BELL SYSTEM PRACTICES
AT&TCo Standard

SECTION 231-118-320
Issue 3, February 1981

RECENT CHANGE MESSAGE PROGRAM LISTINGS, SYSTEM ACKNOWLEDGEMENT, AND RC18 AND RC16 OUTPUT MESSAGES (THROUGH 1E5 GENERIC PROGRAM)

2-WIRE NO. 1 ELECTRONIC SWITCHING SYSTEM

CONTENTS PAGE

NOTICE

Not for use or disclosure outside the
Bell System except under written agreement

Printed in U.S.A.

CONTENTS PAGE

Figures

CONTENTS PAGE

Tables

1. GENERAL

1.01 This section describes the contents of recent change (RC) message program listings, system acknowledgment, and RC18 and RC16 output messages. This information applies to the 2-Wire No. 1 Electronic Switching System (ESS) (through 1E5 generic programs). General RC information is given in Section 231-118-321.

1.02 This section is reissued for the following reason:

(a) To delete generics prior to 1E3.

(b) To add more detail to pident contents and layout description (Part 2).

(c) To make minor corrections as required.

Program Listing

1.03 A program listing (PR) consists of a computer generated hardcopy list of program instructions and related information for one or more program units identified by program identifications (pidents). A pident is that segment of a program that is compiled as a unit by a compiler program (on a general purpose computer). Most PRs consist of a single pident; only a few contain more than one. This listing produced as a result of this compilation becomes the PR or a part of it if pidents are combined to perform a system function. No pident is split between PRs. The PR contains an index sheet which lists the pidents it contains and their issues, followed by the listings for each pident. PRs are numbered as PR-1A... for No. 1 ESS (... is some unique 3-digit number).

Program Identification

1.04 The recent change message interpretation programs form a system to interpret RC input messages, check them for validity, and assemble the input data into the proper format for storage in translation memory. The RC message system consists of over 70 message pidents (1 per message), 10 code (or control) pidents, and 2 pidents shared by all the message pidents. The message pidents are tables of data defining the messages and the control pidents are the programs of executable code (ie, ESS instructions) that perform the functions of the RC message system. Many of these functions are accomplished by applying fixed algorithms to the data in the message tables; therefore, a substantial portion of the system is table driven by the message pidents.

1.05 Table A lists the 12 nonmessage pidents, a representative message pident, and references to associated PR documents. A brief description of the function of the control, shared, and message types of pidents shown in Table A follows.

TABLE A

NONMESSAGE AND REPESENTATIVE MESSAGE PIDENTs

PIDENT	TYPE	TITLE (EACH PREFIXED BY RECENT CHANGE)	PR-NO. 1 ESS
RCIG	Control	Initialization and General Control	PR-1A300
RCIE		Input Editor	PR-1A301
RCKI		Keyword Input	PR-1A302
RCVC		Validity Check	PR-1A303
RCTF		New Pass Translation Format	PR-1A304
RCCH		Change Pass Translation Format	PR-1A305
RCFI		Format Interpretation	PR-1A306
RCWL		Work List Processing	PR-1A307
RCDY		Delayed Order	PR-1A308
RSUB	↓	General Purpose Subroutines	PR-1A309
RCTS	Shared	Table Subroutines	PR-1A319
RCSI	↓	Shared Information	PR-1A320
RCxx*	Message	(A Pident Per Message)	PR-1A3yy+

* Two Letters Identifying a Message

+ 21, 22,. . . nn

Control Pidents

- ***RCIG*** (Initialization and General Control)—Determine availability of the RC system for the first line of the message and directs the flow of control among other modules in the RC system.

- ***RCIE*** (Input Editor)—Links the input/output terminal (IOT) buffer and the RC message processing system.

- ***RCKI*** (Keyword Input)—Performs processing and local error checking of RC message keyword units.

- ***RCVC*** (Validity Checks)—Assures valid translation data base transitions.

- ***RCTF*** (New Pass Translation Format)—***RCCH*** (Change Pass Translation Format) and ***RCFI*** (Format Interpretation)—The format section builds images of translation data in the RC work list and RC auxiliary area on NEW and CHG messages and retrieves translation data from program store and RC area on CHG and OUT messages.

- ***RCWL*** (Work List Processing)—Processes the work list (WL) entries made by the message pidents.

- ***RCDY*** (Delayed Order)—Processes delayed service order by storing message storage buffer (MSB) input in a delayed activation block (DAB) for future reloading in the MSB.

Shared Pidents

- ***RCTS*** (Table Subroutines)—Consists of special purpose subroutines used by work list pident RCWL and the RC message pidents.

- ***RCSI*** (Shared Information)—Consists of five data tables which are shared by several of the code (control) pidents during RC message processing.

Message Pidents

- ***RCxx*** —In general there is a message pident for each type of translator. The message pident is made up of data tables which provide the decision information to control the processing flow of the associated code (control) pidents.

System Acknowledgment

1.06 The system's response to an RC input message is a TTY acknowledgment (TACK). If at anytime during an RC message input an input control character (!, %, ., &, or &) is received, the system responds ***immediately*** with some form of TACK. The TACK is immediate, in that, no other output message can intervene. The type of TACK received will indicate correct or incorrect input of message, or some system condition preventing acceptance of the input message. See Part 3 for a detailed description of the various TACK responses.

Output Message (OM)

1.07 The ESS provides RC18, RC16, and/or RC FAILURE output message in response to the RC input messages. Twenty-one types of RC18 messages are available to indicate accepted RC messages, irregular system conditions affecting RC messages, or rejected RC messages that contain errors (see Part 4). The RC16 output messages consist of data (in octal) for use as an aid in the analysis of an error resulting from the generic program. These OMs have a lower priority rating than a TACK and could be delayed by some higher priority message.

2. DESCRIPTION OF RECENT CHANGE MESSAGE PROGRAM LISTINGS

2.01 A separate message pident is provided for each RC input message with the exception of the delayed mode activation message RC:ACT which is contained in control pident RCDY (RC delay order) PR-1A308. Table B displays the RC input messages and their corresponding pidents, PR numbers and message indexes. Each pident is provided with a message head table, an input section, a validity section, a translation format section, and an error dictionary section which is useful in the interpretation of RC18 and RC16 output messages.

PROGRAM LISTING PAGE FORMAT

2.02 The PR follows a standardized format in presenting significant information about the program. The pident name consists of four alphabetic characters PPPP; the pident issue consists of up to seven alphanumeric characters PPPPAAA. The first four characters must be identical to the program name. The remaining three characters identify the generic program. At the top of each page is a title line giving the time and date of assembly, the pident name and generic program number, the issue number of the associated compool libraries, and the version number of the program. At the bottom of each page is a title line giving the pident name and program generic number, issue number, page number, and the PR number

TABLE B

RC MESSAGE INDEXES

MESSAGE INDEX (ii)	MESSAGE (RC:........)	MESSAGE PIDENT	PR REF. (PR-1A....)	MESSAGE INDEX	MESSAGE (RC:.......)	MESSAGE PIDENT	PR REF. (PR-1A....)
0	LINE	RCLI	336	38	TRFHC	RCHC	335
1	PSWD	RCPS	334	39	TRFSLB	RCLB	337
2	TG	RCTG	359	40	JUNCT	RCJG	334
3	RI	RCRI	347	41	PLM	RCPM	343
4	CHRGX	RCCI	326	42	TGBVT	RCBV	321
5	TRK	RCTK	360	43	OBS	RCOB	341
				44	CAMA	RCCA	322
7	CFTRK	RCCF	325				
8	PSBLK	PCPB	342				
9	TGMEM	RCTM	362	47	TOLDIG	RCTL	361
10	DIGTRN	RCDG	328	48	TNCTX	RCTX	366
11	RATPAT	RCRP	348	49	ANIDL	RCAI	367
12	SCLIST	RCSC	350	50	TOBS	RCTO	371
13	MLHG	RCHG	332	51	NMTGC	RCNM	372
14	MSN	RCSN	355	52	ACT	RCDY	308
15	CPD	RCCD	324	53	CTRF	RCCT	373
16	NOGRAC	RCNR	340	54	NUTS	RCNU	379
17	SUBTRAN	RCST	356	55	DALNK	RCDL	374
18	NOCNOG	RCNN	339	56	DATER	RCDT	376
19	CTXEXR	RCXR	369	57	DAMBI	RCDX	377
20	SIMFAC	RCSF	351	58	DAMSK	RCDM	375
21	CTXCB	RCCX	327	59	GENT	RCCT	378
22	DITABS	RCDI	330	60	CLAM	RCCM	380
23	CTXDI	RCXD	368	61	CFV	RCFV	383
24	CCOL	RCCC	323	62	UNASSIGNED (No.1 ESS Unit Type)		
25	ROTL	RCRT	370	63	CXDICH	RCXI	382
26	DNHT	RCDH	329	64	DLG	RCDA	381
27	TMBCGA	RCTC	357	65	(Reserved for AMPS)		
28	TNDM	RCTN	363	66	CCIS	RCIS	311
29	IDDD	RCID	333	67	CFG	RCCG	314
30	TDXD	RCTD	358	68	EPSID	RCEI	312
31	FLXDG	RCFD	331	69	AC	RCAC	315
32	FLXRS	RCRS	349	70	ACTABL	RCAT	316
33	FLXRD	RCRD	346	71	ESCO	RCES	317
34	MPTY	RCMP	338	72	ESN	RCEN	318
35	TWOPTY	RCTP	364	73	TNESN	RCER	384
36	PLUG	RCPU	345	74	TKCONV	RCTV	385
37	TRFLCU	RCTU	365				

(Fig. 1). In addition, within the program section, every program unit will start on a new page and have the program unit name at the top of each page.

PROGRAM LINE FORMAT

2.03 Each line in the program section of a PR contains a specified set of fields of information. It is not necessary for each line to include all fields of information as this depends on the type of instruction. The line format may include:

(a) Octal machine language

(b) Decimal numbers denoting line numbers

(c) Source data set line numbers

(d) Marco generation level numbers

(e) Program language statement

(f) In-line comments.

Figure 1 provides a layout of the line format and Table C gives a description of columns on the line format.

A. Pident Introduction

2.04 Each pident starts with Generic Feature Information and a Program Administration Section. The Generic Feature Information includes the generic and PR number and provides in feature and package parameters that are set for the particular assembly. The Program Administration Section includes program corrections (if any) that are included in the assembly.

B. Message Head Table

2.05 The message head table follows the Program Administration Section and contains the following:

- Pident message identification
- Relative address of associated tables
- Types of RC messages allowed (NEW, CHG, OUT)
- Number of associated keyword units
- Number of MSB (messge storage buffer) entries

Note: The message storage buffer is a storage facility in the call store area. The MSB consists of the MSB proper and auxiliary areas. The MSB proper consists of a one-word entry assigned for each keyword in the RC message, and the MSB auxiliary area is for use when the one-word entry is inadequate to store the keyword data.

