LY20-8058-0

Program Product

Offline IBM 3800 Utility Logic

Program Number 5748–UT2

IBM

First Edition (November 1977)

This is a new publication describing the program product Offline IBM 3800 Utility which will be available under OS/VS1 Release 6, OS/VS Release 1.7 (SVS), OS/VS Release 3.7 (MVS), OS/MVT Release 21.8, and DOS/VS Release 34.

Information in this publication is subject to significant change. Any such changes will be published in new editions or technical newsletters. Before using the publication, consult the latest *IBM System/370 Bibliography,* GC20-0001, and the technical newsletters that amend the bibliography, to learn which editions and technical newsletters are applicable and current.

Requests for copies of IBM publications should be made to the IBM branch office that serves you.

Forms for readers' comments are provided at the back of the publication. If the forms have been removed, comments may be addressed to IBM Corporation, P. O. Box 50020, Programming Publishing, San Jose, California 95150. All comments and suggestions become the property of IBM.

CONTENTS

Figures

ABOUT THIS MANUAL

This manual explains the logic of the utility program which creates tape files that control the Offline 3800 Printer.

The purpose of this manual is to help the IBM programming service representative (PSR) to isolate a programming failure as quickly as possible. Although this manual is written primarily for the PSR, system programmers and others may also be interested in some of the information presented here.

Organization of The Manual

- Front matter—preface, table of contents, list of figures
- Introduction—a conceptual overview
- Method of Operation—a functional overview
- Program Organization—a list of procedures and subprocedures
- Data Areas—examples of some major data areas
- Diagnostic aids—RETAIN keywords explained and defined
- Index

How to use The Manual

First study the table of contents as an outline of the manual and the software logic; then thumb through the manual for an impression of how the information is presented and how you may use it to help solve your problem.

Structured Documentation

Each pair of facing pages contains all the information on the topic you will be reading about; thus, you will not have to break your thinking to refer to figures and tables located elsewhere in the manual. Usually a figure will be on the right page and the related text will be on the facing page. You should try to grasp the concept of the topic by first studying just the figure and then reading the text.

Some DOS/VS module names are different from the OS/VS module names in complying with internal programming standards. Those names which begin with "CIPO" for OS/VS must begin with "CIPD" for DOS/VS. In this manual, the OS/VS spelling is used to represent the module for either operating system.

Related Publications

- Offline IBM 3800 Utility, SH20-9138

This manual provides information about planning, installing, and using the IBM 3800 Printing Subsystem.

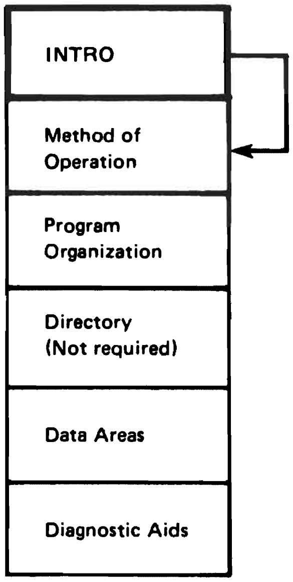
- IBM Offline 3800 Utility Program Listing LJB60009 (LJB60010 DOS/VS)

Each module is on a separate microfiche frame. The prolog of each module describes the required data areas.

INTRODUCTION

This section provides a self-contained overview, and refers mostly to the Method of Operation section, which describes the functions performed in greater detail. You should read this Introduction to gain a high-level understanding of how the program is used.

The IBM Offline 3800 Printer Utility is introduced in terms of INPUT, PROCESSING, and OUTPUT; OUTPUT is discussed more extensively.



Conceptual Overview

The purpose of the Offline IBM 3800 Printer is to generate a control tape to define the print characteristics of an output data set for the 3800 printer. The output data set can be a separate data set on the control tape or some other tape. The utility is also used to maintain the control modules from which the control tape is generated. The control modules do not maintain executable code.

Input

Figure 1 shows that the utility can be used with either OS/VS or DOS/VS. Utility control statements control the processing. The general format of a utility control statement is:

Label OPERATION OPERAND,OPERAND

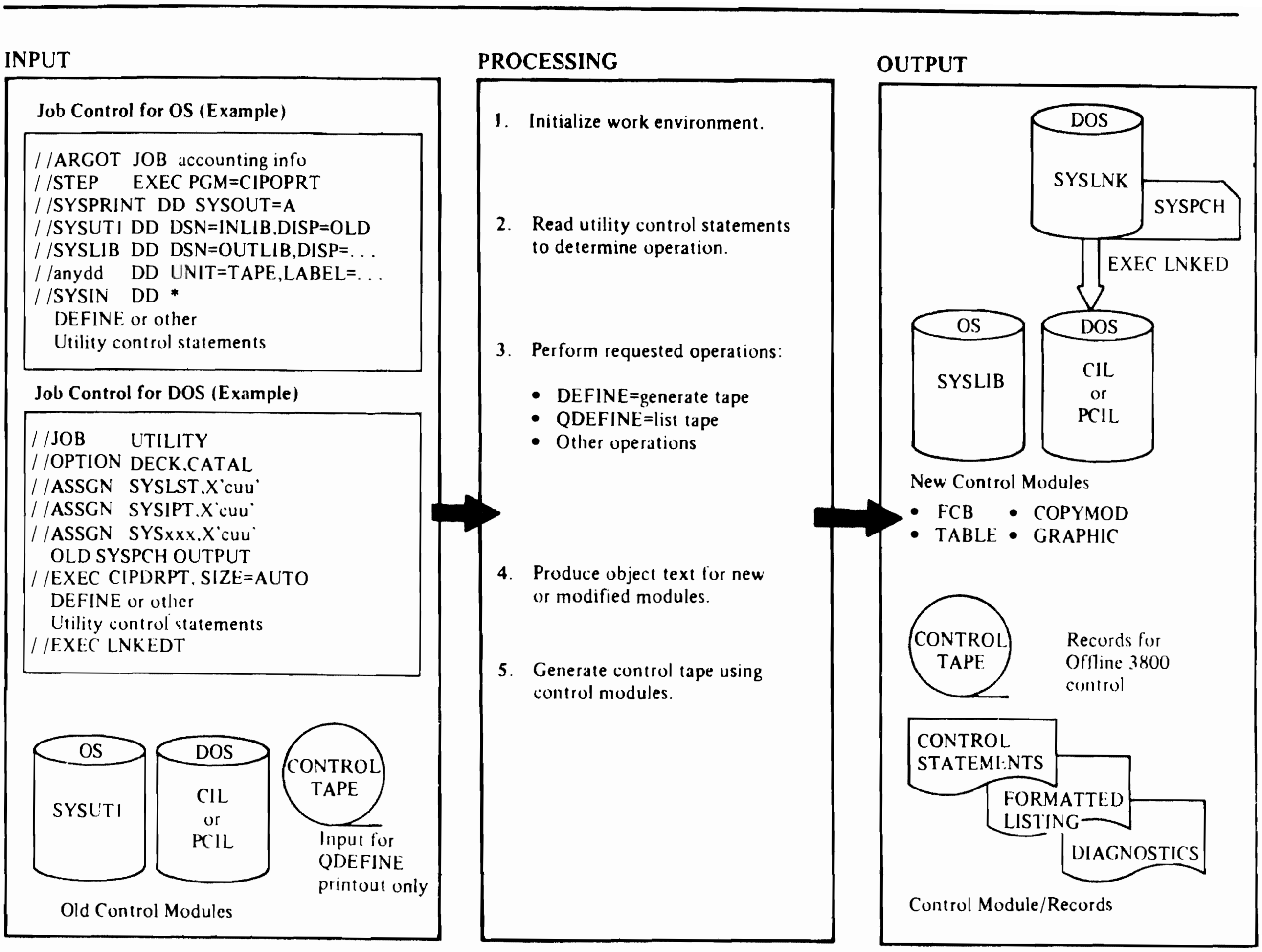
The "OPERATION" (above) can be INCLUDE, FCB, COPYMOD, TABLE, GRAPHIC, NAME, OPTION, DEFINE, or QDEFINE. The "OPERAND(s)" can be one or more keywords which correspond to the operation and complete its meaning. The old control modules can be modified or included with the output without modification. The control tape does not contain control modules.

Processing

Figure 1 shows that the computing space is initialized by the main procedure; the procedure CIPOSCAN is called to read and process the control statements. When the operation is determined, the appropriate subprocedure is then called to generate the required 3800 control module or to perform other operations specified by the control statements. The operations of maintaining the control modules and generating a control tape are independent of each other.

Output

Figure 1 shows the four types of modules maintained by the utility: Forms Control Buffer, Character Arrangement Table, Copy Modification, and Graphic Modification. The control tape contains records with information extracted from the control modules. Updated modules are automatically listed; however, listings can be obtained in a separate operation.



Job Control for OS (Example)

```
//ARGOT  JOB  accounting info
//STEP      EXEC PGM=CIPOPRT
//SYSPRINT  DD  SYSOUT=A
//SYSUT1 DD  DSN=INLIB.DISP=OLD
//SYSLIB  DD  DSN=OUTLIB,DISP=. . .
//anydd     DD  UNIT=TAPE,LABEL=. . .
//SYSIN    DD  *
  DEFINE or other
  Utility control statements
```

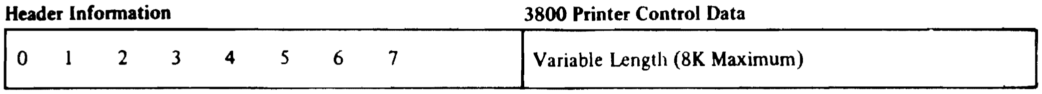
Job Control for DOS (Example)

```
//JOB       UTILITY
//OPTION DECK.CATAL
//ASSGN   SYSLST.X'cuu'
//ASSGN   SYSIPT.X'cuu'
//ASSGN   SYSxxx.X'cuu'
  OLD SYSPCH OUTPUT
//EXEC CIPDRPT. SIZE=AUTO
  DEFINE or other
  Utility control statements
//EXEC LNKEDT
```

1. Initialize work environment.
2. Read utility control statements to determine operation.
3. Perform requested operations:
   - DEFINE=generate tape
   - QDEFINE=list tape
   - Other operations
4. Produce object text for new or modified modules.
5. Generate control tape using control modules.

Figure 1. A Conceptual Overview

General Structure of Output

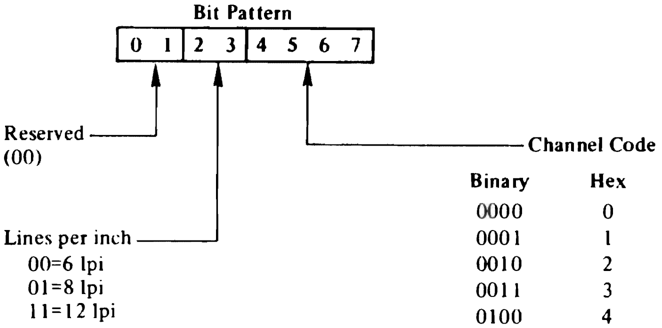


Module name (4 bytes)
Reserved (2 bytes)
Length of data (2 bytes)

Contains information for one of the following:
- - Forms control buffer
- - Copy modification
- - Character arrangement table
- - Graphic character modification

Forms Control Buffer Module

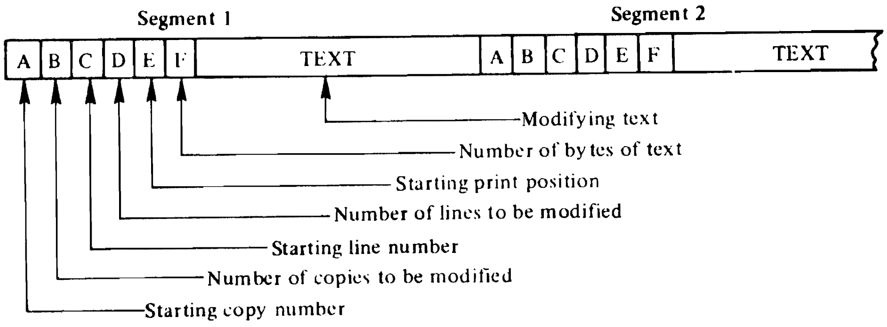
The module sho-vn below specifies control information for the vertical line spacing and any one of 12 channel codes per line.



Binary	Hex	Channel
0000	0	- (null)
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12

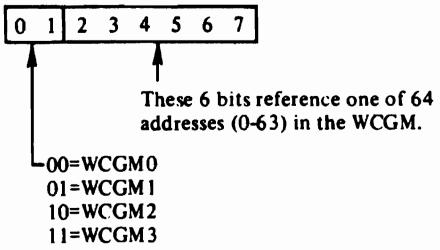
Copy Modification Module

The module shown below specifies data that is to be printed on every page for the specified number of copies of the output file.



Character Arrangement Table Module

The module shown below indicates how the input data is translated into addresses which are used to select the characters from the appropriate Writable Character Generation Module (WCGM).

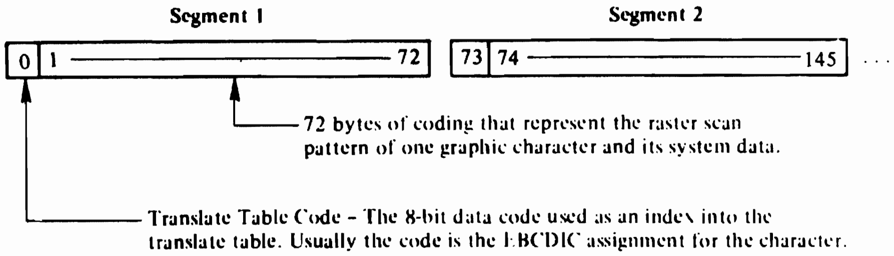


These 6 bits reference one of 64 addresses (0-63) in the WCGM.

