 解除手順を行ないます。(NTモード)

引数の説明　TEI　:解除するTEI 値。

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-53　:NTモード・エラー。
-60　:引数エラー。

使用例
```
REMOVE_TEI(64)
TEI=64
REMOVE_TEI(TEI)
```

機能説明　レイヤ2自動モードで、TEI 解除手順を行なう関数です。
この関数は、本器がNTモードのときに使用します。
この関数が実行されると、引数で指定されたTEI 値をAiフィールド中に付加したID解除メッセージを2回続けて送出します。

制限事項　引数が0～127 以外の場合は、引数エラーになります。
モードがトランスペアレント・モードやTEモードになっていると使用できません。

(39) VERIFY_TEI

呼出し形式　VERIFY_TEI( )

概　要　TEI 検証手順を行ないます。（TEモード）

引数の説明

関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-52　:TEモード・エラー。
-71　:TEI エラー。

使用例　VERIFY_TEI( )

機能説明
レイヤ2自動モードで、TEI 検証手順を起動する関数です。
この関数は本器がTEモードのときに使用します。
この関数が実行されるとID検証要求メッセージが送出されます。Aiフィールドには、あらかじめ設定されているTEI 値が入れられます。

制限事項
あらかじめ設定されているTEI 値が0 ～126 以外の場合は、TEI エラーになります。
TEI 値の設定変更は、ACT_TEI 関数などを用いて行なうことができます。
この関数はトランスペアレント・モードやNTモードでは使用できません。

(40) ＬＩＮＫＯＮ

呼出し形式　LINKON( )

概　要　リンクの設定を行ないます。

引数の説明

関数値
0　:正常終了。
-51　:レイヤ２自動モード・エラー。
-70　:SAPIエラー。
-71　:TEI エラー。
-100 :リンク設定拒否。
-101 :最大同時リンク数オーバー。

使用例　LINKON( )

機能説明
レイヤ２自動モードでリンクの設定を行なう関数です。
リンクの設定に使用するSAPI, TEI ペアには、あらかじめ設定されたものを通常は使用します。
しかし、本器がTEモードのとき、この関数の実行がシミュレーションが行なわれてから１回目である場合は、リンクの設定を行なう前にTEI の割当手順が行なわれ、そこで割り当てられたTEI 値が使用されます。
ただし、LINKON関数が実行される前に、REQ_TEI 関数が実行されている場合は、シミュレーションが行なわれてから１回目の実行であってもTEI 割当手順は行なわれず、あらかじめ設定してあったTEI 値が使用されます。
逆にLINKON関数が実行される前に、NEXT_TEI関数が実行されている場合は、シミュレーションが行なわれてから１回目の実行でなくてもTEI 割当手順が行なわれ、そこで設定されたTEI 値が使用されます。
上記の例は、本器がTEモードのときに有効であり、本器がNTモードの場合には常にあらかじめ設定されているSAPI, TEI ペアが使用されます。
SAPI値の設定変更には、ACT_SAPI関数を使用します。また、SAPI値の設定変更を禁止したい場合には、LOCK_SAPI関数を使用します。
また、本器では相手からリンクの設定を受けた場合には、そのSAPI値に設定変更されます。
しかし、相手からのリンクの設定を受け付ける前にLOCK_SAPI 関数が使用されている場合は、SAPI値の設定変更は起こりません。TEI 値の設定には、ACT_TEI 関数を使用します。TEI 値の設定変更を禁止したい場合には、LOCK_TEI 関数を使用します。
TEI 値の設定は、ACT_TEI 関数の他にREQ_TEI 関数が実行された場合や、リンクの設定が行なわれた場合に起こります。また、本器がNTモード時にTEI の割当をした場合にも、起こります。
しかし、これらのイベントが行なわれる前にLOCK_TEI関数が実行されている場合は、相手からのリンク設定によるTEI 値の設定変更は起こりません。

制限事項　設定されているSAPI値が0、16、63以外のときにこの関数が実行されると、SAPIエラーになります。設定されているTEI 値に未登録のTEI 値を使用すると、TEI エラーになります。
TEI 値の登録には、REG_TEI 関数を使用します。
本器では、最大8リンクまで同時にリンク設定が可能です。これ以上同時にリンクを設定しようとすると、最大同時リンク数エラーになります。

(41) L I N K O F F

呼出し形式　LINKOFF( )

概　要　リンクの解放を行ないます。

引数の説明

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-102 :リンク未設定エラー。

使用例　LINKOFF( )

機能説明　レイヤ2自動モードで、リンクの解放を行なう関数です。
あらかじめ設定してあるSAPI値、TEI 値をもつリンクの解放を行ないます。

制限事項　解放するはずのリンクが設定されていなかった場合には、リンク未設定エラーになります。
SAPI値の設定には、ACT_SAPI関数を用いて行ないます。
TEI 値の設定には、ACT_TEI 関数を用いて行ないます。
この関数は、トランスペアレント・モードでは使用できません。

(42) WAIT_LINK

呼出し形式　WAIT_LINK(SEC)

概　要　相手からのレイヤ2 リンク設定待ち状態にします。

引数の説明　SEC　:相手からリンクが設定されるのを待つ時間(100msec)
(0≦SEC ≦16777215)

関数値　0 ～7　:リンク番号
-51　:レイヤ2 自動モード・エラー
-60　:引数エラー
-125　:タイムアウト

使用例
```
PH_ACT( )
WAIT_LINK(3000)
SENDI("CALPRO")
```

機能説明　レイヤ2 リンクが設定されるまで待つ関数です。
本関数実行後に、相手からレイヤ2 リンクが設定されると内部的なリンク番号が関数値として返されます。このリンク番号からGET_SAPI(LINK),GET_TEI(LINK) 関数を用いて設定されたレイヤ2 リンクのSAPI値、TEI 値を知ることができます。
本関数実行前に、既にレイヤ2 リンクが設定されている場合、新たなレイヤ2 リンクが設定されるまで待ちます。引数で指定した時間を過ぎてもレイヤ2 リンクが設定されない場合は、"タイムアウト"を示す関数値を返して終了します。

制限事項　本関数はレイヤ2 自動モードで実行する関数のため、トランスペアレント・モードでは実行できません。

(43) L_STATUS

呼出し形式　L_STATUS(SAPI,TEI)

概　要　リンクが設定されているかを調べます。

引数の説明　SAPI :リンクの状態を調べるSAPI値。
TEI :リンクの状態を調べるTEI 値。

関数値　0 :リンク未設定。
1 :リンク設定済。
-51 :レイヤ2自動モード・エラー。

使用例
```
L_STATUS(0,64)
SAPI=16
TEI =65
L_STATUS(SAPI,TEI)
```

機能説明　レイヤ2自動モードで、引数で指定したSAPI, TEI ペアのリンクが設定されているかどうか調べる関数です。
調べた結果は、関数値によって返されますが、1が関数値として返された場合には、リンクが設定済みである事を示し、0が関数値として返された場合には、リンクは設定されていないことを示します。

制限事項　この関数は、トランスペアレント・モードでは使用できません。

(44) SET_BUSY

呼出し形式　SET_BUSY( )

概　要　自局をビジー状態にします。

引数の説明

関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-70　:SAPIエラー。
-71　:TEI エラー。
-72　:リンク・エラー。

使用例　SET_BUSY( )

機能説明　レイヤ2自動モードで自局をビジー状態にする関数です。
ビジー状態に使用するSAPI、TEI ペアは、あらかじめ設定されたものを使用します。
TEI が127 の場合はあらかじめ設定されたSAPI値を持つすべてのリンクをビジー状態にします。

制限事項　設定されているSAPI値が、0 、16、63以外のときにこの関数が実行されると、SAPIエラーになります。
設定されているTEI 値に、未登録のTEI 値を使用すると、TEI エラーになります。
TEI 値の登録には、REG_TEI 関数を使用します。
リンクの設定がされていないSAPI, TEI ペアに対して、この関数が実行されるとリンク・エラーになります。ただし、TEI が127 の場合にはこの限りではありません。
この関数は、トランスペアレント・モードでは使用できません。

(45) REL_BUSY

呼出し形式　REL_BUSY( )

概　要　自局のビジー状態を解除します。

引数の説明

関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-70　:SAPIエラー。
-71　:TEI エラー。
-72　:リンク・エラー。

使用例　REL_BUSY( )

機能説明　レイヤ2自動モードで、自局のビジー状態を解除する関数です。
あらかじめ設定されたSAPI, TEI ペアに対してビジー状態を解除します。
設定されているTEI 値が127 の場合は、設定されているSAPI値を持つすべてのリンクのビジー状態を解除します。

制限事項　設定されているSAPI値が、0 、16、63以外のときに、この関数が実行されると、SAPIエラーになります。
設定されているTEI 値に、未登録のTEI 値を使用すると、TEI エラーになります。
TEI 値の登録には、REG_TEI 関数を使用します。
リンクの設定がされていないSAPI, TEI ペアに対して、この関数が実行されるとリンク・エラーになります。ただし、TEI が127 の場合はこの限りではありません。
この関数は、トランスペアレント・モードでは使用できません。

(46) PROHIBIT_L

呼出し形式　PROHIBIT_L( )

概　要　新たなリンクの設定を禁止します。

引数の説明

関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-70　:SAPIエラー。

使用例　PROHIBIT_L( )

機能説明　レイヤ2自動モードで、あらかじめ設定されたSAPIについて新たなリンクの設定を禁止する関数です。
この関数が実行されると、相手局からの新たなリンクの設定は禁止されますが、自局からのリンクの設定はできます。

制限事項　設定されているSAPI値が、0 、16、63以外のときに、この関数が実行されるとSAPIエラーになります。
この関数はトランスペアレント・モードでは使用できません。

(47) PERMIT_L

呼出し形式　PERMIT_L( )

概　要　リンク設定不許可状態を解除します。

引数の説明

関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-70　:SAPIエラー。

使用例　PERMIT_L

機能説明　レイヤ2自動モードで、PROHIBIT_L 関数で設定したリンク設定不許可状態を解除する関数です。
あらかじめ設定されているSAPIに対して、リンク不許可状態を解除します。

制限事項　この関数は、トランスペアレント・モードでは使用できません。

(48) REG_TEI

呼出し形式　REG_TEI(TEI)

概　要　TEI 値を本器に登録し、そのTEI 値のフレームを受信可能にします。

引数の説明　TEI　:登録を行なうTEI 値。

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-60　:引数エラー。

使用例
```
REG_TEI(64)
TEI=65
REG_TEI(TEI)
```

機能説明　レイヤ2自動モードで、TEI の登録を行なう関数です。
引数で指定されたTEI 値について、TEI の登録が行なわれます。

制限事項　1度登録を行なったTEI を再び登録しようとすると引数エラーになります。
引数が0 ～126 以外の場合は、引数エラーになります。
この関数はトランスペアレント・モードでは使用できません。

(49) REL_TEI

呼出し形式　REL_TEI(TEI)

概　要　TEI 値を解除し、そのTEI 値のフレームを受信しないようにします。

引数の説明　TEI　:解除するTEI 値。

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-60　:引数エラー。

使用例
```
REL_TEI(64)
TEI = 65
REL_TEI(TEI)
```

機能説明　レイヤ2自動モードで、登録されているTEI 値の解除を行なう関数です。
引数には、0 ～127 の値が設定できますが、127 を設定すると登録されている全てのTEI 値が解放されます。

制限事項　引数に0 ～127 以外の値が設定されると引数エラーになります。この関数は、トランスペアレント・モードでは使用できません。

(50) NEXT_TEI

呼出し形式　NEXT_TEI( )

概　要　新たなTEI 値を使用することを宣言します。(TEモード)

引数の説明

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。

使用例　NEXT_TEI( )

機能説明　レイヤ2自動モードで、新たなTEI 値を用いてリンクを設定したいときに使用する関数です。
この関数は、本器がTEモードのときにLINKON関数とペアで使用します。
LINKON関数はリンクの設定を行なう関数です。シミュレーションが起動されてから1回目の実行の際にのみ、リンクの設定を行なう前にTEIの割当手順が起動されます。したがって、2回目からのLINKON関数では、あらかじめ設定されたSAPI, TEI ペアについてリンクの設定が行なわれ、TEI の割当手順は起動されません。
NEXT_TEI 関数は、TEI の割当手順を再び行なって、新たなTEI 値でリンクの設定をしたい場合に、LINKON関数を実行する前に使用します。
この関数の代わりに、REQ_TEI 関数を用いても同じことができます。

制限事項　この関数はトランスペアレント・モードでは使用できません。
また、NTモードで使用しても何も生じません。

(51) ACT_SAPI

呼出し形式　ACT_SAPI(SAPI)

概　要　フレーム送出時に使用されるSAPI値を設定します。

引数の説明　SAPI :設定をするSAPI値。

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-60　:引数エラー。

使用例
```
ACT_SAPI(0)
SAPI=16
ACT_SAPI(SAPI)
```

機能説明　レイヤ2自動モードで、SAPI値を設定変更する場合に使用する関数です。

制限事項　引数で指定されたSAPI値が0 、16、63以外の場合は、引数エラーになります。
この関数はトランスペアレント・モードでは使用できません。

(52) ACT_TEI

呼出し形式　ACT_TEI(TEI)

概　要　フレーム送出時に使用されるTEI 値を設定します。

引数の説明　TEI　:設定を行なうTEI 値。

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-60　:引数エラー。

使用例
```
ACT_TEI(64)
TEI=65
ACT_TEI(TEI)
```

機能説明　レイヤ2自動モードで、TEI 値を設定変更する場合に使用する関数です。

制限事項　引数で指定されたTEI 値が未登録の場合は、引数エラーになります。
この関数は、トランスペアレント・モードでは使用できません。

(53) ACT_LINK

呼出し形式　ACT_LINK(LINK)

概　要　フレーム送出時に使用されるリンク番号を設定します。

引数の説明　LINK :リンク番号 (0 ≦LINK≦7)

関数値
0　:正常終了。
-51　:レイヤ2 自動モード・エラー。
-60　:引数エラー。
-102 :リンク未設定エラー。

使用例
```
PH_ACT( )
REQ_TEI( )
LINK1=LINKON( )
REQ_TEI( )
LINK2=LINKON( )
ACT_LINK(LINK1)
SENDI("SETUP1")
ACT_LINK(LINK2)
SENDI("SETUP2")
```

機能説明　I フレーム送出時に、使用するレイヤ2 リンクを変更する関数です。引数で与えられるLINKは内部的に使用しているリンク番号で、LINKON, WAIT_LINK, SEE_LINK, GET_LINK関数などで知ることができます。引数LINKが指定したレイヤ2 リンクが設定されていない場合は、" リンク未設定エラー" となります。

制限事項　本関数はレイヤ2 自動モード用関数のため、トランスペアレント・モードでは実行できません。

(54) LOCK_SAPI

呼出し形式　LOCK_SAPI( )

概　要　SAPI値が自動で設定変更されるのを禁止します。

引数の説明

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。

使用例　LOCK_SAPI( )

機能説明　レイヤ2自動モードで、相手からのリンク設定によりSAPI値が設定変更されることを禁止する関数です。
ただし、この関数が実行された後にもACT_SAPI関数によりSAPI値を設定変更することはできます。

制限事項　LOCK_SAPI関数によりSAPI値変更不可状態になりますが、これを解除するには、FLEX_SAPI関数を用いて行ないます。
この関数は、トランスペアレント・モードでは使用できません。

(55) L O C K _ T E I

呼出し形式　LOCK_TEI( )

概　要　TEI 値が自動で設定変更されるのを禁止します。

引数の説明

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。

使用例　LOCK_TEI( )

機能説明　レイヤ2自動モードで、相手からのリンク設定によるTEI 値の設定変更を禁止する関数です。
ただし、この関数が実行された後でも、ACT_TEI 関数を用いて、TEI 値を設定変更することはできます。

制限事項　LOCK_TEI 関数を実行することによりTEI 値変更不可状態になりますが、この状態を解除するためには、FLEX_TEI 関数を用います。
この関数は、トランスペアレント・モードでは使用できません。

(56) LOCK_LINK

呼出し形式　LOCK_LINK( )

概　要　リンク番号が自動で設定変更されるのを禁止します。

引数の説明

関数値　0　:正常終了。
-51　:レイヤ2 自動モード・エラー。

使用例
```
LINKON( )
LOCK_LINK( )
SENDI("SETUP")
```

機能説明　相手からのリンク設定によりSAPI値、TEI 値が設定変更されるのを禁止する関数です。
ただし、ACT_SAPI, ACT_TEI, ACT_LINK関数などでSAPI値、TEI 値を設定変更することはできます。LOCK_LINK関数で設定したSAPI値、TEI 値変更不可状態はFLEX_LINK関数により解除できます。

制限事項　本関数は、レイヤ2 自動モードで使用する関数のため、トランスペアレント・モードでは実行できません。

(57) FLEX_SAPI

呼出し形式　FLEX_SAPI( )

概　要　SAPI値が自動で設定変更されるようにします。

引数の説明

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。

使用例　FLEX_SAPI( )

機能説明　レイヤ2自動モードで、SAPI値変更不可状態を解除する関数です。

制限事項　この関数は、トランスペアレント・モードでは使用できません。

(58) F L E X _ T E I

呼出し形式　FLEX_TEI( )

概　要　TEI 値が自動で設定変更されるようにします。

引数の説明

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。

使用例　FLEX_TEI( )

機能説明　レイヤ2自動モードで、TEI 値変更不可状態を解除する関数です。

制限事項　この関数はトランスペアレント・モードでは使用できません。

(59) FLEX_LINK

呼出し形式　FLEX_LINK( )

概　要　リンク番号が自動で設定変更されるようにします。

引数の説明

関数値　0　:正常終了。
-51　:レイヤ2 自動モード・エラー。

使用例
```
LINKON( )
LOCK_LINK( )
SENDI("SETUP")
FLEX_LINK( )
```

機能説明　LOCK_LINK関数によりリンクが自動で変更されるのを禁止した状態を解除し、再びリンクが自動変更されるのを許可する関数です。

制限事項　本関数はレイヤ2 自動モードで使用する関数のため、トランスペアレント・モードでは実行できません。

(60) GET_SAPI

呼出し形式　GET_SAPI(LINK)

概　要　フレーム送出時に使用するSAPI値を調べます。

引数の説明　LINK　:リンク番号　(0 ～ 7)

関数値
0 ～ 63　:調べたSAPI値。
-51　:レイヤ2 自動モード・エラー。
-60　:引数エラー。
-102　:リンク未設定エラー。

仕様実例
```
LINK=WAIT_LINK(3000)
SAPI=GET_SAPI(LINK)
```

機能説明　引数LINKのレイヤ2 リンクが設定されているか調べにいき、設定されていればそのリンクのSAPI値を関数値として返します。引数LINKのレイヤ2 リンクが設定いない場合に、"リンク未設定エラー"の関数値を返します。

制限事項　本関数はレイヤ2 自動モードで使用する関数のため、トランスペアレント・モードでは実行できません。

(61) GET_TEI

呼出し形式　GET_TEI(LINK)

概　要　フレーム送出時に使用する TEI値を調べます。

引数の説明　LINK　:リンク番号　(0 ～ 7)

関数値
- 0 ～ 127　:調べた TEI値。
- -51　:レイヤ2 自動モード・エラー。
- -60　:引数エラー。
- -102　:リンク未設定エラー。

使用例
```
LINK=WAIT_LINK(3000)
TEI=GET_TEI(LINK)
```

機能説明　引数LINKのレイヤ2 リンクが設定されているか調べにいき、設定されていればそのリンクの TEI値を関数値として返します。引数LINKのレイヤ2 リンクが設定されていなかったら"リンク未設定エラー"の関数値を返します。

制限事項　本関数はレイヤ2 自動モードで使用する関数のため、トランスペアレント・モードでは実行できません。

(62) GET_LINK

呼出し形式　GET_LINK(SAPI,TEI)

概　要　フレーム送出時に使用されるリンク番号を調べます。

引数の説明
SAPI　:SAPI値
TEI　:TEI 値

関数値
0 ～ 7　:調べたリンク番号。
-51　:レイヤ2 自動モード・エラー。
-60　:引数エラー。
-102　:リンク未設定エラー。

使用例
```
LINK=GET_LINK(0,64)
ACT_LINK(LINK)
SEND1("SETUP")
```

機能説明　引数SAPI,TEIペアリンクが設定されているか調べにいき、設定されていればそのリンク番号を関数値として返します。引数SAPI,TEIペアのリンクが設定されていなければ”リンク未設定エラー”の関数値を返します。

制限事項　本関数はレイヤ2 自動モードで使用する関数のため、トランスペアレント・モードでは実行できません。

(63) SEE_LINK

呼出し形式　SEE_LINK( )

概　要　現在設定されているリンク番号を調べます。

引数の説明

関数値　0 ～ 7 :リンク番号。
-51　:レイヤ2 自動モード・エラー。
-102　:リンク未設定エラー。

使用例
```
RECEIVE(0)
LINK=SEE_LINK()
PRINT("LINK=%D\N",LINK)
```

機能説明　本関数が実行されている時点で、設定されているレイヤ2 リンクがあれば有効になっているレイヤ2 リンクのリンク番号を関数値として返します。
しかし、設定されているレイヤ2 リンクが無い場合は”リンク未設定エラー”となります。

制限事項　本関数はレイヤ2 自動モードで使用する関数のため、トランスペアレント・モードでは実行できません。

(64)～(78) Q.931 パラメータ読み出し関数

呼出し形式

	[関数名]	[読み出すパラメータ]
(64)	RXSAPI()	SAPI値
(65)	RXTEI()	TEI 値
(66)	RXTYPE()	レイヤ2 フレームのタイプ
(67)	RXPD()	プロトコル識別子
(68)	RXCRL()	呼番号長
(69)	RXCRF()	呼番号フラグ
(70)	RXCRV()	呼番号
(71)	RXMSG()	メッセージ種別
(72)	RXINFO_NUM()	情報要素数
(73)	RXINFO_ELM(N)	情報要素識別子
(74)	RXINFO_LEN(N)	情報要素内容長
(75)	RXINFO_VAL(N,M)	情報内容
(76)	RXINFO_POS(ELM)	情報要素識別子の位置
(77)	RXCS_VAL()	理由表示値
(78)	RXCHAN_NUM()	チャネル番号

概　要　　受信フレームの任意のQ.931 パラメータを読み出します。

引数の説明

N	:情報要素の位置。
M	:情報内容の位置。
ELM	:情報要素識別子。

関数値

0 ～ 255	:読み出したQ.931 パラメータの値。
-60	:引数エラー。
-80	:フレーム未受信エラー。
-82	:読み出すパラメータが受信フレーム中に存在しない。

