MASTER

IS - 2244

Mathematics and Computers
(UC-32) TID-4500

UNITED STATES ATOMIC ENERGY COMMISSION

Research and Development Report

AN IMPLEMENTATION OF A PL/I SYSTEMS
PROGRAM WHICH DEMONSTRATES THE
FEASIBILITY OF SYSTEMS PROGRAMMING
IN A HIGH LEVEL LANGUAGE

by

G. E. Hedrick

NOTICE

**PORTIONS OF THIS REPORT ARE ILLEGIBLE. It
has been reproduced from the best available
copy to permit the broadest possible avail-
ability.**

January 1970

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

Ames Laboratory
at
Iowa State University of Science and Technology
Robert S. Hansen, Director
Contract W-7405 eng-82

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

IS - 2244

This report is distributed according to the category Mathematics and Computers (UG-32) as listed in TID-4500.

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

- A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or
- B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

NOTICE

PORTIONS OF THIS REPORT ARE ILLEGIBLE. It
has been reproduced from the best available
copy to permit the broadest possible avail-
ability.

Printed in the United States of America
Available from
Clearinghouse for Federal Scientific and Technical Information
National Bureau of Standards, U. S. Department of Commerce
Springfield, Virginia 22151
Price: Printed Copy \$3.00; Microfiche \$0.65

CONTENTS

ABSTRACT	v
SECTION I. PRELIMINARY ANALYSIS	1
1.1 Purpose of Program	1
1.2 Theoretical Development	1
SECTION II. METHOD OF IMPLEMENTATION	4
2.1 Statement of the Algorithm	4
2.2 The Program	4
SECTION III. OPERATING INSTRUCTIONS	8
APPENDIX I. FLOW CHART	10
APPENDIX II. LISTING	13

IS - 2244

IS - 2244

AN IMPLEMENTATION OF A PL/I SYSTEMS PROGRAM
WHICH DEMONSTRATES THE FEASIBILITY OF SYSTEMS
PROGRAMMING IN A HIGH LEVEL LANGUAGE

by

G. E. Hedrick

ABSTRACT

A method for writing systems programs in high level (PL/I) language instead of the more cumbersome assembly language is presented. Such programs are loader programs which permit one of the standard job steps to be circumvented, possibly resulting in a savings of computer time.

IS - 2244

SECTION I. PRELIMINARY ANALYSIS

1.1 Purpose of the Program

The purpose of this program is to show the convenience of using a high level language to perform systems work. A loader program is written in PL/1. This program was written in about six man-months of time, a great saving when compared to the normal amount of time required for writing such systems programs.

Since the PL/1 compiler does a great deal of the work which would normally be done in the writing of the assembly program, less time is required to implement a systems program by this technique than by the standard method using assembler language. The programmer is not saddled with the problem of remembering what is in every register at any specified time.

Although this program is written in PL/1, any high level language with string manipulation operation available could be used. The detailed logic is peculiar to the IBM 360, but the generalized process is applicable to any loader program. The use of a loader increases the speed of execution of a total program since the linkage-editor step may be bypassed.

1.2 Theoretical Development

The purpose of the loader is to bypass the OS/360 linkage editor to obtain speed, and it is this fact that is in mind when the loader is not assigned all of the capabilities of the system linkage-editor. Certain restrictions must be made in the interest of speed while others must be made in the interest of the coding effort and logic synthesis which are involved.

Every control section which is to be used must be included in the object deck so that disk search time in seeking library functions is reduced to zero. This also effects a corresponding reduction in the coding which is necessary to

handle automatic library calls.

Since the most popular language in use for scientific programs at the Ames installation is FORTRAN, all object modules may be restricted to those which have been generated from FORTRAN source cards.

There are several different types of records which may be used as input to the loader. External symbol dictionary records provide information about control sections, external symbols, common, and pseudo-registers. An external symbol dictionary record is eighty bytes long and is divided into several fields. Each record may give information for as many as three items which are or need to be in the external symbol dictionary. There is always at least one of these items in an external symbol dictionary record.

Relocation dictionary records, like external symbol dictionary records, are eighty bytes long and divided into several internal fields. They are used to indicate at what points, internal to a control section, there are references to a different control section or to some other external symbol. It is possible to hold an address table for each control section or other external symbol. Then when a relocation directory record is encountered, it should be a trivial matter to chain through the table to alter the internal structure of the proper control section.

The eighty-byte text records have only one field other than the identifying fields. That field contains the actual text of the program to be loaded. This text could then be loaded directly.

The eighty-byte end records signal the end of the current set of control sections and common blocks. This may allow for some table updating if address tables are used. It may also provide some convenient space for internal program bookkeeping.

The loader only accepts eighty-byte records. Any record which has not pre-

viously been described is ignored.

When it is observed that every type of record that may possibly be encountered by the program is exactly eighty bytes long, then it is possible to observe that the sequential records may be input to the same storage area. The problem which arises is one of formatting. This may be reduced in its severity by using the capabilities of PL/1. A simple eighty-byte character string may be used as an input area, and by the use of the unaligned and based attributes it is possible to overlay a variable corresponding to each particular type of record. These variables in turn should be structure variables to force each separate field to be defined at the proper location and be given a name within the over-all structure.

Another consideration is that of passing control to the user program. This also may be greatly simplified by the use of the PL/1 capabilities. A label variable may be assigned a value which is legal in the current block; an unaligned bit string may be overlayed on this variable; and boolean operations may be used to substitute the address of the user's first control section.

To return to the loader from the FORTRAN program requires that the finish condition be intercepted by the loader. This will prevent the transfer control to the OS monitor.

Error checking may be performed throughout the loader program. Each check should be placed at a location where it has been empirically determined that a large number of errors occur. Information messages may be treated almost synonymously with error messages. Information messages, however, do not require the amount of corrective action that is required during the recovery of an actual error.

SECTION II. METHOD OF IMPLEMENTATION

2.1 Statement of the Algorithm

The entire program is based on one algorithm which has had necessary additions and deletions made to it from time to time. The skeleton of this algorithm is:

- 1) read an input record
- 2) test the record type
- 3) go to the area to process that record type
- 4) repeat from 1) until an end-of-file is encountered.

A flow chart which describes this algorithm in more detail is given in Appendix 1.

2.2 The Program

The program is written as one PL/1 procedure in which there are several on conditions and sections which are almost logically independent. Each one of these conditions and sections of code will be considered in its turn.

The FORTRAN condition is programmer defined and is raised at the completion of a load. When this condition is raised, the required internal bookkeeping is performed, the address of the initial entry to the user program is calculated, and control is passed to the calculated address.

The ERROR condition is the standard PL/1 condition and is raised whenever an unrecognizable error occurs. If it is impossible to recover from an error, the loader abnormally terminates, printing a message to that effect.

The FINISH condition is the standard PL/1 condition and is used to prevent direct returns to the OS/360 monitor.

The ENDFILE condition is also the standard PL/1 condition and is used to detect the end of input to the loader.

The code to handle the processing of text records is at label TXT. One replacement statement is used for this processing. Only one statement is required due to the availability of the PL/1 substring function.

The code to handle the processing of external symbol dictionary records begins at labels ESD and continues to immediately preceding label RLD. This code is itself divided into several sections depending upon the type of external symbol dictionary entry that is under consideration.

Code common to all external symbol dictionary types is processed starting at label ESDR. This code determines the type of entry under consideration, handles basic bookkeeping, and transfers to the appropriate section.

Pseudo-registers are handled by updating the name table and then updating the pseudo-register vector. Each successive component of the pseudo-register vector contains the name of a different pseudo-register. The dimension of this vector remains undefined until the last pseudo-register has been encountered. The final dimension of the pseudo-register vector is simply the total number of pseudo-registers that were encountered during the loading process.

A simple external reference entry requires only that the name, address, and adjustment tables be set to the appropriate values. These entries will then be referenced by some other external symbol dictionary entry.

A private code external symbol dictionary entry causes a renaming process to occur. Unnamed private code is renamed as if it had been written as a main program. As a result, the name table is updated following the address, length, and adjustment table updates.

External symbol dictionary entries for label reference, external section definition, and label definition either are not supported or require a minimum amount of processing by the loader programs.

Almost all of the external symbol dictionary entries require some kind of

table update. The section definition entry is no exception. The names, address, length, and adjustment tables must be updated each time that a section definition is encountered. In addition, a check must be made to determine if the virtual length of the control section is stated on the end record. If it is given on the end card, the updating of the length table is postponed until the corresponding end record is processed. The location counter is updated and the storage alignment is forced to a double word boundary. This allows any future entries that must be aligned on a double word boundary to be aligned properly.

The "common" external symbol dictionary reference entry is processed in the same way as a section definition entry. The essential difference is that these entries are entered from the back of the user memory area toward the front whereas the section definitions are entered front to back.

Relocation dictionary records permit the updating of the tables which have been created by the external symbol dictionary record processing. The relocation dictionary records also allow the loader program to resolve the external references in the user program. These records indicate at what point in a user control section an external reference is made and where in the address table the address of that reference may be found. Once this address is found, it is entered into the control section in which the external reference occurs completing the patching of external references.

End records are processed by either updating the length table or essentially performing no computation. The length table will be updated only if the corresponding section definition has indicated that the length of the current control section is stated on this end record.

This loader program is very restricted and is designed to be used to demonstrate the feasibility of systems programming in a high level language. The restrictions which apply to this program but do not necessarily apply to the

system program include: 1) only object modules which have been created from FORTRAN source programs are permitted; 2) no automatic library calls are permitted; 3) the maximum length for the entire user program must be 16K or less; and 4) very few of the standard options are supported. It may be observed, therefore, that the IBM loader should be used whenever it is available.

SECTION III. OPERATING INSTRUCTIONS

There are several different modes of operating the loader program. The most common and by far the easiest is to load and execute one object module which was created from a FORTRAN source program. This may be accomplished with the following job control:

```
//JOBNAME    JOB    'ACCT,TIME=nn,SIZE=96K;NAME
//JOBLIB      DD      DSN=PROG.A0246.LOADER,VOL=SER=LIBPAK,UNIT=2314,DISP=SHR
//S1          EXEC      PGM=LOADER
//SYSPRINT     DD      SYSOUT=(A,NOPAGING),
//              DCB=(RECFM=FBA,LRECL=125,BUFNO=1,BLKSIZE=1879)          X
//SYSIN        DD      *
                user program
/
/*
//FT01F001    DD      *
                user data
/*
//FT03F001    DD      SYSOUT=(A,NOPAGING),
//              DCB=(RECFM=FBA,LRECL=125,BUFNO=1,BLKSIZE=1879)          X
/*
```

A second mode of operation is that in which it is desired to obtain a listing of the source code of the loader as well as to have the loader function.