C. Definition and Reference Tables

2.06 At the end of the pident program section, a list of symbol references and a list of macros are provided. Each symbol (name) appearing in the pident is listed and referenced to the page and line numbers of its occurrences. Figure 2(A) provides a layout of the definition and reference tables and Table D gives a description of the columns on the definition and reference tables.

2.07 A separate list of macros are provided as part of the cross reference list (following the symbol references) providing the ***count*** or times a macro is used within a pident and the page and line reference numbers applicable to the macro call. Refer to Fig. 2(B) for an example. The columns of the cross reference list and their corresponding descriptions are listed in Table D.

INPUT SECTION

2.08 The input section, following the message head table, always contains a keyword table. It may also contain an internal keyword table, a keyword equivalence table, and a keyword collision table. The input section is concerned with reading in data. A description of these four tables is given in paragraphs A through D.

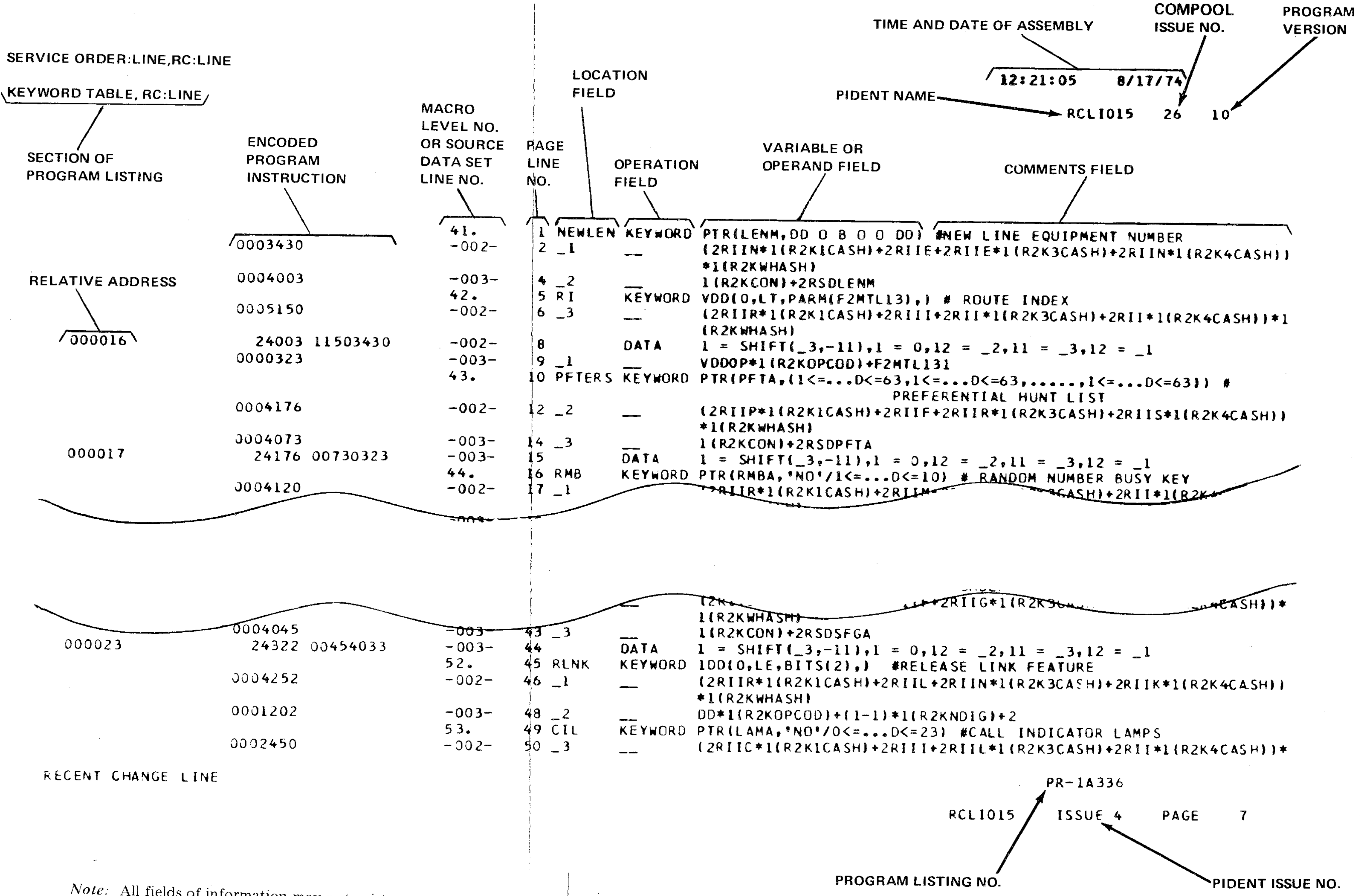
A. Keyword Table

2.09 The keyword table contains all valid keywords for the message in condensed hased form. This hashed form is a number created by combining the keyword character codes according to an arbitrary but fixed algorithm to achieve a compact code for the total keyword.

TABLE C

DESCRIPTION OF COLUMNS ON THE LINE FORMAT

FIELD OF INFORMATION	DESCRIPTION
RELATIVE ADDRESS	Octal representation of the displacement of the instruction or data that has been translated by SWAP on that line of the pident listing. In each pident, the 6-digit address begins with 000000; the number identifies the relative location of the word.
ABSOLUTE ADDRESS	(Core loaded program). The actual program store or call store address of the instruction.
ENCODED PROGRAM INSTRUCTION	Octal representation of the encoded program instruction or data. In No. 1 ESS the 13 digits to the right of each relative address represents an instruction or data containing 37 binary bits. In No. 1A ESS there are 8 digits to the right of the relative address containing 24 binary bits. In No. 1 ESS the three preceding 12-bit words are used to derive this data statement. In No. 1A ESS only two 12-bit words are used. This column may also contain relative octal addresses of the EQU psuedo-operation. In the case of recent change PRs this column may contain the octal representation of a macro expansion.
ADDRESS VALUE	One alpha character that denotes that the address computation for the instruction may not be complete and tells what the loader will do to complete the address computation for the instruction. "R" denotes relative address meaning that the loader does nothing to the address. "L" denotes local (absolute) address meaning that the loader will add in the starting address of the pident to the address. "V" denotes "extern" or a reference to an address external to this program which is resolved by going through the transfer vector table. It tells the loader to find the symbols address in the transfer vector table. "X" denotes "direct" or a reference to an address external to this program which should be resolved without an entry in the transfer vector table.
SOURCE DATA SET LINE NUMBER	A decimal number which specifies a line in the source data set. The lines of the source data set are continuously numbered from the beginning to the end of the program listing and are used when modifying a source program. The format of the source data set line number may be XXXXX.XXXX or XXXXX., where X is a decimal digit.
MACRO GENERATION LEVEL NUMBER	A level number which identifies which level of nested macros generated this line of the program listing. (A nested macro is a macro that is called by another macro.) The macro level number format is "-XXX-" where X is a decimal digit. Those lines which include macro level numbers are macro or EPL expansions produced by the assembler program from the last source data set line preceding the macro expansion line.
PAGE LINE NUMBER	Decimal digits denoting a line (01 to 50) per a page of a program listing.
LOCATION FIELD	A symbolic name (address) of a program point by which other instructions may refer to the instruction named. Such symbolic names are required at points to which control is transferred.
OPERATION FIELD	Symbolic instruction operation code, macro name, or pseudo-operation instruction code. The instruction code can consist of one to seven letters; macro names can be larger.
VARIABLE FIELD or OPERAND FIELD	Contains the parameters related to the operation designated in the operation field. The variable or operand field is formed by several subfields.
COMMENTS FIELD	Begins with a number sign (#) and contains comments, remarks, or references made by the programmer to aid in the maintainability and usability of the program listing.



TIME AND DATE OF ASSEMBLY
COMPOOL ISSUE NO.
PROGRAM VERSION
SERVICE ORDER:LINE,RC:LINE
KEYWORD TABLE, RC:LINE
SECTION OF PROGRAM LISTING
LOCATION FIELD
PIDENT NAME
MACRO LEVEL NO. OR SOURCE DATA SET LINE NO.
ENCODED PROGRAM INSTRUCTION
PAGE LINE NO.
OPERATION FIELD
VARIABLE OR OPERAND FIELD
COMMENTS FIELD
RELATIVE ADDRESS
PROGRAM LISTING NO.
PIDENT ISSUE NO.

```
                                                                                  12:21:05    8/17/74
                                                                          RCLI015    26    10

                                    41.      1 NEWLEN KEYWORD PTR(LENM,DD 0 8 0 0 DD) #NEW LINE EQUIPMENT NUMBER
           0003430                 -002-     2 _1     __      (2RIIN*1(R2K1CASH)+2RIIE+2RIIE*1(R2K3CASH)+2RIIN*1(R2K4CASH))
                                                              *1(R2KWHASH)
           0004003                 -003-     4 _2     __      1(R2KCON)+2RSDLENM
                                    42.      5 RI     KEYWORD VDD(0,LT,PARM(F2MTL13),) # ROUTE INDEX
           0005150                 -002-     6 _3     __      (2RIIR*1(R2K1CASH)+2RIII+2RII*1(R2K3CASH)+2RII*1(R2K4CASH))*1
                                                              (R2KWHASH)
 000016      24003 11503430        -002-     8        DATA    1 = SHIFT(_3,-11),1 = 0,12 = _2,11 = _3,12 = _1
           0000323                 -003-     9 _1     __      VDDOP*1(R2KOPCOD)+F2MTL131
                                    43.     10 PFTERS KEYWORD PTR(PFTA,(1<=...D<=63,1<=...D<=63,......,1<=...D<=63)) #
                                                                                    PREFERENTIAL HUNT LIST
           0004176                 -002-    12 _2     __      (2RIIP*1(R2K1CASH)+2RIIF+2RIIR*1(R2K3CASH)+2RIIS*1(R2K4CASH))
                                                              *1(R2KWHASH)
           0004073                 -003-    14 _3     __      1(R2KCON)+2RSDPFTA
 000017      24176 00730323        -003-    15        DATA    1 = SHIFT(_3,-11),1 = 0,12 = _2,11 = _3,12 = _1
                                    44.     16 RMB    KEYWORD PTR(RMBA,'NO'/1<=...D<=10) # RANDOM NUMBER BUSY KEY
           0004120                 -002-    17 _1             (2RIIR*1(R2K1CASH)+2RII                 CASH)+2RII*1(R2K

                                                      __      (2R                        +2RIIG*1(R2K3                   CASH))*
                                                              1(R2KWHASH)
           0004045                 -003-    43 _3     __      1(R2KCON)+2RSDSFGA
 000023      24322 00454033        -003-    44        DATA    1 = SHIFT(_3,-11),1 = 0,12 = _2,11 = _3,12 = _1
                                    52.     45 RLNK   KEYWORD 1DD(0,LE,BITS(2),)  #RELEASE LINK FEATURE
           0004252                 -002-    46 _1     __      (2RIIR*1(R2K1CASH)+2RIIL+2RIIN*1(R2K3CASH)+2RIIK*1(R2K4CASH))
                                                              *1(R2KWHASH)
           0001202                 -003-    48 _2     __      DD*1(R2KOPCOD)+(1-1)*1(R2KNDIG)+2
                                    53.     49 CIL    KEYWORD PTR(LAMA,'NO'/0<=...D<=23) #CALL INDICATOR LAMPS
           0002450                 -002-    50 _3     __      (2RIIC*1(R2K1CASH)+2RIII+2RIIL*1(R2K3CASH)+2RII*1(R2K4CASH))*

RECENT CHANGE LINE
                                                                                       PR-1A336

                                                                              RCLI015    ISSUE 4    PAGE    7
```

Note: All fields of information may not exist on every line throughout a program listing. For example, when keyword information (comments only) is presented, the SOURCE DATA SET LINE NUMBER and the PAGE LINE NUMBER only may be on a line. Also, the actual positions of these fields of information may vary in the program listing.