00=WCGM0
01=WCGM1
10=WCGM2
11=WCGM3

Graphic Character Modification Module

The module shown below indicates the scan patterns of user-designed characters and/or characters from IBM-supplied modules.



72 bytes of coding that represent the raster scan pattern of one graphic character and its system data.

Translate Table Code – The 8-bit data code used as an index into the translate table. Usually the code is the EBCDIC assignment for the character.

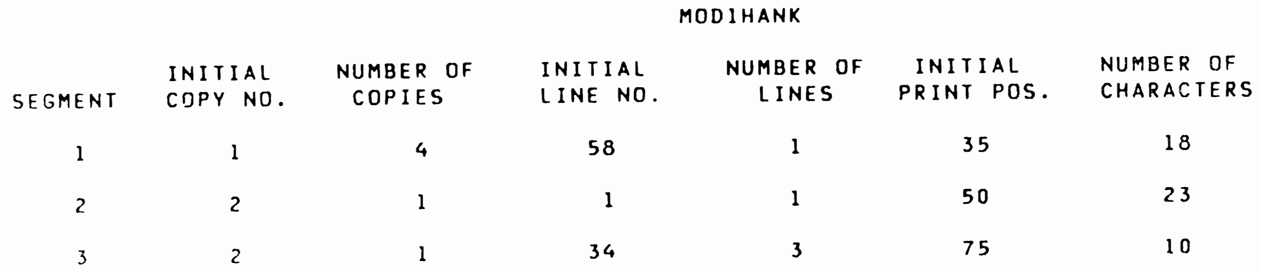
Control Tape Record

The control tape records illustrated below are read into the 3800 buffers by the tape-to-print feature. Detailed information about the control character is provided in *Offline IBM 3800 Utility*. The maximum block size is 8192 bytes.

	Block Length	00	Record Length	00	Control Character	Related Data	Record Length	00	Control Character	Related Data
Bytes	(2)	(2)	(2)	(2)	(1)	Vary	(2)	(2)	(2)	Vary

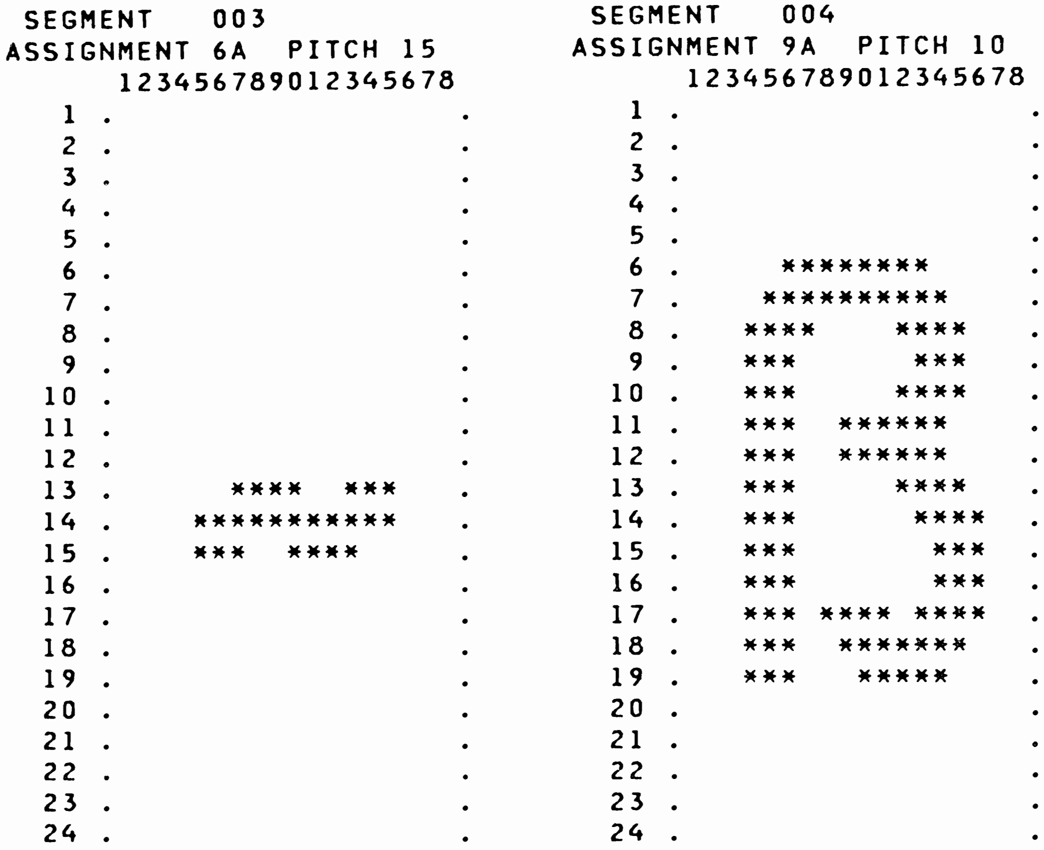
Examples of Output Listings

The following is an example of an OUTPUT listing of three segments of a copy modification module.



```
                                       MODIHANK

                INITIAL     NUMBER OF      INITIAL      NUMBER OF    INITIAL       NUMBER OF
   SEGMENT      COPY NO.     COPIES        LINE NO.       LINES     PRINT POS.     CHARACTERS

      1            1            4             58            1           35             18

      2            2            1              1            1           50             23

      3            2            1             34            3           75             10
```

The following is an example of an OUTPUT listing of two segments of a graphic character modification module.

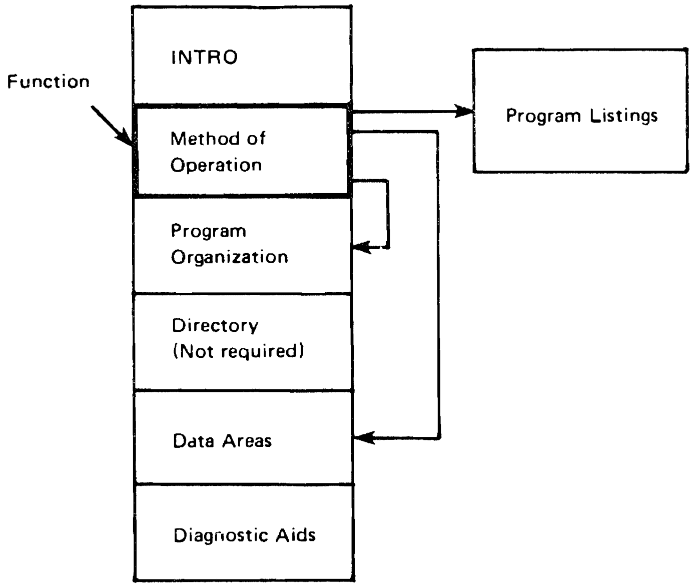


```
 SEGMENT   003                 SEGMENT   004
ASSIGNMENT 6A   PITCH 15      ASSIGNMENT 9A   PITCH 10
       123456789012345678            123456789012345678
    1 .                  .        1 .                  .
    2 .                  .        2 .                  .
    3 .                  .        3 .                  .
    4 .                  .        4 .                  .
    5 .                  .        5 .                  .
    6 .                  .        6 .     ********     .
    7 .                  .        7 .    **********    .
    8 .                  .        8 .   ****    ****   .
    9 .                  .        9 .   ***      ***   .
   10 .                  .       10 .   ***     ****   .
   11 .                  .       11 .   ***  ******    .
   12 .                  .       12 .   ***  ******    .
   13 .      ****  ***   .       13 .   ***     ****   .
   14 .    ***********   .       14 .   ***      ****  .
   15 .    ***  ****     .       15 .   ***       ***  .
   16 .                  .       16 .   ***       ***  .
   17 .                  .       17 .   *** **** ****  .
   18 .                  .       18 .   ***  *******   .
   19 .                  .       19 .   ***   *****    .
   20 .                  .       20 .                  .
   21 .                  .       21 .                  .
   22 .                  .       22 .                  .
   23 .                  .       23 .                  .
   24 .                  .       24 .                  .
```

METHOD OF OPERATION

This section discusses function; it provides you with module names and data area names which will be used in understanding the Program Listing and other sections of this manual.

Function is discussed in terms of input statements which designate functions; programming procedures invoked to perform the functions; and the two data areas which link the input statements to the procedures.



Functional Overview

Function is determined by utility control statements (Figure 2). The utility comprises various major functional procedures (Figure 3).

The Operation Table (OPTAB) and the Keyword Table (KEYTAB) shown in Figure 4 link the operations requested on the utility control statements with the subprocedures invoked to compete the operation.

The main procedure, CIPOPRT, calls subprocedures according to function; for example, if the FCB control statement is encountered, the subprocedure CIPOFCB is called to perform the specified function; or if the TABLE control statement is encountered, the subprocedure named CIPOTABL is called. The subprocedures are listed in Chapter 3.

The utility is table-driven by OPTAB and KEYTAB (described in Chapter 4). The following is a view of how these tables function, assuming the input utility control statement: FORM FCB SIZE=70, LPI=80:

- Scan OPTAB for an entry with RTNAME=FCB
- Set ACTIVOP (work area)=OPNUM (OPTAB), entries from top of table
- Get address of operation build routine from BLDPTR (OPTAB)
- Set output module prefix to "FCB3"; set defaults
- Scan KEYTAB for an entry with KEYWD=SIZE
- Set ACTIVKEY (work area) = (entries from first KEYTAB entry)
- Scan OPFLAGS (OPTAB); check that SIZE is valid with FCB operation
- Set KWDUSE (work area) =X (SIZE already used in this operation)
- Get address of SIZE processing routine from PRMPTR (KEYTAB)
- Process SIZE parameter
- Scan KEYTAB for next keyword (KEYWD=LPI)
- Repeat the steps above for processing keywords
- Build module FCB3 and perform output

Statement	**Function Invoked**
FCB	Generate a Forms Control Buffer module and list
TABLE	Generate a Character Arrangement Table and List
COPYMOD	Generate a Copy Modification Module and list
GRAPHIC	Generate a Graphic Modification Module and list
INCLUDE	Include a module with the subsequent statement
NAME	End an operation
OPTION	Supply optional information for selected operation.
DEFINE	Create a sequential data set as output9
QDEFINE	List contents of data set created by DEFINE

Figure 2. Utility Control Statements and Function

Procedure	**Function Performed**
CIPOPRT	Main procedure, initialize storage, invoke other procedures required to process the operations specified by the utility control statements
CIPOSCAN	Read and scan utility control statements and determine the operation using table look-up with OPTAB (operation table) and KEYTAB (keyword table)
CIPORITE	Produce object module for output
CIPOMSGS	Accept message number in field MSGN and edit data in MSGEDIT list in work area. Select message; insert variable data if needed. Write message to output device.
CIPOFCB	Construct and list FCB module
CIPOCOPY	Construct and list COPY module
CIPOGRAF	Construct and list GRAPHIC module
CIPOTABL	Construct and list TABLE module
CIPODEFN	Construct CONTROL RECORDS (tape)

Note: For DOS/VS names, use prefix CIPD in place of CIPO

Figure 3. Programming Procedures and Function

OPTAB

Routine Address	Operation Name	NOB	CHK/PTR	Operation Number	Valid Keywords

KEYTAB

Key Length	Routine Address	Flags/ Controls	Parameter Definitions	Routine Name

Figure 4. One Entry of Tables OPTAB and KEYTAB

Procedures and Subprocedures

The following lists the procedures and subprocedures for the IBM 3800 Printer Utility.

CIPOPRT Subprocedures	**Subfunction Performed**
INCLUDE	Process INCLUDE statement
INVOP	Process invalid operation
NAME	Process NAME statement
FIND	Locate member in library
LOADLIB	Load text into storage
PRINTQ	Output the print queue
PRTCARD	Print control statements
EODAD	Process end-of-file on SYSIPT
CONVERT	Binary-to-decimal conversion
DELCHK	Verifies delete request
READLIB	Load text records into buffer
OPTION	Process OPTION statement
QDEFINE	Process QDEFINE statement
SYNAD (OS only)	I/O error processor
TSYNAD (OS only)	Tape error processor
BKSZCK (OS only)	Ensure correct SYSPRINT block size
ABEND (OS only)	DCB abend exit
BKSZFIX (OS only)	Ensure correct block size (library)
TAPEND	Tape end of file routine
MOVELONG	Move long data strings

CIPOCOPY Subprocedures	**Subfunctions Performed**
COPYMOD	Create COPYMOD modules
COPYPRT	Print COPYMOD modules
COPCHK	Check COPYMOD modules

CIPOTABL Subprocedures	**Subfunctions Performed**
TABLE	Create TABLE module
TABLPRT	Print TABLE module
TABCHK	Check TABLE module

CIPOFCB Subprocedures	**Subfunctions Performed**
FCB	Create FCB module
FCBPRT	Print FCB module

CIPOGRAF Subprocedures	**Subfunctions Performed**
GRAPHIC	Creates GRAPHIC modules
GCMPRT	Print GRAPHIC modules
GRFCHK	Check GRAPHIC modules