The job control to effect this is:

```
//JOBNAME      JOB    'ACCT,TIME=nn,SIZE96K',NAME
//EXEC          PLIF,          X
//              PARM.PL1L='MACRO',          X
//              REGION.GO=96K
```



```
//PL1L.MACRO DD DSN=PROG.A0246,MACROLIB,DISP=SHR,UNIT=2314, X
```

```
// VOL=SER=LIBPAK
```

```
//PL1L.SYSIN DD *
```

```
%INCLUDE MACRO(LOADESC);
%INCLUDE MACRO(DECLARE);
%INCLUDE MACRO(CONDITI);
%INCLUDE MACRO(STARTLP);
%INCLUDE MACRO(TEXTCRD);
%INCLUDE MACRO(ESDCARD);
%INCLUDE MACRO(RLDCARD);
%INCLUDE MACRO(OTHEOUT);
%INCLUDE MACRO(ABTERMN);
```

```
/*
```

```
//GO.SYSIN DD *
```

```
user program
```

```
/*
```

```
//GO.FT01F001 DD *
```

```
user data
```

```
//GO.FT03F001 DD SYSOUT=(A,NOPAGING),
```

```
// DCB=(RECFM=FBA,LRECL=125,BUFNO=1,BLKSIZE=1879) X
```

```
/*
```

In order to test the loader program the following three cards should be substituted for the

```
//SYSIN DD *
```

```
card or the
```

```
//GO.SYSIN DD *
```

card in either of the preceding examples:

```
//SYSIN DD DSN=SEQ.A0246.TSTLD,DISP=SHR,UNIT=2314,VOL=SER=LIBPAK
```

```
// DD *
```

```
/
```

APPENDIX I. FLOW CHART

LOADER

APPENDIX II. LISTING

```
%INCLUDE MACRO(LOADESC); %INCLUDE MACRO(DECLARE);
```

COMPILE-TIME MACRO PROCESSOR
MACRO SOURCE2 LISTING

```
1 %INCLUDE MACRO(LOADESC); %INCLUDE MACRO(DECLARE);
2 %INCLUDE MACRO(CONDITI); %INCLUDE MACRO(STARTLP);
3 %INCLUDE MACRO(TEXTCRD); %INCLUDE MACRO(ESDCARD);
4 %INCLUDE MACRO(RLDCARD); %INCLUDE MACRO(OTHEOUT);
5 %INCLUDE MACRO(ABTERMN);
```

INCLUDED TEXT FOLLOWS FROM DD.MEMBER = MACRO .LOADESC

```
6 /*          FORTRAN LOADER PROGRAM          */          00000100
7 LOADER: PROC(PARM) OPTIONS(MAIN);          00000500
8 /*          00000700
9 DESCRIPTION OF THE PROGRAM:          00000900
10 -----          00001100
11          THIS PROGRAM IS DESIGNED TO BE USED TO LOAD COMPLETE          00001300
12 OBJECT PROGRAMS FROM OBJECT CARDS. THERE ARE SOME          00001500
13 RESTRICTIONS ON THE NATURE OF THE JOBS WHICH WILL BE ACCEPTED          00001700
14 AND THERE ARE SOME MODIFICATIONS WHICH SHOULD BE MADE TO THE          00001900
15 PROGRAM TO INCREASE ITS SPEED AND EFFICIENCY. AN ATTEMPT          00002100
16 HAS BEEN MADE TO HAVE THE PROGRAM TO BE LARGELY          00002300
17 SELF-DOCUMENTING, PRIMARILY BY THE USE OF JUDICIOUSLY          00002500
18 CHOSEN VARIABLE NAMES. COMMENTS ARE USED TO OBTAIN THIS          00002700
19 DOCUMENTATION TO SOME EXTENT BUT THE LARGER PART IS DONE          00002900
20 VIA VARIABLE NAMES.          00003100
21          00003300
22          00003500
23 METHOD OF SOLUTION          00003700
24 -----          00003900
25          THE OVERALL METHOD OF SOLUTION IS PERFORMED IN THREE          00004100
26 PHASES: 1) INITIALIZATION PHASE, 2) LOAD PHASE,          00004300
27 AND 3) EXECUTE PHASE.          00004500
28          THE INITIALIZATION PHASE CONSISTS: 1) OF DESCRIBING          00004700
29 THE INPUT FORMATS OF THE VARIOUS TYPES OF OBJECT CARDS;          00004900
30 2) OF SETTING TRAPS TO HANDLE ERRORS IN A LOAD AND FOR          00005100
31 TERMINATIONS OF A LOAD; AND 3) OF WRITING A HEADING FOR THE          00005300
32 OUTPUT FROM THE LOADER. THE INPUT FORMATS ARE DESCRIBED BY          00005500
33 THE USE OF STRUCTURES WHICH ARE EFFECTIVELY OVERLAYED ON          00005700
34 THE INPUT CARD IMAGE BY MAKING THEM BASED VARIABLES.          00005900
35 THE TRAPS ARE SET UP BY THE USE OF THE PL/1 ON CONDITIONS,          00006100
36 AND THE HEADING IS WRITTEN USING A SIMPLE PUT STATEMENT.          00006300
37          THE LOAD PHASE HANDLES EACH OBJECT CARD INDIVIDUALLY AND          00006500
38 THE PROCESSING DEPENDS UPON THE TYPE OF CARDS. EACH          00006700
39 SUCCESSIVE CARD IS READ INTO THE SAME CHARACTER STRING          00006900
40 BUT DIFFERENT TYPES OF CARDS ARE FORMATTED BY THE USE OF          00007100
41 DIFFERENT STRUCTURES. IMMEDIATELY AFTER EACH CARD HAS BEEN          00007300
42 READ INTO CORE A TRANSFER, BASED OF THE CARD TYPE IN COLUMNS          00007500
43 2-4, IS MADE. THE TYPES CHECKED FOR ARE TXT CARDS, ESD CARDS,          00007700
44 RLD CARDS, END CARDS, AND OTHER CARDS. IF THE CARD IS A TXT          00007900
45 CARD THE TEXT IS MOVED IMMEDIATELY TO THE PROGRAM STORAGE          00008100
```

```
%INCLUDE MACRO(LOADESC); %INCLUDE MACRO(DECLARE);
```

MACRO SOURCE2 LISTING

```

46      ARRAY. IF IT IS AN ESD CARD THE APPROPRIATE TABLES ARE 00008300
47      MODIFIED, AND IF IT IS AN RLD CARD THE STORAGE ARRAY IS 00008500
48      MODIFIED AS REQUESTED. OTHER CARDS, INCLUDING LINK-EDIT 00008700
49      CONTROL CARDS, ARE IGNORED. 00008900
50 00009100
51 00009300
52 DESCRIPTION OF INPUT 00009500
53 ----- 00009700
54      THERE ARE TWO TYPES OF ACCEPTABLE INPUT: ANY 00009900
55      LEGITIMATE LINK EDIT INPUT AND / OR A CARD WITH A SLASH 00010100
56      IN COLUMN 1. A SLASH IN CARD COLUMN 1 INDICATES THAT A NEW 00010300
57      JOB FOLLOWS AND IS USED TO TERMINATE A LOAD WHEN IT IS 00010500
58      IGNORED AS ARE UNSUPPORTED TYPES OF ESD CARDS. 00010900
59      THE CARDS WHICH ARE ACCEPTED AND SUPPORTED INCLUDE TEXT 00011100
60      CARDS, RELOCATION DIRECTORY CARDS, EXTERNAL SYMBOL DICTIONARY 00011300
61      CARDS, AND END CARDS. NOT ALL EXTERNAL SYMBOL DICTIONARY 00011500
62      CARDS ARE SUPPORTED HOWEVER, THE ONES THAT ARE NOT SUPPORTED 00011700
63      ENCOUNTERED. LINK EDITOR CONTROL CARDS ARE ACCEPTED BUT 00010700
64      ARE EXTERNAL DUMMY SECTION CARDS. 00011900
65 00012100
66 00012300
67 DESCRIPTION OF OUTPUT 00012500
68 ----- 00012700
69      THERE ARE SEVERAL DISTINCT TYPES OF OUTPUT FROM THIS 00012900
70      LOADER PROGRAM. 00013100
71      DEBUG OUTPUT IS INSTALLED ONLY TO AID IN TESTING LOGIC 00013300
72      OF THE PROGRAM. IT SERVES NO USEFUL PURPOSE TO THE USER BUT 00013500
73      SERVES IN THE INSTALLATION AND MAINTENANCE OF THE LOADER. 00013700
74      MESSAGE OUTPUT IS OF 3 TYPES. INFORMATION MESSAGES ARE 00013900
75      PRINTED IN THE FORM OF IEWXXXXI MESSAGE AND SERVE TO IMPART 00014100
76      INFORMATION TO THE USER. WARNING MESSAGES ARE PRINTED IN THE 00014300
77      FORM OF IEWXXXXW MESSAGE AND SERVE TO FLAG QUESTIONABLE 00014500
78      CONDITIONS, BUT WILL NOT CAUSE TERMINATION OF THE LOAD. 00014700
79      ERROR MESSAGES ARE PRINTED IN THE FORM OF IEWXXXXX MESSAGE. 00014900
80      ERROR MESSAGES CAUSE IMMEDIATE TERMINATION OF THE CURRENT LOAD 00015100
81      AND OCCASIONALLY WILL CAUSE THE LOADER ITSELF TO TERMINATE. 00015300
82      THE LAST TYPE OF OUTPUT IS A MAP OF THE LOADED 00015500
83      PROGRAM. THIS IS PRINTED AT THE END OF EACH LOAD PRIOR TO 00015700
84      TRANSFERING CONTROL TO THE PROBLEM PROGRAM. 00015900
85 */ 00019100
86 00019300

```

```
INCLUDED TEXT FOLLOWS FROM DD.MEMBER = MACRO .DECLARE
```

```

87 DCL CARD CHAR(80) UNALIGNED, 00019500
88 TYPE CHAR(3) UNALIGNED DEF CARD POS(2), 00019700
89 IDSEQ CHAR(8) UNALIGNED DEF CARD POS(73), 00019900
90 SEQNO CHAR(4) DEF CARD POS(77), 00020100

```

91	SEQUENCE CHAR(4) DEF CARD POS(73),	00020300
92	SEQUENCE CHAR(4) STATIC INIT(0),	00020500
93	SEQUENCE CHAR(4) STATIC INIT(' '),	00020700
94	END_OF_JOB CHAR(1) DEF CARD POS(1),	00020900
95	PAGE CHAR(40),	00021100
96	(A,B,C,D,E, P,Q) POINTER,	00021300
97	XED LABEL BASED(A),	00021500
98	CARD_TYPES CHAR(15) STATIC INIT(' TXTESDRLDEND'),	00021700
99	RNT(0:63) FIXED BIN(24,0), /* RENUMBERING TABLE. */	00021900
100	TYP(0:4) LABEL INIT(OTHER,TXT,ESD,RLD,QUIT),	00022100
101	1 STAR UNALIGNED, /* INITIALIZATION TABLE. */	00022300
102	(2 ADDRESS,2 LGTH) FIXED BIN(24,0) INIT(0),	00022500
103	2 ACTUAL_NAME CHAR(8) INIT(' '),	00022700
104	2 ADJUST FIXED BIN(24,0) INIT(0),	00022900
105	1 TABLES(64) /* ADDRESS ASSIGNMENT TABLE */ LIKE STAB,	00023100
106	1 TXT_CARD UNALIGNED BASED(B), /* FORMAT TXT RECORD. */	00023300
107	2 FILLER CHAR(5),	00023500
108	2 ADDRESS BIT(24),	00023700
109	2 SPACE CHAR(2),	00023900
110	2 NUMBER_OF_BYTES BIT(16),	00024100
111	2 ESD_ID_HOLD CHAR(2),	00024300
112	2 ESD_ID BIT(16),	00024500
113	2 TEXT CHAR(56),	00024700
114	/* END OF TEXT FORMATTING. */	00024900
115	1 ESD_CARD UNALIGNED BASED(B), /* FORMAT ESD RECORD */	00025100
116	2 FILLER CHAR(10),	00025300
117	2 NUMBER_OF_BYTES BIT(16),	00025500
118	2 FILLED CHAR(2),	00025700
119	2 ESD_ID BIT(16),	00025900
120	2 ITEMS,	00026100
121	3 ITEM1,	00026300
122	4 ESD_NAME CHAR(8),	00026500
123	4 FLAGS BIT(8),	00026700
124	4 ADDRESS BIT(24),	00026900
125	4 ALIGNMENT BIT(8),	00027100
126	4 LGTH BIT(24),	00027300
127	3 ITEM2 LIKE ITEM1,	00027500
128	3 ITEM3 LIKE ITEM1,	00027700
129	DMEM(2112) FLOAT BIN(53),	00027900
130	CMEM CHAR(16884) /* SET UP MEMORY ARRAYS */ BASED(E),	00028100
131	MEMORY(0:16883) CHAR(1) UNALIGNED BASED(E),	00028300
132	1 RLD_CARD UNALIGNED BASED(B), /* FORMAT RLD RECORD */	00028700
133	2 FILLER CHAR(10),	00028900
134	2 NUMBER_OF_BYTES BIT(8),	00029100
135	2 FILLED CHAR(4),	00029300
136	2 RELOCATION_INFORMATION,	00029500
137	3 R_ESD_ID BIT(16),	00029700
138	3 P_ESD_ID BIT(16),	00029900
139	3 FLAGS,	00030100


```
%INCLUDE MACRO(LUADESC); %INCLUDE MACRO(DECLARE);
```