使用例

```
RECEIVE(0)
CRV=RXCRV()
PRINT("Call Ref=%D＼N",CRV)
```

機能説明　　受信フレームのQ.931 パラメータを読み出す関数群です。
受信したフレームを順に読むことができます。
受信フレームを読み出したい場合はRECEIVE 関数を実行し、受信フレームを内部に取り込んでから本関数群を実行します。次の受信フレームを読み出したい場合は、再びRECEIVE 関数を実行します。
このときも受信フレームが無い場合は受信待ち状態になります。本関数は受信フレームの先頭から内容を解釈していき、該当するX.25パラメータを関数値として返しますが、該当するX.25パラメータが複数存在する場合(RXINFO_ELM, RXINFO_LEN, RXINFO_VAL)は、引数N, Mでその位置を確定させる必要があります。
また、メッセージの内容を解釈していった結果読み出すパラメータが存在しない場合 "読み出すパラメータが存在しない" という意味の関数値を返します。

補足説明　　RXINFO_LEN, RXINFO_VAL, RXINFO_POS, RXCHAN_NUM 関数については以降に説明します。

(74) RXINFO_LEN

呼出し形式　RXINFO_LEN(N)

概　要　受信フレームの情報要素内容長を読み出します。

引数の説明　N　:情報要素内容長を読み出す情報要素の位置。

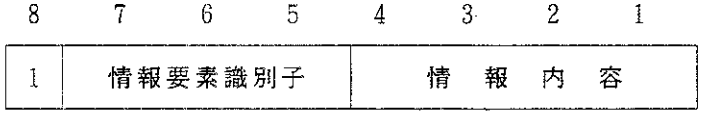
関数値
0 ～ 255 :受信フレームのレイヤ3メッセージ内に含まれる情報要素内容長（JT-Q931)。
-60 :引数エラー。
-80 :フレーム未受信エラー。
-82 :JT-Q931 で示す情報内容が存在しない。

使用例
```
LEN1=RXINFO_LEN(1)
NUM=2
LEN=RXINFO_LEN(NUM)
```

機能説明　"(64) ～(78) Q.931パラメータ読み出し関数" の頁を参照して下さい。

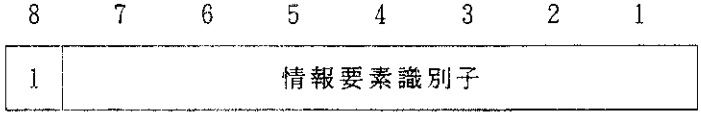
詳細説明　引数で指定した情報要素の情報要素内容長を関数値として返しますが、引数で指定した位置に情報要素が存在しない場合や、情報要素があっても情報内容が無い場合には、引数エラーになります。情報要素数以上の値が引数として指定されたときには、引数エラーが起こります。下図を参照して下さい。

a) 単一固定長情報要素のフォーマット（タイプ1）



RXINFO_LEN (N) = 0

b) 単一固定長情報要素のフォーマット（タイプ2）



RXINFO_LEN (N) = -82 （エラー）

c) 可変長情報要素の場合

RXINFO_LEN (N) = 情報要素内容長

(75) RXINFO_VAL

呼出し形式　RXINFO_VAL(N,M)

概　要　受信フレームの情報内容を読み出します。

引数の説明
N　:情報要素内容を読み出し情報要素の位置。
M　:情報内容中から読み出しデータのオクテット値。

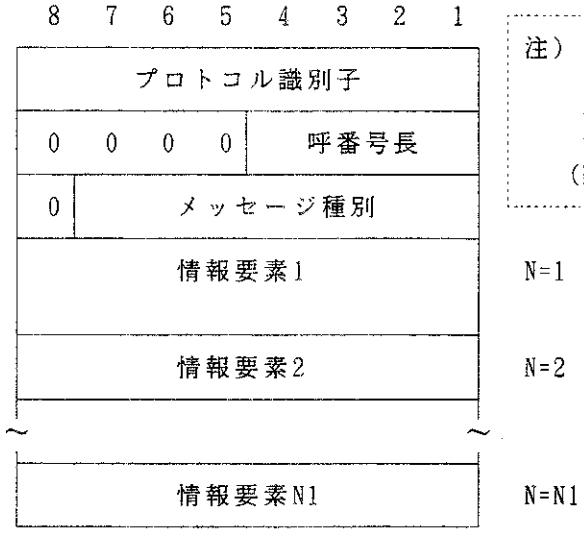
関数値
0 ～ 255 :受信フレームのレイヤ3 メッセージ内に含まれる情報内容（JT-Q931)。
-60　:引数エラー。
-80　:フレーム未受信エラー。
-82　:JT-Q931 で示す情報内容が存在しない。

使用例
```
VAL=RXINFO_VAL(1,1)
N=1
M=2
VAL2=RXINFO_VAL(N,M)
```

機能説明　"(64) ～(78) Q.931パラメータ読み出し関数" の頁を参照して下さい。

詳細説明　引数で指定した位置の情報内容を関数値として返します。
引数の設定の仕方は、下図に示します。下図以外の設定がされた場合には、引数エラーになります。

① RXINFO_VAL(N,M)関数における引数N の指定の仕方。



② RXINFO_VAL(N,M)関数における引数M の指定の仕方。

A) 単一固定長情報要素のフォーマット（タイプ1）

8	7	6	5	4	3	2	1	
1	情報要素識別子			情 報 内 容				M＝0

RXINFO_LEN(N)＝0
RXINFO_VAL(N,0)＝“情報内容”

B) 単一固定長情報要素のフォーマット（タイプ2）

8	7	6	5	4	3	2	1	
1	情報要素識別子							有効なMはない。

RXINFO_LEN(N)＝-82(引数エラー)
RXINFO_VAL(N,M)＝-82(引数エラー)

注）このタイプの情報要素の場合には、RXINFO-LEN関数やRXINFO_VAL 関数は実行できません。

C) 可変長情報要素のフォーマット

8	7	6	5	4	3	2	1	
0	情報要素識別子							
情報要素内容長								
情 報 内 容 1								M=1
情 報 内 容 2								M=2
情 報 内 容 3								M=3
～							～	
情 報 内 容 M1-1								M=M1-1
情 報 内 容 M1								M=M1

注）引数Mは、1 ≦M≦M1の範囲で指定します。
(M1 は情報要素内容長)

(76) RXINFO_POS

呼出し形式　RXINFO_VAL(N,M)

概　要　情報要素識別子の位置を調べます。

引数の説明　ELM　:情報要素識別子の位置。

関数値
0 ～ 4200	:情報要素識別子の位置。
-60	:引数エラー。
-80	:フレーム未受信エラー。
-82	:読み出すパラメータが受信フレーム中に存在しない。

使用例
```
RECEIVE(0)
POS=RXINFO_POS(H'18')
CHAN=EXTRACT(POS+2)&H'03'
PRINT("CHANNEL=%D\N",CHAN)
```

機能説明　受信フレーム中にある引数ELM の情報要素識別子の位置を関数値として返す関数です。
特定の情報要素識別子の内容について読み出したい場合に、本関数は有効です。本関数で指定の情報要素識別子の位置を調べ、EXTRACT関数でその位置をもとに内容を読み出すと簡単に知りたい内容を得ることができます。ただし、コード群0 以外の識別子についても同様に位置を返すので注意が必要です。
本関数を実行する前に必ずRECEIVE 関数を実行し、受信フレームを内部に取り込んで下さい。複数のフレームを受信している場合、RECEIVE 関数を実行するたびに読み出す対象となる受信フレームが更新されていきます。この順番は、フレームを受信した順です。
RECEIVE 関数を実行すると次の受信フレームに読み出す対象のフレームを切り換えます。このとき次のフレームがまだ受信されていない場合は、フレーム受信待ち状態になります。
RECEIVE 関数が実行されていない場合または、フレーム未受信の状態で本関数を実行する場合に、フレーム未受信エラーになります。
また、引数ELM の情報要素識別子が受信フレーム中に存在しない場合は引数エラーになります。
受信フレームがQ.931 フォーマットでない場合には、”読み出すパラメータが受信フレーム中に存在しない”と言う意味の関数値を返します。

(78) RXCHAN_NUM

呼出し形式　RXCHAN_NUM

概　要　受信フレームのチャネル番号を読み出します。

引数の説明

関数値

0	:任意のチャネル。
1 ～ 24	:B チャネルの番号。
101	:H0チャネル・ユニット(ch1～6 使用)。
102	:H0チャネル・ユニット(ch7～12使用)。
103	:H0チャネル・ユニット(ch13 ～18使用)。
104	:H0チャネル・ユニット(ch19 ～24使用)。
200	:H11 チャネル。
-80	:フレーム未受信エラー。
-82	:読み出すフレームが受信フレーム中に存在しない。

使用例

```
RECEIVE(0)
CHAN=RXCHAN_NUM( )
PRINT("CHANNEL=%D\N",CHAN)
```

機能説明　"(64) ～(78) Q.931パラメータ読み出し関数" を参照して下さい。

詳細説明　受信したフレームのチャネル識別子内にあるチャネル番号を関数値として返します。関数値として1 ～24が返される場合、情報チャネルの選択はそれぞれB1～B24 の関数値で示された番号のB チャネルであることが分かります。また0 が返される場合は、情報チャネル選択は任意のチャネルであることを意味します。情報チャネル選択が "チャネルなし" の場合には、関数値として-82 が返されます。インタフェース・タイプが一次群の場合で、H0チャネルが指定される場合は、上記の関数値に示すとおり、101 ～104 の値が返されます。ただし、一次群の場合でB チャネルが複数指定される場合、先頭に書かれるB チャネル（番号で指定の場合）または、B チャネルの一番若い番号（マップで指定の場合）が返されます。またH11 チャネルが指定された場合は、200 が関数値として返されます。

(79)～(97) X.25パラメータ読み出し関数群

呼出し形式	[関数名]	[挿入するパラメータ]
	(79) RXGFI( )	ゼネラルフォーマット識別子(GFI)
	(80) RXQ( )	Q ビット
	(81) RXD( )	D ビット
	(82) RXLCGN( )	論理チャネルグループ番号(LCGN)
	(83) RXLCN( )	論理チャネル番号(LCN)
	(84) RXTYP( )	パケットタイプ識別子
	(85) RXPR( )	受信順序番号P(R)
	(86) RXPS( )	送信順序番号P(S)
	(87) RXM( )	M ビット
	(88) RXCLL( )	発呼ユーザアドレス長
	(89) RXCDL( )	着呼ユーザアドレス長
	(90) RXDA(N)	着呼ユーザアドレス(N番目)
	(91) RXSA(N)	発呼ユーザアドレス(N番目)
	(92) RXFL( )	ファシリティ長
	(93) RXF(N)	ファシリティ(Nオクテット目)
	(94) RXCAUSE( )	原因
	(95) RXDIAG( )	診断符号
	(96) RXDTL( )	データ長
	(97) RXDATA(N)	データ(Nオクテット目)

概　要　受信フレームの任意のX.25パラメータを読み出します。

引数の説明　N　:N 番目を指定。

関数値
```
0 ～ 255 :読み出したパラメータの値。
-60      :引数エラー。
-80      :フレーム未受信エラー。
-82      :読み出すパラメータが受信フレーム中に存在しない。
```

使用例
```
RECEIVE(0)
GFI=RXGFI( )
PRINT("GFI=%D ＼N",GFI)
```

機能説明　受信フレームのX.25パラメータを読み出す関数です。
フレームを受信した順に読むことができます。
受信フレームを読み出したい場合はRECEIVE 関数を実行し、受信フレームを内部に取り込んでから本関数群を実行します。次の受信フレームを読み出したい場合は、再びRECEIVE 関数を実行します。このとき受信フレームが無い場合は、受信待ち状態になります。本関数は受信フレームの先頭から内容を解釈していき、該当するX.25パラメータを関数値として直しますが、該当のX.25パラメータが複数存在する場合(RXSA,RXDA,RXF,RXDATA など) 引数N で、何番目のパラメータを読み出すのか指定する必要があります。
このときN が不適切であると、引数エラーとなります。
また、メッセージの内容を解釈していった結果、読み出すパラメータが存在しない場合、”読み出すパラメータが受信フレーム中に存在しない”という意味の関数値を返します。

補足説明　RXTYP( )関数で返される関数値とパケット名は、以下の対応となっています。

パケット名	関数値		パケット名	関数値	
	10進	16進		10進	16進
CR	11	0B	SF	255	FF
CN	11	0B	DT	0	00
CA	15	0F	RR	1	01
CC	15	0F	RNR	5	05
CQ	19	13	RQ	27	1B
CI	19	13	RI	27	1B
CF	23	17	RF	31	1F
SQ	251	FB	IT	35	23
SI	251	FB	IF	39	27

13. Bチャネル・シミュレーション

13.1 共通関数

[表13-1] に共通関数を表示します。

表 13 - 1 共通関数

(1) INSERT	送信フレームの任意のオクテットの値を変える。
(2) SET_FRLEN	送信フレームのフレーム長を変える。
(3) SET_CHANN	送受信するチャネルを指定する。
(4) RECEIVE	タイムアウト／フレーム／キー入力／イベント受信待ち状態にする。
(5) T_START TM_START	タイマを起動する。（分解能 1秒のタイマ起動） タイマを起動する。（分解能100m秒のタイマ起動）
(6) T_STOP	タイマを停止する。
(7) READ_TIMER	タイムアウトしたタイマのIDを得る。
(8) SEND_EVENT	他チャネルにイベントを送る。
(9) READ_EVENT	他チャネルから送られてきたイベントの内容を読み出す。
(10) EXTRACT	受信フレームの任意のオクテットの値を読み出す。
(11) RXFRLEN	受信フレーム・データ長を調べる。
(12) WAIT	指定の時間プログラムを停止する。
(13) PH_ACT	レイヤ１を起動する。
(14) PH_DEACT	レイヤ１を停止する。
(15) READ_VAL	キー入力された数値を読み出す。

次ページ以降に各関数の説明をします。

(1) INSERT

呼出し形式　INSERT("NAME",n,DT)

概　要　送信フレームの任意のオクテット値を書き換えます。

引数の説明　"NAME" :データの変更を行なうフレームのフレーム名。
n :データの変更を行なうオクテット値。
DT :データの変更値。(0～255 の範囲)

関数値　0 :正常終了。
-60 :引数エラー。
-61 :フレーム名エラー。

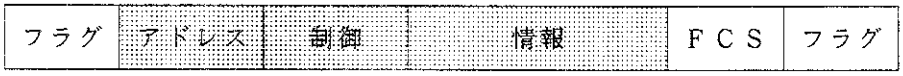
使用例　INSERT("SABME",2,H'81')

機能説明　引数で指定したフレーム名の任意のオクテットのデータを変更する関数です。

注意事項　一度、変更した値は、シミュレーションをストップするまで有効です。
トランスペアレント・モードとレイヤ2自動モードとでは変更できるデータの領域が違います。(下図参照)
下図で斜線部の領域が変更可能です。
また、変更位置は斜線部の左側から順に、1から数えたオクテットとします。

制限事項　引数で指定したフレーム名のフレームがない場合は、フレーム名エラーになります。
また、データのない場所を変更しようとすると、引数エラーになります。

① トランスペアレント・モード



② レイヤ2自動モード

(2) SET_FRLEN

呼出し形式　SET_FRLEN("NAME",LEN)

概　要　送信フレームのデータ長を変えます。

引数の説明　"NAME" :フレーム名
LEN :変更するデータ長
トランスペアレントモード時 :1～4200
レイヤ2自動実行モード時 :1～4198または1～4197
（Max値は、レイヤ2のメッセージに依存する）

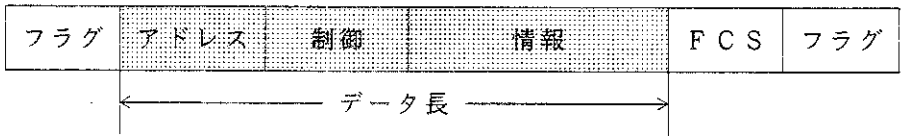
関数値　0 :正常終了。
-60 :引数エラー。
-61 :フレーム名エラー。

使用例　SET_FRLEN("SPL1",10)

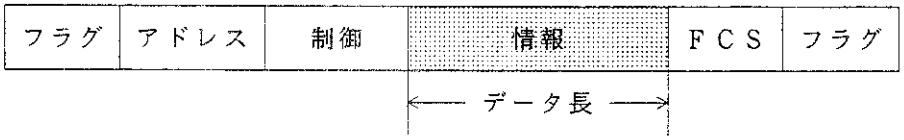
機能説明　引数で指定したフレーム名のデータ長を変更する関数です。

注意事項　一度フレーム長を変えるとシミュレーションを停止するまで有効です。トランスペアレントモードとレイヤ2自動実行モードでは、変更できるデータ長の意味が異なります。（下図参照）

① トランスペアレント・モード



② レイヤ2自動実行モード



(3) SET_CHANN

呼出し形式　SET_CHANN(CHANN)

概　要　送受信するチャネルを指定します。

引数の説明　CHANN : Bチャネル番号
0...チャネルを選択しない。
1...B1チャネルを選択。
2...B2チャネルを選択。
.
.
24...B24チャネルを選択。

関数値　0 :正常終了。
-163 :指定したチャネルが存在しない。

使用例　SET_CHANN(1)

機能説明　使用する Bチャネルを指定する関数です。この関数を実行することにより、引数で指定したチャネルでフレームの送受信を行なうことができます。
デフォルトは、チャネルを選択していない状態になっています。回線にデータを送受信する場合は、必ずこの関数を実行して下さい。

(4) RECEIVE

呼出し形式　RECEIVE(TIMER_ID)

概　要　タイムアウト／フレーム／イベント受信待ち状態にします。

引数の説明　TIMER_ID :タイマ番号。

関数値
0	:タイムアウトによる終了。
1	:フレーム受信による終了。
2	:キー入力による終了。
10	:D チャネルからのイベント受信による終了。
20	:B チャネルからのイベント受信による終了。

使用例
```
TM_ID=1
TM_SEC=10
T_START(TM_ID,TM_SEC)
RET=RECEIVE(TM_ID)
IF RET==0 THEN
   PRINT("TIME OUT＼N")
END
```

機能説明　タイムアウト，フレーム受信，他チャネルからのイベント受信待ち状態になります。
上記のイベントを受信するとこの関数は終了しますが、そのとき関数値で終了理由を返します。
タイマは、本関数とT_START，T_STOP関数を用いて管理します。
タイマの起動は、T_START関数で行ない、タイムアウトしたそのイベントは本関数で受け取ります。そのとき、本関数の引数TIMER_ID以外のタイマがタイムアウトしたとしてもそれは受信されません。ただし、TIMER_IDが0の時は例外で、すべてのタイマIDのタイムアウトイベントを受信します。このときどのタイマがタイムアウトしたかはREAD_TIMER関数で知ることができます。タイマの停止は、T_STOP関数で行ないます。
すでに、フレームを受信している状態でRECEIVE関数が実行されると関数値として1を返し終了します。このときRXで始まる受信フレーム読み出し関数が利用できます。
受信した次のフレームの内容を読み出したい場合は、再びRECEIVE関数を実行します。RECEIVE関数を実行するごとに受信したフレームの順に、その内容をRXで始まる関数を使用することにより読み出すことができます。
受信したフレームが無い場合や、次のフレームをまだ受信していない場合は、受信待ち状態になりフレームを受信した後、上記の様にフレーム内容を読み出せるようになります。
INPUT 関数が実行されていないときに、Command ラインから数値が入力されると、関数値2 を返します。この数値は、READ_VAL関数で読み出すことができます。（INPUT 関数参照）
他チャネルからのイベントを受信した場合には、関数値として上記の値を返します。受信した他チャネルからのメッセージはREAD_MESSAGE関数で読むことができます。

(5) T_START／TM_START

呼出し形式　T_START(TIMER_ID,SEC)
TM_START(TIMER_ID,SEC)

概　要　タイマを起動します。

引数の説明
TIMER_ID　:起動するタイマのタイマ番号。(1～16777215)
SEC　:タイムアウト値。
関数T_START では、1秒単位　(0～16777215)
関数TM_START では、100m秒単位　(0～16777215)

関数値
0　:正常終了。
-60　:引数エラー。
-120　:最大同時起動タイマ数オーバ。

使用例
```
T_START(201,1)
T202=202
S=2
T_START(T202,S)
```

機能説明　引数で指定したタイマ番号のタイマを起動する関数です。
この関数が実行されたときに、同じタイマ番号のタイマが既に起動されている場合は、そのタイマを停止し、新たにタイマを起動させます。

制限事項　タイムアウト値に 16777215 をこえる値が設定されると引数エラーになります。
最大同時起動タイマ数以上タイマが起動されると最大同時起動タイマ数オーバのエラーになります。
最大同時起動タイマ数は、関数T_START では 5個、関数TM_STARTでは20個まで可能です。

(6) T_STOP

呼出し形式　T_STOP(TIMER_IA)

概　要　タイマを停止します。

引数の説明　TIMER_ID　:停止するタイマのタイマ番号。(0～16777215)

関数値

0	:正常終了。
-60	:引数エラー。
-121	:起動しているタイマがない。

使用例

```
T_STOP(201)
T202=202
T_STOP(T202)
```

機能説明　引数で指定したタイマ番号のタイマを停止する関数です。

制限事項　引数で指定したタイマ番号のタイマが起動していない場合は、エラー・コードを関数値として返すだけで、他に何もしません。

(7) READ_TIMER

呼出し形式　READ_TIMER( )

概　要　タイムアウトしたタイマのIDを得ます。

引数の説明

関数値　0 ～16777215　:タイムアウトしたタイマの番号。
-121　:タイムアウトしたタイマが無い。

使用例

```
T_START(1,5)
T_START(2,10)
RET=RECEIVE(0)
IF RET==0 THEN
   TIMER=READ_TIMER( )
END
PPINT("TIMER=%D\N",TIMER)
```

機能説明　タイムアウトしたタイマの番号を知ることができます。
タイマの使い方は、RECEIVE関数の説明を参考にして下さい。

(8) SEND_EVENT

呼出し形式　SEND_EVENT(DEST,MSG)

概　要　他チャネルにイベントを送ります。

引数の説明　DEST :10:Dチャネル。
20:Bチャネル。
MSG :送出するイベント。（0～16777215）

関数値　0 :正常終了。
-60 :引数エラー。

使用例　SEND_EVENT(10,120)
（Dチャネルに120というメッセージを送る）

機能説明　他チャネルにメッセージを送る関数です。
送ったメッセージは、送られたチャネルでRECEIVE関数と
READ_EVENT関数を使用することにより読み出すことができます。

制限事項　送るメッセージは、0～16777215の数です。
それ以外が指定されると引数エラーとなります。

(9) READ_EVENT

呼出し形式　READ_EVENT( )

概　要　他チャネルから送られてきたイベントの内容を読み出します。

引数の説明

関数値　0 ～16777215 :イベント内容。
-156 :イベント未受信。

使用例
```
RET=RECEIVE(0)
IF RET==10 THEN
   MSG=READ_EVENT( )
   PRINT("MESSAGE=[%X]＼N",MSG)
END
```

機能説明　他チャネルから送られてきたイベントの内容を知ることができます。他チャネルからのイベントの送出があると、そのイベントの発生をRECEIVE 関数により知ることができます。RECEIVE 関数の関数値により、他チャネルからのイベントの受信があったのを確認した後、本関数を実行するとイベント内容を知ることができます。

(10) EXTRACT

呼出し形式　EXTRACT(n)

