Technical Memorandum No. 102-III

03674-23-M

CPS PROGRAM LOGIC MANUAL

Volume III

CPS 8-K FORTRAN PACKAGE

by

G. N. Cederquist
K. Metzger

Approved by: Theodore G Birdsall

for

COOLEY ELECTRONICS LABORATORY
Department of Electrical Engineering
The University of Michigan
Ann Arbor, Michigan

Contract No. Nonr-1224(36)
NR187-200
Office of Naval Research
Department of the Navy
Washington, D. C. 20360

December 1969

PREFACE

TO THE SERIES

This report of four volumes is intended to document the May 21, 1969 version of the Cooley Programming System (CPS) which was developed at the Cooley Electronics Laboratory of The University of Michigan. The four volumes are titled:

Volume 1: CPS System Architecture and Conventions

Volume 2: CPS Basic Programming Package

Volume 3: CPS FORTRAN Package

Volume 4: CPS System Utility Programs

The four volumes were written in order to take a snapshot of CPS at one point in its continuing development. This version of CPS is considered to be a first generation system; successive versions are on the drawing boards and internally resemble their parent less and less every day.

CPS is a generalized programming and file management system written for use on the PDP-8 processor of Digital Equipment Corporation's LINC-8 computer. A minimum memory size of 8192 words is required. Extensive use is made of the two tape units present on every LINC-8 for both file storage and system residence.

Using CPS, programs can be entered, edited, assembled (or compiled), loaded and executed entirely from the keyboard without the use of paper tape. CPS provides power and flexibility normally only found on larger computers and in fact was modeled after the Michigan Terminal System which operates on an IBM SYSTEM/360 model 67.

In addition to a comprehensive file management and control system CPS contains:

- Symbolic Text Editor
- MACRO-8 Assembler
- SABR Assembler
- 8-K FORTRAN Compiler
- Two loaders
- Various utility programs

Each of the above programs contains service routines which permit automatic communication with the central file system and which allow direct access to CPS files. The general policy followed in implementing CPS was to borrow and adapt as much of DEC's software as possible in order to speed system development.

The responsibility (or blame) for various segments of CPS is divided as follows:

Gerald Cederquist System design and conventions, Control Program, Absolute Assembler, Absolute Loader, MARKP8 (tape marking program), and FILE-COPY (file copying program).

Kurt Metzger	SABR, FORTRAN, Relocating Loader, PAPERBIN (binary paper tape input program), TAPCOPY (tape copying program), and assorted tape routines.
joint effort	Text Editor, I/O Control System, and various compromises.

Work started on CPS in November of 1968 with the first workable version being completed in February of 1969. The FORTRAN-SABR package was incorporated in the March-April period of 1969. Since this time CPS has been in use at CEL in the development of digital signal processing programs for project MIMI. It has been found to be a very effective tool and has greatly decreased program development time and programmer frustration. Tasks which formerly took over a month to complete using the DEC 8-LIBRARY System are now routinely completed in one to two weeks.

The bulk of CPS and the associated routines were hurriedly written since the authors were effectively stealing time from their thesis research. Consequently portions of the code were done in a quick and dirty manner. Now that several months have passed, the fact that these portions were quick has dimmed in memory but the dirt remains.

The authors would like to thank Dr. T. G. Birdsall of CEL for his continued encouragement and support and Mr. C. Conley of DEC for his assistance in providing the FORTRAN-SABR package for use in CPS.

G. Cederquist
K. Metzger

Ann Arbor, Michigan
December 1969

TABLE OF CONTENTS

PREFACE

TO VOLUME 3

This volume describes the modifications and support routines which were added to DEC's 8-K FORTRAN system in order to allow it to interface with CPS. The net effect of these alterations was to completely eliminate the need for paper tape and to allow 8-K FORTRAN to function as an integral part of CPS.

The current version of 8-K FORTRAN as part of an operating system is the third such implementation made at CEL. The first such system was completed in August of 1968 and was a rudimentary two tape system. The unit 0 tape was a slightly modified 8-Library System tape and the unit 1 tape contained a CEL written file management and operating system. This was used to store, load and execute FORTRAN programs. The need for paper tape was completely eliminated. The second system never got off its feet because it contained an early (non-working) version of SABR. By the time SABR was fixed CPS was on the air.