MACRO SOURCE2 LISTING

```

140          4 T BIT(4),                                00030300
141          4 L BIT(2),                                00030500
142          4 S BIT(1),                                00030700
143          4 X BIT(1),                                00030900
144          3 ADDRESS BIT(24),                          00031100
145          /* END OF RLD FORMATTING. */                00031300
146          1 END_CARD UNALIGNED BASED(B), /* FORMAT LAST RECORD IN SET */ 00031500
147          2 FILLER CHAR(5),                          00031700
148          2 EP_ADDRESS BIT(24),                      00031900
149          2 HOLD CHAR(6),                            00032100
150          2 ID_OF_ENTRY BIT(16),                     00032300
151          2 FILLED CHAR(12),                         00032500
152          2 LENGTH BIT(32),                          00032700
153          /* END OF RECORD FORMATTING */              00032900
154          IHESARC ENTRY(FIXED BIN(15,0)),             00033100
155          RC FIXED BIN(15,0) STATIC INIT(0),          00033300
156          (ZA BASED(P),ZB BASED(Q)) FIXED BIN(31,0);  00033500
157          PSRV(64) CHAR(8),                          00033700
158          PSRVCT FIXED BIN(24,0) INIT(0),             00033900
159          UPTS CHAR(60) STATIC INIT                  00034100
160          ('SIZELETXCALMAPLISTNCALOVLYTESTRENTREUSHIARSCTRNXREFOLDCREFR' 00034300
161          ), /* LINK EDIT AND/OR LOAD OPTIONS. */    00034500
162          SIZE CHAR(4) DEF UPTS ,                    00034700
163          XCAL CHAR(4) DEF UPTS POS(8),               00034900
164          LET CHAR(3) DEF UPTS POS(5),                00035100
165          MAP CHAR(3) DEF UPTS POS(12),               00035300
166          LIST CHAR(4) DEF UPTS POS(15),              00035500
167          NCAL CHAR(4) DEF UPTS POS(19),              00035700
168          OVLY CHAR(4) DEF UPTS POS(23),              00035900
169          TEST CHAR(4) DEF UPTS POS(27),              00036100
170          RENT CHAR(4) DEF UPTS POS(31),              00036300
171          REUS CHAR(4) DEF UPTS POS(35),              00036500
172          HIAR CHAR(4) DEF UPTS POS(39),              00036700
173          SCTR CHAR(4) DEF UPTS POS(43),              00036900
174          NE CHAR(2) DEF UPTS POS(47),                00037100
175          XREF CHAR(4) DEF UPTS POS(49),              00037300
176          OL CHAR(2) DEF UPTS POS(53),                00037500
177          DC CHAR(2) DEF UPTS POS(55),                00037700
178          REFR CHAR(4) DEF UPTS POS(57),              00037900
179          ESDTYPES(0:6) CHAR(8) STATIC INIT          00038100
180          ('SD OK XD','LD','ER','LR','PC','CM','PR'), 00038300
181          ESD_TYPES(8) LABEL INIT(SD,LD,ER,LR,PC,CM,PR,XD), 00038500
182          TYPES_OF_ESD BIT(64) STATIC INIT           00038700
183          ('00000000000000001000000100000001100000100000001010000011001000000038900 00038900
184          000'B                                     00039100
185          ),                                         00039700
186          ESD_TYPE_ARRAY(8) BIT(8) DEF TYPES_OF_ESD POS(1), 00039900
187          II STATIC FIXED BIN(24,0),                00040100
188          1 ESD_ITEM,                               00040300

```

```
%INCLUDE MACRO(LUADESC); %INCLUDE MACRO(DECLARE);
```

```
MACRO SOURCE2 LISTING
```

```

189          2 ESD_NAME CHAR(8),                                00040500
190          2 FLAGS BIT(8),                                    00040700
191          2 ADDRESS BIT(24),                                  00040900
192          2 ALIGNMENT BIT(8),                                00041100
193          2 LENGTH BIT(24),                                   00041300
194          1 LTABLE(64),                                       00041500
195          2 (R,P,A) FIXED BIN(24,0),                          00041700
196          K FIXED BIN(24,0) STATIC INIT(1),                  00041900
197          TCH CHAR(5) VAR,                                     00042100
198          XFLG BIT(1) STATIC,                                  00042300
199          EXECA(2) FIXED BIN(31,0),                           00042500
200          CARDX(80) CHAR(1) DEF CARD,                         00042700
201          UNENDCARD BIT(1) STATIC INIT('1'B),                00042900
202          START_OF_COMMON FIXED BIN(24,0) STATIC INIT(16884), 00043100
203          EXECB BIT(64) UNALIGNED BASED(0);                   00043300
204          DCL (TCT, LOCATION, I, J) FIXED BIN(24,0) STATIC INIT(0); 00043500
205          /*                                                  00043700
206          */                                                  00043900

```

```
INCLUDED TEXT FOLLOWS FROM DD.MEMBER = MACRO .CONDITI
```

```

207          ON ERROR SNAP BEGIN;                                00044300
208          ON ERROR SYSTEM;                                    00044500
209          PUT FILE(SYSPRINT) PAGE EDIT                         00044700
210          ('IEW8000 LOADER TERMINATES DUE TO UNRECOVERABLE ERROR.' 00044900
211          ,TIME) (A,X(2),A);                                  00045100
212          GO TO TERMINATE;                                     00045300
213          END;                                                 00045500
214          A,D=ADDR(EXECA(1)); XEQ=QUIT; B=ADDR(CARD); C=ADDR(CMEM); 00046100
215          RNT=0; P=ADDR(A); Q=ADDR(B); E=ADDR(DMEM(1));       00046300
216          OPEN FILE(SYSIN) RECORD INPUT,                       00046500
217          FILE(SYSPRINT) PAGESIZE(55) LINESIZE(120);          00046700
218          NEXT_JOB:                                           00047100
219          ON CONDITION(FORTRAN) BEGIN;                         00047300
220          /* INSERT AUTOMATIC LIBRARY CALLS HERE. */          00047500
221          RLD_CLEANUP:                                         00047700
222          DO I=1 TO K-1;                                       00047900
223          (SUBRG):                                             00048000
224          MEMORY(TABLES.ADDRESS(LTABLE(I).P+LTABLE(I).A)) 00048100
225          =UNSPEC(ADDR(MEMORY(LTABLE(I).R)));                 00048300
226          END RLD_CLEANUP;                                     00048500
227          PUT FILE(SYSPRINT) PAGE EDIT                         00048700
228          ('IEW8999I NORMAL LOAD COMPLETED.') (A);            00048900
229          DO II=1 TO TCT BY 1;                                  00049000
230          PUT FILE(SYSPRINT) EDIT(TABLES(II))                 00049100
231          (SKIP,2 F(9),X(4),A,F(9));                          00049300
232          END; /* MEMORY MAP. */                              00049400
233          DO II=1 TO TCI BY 1;                                  00049500

```

```
%INCLUDE MACRO(LUADESC); %INCLUDE MACRO(DECLARE);
```

MACRO SOURCE2 LISTING

```

234      IF TABLES.ACTUAL_NAME(II)='MAIN' THEN 00049700
235      DO; 00049900
236      SUBSTR(EXECB,41,24)= UNSPEC( ADDR(MEMORY( 00050100
237      (TABLES.ADDRESS( 00050300
238      II 00050500
239      GO TO XEQ; 00050700
240      END; 00050900
241      END; 00051100
242      END; 00051300
243      ON FINISH GO TO NEXT_JOB; 00051500
244      ON ENDFILE(SYSIN) BEGIN; 00051700
245      PUT FILE(SYSPRINT) PAGE EDIT('LOADER TERMINATED AT', 00051900
246      TIME) (2 A); 00052100
247      GO TO TERMINATE; 00052300
248      END; 00052500
249      DO TCH=OVLY,TEST,RENT,REUS,HIAR,SCTR,DC,REFR; 00052700
250      IF INDEX(PARM,TCH)~=0 THEN PUT FILE(SYSPRINT) EDIT 00052900
251      ('IEW8050W WARNING -- UNSUPPORTED OPTION:',TCH) (SKIP,A,A); 00053100
252      END; 00053300
253      DO I=1 TO 64; 00053500
254      TABLES.ADDRESS(I),TABLES.LNGTH(I),TABLES.ADJUST(I)=0.; 00053700
255      TABLES.ACTUAL_NAME(I)=STAB.ACTUAL_NAME; 00053900
256      END; 00054100
257      SUBSTR(EXECB,41,24)=(24)'0'B; 00054300
258      PUT FILE(SYSPRINT) EDIT(' THE LOADER',DATE) (SKIP,COL(50),A, 00054500
259      COL(100),A); 00054700
260      /* 00054725
261      00054750
262      00054775
263      00054800
264      */ 00054825

```

```
INCLUDED TEXT FOLLOWS FROM DD.MEMBER = MACRO .STARTLP
```

```

265      /* LOAD LOOP */ 00054900
266      DO WHILE('1'B); 00055100
267      READ FILE(SYSIN) INTO(CARD); 00055300
268      IF END_OF_JOB='/' THEN SIGNAL CONDITION(FORTRAN); 00055500
269      IF INDEX(PARM,LIST)~=0 THEN PUT FILE(SYSPRINT) EDIT 00055700
270      ('IEW9999I',TYPE,'CARD PROCESSED. SEQUENCE=',IDSEQ) 00055900
271      (SKIP,A(8),X(1),A(3),X(1),A(26),A(8)); 00056100
272      IF IDSEQ ~= ' ' THEN DO; 00056300
273      IF SEQNO<SEQNOA THEN DO; 00056500
274      IF SEQNAME=SEQNAMEA THEN 00056700
275      PUT FILE(SYSPRINT) EDIT 00056900
276      ('IEW8070W WARNING -- SEQUENCE ERROR.') (SKIP,A) ; 00057100
277      END; 00057300
278      ELSE DO; 00057500