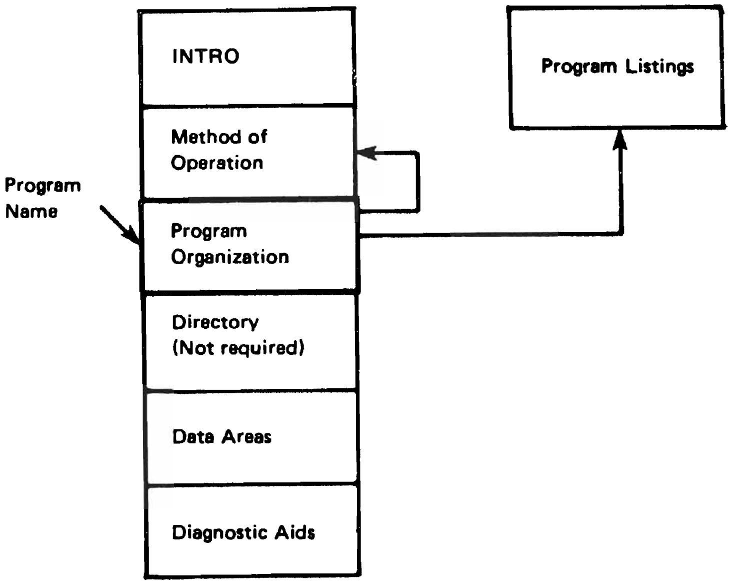
CIPODEFN Subprocedures	Subfunctions Performed
DEFINE	Create 3800 Offline control file
DEFPRT	Print 3800 (01) and (02) commands
LOADTAB	Process Character Arrangement Tables
CPYMODLD	Process Copy Modification modules
BLDTREC	Write records to control file

CIPOSCAN Subprocedures	Subfunctions Performed
A keyword name	Process respective keyword
GETCARDN	Read control statement
SCAN	Scan control statement fields
FINDPARM	Process keyword parameters
INVKEY	Process invalid keywords
SCANTEXT	Process text part of input
CONVERTR	Determine conversion
CONVERTX	Convert hexadecimal specifications
CONVERT	Decimal to binary conversion

CIPORITE Subprocedures	Subfunctions Performed
PUTEXT (DOS only)	Write object text to CNTLTAPE/SYSLNK
NUMBER (DOS only)	Generate sequence number for object text
WRITE (OS only)	Write modules to library
WRBLOCK (OS only)	Prepare block of data for library
WRTEXT (OS only)	Prepare text part of block for library

PROGRAM ORGANIZATION

This section shows program organization in terms of a structural overview of the programming procedures and high-level flowcharts. You should study this section to gain an understanding of the organization and structure of the program listing, and the logic flow of some of the major procedures.



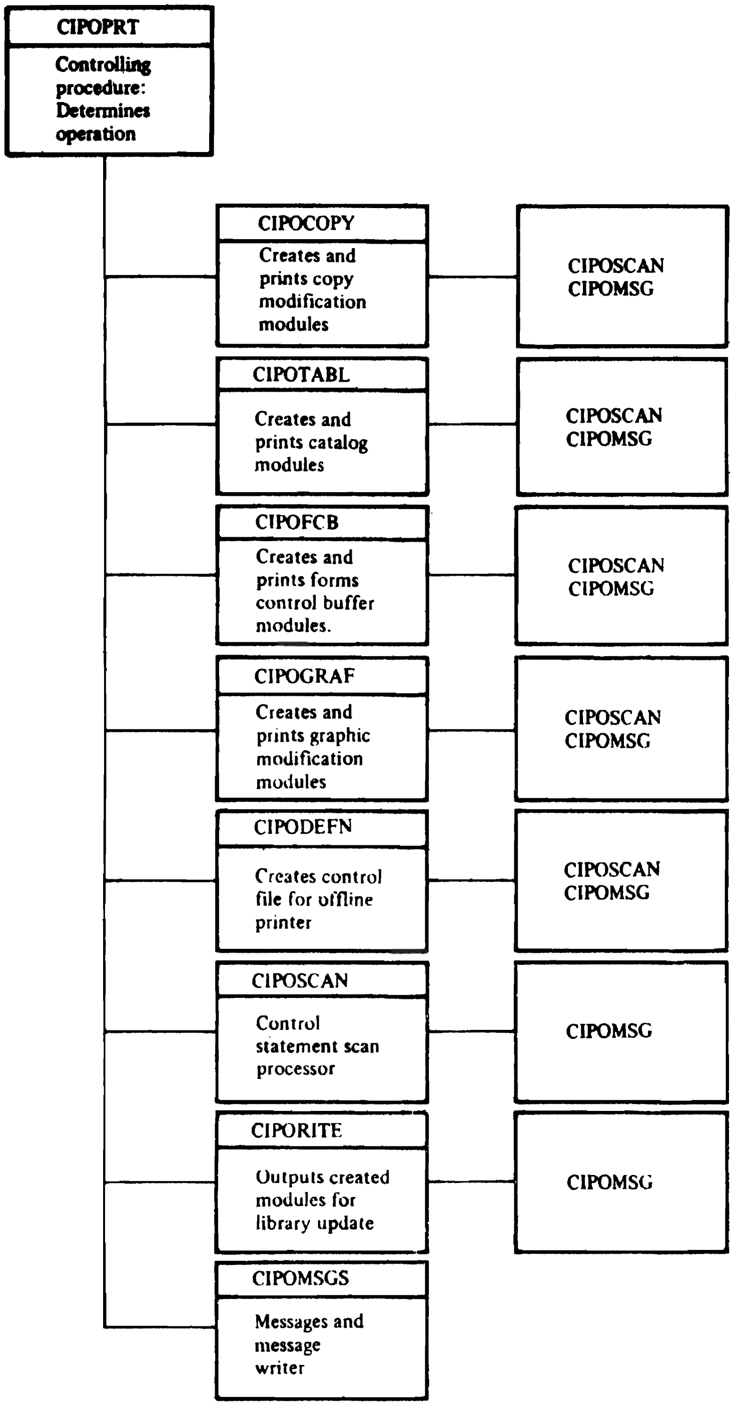

CIPOPRT
Controlling procedure: Determines operation
CIPOCOPY
Creates and prints copy modification modules
CIPOSCAN
CIPOMSG
CIPOTABL
Creates and prints catalog modules
CIPOSCAN
CIPOMSG
CIPOFCB
Creates and prints forms control buffer modules.
CIPOSCAN
CIPOMSG
CIPOGRAF
Creates and prints graphic modification modules
CIPOSCAN
CIPOMSG
CIPODEFN
Creates control file for offline printer
CIPOSCAN
CIPOMSG
CIPOSCAN
Control statement scan processor
CIPOMSG
CIPORITE
Outputs created modules for library update
CIPOMSG
CIPOMSGS
Messages and message writer