概　要　受信フレームの任意のオクテットの値を読み出します。

引数の説明　n　:読み出すデータのオクテット値。
（1～受信フレーム長）の値が設定能です。

関数値　0 ～255　:読み出したデータの値。
-60　:引数エラー。
-80　:フレーム未受信エラー。

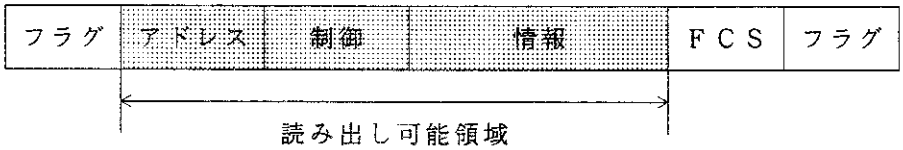
使用例
```
CF1=EXTRACT(3)
AA=4
CF2=EXTRACT(AA)
```

機能説明　受信したフレームの任意のオクテットの値を読み出す関数です。

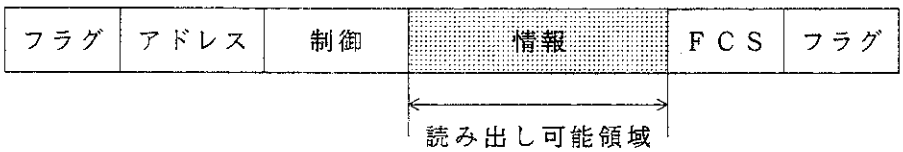
注意事項　トランスペアレント・モードとレイヤ２自動とでは読み出せるデータの領域が違います。（下図参照）
下図で斜線部の領域が読み出し可能領域です。また、読み出す位置の指定は斜線部の左側から順に、１から数えたオクテットで行ないます。

制限事項　引数で設定したオクテットのデータを関数値として返します。引数は、１～受信フレーム長までの値が設定可能です。それ以外を設定すると引数エラーになります。また、フレームを受信していないかRECEIVE関数を実行していない状態でこの関数が実行されるとフレーム未受信エラーになります。

① トランスペアレント・モード



② レイヤ２自動モード



(11) RXFRLEN

呼出し形式　RXFRLEN( )

概　要　受信フレーム・データ長を調べます。

引数の説明

関数値　0 ～65535 :受信フレームのデータ長。
-80 :フレーム未受信エラー。

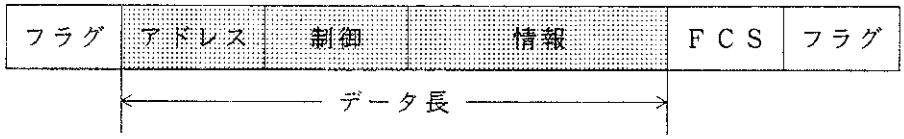
使用例　LENGTH=RXFRLEN( )

機能説明　受信フレームのデータ長を関数値として返します。

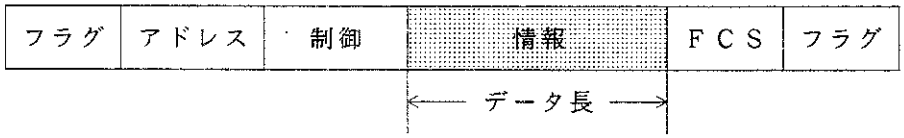
注意事項　トランスペアレント・モードとレイヤ2自動モードでは、データ長の定義が違います。（下図参照）

制限事項　この関数はフレームを受信していないか、またはフレームを受信していてもRECEIVE関数を実行していないと、フレーム未受信エラーとなります。

① トランスペアレント・モード



② レイヤ2自動モード



⑿ WAIT

呼出し形式　WAIT(SEC)

概　要　指定の時間プログラムを停止します。

引数の説明　SEC　:ウエイトする時間。
100msec単位で、0～16777215の値が設定できます。

関数値　0　:正常終了。
-60　:引数エラー

使用例
```
WAIT(10)
SEC=5
WAIT(SEC)
```

機能説明　指定した時間プログラムの実行を停止します。

(13) PH_ACT

呼出し形式　PH_ACT( )

概　要　レイヤ1を起動します。

引数の説明

関数値
0　:正常終了。
-56　:基本インタフェース以外の宣言がされている。
-160 :レイヤ1エラー。

使用例　PH_ACT( )

機能説明　レイヤ1の起動を行なう関数です。

注意事項　レイヤ1の起動に失敗すると、レイヤ1エラーを示す関数値が返ります。
この関数は、レイヤ1 を基本インタフェースに宣言しているときのみ有効です。

(14) PH_DEACT

呼出し形式　PH_DEACT( )

概　要　レイヤ1を停止させます。

引数の説明

関数値　0　:正常終了。
-56　:基本インタフェース以外の宣言がされている。
-53　:NTモードエラー。

使用例
```
PH_DEACT( )
```

機能説明　レイヤ1を停止する関数です。

制限事項　NTモード以外では、実行できません。
この関数は、レイヤ1 を基本インタフェースに宣言しているときのみ有効です。

(15) READ_VAL

呼出し形式　READ_VAL( )

概　要　キー入力された数値を読み出します。

引数の説明

関数値　0 ～1677215　:キー入力された数値
-175　:キー入力されていない。

使用例
```
RET=RECEIVE(0)
IF RET==2 THEN
   VAL=READ_VAL( )
   PRINT("KEY VALUE=%D＼N",VAL)
END
```

機能説明　Command ラインから入力された数値を読み出します。Command ラインから入力された数値を読み出す関数としてINPUT 関数がありますが、INPUT 関数によりキー入力受信待ち状態になっているときは、INPUT 関数が優先されます。従って、RECEIVE 関数にキー入力の情報は渡されません。
INPUT 関数による受信待ち状態か、RECEIVE 関数による受信待ち状態かは、画面下部に表示されているアイコン表示で識別します。
（それぞれ、"INPUT","RCV?"と表示されます。）

13.2　トランスペアレント・モード用関数

表 13 - 2　トランスペアレント・モード用関数

フレーム送信関連		
(1)	SENDT	フレームを送信する。（メッセージ・データをそのまま送出）
(2)	SENDF	フレームを送信する。(N(S),N(R),P/Fを挿入して送出)
(3)	INCVS	送信状態変数 V(S) を＋1 する。
(4)	INCVR	受信状態変数 V(R) を＋1 する。
(5)	SETVS	送信状態変数 V(S) の値を設定する。
(6)	SETVR	受信状態変数 V(R) の値を設定する。
(7)	INS_ADRS	送信フレームのアドレス値を書き換える。
(8)	INS_NR	送信フレームのN(R)値を書き換える。
(9)	INS_NS	送信フレームのN(S)値を書き換える。
(10)	INS_PF	送信フレームのP/F ビット値を書き換える。
(11)	INS_TYPE	送信フレームのフレーム種別を書き換える。
(12)	INS_CF1	送信フレーム制御フィールドの第１オクテットの値を書き換える。
(13)	INS_CF2	送信フレーム制御フィールドの第２オクテットの値を書き換える。
(14)	INS_FRCF1	送信FRMRフレーム情報フィールド内の第１オクテット値を書き換える。
(15)	INS_FRCF2	送信FRMRフレーム情報フィールド内の第２オクテット値を書き換える。
(16)	INS_FRVS	送信FRMRフレーム情報フィールド内の V(S) 値を書き換える。
(17)	INS_FRVR	送信FRMRフレーム情報フィールド内の V(R) 値を書き換える。
(18)	INS_FRCR	送信FRMRフレーム情報フィールド内の C/Rビット値を書き換える。
(19)	INS_FRWXYZ	送信FRMRフレーム情報フィールド内の WXYZ ビット値を書き換える。
フレーム受信関連		
(20)	RXADRS	受信フレームのアドレス値を読み出す。
(21)	RXNR	受信フレームのN(R)値を読み出す。
(22)	RXNS	受信フレームのN(S)値を読み出す。
(23)	RXPF	受信フレームのP/F ビット値を読み出す。
(24)	RXTYPE	受信フレームのフレーム種別を読み出す。
(25)	RXCF1	受信フレーム制御フィールドの第１オクテットの値を読み出す。
(26)	RXCF2	受信フレーム制御フィールドの第２オクテットの値を読み出す。
(27)	RXFRCF1	受信FRMRフレーム情報フィールド内の第１オクテット値を読み出す。
(28)	RXFRCF2	受信FRMRフレーム情報フィールド内の第２オクテット値を読み出す。
(29)	RXFRVS	受信FRMRフレーム情報フィールド内の V(S) 値を読み出す。
(30)	RXFRVR	受信FRMRフレーム情報フィールド内の V(R) 値を読み出す。
(31)	RXFRCR	受信FRMRフレーム情報フィールド内の C/Rビット値を読み出す。
(32)	RXFRWXYZ	受信FRMRフレーム情報フィールド内の WXYZ ビット値を読み出す。

(1)　SENDT

呼出し形式　　SENDT("NAME")

概　要　　フレームを送信します。（メッセージ・データをそのまま送出）

引数の説明　　"NAME" :フレーム名。

関数値　　0　　:正常終了。
-50　　:トランスペアレント・モード・エラー。
-61　　:フレーム名エラー。

使用例　　SENDT("SPL1")

機能説明　　トランスペアレント・モードでのフレームの送出を行なう関数です。引数で指定した名前のフレームをそのまま送出します。

注意事項　　SENDF関数では、内部の状態変数 V(S),V(R)を N(S),N(R)としてフレーム中に挿入して送出しますが、本関数は一切加工せずに回線上に送出します。

制限事項　　レイヤ2自動モードでは、実行できません。

(2)　SENDF

呼出し形式　　SENDF("NAME",PF)

概　要　　フレームを送信します。（ N(S),N(R),P/Fを挿入して送出）

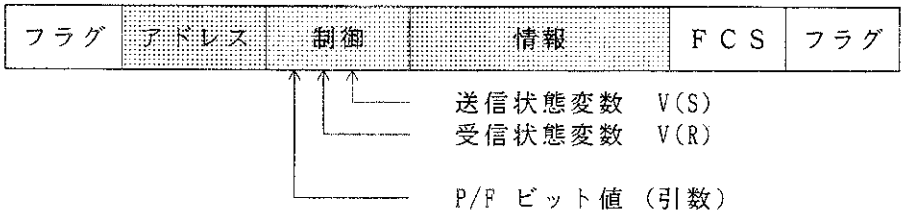
引数の説明　　"NAME" :送出するフレーム名。
PF　　:送出するフレームのP/F ビット値。

関数値　　0　　:正常終了。
-50　　:トランスペアレント・モード・エラー。
-61　　:フレーム名エラー。
-110　　:送出フレーム長エラー。

使用例
```
SENDF("ID_REQ",0)
PF= 1
SENDF("SABME",1)
```

機能説明　　トランスペアレント・モードでのフレームの送出を行なう関数です。引数で指定されたフレーム名のフレームを送出します。その際、引数で指定されたP/F ビット値、送信シーケンス番号N(S)、または受信シーケンス番号N(R)を付加して送信します。
送信状態変数V(S)は、INCVS 関数やSETVS 関数により変更が可能です。
また、受信状態変数V(R)も、同様にINCVS 関数やSETVS 関数により変更が可能です。
SENDF 関数によりフレーム送信する際に付加されるデータを下図に示します。

制限事項　　レイヤ２自動モードでは、実行できません。



(3)　INCVS

呼出し形式　　INCVS( )

概　要　　送信状態変数 V(S) を＋１します。

引数の説明

関数値　　0　　:正常終了。
-50　:トランスペアレント・モード・エラー。

使用例　　INCVS( )

機能説明　　送信状態変数V(S)の値を＋１更新する関数です。
この値は、番号制情報フレーム(Iフレーム）を送出する時にのみ
N(S)値としてフレームに付加されます。

制限事項　　この関数は、レイヤ２自動モードでは実行できません。

(4)　INCVR

呼出し形式　　INCVR( )

概　要　　受信状態変数 V(R) を＋1します。

引数の説明

関数値　　0　　:正常終了。
-50　:トランスペアレント・モード・エラー。

使用例　　INCVR( )

機能説明　　受信状態変数V(R)の値を＋1更新する関数です。
この値は、番号制情報フレーム(Iフレーム）と番号制監視フレーム(Sフレーム）を送出する時にN(R)値として、送信フレームに付加されます。

制限事項　　この関数は、レイヤ2自動モードでは実行できません。

(5)　S E T V S

呼出し形式　　SETVS(VS)

概　要　　送信状態変数 V(S) の値を設定します。

引数の説明　　VS　:設定するV(S)値。
モジュロ８宣言時は 0≦VS≦7
モジュロ１２８宣言時は 0≦VS≦127

関数値　　0　:正常終了。
-50　:トランスペアレント・モード・エラー。
-60　:引数エラー。

使用例
```
SETVS(5)
VS=6
SETVS(VS)
```

機能説明　　送信状態変数V(S)の値を設定する関数です。
この値は、番号制情報フレーム(Iフレーム）を送出するときにのみ
N(S)値として送信フレームに付加されます。

制限事項　　この関数は、レイヤ２自動モードでは実行できません。

(6)　SETVR

呼出し形式　　SETVR(VR)

概　要　　受信状態変数 V(R) の値を設定します。

引数の説明　　VR　:設定するV(R)値。
モジュロ8宣言時は 0≦VR≦7
モジュロ128宣言時は 0≦VR≦127

関数値　　0　:正常終了。
-50　:トランスペアレント・モード・エラー。
-60　:引数エラー。

使用例
```
SETVR(5)
VR=6
SETVR(VR)
```

機能説明　　受信状態変数V(R)の値を設定する関数です。
この値は、番号制情報フレーム(Iフレーム)と番号制監視フレーム(Sフレーム)を送出するときにのみ、N(R)値として送信フレームに付加されます。

制限事項　　この関数は、レイヤ2自動モードでは実行できません。

(7)～(19)　LAPBパラメータ挿入関数群

呼出し形式	[関数名]	[挿入するパラメータ]
(7)	INS_ADRS("NAME",ADRS)	アドレス値
(8)	INS_NR("NAME",NR)	N(R)値
(9)	INS_NS("NAME",NS)	N(S)値
(10)	INS_PF("NAME",PF)	P/F ビット値
(11)	INS_TYPE("NAME",TYPE)	フレーム種別
(12)	INS_CF1("NAME",CF1)	制御フィールド第１オクテット値
(13)	INS_CF2("NAME",CF2)	制御フィールド第２オクテット値
(14)	INS_FRCF1("NAME",FRCF1)	FRMRフレーム情報フィールド内第１オクテット値
(15)	INS_FRCF2("NAME",FRCF2)	FRMRフレーム情報フィールド内第２オクテット値
(16)	INS_FRVS("NAME",FRCVS)	FRMRフレーム情報フィールド内V(S)値
(17)	INS_FRVR("NAME",FRVR)	FRMRフレーム情報フィールド内V(R)値
(18)	INS_FRCR("NAME",FRCR)	FRMRフレーム情報フィールド内C/R ビット値
(19)	INS_FRWXYZ("NAME",FRWXYZ)	FRMRフレーム情報フィールド内WXYXビット値

概　要　　送出フレームの任意のパラメータを書き換えます。

引数の説明

NAME	:フレーム名
ADRS	:アドレス値
NR	:N(R)値
NS	:N(S)値
PF	:P/F ビット値
TYPE	:フレーム種別
CF1	:制御フィールド第１オクテット値
CF2	:制御フィールド第２オクテット値
FRCF1	:FRMRフレーム情報フィールド内第１オクテット値
FRCF2	:FRMRフレーム情報フィールド内第２オクテット値
FRVS	:FRMRフレーム情報フィールド内V(S)値
FRVR	:FRMRフレーム情報フィールド内V(R)値
FRCR	:FRMRフレーム情報フィールド内C/R ビット値
FRWXYZ	:FRMRフレーム情報フィールド内WXYZビット値

関数値

0	:正常終了。
-50	:トランスペアレント・モードエラー。
-60	:引数エラー。
-61	:フレーム名エラー。
-85	:書き換えられるメッセージが不適切。

使用例

```
INS_ADRS("SPL1",1)
SENDF("SPL1",0)
```

機能説明　　メッセージ中の任意のLAPBパラメータを挿入する関数群です。
本関数群は、メッセージの先頭から内容を解釈していき、挿入するLAPBパラメータの位置を探していきます。
挿入するLAPBパラメータの位置が発見されると、そこを引数で指定された値に書き換えます。
このとき挿入するLAPBパラメータの位置が発見されないと”書き換えられるメッセージが不適切である”という意味の関数値を返します。

補足説明　　INS_TYPE("NAME",TYPE) 関数の引数TYPEとフレーム種別は以下の対応となっています。

フレーム種別	TYPE		フレーム種別	TYPE	
	10進	16進		10進	16進
I	0	00	DM	15	0F
RR	1	01	UI	3	03
RNR	5	05	DISC	67	43
REJ	9	09	UA	99	63
SABME	111	6F	FRMR	135	87
SABM	47	2F	XID	175	AF

制限事項　　本関数は、レイヤ２自動モードでは実行できません。

⑳～㉜　LAPBパラメータ読み出し関数群

呼出し形式

[関数名]	[読み出すパラメータ]
⑳ RXADRS( )	アドレス値
㉑ RXNR( )	N(R)値
㉒ RXNS( )	N(S)値
㉓ RXPF( )	P/F ビット値
㉔ RXTYPE( )	フレーム種別
㉕ RXCF1( )	制御フィールド第１オクテット値
㉖ RXCF2( )	制御フィールド第２オクテット値
㉗ RXFRCF1( )	FRMRフレーム情報フィールド内 第１オクテット値
㉘ RXFRCF2( )	FRMRフレーム情報フィールド内 第２オクテット値
㉙ RXFRVS( )	FRMRフレーム情報フィールド内 V(S)値
㉚ RXFRVR( )	FRMRフレーム情報フィールド内 V(R)値
㉛ RXFRCR( )	FRMRフレーム情報フィールド内 C/R ビット値
㉜ RXFRWXYZ( )	FRMRフレーム情報フィールド内 WXYZビット値

概　要　　受信フレームの任意のパラメータを読み出します。

引数の説明

関数値

```
0 ～ 255 :受信フレームのパラメータ。
-50      :トランスペアレント・モード・エラー。
-80      :フレーム未受信エラー。
-82      :読み出すパラメータが存在しない。
```

使用例

```
RECEIVE(0)
ADRS=RXADRS( )
PRINT("ADDRESS=%D＼N",ADRS)
```

機能説明　　受信フレームのLAPBパラメータを読み出す関数です。
フレームを受信した順に読むことができます。
受信したフレームを読み出したい場合は、RECEIVE 関数を実行し、受信フレームを内部に取り込んでから本関数群を実行します。次の受信フレームを読み出したい場合は、再びRECEIVE 関数を実行します。このとき受信フレームが無い場合は、受信待ち状態になります。

注意事項

受信したフレームが無い状態やRECEIVE 関数を実行していない状態で本関数が実行されるとフレーム未受信エラーとなります。
また、本関数は受信したフレームを先頭から解釈していき、LAPBパラメータのある位置を探していきます。
LAPBパラメータのある位置が発見されると、その値を関数値として返し終了します。
しかし、LAPBパラメータが見つからないと”読み出すパラメータが存在しない”という意味のエラー値を返します。

補足説明

㉗～㉜のRXFRで始まるLAPB読み出し関数は、受信したフレームがFRMRフレームで無い場合、”受信フレームがFRMRフレームで無い”という意味の関数値を返します。
RXTYPE( ) 関数で返される関数値とフレーム種別は以下の対応となっています。

フレーム種別	TYPE		フレーム種別	TYPE	
	10進	16進		10進	16進
I	0	00	DM	15	0F
RR	1	01	UI	3	03
RNR	5	05	DISC	67	43
REJ	9	09	UA	99	63
SABME	111	6F	FRMR	135	87
SABM	47	2F	XID	175	AF

制限事項

本関数はトランスペアレント・モードで実行する関数のため、レイヤ２自動モードでは実行できません。

13.3 レイヤ2自動モード用関数

表 13 - 3 レイヤ2自動モード用関数(1/2)

フレーム送信関連	
(1) SENDI	Iフレームを送信する。
(2) SENDPKT	パケットを送信する。
(3) INCPS	送信順序番号 P(S) を+1する。
(4) INCPR	受信順序番号 P(R) を+1する。
(5) SETPS	送信順序番号 P(S) の値を設定する。
(6) SETPR	受信順序番号 P(R) の値を設定する。
(7) INS_GFI	送信パケットのゼネラルフォーマット識別子の値を書き換える。
(8) INS_Q	送信パケットのQビット値を書き換える。
(9) INS_D	送信パケットのDビット値を書き換える。
(10) INS_LCGN	送信パケットの論理チャネルグループ番号を書き換える。
(11) INS_LCN	送信パケットの論理チャネル番号を書き換える。
(12) INS_TYP	送信パケットのパケットタイプ識別子を書き換える。
(13) INS_PR	送信パケットの受信順序番号 P(R) を書き換える。
(14) INS_PS	送信パケットの送信順序番号 P(S) を書き換える。
(15) INS_M	送信パケットのモアデータ表示値を書き換える。
(16) INS_CLL	送信パケットの発呼ユーザアドレス長を書き換える。
(17) INS_CDL	送信パケットの着呼ユーザアドレス長を書き換える。
(18) INS_DA	送信パケットの着呼ユーザアドレスを書き換える。
(19) INS_SA	送信パケットの発呼ユーザアドレスを書き換える。
(20) INS_FL	送信パケットのファシリティ長を書き換える。
(21) INS_F	送信パケットのファシリティを書き換える。
(22) INS_CAUSE	送信パケットの切断／リスタート／リセット原因を書き換える。
(23) INS_DIAG	送信パケットの診断符号を書き換える。
(24) SETDTL	送信パケット内のデータ長を書き換える。
(25) INS_DATA	送信パケット内のデータを書き換える。
リンク関連	
(26) LINKON	リンクの設定を行なう。
(27) LINKOFF	リンクの解放を行なう。
(28) WAIT_LINK	相手リンク設定待ち状態にする。
(29) L_STATUS	リンクが設定されているかを調べる。
(30) SET_BUSY	自局をビジー状態にする。
(31) REL_BUSY	自局のビジー状態を解除する。

表 13 - 3 レイヤ2自動モード用関数(2/2)

フレーム受信関連	
(32) RXGFI	受信パケットのゼネラルフォーマット識別子の値を読み出す。
(33) RXQ	受信パケットのQ ビット値を読み出す。
(34) RXD	受信パケットのD ビット値を読み出す。
(35) RXLCGN	受信パケットの論理チャネルグループ番号を読み出す。
(36) RXLCN	受信パケットの論理チャネル番号を読み出す。
(37) RXTYP	受信パケットのパケットタイプ識別子を読み出す。
(38) RXPR	受信パケットの受信順序番号 P(R) を読み出す。
(39) RXPS	受信パケットの送信順序番号 P(S) を読み出す。
(40) RXM	受信パケットのモアデータ表示値を読み出す。
(41) RXCLL	受信パケットの発呼ユーザアドレス長を読み出す。
(42) RXCDL	受信パケットの着呼ユーザアドレス長を読み出す。
(43) RXDA	受信パケットの着呼ユーザアドレスを読み出す。
(44) RXSA	受信パケットの発呼ユーザアドレスを読み出す。
(45) RXFL	受信パケットのファシリティ長を読み出す。
(46) RXF	受信パケットのファシリティを読み出す。
(47) RXCAUSE	受信パケットの切断／リスタート／リセット原因を読み出す。
(48) RXDIAG	受信パケットの診断符号を読み出す。
(49) RXDTL	受信パケット内のデータ長を読み出す。
(50) RXDATA	受信パケット内のデータを読み出す。