```

```
%INCLUDE MACRO(LUADESC); %INCLUDE MACRO(DECLARE);
```

MACRO SOURCE2 LISTING

```
279          SEQNOA=SEQNU;                                00057700
280          SEQNAMEA=SEQNAME;                             00057900
281          END;                                           00058100
282          END;                                           00058300
283          GO TO TYP(INDEX(CARD_TYPES,TYPE)/3.);         00058500
```

```
INCLUDED TEXT FOLLOWS FROM DD.MEMBER = MACRO .TEXTCRD
```

```
284      TXT:      DO;                                       00058700
285                ON STRG      GO TO ABANDON;              00058900
286      (STRG):                                         00059100
287                SUBSTR(CMEM,TXT_CARD.ADDRESS             00059300
288                + TABLES.ADDRESS(RNT(TXT_CARD.ESD_ID))+1- 00059500
289                TABLES.ADJUST(RNT(TXT_CARD.ESD_ID)),     00059700
290                TXT_CARD.NUMBER_OF_BYTES)=TXT_CARD.TEXT;  00059900
291                GO TO NEXT_CARD;                          00060100
292      END TXT;                                         00060300
293      /*                                              00060500
294      */                                              00060700
```

```
INCLUDED TEXT FOLLOWS FROM DD.MEMBER = MACRO .ESDCARD
```

```
295      ESD:                                             00061100
296      /*                                              00061300
297      */                                              00061500
298      ESD_ITEM=ESD_CARD.ITEMS.ITEM1;                  00061900
299      ESDR:      DO I=17 BY 16 WHILE(I<ESD_CARD.NUMBER_OF_BYTES+16); 00062100
300                IF(ESD_ITEM.ESD_NAME='                ') & (ESD_ITEM.FLAGS='00000101'B) 00062300
301                THEN ESD_ITEM.ESD_NAME='#COM####';      00062500
302                                                        00062600
303                IF ESD_ITEM.FLAGS ^= '00000110'B THEN DO; 00062700
304                  TABLES.ACTUAL_NAME(TCT+1)=ESD_ITEM.ESD_NAME; 00062900
305                  DO J=1 BY 1 WHILE(TABLES.ACTUAL_NAME(J)      00063100
306                  ^= TABLES.ACTUAL_NAME(TCT+1)); END;      00063300
307                  IF J=TCT+1 THEN TCT=TCT+1;                00063500
308                  RNT(ESD_CARD.ESD_ID)=J;                    00064100
309                  END;                                         00064700
310      M=MOD(ESD_ITEM.FLAGS,8);                            00064750
311      IF INDEX(PARM,LIST) ^= 0 THEN PUT FILE(SYSPRINT)      EDIT 00064900
312      ('IEW9995I',ESDTYPES(MOD(ESD_ITEM.FLAGS,8))),         00065100
313      'PROCESSED.  FLAGS=',ESD_ITEM.FLAGS)                 00065300
314      (SKIP,A,X(2),A((SUBSTR(ESDTYPES(M),8,1)='D')*6.0+2),X(2),A,A); 00065400
315      DO II=1 TO 8; IF ESD_ITEM.FLAGS=ESD_TYPE_ARRAY(II) THEN 00065500
316      GO TO ESD_TYPES(II);                                  00065700
317      END;                                                  00065800
318      PUT FILE(SYSPRINT)      EDIT                          00065900
319      ('IEW8055W WARNING -- UNSUPPORTED ESD FLAG:',         00066100
```

```
%INCLUDE MACRO(LUADESC); %INCLUDE MACRO(DECLARE);
```

MACRO SOURCE2 LISTING

```

320      ESD_ITEM.FLAGS) (SKIP,A,X(2),A);          00066300
321      GO TO NXT_PART;                          00066400
322  PK:      PSRV(PSRVCT+1)=ESD_ITEM.ESD_NAME;    00066500
323           DO J=1 BY 1 WHILE(PSRV(J)≠PSRV(PSRVCT+1)); END; 00066700
324           IF J=PSRVCT+1 THEN PSRVCT=PSRVCT+1;  00066900
325           GO TO NXT_PART;                      00067100
326  SD:      DO;                                00067300
327           IF ESD_ITEM.LNGTH=0 THEN ONENDCARD='1'B; 00067500
328           TABLES.ADJUST(J)=ESD_ITEM.ADDRESS;  00067700
329           TABLES.ADDRESS(J)=LOCATION+1;        00067900
330           TABLES.LNGTH(J)=ESD_ITEM.LNGTH;    00068100
331           LOCATION=LOCATION+ESD_ITEM.LNGTH;     00068300
332           /* DOUBLE WORD ALIGNMENT */ LOCATION=(LOCATION+7)/8.; 00068500
333           LOCATION=LOCATION*8.;                 00068700
334           END SD;                             00068900
335      GO TO NXT_PART;                          00069100
336  ER:      DO;                                00069300
337           TABLES(J).ADDRESS,                 00069500
338           TABLES(J).LNGTH,                   00069700
339           TABLES(J).ADJUST=0;                00069900
340           GO TO NXT_PART;                     00070100
341           END ER;                             00070300
342  CM:      DO;                                00070500
343           /* ALLLOCATE COMMON FROM THE BACK OF THE MEMORY ARRAY TO AVOID 00070700
344           CONFLICT WITH LENGTH OF SD'S ON END CARDS. */ 00070900
345           IF TABLES(TCT).ACTUAL_NAME =ESD_ITEM.ESD_NAME 00071100
346           THEN DO;                             00071300
347           TABLES.ADJUST(J)=ESD_ITEM.ADDRESS;  00071500
348           /* DOUBLE WORD ALIGNMENT */ ESD_ITEM.LNGTH=BINARY 00071700
349           (ESD_ITEM.LNGTH+7,15,0)/8.0;         00071900
350           START_OF_COMMON,                     00072100
351           TABLES.ADDRESS(J)=START_OF_COMMON-ESD_ITEM.LNGTH*8; 00072300
352           /* CHECK PROGRAM LENGTH AGAINST MAXIMUM LENGTH */ 00072500
353           IF START_OF_COMMON-LOCATION<0 THEN GO TO ABANDON; 00072700
354           TABLES.LNGTH(J)=ESD_ITEM.LNGTH;     00072900
355           GO TO NXT_PART;                      00073100
356           END CM;                             00073300
357           GO TO NXT_PART;                     00073450
358           END CM;                             00073500
359           GO TO NXT_PART;                     00073700
360           GO TO NXT_PART;                     00073900
361           GO TO NXT_PART;                     00074100
362           GO TO NXT_PART;                     00074300
363           GO TO NXT_PART;                     000744900
364           GO TO NXT_PART;                     00074900
365           GO TO NXT_PART;                     00075100
366           GO TO NXT_PART;                     00075300
367           GO TO NXT_PART;                     00075400
368           GO TO NXT_PART;                     00075500
369           GO TO NXT_PART;                     00075700
370           GO TO NXT_PART;                     00075900
371           GO TO NXT_PART;                     00076100
372           GO TO NXT_PART;                     00076300
373           GO TO NXT_PART;                     00076500
374           GO TO NXT_PART;                     00076700

```

```
%INCLUDE MACRO(LUADESC); %INCLUDE MACRO(DECLARE);
```

MACRO SOURCE2 LISTING

```

369      ) (SKIP,A);
370 LD:    /* ALL LABEL DEFINITIONS SHOULD HAVE BEEN PROCESSED BY NOW */
371 NXT_PART:
372      IF ESD_ITEM.ESD_NAME/= ESD_CARD.ITEMS.ITEM2.ESD_NAME THEN
373          ESD_ITEM=ESD_CARD.ITEMS.ITEM2; ELSE
374          ESD_ITEM=
375          ESD_CARD.ITEMS.ITEM3;
376      END ESDR;
377      GO TO NEXT_CARD;

```

```

00076900
00077100
00077300
00077500
00077700
00077900
00078100
00078300
00078500

```

```
INCLUDED TEXT FOLLOWS FROM DD.MEMBER = MACRO .RLDCARD
```

```

378 RLD:
379     DO;
380     I=25;
381     DO WHILE(I< RLD_CARD.NUMBER_OF_BYTES +25);
382         XFLG=RLD_CARD.RELOCATION_INFORMATION.FLAGS.X;
383         IF TABLES.ADDRESS(RNT(
384             RLD_CARD.RELOCATION_INFORMATION.R_ESD_ID))=0 THEN
385             DO;
386                 LTABLE(K).R=RNT
387                 (RLD_CARD.RELOCATION_INFORMATION.R_ESD_ID);
388                 LTABLE(K).P=RNT
389                 (RLD_CARD.RELOCATION_INFORMATION.P_ESD_ID);
390                 LTABLE(K).A =
391                 RLD_CARD.RELOCATION_INFORMATION.ADDRESS;
392                 K=K+1;
393             END;
394         ELSE
395             MEMORY(TABLES.ADDRESS(RNT(
396                 RLD_CARD.RELOCATION_INFORMATION.P_ESD_ID+
397                 RLD_CARD.RELOCATION_INFORMATION.ADDRESS)))=
398             UNSPEC(ADDR(MEMORY(
399                 TABLES.ADDRESS(RNT(
400                     RLD_CARD.RELOCATION_INFORMATION.R_ESD_ID)))));
401             IF XFLG THEN DO;
402                 SUBSTR(CARD,21,4)=SUBSTR(CARD,I,4);
403                 I=I+4; END;
404             ELSE DO;
405                 SUBSTR(CARD,17,8)=SUBSTR(CARD,I,8);
406                 I=I+8;
407             END;
408         END;
409         GO TO NEXT_CARD;
410     END RLD;

```

```

00078700
00078900
00079100
00079300
00079500
00079700
00079900
00080100
00080300
00080500
00080700
00080900
00081100
00081300
00081500
00081700
00081900
00082100
00082300
00082500
00082700
00082900
00083100
00083300
00083500
00083700
00083900
00084100
00084300
00084500
00084700
00084900
00085100

```

```
INCLUDED TEXT FOLLOWS FROM DD.MEMBER = MACRO .OTHEOUT
```

```
%INCLUDE MACRO(LOADESC); %INCLUDE MACRO(DECLARE);
```

MACRO SOURCE2 LISTING

```

411  OTHER:                                00085300
412      PUT FILE(SYSPRINT) EDIT(          00085500
413      'IEW8065W WARNING -- LKED CONTROL NOT SUPPORTED.')(SKIP,A); 00085700
414      GO TO NEXT_CARD;                  00085900
415  QUIT:                                00086100
416      DO;                              00086300
417      ON SIZE GO TO ABANDON;             00086500
418      IF(END_CARD.ID_OF_ENTRY='0100000001000000'B) | 00086700
419      (END_CARD.ID_OF_ENTRY='0000000000000000'B) 00086900
420      THEN END_CARD.ID_OF_ENTRY='0000000000000001'B; 00087100
421      IF ONENDCARD THEN DO;             00087300
422      IF INDEX(PARM,LIST)~=0 THEN        00087500
423      (SIZE):                            PUT FILE(SYSPRINT) EDIT 00087600
424      ('IEW9998I LENGTH TAKEN FROM END CARD FOR ESD-ID= ', 00087700
425      END_CARD.ID_OF_ENTRY,' LENGTH= ', END_CARD.LENGTH) 00087900
426      (SKIP,A,F(3),A(8),F(8));          00088100
427      (SIZE):                            00088200
428      TABLES.LENGTH(RNT(END_CARD.ID_OF_ENTRY))= 00088300
429      END_CARD.LENGTH;                   00088500
430      (SIZE):                            00088600
431      LOCATION=LOCATION+END_CARD.LENGTH; 00088700
432      ONENDCARD='0'B;                    00088900
433      LOCATION=LOCATION*8.;                00089300
434      END;                               00089500
435      REVERT SIZE;                       00089700
436      END QUIT;                          00089900
437      /* DOUBLE WORD ALIGNMENT */ LOCATION=(LOCATION+7.)/8.; 00088900
438  NEXT_CARD:                            00090100
439      END;                               00090300

```

```
INCLUDED TEXT FOLLOWS FROM DD.MEMBER = MACRO .ABTERMIN
```

```

440  ABANDON:                             00090500
441      DO WHILE(END_OF_JOB~/='/');        00090700
442      READ FILE(SYSIN) INTO(CARD);       00090900
443      END; /* ABOKT LOOP */              00091100
444      START_OF_COMMON=16884;              00091300
445      ONENDCARD='0'B;                     00091500
446      LOCATION=0;                         00091700
447      PUT FILE(SYSPRINT) EDIT(           00091900
448      'IEW8010 PROGRAM TOO LARGE. CURRENT LOAD ABORTED AT',TIME) 00092100
449      (SKIP,A,X(2),A);                   00092200
450      GO TO NEXT_JOB;                     00092300
451  TERMINATE:                             00092500
452      REVERT FINISH;                      00092700
453      STOP;                              00092900
454  END LOADER;                            00093100

```

```
%INCLUDE MACRO(LUADESC); %INCLUDE MACRO(DECLARE);
```

```
NO ERROR OR WARNING CONDITION HAS BEEN DETECTED FOR THIS MACRO PASS.
```



```
*INCLUDE MACRO(LOADESC); *INCLUDE MACRO(DECLARE);
```