Figure 5. Structural Overview of Procedures

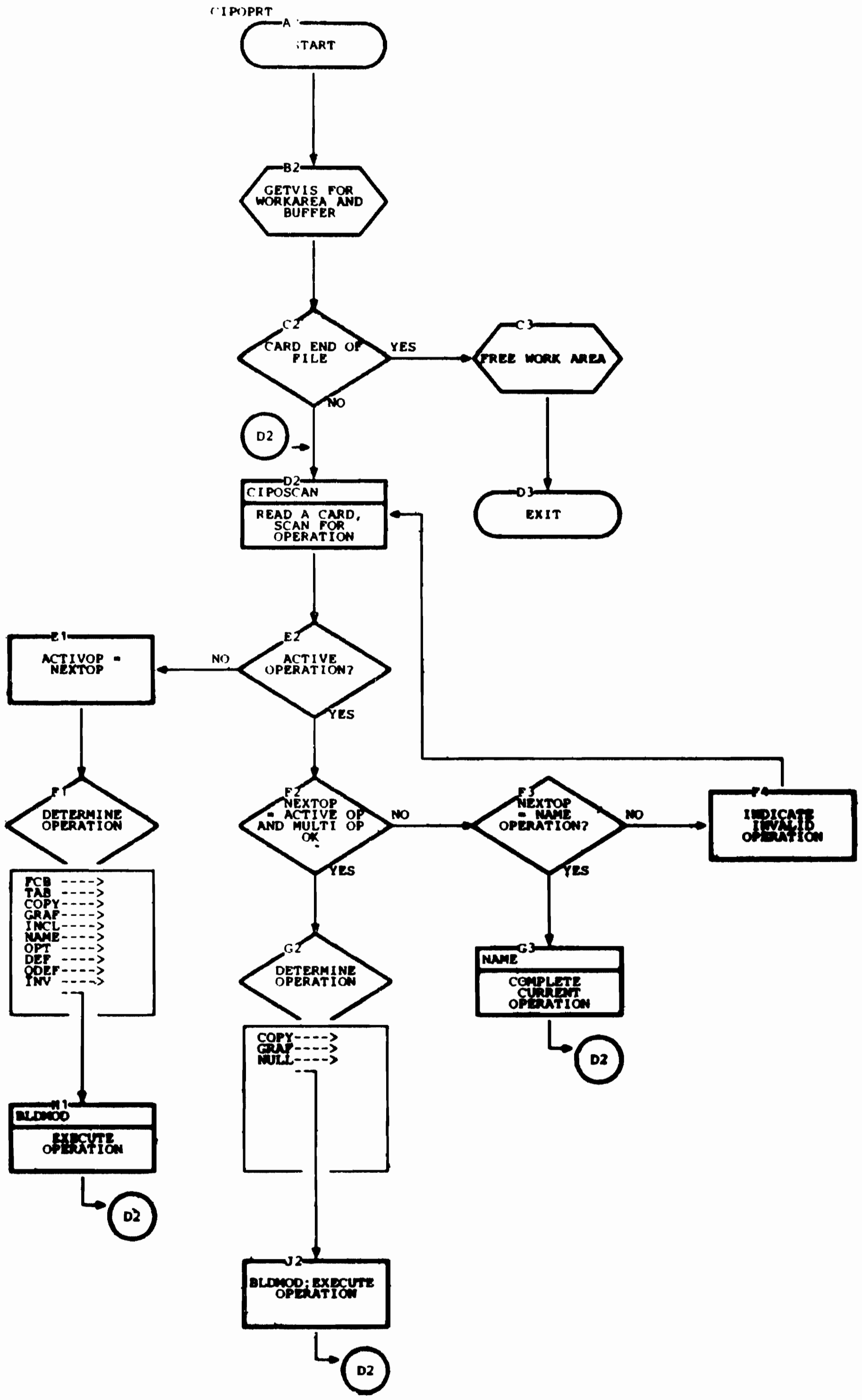


Figure 6. CIPOPRT Procedure

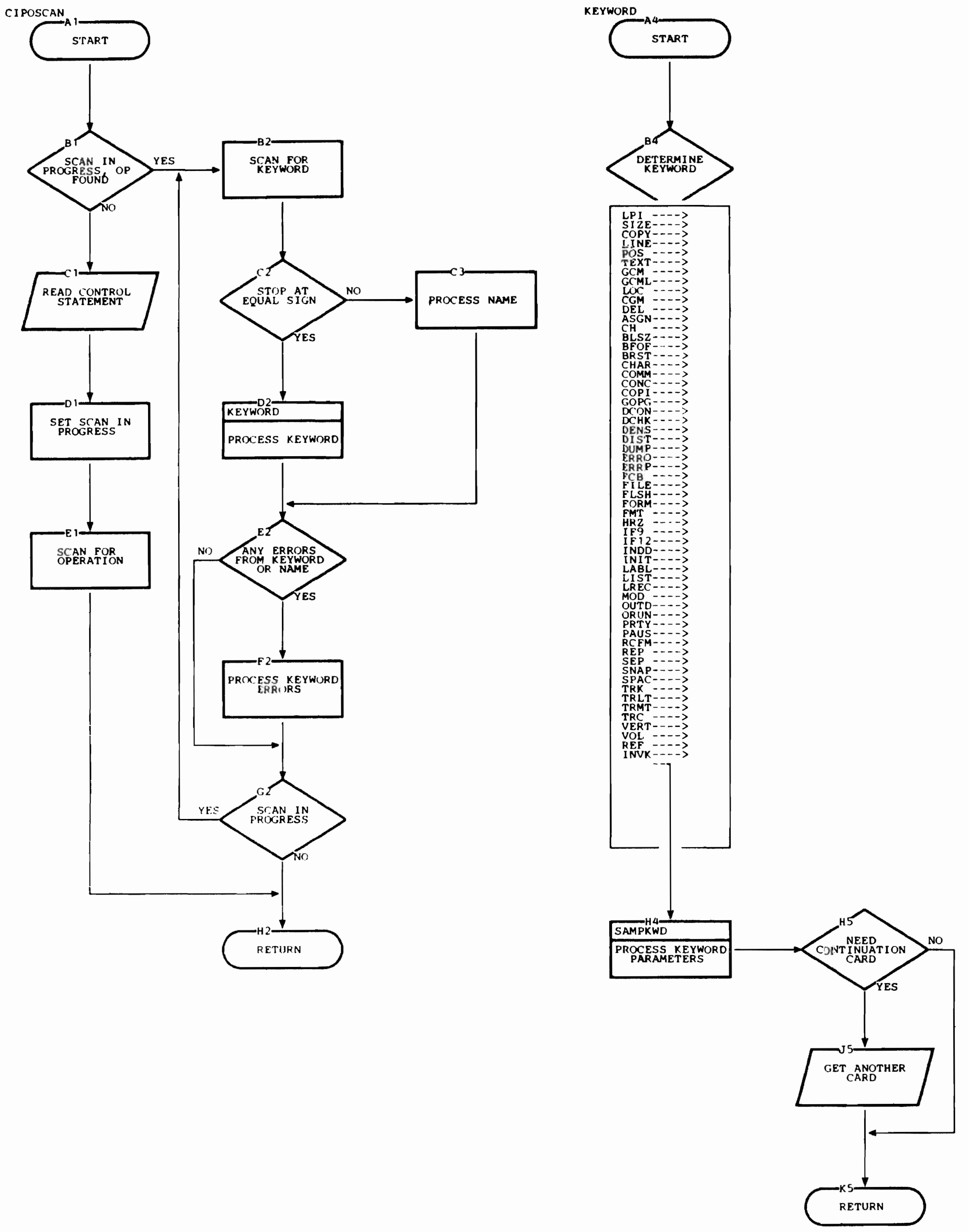


Figure 7. CIPOSCAN Procedure

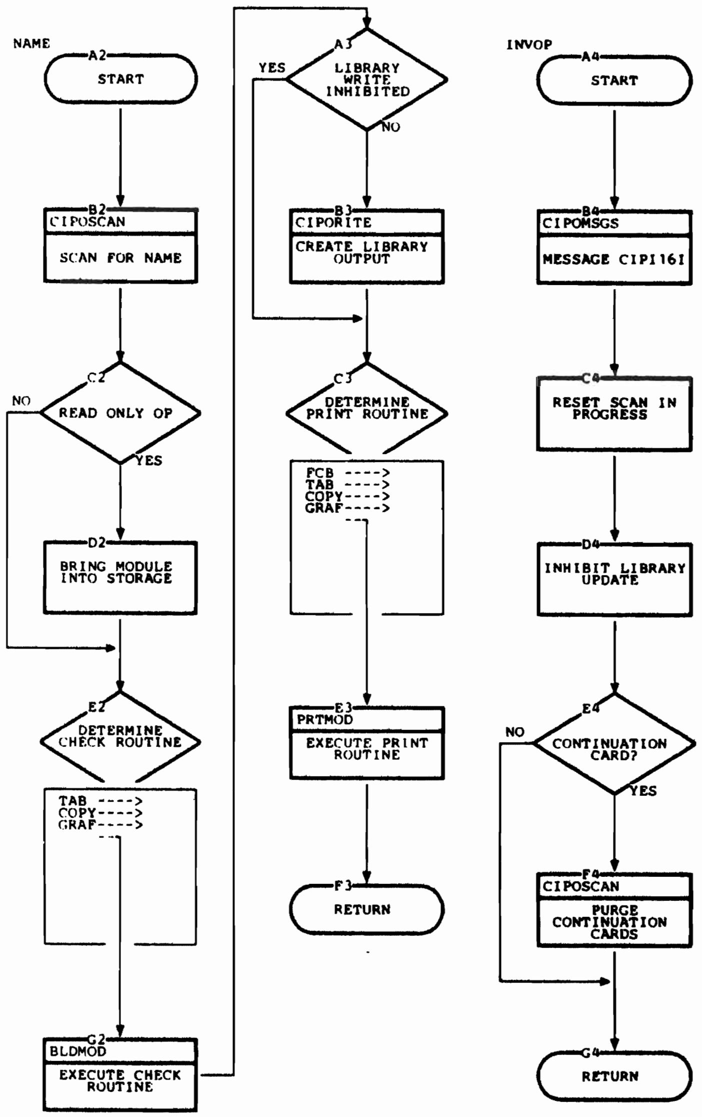


Figure 8. NAME Procedure and INVOP Procedure

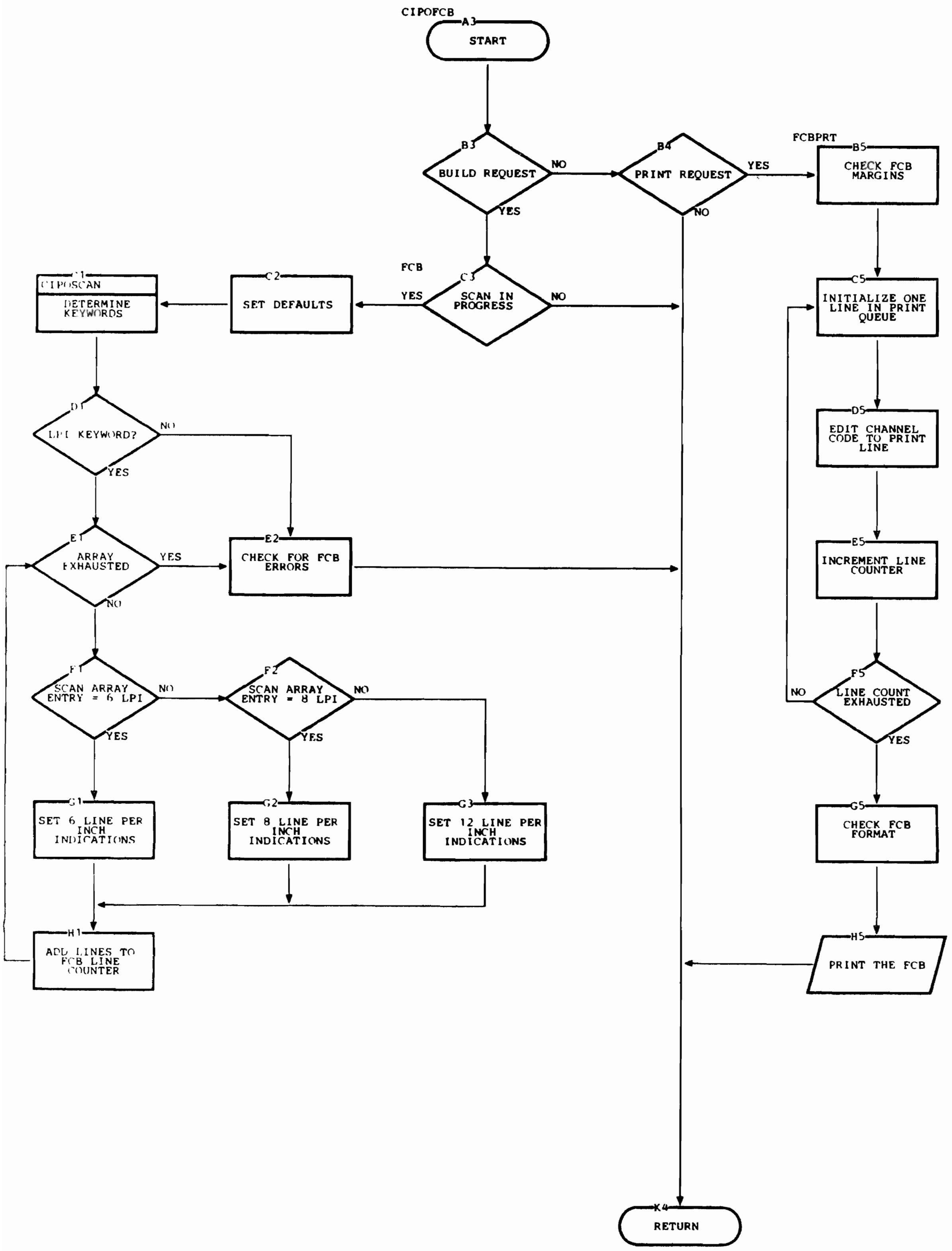


Figure 9. CIPOFCB Procedure

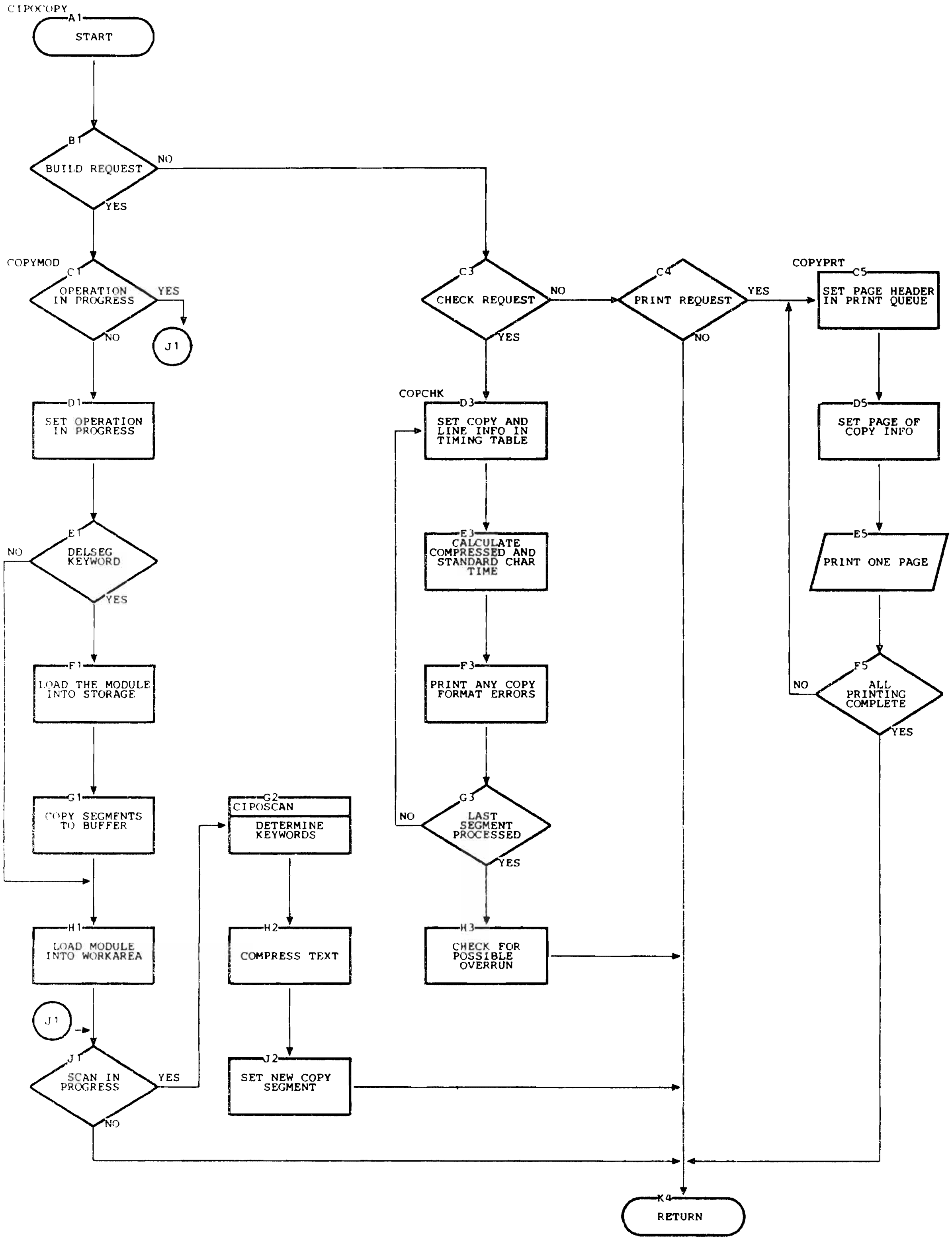


Figure 10. CIPOCOPY Procedure

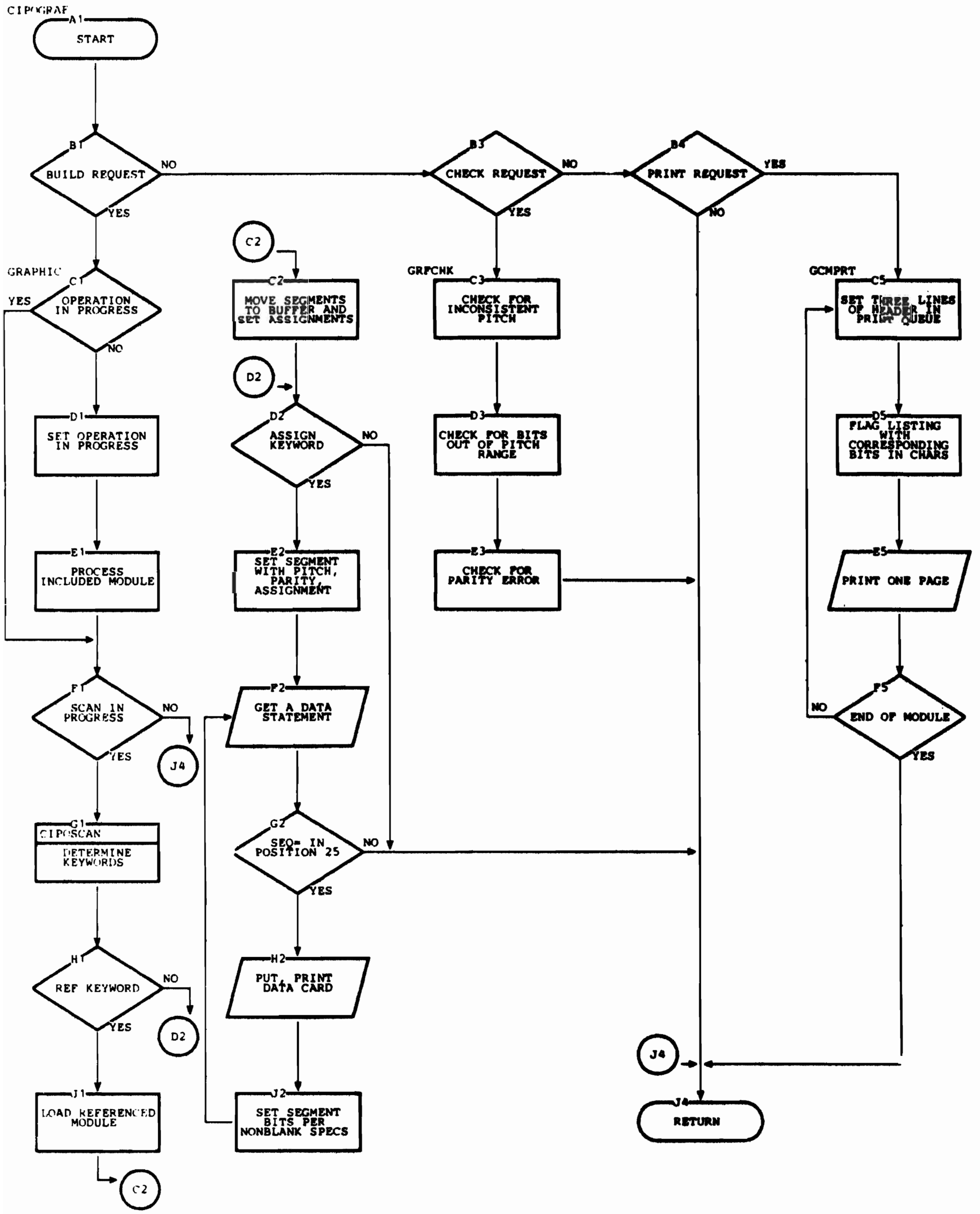


Figure 11. CIPOGRAF Procedure

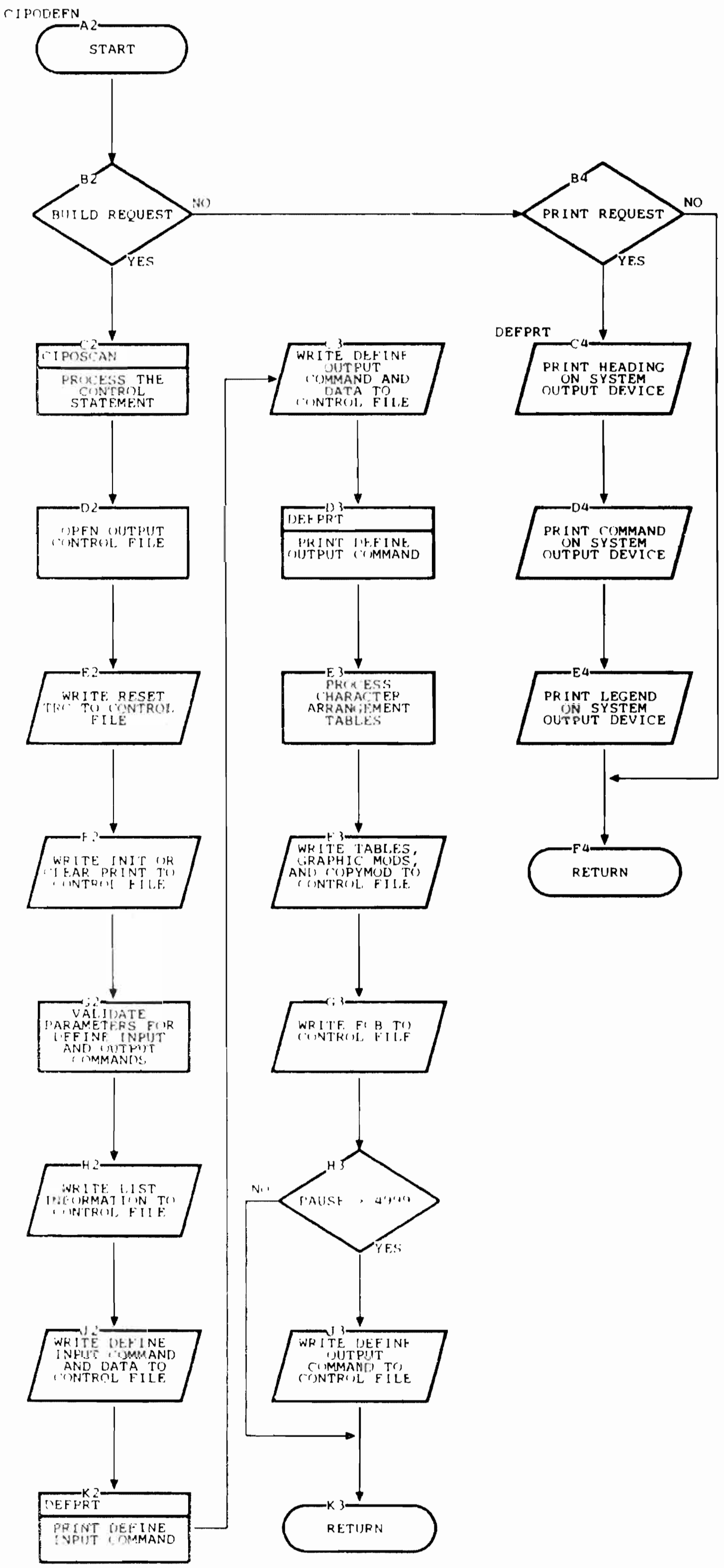

CIPODEFN
A2
START
B2
BUILD REQUEST
NO
YES
B4
PRINT REQUEST
NO
YES
C2
CIPOSCAN
PROCESS THE CONTROL STATEMENT
C3
WRITE DEFINE OUTPUT COMMAND AND DATA TO CONTROL FILE
DEFPRT
C4
PRINT HEADING ON SYSTEM OUTPUT DEVICE
D2
OPEN OUTPUT CONTROL FILE
D3
DEFPRT
PRINT DEFINE OUTPUT COMMAND
D4
PRINT COMMAND ON SYSTEM OUTPUT DEVICE
E2
WRITE RESET TRC TO CONTROL FILE
E3
PROCESS CHARACTER ARRANGEMENT TABLES
E4
PRINT LEGEND ON SYSTEM OUTPUT DEVICE
F2
WRITE INIT OR CLEAR PRINT TO CONTROL FILE
F3