(1) SENDI

呼出し形式　SENDI("NAME")

概　要　Iフレームを送信します。

引数の説明　"NAME" :フレーム名。

関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-61　:フレーム名エラー。
-72　:リンク未設定エラー。
-110　:送出フレーム長エラー。

使用例
```
PH_ACT( )
LINKON( )
SENDI("SPL1")
```

機能説明　レイヤ2自動モードで、Iフレームを送出する関数です。
引数で指定した名前のメッセージをIフレームとして送出します。
メッセージは、あらかじめメッセージ・ビルダで作成する必要があります。作成したメッセージに名前は必ず付けて下さい。
Iフレームを送出するためにはレイヤ2リンクか設定されていなければなりません。リンクの設定はLINKON関数を用いて行なうか、相手局からのリンク設定要求を受け付けて行ないます。

注意事項　本関数でフレームが送出されない場合以下の点をチェックして下さい。

1. レイヤ1は起動しているか。
2. レイヤ2リンクは設定されているか。
3. メッセージ・ビルダで引数で指定した名前のメッセージを作成してあるか。

(2) SENDPKT

呼出し形式　SENDPKT("NAME",LCGN,LCN)

概　要　パケットを送信します。

引数の説明
"NAME" :フレーム名。
LCGN :論理チャネルグループ番号。(0～15)
LCN :論理チャネル番号。(0～255)

関数値
0 :正常終了。
-51 :レイヤ2自動モード・エラー。
-60 :引数エラー。
-61 :フレーム名エラー。
-72 :リンク未設定エラー。
-110 :送出フレーム長エラー。

使用例
```
PH_ACT( )
LINKON( )
LCGN=0
LCN=1
SENDPKT("CR",LCGN,LCN)
```

機能説明　レイヤ2自動モードで、パケツトフレームを送出する関数です。
引数で指定した名前のメッセージをIフレームとして送出します。このとき引数で指定したLCGNとLCN を送出するメッセージに挿入します。また、送出するメッセージのタイプがDTパケットのときは、引数LCGN,LCNで指定された論理チャネルのP(R),P(S) が挿入されます。同様、RR,RNRパケットのときも指定された論理チャネルのP(R)がメッセージ中に挿入されます。
メッセージは、あらかじめメッセージ・ビルダで作成しておく必要があります。作成したメッセージに名前を必ず付けて下さい。
本関数でフレームを送出するためにはレイヤ2リンクが設定されていなければなりません。リンクの設定はLINKON( ) 関数を用いて行なうか、相手局からのリンク設定要求を受け付けて行ないます。

注意事項　本関数でフレームが送出されない場合、以下の点をチェックして下さい。

1. レイヤ1は起動しているか。
2. レイヤ2リンクは設定されているか。
3. メッセージ・ビルダで引数で指定した名前のメッセージを作成してあるか。

(3) INCPS

呼出し形式　INCPS(LCGN,LCN)

概　要　送信順序番号 P(S) を+1 します。

引数の説明　LCGN :論理チャネルグループ番号。(0～15)
LCN :論理チャネル番号。(0～255)

関数値　0 :正常終了。
-51 :レイヤ2自動モード・エラー。
-60 :引数エラー。

使用例
```
INCPS(0,1)
SENDPKT("DT",0,1)
```

機能説明　送信順序番号P(S)を+1 更新する関数です。
引数で指定した論理チャネル(LCGN とLCN)のP(S)値を変更します。
この値は、DTパケットを送出するときにメッセージ中に挿入されます。

制限事項　本関数は、トランスペアレント・モードでは実行できません。

(4) INCPR

呼出し形式　INCPR(LCGN,LCN)

概　要　受信順序番号 P(R) を＋1 します。

引数の説明　LCGN :論理チャネルグループ番号。(0～15)
LCN :論理チャネル番号。(0～255)

関数値　0 :正常終了。
-51 :レイヤ2自動モード・エラー。
-60 :引数エラー。

使用例
```
INCPR(0,1)
SENDPKT("DT",0,1)
```

機能説明　受信順序番号P(R)を＋1 更新する関数です。
引数で指定した論理チャネル（LCGNとLCN)のP(R)値を変更します。
この値は、DT,RR,RNR パケットを送出するときにメッセージ中に挿入されます。

制限事項　本関数は、トランスペアレント・モードでは実行できません。

(5) SETPS

呼出し形式　SETPS(LCGN,LCN,PS)

概　要　送信順序番号 P(S) の値を設定します。

引数の説明
LCGN :論理チャネルグループ番号。(0～15)
LCN :論理チャネル番号。(0～255)
PS :送信順序番号。(0～127)

関数値
0 :正常終了。
-51 :レイヤ2自動モード・エラー。
-60 :引数エラー。

使用例
```
SETPS(0,1,7)
SENDPKT("DT",0,1)
```

機能説明　送信順序番号P(S)の値を設定する関数です。
引数で指定した論理チャネル（LCGNとLCN)のP(S)値を変更します。
この値は、DTパケットを送出するときのみメッセージ中に挿入されます。

制限事項　本関数は、トランスペアレント・モードでは実行できません。

(6) SETPR

呼出し形式　SETPR(LCGN,LCN,PR)

概　要　受信順序番号 P(R) の値を設定します。

引数の説明
LCGN :論理チャネルグループ番号。(0～15)
LCN :論理チャネル番号。(0～255)
PR :受信順序番号。(0～127)

関数値
0 :正常終了。
-51 :レイヤ2自動モード・エラー。
-60 :引数エラー。

使用例
```
SETPR(0,1,3)
SENDPKT("DT",0,1)
```

機能説明　受信順序番号P(R)の値を設定する関数です。
引数で指定した論理チャネル（LCGNとLCN)のP(R)値を変更します。
この値は、DT,RR,RNR パケットを送出するときのみメッセージ中に挿入されます。

制限事項　本関数は、トランスペアレント・モードでは実行できません。

(7)～(25) X.25パラメータ挿入関数群

呼出し形式

	[関数名]	[挿入するパラメータ]
(7)	INS_GFI("NAME",GFI)	ゼネラルフォーマット識別子(GFI)
(8)	INS_Q("NAME",Q)	Q ビット
(9)	INS_D("NAME",D)	D ビット
(10)	INS_LCGN("NAME",LCGN)	論理チャネルグループ番号(LCGN)
(11)	INS_LCN("NAME",LCN)	論理チャネル番号(LCN)
(12)	INS_TYP("NAME",TYP)	パケットタイプ識別子(TYP)
(13)	INS_PR("NAME",PR)	受信順序番号 (P(R))
(14)	INS_PS("NAME",PS)	送信順序番号 (P(S))
(15)	INS_M("NAME",M)	M ビット
(16)	INS_CLL("NAME",CLL)	発呼ユーザアドレス長
(17)	INS_CDL("NAME",CDL)	着呼ユーザアドレス長
(18)	INS_DA("NAME",N,DA)	着呼ユーザアドレス(N番目)
(19)	INS_SA("NAME",N,SA)	発呼ユーザアドレス(N番目)
(20)	INS_FL("NAME",FL)	ファシリティ長
(21)	INS_F("NAME",N,F)	ファシリティ(Nオクテット目)
(22)	INS_CAUSE("NAME",CAUSE)	原因
(23)	INS_DIAG("NAME",DIAG)	診断符号
(24)	SET_DTL("NAME",LEN)	データ長
(25)	INS_DATA("NAME",N,DATA)	データ(Nオクテット目)

概　要　　送出フレームの任意のパラメータを書き換えます。

引数の説明

NAME	:フレーム名
GFI	:ゼネラルフォーマット識別子(GFI)
Q	:Q ビット
D	:D ビット
LCGN	:論理チャネルグループ番号(LCGN)
LCN	:論理チャネル番号(LCN)
TYP	:パケットタイプ識別子
PR	:受信順序番号
PS	:送信順序番号
M	:M ビット
CLL	:発呼ユーザアドレス長
CDL	:着呼ユーザアドレス長
N	:N 番目を指定
DA	:着呼ユーザアドレス
SA	:発呼ユーザアドレス
FL	:ファシリティ長
F	:ファシリティ
CAUSE	:原因
DIAG	:診断符号
LEN	:データ長
DATA	:データ

関数値　　0　　:正常終了。
-60　:引数エラー。
-61　:フレーム名エラー。
-85　:書き換えられるメッセージが不適切。

使用例　　INS_GFI("CR",1)
SENDPKT("CR",0,1)

機能説明　　メッセージ中の任意のパラメータを書き換える関数群です。
本関数は、メッセージの先頭から内容を解釈していき、挿入するパラメータの位置を探します。
挿入するパラメータの位置が発見されるとそこを引数で指定された値に書き換えます。このとき該当するパラメータが複数の位置に渡って存在する場合、引数N でその位置を確定させる必要があります。（INS_DA, INS_SA, INS_F, INS_DATA関数などがこれに当てはまります。）
また、メッセージ内容を解釈していった結果、挿入するパラメータの位置が存在しない場合、" 書き換えられるメッセージが不適切" という意味の関数を返します。

補足説明　　INS_TYP("NAME",TYP) 関数で、引数TYP とパケット名は以下の対応となっています。

パケット名	TYPE		パケット名	TYPE	
	10進	16進		10進	16進
CR	11	0B	SF	255	FF
CN	11	0B	DT	0	00
CA	15	0F	RR	1	01
CC	15	0F	RNR	5	05
CQ	19	13	RQ	27	1B
CI	19	13	RI	27	1B
CF	23	17	RF	31	1F
SQ	251	FB	IT	35	23
SI	251	FB	IF	39	27

SET_DTL("NAME",LEN) 関数は、挿入関数ではありませんが
INS_DATAと密接な関係があります。
INS_DATA関数がパケット中のデータを変えるのに対してSET_DATA関数は、そのデータ長を変える関数です。

(26) LINKON

呼出し形式　LINKON( )

概　要　リンクの設定を行ないます。

引数の説明

関数値
```
0    :正常終了。
-51  :レイヤ2自動モード・エラー。
-100 :リンクの設定が拒否された。
-131 :SABM/SABMEコマンドをN2回送出したが応答がない。
```

使用例
```
PH_ACT( )
LINKON( )
SENDI("SPL1")
```

機能説明　レイヤ2リンクの設定を行なう関数です。
このとき、アドレスとモジュロは宣言文によりあらかじめ宣言しておく必要があります。
レイヤ2リンクが設定されていないと、SENDI, SENDPKT関数でフレームを送出することはできません。
すでにレイヤ2リンクが設定されている場合は、再設定となります。
レイヤ2リンクを管理する他の関数として以下のものがあります。

```
LINKOFF( )      : リンクの解放をする。
WAIT_LINK(SEC)  : リンクの設定待ち状態にする。
L_STATUS( )     : リンクの状態を調べる。
```

制限事項　本関数は、レイヤ2自動モードで実行する関数のため、トランスペアレント・モードでは実行できません。

(27) LINKOFF

呼出し形式　LINKOFF( )

概　要　リンクの解放を行ないます。

引数の説明

関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-102 :レイヤ2リンクが設定されていない。
-131 :DISCコマンドをN2回送出したが応答がない。

使用例
```
PH_ACT( )
LINKON( )
LINKOFF( )
```

機能説明　レイヤ2リンクを解放する関数です。
レイヤ2リンクが設定されていない状態で本関数が呼ばれた場合は、"レイヤ2リンクが設定されていない"という意味の関数値が返され終了します。

制限事項　本関数は、レイヤ2自動モードで実行する関数のため、トランスペアレント・モードでは実行できません。

㉘ WAIT_LINK

呼出し形式　WAIT_LINK(SEC)

概　要　相手リンク設定待ち状態にします。

引数の説明　SEC　:相手からリンクが設定されるのを待つ時間。
（100msec 単位）
（0 ≦SEC ≦16777215）

関数値　0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-60　:引数エラー。
-125 :タイムアウト。

使用例
```
PH_ACT( )
WAIT_LINK(3000)
SENDI("SPL1")
```

機能説明　レイヤ2リンクが設定されるまで待つ関数です。
すでにレイヤ2リンクが設定されている場合には、正常終了を示す関数値を返して終了します。
レイヤ2リンクを待つ時間は引数で指定できます。
引数で指定された時間待ってもレイヤ2リンクが設定されない場合は”タイムアウト”を示す関数値を返して終了します。

制限事項　本関数は、レイヤ2自動モードで実行する関数のため、トランスペアレント・モードでは実行できません。

(29) L_STATUS

呼出し形式　L_STATUS( )

概　要　リンクが設定されているかを調べます。

引数の説明

関数値
0　:リンクが設定されていない。
1　:リンクが設定されている。
-51　:レイヤ2自動モード・エラー。

使用例
```
PH_ACT( )
PRINT("LINK STATUS=%D\N",L_STATUS( ))
LINKON( )
PRINT("LINK STATUS=%D\N",L_STATUS( ))
```

機能説明　レイヤ2リンクが設定されているかどうかを調べる関数です。レイヤ2リンクが設定されている場合には、関数値として0が返されます。逆にレイヤ2リンクが設定されていない場合には、関数値として1が返されます。

制限事項　本関数は、レイヤ2自動モードで実行する関数のため、トランスペアレント・モードでは実行できません。

㉚ SET_BUSY

呼出し形式　SET_BUSY( )

概　要　自局をビジー状態にします。

引数の説明

関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-102 :リンクが設定されていない。

使用例
```
PH_ACT( )
LINKON( )
SET_BUSY( )
```

機能説明　自局をビジー状態にする関数です。
本関数を実行するためには、レイヤ2リンクが設定されている必要があります。
ビジー状態を解除するときには、REL_BUSY関数を使用します。

制限事項　本関数は、レイヤ2自動モードで実行する関数のため、トランスペアレント・モードでは実行できません。

(31) REL_BUSY

呼出し形式　REL_BUSY( )

概　要　自局のビジー状態を解除します。

引数の説明

関数値
0　:正常終了。
-51　:レイヤ2自動モード・エラー。
-102 :リンクが設定されていない。
-109 :ビジー状態になっていない。

使用例
```
PH_ACT( )
LINKON( )
SET_BUSY( )
WAIT(30)
REL_BUSY( )
```

機能説明　自局のビジー状態を解除する関数です。
本関数はSET_BUSY関数で設定されたビジー状態を解除するときに使用します。

制限事項　本関数は、レイヤ2自動モードで実行する関数のため、トランスペアレント・モードでは実行できません。

㉜～㊿ X.25パラメータ読み出し関数群

呼出し形式

[関数名]	[読み出すパラメータ]
㉜ RXGFI( )	ゼネラルフォーマット識別子(GFI)
㉝ RXQ( )	Q ビット
㉞ RXD( )	D ビット
㉟ RXLCGN( )	論理チャネルグループ番号(LCGN)
㊱ RXLCN( )	論理チャネル番号(LCN)
㊲ RXTYP( )	パケットタイプ識別子
㊳ RXPR( )	受信順序番号P(R)
㊴ RXPS( )	送信順序番号P(S)
㊵ RXM( )	M ビット
㊶ RXCLL( )	発呼ユーザアドレス長
㊷ RXCDL( )	着呼ユーザアドレス長
㊸ RXDA(N)	着呼ユーザアドレス(N番目)
㊹ RXSA(N)	発呼ユーザアドレス(N番目)
㊺ RXFL( )	ファシリティ長
㊻ RXF(N)	ファシリティ(Nオクテット目)
㊼ RXCAUSE( )	原因
㊽ RXDIAG( )	診断符号
㊾ RXDTL( )	データ長
㊿ RXDATA(N)	データ(Nオクテット目)

概　要

受信フレームの任意のパラメータを読み出します。

引数の説明

N　　:N 番目を指定。

関数値

0 ～ 255 :読み出したパラメータの値。
-60　　:引数エラー。
-80　　:フレーム未受信エラー。
-82　　:読み出すパラメータが受信フレーム中に存在しない。

使用例

```
RECEIVE(0)
GFI=RXGFI( )
PRINT("GFI=%D ＼N",GFI)
```

機能説明

受信フレームのX.25パラメータを読み出す関数です。
フレームを受信した順に読むことができます。
受信フレームを読み出したい場合はRECEIVE 関数を実行し、受信フレームを内部に取り込んでから本関数群を実行します。次の受信フレームを読み出したい場合は、再びRECEIVE 関数を実行します。このとき受信フレームが無い場合は、受信待ち状態になります。
本関数は受信フレームの先頭から内容を解釈していき、該当するX.25パラメータを関数値として直します。該当のX.25パラメータが複数存在する(RXSA,RXDA,RXF,RXDATA など) 場合、引数N で何番目のパラメータを読み出すのかを指定する必要があります。このときNが不適切であると引数エラーとなります。
また、メッセージの内容を解釈していった結果、読み出すパラメータが存在しない場合、”読み出すパラメータが受信フレーム中に存在しない”という意味の関数値を返します。

補足説明　RXTYP( )関数で返される関数値とパケット名は、以下の対応となっています。

パケット名	関数値		パケット名	関数値	
	10進	16進		10進	16進
CR	11	0B	SF	255	FF
CN	11	0B	DT	0	00
CA	15	0F	RR	1	01
CC	15	0F	RNR	5	05
CQ	19	13	RQ	27	1B
CI	19	13	RI	27	1B
CF	23	17	RF	31	1F
SQ	251	FB	IT	35	23
SI	251	FB	IF	39	27

13.4　音声、BERT用関数

表 13 - 4　音声、BERT用関数

(1)　SOUND_ON	ヘッドホン、ブザーから音を発生させる。
(2)　SOUND_OFF	ヘッドホン、ブザーから出ていた音を止める。
(3)　TONE	ブザーから発生される音色を変える。
(4)　VOLUME	ヘッドホン、ブザーから出ていた音量を変える。
(5)　SET_PRBS	BERTで使用する擬似ランダム・パターンを指定する。
(6)　SET_WORD	BERTで使用するワード・パターンを指定する。
(7)　BERT_ON	エラー・レート試験を行なう。

次ページ以降に各関数の説明をします。

(1) SOUND_ON

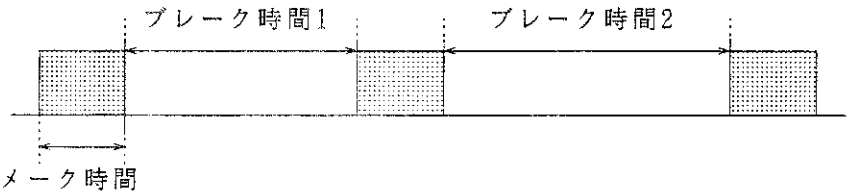
呼出し形式　SOUND_ON(TYPE)

概　要　ヘッドホン、ブザーより音を発生させます。

引数の説明

TYPE	出力先	周波数	ﾒｰｸ時間	ﾌﾞﾚｰｸ時間 1	ﾌﾞﾚｰｸ時間 2
1	ブザー	1kHz	0.125秒	0.125秒	∞
2	ブザー	ワンブルTone	1 秒	2 秒	−
3	ブザー	ワンブルTone	0.5 秒	0.5 秒	−
4	ブザー	ワンブルTone	0.25 秒	0.25 秒	2.25 秒
5	ブザー	ワンブルTone	連 続		
6	ブザー	800HzまたはﾜﾝﾌﾞﾙTone	連 続		
7	ブザー	ワンブルTone	0.5 秒	1 秒	−
8	ブザー	1kHz	0.25 秒	−	−
9	ブザー	1kHz	0.125秒	−	−
10	ﾍｯﾄﾞﾎﾝ	400Hz	連 続		
11	ﾍｯﾄﾞﾎﾝ	400Hz	0.125秒	0.125秒	−
12	ﾍｯﾄﾞﾎﾝ	400Hz/16Hz	1 秒	2 秒	−
13	ﾍｯﾄﾞﾎﾝ	400Hz	0.5 秒	0.5 秒	−
14	ﾍｯﾄﾞﾎﾝ	400Hz	0.25 秒	0.25 秒	−
15	ﾍｯﾄﾞﾎﾝ	400Hz/16Hz	0.5 秒	∞	−
16	ﾍｯﾄﾞﾎﾝ	400Hz	0.25 秒	0.25 秒	0.25 秒
17	ﾍｯﾄﾞﾎﾝ	250Hz	連 続		
18	ﾍｯﾄﾞﾎﾝ	1kHz	連 続		
19	ﾍｯﾄﾞﾎﾝ	1kHz	0.1 秒	∞	−
20	ﾍｯﾄﾞﾎﾝ	Bチャンネル音声			

ただし、メーク時間、ブレーク時間1 、ブレーク時間2 は以下のように定義します。



関数値

0 :正常終了。
-60 :引数エラー。

使用例

```
PH_ACT( )
SOUND_ON(1)
```

機能説明

ヘッドホン、ブザーから音を発生させる関数です。
TYPEが0～9までは内蔵のブザーから音が発生します。また、10～20ではヘッドホーンから音が発生します。
TYPEが20のときはハンドセットにより、相手からの音声を聞くことができるとともに、相手に音声を伝えることもできます。
音を制御する関数として以下の関数があります。

```
SOUND_OFF( )          :音の発生をとめる。
TONE(TYP)             :ブザー音のトーンを変える。
VOLUME(TYPE,VOL)      :音量を変える。
```

制限事項

音は、レイヤ1が起動しているときのみ発生します。
また、ヘッドホンとブザーの両方から同時に音を発生させることはできません。

(2) SOUND_OFF

呼出し形式　SOUND_OFF( )

概　要　ヘッドホン/ブザーから出ていた音を止めます。

引数の説明

関数値　0　:正常終了。

使用例
```
PH_ACT( )
SOUND_ON(1)
WAIT(50)
SOUND_OFF( )
```

機能説明　ヘッドホン/ブザーから出ていた音を止める関数です。

(3) TONE

呼出し形式　TONE(TYP)

概　要　ブザーから発生される音色を変えます。

引数の説明

TYP	周波数
0	1.0kHzまたは1.0kHzの16Hzワンブル
1	1.0kHzまたは1.3kHzの16Hzワンブル
2	0.8kHzまたは1.0kHzの16Hzワンブル
3	0.8kHzまたは1.0kHzの 8Hzワンブル
4	0.5kHzまたは0.65kHzの16Hzワンブル
5	0.4kHzまたは0.5kHzの16Hzワンブル
6	0.4kHzまたは0.5kHzの 8Hzワンブル

関数値　0　:正常終了。
-60　:引数エラー。

使用例
```
PH_ACT( )
TONE(2)
SOUND_ON(1)
```

機能説明　ブザーから発生する音色を変える関数です。
SOUND_ON(TYPE)関数で、TYPEが1～9でブザーから音を出すときに有効な関数です。

(4) VOLUME

呼出し形式　VOLUME(TYP,VOL)