Using CPS as a programming aid it was possible to imbed DEC's 8-K FORTRAN into CPS in less than a month. The implementation of the relocatable loader required the greatest amount of time since its support code was the largest and most complicated portion of the FORTRAN package. In putting 8-K FORTRAN into CPS the primary emphasis was on rapidly getting a working system. Since implementing 8-K FORTRAN new programming and tape handling tricks have been learned and it is recognized that some of the procedures and code can be improved.

8-K FORTRAN has been found to be a very useful programming tool and is being extensively used at CEL to post process the output of various signal processing programs.

CHAPTER 1

CPS FORTRAN COMPILER

The FORTRAN compiler used in CPS is Version 04 of DEC's 8-K FORTRAN compiler. We have patched 26 locations and supplied additional support code in order to interface it with the CPS file system. The compiler is stored in the coreimage file *FORTRAN. *FORTRAN accepts input either from designated symbolic files or from the ASR-33 Teletype. Its output is placed into the -S working area. The *FORTRAN output is in two sections and each half is placed into a separate file. If the compilation is successful the SABR assembler is automatically loaded into memory and is given the task of assembling the *FORTRAN output. The resulting binary is placed in the -B working area.

Operation proceeds in the following manner:

1. The user types the command

```
RUN *FORTRAN FILE1+...+FILE6 P=...
```

into the CPS file system. The files FILE1 through FILE6 are assumed to contain a valid FORTRAN program with the last named file being terminated with the standard FORTRAN statement, END. A maximum of 6 files can be compiled as a single FORTRAN program. If no file names are supplied the ASR-33 is assumed to be the active input device.

2. The CPS file system loads the coreimage file *FORTRAN and starts it at location 15600 (in the support code). The following operations take place before control is passed to the compiler itself:

> The communication area (CA) file update switches are set. However, no file entries are made at this time.
>
> The input file list is checked to see if the first entry is -3 which indicates that the ASR-33 is to be used. If it is, RDRSW is set to 0, no file transfer operations are made and the compiler is started at location 11000.
>
> If the input is to be from CPS symbolic files then these files are transferred into the binary working area starting in block 1 on tape unit 1. This transfer is made so that the *FORTRAN output can be placed into -S for later use with SABR. This transfer is fairly fast and cuts down on later tape operations. The subroutine TRANS is used in this transfer. The transfer is effected using a 5 page buffer. During the transfer process the appropriate modifications are made in the CA in order to reflect the change in file locations. The original files are unmodified. When all transfers have been made (a maximum of 6) the SETUP routine is called. This routine is used to setup the input tape calls and handles the sequencing through input files. After the tape calls have been initialized the compiler is started at location 11000.

3. The compiler requests an input character by doing a JMS to the FEEDF subroutine located at 13315. This routine then obtains a character either from the reader using the RKBD routine (which forces the parity channel to 1 so we can use a dataphone for input) or from the specified input files using the DCODE subroutine.

The DCODE routine accesses the input data files through a heirarchy of subroutines, BITS, GETWOR, and SETUP. It is assumed that the input files use the *ED compressed data format. DCODE supplies each carriage return with a line feed.

A two page input buffer located at 15200 is used for input.

When a CTRL-C is encountered in the input string (the *ED end of file symbol) the input support routines cause the next user named file to be used for input. If there are no more files (i.e., END statement) an error message is typed and control is returned to the CPS file system.

4. Each time the compiler outputs a character it does a JMS to the support routine FWR at 13200. This routine accepts ASCII from the compiler, using the COMPRES routine packs it into the *ED format and places the result into the output buffer (4 pages long starting at 13600). If a (CTRL-C) has been struck at the keyboard, the compiler output is echoed on the ASR-33. This can be turned off by striking any other character.

COMPRES converts the ASCII output into *ED format and uses PUTOUT to form the packed word. Line feeds are ignored. PUTOUT uses the routine PLACE to place a packed word in the actual buffer. When the output buffer is full, WBUF is used to write it onto tape and to reset the buffer pointers.

There is approximately an 8 to 1 expansion in file size between input and output.

5. When the END statement is found by the compiler it does a JMS to the routine FMID at 13263. This routine terminates the current buffer with a (CTRL-C) character and writes it onto tape. FMID then sets up file pointers in the CA which SABR will later use to determine where its input is located. Two file entries are made. The file just formed on tape is set up as the second file entry. The first file entry points to the file to be formed next using the output to be produced when the compiler is re-entered.