WRITE TABLES, GRAPHIC MODS, AND COPYMOD TO CONTROL FILE
F4
RETURN
G2
VALIDATE PARAMETERS FOR DEFINE INPUT AND OUTPUT COMMANDS
G3
WRITE FCB TO CONTROL FILE
H2
WRITE LIST INFORMATION TO CONTROL FILE
H3
PAUSE > 4999
NO
YES
J2
WRITE DEFINE INPUT COMMAND AND DATA TO CONTROL FILE
J3
WRITE DEFINE OUTPUT COMMAND TO CONTROL FILE
K2
DEFPRT
PRINT DEFINE INPUT COMMAND
K3
RETURN

Figure 12. CIPODEFN Procedure

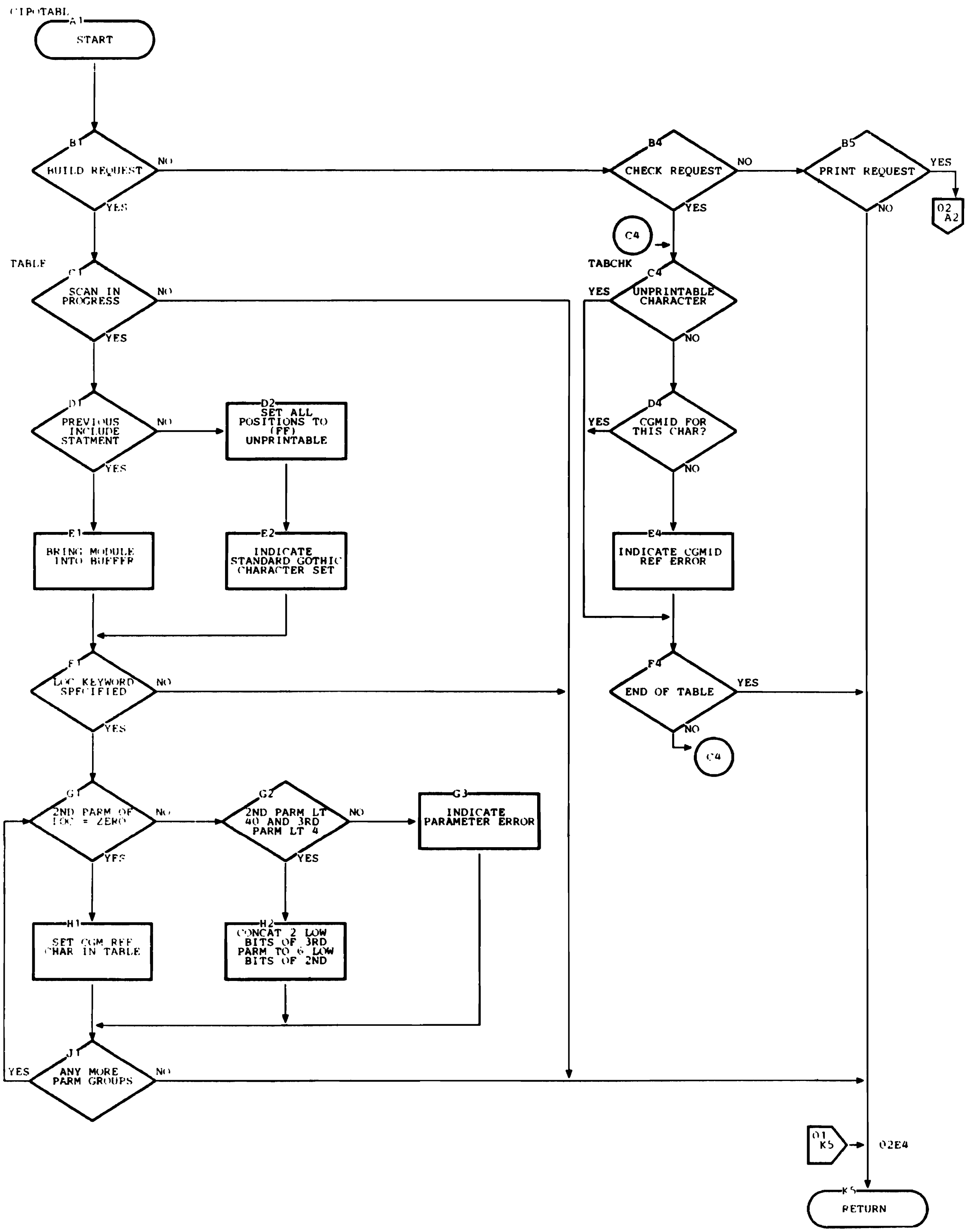


Figure 13 (Part 1 of 2). CIPOTABL Procedure

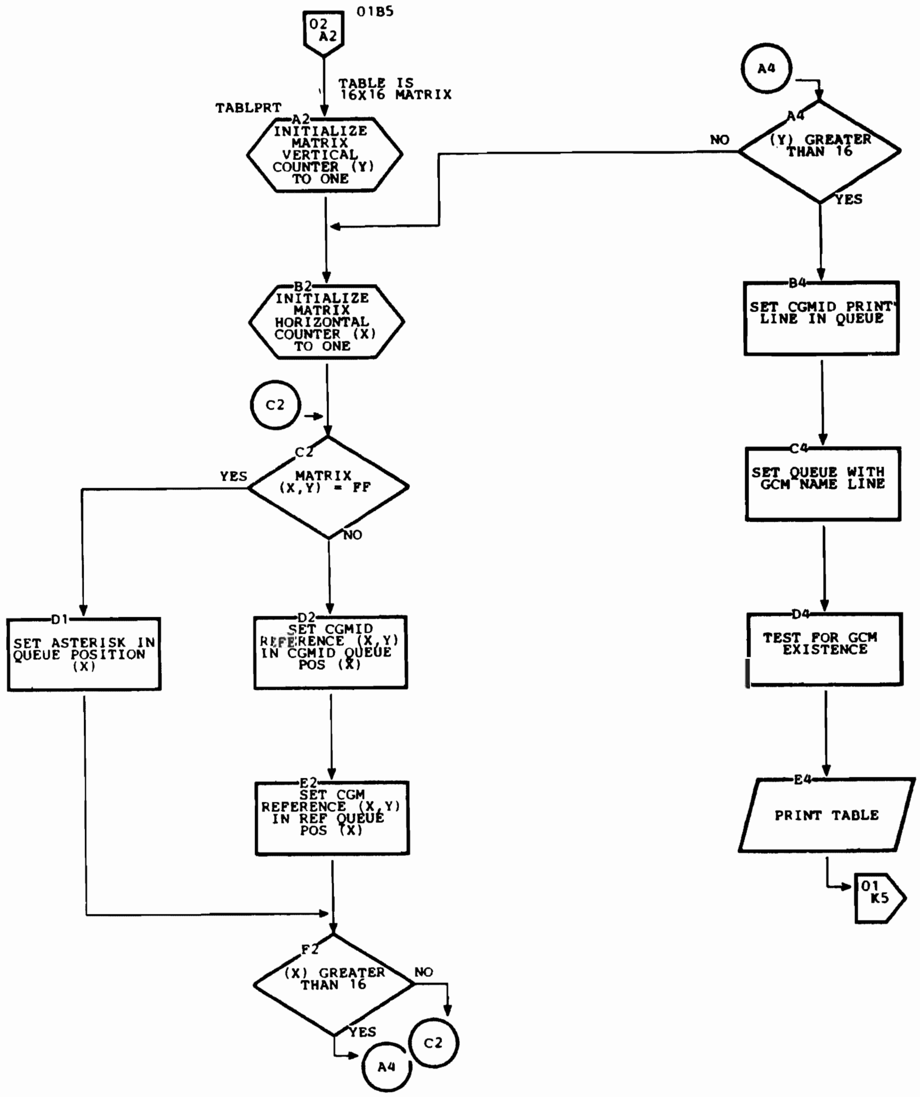

02
A2
01B5
TABLE IS
16X16 MATRIX
TABLPRT
A2
INITIALIZE MATRIX VERTICAL COUNTER (Y) TO ONE
B2
INITIALIZE MATRIX HORIZONTAL COUNTER (X) TO ONE
C2
C2
MATRIX (X,Y) = FF
YES
NO
D1
SET ASTERISK IN QUEUE POSITION (X)
D2
SET CGMID REFERENCE (X,Y) IN CGMID QUEUE POS (X)
E2
SET CGM REFERENCE (X,Y) IN REF QUEUE POS (X)
F2
(X) GREATER THAN 16
NO
YES
C2
A4
A4
A4
(Y) GREATER THAN 16
NO
YES
B4
SET CGMID PRINT LINE IN QUEUE
C4
SET QUEUE WITH GCM NAME LINE
D4
TEST FOR GCM EXISTENCE
E4
PRINT TABLE
01
K5