Fig. 1—Layout of a Typical Line Format (Table C Included)

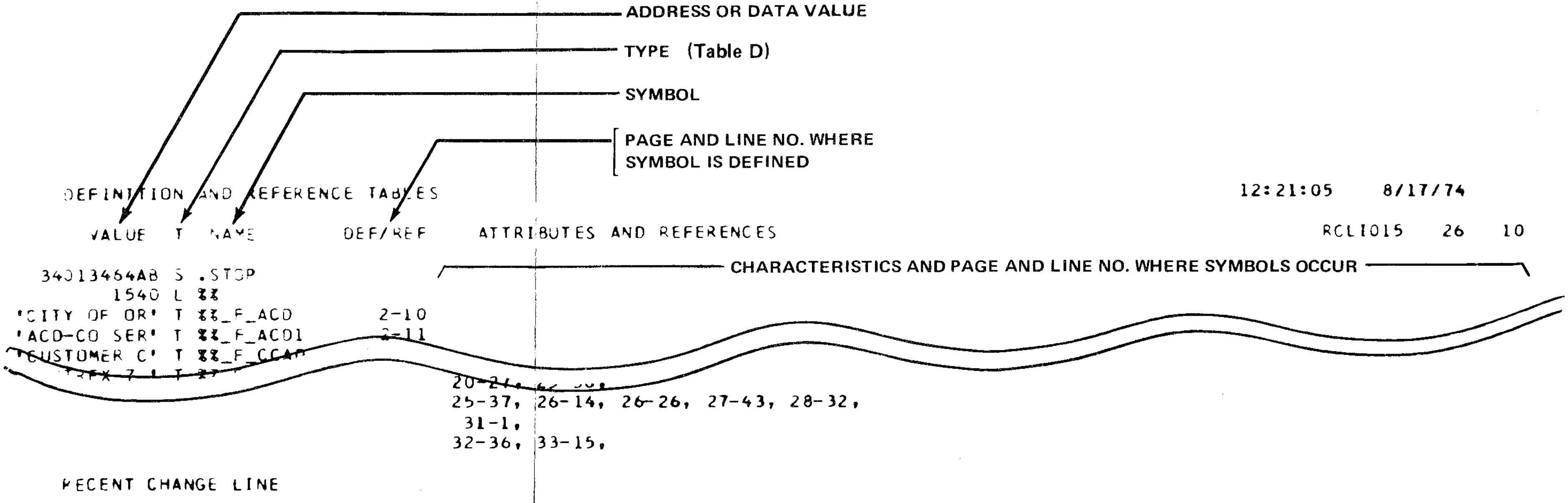
TABLE D
DESCRIPTION OF COLUMNS ON THE DEFINITION AND REFERENCE TABLES

COLUMN HEADING		MEANING
VALUE (for symbol list) or COUNT (for macro list)		Octal address or other data value dependent upon type of data as specified under the "TYPE" column.
T	T	Type of symbol as specified below:
	A	ABSOLUTE — fixed location of data in memory
	B	ATTRIBUTE NAME — the name assigned to a characteristic of a symbol
	C	CALL STORE TABLE — data address of first word in call store table, scatable or block
	F	PROGRAM STORE TABLE — data address of first word in program store table, scatable or block
	G	GLOBAL — pseudo-operation for global label with this name in the listing
	I	ITEM — address of data bits in data layout
	J	TRUTH VALUE or BOOLEAN — A value that a boolean expression is to have TRUE (FALSE) if its value is non-zero (zero)
	K	FUNCTION NAME — The name of a user or predefined function. This name may be used as an operand of an arithmetic expression just as a symbol or number might be used (ie, x(ae) causes any unqualified numbers, including letters A-F, to be interpreted as hexadecimal).
	L	LOCAL — pseudo-operation for local label with this name in listing
	M	MACRO NAME — instead of column "VALUE", all macros are listed in a separate list where the first column is "COUNT" (number of times macro is used in pident)
	N	PURE NUMBER — A string of numerics that represent an N-bit integer.
	O	MACHINE OPERATION — An instruction which when decoded provides an action for the machine hardware to perform.
	P	ITEM IN REGISTER — The layout of items. This is the name of the item which makes up the subdivision of the layout for a particular register.
	R	REGISTER — central control register name
	S	SEQUENCE SYMBOL — A source level symbol preceded by an underscore which has meaning only to the assembler
	T	TEXT — character string in quotes
	U	UNDEFINED SYMBOL — miscellaneous symbol
	V	EXTERN — pseudo-operation for an external location label in the vector table for a transfer to a point outside this program
	X	DIRECT — pseudo-operation for an external location label
	Y	ARRAY NAME — Name of an array of arithmetic values defined by using the ARRAY pseudo-operation
	Z	PSEUDO-OPERATION NAME — An ESS assembly statement that provides instruction to the assembler or loader and does not generate machine code

TABLE D (Contd)

DESCRIPTION OF COLUMNS ON THE DEFINITION AND REFERENCE TABLES

COLUMN HEADING	MEANING
NAME	Symbolic name in alphabetical order, special symbols first, numbers last
DEF/REF	Page and line number reference where symbol is defined
ATTRIBUTES and/or REFERENCES	Symbol characteristics and page and line number references of occurrences for symbol.


ADDRESS OR DATA VALUE
TYPE (Table D)
SYMBOL
PAGE AND LINE NO. WHERE SYMBOL IS DEFINED
CHARACTERISTICS AND PAGE AND LINE NO. WHERE SYMBOLS OCCUR

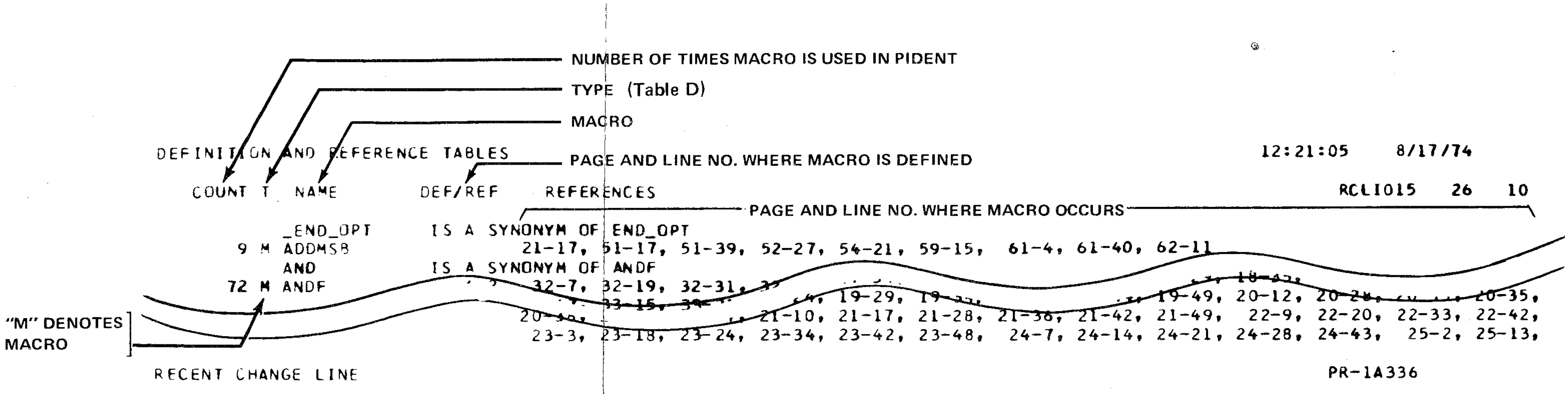
```
    DEFINITION AND REFERENCE TABLES                                                        12:21:05    8/17/74

       VALUE  T  NAME          DEF/REF    ATTRIBUTES AND REFERENCES                          RCLI015    26    10

     34013464AB S .STOP
           1540 L %%
  'CITY OF OR' T %%_F_ACD         2-10
  'ACD-CO SER' T %%_F_ACD1        2-11
  'CUSTOMER C' T %%_F_CCAP
                                         20-27,
                                         25-37, 26-14, 26-26, 27-43, 28-32,
                                          31-1,
                                         32-36, 33-15,

    RECENT CHANGE LINE
```

A (Symbol List)



```
DEFINITION AND REFERENCE TABLES                                      12:21:05    8/17/74
 COUNT T  NAME         DEF/REF    REFERENCES                         RCLI015    26    10
          _END_OPT      IS A SYNONYM OF END_OPT
     9 M ADDMSB           21-17, 51-17, 51-39, 52-27, 54-21, 59-15,  61-4, 61-40, 62-11
          AND           IS A SYNONYM OF ANDF
    72 M ANDF              32-7, 32-19, 32-31,
                                            19-29,                19-49, 20-12,               20-35,
                          20-             21-10, 21-17, 21-28, 21-36, 21-42, 21-49,  22-9, 22-20, 22-33, 22-42,
                           23-3, 23-18, 23-24, 23-34, 23-42, 23-48,  24-7, 24-14, 24-21, 24-28, 24-43,  25-2, 25-13,
```

NUMBER OF TIMES MACRO IS USED IN PIDENT
TYPE (Table D)
MACRO
PAGE AND LINE NO. WHERE MACRO IS DEFINED
PAGE AND LINE NO. WHERE MACRO OCCURS
"M" DENOTES MACRO
RECENT CHANGE LINE

B (Macro List)

Fig. 2—Layout of the Definition and Reference Tables (Symbol and Macro List) (Table D Included)

2.10 For each keyword, the table contains the following:

- Hashed form of the keyword.