概　要　マイク/ヘッドホン/ブザーから出ている音量を変えます。

引数の説明
TYP 0 :マイクの音量を変える。
TYP 1 :ヘッドホンの音量を変える。
TYP 2 :ブザーの音量を変える。
VOL :音量
TYP 0 :0≦VOL≦63（マイク）
TYP 1 :0≦VOL≦63（ヘッドホン）
TYP 2 :0≦VOL≦ 4（ブザー）

関数値
0 :正常終了。
-60 :引数エラー。

使用例
```
PH_ACT( )
VOLUME(2,2)
SOUND_ON(1)
```

機能説明　マイク/ヘッドホン/ブザーから出ている音量を変える関数です。
マイクの音量を変えるときは、TYPに0を指定します。このときは、VOLに0～63の値を指定します。
ヘッドホンから出ている音量を変えるときはTYPに1を指定します。このときは、VOLに0～63の値を指定します。
ブザーから出ている音量を変えるときはTYPに2を指定します。このときは、VOLに0～4値を指定します。
VOL の数字が大きいほど音量は大きくなります。

(5) SET_PRBS

呼出し形式　SET_PRBS(PRBS)

概　要　BERTで使用する擬似ランダムのパターンを指定します。

引数の説明

PRBS	パターン
1	7段の擬似ランダム
2	9段の擬似ランダム
3	10段の擬似ランダム
4	11段の擬似ランダム
5	15段の擬似ランダム
6	17段の擬似ランダム
7	19段の擬似ランダム
8	20段の擬似ランダム
9	23段の擬似ランダム

関数値　0　:正常終了。
-60　:引数エラー。

使用例

```
SET_PRBS(1)
CHECK=1000000
ERR=BERT_ON(CHECK)
PRINT("CHECK BITS=%D ERROR BITS=%D＼N",CHECK,ERR)
```

機能説明

BERTで使用する擬似ランダムのパターンを指定する関数です。
実際のエラー・レート試験はBERT_ON 関数で行ないます。
擬似ランダム・パターンの生成多項式を以下に示します。

基準パターン :2^N-1

段数N	生成多項式
7	X^7+X^6+1
9	X^9+X^5+1
10	$X^{10}+X^7+1$
11	$X^{11}+X^9+1$
15	$X^{15}+X^{14}+1$
17	$X^{17}+X^{14}+1$
19	$X^{19}+X^5+X^2+X^1+1$
20	$X^{20}+X^3+1$
23	$X^{23}+X^{18}+1$

(6) SET_WORD

呼出し形式　SET_WORD(WORD)

概　要　BERTで使用するワード・パターンを指定します。

引数の説明　WORD　:ワード・パターン(0～65535)。

関数値　0　:正常終了。
-60　:引数エラー。

使用例
```
SET_WORD(1234)
CHECK=1000000
ERR=BERT_ON(CHECK)
PRINT("CHECK BITS=%D ERROR BITS=%D＼N",CHECK,ERR)
```

機能説明　BERTで使用するワード・パターンを指定する関数です。
実際のエラー・レート試験はBERT_ON関数で行ないます。

(7) BERT_ON

呼出し形式　BERT_ON(BITS)

概　要　BERT試験を行ないます。

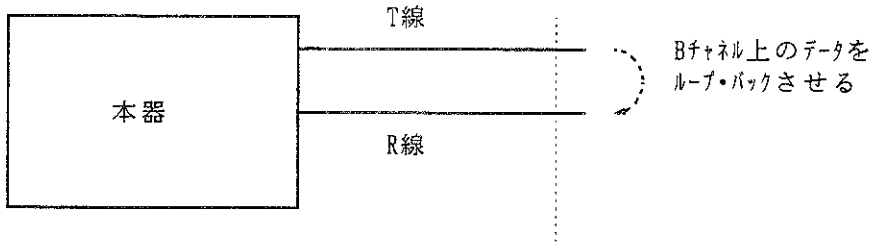
引数の説明　BITS　:BERT 試験を行なうビット数(0～16777215)。

関数値　0～65535:読み出したデータ。
-60　:引数エラー。
-165　:同期外れ。
-170　:カウンタのオーバフロー。

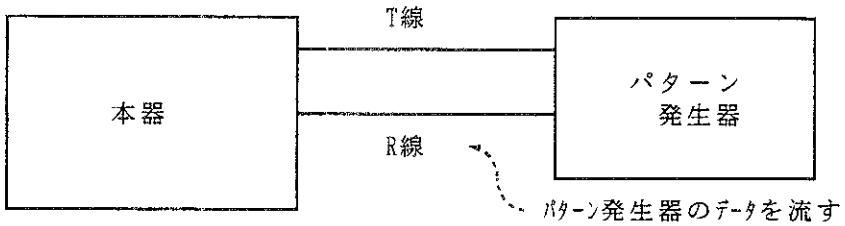
使用例
```
SET_PRBS(1)
CHECK=1000000
ERR=BERT_ON(CHECK)
PRINT("CHECK BITS=%D ERROR BITS=%D＼N",CHECK,ERR)
```

機能説明　引数BITSビットのエラー・レート試験を行ないます。
この引数BITSの試験が終了すると関数値としてエラービット数を返します。
引数BITSビットとエラー・ビット数からエラー・レートが分かります。BERT試験を行なう環境として以下の二つのどちらかの接続が必要になります。

① ループ・バック試験



② パターン発生器を使用した試験



①のループ・バック試験は、相手局またはエント・ユーザなどでBERT試験をしているBチャネルをループ・バックさせる必要があります。

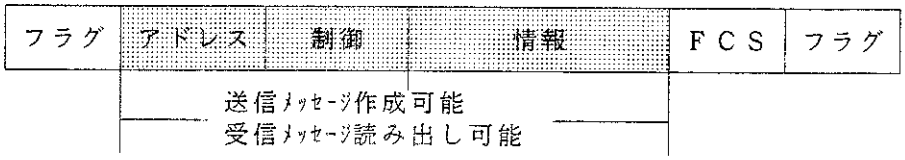
②のパターン発生器を使用した試験では、相手局から本器で設定したパターンと同じものを発生させる必要があります。
本関数が実行されると、まずパターンの同期をとりにいきます。ある一定時間たっても同期がとれない場合、同期外れエラーとなります。
またエラー数が多く、内部のエラー・カウンタがオーバ・フローすると、"カウンタがオーバ・フローした"という意味の関数値を返して終了します。

14. シミュレーションを使いこなすために

14.1 トランスペアレント・モードとレイヤ2自動実行モードの相違

(1) トランスペアレント・モード

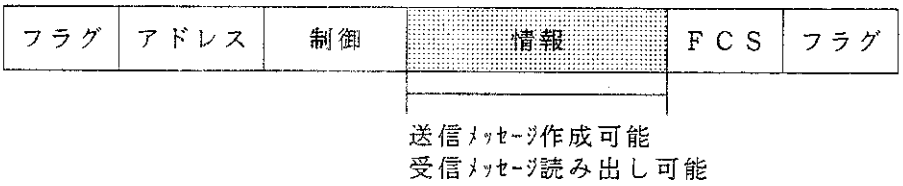
トランスペアレント・モードは、レイヤ2 レベルでシミュレーションを行ないたい場合に指定します。
フレーム中の以下の部分についてメッセージを作成したり、受信したフレームの内容を読み出したりすることができます。



トランスペアレント・モードでフレームを送出する場合、送信状態変数V(S)や受信状態変数V(R)の値を自己管理する必要があります。

(2) レイヤ2 自動実行モード

レイヤ2 自動実行モードとは、レイヤ3 レベルでシミュレーションを行ないたい場合に指定します。
従ってレイヤ2 レベルのフレームのやりとり(RR 送信、送・受信シーケンス番号の管理、TEI 管理手順など) を意識することなくレイヤ3 レベルのシミュレーションを行なうことができます。
メッセージを作成したり、受信したフレームの内容を読み出せる領域は情報部だけです。(下図参照)



(3) 使い分け

トランスペアレント・モードは、レイヤ2 レベルのソフトのデバックを行なう場合やレイヤ2 で異常シーケンス(V(S),V(R)の値が正しくないなど)を起こしたい場合、異常フレーム(SAPI 値不適切など) を送出したい場合などに用います。
レイヤ2 は正常シーケンスに指定し、レイヤ3 のみをシミュレーションしたい場合にはレイヤ2 自動実行モードを使用します。ただし、このレイヤ2 自動実行モードではレイヤ2 レベルで異常フレーム、および異常シーケンスを起こすことはできません。
目的に応じて使い分けて下さい。

14.2 タイマの使用方法

決められた時間だけしか、期待したフレームを待ちたくない場合にタイマを使用します。
タイマを使用するためには、以下の関数を使用します。

RECEIVE(ﾀｲﾏ 番号)	:フレーム、タイムアウト・イベント、他チャネルからのイベント受信用の関数。
T_START(ﾀｲﾏ番号､ﾀｲﾑｱｳﾄ値)	:タイマ番号のタイマを起動させる。
T_STOP(ﾀｲﾏ番号)	:タイマ番号のタイマを停止させる。

[例1]

10秒ごとにシミュレーション画面に"TIME OUT!!"と表示させるプログラムについて示します。

```
FUNC MAIN( )                                  …関数の始まり
     PRINT("***  TIMER1. PRG  ***＼N")
     WHILE(1)                                 …永久ループの始まり

       T_START(1,10)                          …ﾀｲﾏ番号1のﾀｲﾏを起動
       RECEIVE(1)                             …ﾀｲﾏ番号1で受信待ち
       PRINT("TIMEOUT!!＼N")                  …メッセージの表示

     END                                      …永久ループの終り
RETURN                                        …関数の終り
```

上記のプログラムで、RECEIVE(1)をRECEIVE(2)にするとタイマ番号2 のタイマは起動されていないので、上記のプログラムのように10秒ごとに表示することはありません。つまり、RECEIVE(タイマ番号)関数は引数であるタイマ番号のタイムアウト・イベントのみを監視していることになります。従って、引数以外のタイマ番号のタイマがタイムアウトしたとしても、この関数に対して何ら影響を与えません。
しかし、RECEIVE(0)のように引数タイマ番号が0 のときは特別な動きをします。RECEIVE(0)は、すべてのタイマのタイムアウト・イベントを受信します。また、このときREAD_TIMER( ) 関数を用いることにより、どのタイマがタイムアウトしたのかが分かります。タイマの停止はT_STOP(タイマ番号) 関数を使用することにより、引数のタイマ番号を持つタイマが停止します。

[例2]

複数のタイマを起動させ、タイムアウトしたタイマの番号を表示させるプログラムを以下に示します。
受信関数にはRECEIVE(0)を用い、すべてのタイムアウト・イベントの受信を許可します。

```
FUNC MAIN( )
     PRINT("***  TIMER2. PRG  *** ＼N")
     T_START(1,10)                          …タイマ番号1のタイマを起動(タイムアウト値10秒)
     T_START(2,12)                          …タイマ番号2のタイマを起動(タイムアウト値12秒)
     T_START(3,14)                          …タイマ番号3のタイマを起動(タイムアウト値14秒)

     WHILE

       RECEIVE(0)                           …タイムアウトしたタイマ番号をTIMER変数に代入
       TIMER=READ_TIMER( )                  …タイムアウトしたタイマ番号の表示
       PRINT("TIMER ID=%D＼N",TIMER)

     END

RETURN
```

実行結果

```
     ***  TIMER2. PRG  ***
     TIMER ID=1
     TIMER ID=2
     TIMER ID=3
```

[例3]

タイムアウト値10秒のタイマを起動させ、タイムアウトすると"TIMEOUT!!"と表示し、タイムアウトする前にフレームを受信すると"RECEIVE FRAME" と表示するプログラムを以下に示します。

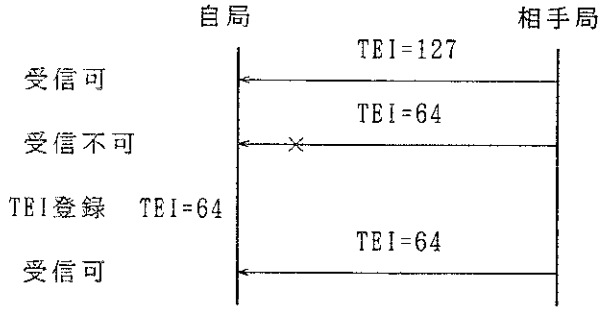
```
FUNC MAIN( )
     T_START(1,10)                  …ﾀｲﾏ番号1(ﾀｲﾑｱｳﾄ10秒)を起動
     RET=RECEIVE(1)                 …ﾀｲﾏ番号1のｲﾍﾞﾝﾄとﾌﾚｰﾑ受信を待つ
     IF RET==0 THEN
        PRINT("TIMEOUT!!")          …ﾀｲﾑｱｳﾄの場合"TIMEOUT"を表示
     ELSE
       PRINT("RECEIVE FRAME")       …ﾀｲﾑｱｳﾄ以外の場合"RECEIVE FRAME"を表示
     END
RETURN
```

14.3 本器でのSAPI, TEI 管理（レイヤ2自動実行モード）（Dチャネル・シミュレーション）

(1) TEI 値の登録

① 登録

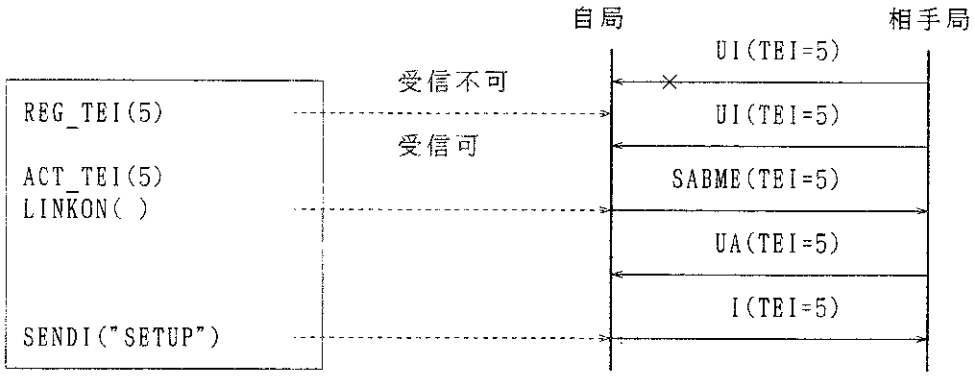
TEI 値を登録すると登録したTEI 値のフレームを受信することができるとともに、登録したTEI 値のUIコマンド、XID コマンド、XID レスポンスが送出可能になります。ただし、TEI=127 は最初から登録されているので登録する必要はありません。また、NTモードの場合、0 ～63は最初から登録済です。



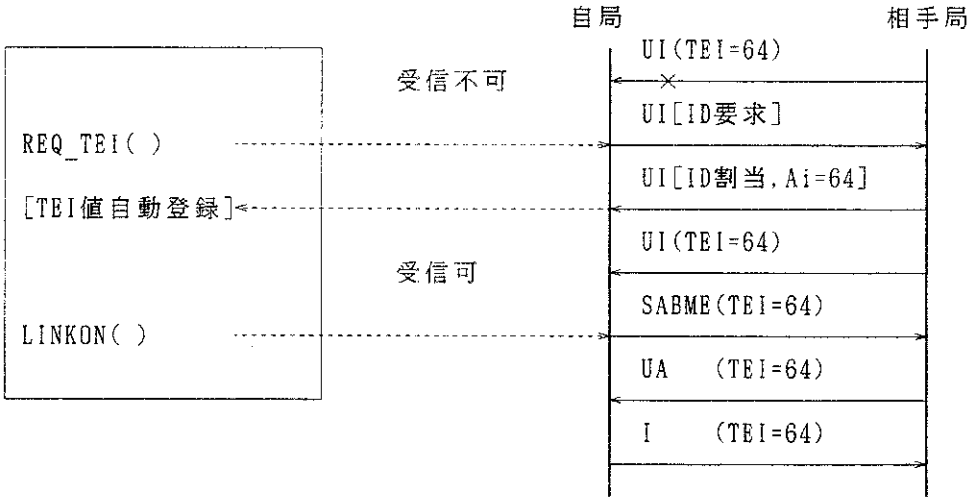
TEI の登録は、REG_TEI( )関数を用います。(12.3 節(48)を参照)
REG_TEI( )関数によるTEI 登録は、非自動割当端末をシミュレートするとき以外は使用しません。
TEI 登録は、TEI 割当手順により新たなTEI が割り当てられたときに自動的に行なわれます。

② 登録例

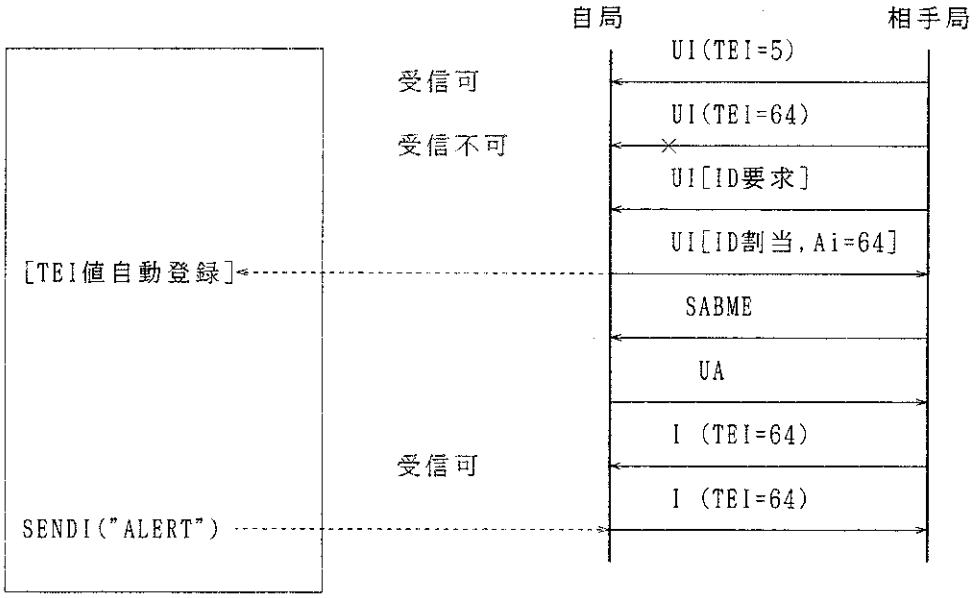
［例1］ 非自動割当端末のシミュレート



［例2］ 自動割当端末のシミュレート



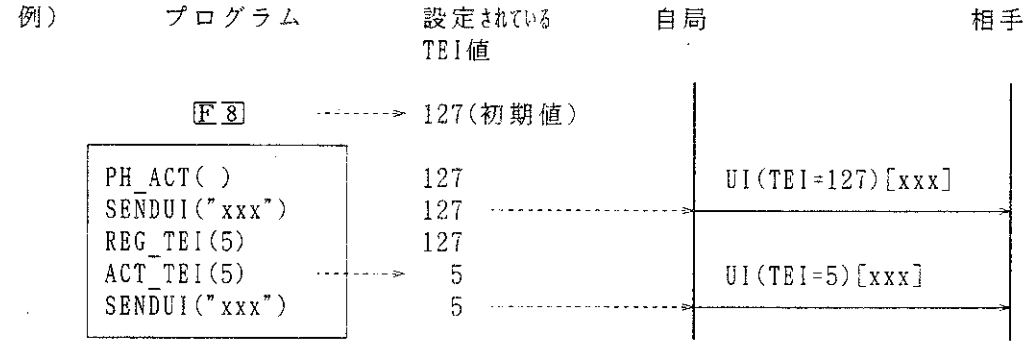
[例3] 網側のシミュレート



シミュレーション・モードを宣言文によりNTと指定するとTEI 値の0 ～63と127 は登録済の状態になります。
これらのTEI 値をもったUIフレームは、シミュレーションが開始してレイヤ1 が起動されるとすぐ受信可能となります。

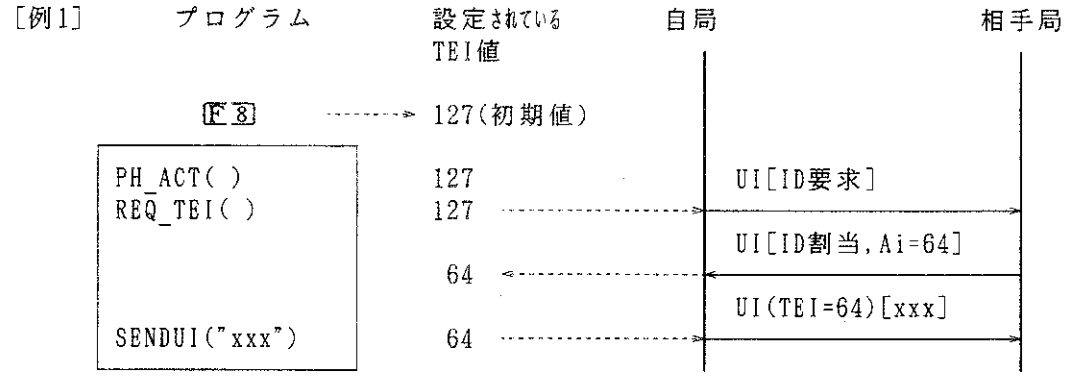
③ 設定

TEI 値を設定するとそのTEI 値を持つフレームが次回からの送信関数により送出されます。
TEI 値の設定はACT_TEI( )関数を用いて行ないます。設定できるTEI 値は、登録されているTEI 値だけです。
非自動端末がTEI 値5 でフレームを送出するプログラムの例を以下に示します。

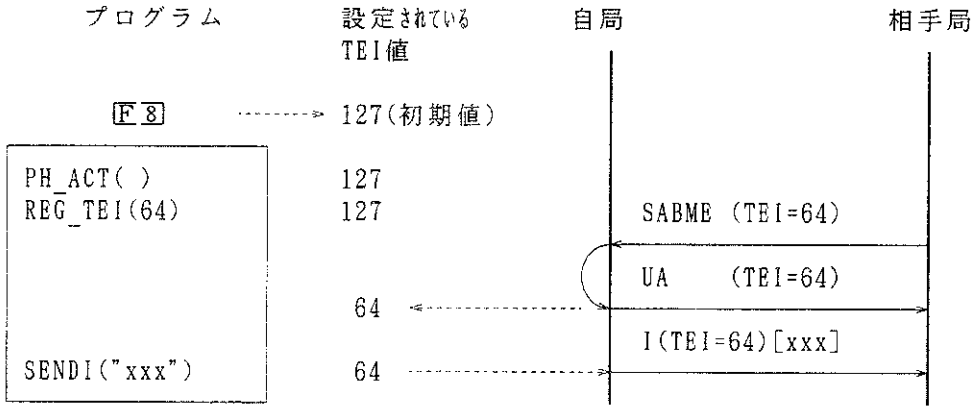


TEI 値はACT_TEI( )関数で設定する以外に下記の条件のときに自動設定されます。
- REQ_TEI( )関数により、TEI 割当要求を起動し、網からTEI 値を割当てられた場合
- レイヤ2 リンクの設定が起きた場合
  - LINKON( ) 関数によりリンク設定をした場合
  - 相手局からリンクの設定をした場合