Figure 13 (Part 2 of 2). CIPOTABL Procedure

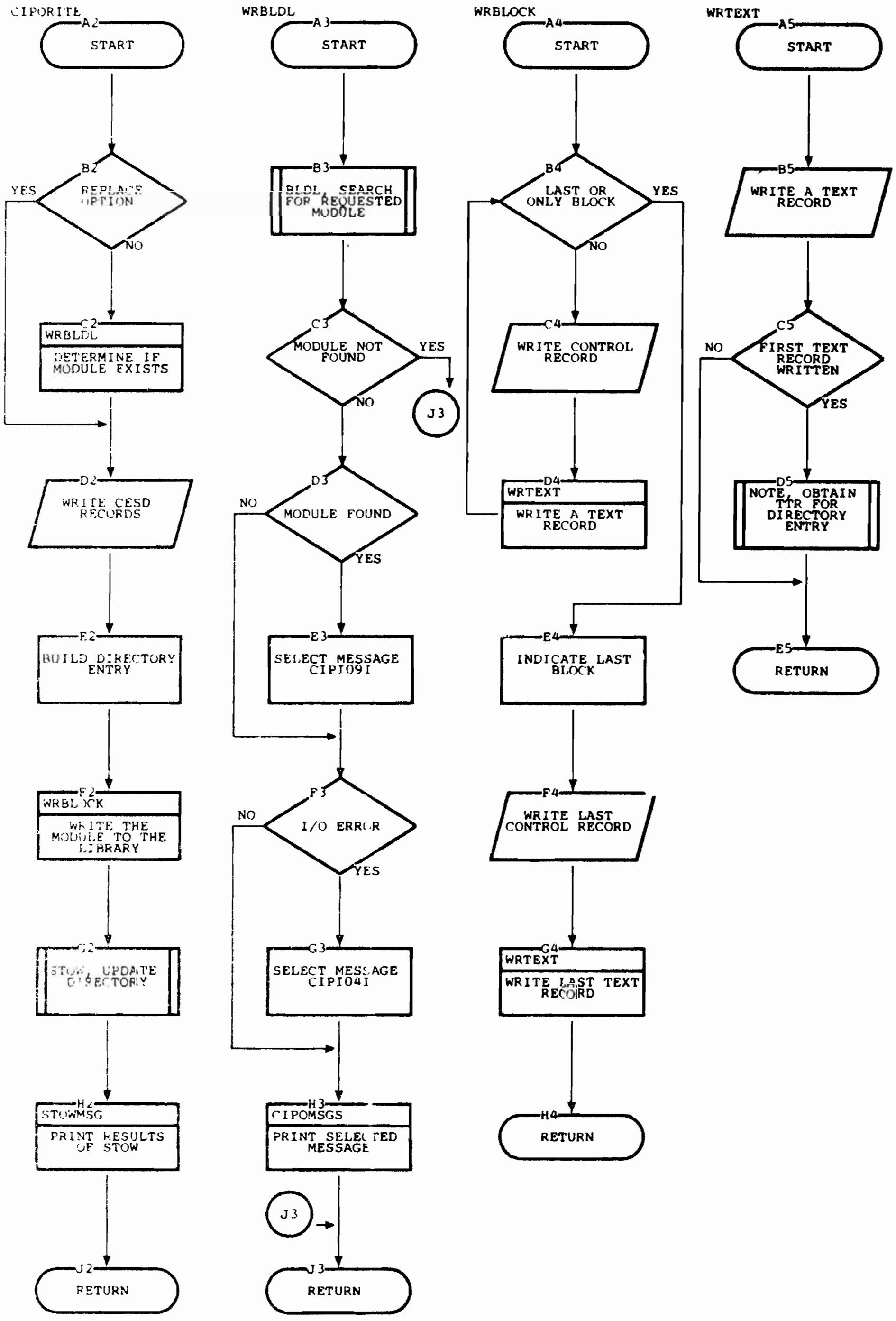


Figure 14. CIPORITE Procedure

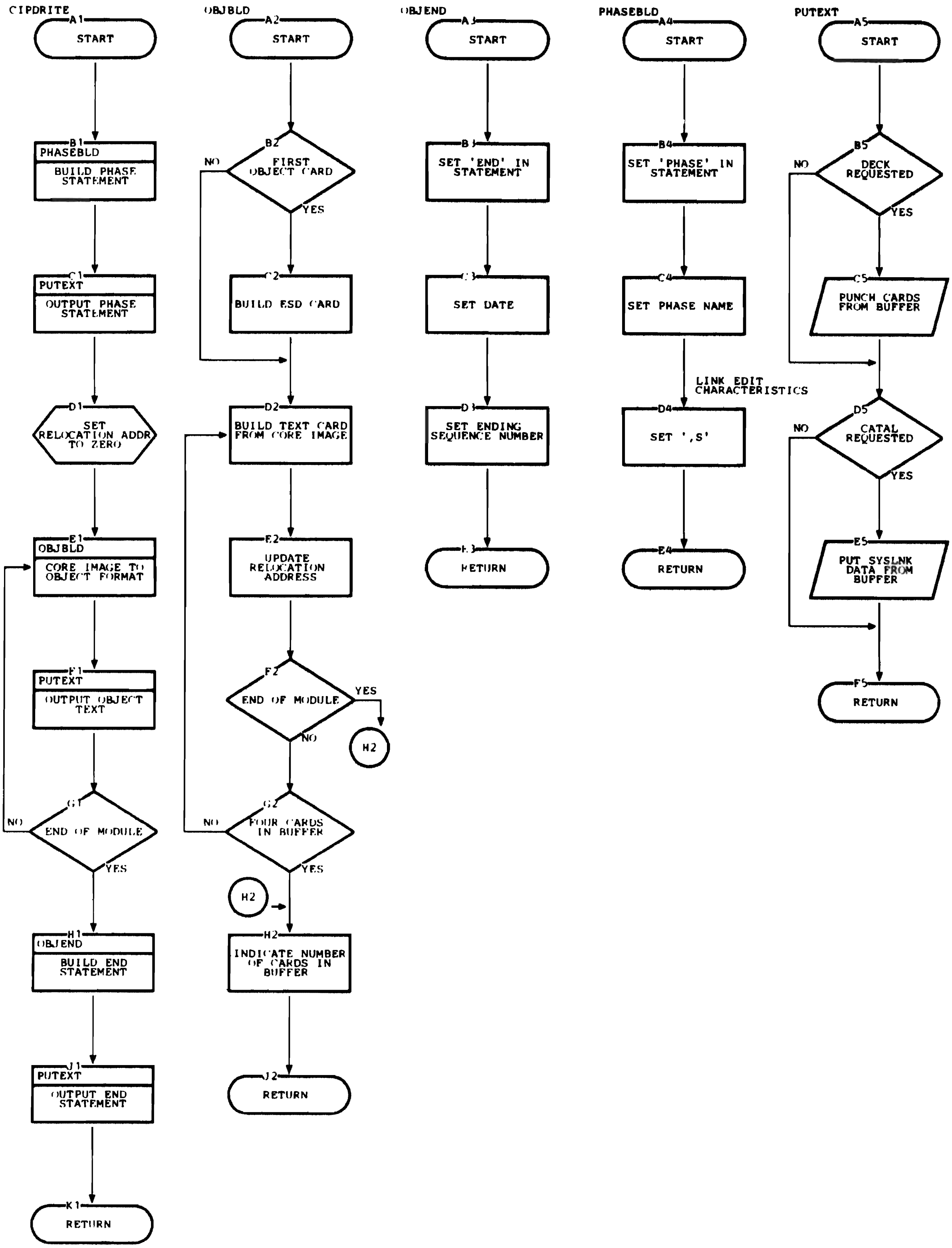
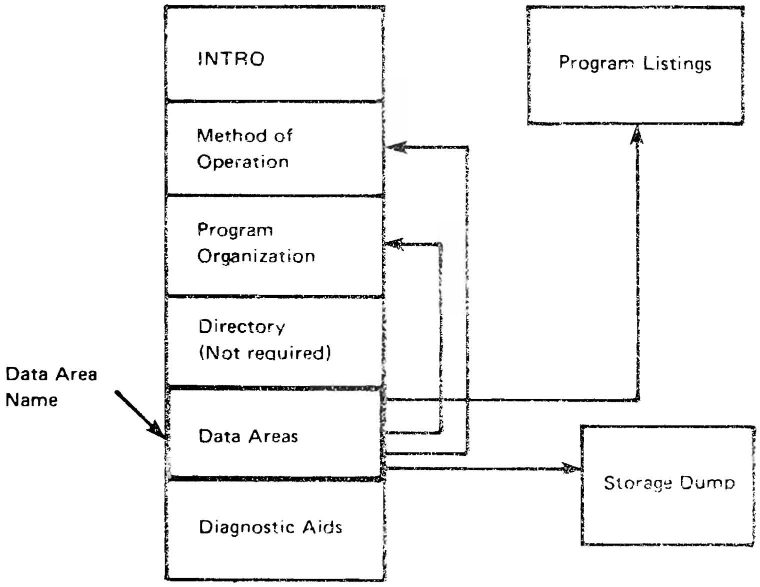


Figure 15. CIPDRITE (DOS/VS) Procedures

DATA AREAS

The Data Areas section provides data area names and formats useful in storage dump interpretation. It provides the names of object modules using each data area and, hence, guides you to the Program Organization section.



Data Area Descriptions

Figures 16, 17, 18, and 19 show some of the major data areas for the IBM 3800 Printer Utility. The data area descriptions are shown in the prolog of each module.

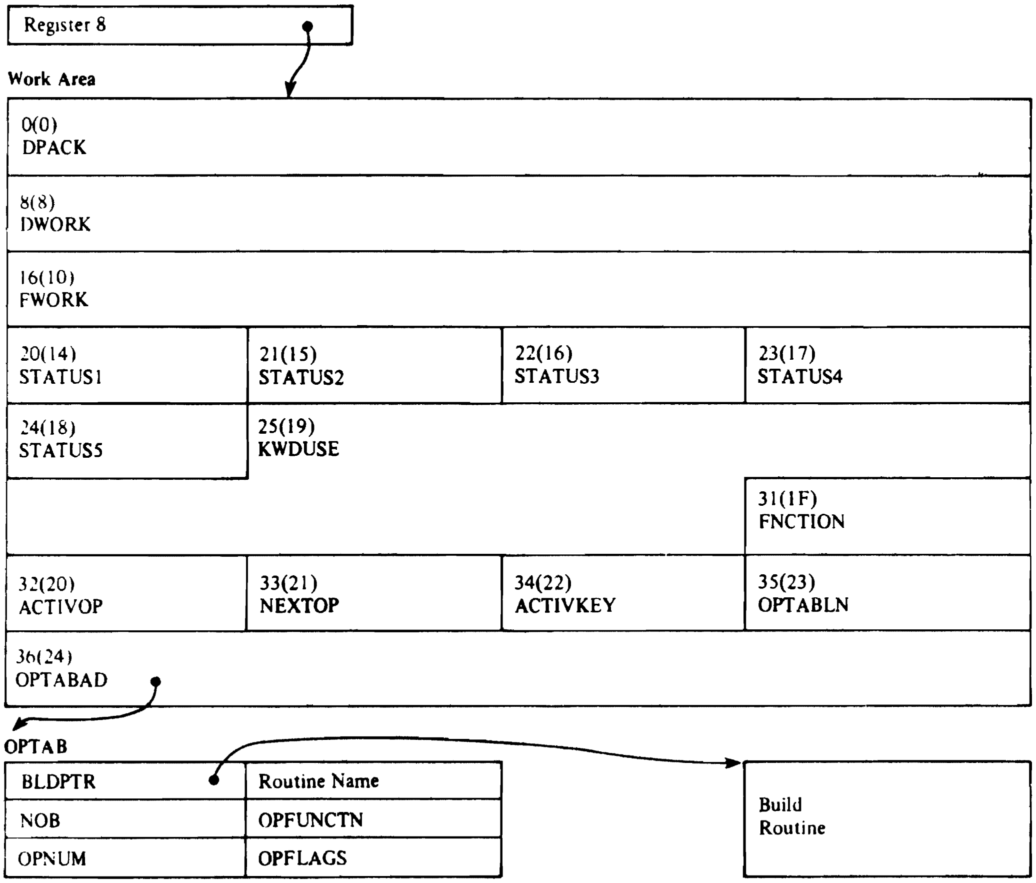


Figure 16. Data Area Overview

```
DECLARE
    1 KEYADDR(LENGTH(KEYTAB)/LENGTH(KELEM))  DEF(KEYTAB),
      2 KLEN FIXED(8),                    NUMBER OF BYTES IN NAME.
      2 PRMPTR PTR(24),                   POINTER TO ROUTINE.
      2 KFINDX     PTR(8),                KEYWORD ALLOWED INDEX.
      2 KFLAG      BIT(8),                KEYWORD TEST FLAG.
      2 KSINDX     PTR(8),                KEYWORD SET INDEX.
      2 KSETS      BIT(8),                KEYWORD SET BIT.
      2 KFLAGS     BIT(8),                KEYWORD ATTRIBUTE FLAGS.
        3 KMULT    BIT(1),                MULTIPLE PARAMETERS ALLOWED
        3 KPARN1   BIT(1),                SINGLE PARENTHESIS ALLOWED.
        3 KPARN2   BIT(1),                DOUBLE PARENTHESIS ALLOWED.
        3 VARI     BIT(1),                VARIABLE LENGTH KEYWORD.
      2 KCNT       FIXED(8),              MAXIMUM PARAMETERS ALLOWED.
      2 KPARM      BIT(48),               PARAMETER DEFINITIONS.
      2 KEYWD      CHAR(8),               NAME OF ROUTINE.
    1 PARMATTR(3)                         REDEFINE KPARM AS AN ARRAY.
              BASED(ADDR(KPARM(ACTIVKEY))),
        3 KPFLAGS BIT(8),                 PARAMETER ATTRIBUTE FLAGS.
          4 KDEC  BIT(1),                 DECIMAL DATA.
          4 KHEX  BIT(1),                 HEXADECIMAL DATA.
          4 KCHAR BIT(1),                 ALPHAMERIC DATA.
          4 KSPEC BIT(1),                 SPECIAL DATA.
        3 KPLEN    FIXED(8);              MAXIMUM LENGTH OF PARAMETER
```