- Either a description of the data that must accompany the keyword (including limited number of basic checks, such as type of data, number of digits, etc) or an index into the supplementary data assembly Table for more complex forms of keyword data.

- An indicator for allowing the keyword to be repeated in segmented messages.

B. Internal Keyword Table

2.11 The internal keyword table is actually part of the keyword table table is located at the end of the keyword table. The internal keyword table contains no user inputted keywords. It is merely a list of keywords used internally by the message pident. The purpose of this table is to assign MSB locations and truth value indexes to the internal keywords.

C. Keyword Equivalence Table

2.12 The keyword equivalence table will be located immediately after the internal keyword table. Entries in this table are made in order to specify any alternate names for keywords listed in the keyword table. The table contains:

- Hash code of alternate name

- Keyword index of the equivalent keyword in the keyword table.

D. Keyword Collision Table

2.13 The keyword collision table is located immediately after the keyword equivalence table. Entries are made into this table when two keywords have the same hash code. Each entry contains:

- Complete character codes for the collision keyword.

- Keyword index of the keyword in the keyword table.

VALIDITY SECTION

2.14 The validity section of a message pident always contains a data check table, validity (tree) table and a format selector table for each message type (NEW, CHG, OUT or equivalents) allowed by the message. In some cases, the validity tables for CHG messages will be combined with the validity tables for OUT messages. The validity section is concerned with whether the input data is valid. Following is a description of the three basic types of validity tables.

A. Data Check Table

2.15 The data check table contains the following:

- The detailed checks performed on keyword data which includes comparing keyword data with translation values, checking translations assignments, and testing for specific values of keyword data.

- The algorithms to calculate internal keyword value.

B. Validity (Tree) Table

2.16 The validity (tree) table specifies and contains the following:

- The valid combinations of keywords and keyword data by establishing a validity tree (each node on the tree is provided a name).

- A keyword (KW) value for each node and a set of tests that must be passed to set the truth value bit for a particular node. Some nodes cause RC18 messages to be printed if they fail the set of tests. In such cases, reference to the proper (NEW, CHG, or OUT) validity (tree) error dictionary (paragraph 2.29) will be required to determine the error.

C. Format Selector Table

2.17 The format selector table contains a set of Boolean operations to be performed upon the truth value bits to derive the format selector bits which control the execution of the translation format table.

TRANSLATION FORMAT SECTION

2.18 The translation format section always has a translation format table and may have a change transition screening table, change keyword list, and a change YES/NO keyword list. This section is concerned with the format of the translation output of the message. A description of the tables contained in this section is given in subparagrphs A through D.

A. Translation Format Table

2.19 The translation format table contains the data format description of the translator words changed by the RC message. Each item in a word of the affected translator is described in terms of size, displacement, auxiliary block word number, etc. The new contents of the translator words and items are described in terms of:

(a) truth value bits,

(b) format selector bits,

(c) data in the MSB (message storage buffer),

(d) data in the MSB auxiliary area.

The control pident RCTF (RC new pass translation format) PR-1A304 uses this data to build translaton data from MSB data and the MSB auxiliary area data. The control pident RCCH (RC change pass translation) PR-1A305 uses it to reconstruct MSB data, MSB auxiliary area data, and truth value bits from old data in translations.

B. Change Translation Screening Table

2.20 The change translation screening table is used only on CHG or OUT messages during the reconstruction of keyword value from old translation data under control of the translation format table. This screening table is checked each time a keyword is reconstructed. Entries into this table can indicate any of the following:

- Old and new data can be compared (failure to match results in RC18 CHGER).
- The truth value bit can be unconditionally set for the keyword.
- The new value of a keyword can be stored in the MSB location of an internal keyword and the old data from translations can be stored in the MSB location of the keyword.

C. Change Keyword List

2.21 Entries into the change keyword list are used at the end of the change pass of the format table to move data from one keyword to another in preparation for a new pass of the format table and allows truth table bits for the moved data to be updated.

D. Change YES/NO Keyword List

2.22 Entries must be made into the change YES/NO keyword list for all YES/NO and NO/data keywords if CHG or OUT are valid forms for the message. The entries are used at the end of the CHG pass to correctly set the truth value bit for these keywords, as follows:

- If a value for the keyword was input, then the new value is used.
- If the keyword has not been input and the old data from translations indicates NO, then the truth value bit remains 0.
- If the keyword has not been input and the old data incidates YES or data, then the truth value bit is set and the old data value is entered in the MSB.

ERROR DICTIONARY SECTION

2.23 Following the data tables in a PR is the pident error dictionary. The error dictionary section is designed to be used in conjunction with the following three types of output messages:

- ***RC18 INPUT***—An input error has been detected while the program was checking the keyword unit.
- ***RC18 VALER***—A validity error has been detected—specifically, a fatal node has failed. Reference to the validity (tree) error dictionary is required to determine the error.
- ***RC18 XLER***—A translation error has been detected:

(a) A nonexistent head table or translator

(b) An invalid address for a required pointer associated with an auxiliary block in translation

(c) Erroneous pointer to a word that should be located in a translation auxiliary block (lies beyond the end of the block).

2.24 The error dictionary section consists of some or all of the following parts, as appropriate for the particular message.

A. Keyword Table

2.25 The keyword table located in the error dictionary section contains all valid keywords for a message as well as the data type associated with each keyword (YES/NO, data/NO, data) and a brief description o the keyword. The keyword table will be located at the beginning of the error dictionary section.

B. Synonymous Keyword Table

2.26 As indicated by its name, the synonymous keyword table contains a .list of keywords and their equivalents. The use of either of two equivalent keywords will be accepted by the ESS and mean the same thing. The synonymous keyword table follows the keyword table in the error dictionary section.

C. Keyword (Table) Error Dictionary

2.27 The keyword (table) error dictionary is referenced when the table identifier in the RC18 INPUT error message is KWT. The dictionary contains index, data type, and error description of associated keyword units by separate NEW, CHG, and OUT tables, as appropriate for the message.

D. Data Check Table Dictionary

2.28 The data check table dictionary contains indices, labels, and descriptions of each data check performed on associated keyword units. The tables are provided with prefixed NEW, CHG, and OUT to the table title, as appropriate for the message.

E. Validity (Tree) Error Dictionary

2.29 The validity (tree) error dictionary contains nodes, labels, and descriptions of each validity check performed. The tables are provided with prefixed NEW, CHG, and OUT to the table title as appropriate for the message. The validity (tree) error dictionary is referenced when a test established by the validity (tree) table has failed, resulting in an RC18 VALER output message.

F. Format Selector List

2.30 The format selector list contains a list of format selector names and their corresponding equivalencies. This error dictionary is generated ***only*** for format selectors of type NEW messages.

G. Validity Tree Diagram

2.31 The validity tree diagram (applicable to earlier generic programs only) provides a graphical description of valid keyword unit combinations for use in RC messages. The diagram uses node symbols to associate validity checking procedures with the cause and source of rejected input messages.

(a) The validity tree processing consists of performing Boolean functions upon previously defined truth values (keyword units, data checks, and validity tree nodes). The validity tree starts with a single word (12 bits) which defines the Boolean functions of the first node. The two functions of a node are:

(1) For the node to be true or false

true: the node value is 1

false: the node value is 0

(2) For the node to fail (and, in turn, the message) or not.

Where a node failure causes a message failure (rejection), it is called a ***fatal*** node. Where a node failure does not in itself result in an input message rejection, the node is called ***nonfatal.***

(b) Table E lists the nodes and symbols descriptions used in graphical presentations of validity tree diagrams.

TABLE E

VALIDITY TREE NODES

SYMBOL*	NODE	NODE = 1 if and only if:	FATAL if and only if:	DESCRIPTION
&	AND	all args† = 1		AND (non-fatal)
•	ANDF		Node = 0	AND, Fatal if fail
*	ANDFN		Node = 0 and Sum ≠ 0	AND, Fatal if fail and Sum of args non 0
†	OR	at least one arg = 1		OR (non-fatal)
	ORF		Node = 0	OR, Fatal if fail
X	XORF	exactly one arg = 1	Node = 0	Exclusive OR, Fatal if fail
#	XORFN		Node - 0 and Sum ≠ 0	Exclusive OR, Fatal if fail and sum of args‡ non 0
@	BCF	all args have same value	Node = 0	Biconditional, Fatal if fail
0 (†)	OPT (for any)	any arg of list 1 = 1, or all args of list 2 = 0	Node = 0	Optional: FORANY (list 1) OPTIONAL (list 2)
0 (&)	OPT (if all)	all args of list 1 = 1, or all args of list 2 = 0	Node = 0	Optional: IFALL (list 1) OPTIONAL (list 2)
a | (b)	IMP (a⇒b)	a=1=b	a=1 and b=0	Implies

*These symbols were used on validity tree diagrams that appear in some message pident PSs. However, they are not reliable because they have not been updated, and are being removed from all PRs.

†args is arguments

‡Where the ***sum of args*** is the node value sum of the adjacent node located on all branches extending from a specified or referenced node.

SHARED PIDENTS (RCTS AND RCSI)

2.32 Shared pident RCTS (RC table subroutine) PR-1A319 and pident RCSI (RC shared information) PR-1A320 will sometimes be referenced while using other pidents in connection with problem analysis procedures.

2.33 Pident RCTS consist of special purpose subroutines and are used by RC message pidents—input (through pident RCSI), data check, and format—and by control pident RCWL (RC work list) PR-1A307. The subroutines provide functions that are too specialized to be handled efficiently by the macro languages of input, data check, and format.

2.34 Pident RCSI consists of five data tables accessed by several code pidents during RC message processing. The tables are as follows:

(1) ***Message Head Table:*** The message head table contains a vector for each RC message pident. This table is used by control pident RCIG (RC and general control) PR-1A300.

(2) ***TAG and Assignment Table:*** Each table entry provides the structure of a translator in terms of head table, number of bits in selector and index, primary translation word (PTW) and head table unassignment codes, and power of expansion values. These tables are used by routine RSGTG in control pident

RSUB (RC subroutines) PR-1A309 when generating the program store address (TAG) of a translator.

(3) ***Parameter Table:*** The table is a transfer vector (TV) table for the parameters used in format checking of the keyword data. This table is used by control pident RCKI (RC keyword input) PR-1A302.

(4) ***TV Table for Supplementary Input Tables:*** Each table entry contains relocatable addresses of two supplementary input table assembly routines. The table is used when the entry in the keyword table of message pident points to a supplementary input table assembly routine in RCSI. This table is used by control pident RCKI.