SOURCE LISTING.

```
1  /*          FORTRAN LOADER PROGRAM          */
   LOADER: PROC(PARM) OPTIONS(MAIN);
   /*
```

DESCRIPTION OF THE PROGRAM:

```
-----
      THIS PROGRAM IS DESIGNED TO BE USED TO LOAD COMPLETE
      OBJECT PROGRAMS FROM OBJECT CARDS. THERE ARE SOME
      RESTRICTIONS ON THE NATURE OF THE JOBS WHICH WILL BE ACCEPTED
      AND THERE ARE SOME MODIFICATIONS WHICH SHOULD BE MADE TO THE
      PROGRAM TO INCREASE ITS SPEED AND EFFICIENCY. AN ATTEMPT
      HAS BEEN MADE TO HAVE THE PROGRAM TO BE LARGELY
      SELF-DOCUMENTING, PRIMARILY BY THE USE OF JUDICIOUSLY
      CHOSEN VARIABLE NAMES. COMMENTS ARE USED TO OBTAIN THIS
      DOCUMENTATION TO SOME EXTENT BUT THE LARGER PART IS DONE
      VIA VARIABLE NAMES.
```

METHOD OF SOLUTION

```
-----
      THE OVERALL METHOD OF SOLUTION IS PERFORMED IN THREE
      PHASES: 1) INITIALIZATION PHASE, 2) LOAD PHASE,
      AND 3) EXECUTE PHASE.
```

```
      THE INITIALIZATION PHASE CONSISTS: 1) OF DESCRIBING
      THE INPUT FORMATS OF THE VARIOUS TYPES OF OBJECT CARDS;
      2) OF SETTING TRAPS TO HANDLE ERRORS IN A LOAD AND FOR
      TERMINATIONS OF A LOAD; AND 3) OF WRITING A HEADING FOR THE
      OUTPUT FROM THE LOADER. THE INPUT FORMATS ARE DESCRIBED BY
      THE USE OF STRUCTURES WHICH ARE EFFECTIVELY OVERLAYED ON
      THE INPUT CARD IMAGE BY MAKING THEM BASED VARIABLES.
      THE TRAPS ARE SET UP BY THE USE OF THE PL/1 ON CONDITIONS,
      AND THE HEADING IS WRITTEN USING A SIMPLE PUT STATEMENT.
```

```
      THE LOAD PHASE HANDLES EACH OBJECT CARD INDIVIDUALLY AND
      THE PROCESSING DEPENDS UPON THE TYPE OF CARDS. EACH
      SUCCESSIVE CARD IS READ INTO THE SAME CHARACTER STRING
      BUT DIFFERENT TYPES OF CARDS ARE FORMATTED BY THE USE OF
      DIFFERENT STRUCTURES. IMMEDIATELY AFTER EACH CARD HAS BEEN
      READ INTO CORE A TRANSFER, BASED OF THE CARD TYPE IN COLUMNS
      2-4, IS MADE. THE TYPES CHECKED FOR ARE TXT CARDS, ESD CARDS,
      RLD CARDS, END CARDS, AND OTHER CARDS. IF THE CARD IS A TXT
      CARD THE TEXT IS MOVED IMMEDIATELY TO THE PROGRAM STORAGE
      ARRAY. IF IT IS AN ESD CARD THE APPROPRIATE TABLES ARE
      MODIFIED, AND IF IT IS AN RLD CARD THE STORAGE ARRAY IS
      MODIFIED AS REQUESTED. OTHER CARDS, INCLUDING LINK-FIT
      CONTROL CARDS, ARE IGNORED.
```

DESCRIPTION OF INPUT

```
-----
```

```
%INCLUDE MACRO(LUADESC); %INCLUDE MACRO(DECLARE);
```

THERE ARE TWO TYPES OF ACCEPTABLE INPUT: ANY
LEGITIMATE LINK EDIT INPUT AND / OR A CARD WITH A SLASH
IN COLUMN 1. A SLASH IN CARD COLUMN 1 INDICATES THAT A NEW
JOB FOLLOWS AND IS USED TO TERMINATE A LOAD WHEN IT IS
IGNORED AS ARE UNSUPPORTED TYPES OF ESD CARDS.

THE CARDS WHICH ARE ACCEPTED AND SUPPORTED INCLUDE TEXT
CARDS, RELOCATION DIRECTORY CARDS, EXTERNAL SYMBOL DICTIONARY
CARDS, AND END CARDS. NOT ALL EXTERNAL SYMBOL DICTIONARY
CARDS ARE SUPPORTED HOWEVER, THE ONES THAT ARE NOT SUPPORTED
ENCOUNTERED. LINK EDITOR CONTROL CARDS ARE ACCEPTED BUT
ARE EXTERNAL DUMMY SECTION CARDS.

DESCRIPTION OF OUTPUT

THERE ARE SEVERAL DISTINCT TYPES OF OUTPUT FROM THIS
LOADER PROGRAM.

DEBUG OUTPUT IS INSTALLED ONLY TO AID IN TESTING LOGIC
OF THE PROGRAM. IT SERVES NO USEFUL PURPOSE TO THE USER BUT
SERVES IN THE INSTALLATION AND MAINTENANCE OF THE LOADER.

MESSAGE OUTPUT IS OF 3 TYPES. INFORMATION MESSAGES ARE
PRINTED IN THE FORM OF IEWXXXXI MESSAGE AND SERVE TO IMPART
INFORMATION TO THE USER. WARNING MESSAGES ARE PRINTED IN THE
FORM OF IEWXXXXW MESSAGE AND SERVE TO FLAG QUESTIONABLE
CONDITIONS, BUT WILL NOT CAUSE TERMINATION OF THE LOAD.
ERROR MESSAGES ARE PRINTED IN THE FORM OF IEWXXXXE MESSAGE.
ERROR MESSAGES CAUSE IMMEDIATE TERMINATION OF THE CURRENT LOAD
AND OCCASIONALLY WILL CAUSE THE LOADER ITSELF TO TERMINATE.

THE LAST TYPE OF OUTPUT IS A MAP OF THE LOADED
PROGRAM. THIS IS PRINTED AT THE END OF EACH LOAD PRIOR TO
TRANSFERRING CONTROL TO THE PROGRAM.

```
*/
```

```
DCL
```

```
CARD CHAR(80) UNALIGNED,
TYPE CHAR(3) UNALIGNED DEF CARD POS(2),
IDSEQ CHAR(8) UNALIGNED DEF CARD POS(73),
SEQNO CHAR(4) DEF CARD POS(77),
SEQNAME CHAR(4) DEF CARD POS(73),
SEQNOA CHAR(4) STATIC INIT(0),
SEQNAMEA CHAR(4) STATIC INIT(' '),
END_OF_JOB CHAR(1) DEF CARD POS(1),
PARM CHAR(40),
(A,B,C,D,E, P,Q ) POINTER,
XEO LABEL BASED(A),
CARD_TYPES CHAR(15) STATIC INIT(' TXTESORLUEND'),
RNT(0:63) FIXED BIN(24,0), /* RENUMBERING TABLE. */
TYP(0:4) LABEL INIT(OTHER,TXT,ESD,RLO,QUIT),
1 STAB UNALIGNED, /* INITIALIZATION TABLE. */
  (2 ADDRESS,2 LENGTH) FIXED BIN(24,0) INIT(0),
  2 ACTUAL_NAME CHAR(8) INIT(' '),
  2 ADJUST FIXED BIN(24,0) INIT(0),
1 TABLES(64) /* ADDRESS ASSIGNMENT TABLE */ LIKE STAB,
```

```
%INCLUDE MACRO(LUADESC); %INCLUDE MACRO(DECLARE);
```

```

1 TXT_CARD UNALIGNED BASED(8), /* FORMAT TXT RECORD. */ 106
2 FILLER CHAR(5), 107
2 ADDRESS BIT(24), 108
2 SPACE CHAR(2), 109
2 NUMBER_OF_BYTES BIT(16), 110
2 ESD_ID_HOLD CHAR(2), 111
2 ESD_ID BIT(16), 112
2 TEXT CHAR(56), 113
/* END OF TEXT FORMATTING. */ 114
1 ESD_CARD UNALIGNED BASED(8), /* FORMAT ESD RECORD */ 115
2 FILLER CHAR(10), 116
2 NUMBER_OF_BYTES BIT(16), 117
2 FILLED CHAR(2), 118
2 ESD_ID BIT(16), 119
2 ITEMS, 120
3 ITEM1, 121
4 ESD_NAME CHAR(8), 122
4 FLAGS BIT(8), 123
4 ADDRESS BIT(24), 124
4 ALIGNMENT BIT(8), 125
4 LENGTH BIT(24), 126
3 ITEM2 LIKE ITEM1, 127
3 ITEM3 LIKE ITEM1, 128
DMEM(2112) FLOAT BIN(53), 129
CMEM CHAR(16884) /* SET UP MEMORY ARRAYS */ BASED(E), 130
MEMORY(0:16883) CHAR(1) UNALIGNED BASED(E), 131
1 RLD_CARD UNALIGNED BASED(8), /* FORMAT RLD RECORD */ 132
2 FILLER CHAR(10), 133
2 NUMBER_OF_BYTES BIT(8), 134
2 FILLED CHAR(4), 135
2 RELOCATION_INFORMATION, 136
3 R_ESD_ID BIT(16), 137
3 P_ESD_ID BIT(16), 138
3 FLAGS, 139
4 T BIT(4), 140
4 L BIT(2), 141
4 S BIT(1), 142
4 X BIT(1), 143
3 ADDRESS BIT(24), 144
/* END OF RLD FORMATTING. */ 145
1 END_CARD UNALIGNED BASED(8), /* FORMAT LAST RECORD IN SET */ 146
2 FILLER CHAR(5), 147
2 EP_ADDRESS BIT(24), 148
2 HOLD CHAR(6), 149
2 ID_OF_ENTRY BIT(16), 150
2 FILLED CHAR(12), 151
2 LENGTH BIT(32), 152
/* END OF RECORD FORMATTING */ 153
IHESAKC ENTRY(FIXED BIN(15,0)), 154
RC FIXED BIN(15,0) STATIC INIT(0), 155
(ZA BASED(P), ZB BASED(Q)) FIXED BIN(31,0); 156
PSRV(64) CHAR(8), 157