以下にTEI 値自動設定の例を示します。

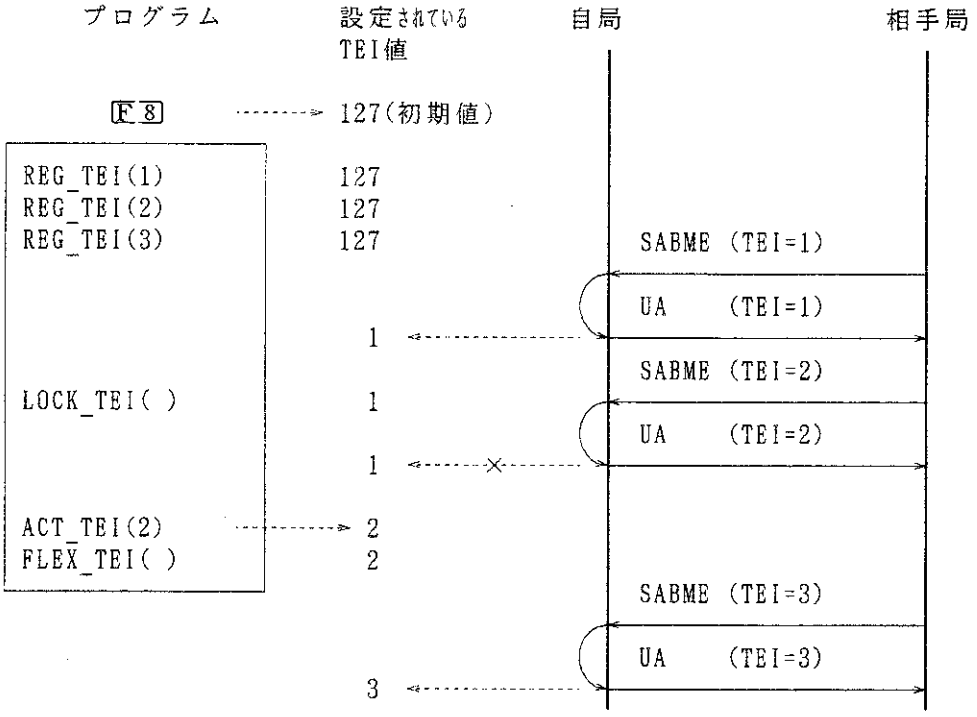


［例2］



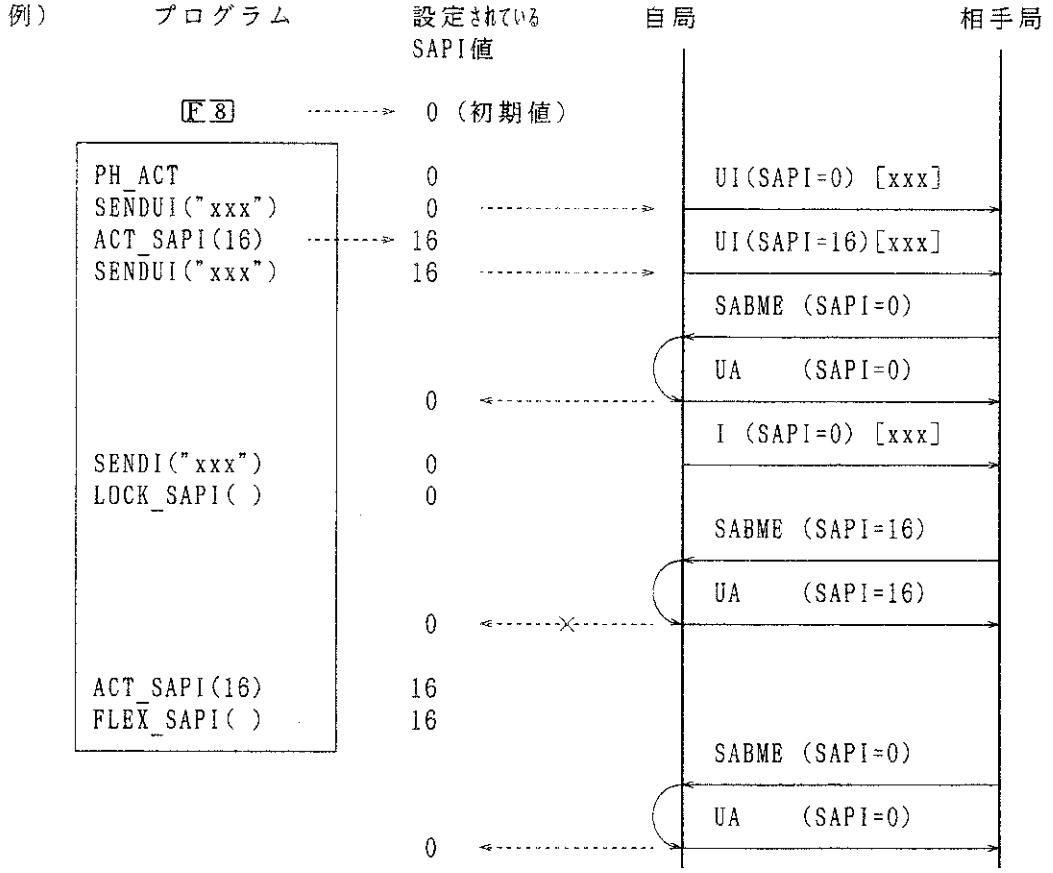
この自動設定を禁止したい場合はLOCK_TEI( )関数を使用します。これによりTEI値変更不可状態になります。
これはTEI の自動設定を禁止するだけで、ACT_TEI( )関数によるTEI の設定変更は禁止されていません。
このTEI 値変更不可状態は、FLEX_TEI( )関数により解除できます。

例）



(2) SAPI値の設定

SAPI値には、0 、16、63の3 種類が登録されています。この3 種類のどれかを設定することにより、そのSAPI値でフレームを送出することができます。
SAPI値の設定には、ACT_SAPI関数を用います。
また、レイヤ2 リンクの設定が生じた場合には、そのSAPI値が新たに自動設定されます。この自動設定を禁止したい場合は、LOCK_SAPI( ) 関数を用います。これによりSAPI変更不可状態になります。しかし、これは相手局からのレイヤ2 リンク設定によるSAPI値の自動設定を禁止するだけでACT_SAPI( ) 関数によるSAPI値の変更は可能です。このSAPI値変更不可状態は、FLEX_SAPI( ) 関数により解除することができます。



(3) リンク番号の使い方

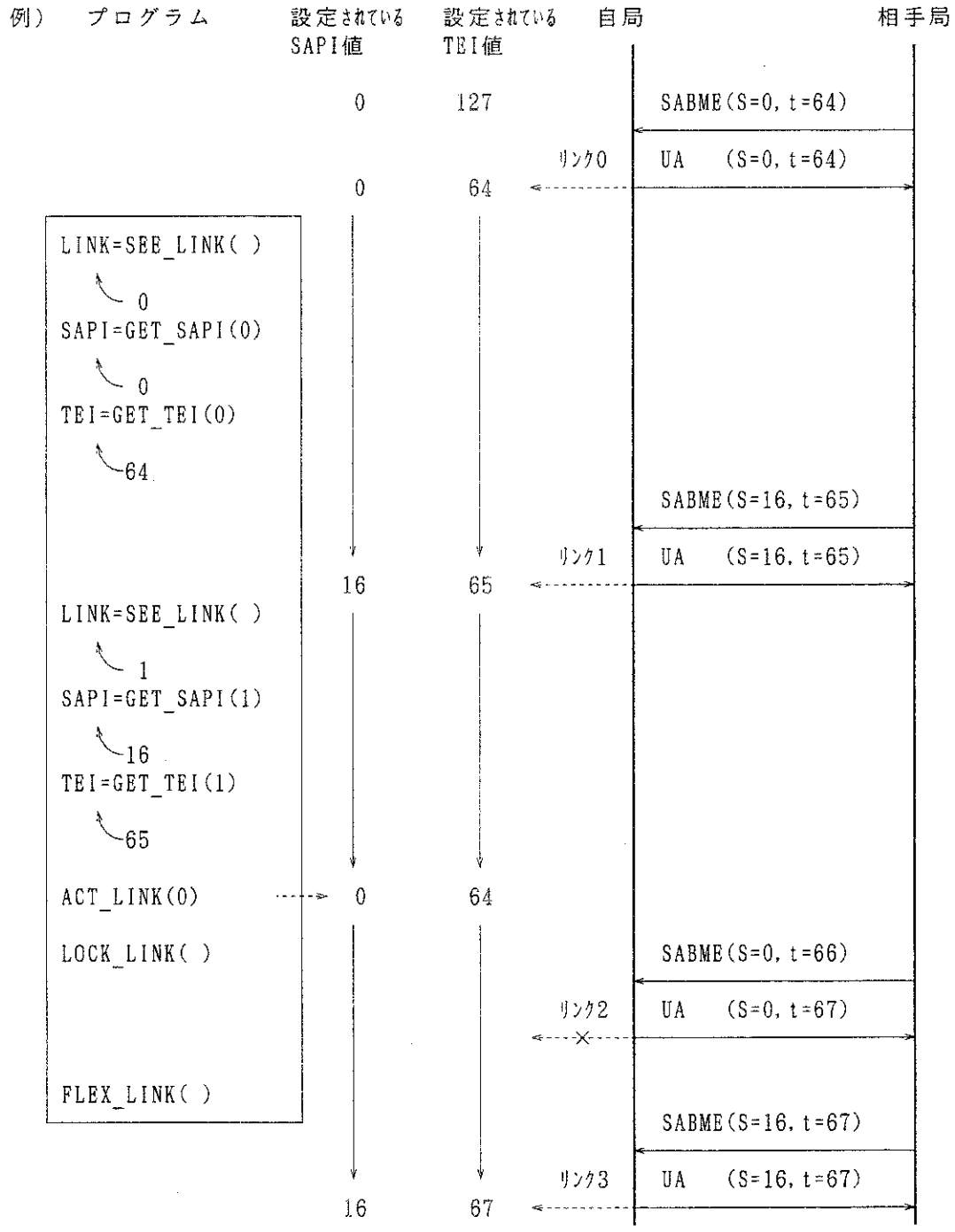
本器では最大8 リンクまで同時に接続して、シミュレーションを行なうことができます。このとき、それぞれのリンクはSAPI、TEI で管理します。この管理を行なうために以下の（前述もしている）関数を使用します。

REG_TEI
ACT_TEI
LOCK_TEI
FLEX_TEI
ACT_SAPI
LOCK_SAPI
FLEX_SAPI

本器では、リンクが張られるごとに内部的にリンク番号が付けられます。このリンク番号からSAPI値、TEI 値を知ることもできます。
また、SAPI、TEI 値両方の自動設定変更を許可したり、禁止したりすることも可能です。
リンク番号を用いた関数の概要を以下に示します。

関数	概要
SEE_LINK( )	現在設定されているリンク番号を調べる。
GET_SAPI(LINK)	引数のリンク番号のリンクのSAPI値を調べる。
GET_TEI(LINK)	引数のリンク番号のリンクのTEI 値を調べる。
GET_LINK(SAPI,TEI)	引数SAPI、TEI のリンク番号を調べる。
ACT_LINK(LINK)	引数のリンク番号のリンクに設定する。
LOCK_LINK( )	リンクの自動設定を禁止する。
FLEX_LINK( )	リンクの自動設定を許可する。

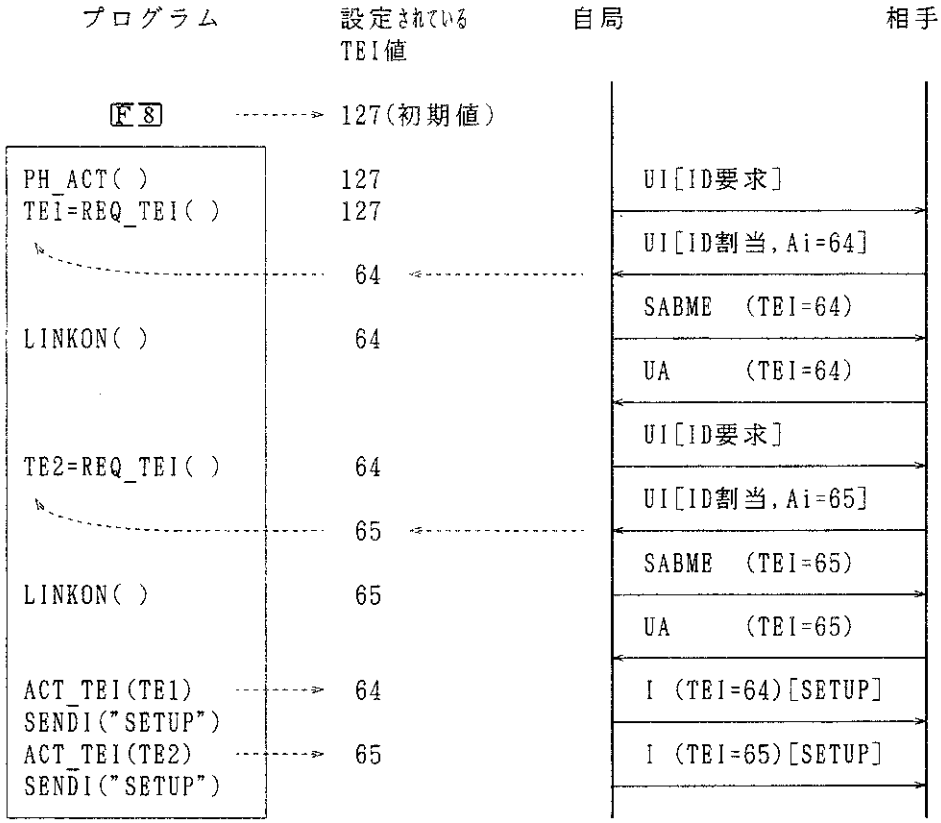
これらの関数を使用した例を以下に示します。


例) プログラム
設定されている SAPI値
設定されている TEI値
自局
相手局
0
127
SABME(S=0, t=64)
リンク0
UA (S=0, t=64)
0
64
LINK=SEE_LINK( )
0
SAPI=GET_SAPI(0)
0
TEI=GET_TEI(0)
64
SABME(S=16, t=65)
リンク1
UA (S=16, t=65)
16
65
LINK=SEE_LINK( )
1
SAPI=GET_SAPI(1)
16
TEI=GET_TEI(1)
65
ACT_LINK(0)
0
64
LOCK_LINK( )
SABME(S=0, t=66)
リンク2
UA (S=0, t=67)
FLEX_LINK( )
SABME(S=16, t=67)
リンク3
UA (S=16, t=67)
16
67

(4) レイヤ2 マルチリンクの管理例

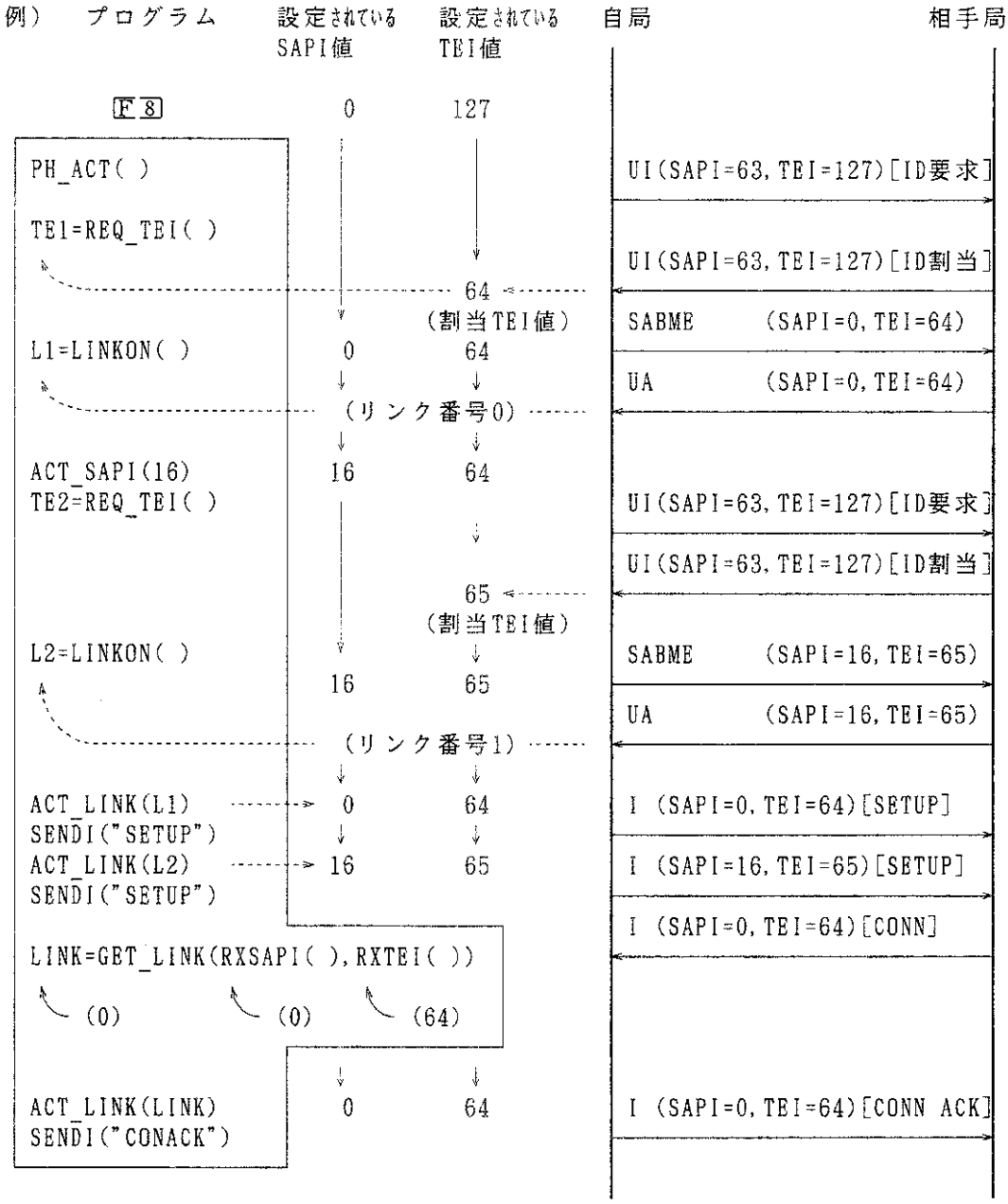
① TEI を用いたマルチリンクの設定

本器が端末2 台分をシミュレートするプログラムについて示します。
ここでは網からTEI 値を割り当ててもらい、それぞれのTEI 値でSETUP フレームを送出するケースについて示します。



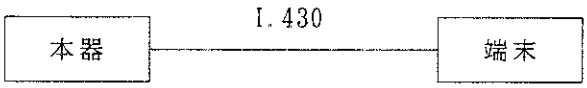
② リンク番号を用いたマルチリンクの設定

本器が端末2 台分をシミュレートするプログラムについて示します。
①の例では、TEI 値でリンクを管理していますが、リンク番号で管理することもできます。リンクの設定はACT_LINK(LINK)で行ない、受信フレームからリンク番号を得るのはGET_LINK(SAPI, TEI) で行ないます。
以下にその例と内部的な動作について説明します。



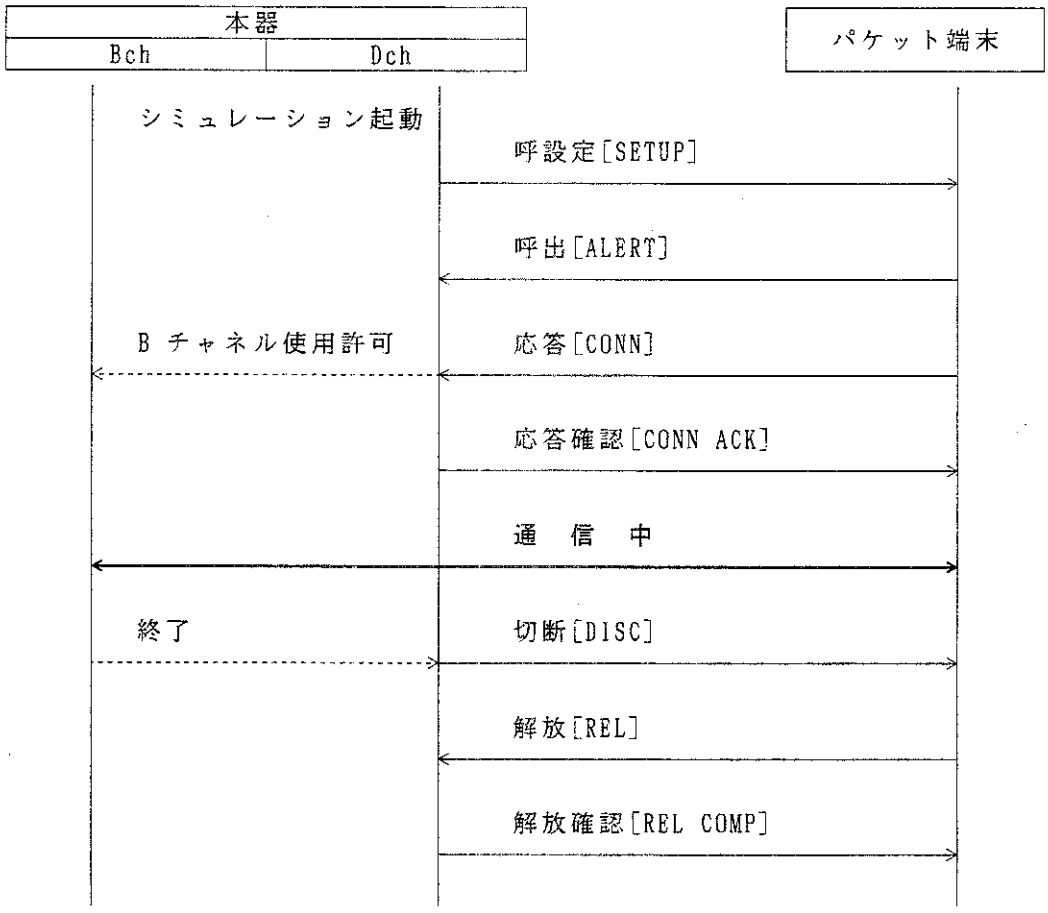
(5) 網の着呼側シミュレーション(Dチャネルの呼制御) 例

以下に示すプログラムは、次の環境で使われるのを前提としています。



プログラムを実行すると、本器は端末に対して呼設定を送ります。
端末から応答メッセージを受信すると、D チャネル・シミュレータは、B チャネル・シミュレータに対してこのメッセージの受信と使用するB チャネルの番号を通知します。これにより、B チャネルが使用できるようになり、B チャネルを用いて端末とパケットのやりとりを行ないます。
B チャネル上でのパケットのやりとりが終了すると、B チャネル・シミュレータは、D チャネル・シミュレータに対してB チャネルシミュレーション終了のメッセージを送ります。
D チャネル・シミュレータはそれを受信すると切断手順に入ります。
図示すると (例) のようになります。