(5) ***Supplementary Input Table:*** Each table entry consists of two or more 12-bit words that describe the data of a keyword and indicate whether to store the data in the MSB or MSB auxiliary area. Any of the RC message pidents can point to entries in this table. This table is used by control pident RCKI, using the TV table. Routines in this table may point directly to other routines in the table without having to refer back to the TV table.

3. TTY ACKNOWLEDGMENT (TACK) SYSTEM RESPONSE

3.01 A TACK is an ***immediate*** response from the ESS after a specific input control character is received. The term immediate means that no other output can intervene. There are seven TACK categories:

(1) Acceptable

(2) Line Error

(3) Message Error

(4) Busy

(5) Invalid Heading

(6) Internal Program Error

(7) Timeout

A. Acceptable TACK (OK)

3.02 Acceptance of an input is indicated either by a carriage return and line feed or a printed OK immediately following an input control character. A carriage return and line feed occurs following an input check (/), a line cancel (#), or a message cancel (&) character. An OK is printed immediately following an end-of-message (! or .) or an end-of-segment (%) character. The carrier return and line feed or OK indicate that the system is available for use, that no format errors were detected in the input, and that the information is entered into the system as instructed. In the case where a line cancel (#) or message cancel (&) character was inputted, the acceptable TACK response indicates that the required action has been taken. Examples of the acceptable TACK follows:

```
RC:LINE:
ORD 4804
TN 9227468
OE 01302103
LCC 1FR!OK

            RC18   0    4804    ACPT
                        6/5     18:20
```

```
RC:LINE;CHG:
ORD 1765
TN 9222828
OE 01004502
ICP!OK

            RC18   0    1765    ACPT
                        7/19    13:55
```

```
RC:TWOPTY;OUT:/
ORD 3614/
TN 9220130!OK

           RC18   35   3614   ACPT
                       5/30   11:54
```

```
RC:SCLIST:/
ORD 5
TN 8920102
CODE 1.DGS   2228811%OK

CODE 2.DGS   6247230%OK

CODE 3.DGS   6248588!OK

           RC18   12   5      ACPT
                       3/26   10:25
```

B. Line Error TACK

3.03 The line error TACK is printed immediately following the input check character (/) and the end-of-segment character (%) in the following form:

aa = Two letter TACK code indicating type of error detected (Table F)

bb = Character (column) number, in octal, of the first character which may be in error. The character (column) spaces of the printout line are numbered from left to right, including spaces, beginning with 1.

Upon determining the error, the erroneous input line can be retyped correctly into the system without affecting the preceding portion of the message. Examples of line error TACK are shown below:

```
RC:LINE;OUT:/
ORD 1211/
TN 922165/MD 12
```

(more decimal digits expected in TN)

```
RC:LINE:/
ORD 1505/
TN 9228950/
OE 01014501/
TTC 7/IA 05
```

(invalid character received for TTC)

TABLE F

LIST OF ERROR CODES APPEARING IN CORRECTABLE ERROR TACKS* AND IN RC18 OUTPUT MESSAGES

ERROR CODE	OCTAL CODE	TYPE OF ERROR
DE	00000021	Data exceeds allocated buffer (too many arguments for the multiargument keyword)
DL	00000020	Data lower bound violated (data too small)
DS	00000030	Discrete data not valid (data does not equal any of the specific valid values)
DU	00000007 00000011 00000016	Data upper bound exceeded (data too large)
	00000017	Data greater than or equal to upper bound
I	INED	Invalid backspace when just after a carriage return and no character yet typed into the buffer. The — (backspace) appears as ← on some versions of IOT keyboards.
IA	00000015 00000033	Invalid character; alphanumeric (letter or number) expected
	00000046	Invalid character; hyphen or alphanumeric expected
IB	00000014	Invalid character; binary digit expected
IC	INED	Illegal character (not one of the set of characters recognized as valid by the recent change program)
ID	00000012	Invalid character; decimal digit expected
IN	00000032	Invalid character; numeric digit expected
IO	00000013	Invalid character; octal digit expected
IS	00000023	Invalid character, range-hyphen (—) or comma (,) expected
IT	00000044	Invalid TOUCH-TONE® digit
KI	00000000 00000002	Keyword invalid for this message
KR	00000001	Keyword repeated in message illegally
KT	00000003 00000027	Keyword unit termination valid or missing where expected (too many characters in fixed-length data fields)
KU	00000045	Keyword unavailable because associated package is not loaded

TABLE F (Contd)

LIST OF ERROR CODES APPEARING IN CORRECTABLE ERROR TACKS* AND IN RC18 OUTPUT MESSAGES

ERROR CODE	OCTAL CODE	TYPE OF ERROR
LP	00000026	Left parenthesis missing
MB	00000036	More binary digits expected
MD	00000034	More decimal digits expected
MK	00000004	More keywords than permitted on one line
MO	00000035	More octal digits expected
RP	00000025	Right parenthesis missing
SE	00000031	Error detected by input subroutine. (See pident RCSI in PR-1A320 or in PR-6A320 for more information)
SI	00000005	Segmenting illegal in this message
YN	00000022	Invalid data for YES/NO keyword

* A correctable error TACK consists of one of these codes followed by a 2-digit decimal column number of the error.

C. Message Error TACK (ER)

3.04 ER is the message error TACK and is printed immediately following the ! (end of message symbol) or the % (end of segment symbol). The ER TACK indicates that a noncorrectable error is contained in the message and that the entire message or message segment is removed from the system. The message must be reinputted after correction as indicated by the RC18 message which follows the ER. Examples of the ER TACK follow:

```
RC:LINE:
ORD 3504
TN 9620400
OE 02317101
LCC 1MB!ER

        RC18   0    3504    INED
                            IC    00000077    LC00010006  INP  00000017  STA 00000003
                    5/30 11:53
```

```
RC:LINE;CHG:
ORD 1287
MLH 5
TER 2
OE 01302001
NEWOE   01302203
TN 9221390
NONH!ER

        RC18   0    0       INPUT
                                   00000000   KI  00020006  KWT    00000340
                          7/3 15:08
```

```
RC:LINE:
ORD 1987
TN 9220680
OE 02207401
LCC 1MB!ER

        RC18   0    1987    VALER
                        NEW 00000231
                    7/19 13:53
```

```
RC:LINE:
ORD 1902
TN 9225196
OE 04614600
LCC 1MB
GND!ER

        RC18   0    1902    XLER
        PTA  00000000  CON  00000000  TAG  00000000  UNA  15236210  TIN  17000406
          30000000
                    6/11  5:29
```

D. Busy TACK (BUSY)

3.05 The BUSY TACK is printed immediately following the !, /, and % input characters. This TACK indicates that the recent change programs are busy on another channel and that the input must be repeated until an OK TACK is printed. An example of the BUSY TACK message is shown below:

```
RC:LINE;OUT:
ORD 1654
TN 9221366!BUSY


RC:LINE;OUT:/BUSY

RC:LINE;OUT:/BUSY

RC:LINE;OUT:
ORD 1654
TN 9221366!OK


          RC18    0     1654    ACPT
                          7/19 14:00
```

E. Invalid Heading TACK (?IH)

3.06 The ?IH is the invalid heading TACK and is printed immediately following a / input character on the first line of a message indicating that an error has been detected in the heading line. Upon correction of the error, the heading line must be reinputted. An example of the ?IH TACK is shown below:

```
RC:LIME:/?IH
(Line misspelled)
```

F. Internal Program Error TACK (<INTERR>)

3.07 <INTERR>is the internal program error and is printed immediately following the !, /, and % input characters. The <INTERR> TACK notifies the operator that the program has an internal error and the raw data dumps provided in the RC16 messages can be analyzed to determine the program error. The input message cannot be processed until the program error is corrected. An example of the <INTERR> TACK follows:

```
RC:LINE:
ORD 305
MLH 355
TER 32
OE 00406501
LCC RXR
CTX 2
TN 8941581
STAH
FREE
MSN 100407!<INTERR>

     25  RC16  INTERR
         00022106  00000000. . .00000002

     .     .     .

     .     .     .

     .     .     .

     26  RC16  INTERR
         00000002  00000001. . .00000450
         27770036  00022106. . .20000005
         00002002  01206347. . .01176123
```

G. Timeout TACK (<TIMEOUT>)

3.08 The <TIMEOUT> TACK is printed immediately following an IOT character (except ! and .) in a message that is typed into the system after 45 seconds has elapsed since the last character was typed, or if a tape input is used and the tape has been stopped for 45 seconds. It is necessary to reinput the entire message. An example of the <TIMEOUT> TACK is shown below:

```
RC:LINE:
ORD 13
TN 861 1999
OE 07003703
LCC 1FR <TIMEOUT>
```

4. TTY OUTPUT MESSAGES FOLLOWING THE TACK

4.01 The output message (OM) following the TACK of an RC input message is an RC18 or an RC16 output message. These OMs may indicate acceptance of an input message, may explain errors detected in the input message, or may provide a series of data dumps to analyze problems arising from program errors.

A. RC18 Output Message

4.02 Almost all output messages encountered are RC18 messages. An RC18 message may indicate acceptance of an RC input message, or provide data resulting from successful execution of particular messages. Other RC18 messages provide coded data for error diagnosis and/or identification of error sources.

4.03 The RC18 output message consists of 21 types which fall into 5 categories as follows:

Note: **RC18 output message types are explained in Table G.**

RC18 MESSAGE TYPE	CATEGORY
ACPT	Nonerror
BRKRC	
INFO	
CHGER	Errors that do not require reference to the PR error dictionary.
DELAY	
INED	
MTYPE	
NOCR	
OBJT	
PUNCT	
INPUT	Errors that require reference to the PR error dictionary.
VALER	
XLER	
NAUX	Software problems (system resources exceeded)
NAWL	
NLLS	
NOPS	
NPRC	
PLUGF	
PRTER	Errors that pertain to internal generic program
TABER	

4.04 The format of an RC18 message and the associated explanation are as follows:

MESSAGE INDEX	ORDER NUMBER	RC18 MESSAGE TYPE
RC18 i̸i	n̸n̸n̸n̸n̸n	t̸t̸t̸tt
xxx zzzzzzzz	xxx zzzzzzzz . . .	xxx zzzzzzzz
1st Data Unit	2nd Data Unit	Last Data Unit (Variable Number)

i̸i = Message index which is a decimal number 1 or 2 digits and identified the RC message (See Table B).

n̸n̸n̸n̸n̸n = The 1-to-6 digit (with possible letter prefix) order number assigned to the RC input message (keyword ORD). If no order number was inputted, a 0 will be printed.

t̸t̸t̸tt = A 2-to-5 letter code identifying the RC18 message type (See Table G).

xxx zzzzzzzz = Data units as explained in Table G. The quantity of units may vary from none to a maximum of 5 per line as required for each type message.