```



```
%INCLUDE MACRO(LOADESC); %INCLUDE MACRO(DECLARE);
```

```

8      209
9      210
10     211
11     212
12     213
13     214
14     215
15     216
16     217
17     218
18     219
19     220
20     221
21     222
22     223
23     224
24     225
25     226
26     227
27     228
28     229
29     230
30     231
31     232
32     233
33     234
34     235
35     236
36     237
37     238
38     239
39     240
40     241
41     242
42     243
43     244
44     245
45     246
46     247
47     248
48     249
49     250
50     251
51     252
52     253
53     254
54     255
55     256
56     257
57     258
58     259
59     260
60     261
61     262
62     263
63     264
64     265
65     266
66     267
67     268
68     269
69     270
70     271
71     272
72     273
73     274
74     275
75     276
76     277
77     278
78     279
79     280
80     281
81     282
82     283
83     284
84     285
85     286
86     287
87     288
88     289
89     290
90     291
91     292
92     293
93     294
94     295
95     296
96     297
97     298
98     299
99     300
100    301
101    302
102    303
103    304
104    305
105    306
106    307
107    308
108    309
109    310
110    311
111    312
112    313
113    314
114    315
115    316
116    317
117    318
118    319
119    320
120    321
121    322
122    323
123    324
124    325
125    326
126    327
127    328
128    329
129    330
130    331
131    332
132    333
133    334
134    335
135    336
136    337
137    338
138    339
139    340
140    341
141    342
142    343
143    344
144    345
145    346
146    347
147    348
148    349
149    350
150    351
151    352
152    353
153    354
154    355
155    356
156    357
157    358
158    359
159    360
160    361
161    362
162    363
163    364
164    365
165    366
166    367
167    368
168    369
169    370
170    371
171    372
172    373
173    374
174    375
175    376
176    377
177    378
178    379
179    380
180    381
181    382
182    383
183    384
184    385
185    386
186    387
187    388
188    389
189    390
190    391
191    392
192    393
193    394
194    395
195    396
196    397
197    398
198    399
199    400
200    401
201    402
202    403
203    404
204    405
205    406
206    407
207    408
208    409
209    410
210    411
211    412
212    413
213    414
214    415
215    416
216    417
217    418
218    419
219    420
220    421
221    422
222    423
223    424
224    425
225    426
226    427
227    428
228    429
229    430
230    431
231    432
232    433
233    434
234    435
235    436
236    437
237    438
238    439
239    440
240    441
241    442
242    443
243    444
244    445
245    446
246    447
247    448
248    449
249    450
250    451
251    452
252    453
253    454
254    455
255    456
256    457
257    458
258    459
259    460
260    461
261    462
262    463
263    464
264    465
265    466
266    467
267    468
268    469
269    470
270    471
271    472
272    473
273    474
274    475
275    476
276    477
277    478
278    479
279    480
280    481
281    482
282    483
283    484
284    485
285    486
286    487
287    488
288    489
289    490
290    491
291    492
292    493
293    494
294    495
295    496
296    497
297    498
298    499
299    500
300    501
301    502
302    503
303    504
304    505
305    506
306    507
307    508
308    509
309    510
310    511
311    512
312    513
313    514
314    515
315    516
316    517
317    518
318    519
319    520
320    521
321    522
322    523
323    524
324    525
325    526
326    527
327    528
328    529
329    530
330    531
331    532
332    533
333    534
334    535
335    536
336    537
337    538
338    539
339    540
340    541
341    542
342    543
343    544
344    545
345    546
346    547
347    548
348    549
349    550
350    551
351    552
352    553
353    554
354    555
355    556
356    557
357    558
358    559
359    560
360    561
361    562
362    563
363    564
364    565
365    566
366    567
367    568
368    569
369    570
370    571
371    572
372    573
373    574
374    575
375    576
376    577
377    578
378    579
379    580
380    581
381    582
382    583
383    584
384    585
385    586
386    587
387    588
388    589
389    590
390    591
391    592
392    593
393    594
394    595
395    596
396    597
397    598
398    599
399    600
400    601
401    602
402    603
403    604
404    605
405    606
406    607
407    608
408    609
409    610
410    611
411    612
412    613
413    614
414    615
415    616
416    617
417    618
418    619
419    620
420    621
421    622
422    623
423    624
424    625
425    626
426    627
427    628
428    629
429    630
430    631
431    632
432    633
433    634
434    635
435    636
436    637
437    638
438    639
439    640
440    641
441    642
442    643
443    644
444    645
445    646
446    647
447    648
448    649
449    650
450    651
451    652
452    653
453    654
454    655
455    656
456    657
457    658
458    659
459    660
460    661
461    662
462    663
463    664
464    665
465    666
466    667
467    668
468    669
469    670
470    671
471    672
472    673
47
```

```
%INCLUDE MACRO(LoadESC); %INCLUDE MACRO(DECLARE);
```

```

260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311

/*
/* LOAD LOOP */
DO WHILE('1'B);
READ FILE(SYSIN) INTO(CARD);
IF END_OF_JOB='/' THEN SIGNAL CONDITION(FORTRAM);
IF INDEX(PARM,LIST)~=0 THEN PUT FILE(SYSPRINT) EDIT
('IEW9999I',TYPE,'CARD PROCESSED. SEQUENCE=',IDSEQ)
(SKIP,A(8),X(1),A(3),X(1),A(26),A(8));
IF IDSEQ ~= ' ' THEN DO;
IF SEQNO<SEQNOA THEN DO;
IF SEQNAME=SEQNAMEA THEN
PUT FILE(SYSPRINT) EDIT
('IEW8070W WARNING -- SEQUENCE ERROR.') (SKIP,A) ;
END;
ELSE DO;
SEQNOA=SEQNO;
SEQNAMEA=SEQNAME;
END;
END;
GO TO TYP(INDEX(CARD_TYPES,TYPE)/3.);
TXT: DO;
ON STNG GO TO ABANDON;
(STRG):
SUBSTR(CMEM,TXT_CARD.ADDRESS
+ TABLES.ADDRESS(RNT(TXT_CARD.ESD_ID))+1-
TABLES.ADJUST(RNT(TXT_CARD.ESD_ID)),
TXT_CARD.NUMBER_OF_BYTES)=TXT_CARD.TEXT;
GO TO NEXT_CARD;
END TXT;
/*
*/
ESD:
/*
*/
ESD_ITEM=ESD_CARD.ITEMS.ITEM1;
DO I=17 BY 16 WHILE(I<ESD_CARD.NUMBER_OF_BYTES+16);
ESDR: IF(ESD_ITEM.ESD_NAME=' ') & (ESD_ITEM.FLAGS='00000101'B)
THEN ESD_ITEM.ESD_NAME='#COM###';
IF ESD_ITEM.FLAGS ~= '00000110'B THEN DO;
TABLES.ACTUAL_NAME(TCT+1)=ESD_ITEM.ESD_NAME;
DO J=1 BY 1 WHILE(TABLES.ACTUAL_NAME(J)
~= TABLES.ACTUAL_NAME(TCT+1)); END;
IF J=TCT+1 THEN TCT=TCT+1;
RNT(ESD_CARD.ESD_ID)=J;
END;
M=MOD(ESD_ITEM.FLAGS,8);
IF INDEX(PARM,LIST)~=0 THEN PUT FILE(SYSPRINT) EDIT

```

```
#INCLUDE MACRO(LoadESC); *INCLUDE MACRO(DECLARE);
```

```

001      (*IEW9995I*,ESDTYPES(MOD(ESD_ITEM.FLAGS,8)),
002      *PROCESSED.  FLAGS=*,ESD_ITEM.FLAGS)
003      (SKIP,A,X(2),A((SUBSTR(ESDTYPES(M),8,1)='D')*6,0+2),X(2),A,A);
004      DO 11=1 TO 8; IF ESD_ITEM.FLAGS=ESD_TYPE_ARRAY(11) THEN
005      GO TO ESD_TYPES(11);
006      END;
007      PUT FILE(SYSPRINT)      EDIT
008      (*IEW8055W WARNING -- UNSUPPORTED ESD FLAG:*,
009      ESD_ITEM.FLAGS) (SKIP,A,X(2),A);
010      GO TO NXT_PART;
011      PSRV(PSRVCT+1)=ESD_ITEM.ESD_NAME;
012      DO J=1 BY 1 WHILE(PSRV(J)~=PSRV(PSRVCT+1)); END;
013      IF J=PSRVCT+1 THEN PSRVCT=PSRVCT+1;
014      GO TO NXT_PART;
015      SO:
016      DO;
017      IF ESD_ITEM.LENGTH=0 THEN UNENDCARD='1';
018      TABLES.ADJUST(J)=ESD_ITEM.ADDRESS;
019      TABLES.ADDRESS(J)=LOCATION+1;
020      TABLES.LENGTH(J)=ESD_ITEM.LENGTH;
021      LOCATION=LOCATION+ESD_ITEM.LENGTH;
022      /* DOUBLE WORD ALIGNMENT */ LOCATION=(LOCATION+7)/8.;
023      LOCATION=LOCATION*8.;
024      END SO;
025      GO TO NXT_PART;
026      EK:
027      DO;
028      TABLES(J).ADDRESS,
029      TABLES(J).LENGTH,
030      TABLES(J).ADJUST=0;
031      GO TO NXT_PART;
032      END EK;
033      CM:
034      DO;
035      /* ALLOCATE COMMON FROM THE BACK OF THE MEMORY ARRAY TO AVOID
036      CONFLICT WITH LENGTH OF SO'S ON END CARDS. */
037      IF TABLES(TCT).ACTUAL_NAME =ESD_ITEM.ESD_NAME
038      THEN DO;
039      TABLES.ADJUST(J)=ESD_ITEM.ADDRESS;
040      /* DOUBLE WORD ALIGNMENT */ ESD_ITEM.LENGTH=BINARY
041      (ESD_ITEM.LENGTH+7,15,0)/8.0;
042      START_OF_COMMON,
043      TABLES.ADDRESS(J)=START_OF_COMMON-ESD_ITEM.LENGTH*8;
044      /* CHECK PROGRAM LENGTH AGAINST MAXIMUM LENGTH */
045      IF START_OF_COMMON-LOCATION<0 THEN GO TO ABANDON;
046      TABLES.LENGTH(J)=ESD_ITEM.LENGTH;
047      GO TO NXT_PART;
048      END CM;
049      GO TO NXT_PART;
050      LR:

```

```
*INCLUDE MACRO(LUADESC); *INCLUDE MACRO(DECLARE);
```

```

136      PC:      ESD_ITEM.ESD_NAME, TABLES(J).ACTUAL_NAME='MAIN';
137      GO TO NXT_PART;
138      XD:      PUT FILE(SYSPRINT) EDIT('IEW8060 WARNING -- DUMMY SECTIONS NOT
139      SUPPORTED'
140      ) (SKIP,A);
141      LD:      /* ALL LABEL DEFINITIONS SHOULD HAVE BEEN PROCESSED BY NOW */
142      NXT_PART:
143      IF ESD_ITEM.ESD_NAME= ESD_CARD.ITEMS.ITEM2.ESD_NAME THEN
144      ESD_ITEM=ESD_CARD.ITEMS.ITEM2; ELSE
145      ESD_ITEM=
146      ESD_CARD.ITEMS.ITEM3;
147      END ESDR;
148      GO TO NEXT_CARD;
149      RLD:
150      DO;
151      I=25;
152      DO WHILE(I< RLD_CARD.NUMBER_OF_BYTES +25);
153      XFLG=RLD_CARD.RELOCATION_INFORMATION.FLAGS.X;
154      IF TABLES.ADDRESS(RNT(
155      RLD_CARD.RELOCATION_INFORMATION.R_ESD_ID))=0 THEN
156      DO;
157      LTABLE(K).R=RNT
158      (RLD_CARD.RELOCATION_INFORMATION.R_ESD_ID);
159      LTABLE(K).P=RNT
160      (RLD_CARD.RELOCATION_INFORMATION.P_ESD_ID);
161      LTABLE(K).A =
162      RLD_CARD.RELOCATION_INFORMATION.ADDRESS;
163      K=K+1;
164      END;
165      ELSE
166      MEMORY(TABLES.ADDRESS(RNT(
167      RLD_CARD.RELOCATION_INFORMATION.P_ESD_ID+
168      RLD_CARD.RELOCATION_INFORMATION.ADDRESS)))=
169      UNSPEC(ADDR(MEMORY(
170      TABLES.ADDRESS(RNT(
171      RLD_CARD.RELOCATION_INFORMATION.R_ESD_ID)))));
172      IF XFLG THEN DO;
173      SUBSTR(CARD,21,4)=SUBSTR(CARD,1,4);
174      I=I+4; END;
175      ELSE DO;
176      SUBSTR(CARD,17,8)=SUBSTR(CARD,1,8);
177      I=I+8;
178      END;
179      END;
180      GO TO NEXT_CARD;
181      END RLD;
182      OTHER:
183      PUT FILE(SYSPRINT) EDIT(
184      'IEW8065H WARNING -- LKED CONTROL NOT SUPPORTED.')(SKIP,A);