Figure 17. Keyword Table (KEYTAB)

The following is a definition of the operation table as a 24-byte array of elements.

```
DECLARE
    1 TABLDEF(*) BASED(OPTABAD),           OPERATION TABLE STRUCTURE.
      2 BLDPTR       PTR(31),              POINTER TO OPERATION ROUTIN
      2 RTNAME       CHAR(8),              OPERATION NAME.
        3 RTNI       CHAR(1),              OPERATION NAME INITIAL.
      2 NOB          FIXED(8),             NUMBER OF CHARS IN NAME.
      2 OPFUNCTN     BIT(8),               FUNCTIONS ALLOWED.
        3 CHKPTR     BIT(1),               CHECK FUNCTION ALLOWED.
        3 PRTPTR     BIT(1),               PRINT FUNCTION ALLOWED.
      2 OPNUM        FIXED(8),             OPERATION NUMBER.
      2 OPFLAGS      BIT(64),              KEYWORDS VALID FOR OPERATN
        3 OPFLAG1    BIT(8),
          4 MLTFLAG  BIT(1),               MULTIPLE CONTROL STATEMENTS
          4 LPIFLAG  BIT(1),               ALLOW LPI=
          4 CHFLAG   BIT(1),                     CH1 THRU CH12=
          4 SIZEFLAG BIT(1),                     SIZE=
          4 COPYFLAG BIT(1),                     COPIES=
          4 LINEFLAG BIT(1),                     LINES=
          4 POSFLAG  BIT(1),                     POS=
          4 TEXTFLAG BIT(1),                     TEXT=
        3 OPFLAG2    BIT(8),
          4 NAMEFLAG BIT(1),               ALLOW NAME
          4 ASGNFLAG BIT(1),                     ASSIGN=
          4 DELFLAG  BIT(1),                     DELSEG=
          4 REFFLAG  BIT(1),                     REF=
          4 GCMFLAG  BIT(1),                     GCM=
          4 GCMLFLG  BIT(1),                     GCMLIST=
          4 LOCFLAG  BIT(1),                     LOC=
          4 CGMIDFLG BIT(1),                     CGMID=
        3 OPFLAG3    BIT(8),
          4 NOFLAG   BIT(1),               ALLOW NO KEYWORDS
          4 FRMTFLAG BIT(1),                     FORMAT=
          4 REPFLAG  BIT(1),                     REPLACE=
          4 OUTFLAG  BIT(1),                     OUTDD=
          4 BRSTFLAG BIT(1),                     BURST=
          4 CHARFLAG BIT(1),                     CHARS=
          4 COPGFLAG BIT(1),                     COPYG=
          4 DCHKFLAG BIT(1),                     DCHK=
        3 OPFLAG4    BIT(8),
          4 FCBFLAG  BIT(1),               ALLOW FCB=
          4 FLSHFLAG BIT(1),                     FLASH=
          4 FORMFLAG BIT(1),                     FORMS=
          4 CH9FLAG  BIT(1),                     IFCHAN9=
          4 CH12FLAG BIT(1),                     IFCHAN12=
          4 INITFLAG BIT(1),                     INIT=
          4 MDFYFLAG BIT(1),                     MODIFY=
          4 XMITFLAG BIT(1),                     TRANSMIT=

        3 OPFLAG5    BIT(8),
          4 SPCEFLAG BIT(1),               ALLOW SPACE=
          4 SEPFLAG  BIT(1),                     SEP=
          4 TRCFLAG  BIT(1),                     TRC=
          4 FSEQFLAG BIT(1),                     FILESEQ=
          4 CONCFLAG BIT(1),                     CONCHAR=
          4 DENSFLAG BIT(1),                     DENSITY=
          4 DCFLAG   BIT(1),                     DATACONV=
          4 EROPFLAG BIT(1),                     ERROPT=
        3 OPFLAG6    BIT(8),
          4 ERPCFLAG BIT(1),               ALLOW ERRPROC=
          4 LABLFLAG BIT(1),                     LABEL=
          4 LRECFLAG BIT(1),                     LRECL=
          4 BUFOFLAG BIT(1),                     BUFOFF=
          4 PRTYFLAG BIT(1),                     PARITY=
          4 RCFMFLAG BIT(1),                     RECFM=
          4 XLATFLAG BIT(1),                     TRANSLAT=
          4 TRCKFLAG BIT(1),                     TRACK=
```

Figure 18. Operation Table (OPTAB)

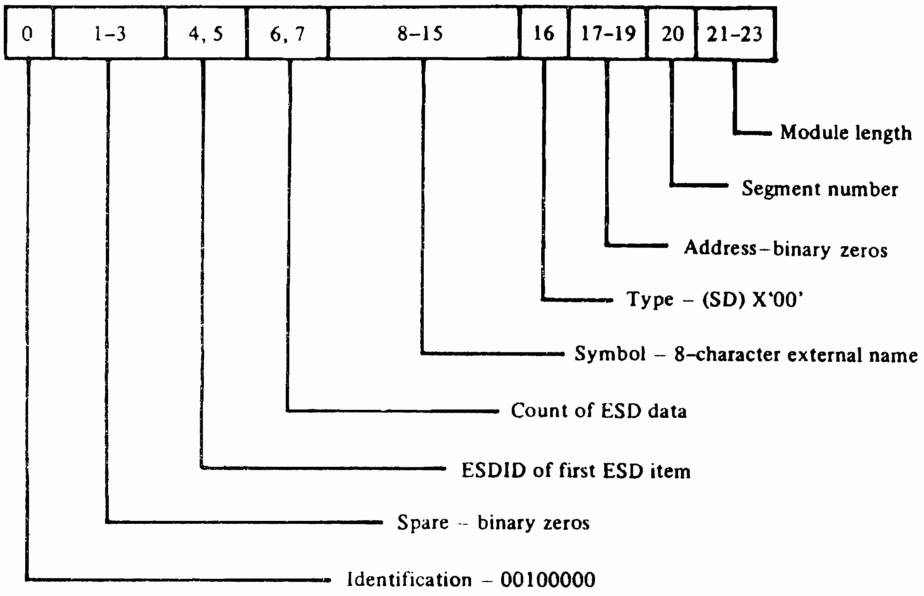
Work Area – CIPDBLKS

```
DECLARE                              GOTTEN STORAGE
  1 WORKAREA       BASED(BUFAD),     WORK AREA AND BUFFER SPACE
      3 DPACK   CHAR(8)BDY(DWORD),   CONVERT TO DECIMAL AREA.
        4 DWORK CHAR(8)BDY(DWORD),   DOUBLE WORD WORK AREA.
        4 FWORK       FIXED(31),     FULL WORD WORK AREA.

      3 STATUS1     BIT(8),          CURRENT STATUS OF UTILITY.
        4 EOFFLG    BIT(1),          END OF FILE PENDING.
        4 CARDIN    BIT(1),          CARD IN BUFFER INDICATION.
        4 CONTINUE  BIT(1),          CONTINUATION CARD INDICATOR
        4 DSEGSW    BIT(1),          DELSEG REQUEST MADE.
        4 PCLASGN   BIT(1),          PCIL IS ASSIGNED.
        4 OPTDECK   BIT(1),          DECK OPTION SELECTED.
        4 OPTCATL   BIT(1),          CATAL OPTION SELECTED.
      3 STATUS2     BIT(8),          CURRENT STATUS OF OPERATION
        4 REPUSE    BIT(1),          REPLACE OPTION INDICATOR.
          5 NOQFMT  BIT(1),          QDEFINE NO FORMAT REQUEST.
        4 INHIBIT   BIT(2),          INHIBIT WRITING/READING.
          5 INHIBITW  BIT(1),        INHIBIT WRITING PDS.
          5 INHIBITR  BIT(1),        INHIBIT READING PDS.
        4 INHIBITP  BIT(1),          INHIBIT PRINTING.
        4 FMTERR    BIT(1),          INDICATE FORMAT ERROR.
        4 FMTERR1   BIT(1),          INDICATE FORMAT ERROR.
        4 PRTYERR   BIT(1),          PARITY ERROR INDICATOR.
          5 SETRAS  BIT(1),          ONE RASTER SCAN SET.
            6 TEXTSW BIT(1),         TEXT SCAN IN PROGRESS.
        4 READMOD   BIT(1),          READ AND PRINT ONLY.
      3 STATUS3     BIT(8),          CURRENT STATUS OF SCAN.
        4 INVDLM    BIT(1),          INVALID DELIMITER.
        4 FOUND     BIT(1),          OPERATION OR KEYWORD FOUND.
        4 INVKWD    BIT(1),          INVALID KEYWORD INDICATOR.
        4 EXCLKWD   BIT(1),          EXCLUSIVE KEYWORD INDICATOR
        4 INVPARM   BIT(1),          INVALID PARAMETER.
        4 INVCHAR   BIT(1),          INVALID CHARACTER.
        4 UNBAL     BIT(1),          UNBALANCED PARENTHESIS.
        4 KWDCOMP   BIT(1),          KEYWORD SCAN COMPLETE.
      3 STATUS4     BIT(8),          CURRENT STATUS (MISC).
        4 KEYIN     BIT(1),          KEYWORD READY.
        4 SCANINP   BIT(1),          SCAN IN PROGRESS.
        4 INCLSW    BIT(1),          INDICATE INCLUDE KEYWORD.
        4 HEXERR    BIT(1),          HEX ERROR DURING CONVERT.
        4 FLDSW     BIT(1),          WITHIN PARM GROUP SWITCH.
        4 PARMGRP   BIT(1),          PARAMETER GROUP SWITCH.
        4 OPINPRG   BIT(1),          OPERATION IN PROGRESS.
        4 LASTQ     BIT(1),          LAST QUOTE OR CARD.
      3 STATUS5     BIT(8),          DEFINE FUNCTION STATUS
        4 INHIBITC  BIT(1),          INHIBIT CONTROL FILE OUTPUT
        4 INZFORCE  BIT(1),      .   FORCE INIT PRINTER COMMAND.
        4 INCMDERR  BIT(1),          ERR IN INPUT CMD DEFINITION
        4 TEOF      BIT(1),          CONTROL FILE END OF FILE.
      3 KWDUSE      BIT(48),         KEYWORD BIT FLAGS.
        4 KWDUSED(6)BIT(8),          KEYWORDS USED IN OPERATION.
      3 FNCTION     BIT(8),          OPERATIONAL FUNCTION.
        4 BUILD     BIT(1),          BUILD FUNCTION.
        4 PRINT     BIT(1),          PRINT FUNCTION.
        4 CHECK     BIT(1),          CHECK FUNCTION.
      3 ACTIVOP     FIXED(8),        ACTIVE OPERATION NUMBER.
      3 NEXTOP      FIXED(8),        NEXT OPERATION NUMBER.
      3 ACTIVKEY    FIXED(8),        ACTIVE KEYWORD.
      3 OPTABLN     FIXED(8),        LENGTH OF OPERATION TABLE.
      3 OPTABAD     PTR(31),         ADDR OF OPERATION TABLE.
```

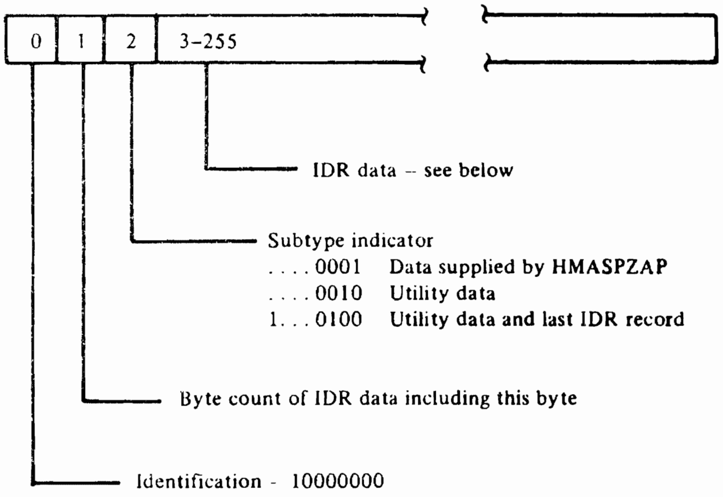
Figure 19. Work Area

CESD Records

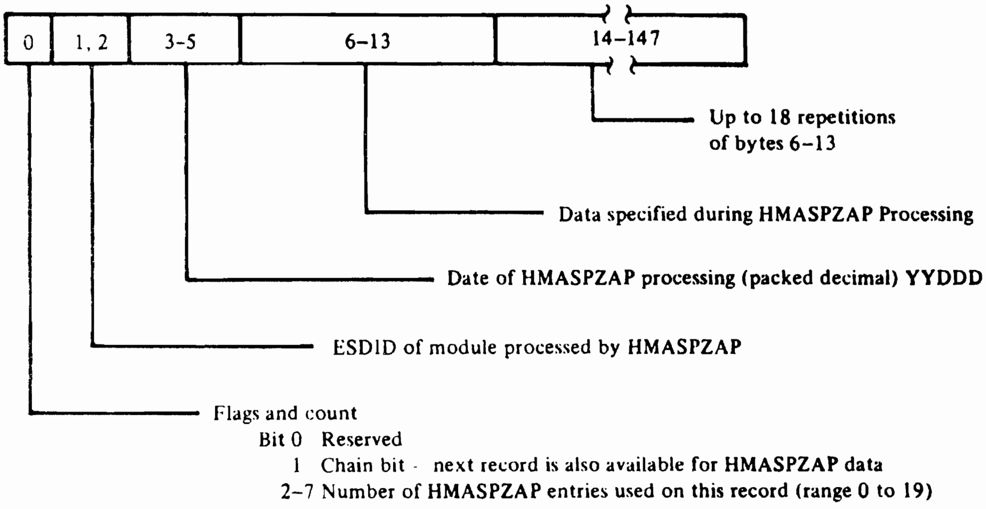
The OS and OS/VS utility writes five CESD records before each text record on the library to maintain compatibility with the linkage editor and other utility programs.