TABLE G

RC18 OUTPUT MESSAGE TYPES

RC18 TYPE (ŧŧŧtt)	EXPLANATION	xxx zzzzzzzz UNITS
ACPT	Message accepted	
BRKRC	Break received (manual interruption of the input program)	

CHGER	Change transition error Mismatch on a CHG or OUT order between keyword input and existing translation data	xxx = MSB (Message Storage Buffer)
		zzzzzzzz = MSB index

DELAY	Error in delayed message, in activation, or deactivation of a delayed message	xxx = SEG — Segmented Message A message of more than one segment cannot be delayed. = ORD — incorrect or no order number specified. In delayed message, the order number must be given, must be less than 262114, and cannot be preceded by a letter character. = ASN Either of two assignment errors: (1) A delayed message entered with an order-number that is the same as an existing delayed order. (2) An activation or deactivation order number that does not match any existing delayed order in memory. zzzzzzzz = Not used for this type message.
INED	Input editor error	xxx = IC (invalid character), or = I_[invalid cancel (backspace) beyond the start of the line] zzzzzzzz = aaa000rr, where aaa = American Standard Code for Information Interchange (ASCII) code of the invalid character and rr is the RC (6-bit) code (see Table H). Both aaa and rr are in octal. — 1st unit xxx = LC (Line Column) zzzzzzzz = rrrrcccc where rrrr = octal line number (see Note) and cccc = octal number of the invalid character. — 2nd unit xxx = INP (Input) zzzzzzzz = Octal column of state table (input) — 3rd unit

TABLE G (Contd)

RC18 OUTPUT MESSAGE TYPES

RC18 TYPE (ƀƀƀtt)	EXPLANATION	xxx zzzzzzzz UNITS	
INED (contd)	Input editor error (contd)	xxx = STA (State) zzzzzzzz = rrrrrrss, where ss = last state and rrrrrr = three states preceding last state (each a 2-digit octal number)	4th unit
		Note: The line number starts with 1 for the heading line.	
INFO	Information associated with successful execution of an RC message	xxx = NOG (Number Group) zzzzzzzz = Octal number group number	1st unit*
		xxx = MBI (Mask Block Index) zzzzzzzz = Octal mask block index number	1st unit*

		xxx = ADD (Address)
		zzzzzzzz = Octal address of PS block seized
			1st unit+

		xxx = OCT (Length)
		zzzzzzzz = Octal length of PS block seized
			2nd unit+

		xxx = HTA (Head Table Address) zzzzzzzz = Octal head table address that must be initialized to the PS block address	3rd unit+
		*Only the 1st unit appears for RC:NOCNOG and RC:DAMSK messages. For RC:NOCNOG the 1st unit will be the number group (NOG). For RC:DAMSK the 1st unit will be the mask block index (MBI). These messages will then be followed by a standard RC18 ACPT message.	
		+Only the 1st and 2nd units appear for RC:PSBLK and RC:SUBTRAN messages using keywords LNGR, LNGL14, or LNGL23. In all other RC:SUBTRAN messages, two RC18 INFO messages appear automatically: 1st and 2nd units appear in the 1st RC18 INFO message and the 3rd unit appears in the 2nd RC18 INFO message. These messages will then be followed by a standard RC18 ACPT message.	
INPUT	Input error	xxx = Three blank spaces zzzzzzzz = Octal error code used by pident RCKI PR-1A302 (see Table F for a listing of error codes)	1st unit
		xxx = Error code (2 or 3 letters — see Table F)	

TABLE G (Contd)

RC18 OUTPUT MESSAGE TYPES

RC18 TYPE (ttttt)	EXPLANATION	xxx zzzzzzzz UNITS
INPUT (contd)	Input error (contd)	zzzzzzzz = rrrcccc where rrr = octal line number (see Note) and cccc = octal column number of the input error xxx = Input Table Identifier: KWT (keyword table) in the applicable PR section or SIT (supplementary input table in PR-1A320). zzzzzzzz = Error index. If KWT: index into keyword (table) error dictionary. If SIT; index into supplementary input table error dictionary. *Note:* The third unit is meaningless if the error code in the second unit is DE, KI, KT, LP, RP, or SI, or if the system is unable to read a keyword because of an error. *Note:* The line number starts with 1 for heading line.
MTYPE	Message type identifier error (The message type identifier is the second subfield of the second field — the symbol after the first semicolon in an RC message).	
NAUX	Not enough RC auxiliary area available.	
NAWL	Not enough available work list space (reduce size of RC meassage)	
NLLS (No. 1)	Not enough link list scratch available	
NOCR	No carriage return after first line (first two fields) of message	
NOPS	No program store block (or not enough) available.	
NPRC	Not enough primary RC register available	

TABLE G (Contd)

RC18 OUTPUT MESSAGE TYPES

RC18 TYPE (ffftt)	EXPLANATION	xxx zzzzzzzz UNITS
OBJT	Message object identifier error (The message identifier, or object, of the message is the first subfield of the second field — the symbol after RC:)	
PLUGF	Plug-up Failure — Message is the result of an attempt to plug-up more lines for service observing than allowed by office parameters. (Refer to Section 231-118-328)	
PRTER	Print Call Error (Internal program error)	
PUNCT	Punctuation Error in Heading Line	
TABER	Table Specification Error (Internal program error)	xxx = TAD (Table Address) zzzzzzzz = 1 + 12-bit PS address where error was detected
VALER	Validity Error	xxx = Validity Type: NEW or C/O (change or out type message) zzzzzzzz = Octal mode number in validity (tree) error dictionary of the applicable PR section

XLER	Translation Error	One or more of the following units will appear:
		xxx = PTA (Primary tag assignment)
		zzzzzzzz = Octal index into pident RCSI's tag and assignment error dictionary (PR-1A320) in order to identify the translator associated with a problem.
		xxx = CON (Contents)
		zzzzzzzz = Contents of primary translation word (PTW) if TAG (next unit) is nonzero, otherwise, garbage
		xxx = TAG
		zzzzzzzz = Tag (address of PTW)

TABLE G (Contd)

RC18 OUTPUT MESSAGE TYPES

RC18 TYPE (t̸t̸t̸tt)	EXPLANATION	xxx zzzzzzzz UNITS
XLER (contd)		xxx = UNA — unassigned head table or translator ALG — auxiliary block length exceeded AAD — auxiliary block address invalid (aux block does not exist)
		zzzzzzzz = 1 + 12-bit PS address where error was detected
		xxx = TIN (Translation Input) zzzzzzzz = Translation input (for example, directory number or line equipment number)
		xxx = IDX (Index or #) zzzzzzzz = Current truth index in data check table

B. RC18 Interpretation Procedure

4.05 The meaning of an RC18 output message may be determined in the following manner:

(1) The message index (i̸i) is used to identify the message by using Table B.

(2) The order number of the associated RC input is specified in the m̸n̸n̸n̸n̸n̸n location.

(3) The message type (t̸t̸t̸tt) is explained in Table G.

(4) If the message type is one of the applicable error types, data unit(s) will identify the cause(s) of the system rejection by use of Table G. If the message type is INFO, the data units will specify information concerning the memory storage area for the RC input as specified in Table G.

C. ACPT-Type RC18 Message

4.06 The ACPT message indicates that the input message was accepted and executed by the system as specified. An example follows:

```
RC:LINE;OUT:
ORD 4611
TN 9221554!OK

          RC18   0    4611    ACPT
                 6/5  17:58
```

TABLE H

RC 6-BIT AND ASCII CODES ASSOCIATED WITH INPUT TERMINAL CHARACTERS AND OPERATIONS

TELETYPEWRITER CHARACTER OR OPERATION	RC 6-BIT CODE (rr)	ASCII CODE (aaa)	TELETYPEWRITER CHARACTER OR OPERATION	RC 6-BIT CODE (rr)	ASCII CODE (aaa)
0	00	060	U	36	125
1	01	061	V	37	126
2	02	062	W	40	127
3	03	063	X	41	130
4	04	064	Y	42	131
5	05	065	Z	43	132
6	06	066	— (DASH)	51	055
7	07	067	TAB	52	011
8	10	070	:	53	072
9	11	071	;	54	073
A	12	101	SPACE	55	040
B	13	102	,	56	054
C	14	103	)	57	051
D	15	104	(	60	050
E	16	105	NEW LINE	61	012
F	17	106	RETURN	62	015
G	20	107	!	63	041
H	21	110	%	64	045
I	22	111	"	65	042
J	23	112	&	66	046
K	24	113	X OFF	70 (Note 1)	023
L	25	114	RUB OUT	70 (Note 1)	177
M	26	115	VT (Vertical Tab)	70 (Note 1)	013
N	27	116			
O	30	117	FORM (Form Feed)	70 (Note 1)	014
P	31	120	$	72	044
Q	32	121		73	137
R	33	122	_(UNDERSCORE)	77	(Note 2)
S	34	123			
T	35	124		77	000 (Note 3)

Note 1: Note that 70 is the RC code for RUBOUT, X OFF, VT and FORM. All are ignored, so there is no reason to distinguish between them.

Note 2: Any ASCII code not used by the RC program is converted to RC code 77.

Note 3: This combination is used to indicate an input character parity error.

D. BRKRC-Type RC18 Message

4.07 The BRKRC message indicates that a manual break of interruption of the input program was received by the system. This would normally occur as a result of depressing the BREAK key on any input keyboard connected to that channel. The input message must be correctly reinputted into the system. The following example illustrates the use of the BREAK key following the ! in the first message and its correct reinput:

```
RC:LINE:
ORD 1505
TN 9228950
OE 01014501
LCC 1R!

?

        RC18   0    1505    BRKRC
               5/30  11:19


RC:LINE:
ORD 1505
TN 9228950
OE 01014501
LCC 1MR!OK

        RC18   0    1505    ACPT
               5/30  11:20
```

E. CHGER-Type RC18 Message

4.08 The CHGER message indicates that there is a discrepancy between a keyword of a CHG or OUT input message and the existing translation information in the system. This could be an error such as attempting to CHG or OUT a feature on a line which does not exist in the translation information for that line or inconsistency in input information. An example of a CHGER message follows:

```
RC:LINE;OUT:
ORD 12
TN 8938338
OE 07402502!ER

        RC18   0     12      CHGER
        MSB 00000035
        3/28  16:51
```

F. DELAY-Type RC18 Message

4.09 The DELAY message indicates an error in a delayed input message which falls into three categories as follows:

(1) SEG—A segmented input message (containing more than one segment using a % character) ***cannot*** be delayed.

(2) ORD—An incorrect or no order number was in the input message.

(3) ASN—An assignment error either by inputting a delay order number that is already in the system or inputting an activation or deactivation order for a delay order number which does exist in the system.