```



```
*INCLUDE MACRO(LUAFESC); *INCLUDE MACRO(DECLARE);
```

```

168      GO TO NEXT_CARD; 414
169      QUIT: 415
          DU; 416
170      ON SIZE GO TO ABANDON; 417
172      IF(END_CARD.ID_OF_ENTRY='0100000001000000'B) 418
          (END_CARD.ID_OF_ENTRY='0000000000000000'B) 419
173      THEN END_CARD.ID_OF_ENTRY='0000000000000001'B; 420
174      IF UNENDCARD THEN DU; 421
176      IF INDEX(PARM,LIST)~=0 THEN 422
177      (SIZE): PUT FILE(SYSPRINT) EDIT 423
          ('IEW99981 LENGTH TAKEN FROM END CARD FOR ESD-ID= ', 424
          END_CARD.ID_OF_ENTRY,' LENGTH= ', END_CARD.LENGTH) 425
          (SKIP,A,F(3),A(8),F(8)); 426
          (SIZE): 427
178      TABLES.LENGTH(RNT(END_CARD.ID_OF_ENTRY))= 428
          END_CARD.LENGTH; 429
          (SIZE): 430
          LOCATION=LOCATION+END_CARD.LENGTH; 431
          UNENDCARD='0'B; 432
          LOCATION=LOCATION*8.; 433
182      END; 434
183      REVERT SIZE; 435
184      END QUIT; 436
185      /* DOUBLE WORD ALIGNMENT */ LOCATION=(LOCATION+7.)/8.; 437
186      NEXT_CARD: 438
          END; 439
187      ABANDON: 440
          DO WHILE(END_OF_JOB~/='/'); 441
188      READ FILE(SYSIN) INTO(CARD); 442
189      END; /* ABORT LOOP */ 443
190      START_OF_COMMON=16884; 444
191      UNENDCARD='0'B; 445
192      LOCATION=0; 446
193      PUT FILE(SYSPRINT) EDIT( 447
          'IEB010 PROGRAM TOO LARGE. CURRENT LOAD ABORTED AT',TIME) 448
          (SKIP,A,X(2),A); 449
          GO TO NEXT_JOB; 450
194      TERMINATE: 451
          REVERT FINISH; 452
196      STOP; 453
197      END LOADER; 454

```

%INCLUDE MACRO(LORDDESC); %INCLUDE MACRO(DECLARE);

ATTRIBUTE AND CROSS-REFERENCE TABLE

DCL NO.	IDENTIFIER	ATTRIBUTES AND REFERENCES
2	A	AUTOMATIC,ALIGNED,POINTER 11,12,16,33
3	A	IN LTABLE(64),AUTOMATIC,ALIGNED,BINARY,FIXED(24,0) 23,151
187	ABANDON	STATEMENT LABEL CONSTANT 75,129,171
2	ACTUAL_NAME	IN STAB,AUTOMATIC,UNALIGNED,INITIAL,STRING(8),CHARACTER 50
2	ACTUAL_NAME	IN TABLES(64),AUTOMATIC,UNALIGNED,INITIAL,STRING(8),CHARACTER 30,50,85,86,86,123,135
	ADDR	GENERIC,BUILT-IN FUNCTION 11,13,14,16,17,18,23,32,154
2	ADDRESS	IN TXT_CARD,BASED(B),UNALIGNED,STRING(24),BIT 76
2	ADDRESS	IN ITEM3 IN ITEMS IN ESD_CARD,BASED(B),UNALIGNED,STRING(24),BIT
2	ADDRESS	IN ITEM2 IN ITEMS IN ESD_CARD,BASED(B),UNALIGNED,STRING(24),BIT
2	ADDRESS	IN ITEM1 IN ITEMS IN ESD_CARD,BASED(B),UNALIGNED,STRING(24),BIT
2	ADDRESS	IN RELOCATION_INFORMATION IN RLD_CARD,BASED(B),UNALIGNED,STRING (24),BIT 151,154
2	ADDRESS	IN STAB,AUTOMATIC,UNALIGNED,INITIAL,BINARY,FIXED(24,0)
2	ADDRESS	IN TABLES(64),AUTOMATIC,ALIGNED,INITIAL,BINARY,FIXED(24,0) 23,32,49,76,111,119,127,147,154,154
3	ADDRESS	IN ESD_ITEM,AUTOMATIC,UNALIGNED,STRING(24),BIT 110,125
2	ADJUST	IN STAB,AUTOMATIC,UNALIGNED,INITIAL,BINARY,FIXED(24,0)
2	ADJUST	IN TABLES(64),AUTOMATIC,ALIGNED,INITIAL,BINARY,FIXED(24,0) 49,76,110,119,125
2	ALIGNMENT	IN ITEM5 IN ITEMS IN ESD_CARD,BASED(B),UNALIGNED,STRING(8),BIT

```
#INCLUDE BACKR(LOADESC); #INCLUDE BACKR(DECLARE);
```

OCL NO.	IDENTIFIER	ATTRIBUTES AND REFERENCES
2	ALIGNMENT	IN ITEM2 IN ITEMS IN ESD_CARD,BASED(6),UNALIGNED,STRING(8),BIT
2	ALIGNMENT	IN ITEM1 IN ITEMS IN ESD_CARD,BASED(6),UNALIGNED,STRING(8),BIT
3	ALIGNMENT	IN ESD_ITEM,AUTOMATIC,UNALIGNED,STRING(8),BIT
2	B	AUTOMATIC,ALIGNED,POINTER 13,17,76,76,76,76,76,79,80,90,138,139,140,145,146,147,149,150,151 154,154,154,172,172,173,177,177,178,178,179
	BINARY	GENERIC,BUILT-IN FUNCTION 126
2	C	AUTOMATIC,ALIGNED,POINTER 14
2	CARD	AUTOMATIC,UNALIGNED,STRING(80),CHARACTER 13,55,157,157,161,161,188
2	CARD_TYPES	STATIC,UNALIGNED,INITIAL,STRING(15),CHARACTER 72
3	CARDX	(80)AUTOMATIC,DEFINED,UNALIGNED,STRING(1),CHARACTER
122	CM	STATEMENT LABEL CONSTANT
2	CMEM	BASED(6),UNALIGNED,STRING(16884),CHARACTER 14,76
2	D	AUTOMATIC,ALIGNED,POINTER 11,32,52
	DATE	BUILT-IN FUNCTION 53
3	DC	AUTOMATIC,DEFINED,UNALIGNED,STRING(2),CHARACTER 44
2	DMEM	(2112)AUTOMATIC,ALIGNED,BINARY,FLOAT(DOUBLE) 18
2	E	AUTOMATIC,ALIGNED,POINTER 14,18,23,23,32,76,154,154
2	END_CARD	BASED(6),STRUCTURE
2	END_OF_JOB	AUTOMATIC,DEFINED,UNALIGNED,STRING(1),CHARACTER 56,187

#INCLUDE <stdio.h>(LDEFESD); #INCLUDE <stdio.h>(DECLARE);

DCL NO.	IDENTIFIER	ATTRIBUTES AND REFERENCES
2	EP_ADDRESS	IN END_CARD, BASED(B), UNALIGNED, STRING(24), BIT
128	ER	STATEMENT LABEL CONSTANT
79	ESD	STATEMENT LABEL CONSTANT
2	ESD_CARD	BASED(B), STRUCTURE
2	ESD_ID	IN TXT_CARD, BASED(B), UNALIGNED, STRING(16), BIT 76, 76
2	ESD_ID	IN ESD_CARD, BASED(B), UNALIGNED, STRING(16), BIT 90
2	ESD_ID_HOLD	IN TXT_CARD, BASED(B), UNALIGNED, STRING(2), CHARACTER
3	ESD_ITEM	AUTOMATIC, STRUCTURE 79, 139, 140
2	ESD_NAME	IN ITEM3 IN ITEMS IN ESD_CARD, BASED(B), UNALIGNED, STRING(8), CHARACTER
2	ESD_NAME	IN ITEM2 IN ITEMS IN ESD_CARD, BASED(B), UNALIGNED, STRING(8), CHARACTER 138
2	ESD_NAME	IN ITEM1 IN ITEMS IN ESD_CARD, BASED(B), UNALIGNED, STRING(8), CHARACTER
3	ESD_NAME	IN ESD_ITEM, AUTOMATIC, UNALIGNED, STRING(8), CHARACTER 81, 82, 85, 101, 123, 135, 138
3	ESD_TYPE_ARRAY	(8) AUTOMATIC, DEFINED, UNALIGNED, STRING(8), BIT 96
3	ESD_TYPES	(8) AUTOMATIC, INITIAL, LABEL 97
80	ESDR	STATEMENT LABEL CONSTANT
3	ESDTYPES	(0:6) STATIC, UNALIGNED, INITIAL, STRING(8), CHARACTER 94, 94
3	EXECA	(2) AUTOMATIC, ALIGNED, BINARY, FIXED(31, 0) 11
3	EXECB	BASED(B), UNALIGNED, STRING(64), BIT 32, 52
2	FILLED	IN ESD_CARD, BASED(B), UNALIGNED, STRING(12), CHARACTER
2	FILLED	IN RLD_CARD, BASED(B), UNALIGNED, STRING(4), CHARACTER

```
%INCLUDE BACKU(LOPDESC); %INCLUDE BACKU(DECLARE);
```

DCL NO.	IDENTIFIER	ATTRIBUTES AND REFERENCES
2	FILLED	IN END_CARD, BASED(B), UNALIGNED, STRING(12), CHARACTER
2	FILLER	IN TXT_CARD, BASED(B), UNALIGNED, STRING(5), CHARACTER
2	FILLER	IN ESD_CARD, BASED(B), UNALIGNED, STRING(10), CHARACTER
2	FILLER	IN RLD_CARD, BASED(B), UNALIGNED, STRING(10), CHARACTER
2	FILLER	IN END_CARD, BASED(B), UNALIGNED, STRING(5), CHARACTER
2	FLAGS	IN ITEM3 IN ITEMS IN ESD_CARD, BASED(B), UNALIGNED, STRING(8), BIT
2	FLAGS	IN ITEM2 IN ITEMS IN ESD_CARD, BASED(B), UNALIGNED, STRING(8), BIT
2	FLAGS	IN ITEM1 IN ITEMS IN ESD_CARD, BASED(B), UNALIGNED, STRING(8), BIT
2	FLAGS	IN RELOCATION_INFORMATION IN RLD_CARD, BASED(B), STRUCTURE
3	FLAGS	IN ESD_ITEM, AUTOMATIC, UNALIGNED, STRING(8), BIT 81, 83, 92, 94, 96, 99
	FORTRAIN	ON CONDITION CONDITION 20, 57
3	HIAR	AUTOMATIC, DEFINED, UNALIGNED, STRING(4), CHARACTER 44
2	HOLD	IN END_CARD, BASED(B), UNALIGNED, STRING(6), CHARACTER
4	I	STATIC, ALIGNED, INITIAL, BINARY, FIXED(24, 0) 22, 23, 23, 23, 48, 49, 49, 49, 50, 80, 80, 144, 145, 157, 158, 158, 161, 162, 162
2	ID_OF_ENTRY	IN END_CARD, BASED(B), UNALIGNED, STRING(16), BIT 172, 172, 173, 177, 178
2	IDSEQ	AUTOMATIC, DEFINED, UNALIGNED, STRING(8), CHARACTER 59, 60
2	IHSARC	EXTERNAL, ENTRY, BINARY, FIXED(15, 0)
3	II	STATIC, ALIGNED, BINARY, FIXED(24, 0) 26, 27, 29, 30, 32, 95, 96, 97
	INDEX	GENERIC, BUILT-IN FUNCTION 45, 58, 72, 93, 176
2	ITEM1	IN ITEMS IN ESD_CARD, BASED(B), STRUCTURE 79

#INCLUDE MACRO(LDADESC); #INCLUDE MACRO(DECLARE);