Identification Record


0
1
2
3-255
IDR data – see below
Subtype indicator
. . . . 0001 Data supplied by HMASPZAP
. . . . 0010 Utility data
1. . . 0100 Utility data and last IDR record
Byte count of IDR data including this byte
Identification - 10000000

HMASPZAP Data



Up to 18 repetitions of bytes 6-13

Data specified during HMASPZAP Processing

Date of HMASPZAP processing (packed decimal) YYDDD

ESDID of module processed by HMASPZAP

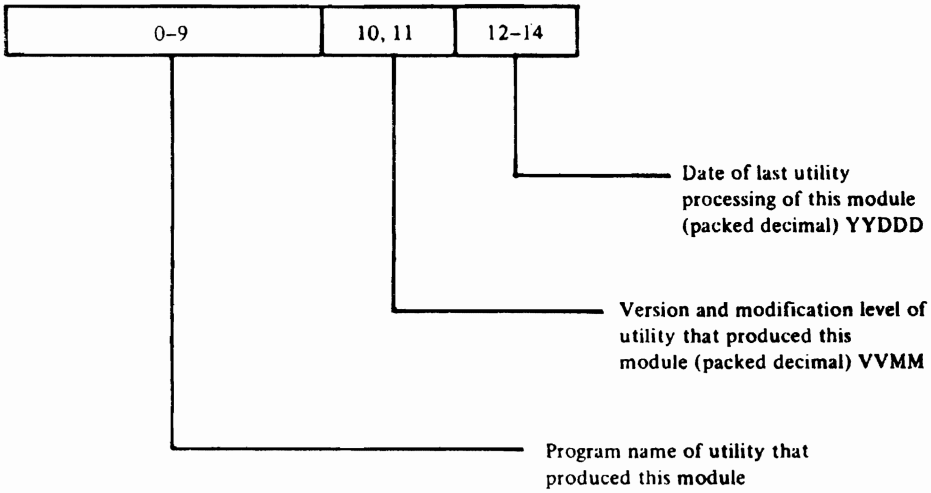
Flags and count

Bit 0 Reserved

1 Chain bit - next record is also available for HMASPZAP data

2-7 Number of HMASPZAP entries used on this record (range 0 to 19)

Utility Data

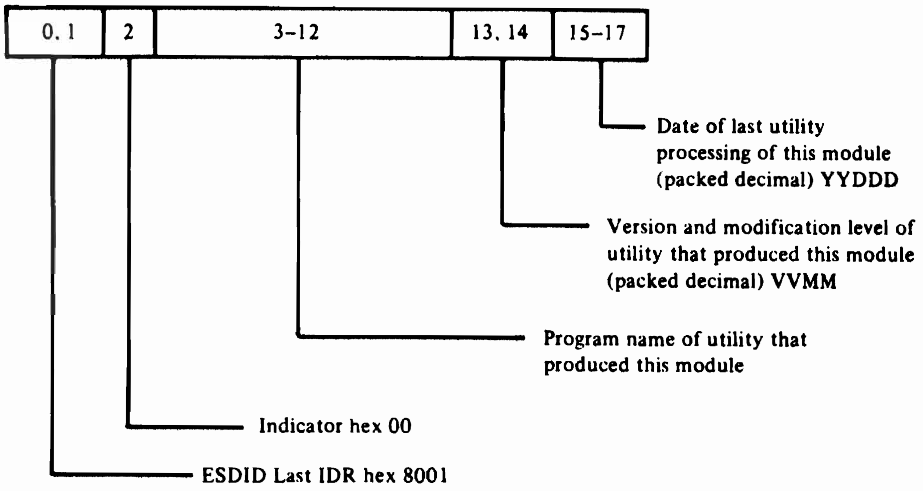


Date of last utility processing of this module (packed decimal) YYDDD

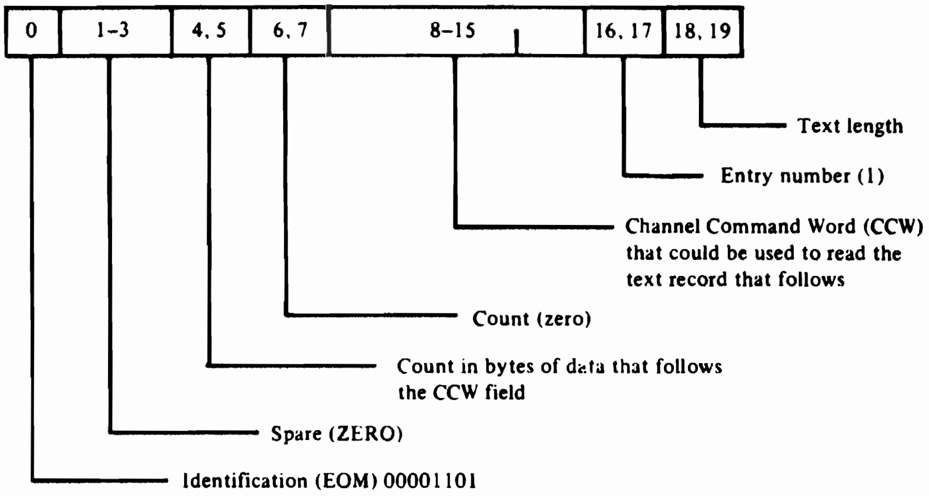
Version and modification level of utility that produced this module (packed decimal) VVMM

Program name of utility that produced this module

Utility Data (Supplied for Linkage Editor Compatibility)

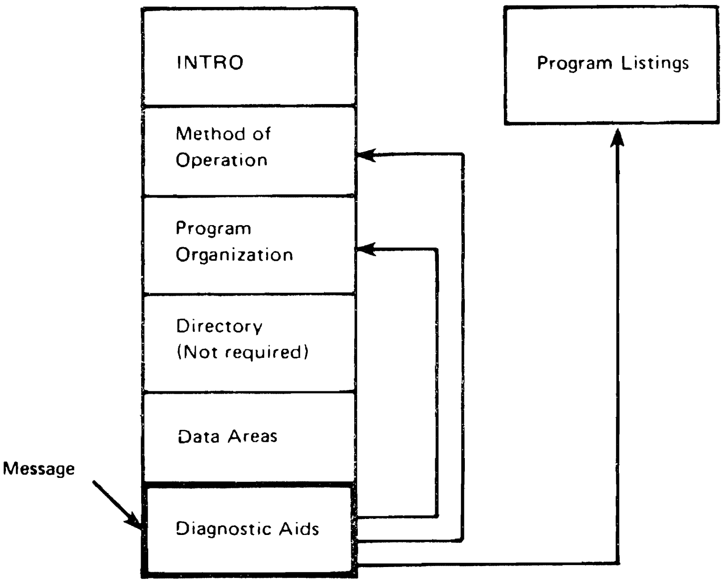

0, 1
2
3–12
13, 14
15–17
Date of last utility processing of this module (packed decimal) YYDDD
Version and modification level of utility that produced this module (packed decimal) VVMM
Program name of utility that produced this module
Indicator hex 00
ESDID Last IDR hex 8001

CESD (Last Record)


0
1–3
4, 5
6, 7
8–15
16, 17
18, 19
Text length
Entry number (1)
Channel Command Word (CCW) that could be used to read the text record that follows
Count (zero)
Count in bytes of data that follows the CCW field
Spare (ZERO)
Identification (EOM) 00001101

DIAGNOSTIC AIDS

This section guides you in generating RETAIN/EWS keywords for the Offline 3800 Printer Utility.



Retain Keyword Concept

The RETAIN keywords provided in this chapter will help the PSR to use the RETAIN and EWS systems in a standard way for this program. The PSR must:

1. Identify the failure by a string of RETAIN keywords
2. Search RETAIN (or EWS) for the fix
3. Apply the fix if found
4. Add keywords to the APAR if fix not found

Example of Keyword Entry

The following string of keywords would be entered as a retain search argument to identify a failure where the type of defect was a WAIT which occurred in procedure CIPOSCAN, and subprocedure GETCARDN; the resource involved was SYSIN:

AR340 TPTF2 WAIT CIPOSCAN GETCARDN SYSIN

Five Types of Keywords

There are five types of keywords in general:

1. Product identification keywords:

xxxxxx..	Component I D
ARvrm	Applicable version, release, modification
TPTFx	Tested PTF maintenance level

2. Type-of-defect keywords:

LOOP	Symptom of failure was a loop status
WAIT	Symptom of failure was a wait status
INCORROUT	User output defective
PERFM	Performance defect
DOC	Documentation defective
others	

3. Procedure-that-failed keyword
   Name of procedure that failed

4. Subprocedure-that-failed keyword
   Name of subprocedure that failed

5. Resource-involved keyword
   Input/Output devices, or internal tables changed by user

CIPOPRT Keywords

The five general types of keywords are defined here specifically for this utility.

Product-identification-keywords (required):

Component ID	5748-UT200
ARvrm	Ask customer for version, release, modification
TPTFx	Ask customer for PTF maintenance level

Type-of-defect keyword (required):

LOOP	Use if symptom was a loop
WAIT	Use if symptom was a wait
INCORROUT	Use for defective user output
PERFM	Use for unsatisfactory performance
DOC	Use for documentation errors
MSGxxxxxxxx	Replace xxxxxxxx by error message number
ABENDxx	Replace xx by utility abend number
ABENDxxx	Replace xxx by system abend number

Procedure-that-failed keyword (required):

The EBCDIC name of each procedure is coded at the top of the procedure so that the beginning of the procedure is identifiable in a dump. If the point of failure follows the EBCDIC name of a procedure, use that name as the keyword. For DOS/VS procedure names, replace "CIPO" with "CIPD".

CIPOSCAN (scan-in-process bit is not zero)
CIPOPRT
CIPOMSG
CIPOTABL
CIPORITE
CIPOFCB
CIPOTABLE
CIPOCOPY
CIPOGRAF
CIPODEFN
CIPORITE

Subprocedure-that-failed Keywords (not required):

Get the number in the active operation field, ACTIVOP, of the work area. Use the subroutine name (below) corresponding to this number. The work area address is in register 8. The field ACTIVOP is at offset 32 (20) from that address. The field NEXTOP is at offset 33 (21).

INCLUDE	use if ACTIVOP=00 and NEXTOP= 07
NAME	use if ACTIVOP= xx and NEXTOP= 01
FCB	use if ACTIVOP= 04
TABLE	use if ACTIVOP= 06
GRAPHIC	use if ACTIVOP= 05
DEFINE	use if ACTIVOP= 03
QDEFINE	use if ACTIVOP= 09
COPYMOD	use if ACTIVOP= 02
OPTION	use if ACTIVOP= 00 and NEXTOP= 08

Resource-involved Keywords (not required):

Use only if resource was involved and was altered by user.

Register 1 points to the DTF or DCB of a failing device; the EBCDIC name of the device can be found in the DTF for DOS/VS or the TIOT for OS/VS—offset indicated in DCB.

SYSLST	(DOS only)
SYSIPT	(DOS only)
SYSPCH	(DOS only)
SYSLNK	(DOS only)
SYSxxx	(DOS only)
SYSIN	(OS only)
SYSPRINT	(OS only)
SYSSNAP	(OS only)
SYSUT1	(OS only)
SYSLIB	(OS only)

Snap and Dump Control Statements

You can use the SNAP and DUMP control statements to help diagnose a failure. These control statements are explained throughout the Offline IBM 3800 Utility manual.

SNAP=(m,n) will cause a dump of the work area for DOS/VS and a dump of the user computing space (no system space) for OS/VS.

DUMP=(m,n) will cause an ABEND dump for either system.

INDEX

LY20-8058-0

Offline IBM 3800 Printer Utility Logic Printed in U. S. A. LY20-8058-0

IBM®

Offline IBM 3800 Printer
Utility Logic
LY20-8058-0

Reader's Comment Form

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate. Comments may be written in your own language; use of English is not required.

IBM shall have the nonexclusive right, in its discretion, to use and distribute all submitted information, in any form, for any and all purposes, without obligation of any kind to the submitter. Your interest is appreciated.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

Fold on two lines, tape, and mail. No postage necessary if mailed in the U.S.A. (Elsewhere, any IBM representative will be happy to forward your comments.) Thank you for your cooperation.

LY20-8058-0

Reader's Comment Form

Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

IBM Corporation
P.O. Box 50020
Programming Publishing
San Jose, California 95150

Fold and Tape

IBM®

LY20-8058-0

Offline IBM 3800 Printer Utility Logic Printed in U. S. A. LY20-8058-0