The corrected message must be inputted. Examples of the DELAY message follow:

```
RC:SCLIST;;DELAY:
ORD 26
TN 7273081
ADN 3
DGS 5843742%ER

        RC18   12    26      DELAY
     SEG
     7/24  9:28

    (Segmented Message)


RC:LINE;OUT;DELAY:
TN 8684730
CTX 7!ER

        RC18   0     0       DELAY
    ORD
    6/6  10:27

    (No Order Number)


RC:ACT;OUT:
ORD 300!ER

        RC18   52    300     DELAY
    ASN
    6/6  10:26
```

G. INED-Type RC18 Message

4.10 The INED message indicates an input editor error. The output message contains four data units explaining the error and its location by use of Tables G and H. Correct any errors present and reinput message. An example follows with an input character parity error and reinput:

```
RC:LINE:
ORD 4722
TN 9226370
OE 03006202
LCC 1MR!ER

      RC18   0   4722   INED
                     IC 00000077  LC 00040003  INP 00000017  STA 00000003
                 6/5 18:16


RC:LINE:
ORD 4722
TN 9226370
OE 03006202
LCC 1MR!OK

      RC18   0   4722   ACPT
                 6/5 18:17
```

H. INFO-Type RC18 Message

4.11 The INFO message specifies storage location for the input message in up to three data units. (See Table G). This output message is only used with the RC:NOCNOG, RC:DAMSK, RC:PSBLK, and RC:SUBTRAN input messages. As examples the following are presented:

```
RC:DAMSK:
GSZ 154
TERS(10, 11, 12, 13, 14, 15)%OK
TERS(16, 17, 18, 19, 20, 21, 22, 23, 24, 25)!OK

RC18  58  0  INFO
MBI  00000035
      3/20 11:06

RC18  58  0  ACPT
      3/20 11:06



RC:PSBLK:
ORD 131
NEW R12!OK

RC18  8  131  INFO  ADD  02163432  OCT  000014
      6/13 3:14

RC18  8  131  ACPT
      6/13 3:14
```

I. INPUT-Type RC18 Message

4.12 The INPUT message indicates that an error was made in the input message to the system. The following procedure should be used to locate and analyze the error:

(1) Determine which RC message is in error by using the message index (i̸i) and, if specified, the order number (n̸n̸n̸n̸n̸n) in the RC18 message. Refer to Table B for the listing of RC message indexes. For those messages identified by error code DE, KI, KT, LP, RP, or SI the ***error index*** (3rd data unit) should be ignored. The line and column information can still be used to locate the characters in error and, usually, the definition in Table F of the error code is self-explanatory.

(2) Using Table G, locate INPUT type message. Convert the second data unit octal line and column (character) number (zzzzzzzz) to decimal numbers. (See Fig. 3.) Use the decimal line and column numbers to locate the characters in error. Begin line counting with the heading line as number 1. Count column (character) numbers from left to right, including spaces, beginning with 1. The error code (xxx) (Table F) defines the type of error.

	OCTAL NUMBER								
	UNITS DIGIT								
TENS DIGIT	0	1	2	3	4	5	6	7	
0	0	1	2	3	4	5	6	7	DECIMAL NUMBER
1	8	9	10	11	12	13	14	15	
2	16	17	18	19	20	21	22	23	
3	24	25	26	27	28	29	30	31	
4	32	33	34	35	36	37	38	39	
5	40	41	42	43	44	45	46	47	
6	48	49	50	51	52	53	54	55	
7	56	57	58	59	60	61	62	63	

(OCTAL 34 = DECIMAL 28, DECIMAL 34 = OCTAL 42)

Fig. 3—Octal-Decimal Conversion Table

(3) Determine the input message error and the corrections necessary to input an acceptable message.

(4) Reinput the corrected RC message for acceptance by the system.

(5) If the second data unit error code (xxx) is not self-explanatory, determine by using the 3rd data unit which input table error dictionary should be used to obtain more details.

- If the 3rd data unit (xxx) is KWT, locate the keyword table error dictionary associated with the applicable input message program PR- (Table G).
- If the 3rd data unit (xxx) is SIT, locate the supplementary input table error dictionary in PR-1A320 (pident RCSI).

The 3rd data unit's octal number (zzzzzzzz) is the index number into the appropriate error dictionary. The error dictionary will then provide data types and a more detailed error description. Table I lists the codes and symbols used in the data type column of an error dictionary and a description of these codes.

An example of an INPUT output message containing an incorrect TN follows:

```
RC: LINE:
ORD 0005
OE 00000003
TN 5733141!ER

   RC18  0  5  INPUT

         00000031 SE 00040012  SIT 00000257
            4/16 10:50
```

J. MTYPE-Type RC18 Message

4.13 The MTYPE message indicates an error in the message type identifier which is located on the RC heading line in the right subfield of the identification (2nd) field (usually NEW, CHG or OUT). Reinput corrected message. An example follows with an incorrect ICP input:

TABLE I

INPUT DATA TYPE SYMBOLS IN PROGRAM LISTINGS

Data Type	Meaning
D	Decimal digit (0-9)
O	Octal digit (0-7)
B	Binary digit (0-1)
C	Alphabetic character (A-Z)
A	Alphanumeric character (A-Z and 0-9)
CHAR	Also used for character when character can be alphabetic or numeric
. . .D	Variable number of decimal digits with leading zeros optional
. . .O	Variable number of octal digits with leading zeros optional
N	Digit from 2 to 9 (used as first digit of office code; example, NXX)
X	Digit from 0 to 9 (used as second and third digits of office code; example, NXX)
/	Exclusive OR operation. Example, YES/NO means type YES or NO.
‘ ’	Denotes a literal to be inputted. Example, ‘RG’ means must input RG.
| |	Encloses an optional character. Example, |C| means alphabetic character is optional.
<=	Less than or equal
, . . . ,	Stands for the word through when used in an expression that defines a series where only the first and last members of the series are stated in the expression, such as MIN1 , . . . , MIN5

Note: These codes and symbols are used to describe keyword data in the keyword lists, the keyword table error dictionaries, and the supplementary input table error dictionary.

```
RC:LINE;ICP:
ORD 412
IN 5735030
OE 06410702!ER

          RC18    0  0     MTYPE
          7/3  6:29
```

K. NOCR-Type RC18 Message

4.14 The NOCR message indicates that there was no CR and NL following the heading line (first two fields). Reinput message correctly. An example follows:

```
RC:LINE;OUT:ORD 2333
TN 9221914!ER

          RC18    0      2333    NOCR
                         7/19    14:01
```

L. OBJT-Type RC18 Message

4.15 Th OBJT message indicates an error in the message object identifier which is located on the RC heading line in the left subfield of the identification (2nd) field (LINE, TWOPTY, MLHG, SCLIST, etc). Reinput corrected message. An example follows of an incorrect (misspelled) LINE input:

```
RC:LIST:
ORD 1
TN 572087!ER

          RC18    53    0     OBJT
          4/11  9:08
```

M. PUNCT-Type RC18 Message

4.16 The PUNCT message indicates a punctuation error in the heading line (first message line). Correct error and reinput message. An example (omitted colon) follows:

```
RC:LINE;OUT
ORD 33
TN 5349745
OE 02114300!ER

          RC18    0      0        PUNCT
          7/24  14:54
```

N. VALER-Type RC18 Message

4.17 Validity errors (VALER) are detected by ***programmed checks*** that are performed to assure that the combination of the input keywords and the keywords data are consistent with the existing data base (translation memory). These validity errors appear as ***fatal node failures*** and are reported by the VALER type RC18 output message (Table G).

4.18 Three examples of errors which would be reported as validity errors are:

(1) Attempting to assign a line already assigned or to unassign a line which is not assigned

(2) Attempting to assign an unworkable combination of features to a line

(3) Attempting to use RC CHG messages for transitions not permitted by the system.

4.19 An example of a VALER message follows:

```
RC:LINE;OUT:
ORD 1099
TN 9223732!ER

            RC18    0      1099    VALER
                             C/O  00000245
                           7/19  14:01
```

In this example, the PR-1A336 Change Validity (Tree) Error Dictionary, RC:LINE; lists the error description of Node 245 as TN must be assigned.

O. XLER-Type RC18 Message

4.20 The XLER message reports that instructions contained in the RC input message cannot be accomplished with the existing system translation data, which could be inconsistent input data (Table G). Such inconsistencies may also result in a VALER or RC16 INTERR output message. The RC TAG and assignment error dictionary (PR-1A320) list three types of translation errors reportable by the XLER output message as follows:

TRANSLATION ERROR Type	DESCRIPTION
UNA	Head table, translator, or subtranslator is unassigned
ALG	Auxiliary Length error; index exceeds auxiliary block length or attempts to access a word beyond the end of an auxiliary block.
AAD	Invalid auxiliary block address.

4.21 The UNA-type translation output message indicates an error in input data or a procedural error in the sequence of related input messages.

4.22 The ALG- and AAD-type of translation output messages indicate an error in translation data that require further analysis.

4.23 The remainder of the RC TAG and assignment error dictionary provides tabulated information from the TAG and assignment table in pident RCSI (PR-1A320) that identifies all translations. The principal use of the tabulations is to identify a translator that is associated with a problem by using the primary TAG assignment (PTA) index provided in the RC18 output message.

4.24 An example of the XLER message follows:

```
RC:LINE;OUT:
ORD 12
TN  5346256
OE  00114320!ER

      RC18   0    12       XLER
      PTA  00000000  CON  00257027  TAG  02547225
      ALG  31200444  TIN  17400406       #10000000
                                          7/10 6:40
```

4.25 The RC16 INTERR output message is explained in paragraphs 4.30 through 4.38.

P. NAUX, NAWL, NLLS, NOPS, NPRC, PLUGF, PRTER, and TABER Types of RC18 Message

4.30 There are eight types of output messages which pertain to internal generic program problems or resource limitations (Table G). These types of RC18 output messages are:

MESSAGE	CORRECTIVE ACTION
NAUX	RC update or cardwrite.
NAWL	Reduce size of message.
NLLS	Cut in spare program store module.
NOPS	Increase size of translator.
NPRC	RC update or cardwrite.
PLUGF	Remove some lines from service observing.
PRTER	Receive assistance from local TAC, SCCS, or PECC location.
TABER	Receive assistance from local TAC, SCCS, or PECC location.

4.27 The PRTER RC18 message results from a print call error where the print routine expects a particular character(s) for printing but the character is not received (Table G). INFO RC18 messages that provide information associated with an accepted input message but did not provide the expected information to the print routine is one situation where a PRTER message results. Parity check failures are also causes for the PRTER message. Assistance should be requested from the local TAC, SCCS, or from the PECC location.

4.28 The TABER RC18 message results from a table specification error, such as an invalid RC18 message (Table G). An invalid message is generated when an incorrect value is loaded into the buffer for printing. A typical cause for TABER messages is that an associated table is expected to contain particular information, but does not. Assistance should be requested from the local TAC, SCCS, or from the PECC location.