DCL NO.	IDENTIFIER	ATTRIBUTES AND REFERENCES
2	ITEM2	IN ITEMS IN ESD_CARD, BASED(8), STRUCTURE 139
2	ITEM3	IN ITEMS IN ESD_CARD, BASED(8), STRUCTURE 140
2	ITEMS	IN ESD_CARD, BASED(8), STRUCTURE
4	J	STATIC, ALIGNED, INITIAL, BINARY, FIXED(24,0) 86,86,88,90,102,102,104,110,111,112,119,119,119,125,127,130,135
3	K	STATIC, ALIGNED, INITIAL, BINARY, FIXED(24,0) 22,149,150,151,152,152
2	L	IN FLAGS IN RELOCATION_INFORMATION IN RLD_CARD, BASED(8), UNALIGNED, STRING(2), BIT
138	LD	STATEMENT LABEL CONSTANT
3	LET	AUTOMATIC, DEFINED, UNALIGNED, STRING(3), CHARACTER
3	LIST	AUTOMATIC, DEFINED, UNALIGNED, STRING(4), CHARACTER 58,93,176
2	LNTH	IN ITEM3 IN ITEMS IN ESD_CARD, BASED(8), UNALIGNED, STRING(24), BIT
2	LNTH	IN ITEM2 IN ITEMS IN ESD_CARD, BASED(8), UNALIGNED, STRING(24), BIT
2	LNTH	IN ITEM1 IN ITEMS IN ESD_CARD, BASED(8), UNALIGNED, STRING(24), BIT
2	LNTH	IN END_CARD, BASED(8), UNALIGNED, STRING(32), BIT 177,178,179
2	LNTH	IN STAB, AUTOMATIC, UNALIGNED, INITIAL, BINARY, FIXED(24,0)
2	LNTH	IN TABLES(64), AUTOMATIC, ALIGNED, INITIAL, BINARY, FIXED(24,0) 49,112,119,130,178
3	LNTH	IN ESD_ITEM, AUTOMATIC, UNALIGNED, STRING(24), BIT 108,112,113,126,126,127,130
1	LOADER	ENTRY, BINARY, FIXED(15,0)
4	LOCATION	STATIC, ALIGNED, INITIAL, BINARY, FIXED(24,0) 111,113,113,114,114,115,115,128,179,179,181,181,185,185,192
135	LR	STATEMENT LABEL CONSTANT

```
#INCLUDE PACK0(LOADESC); #INCLUDE PACK0(DECLARE);
```

DCL NO.	IDENTIFIER	ATTRIBUTES AND REFERENCES
3	LTABLE	(64)AUTOMATIC,STRUCTURE
	M	AUTOMATIC,ALIGNED,BINARY,FIXED(15,0) 92,94
3	MAP	AUTOMATIC,DEFINED,UNALIGNED,STRING(3),CHARACTER
2	MEMORY	(0:16883)BASED(E),UNALIGNED,STRING(1),CHARACTER 23,23,32,154,154
	MOD	GENERIC,BUILT-IN FUNCTION 92,94
3	NCAL	AUTOMATIC,DEFINED,UNALIGNED,STRING(4),CHARACTER
3	NE	AUTOMATIC,DEFINED,UNALIGNED,STRING(2),CHARACTER
186	NEXT_CARD	STATEMENT LABEL CONSTANT 77,142,165,168
20	NEXT_JCB	STATEMENT LABEL CONSTANT 38,194
2	NUMBER_OF_BYTES	IN TXT_CARD,BASED(B),UNALIGNED,STRING(16),BIT 76
2	NUMBER_OF_BYTES	IN ESD_CARD,BASED(B),UNALIGNED,STRING(16),BIT 80
2	NUMBER_OF_BYTES	IN RLD_CARD,BASED(B),UNALIGNED,STRING(8),BIT 145
138	NXT_PART	STATEMENT LABEL CONSTANT 100,106,117,120,131,134,136
3	OL	AUTOMATIC,DEFINED,UNALIGNED,STRING(2),CHARACTER
3	UNENDCARD	STATIC,UNALIGNED,INITIAL,STRING(1),BIT 109,174,180,191
3	UPTS	STATIC,UNALIGNED,INITIAL,STRING(60),CHARACTER
167	UTHER	STATEMENT LABEL CONSTANT
3	OVLY	AUTOMATIC,DEFINED,UNALIGNED,STRING(4),CHARACTER 44
2	P	AUTOMATIC,ALIGNED,POINTER 16

```
%INCLUDE MACRO(LOADESC); %INCLUDE MACRO(DECLARE);
```

DECL NO.	IDENTIFIER	ATTRIBUTES AND REFERENCES
3	P	IN LTABLE(64),AUTOMATIC,ALIGNED,BINARY,FIXED(24,0) 23,150
2	P_ESD_ID	IN RELOCATION_INFORMATION IN RLD_CARD,BASED(B),UNALIGNED,STRING (16),BIT 150,154
2	PARAM	PARAMETER,UNALIGNED,STRING(40),CHARACTER 1,45,58,93,176
135	PC	STATEMENT LABEL CONSTANT
101	PR	STATEMENT LABEL CONSTANT
3	PSRV	(64)AUTOMATIC,UNALIGNED,STRING(8),CHARACTER 101,102,102
3	PSRVCT	AUTOMATIC,ALIGNED,INITIAL,BINARY,FIXED(24,0) 101,102,104,105,105
2	Q	AUTOMATIC,ALIGNED,POINTER 17
169	QUIT	STATEMENT LABEL CONSTANT 12
3	R	IN LTABLE(64),AUTOMATIC,ALIGNED,BINARY,FIXED(24,0) 23,149
2	R_ESD_ID	IN RELOCATION_INFORMATION IN RLD_CARD,BASED(B),UNALIGNED,STRING (16),BIT 147,149,154
2	RC	STATIC,ALIGNED,INITIAL,BINARY,FIXED(15,0)
3	REFR	AUTOMATIC,DEFINED,UNALIGNED,STRING(4),CHARACTER 44
2	RELOCATION_INFORMATION	IN RLD_CARD,BASED(B),STRUCTURE
3	RENT	AUTOMATIC,DEFINED,UNALIGNED,STRING(4),CHARACTER 44
3	REUS	AUTOMATIC,DEFINED,UNALIGNED,STRING(4),CHARACTER 44
143	RLD	STATEMENT LABEL CONSTANT


```
#INCLUDE BACK0(LOADESC); #INCLUDE BACK0(DECLARE);
```

DCL NO.	IDENTIFIER	ATTRIBUTES AND REFERENCES
2	RLD_CARD	BASED(6), STRUCTURE
22	RLD_CLEANUP	STATEMENT LABEL CONSTANT
2	RNT	(0:63)AUTOMATIC,ALIGNED,BINARY,FIXED(24,0) 15,76,76,90,147,149,150,154,154,178
2	S	IN_FLAGS IN RELOCATION_INFORMATION IN RLD_CARD,BASED(6),UNALIGNED, STRING(1),BIT
3	SCTR	AUTOMATIC,DEFINED,UNALIGNED,STRING(4),CHARACTER 44
107	SD	STATEMENT LABEL CONSTANT
2	SEQNAME	AUTOMATIC,DEFINED,UNALIGNED,STRING(4),CHARACTER 64,69
2	SEQNAMEA	STATIC,UNALIGNED,INITIAL,STRING(4),CHARACTER 64,69
2	SEQNO	AUTOMATIC,DEFINED,UNALIGNED,STRING(4),CHARACTER 62,68
2	SEQNOA	STATIC,UNALIGNED,INITIAL,STRING(4),CHARACTER 62,68
3	SIZE	AUTOMATIC,DEFINED,UNALIGNED,STRING(4),CHARACTER
2	SPACE	IN_TXT_CARD,BASED(6),UNALIGNED,STRING(2),CHARACTER
2	STAB	AUTOMATIC,STRUCTURE
3	START_OF_COMMON	STATIC,ALIGNED,INITIAL,BINARY,FIXED(24,0) 127,127,128,190
	SUBSTR	GENERIC,BUILT-IN FUNCTION 32,52,76,94,157,157,161,161
	SYSIN	FILE,EXTERNAL 19,39,55,188
	SYSPRINT	FILE,EXTERNAL 8,19,25,27,41,46,53,59,65,94,99,137,167,177,193
2	T	IN_FLAGS IN RELOCATION_INFORMATION IN RLD_CARD,BASED(6),UNALIGNED, STRING(4),BIT
2	TABLES	(64)AUTOMATIC,STRUCTURE

```
#INCLUDE BACK0(LUNDESC); #INCLUDE BACK0(DECLARE);
```

DECL NO.	IDENTIFIER	ATTRIBUTES AND REFERENCES
		27
3	TCH	AUTOMATIC, UNALIGNED, STRING(5), CHARACTER, VARYING 44, 45, 46
4	TCT	STATIC, ALIGNED, INITIAL, PRIMARY, FIXED(24,0) 26, 29, 85, 86, 88, 89, 89, 123
195	TERMINATE	STATEMENT LABEL CONSTANT 9, 42
3	TEST	AUTOMATIC, DEFINED, UNALIGNED, STRING(4), CHARACTER 44
2	TEXT	IN TXT_CARD, BASED(6), UNALIGNED, STRING(56), CHARACTER 76
	TIME	BUILT-IN FUNCTION 8, 41, 193
73	TXT	STATEMENT LABEL CONSTANT
2	TXT_CARD	BASED(6), STRUCTURE
2	TP	(0:4)AUTOMATIC, INITIAL, LABEL 72
2	TYPE	AUTOMATIC, DEFINED, UNALIGNED, STRING(3), CHARACTER 59, 72
3	TYPES_OF_ESD	STATIC, UNALIGNED, INITIAL, STRING(64), BIT
	UNSPEC	GENERIC, BUILT-IN FUNCTION 23, 32, 154
2	X	IN FLAGS IN RELOCATION_INFORMATION IN RLD_CARD, BASED(8), UNALIGNED, STRING(1), BIT 146
3	XCAL	AUTOMATIC, DEFINED, UNALIGNED, STRING(4), CHARACTER
137	XD	STATEMENT LABEL CONSTANT
2	XEQ	BASED(A), LABEL 12, 33
3	XFLG	STATIC, UNALIGNED, STRING(1), BIT 146, 155

%INCLUDE MACRO(LOADPSC); %INCLUDE MACRO(DECLARE);

DECL NO.	IDENTIFIER	ATTRIBUTES AND REFERENCES
3	XREF	AUTOMATIC, DEFINED, UNALIGNED, STRING(4), CHARACTER
2	ZA	BASED(P), ALIGNED, BINARY, FIXED(31,0)
2	ZB	BASED(0), ALIGNED, BINARY, FIXED(31,0)

%INCLUDE MACRO(LOADESC); %INCLUDE MACRO(DECLARE);

AGGREGATE LENGTH TABLE

STATEMENT NO.	IDENTIFIER	LENGTH IN BYTES
3	CARDX	80
2	DMEM	16896
2	END_CARD	32
2	ESD_CARD	64
3	ESD_ITEM	16
3	ESD_TYPE_ARRAY	8
3	ESD_TYPES	64
3	ESDTYPES	56
3	EXECA	8
3	LTABLE	768
2	MEMORY	16884
3	PSRV	512
2	RLD_CARD	23
2	RNT	256
2	STAB	20
2	TABLES	1280
2	TXT_CARD	72
2	TYP	40

INCLUDE MACRO(LOADFSC); %INCLUDE MACRO(DECLARE);

STORAGE REQUIREMENTS.

THE STORAGE AREA FOR THE PROCEDURE LABELLED LOADER IS 20772 BYTES LONG.

THE STORAGE AREA FOR THE UN UNIT AT STATEMENT NO. 6 IS 364 BYTES LONG.

THE STORAGE AREA FOR THE UN UNIT AT STATEMENT NO. 21 IS 468 BYTES LONG.

THE STORAGE AREA FOR THE UN UNIT AT STATEMENT NO. 37 IS 296 BYTES LONG.

THE STORAGE AREA FOR THE UN UNIT AT STATEMENT NO. 40 IS 352 BYTES LONG.

THE STORAGE AREA FOR THE UN UNIT AT STATEMENT NO. 74 IS 296 BYTES LONG.

THE STORAGE AREA FOR THE UN UNIT AT STATEMENT NO. 170 IS 296 BYTES LONG.

THE PROGRAM CSECT IS NAMED LOADER AND IS 10456 BYTES LONG.

THE STATIC CSECT IS NAMED *LOADERA AND IS 2363 BYTES LONG.