(例)



● サンプル・プログラム

注意

● 行番号はプログラム説明のため、便宜上使用しています。
実際のプログラムに行番号を付与すると、コンパイルが実行できません。
● 付属のシミュレーション・サンプル・プログラム(3.5インチ・フロッピー・ディスク)に保存されています。

(1/2)

```
 1 /*      ***  NTCOM.PRG  ***
 2
 3       This is a sample in Layer 2 auto execute mode.
 4       This program simulates a network.
 5       This instrument is trying to establish the link.
 6       This program requires B channel simulator to run together.
 7       According to your purpose, please use certain message and
 8       program on B channel
 9         For example
10               packet        D message      "NTPKT_B.MSG"
11                             B program      "DCECOM.PRG"
12                             B message      "DCECOM.MSG"
13
14               Created By ADVANTEST  On OCT./20/1992
15 */
16
17
18 CHANNEL               D
19 LAYER                 3
20 SIMMODE               NT
21 PFEED                 RVS
22
23 FUNC MAIN()
24
25      CALL_PROC = H'02'
26      ALERT     = H'01'
27      CONN      = H'07'
28      DISC      = H'45'
29      REL       = H'4D'
30      REL_COMP  = H'5A'
31
32      COMM_DCH  = 10
33      COMM_BCH  = 20
34
35      PRINT("***  NTCOM.PRG  ***¥N")
36      PRINT("     TE            NT¥N")
37      PRINT(" ------------------------¥N")
38      PH_ACT()
39
40      SENDUI("SETUP")
41      PRINT("               <-  SETUP¥N")
42      CHANN_NUM = 1
43
44      WHILE(1)
45          RET = RECEIVE(0)
46          IF RET == 1 THEN
47              IF RXMSG() == CONN THEN
48                      PRINT("  CONN       ->¥N")
49                      SENDI("CONACK")
50                      PRINT("                <-  CONN ACK¥N")
```

(2/2)

```
51                      CH = RXCHAN_NUM()
52                      IF CH > 0 THEN CHANN_NUM = CH END
53                      MSG_CONN = CONN ¦ L_SHIFT(CHANN_NUM, 8)
54                      SEND_EVENT(COMM_BCH, MSG_CONN)
55                      PRINT("   ***  CONNECTED B%D CHANNEL   ***¥N",CHANN_NUM)
56              END
57
58              IF RXMSG() == CALL_PROC THEN
59                      PRINT("  CALL PROC ->¥N")
60                      CH = RXCHAN_NUM()
61                      IF CH > 0 THEN CHANN_NUM = CH END
62                      SEND_EVENT(COMM_BCH, CALL_PROC)
63              END
64
65              IF RXMSG() == ALERT THEN
66                      PRINT("  ALERT      ->¥N")
67                      CH = RXCHAN_NUM()
68                      IF CH > 0 THEN CHANN_NUM = CH END
69                      SEND_EVENT(COMM_BCH, ALERT)
70              END
71
72              IF RXMSG() == DISC THEN
73                      PRINT("  DISC      ->¥N")
74                      SENDI("REL")
75                      PRINT("                  <-  REL¥N")
76                      SEND_EVENT(COMM_BCH, DISC)
77              END
78
79              IF RXMSG() == REL THEN
80                      PRINT("  REL        ->¥N")
81                      SENDI("RELCOM")
82                      PRINT("                  <-  REL COMP¥N")
83                      SEND_EVENT(COMM_BCH, REL)
84                      EXIT
85              END
86
87              IF RXMSG() == REL_COMP THEN
88                      PRINT("  REL COMP  ->¥N")
89                      SEND_EVENT(COMM_BCH, REL_COMP)
90                      EXIT
91              END
92          END
93
94          IF RET == COMM_BCH THEN
95              MSG = READ_EVENT()
96              IF MSG == DISC THEN
97                      SENDI("DISC")
98                      PRINT("                  <-  DISC¥N")
99              END
100         END
101     END
102
103     LINKOFF()
104 RETURN
```

●プログラム解説

(1/3)

行番号	解説
1～15	コメントです。 /* ～ */ で囲まれた領域は、実行しません。
18～21	宣言文です。（このプログラムは、D チャネル用） レイヤ2 を自動実行して、網をシミュレートし、端末に対してリバース給電を供給することを宣言しています。
23～104	主関数です。 主関数 : FUNC MAIN() ～ RETURN の間に記述したプログラム
25～33	変数の初期化を行なっています。H'' は16進表示であることを示します。
35～37	プリント文です。 シミュレーション画面に" " で囲まれた文字列を表示します。
38	レイヤ1 を起動する関数です。
40	メッセージ・ビルダで作成した"SETUP" という名前のメッセージをUIフレームに乗せて送出します。
44～101	WHILE(1) ～ END で囲まれた部分で永久ループを作っています。 EXIT文を実行するまで永久にこの部分を実行し続けます。
45	端末からのフレーム、またはB チャネルからのメッセージを受信するまで待ちます。上記のイベントを受信するまでプログラム・カウンタはこの行を指したまま動きません。 上記のイベントを受信すると、RECEIVE 関数は以下に示す関数値を返します。 端末からのフレーム受信 : 関数値 1 (RET=1) Bch からのメッセージ受信 : 関数値 20(RET=20)
46～92	RECEIVE(0) で受信したイベントが端末からのフレームだった場合の処理です。
47～56	受信したフレームが、応答だった場合の処理です。 CONN ->
とシミュレーション画面に表示した後、端末に応答確認(CONACK)を Iフレームとして送出します。そしてB チャネル・シミュレータに応答フレームを受信したことと、使用するB チャネルの番号を通知します。
47	受信したフレームが応答フレームかどうか調べます。 RXMSG() は、受信したフレームのメッセージ種別のオクテットを読み出す関数です。この場合、RXMSG() の関数値が7(受信したフレームのメッセージ種別のオクテットが7)のときに48～55の行を実行します。

(2/3)

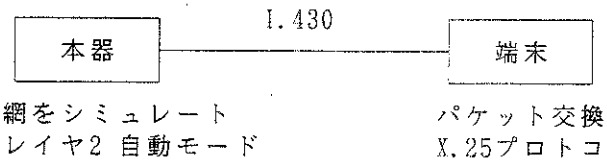
行番号	解説
49	メッセージ・ビルダで作成した"CONACK"という名前のメッセージをI フレームに乗せて送出します。
51～54	B チャネルに応答フレームを受信したことと、使用するチャネルを通知します。B チャネルの番号は、CHANN_NUM の値を使用します。 チャネル番号は、呼設定でB1チャネルを指示し、受信した呼設定受付、呼出または応答メッセージに含まれているチャネル番号を使用します。 42、51、52、60、61、67、68行がこれにあたります。 関数、演算子を使ってB チャンネルに以下のフォーマットのメッセージを送ります。 L_SHIFT : 左へ指定のビット数シフトする関数。 | : ビットORをとる演算子。 SEND_EVENT : 指定のチャネルにメッセージを送る関数。 15 8 7 0 チャネル番号 | メッセージ種別
58～63	受信したフレームが呼設定受付だった場合の処理です。 B チャネル・シミュレータに呼設定受付を受信したことを通知します。
65～70	受信したフレームが呼出だった場合の処理です。 B チャネル・シミュレータに呼出を受信したことを通知します。
72～77	受信したフレームが切断だった場合の処理です。 解放(REL) メッセージを Iフレームに乗せ、端末に送出します。 B チャネル・シミュレータに切断を受信したことを通知します。
79～85	受信したフレームが解放だった場合の処理です。 解放完了(RELCOM)メッセージをＩフレームに乗せて端末に送出します。 B チャネル・シミュレータに解放を受信したことを通知します。 EXIT : WHILE ～ END で作るループを抜ける命令。 (この場合、44行～101 行のループを抜け、103 行を実行します。)
87～91	受信したフレームが解放完了だった場合の処理です。 B チャネル・シミュレータに解放完了を受信したことを通知します。
94～100	B チャネルからのメッセージを受信したときの処理です。 B チャネルからのメッセージ内容が切断要求の場合は、切断手順に入ります。

(3/3)

行番号	解説
95	B チャネルからのメッセージを読み出す関数です。 変数MSG にB チャネルからのメッセージ内容を代入しています。 B チャネルからのメッセージは、 RECEIVE(0) の関数値が20のときのみ READ_EVENT関数で読むことができます。
96～99	B チャネルからのメッセージがDISC (16進の45) の場合は、端末に切断 (DISC)メッセージを Iフレームに乗せて送出します。
103	レイヤ2 を解放する関数です。

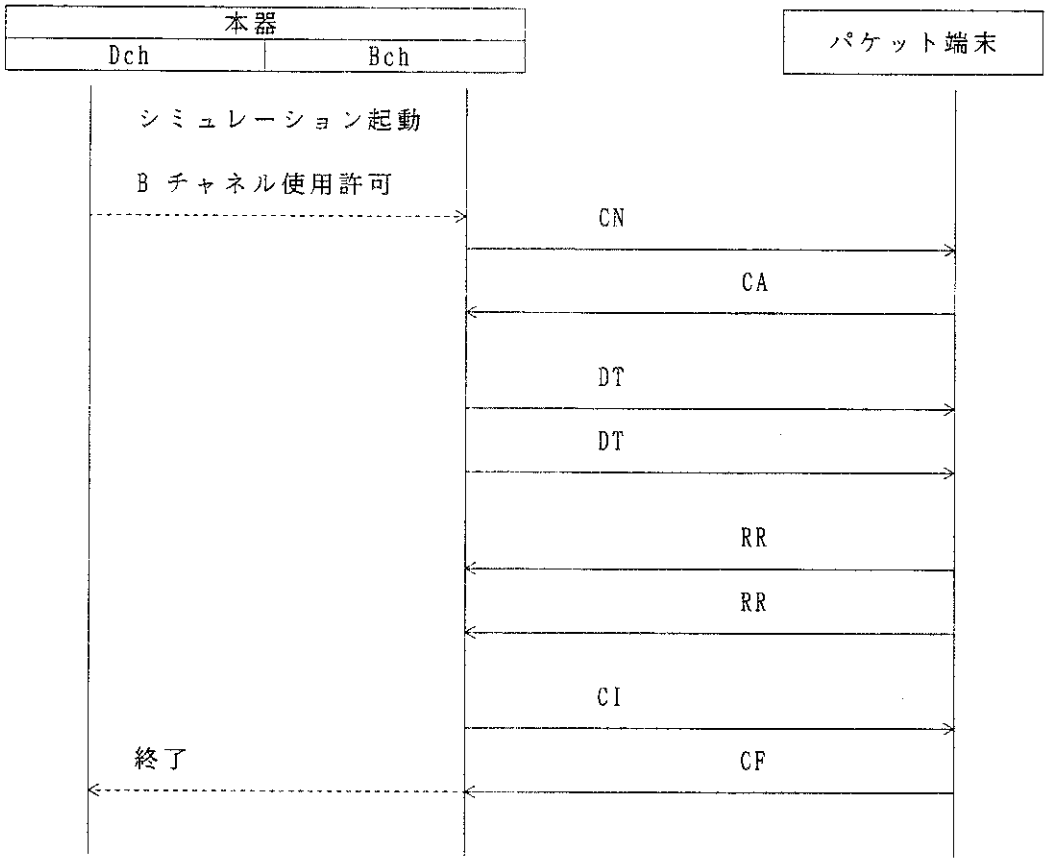
(6) 網の着呼側シミュレーション(Bチャネルのパケットデータ送出) の例

以下に示すプログラムは、次の環境で使われることを前提としています。



プログラムを実行すると、本器はD チャネルからの応答メッセージ待ち状態になります。
D チャネル・シミュレータが端末との間で呼の接続が終了すると、B チャネル・シミュレータに応答メッセージを送ります。これを受信すると、B チャネル・シミュレータは、端末にCNパケットを送ります。端末からCAパケットが返ってくると、端末にDTパケットを送出します。これから、5 秒後に端末にCIパケットを送出して切断手順に入ります。
端末からCFパケットを受信すると、D チャネルに切断メッセージを送り、シミュレーションを終了します。
図示すると (例) のようになります。

(例)



● サンプル・プログラム

注意

● 行番号はプログラム説明のため、便宜上使用しています。
実際のプログラムに行番号を付与すると、コンパイルが実行できません。
● 付属のシミュレーション・サンプル・プログラム(3.5インチ・フロッピー・ディスク)に保存されています。

(1/2)

```
 1 /*       ***  DCECOM.PRG  ***
 2
 3       This is a sample in Layer 2 auto execute mode.
 4       This instrument is trying to establish the link.
 5       Please use the message "DCECOM.MSG".
 6       This program requires D channel simulator to run together.
 7       As for D channel, use the program and the message as follows.
 8               Dch program      "NTCOM.PRG"
 9               Dch message      "NTPKT_B.MSG"
10               Created By ADVANTEST  On NOV./13/1992
11 */
12
13
14 CHANNEL              B
15 LAYER                3
16 SIMMODE              NT
17 ADDRESS              DCE
18 MODULO               8
19
20
21 FUNC MAIN()
22
23       CALL_PROC = H'02'
24       ALERT     = H'01'
25       CONN      = H'07'
26       DISC      = H'45'
27       REL       = H'4D'
28       REL_COMP  = H'5A'
29
30       CA        = H'0F'
31       DT        = H'00'
32       CQ        = H'13'
33       CF        = H'17'
34       RR        = H'01'
35
36       LCGN      = 0
37       LCN       = 3
38       TM_ID     = 1
39       TM_SEC    = 5
40
41       COMM_DCH  = 10
42       COMM_BCH  = 20
43
44       PRINT("  TE        NT¥N")
45       PRINT("---------------¥N")
46
47       WHILE(1)
48                WHILE(1)
49                    RET = RECEIVE(TM_ID)
50                    IF RET == 0 THEN
51                        SENDPKT("CI", LCGN, LCN)
52                        PRINT("          < CI¥N")
53                    END
54
```

(2/2)

```
 55                  IF RET == 1 THEN EXIT END
 56
 57                  IF RET == COMM_DCH THEN
 58                      MSG_DCH = READ_EVENT()
 59                      D_MSG = MSG_DCH & H'FF'
 60                      D_CONT = R_SHIFT(MSG_DCH, 8)
 61
 62                      IF D_MSG == CONN THEN
 63                              SET_CHANN(D_CONT)
 64                              SETPS(LCGN, LCN, 0)
 65                              SETPR(LCGN, LCN, 0)
 66                              LINKON()
 67                              SENDPKT("CN", LCGN, LCN)
 68                              PRINT("          < CN¥N")
 69                      END
 70                  END
 71              END
 72
 73              IF RXTYP() == CA THEN
 74                      PRINT("  CA    >¥N")
 75                      SENDPKT("DT1", LCGN, LCN)
 76                      PRINT("          < DT¥N")
 77                      INCPS(LCGN, LCN)
 78                      SENDPKT("DT2", LCGN, LCN)
 79                      PRINT("          < DT¥N")
 80                      INCPS(LCGN, LCN)
 81                      T_START(TM_ID, TM_SEC)
 82              END
 83
 84              IF RXTYP() == DT THEN
 85                      PRINT("  DT    >¥N")
 86                      INCPR(LCGN, LCN)
 87                      SENDPKT("RR", LCGN, LCN)
 88                      PRINT("          < RR¥N")
 89              END
 90
 91              IF RXTYP() == RR THEN
 92                      PRINT("  RR    >¥N")
 93              END
 94
 95              IF RXTYP() == CQ THEN
 96                      PRINT("  CQ    >¥N")
 97                      SENDPKT("CF", LCGN, LCN)
 98                      PRINT("          < CF¥N")
 99                      EXIT
100              END
101
102              IF RXTYP() == CF THEN
103                      PRINT("  CF    >¥N")
104                      EXIT
105              END
106      END
107
108      WAIT(30)
109      LINKOFF()
110      SEND_EVENT(COMM_DCH, DISC)
111  RETURN
```

●プログラム解説

(1/4)

行番号	解説
1～11	コメントです。 /* ～ */ で囲まれた領域は、実行しません。
14～18	宣言文です。（このプログラムは、B チャンネル用） レイヤ2 を自動実行して、網をシミュレートし、自アドレスはDCE で、モジュロ8 を使用することを宣言しています。
21～111	主関数です。 主関数 ： FUNC MAIN() ～ RETURN の間に記述したプログラム。
23～42	変数の初期化を行なっています。H'' は、16進数表示であることを示します。
44～45	プリント文です。 シミュレーション画面に" " で囲まれた文字列を表示します。
47～106 48～71	WHILE(1) ～ END で囲まれた部分で永久ループを作っています。 EXIT文を実行するまで永久にこの部分を実行し続けます。
49	端末からのフレーム、D チャンネルからのメッセージまたはタイムアウト・イベントを受信するまで待ちます。上記のイベントを受信するまでプログラム・カウンタはこの行を指したまま動きません。 上記のイベントを受信すると、RECEIVE 関数は以下に示す関数値を返します。 タイムアウト・イベント受信 ： 関数値 0 (RET=0) 端末からのフレーム受信 ： 関数値 1 (RET=1) Dch からのメッセージ受信 ： 関数値 10(RET=10)
50～53	RECEIVE(0) で受信したイベントがタイムアウト・イベントだった場合の処理です。 81行で起動したタイマがタイムアウトすると、この処理に入ります。
51	メッセージ・ビルダで作成した"CI"という名前のメッセージをI フレームに乗せて送出します。このとき論理チャンネル・グループ番号(LCGN)と論理チャンネル番号(LCN) は、引数で与えられた値を"CI"メッセージの該当のオクテットに上書きした後、送出します。
55	RECEIVE(0) で受信したイベントが端末からのフレーム受信だった場合の処理です。 EXIT命令 ： WHILE ～ END のループを強制的に抜ける関数。 （この場合は48行～71行のループを抜け、73行を実行します。）

(2/4)

行番号	解説
57～70	RECEIVE(0) で受信したイベントがD チャンネルからのメッセージだった場合の処理です。 指定のB チャンネルを選択した後、レイヤ2 を起動し、CNパケットを端末に送ります。
58	D チャンネルからのメッセージを読み、その値をMSG_DCH変数に代入しています。 READ_EVENT : 他のチャンネルからのメッセージを読み出す関数。
59～60	D チャンネルから送られてくるメッセージ種別とチャネル番号を読みます。 D チャンネルから送られてくるメッセージは以下のフォーマットをしています。 15 8 7 0 チャネル番号 | メッセージ種別 D_MSG(変数) : D チャンネル から送られてきたメッセージ種別を代入する。 D_CONT (変数) : D チャンネル から送られてきたチャネル番号を代入する。 & : ビットAND をする演算子。 R_SHIFT : 右へ指定のビット数シフトする関数。
62～69	D チャンネルからのメッセージ種別が応答だった場合の処理です。 使用するB チャンネルを設定した後、送信順序番号P(S)と受信順序番号P(R)を初期化し、端末にCNパケットを送出します。
63	使用するB チャンネルをD チャンネルから送られてきたチャネル番号に設定します。 SET__CHANN 関数 : 使用するB チャンネルを指定する関数。
64～65	引数で指定した論理チャンネル・グループ番号(LCGN)と論理チャンネル番号(LCN) の送信順序番号P(S)と受信順序番号P(R)を初期化しています。 SETPS : 送信順序番号P(S)の値を設定する関数。 SETPR : 受信順序番号P(R)の値を設定する関数。
66	レイヤ2 リンクを設定します。
67	メッセージ・ビルダで作成した"CN"という名前のメッセージをI フレームに乗せて送出します。このとき論理チャンネル・グループ番号(LCGN)と論理チチャンネル番号(LCN) は、引数で与えられた値を"CN"メッセージの該当のオクテットに上書きした後、送出します。

(3/4)

行番号	解説
73～105	端末からパケットを受信したときの処理です。 端末からのパケット受信の情報は、49行のRECEIVE(0)で受信します。 そして、55行の条件文で端末からパケットを受信したと判断すると、EXIT命令でWHILE ループを抜け、これらの行が実行されます。
73～82	受信したパケットがCAパケットだった場合の処理です。 DTパケットを2 つ送出し、タイマを起動します。このタイマは、ある一定時間経過した後、CIパケットにより切断するために使われます。
75	メッセージ・ビルダで作成した"DT1" という名前のメッセージをI フレームに乗せて送出します。このとき論理チャンネル・グループ番号(LCGN)と論理チャンネル番号(LCN) は、引数で与えられた値を"DT1" メッセージの該当のオクテットに上書きした後、送出します。また、送信順序番号P(S)と受信順序番号P(R)も同様に、送出する前に上書きします。 送信順序番号P(S)の値を+1増加させます。
77	DTパケットを送出したので、送信順序番号P(S)の値を+1増加させています。
81	タイマを起動しています。 この場合、タイマ番号TM_ID(1) でタイムアウト値TM_SEC (5秒) のタイマを起動しています。 このタイムアウト・イベントは49行のRECEIVE(TM_ID)で受信します。
84～89	受信したパケットがDTパケットだった場合の処理です。 受信順序番号P(R)の値を+1増加させ、RRパケットを送出します。
86	受信順序番号P(R)の値を+1増加させます。 DTパケットを受信したので、受信順序番号P(R)の値を+1します。
87	メッセージ・ビルダで作成した"RR"という名前のメッセージをI フレームに乗せて送出します。このとき論理チャンネル・グループ番号(LCGN)と論理チャンネル番号(LCN) は、引数で与えられた値を"RR"メッセージの該当のオクテットに上書きした後、送出します。また、受信順序番号P(R)も同様に、送出する前に上書きします。
91～93	受信したパケットがRRパケットだった場合の処理です。
95～100	受信したパケットがCQパケットだった場合の処理です。 CFパケットを送出し、47行～106 行のWHILE ループを抜けます。

(4/4)

行番号	解説
102 ～105	受信したパケットがCFパケットだった場合の処理です。 47行～106 行のWHILE ループを抜けます。
108	プログラムを指定の時間、中断する関数です。 この場合、3 秒プログラムを中断させた後、109 行の処理に入ります。
109	レイヤ2 リンクを解放します。
110	D チャンネルに、切断メッセージを送出します。

(7) 一次群インタフェースにおけるシミュレーション例

一次群インタフェースにおけるプログラム例を以下に示します。

D チャネル・プログラム例	TECOM.PRG
D チャネル・メッセージ例	PACK_B.MSG
B チャネル・プログラム例	DTECOM.PRG
B チャネル・メッセージ例	DTECOM.MSG

上記のプログラムおよびメッセージを同時使用することにより、一次群インタフェースにて、B チャネル・パケット通信（端末側シミュレーション）および、そのときのD チャネルにおける呼制御が実行できます。ただし、上記例題プログラムをD チャネル、B チャネル同時に実行する必要があります。

●サンプル・プログラム

注意

●行番号はプログラム説明のため、便宜上使用しています。
実際のプログラムに行番号を付与すると、コンパイルが実行できません。
●付属のシミュレーション・サンプル・プログラム(3.5インチ・フロッピー・ディスク)に保存されています。