Q. RC16 Output Messages

4.29 RC16 output messages indicate that the system has encountered problems which apparently are caused by internal errors in the program. Detection of such an error results in a series of seven RC16 INTERR message segments which consists of a varying quantity of eight digit numbers. The segments consists of:

RC16 INTERR MESSAGE NUMBER	QUANTITY OF WORDS
1. CC Registers	6
2. States & Pointers	10
3. Input Control	5
4. Truth Value Tables	16
5. Format Selector	4
6. First Part of PDS	14
7. More PDS	21

These words are data dumps which should provide information to determine the problems encountered in the system program. An example of the RC16 message follows:

```
                RC:LINE:
                ORD 305
                MLH 355
                TER 32
                OE 00406501
                LCC  RXR
                CTX  2
                TN 8941581
                STAH
MESSAGE         FREE
NUMBER          MSN 100407!<INTERR>

   1                  25   RC16    INTERR
                           00022106  00000000  00000000  01177152  15177765  00000002

   2                  25   RC16    INTERR
                           04100000  00001002  00022100  00022153  00022300  00022420  00022432
                           00207334  00207334  00210007

   3                  25   RC16    INTERR
                           00000041  04000006  00000002  00000014  00000001

   4                  26   RC16    INTERR
                           00000461  00000601  00000231  00000401  00000404  00000200  00000203
                           00000010  00000000  00000110  00000040  00000020  00000100  00000003
                           00000000  00000000

   5                  26   RC16    INTERR
                           10010100  00000240  00000000  01177100

   6                  26   RC16    INTERR
                           00022100  00000024  00003062  37777777  01214532  20000005  00000211
                           00000001  21004543  11400002  37777777  00377777  00000000  00000000

   7                  26   RC16    INTERR
                      ↑    00000002  00000001  00000000  01177621  21004543  00000000  00000450
                      |    27770036  00022106  15177765  00000002  00022124  01301001  20000005
                      |    00002002  01206347  00000560  00022124  01176102  20000005  01176123
                      |
```

Number of
Minutes after
the hour

R. RC16 Interpretation Procedure

4.30 RC16 message data analysis requires ability to interpret data in the program listings.

First RC16 INTERR Printout

4.31 The first RC16 INTERR printout message segment consists of six words that provide the octal contents of central control registers at the time an internal error is detected. The sequence of the register content words for are:

```
FFFFFFFF KKKKKKKK  LLLLLLL XXXXXXXX  YYYYYYYY  ZZZZZZZZ
```

The L register word is most important because it contains the return address and thus provides the code pident location at which the error was detected. By proceeding to the location in the specified pident pointed to by the L register, the description of the reason for the INTERR output message is provided. Also, the same description allows interpretation of the other associated registers, as well as the use of the remaining segments of the message.

4.32 Although the contents of the central control registers vary in accordance with the specific INTERR call, the programming requirements used for the RC system permit some general statements about the probable content of the registers.

(a) ***L Register:*** Address of the client program that called the INTERR routine. The correct interpretation of the location pointed to by the L register is the most critical step in the interpretation of the message printout segments. In some instances, the L register word provides all required information. In other instances, some or all of the remaining printout segments must be interpreted. The L register's use within the control pident RSUB is of special importance because RSUB consists of shared subroutines referenced by other code pidents. For this case, L normally indicates the error identity, but does not specify when or the location where the error occurred. For this case, reference is required to the state words and push down scratch to determine the error location.

(b) ***X Register:*** In control pidents RCKI, RCVC, RCFI, RCCH, or RCTF, the X register typically contains the 12-bit address in the table where the INTERR error occurred. Thus, isolation of the error is permitted to a 12-bit word in the message pident.

(c) ***F, K, Y, and Z Registers:*** Use the L register and pident to determine contents.

Second RC16 INTERR Printout

4.33 The second RC16 INTERR printout message segment consists of state and pointer words. The following information shows ten call store words starting with address 17406 through 17417.

```
R2PSW1 R2PSW2 R2PDS   R2NAPDS  R2MSB  R2MSBAUX  R2NAMSBA
R2WL  R2NAWL   R2EWL
```

The four words that provide information of the greatest significance are as follow:

- R2PSW1 Code pident currently in control
- R2NAPDS Points to the next available word of the push down scratch
- R2NAMSBA Specifies next available word in the MSB auxiliary area
- R2NAWL Specifies the next available work list location.

Where the error is identified but not the error location, use C(R2NAPDS) -1 to C(R2PDS), which is all used push down scratch, to construct a history of events leading up to the internal error. Since the scratch system provides client addresses and in some cases the contents of the central control registers when a client called a subroutine, the problem can be identified by working backward from the last used push down scratch entry at C(R2NAPDS) -1/

> *Note:* The R2PDSMSB (points to RC PDS and MSB) program store location provides the size of the push down scratch area (R2PDS and R2NAPDS-128 words) and the size of the message storage buffer (R2MSB) and its associated message buffer auxiliary area (R2MSBAUX and R2NAMSBA) in the left half of the program store word and the starting address of the call store block in the right half of the program store word. The call store block is used exclusively by the RC message interpretation.

Third RC16 INTERR Printout

4.34 The third RC16 INTERR printout message segment provides five input control words located in call store with addresses starting with 17450 as follows:

- 17450 ASCII character last processed
- 17451 Input editor state
- 17452 Line buffer unload pointer
- 17453 Current line
- 17454 Current column.

4.35 The ASCII to RC character table in pident RCIE is used to convert the 7-bit ASCII character code received on the I/O channel into the 6-bit character code used internally by the RC programs.

4.36 The state table for input editor is used to obtain subroutine codes (index to ISUBRT) and next state index which points the program to the proper character subroutine for processing I/O characters.

Fourth and Fifth RC16 INTERR Printout

4.37 The fourth and fifth RC16 INTERR printout message segments provide a Boolean history of input, data check, validity, and format selection of the RC input message. By inspecting the truth value table (fourth printout segment), keyword units that were received can determine which data checks were true, and which validity tree nodes were true. The format selector (fifth printout segment) provides the selected paths by using a format based on the input and translation data.

Sixth and Seventh RC16 INTERR Printout

4.38 The sixth and seventh RC16 INTERR printout message segments provide a part of the 128-word push down scratch. The sixth segment prints the first 14 words of the 128-word push down scratch. The seventh segment prints 21 words: 14 words prior and seven words after R2NAPDS (second RC16 INTERR fourth word) which is located within the 128-word push down scratch. In some circumstances, T-READ procedures are required to determine additional words in the 128-word push down scratch.

5. ABBREVIATIONS AND ACRONYMS

5.01 Table C and D provide a description of information fields located within the line format (Fig. 1) and the definition and reference tables (Fig. 2) which will not be included in the abbreviations. Other abbreviation items are as follows:

AAD—Auxiliary block address invalid (auxiliary block does not exit)

ACPT—Message accepted

ADD—Address

AGN—Assignment

ALG—Auxiliary block length exceeded

ASCII—American Standard Code for Information Interchange

ASN—Assignment

BRKRC—Break received

CHGER—Change error

C/O—Change or out type message

CON—Contents

CR—Carriage return

DE—Data exceeds allocated buffer

DELAY—Error in delay message

DL—Data lower bound violated (data too small)

DS—Discrete data not valid (data does not equal any of the specific valid values)

DU—Data upper bound exceeded (data too large)

ER—Error

HTA—Head table address

I—Invalid backspace

IA—Invalid character - alphanumeric (letter or number) expected

IB—Invalid character - binary digit expected

IC—Illegal character

ID—Invalid character - decimal digit expected

IDX—Index

IM—Input message

IN—Invalid character - numeric digit expected

INED—Input editor error

INFO—Information (associated with execution of message)

INP—Input

INPUT—Input error

INTERR—Internal error

IO—Invalid character - octal digit expected

I/O—Input/Output

IS—Invalid character - range hyphen (-) or comma (,) expected

IT—Invalid TOUCH-TONE® digit

KI—Keyword invalid for this message

KR—Keyword repeated in message illegally

KT—Keyword unit termination invalid or necessary when expected

KU—Keyword unavailable because associated package not loaded

KW—Keyword

KWT—Keyword table

LC—Line column

LP—Left parenthesis missing

MB—More binary digits expected

MBI—Mask block index

MD—More decimal digits expected

MK—More keywords than permitted on one line

MO—More octal digits expected

MSB—Message storage buffer (a storage facility in the call store area)

MTYPE—Message type

NAUX—Not enough auxiliary block available

NAWL—Not enough available work list space

NEW—New

NL—New line (previously line feed)

NLLS—Not enough link list scratch available

NOCR—No carriage return after two fields of messages

NOG—Number group

NOPS—No program store block

NPRC—Not enough primary recent change registers available

OBJT—Message object identifier error

OCT—Octal

OM—Output message

ORD—Order number

PG—Program generic

Pidents—Program identification

PLUGF—Plug-up failure

PR—Program listing

PRTER—Print call error

PTA—Primary tag assignment

PTW—Primary translation word

PUNCT—Punctuation error in heading line

RC—Recent change

RCCH—Recent change pass translation format (PR-1A305)

RCDY—Recent change delay order (PR-1A308)

RCFI—Recent change format interpretation (PR-1A306)

RCIE—Recent change input editor (PR-1A301)

RCIG—Recent change initialization and general control (PR-1A300)

RCKI—Recent change keyword input (PR-1A302)

RCSI—Recent change shared information (PR-1A320)

RCTF—Recent change new pass translation format (PR-1A304)

RCTS—Recent change table subroutine (PR-1A319)

RSUB—Recent change subroutines (PR-1A309)

RCVC—Recent change validity check (PR-1A303)

RCWL—Recent change work list (PR-1A307)

RCxx—One pident per recent change input message. (See Table A)

RP—Right parenthesis missing

SCCS—Switching Control Center System

SE—Error detected by subroutine (see pident RCSI in PR-1A320 for more information)

SEG—Segmented

SI—Segmenting illegal in this message

SIT—Supplementary input table (PR-1A320)

SMB—Segmented

STA—State

SWAP—Switching Assembly Program

TABER—Table specification error (internal progam error)

TAC—Technical assistance center

TACK—TTY acknowledgment

TAD—Table address

TAG—Address of primary translation word

TIN—Translation input

TTY—Teletypewriter

TV—Transfer vector table

UNA—Unassigned head table or translator

VALER—Validity error

XLER—Translation error

YN—Invalid data for YES/NO keyword

!—Exclamation point - end of message symbol

.—Period - end of non-RC message

%—Percent sign - execute and save common data

/—Slash - input check

$—Dollar sign - line cancellation

&—Ampersand - message cancellation.