(1/2)

```
1   /*      ***  TECOM.PRG  ***
2
3           This is a sample in Layer 2 auto execute mode.
4           This program simulates a terminal.
5           This instrument is trying to establish the link.
6           This program requires B channel simulator to run together.
7           According to your purpose, please use certain message and
8           program on B channel
9
10                  packet        D message     "PACK_B.MSG"
11                                B program     "DTECOM.PRG"
12                                B message     "DTECOM.MSG"
13
14                  Created By ADVANTEST  On Nov./13/1994
15  */
16
17
18  CHANNEL         D
19  INTERFACE       PRI
20  LAYER           3
21  SIMMODE         TE
22
23  FUNC    MAIN()
24
25          CALL_PROC = H'02'
26          ALERT     = H'01'
27          CONN      = H'07'
28          DISC      = H'45'
29          REL       = H'4D'
30          REL_COMP  = H'5A'
31
32          COMM_DCH  = 10
33          COMM_BCH  = 20
34
35          PRINT("***  TECOM.PRG  ***\N")
36          PRINT("     TE            NT\N")
37          PRINT(" -------------------------\N")
38
39          LINKON()
40          SENDI("SETUP")
41          PRINT("  SETUP     ->\N")
42          CHANN_NUM = 1
43
44          WHILE(1)
45              RET = RECEIVE(0)
46              IF RET == 1 THEN
```

(2/2)

```
47                  IF RXMSG() == CONN THEN
48                          PRINT("              <-  CONN\N")
49                          SENDI("CONACK")
50                          PRINT("  CONN ACK  ->\N")
51                          CH = RXCHAN_NUM()
52                          IF CH > 0 THEN CHANN_NUM = CH END
53                          MSG_CONN = CONN | L_SHIFT(CHANN_NUM, 8)
54                          SEND_EVENT(COMM_BCH, MSG_CONN)
55                          PRINT("  ***  CONNECTED B%D CHANNEL  ***\N",CHANN_NUM)
56                  END
57
58                  IF RXMSG() == CALL_PROC THEN
59                          PRINT("              <-  CALL PROC\N")
60                          CH = RXCHAN_NUM()
61                          IF CH > 0 THEN CHANN_NUM = CH END
62                          SEND_EVENT(COMM_BCH, CALL_PROC)
63                  END
64
65                  IF RXMSG() == ALERT THEN
66                          PRINT("              <-  ALERT\N")
67                          IF CH > 0 THEN CHANN_NUM = CH END
68                          SEND_EVENT(COMM_BCH, ALERT)
69                  END
70
71                  IF RXMSG() == DISC THEN
72                          PRINT("              <-  DISC\N")
73                          SENDI("REL")
74                          PRINT("  REL       ->\N")
75                          SEND_EVENT(COMM_BCH, DISC)
76                  END
77
78                  IF RXMSG() == REL THEN
79                          PRINT("              <-  REL\N")
80                          SENDI("RELCOM")
81                          PRINT("  REL COMP  ->\N")
82                          SEND_EVENT(COMM_BCH, REL)
83                          EXIT
84                  END
85
86                  IF RXMSG() == REL_COMP THEN
87                          PRINT("              <-  REL COMP\N")
88                          SEND_EVENT(COMM_BCH, REL_COMP)
89                          EXIT
90                  END
91              END
92
93              IF RET == COMM_BCH THEN
94                  MSG = READ_EVENT()
95                  IF MSG == DISC THEN
96                          SENDI("DISC")
97                          PRINT("  DISC      ->\N")
98                  END
99              END
100         END
101
102         LINKOFF()
103 RETURN
```

●プログラム解説

(1/3)

行番号	解説
1～15	コメントです。 /* ～ */ で囲まれた領域は、実行しません。
18～21	宣言文です。（このプログラムは、一次群のD チャネル用） レイヤ2 を自動実行して、端末をシミュレートすることを宣言しています。
23～103	主関数です。 主関数 ： FUNC MAIN() ～ RETURN の間に記述したプログラム
25～33	変数の初期化を行なっています。H'' は16進表示であることを示します。
35～37	プリント文です。 シミュレーション画面に" " で囲まれた文字列を表示します。
39	リンクの設定をする関数です。
40	メッセージ・ビルダで作成した"SETUP" という名前のメッセージを Iフレームに乗せて送出します。
44～100	WHILE(1) ～ END で囲まれた部分で永久ループを作っています。 EXIT文を実行するまで永久にこの部分を実行し続けます。
45	網からのフレーム、またはB チャネルからのメッセージを受信するまで待ちます。上記のイベントを受信するまでプログラム・カウンタはこの行を指したまま動きません。 上記のイベントを受信すると、RECEIVE 関数は以下に示す関数値を返します。 端末からのフレーム受信 ： 関数値 1 (RET=1) Bch からのメッセージ受信 ： 関数値 20(RET=20)
46～91	RECEIVE(0) で受信したイベントが網からのフレームだった場合の処理です。
47～56	受信したフレームが、応答だった場合の処理です。 CONN ->
とシミュレーション画面に表示した後、網に応答確認(CONACK)を Iフレームとして送出します。そしてB チャネル・シミュレータに応答フレームを受信したことと、使用するB チャネルの番号を通知します。
47	受信したフレームが応答フレームかどうか調べます。 RXMSG() は、受信したフレームのメッセージ種別のオクテットを読み出す関数です。この場合、RXMSG() の関数値が7(受信したフレームのメッセージ種別のオクテットが7)のときに48～55の行を実行します。

(2/3)

行番号	解説
49	メッセージ・ビルダで作成した"CONACK"という名前のメッセージをI フレームに乗せて送出します。
51～54	B チャネルに応答フレームを受信したことと、使用するチャネルを通知します。B チャネルの番号は、CHANN_NUM の値を使用します。 チャネル番号は、呼設定でB1チャネルを指示し、受信した呼設定受付、呼出または応答メッセージに含まれているチャネル番号を使用します。 42、51、52、60、61、67、68行がこれにあたります。 関数、演算子を使ってB チャンネルに以下のフォーマットのメッセージを送ります。 L_SHIFT : 左へ指定のビット数シフトする関数。 | : ビットORをとる演算子。 SEND_EVENT : 指定のチャネルにメッセージを送る関数。 15 8 7 0 チャネル番号 | メッセージ種別
58～63	受信したフレームが呼設定受付だった場合の処理です。 B チャネル・シミュレータに呼設定受付を受信したことを通知します。
65～69	受信したフレームが呼出だった場合の処理です。 B チャネル・シミュレータに呼出を受信したことを通知します。
71～76	受信したフレームが切断だった場合の処理です。 解放(REL) メッセージを Iフレームに乗せ、端末に送出します。 B チャネル・シミュレータに切断を受信したことを通知します。
78～84	受信したフレームが解放だった場合の処理です。 解放完了(RELCOM)メッセージを I フレームに乗せて端末に送出します。 B チャネル・シミュレータに解放を受信したことを通知します。 EXIT : WHILE ～ END で作るループを抜ける命令。 (この場合、44行～100 行のループを抜け、103 行を実行します。)
86～90	受信したフレームが解放完了だった場合の処理です。 B チャネル・シミュレータに解放完了を受信したことを通知します。
93～99	B チャネルからのメッセージを受信したときの処理です。 B チャネルからのメッセージ内容が切断要求の場合は、切断手順に入ります。

(3/3)

行番号	解説
94	B チャネルからのメッセージを読み出す関数です。 変数MSG にB チャネルからのメッセージ内容を代入しています。 B チャネルからのメッセージは、 RECEIVE(0) の関数値が20のときのみ READ_EVENT関数で読むことができます。
95～98	B チャネルからのメッセージがDISC (16進の45) の場合は、端末に切断 (DISC)メッセージを Iフレームに乗せて送出します。
102	レイヤ2 を解放する関数です。

● サンプル・プログラム

注意

● 行番号はプログラム説明のため、便宜上使用しています。
実際のプログラムに行番号を付与すると、コンパイルが実行できません。
● 付属のシミュレーション・サンプル・プログラム(3.5インチ・フロッピー・ディスク)に保存されています。

(1/2)

```
1   /*      ***  DTECOM.PRG  ***
2
3           This is a sample in Layer 2 auto execute mode.
4           This program simulates a terminal.
5           This instrument is trying to establish the link.
6           Please use the message "DTECOM.MSG".
7           This program requires D channel simulator to run together.
8           As for D channel, use the program and the message as follows.
9                   Dch program     "TECOM.PRG"
10                  Dch message     "PACK_B.MSG"
11                  Created By ADVANTEST On Oct./06/1993
12  */
13
14
15  CHANNEL         B
16  INTERFACE       PRI
17  LAYER           3
18  SIMMODE         TE
19  ADDRESS         DTE
20  MODULO          8
21
22
23  FUNC    MAIN()
24
25          CALL_PROC = H'02'
26          ALERT     = H'01'
27          CONN      = H'07'
28          DISC      = H'45'
29          REL       = H'4D'
30          REL_COMP  = H'5A'
31
32          CC        = H'0F'
33          DT        = H'00'
34          CI        = H'13'
35          CF        = H'17'
36          RR        = H'01'
37
38          LCGN      = 0
39          LCN       = 3
40          TM_ID     = 1
41          TM_SEC    = 5
42
43          COMM_DCH  = 10
44          COMM_BCH  = 20
45
46          PRINT("  TE       NT\N")
47          PRINT("--------------\N")
48
49          WHILE(1)
50                  WHILE(1)
```

(2/2)

```
51                      RET = RECEIVE(TM_ID)
52                      IF RET == 0 THEN
53                          SENDPKT("CQ", LCGN, LCN)
54                          PRINT("  CQ   >\N")
55                      END
56
57                      IF RET == 1 THEN EXIT END
58
59                      IF RET == COMM_DCH THEN
60                          MSG_DCH = READ_EVENT()
61                          D_MSG = MSG_DCH & H'FF'
62                          D_CONT = R_SHIFT(MSG_DCH, 8)
63
64                          IF D_MSG == CONN THEN
65                                  SET_CHANN(D_CONT)
66                                  SETPS(LCGN, LCN, 0)
67                                  SETPR(LCGN, LCN, 0)
68                                  LINKON()
69                                  SENDPKT("CR", LCGN, LCN)
70                                  PRINT("  CR   >\N")
71                          END
72                      END
73                  END
74
75                  IF RXTYP() == CC THEN
76                          PRINT("         < CC\N")
77                          SENDPKT("DT1", LCGN, LCN)
78                          PRINT("  DT   >\N")
79                          INCPS(LCGN, LCN)
80                          SENDPKT("DT2", LCGN, LCN)
81                          PRINT("  DT   >\N")
82                          INCPS(LCGN, LCN)
83                          T_START(TM_ID, TM_SEC)
84                  END
85
86                  IF RXTYP() == DT THEN
87                          PRINT("         < DT\N")
88                          INCPR(LCGN, LCN)
89                          SENDPKT("RR", LCGN, LCN)
90                          PRINT("  RR   >\N")
91                  END
92
93                  IF RXTYP() == RR THEN
94                          PRINT("         < RR\N")
95                  END
96
97                  IF RXTYP() == CI THEN
98                          PRINT("         < CI\N")
99                          SENDPKT("CF", LCGN, LCN)
100                         PRINT("  CF   >\N")
101                         EXIT
102                 END
103
104                 IF RXTYP() == CF THEN
105                         PRINT("         < CF\N")
106                         EXIT
107                 END
108         END
109
110         WAIT(30)
111         LINKOFF()
112         SEND_EVENT(COMM_DCH, DISC)
113 RETURN
```

●プログラム解説

(1/4)

行番号	解説
1～12	コメントです。 /* ～ */ で囲まれた領域は、実行しません。
15～20	宣言文です。（このプログラムは、一次群のB チャネル用） レイヤ2 を自動実行して、端末をシミュレートし、自アドレスはDTE で、モジュロ8 を使用することを宣言しています。
23～113	主関数です。 主関数 ： FUNC MAIN() ～ RETURN の間に記述したプログラム。
25～44	変数の初期化を行なっています。H'' は、16進数表示であることを示します。
46～47	プリント文です。 シミュレーション画面に" "で囲まれた文字列を表示します。
49～108 50～73	WHILE(1) ～ END で囲まれた部分で永久ループを作っています。 EXIT文を実行するまで永久にこの部分を実行し続けます。
51	網からのフレーム、D チャンネルからのメッセージまたはタイムアウト・イベントを受信するまで待ちます。上記のイベントを受信するまでプログラム・カウンタはこの行を指したまま動きません。 上記のイベントを受信すると、RECEIVE 関数は以下に示す関数値を返します。 タイムアウト・イベント受信 ： 関数値 0 (RET=0) 網からのフレーム受信 ： 関数値 1 (RET=1) Dch からのメッセージ受信 ： 関数値 10(RET=10)
52～55	RECEIVE(0) で受信したイベントがタイムアウト・イベントだった場合の処理です。 83行で起動したタイマがタイムアウトすると、この処理に入ります。
53	メッセージ・ビルダで作成した"CQ"という名前のメッセージをI フレームに乗せて送出します。このとき論理チャンネル・グループ番号(LCGN)と論理チャンネル番号(LCN) は、引数で与えられた値を"CQ"メッセージの該当のオクテットに上書きした後、送出します。
57	RECEIVE(0) で受信したイベントが網からのフレーム受信だった場合の処理です。 EXIT命令 ： WHILE ～ END のループを強制的に抜ける関数。 （この場合は50行～73行のループを抜け、75行を実行します。）

(2/4)

行番号	解説
59～72	RECEIVE(0) で受信したイベントがD チャンネルからのメッセージだった場合の処理です。 指定のB チャンネルを選択した後、レイヤ2 を起動し、CRパケットを端末に送ります。
60	D チャンネルからのメッセージを読み、その値をMSG_DCH変数に代入しています。 READ_EVENT ： 他のチャンネルからのメッセージを読み出す関数。
61～62	D チャンネルから送られてくるメッセージ種別とチャネル番号を読みます。 D チャンネルから送られてくるメッセージは以下のフォーマットをしています。 15 8 7 0 チャネル番号 | メッセージ種別 D_MSG(変数) ： D チャンネル から送られてきたメッセージ種別を代入する。 D_CONT (変数) ： D チャンネル から送られてきたチャネル番号を代入する。 & ： ビットAND をする演算子。 R_SHIFT ： 右へ指定のビット数シフトする関数。
64～71	D チャンネルからのメッセージ種別が応答だった場合の処理です。 使用するB チャンネルを設定した後、送信順序番号P(S)と受信順序番号P(R)を初期化し、網にCRパケットを送出します。
65	使用するB チャンネルをD チャンネルから送られてきたチャネル番号に設定します。 SET__CHANN 関数 ： 使用するB チャンネルを指定する関数。
66～67	引数で指定した論理チャンネル・グループ番号(LCGN)と論理チャンネル番号(LCN) の送信順序番号P(S)と受信順序番号P(R)を初期化しています。 SETPS ： 送信順序番号P(S)の値を設定する関数。 SETPR ： 受信順序番号P(R)の値を設定する関数。
68	レイヤ2 リンクを設定します。
69	メッセージ・ビルダで作成した"CR"という名前のメッセージをI フレームに乗せて送出します。このとき論理チャンネル・グループ番号(LCGN)と論理チチャンネル番号(LCN) は、引数で与えられた値を"CR"メッセージの該当のオクテットに上書きした後、送出します。

(3/4)

行番号	解説
75～107	網からパケットを受信したときの処理です。 網からのパケット受信の情報は、51行のRECEIVE(0)で受信します。 そして、57行の条件文で網からパケットを受信したと判断すると、EXIT命令でWHILE ループを抜け、これらの行が実行されます。
75～84	受信したパケットがCCパケットだった場合の処理です。 DTパケットを2 つ送出し、タイマを起動します。このタイマは、ある一定時間経過した後、CQパケットにより切断するために使われます。
77	メッセージ・ビルダで作成した"DT1" という名前のメッセージをI フレームに乗せて送出します。このとき論理チャンネル・グループ番号(LCGN)と論理チャンネル番号(LCN) は、引数で与えられた値を"DT1" メッセージの該当のオクテットに上書きした後、送出します。また、送信順序番号P(S)と受信順序番号P(R)も同様に、送出する前に上書きします。 送信順序番号P(S)の値を+1増加させます。
79	DTパケットを送出したので、送信順序番号P(S)の値を+1増加させています。
83	タイマを起動しています。 この場合、タイマ番号TM_ID(1) でタイムアウト値TM_SEC (5秒) のタイマを起動しています。 このタイムアウト・イベントは51行のRECEIVE(TM_ID)で受信します。
86～89	受信したパケットがDTパケットだった場合の処理です。 受信順序番号P(R)の値を+1増加させ、RRパケットを送出します。
88	受信順序番号P(R)の値を+1増加させます。 DTパケットを受信したので、受信順序番号P(R)の値を+1します。
89	メッセージ・ビルダで作成した"RR"という名前のメッセージをI フレームに乗せて送出します。このとき論理チャンネル・グループ番号(LCGN)と論理チャンネル番号(LCN) は、引数で与えられた値を"RR"メッセージの該当のオクテットに上書きした後、送出します。また、受信順序番号P(R)も同様に、送出する前に上書きします。
93～95	受信したパケットがRRパケットだった場合の処理です。
97～102	受信したパケットがCIパケットだった場合の処理です。 CFパケットを送出し、49行～108 行のWHILE ループを抜けます。

(4/4)

行番号	解説
104 ～107	受信したパケットがCFパケットだった場合の処理です。 49行～108 行のWHILE ループを抜けます。
110	プログラムを指定の時間、中断する関数です。 この場合、3 秒プログラムを中断させた後、111 行の処理に入ります。
111	レイヤ2 リンクを解放します。
112	D チャンネルに、切断メッセージを送出します。

関数索引

ここでは、各関数に対応する本書でのページを記載しています。
(D ch)、(B ch)の表記は、それぞれD チャネル・シミュレーション、B チャネル・シミュレーション用に使用する関数であることを示します。

—— アルファベット順 ——

【L】

【N】

【P】

【R】

【S】

【T】

【U】

【V】

【W】

本製品に含まれるソフトウェアのご使用について

本製品に含まれるソフトウェア（以下本ソフトウェア）のご使用について以下のことにご注意下さい。
ここでいうソフトウェアには、本製品に含まれる又は共に使用されるコンピュータ・プログラム、将来弊社よりお客様に提供されることのある追加、変更、修正プログラムおよびアップデート版のコンピュータ・プログラム、ならびに本製品に関する取扱説明書等の付随資料を含みます。

使用許諾

本ソフトウェアの著作権を含む一切の権利は弊社に帰属いたします。
弊社は、本ソフトウェアを本製品上または本製品とともに使用する限りにおいて、お客様に使用を許諾するものといたします。

禁止事項

お客様は、本ソフトウェアのご使用に際し以下の事項は行わないで下さい。

- 本製品使用目的以外で使用すること
- 許可なく複製、修正、改変を行うこと
- リバース・エンジニアリング、逆コンパイル、逆アセンブルなどを行うこと

免　　責

お客様が、本製品を通常の用法以外の用法で使用したことにより本製品に不具合が発生した場合、およびお客様と第三者との間で著作権等に関する紛争が発生した場合、弊社は一切の責任を負いかねますのでご了承下さい。

保証について

製品の保証期間は、お客様と別段の取り決めがある場合または当社が特に指定した場合を除き、製品の納入日（システム機器については検収日）から1年間といたします。保証期間中に、当社の責めに帰する製造上の欠陥により製品が故障した場合、無償で修理いたします。ただし、下記に該当する場合は、保証期間中であっても保証の対象から除外させていただきます。

- 当社が認めていない改造または修理を行った場合
- 支給品等当社指定品以外の部品を使用した場合
- 取扱説明書に記載する使用条件を超えて製品を使用した場合（定められた許容範囲を超える物理的ストレスまたは電流電圧がかかった場合など）
- 通常想定される使用環境以外で製品を使用した場合（腐食性の強いガス、塵埃の多い環境等による電気回路の腐食、部品の劣化が早められた場合など）
- 取扱説明書または各種製品マニュアルの指示事項に従わずに使用された場合
- 不注意または不当な取扱により不具合が生じた場合
- お客様のご指示に起因する場合
- 消耗品や消耗材料に基づく場合
- 火災、天変地異等の不可抗力による場合
- 日本国外に持出された場合
- 製品を使用できなかったことによる損失および逸失利益

当社の製品の保証は、本取扱説明書に記載する内容に限られるものとします。

保守に関するお問い合わせについて

長期間にわたる信頼性の保証、国家標準とのトレーサビリティを実現するためにアドバンテストでは、工場から出荷された製品の保守に対し、カスタマ・エンジニアを配置しています。
カスタマ・エンジニアは、故障などの不慮の事故は元より、製品の長期間にわたる性能の保証活動にフィールド・エンジニアとしても活動しています。
万一、動作不良などの故障が発生した場合には、当社のMS（計測器）コールセンタにご連絡下さい。

製品修理サービス

- **製品修理期間**
  製品の修理サービス期間は、製品の納入後10年間とさせていただきます。
- **製品修理活動**
  当社の製品に故障が発生した場合、当社に送っていただく引取り修理、または当社技術員が現地に出張しての出張修理にて対応いたします。

製品校正サービス

- **校正サービス**
  ご使用中の製品に対し、品質および信頼性の維持を図ることを目的に行うもので、校正後の製品には校正ラベルを貼付けし、品質を保証いたします。
- **校正サービス活動**
  校正サービス活動は、株式会社アドバンテスト カスタマサポートに送っていただく引取り校正、または当社技術員が現地に出張しての出張校正にて対応いたします。

予防保守のおすすめ

製品にはエレクトロニクス部品およびメカニカル部品の一部に寿命を考慮すべき部品を使用しているため、定期的な交換を必要とします。適正な交換期間を過ぎて使用し発生した障害に対しては、修理および性能の保証ができない場合があります。
アドバンテストでは、このようなトラブルを未然に防ぐため、予防保守が有効な手段と考え、予防保守作業を実施する体制を整えています。
各種の予防保守を定期的に実施することで、製品の安定稼働を図り、不意の費用発生を防ぐため、年間保守契約による予防保守の実施をお薦めいたします。
なお、年間保守契約は、製品、使用状況および使用環境により内容が変わりますので、最寄りの弊社営業支店にお問い合わせ下さい。

ADVANTEST

http://www.advantest.co.jp

株式会社アドバンテスト

本社事務所
〒100-0005 千代田区丸の内1-6-2 新丸の内センタービルディング
TEL: 03-3214-7500（代）

第4アカウント販売部（東日本）
〒100-0005 千代田区丸の内1-6-2 新丸の内センタービルディング
TEL: 0120-988-971
FAX: 0120-988-973

第4アカウント販売部（西日本）
〒564-0062 吹田市垂水町3-34-1
TEL: 0120-638-557
FAX: 0120-638-568

★計測器に関するお問い合わせ先
（製品の仕様，取扱い，修理・校正等計測器関連全般）
MS（計測器）コールセンタ
TEL 0120-919-570
FAX 0120-057-508
E-mail : icc@acs.advantest.co.jp