6. The compiler is re-entered and produces additional output. This is buffered and written onto tape as before. However a new file area is used. When the compiler finishes this operation it returns control of the machine to the support routine, END at 11635. Compilation is now finished.

7. The END routine terminates the current buffer with a (CTRL-C) and writes the buffer onto tape. The appropriate entry is made in the CA file pointer list. If any errors have occurred during the compilation the file system is re-entered. If there were no errors then a small program is transferred to memory field 0 and started. This program uses the system boot read only tape routine to load the 1 page file system RUN loader. Pointers have been set up so that when the RUN loader is started it proceeds to load SABR and start it. The file pointers in the CA have been set so that the two FORTRAN output

segments are taken as SABR's input and operation proceeds as described in the SABR write-up. The P=... parameters supplied in the RUN *FORTRAN command are passed on to SABR.

Memory Organization

The upper 4-K of memory is organized as indicated below. The lower 4-K contains only patches.

Locations	Contents
11600-11777	SETUP, RKBD, END, LDSABR
12000-12177	COMPRES, PUTOUT, DCODE, BITS
12200-12336	DECOUT, MESAGE, GCDF
13200-13377	FWR, PLACE, FMID, FEEDF, GETWOR
13600-14577	output buffer
15000-15177	RTAPE, WTAPE
15200-15577	input buffer, TRANS (15400 .. gets wiped)
15600-15777	START, SETRDR, NOROOM (all get wiped)
16400-17577	transfer buffer (in the symbol table)

Subroutine Descriptions

DECOUT A decimal output routine ... not used.

MESAGE A modified version of DEC's message typeout routine. The AC is assumed to contain the address of the text to be typed.

GCDF This subroutine does a RDR with a CDF in the AC and sets the data field to 1. Return is made with a data field resetting CDF in the AC.

START The entry point to the FORTRAN support code. The file system starts *FORTRAN here. This routine first sets up the CA update switches, then checks for the proper input device. If the ASR-33 is to be used, a call is made to SETRDR. If the input is to come from CPS files these files are transferred to -B using TRANS. Minus the number of input files is placed in GCNT and the file list pointer TA is set to 7600 (this initializes SETUP). The subroutine SETUP is called in order to initialize the input buffer management routines so that the first compiler input request causes the first input file to be accessed. The compiler is then started at location 11000.

SETRDR Sets RDRSW to 0 which indicates that the input is to come from the ASR-33.

NOROOM Uses MESAGE to type an error message if the files transferred into -B require too much space.

SETUP This routine is called whenever an input file has been emptied and an END statement has not been encountered. The file pointer list is checked to see if another file was named in the invoking RUN command. If there is another file, the input buffer routines are reset so that a read tape operation is forced when the next character is requested. If a file is not present then an error message is typed and control is returned to the file system. RL is set to 7777 and HALF is set to 0 to force the desired tape read.

RKBD Reads one character from the keyboard setting the parity channel to 1.

END Control comes to here when the compiler is all through. The current output buffer is terminated with a (CTRL-C) (a 7703 is placed into the buffer on the assumption that the last line was terminated with a carriage return) and the buffer is written on tape by calling WBUF. The length of the first entry in the CA file pointer list is calculated and placed in this list. If there were any errors in the compilation an error message is typed and control is returned to the file system. If there were no errors then the address of SABR (assumed to be the next file after *FORTRAN and whose address is thus in location 17773) is placed in location 00134 and a 0 is placed in 00135 for use by the RUN loader. A 6 instruction program is transferred to location 07372 and then started. This program simply makes a call to the read-only tape routine contained in the system boot. This call causes the RUN loader to be read into memory and started at location 07400. The RUN loader then reads in SABR and starts it.

TRANS Used to transfer the user designated files from their normal storage areas into -B. The original files are unaltered. The transfer is made using a fixed buffer size of 5 blocks. This is fine if the checksum is correct in the 1 to 4 tape blocks following a specified file. This routine should be changed so that only the exact number of blocks in a file are transferred. RTAPE and WTAPE are called on for the actual tape operations. TRANS is only used at startup and is never used later. It eventually gets clobbered.

FWR Accepts ASCII from the compiler and calls COMPRES to process it. If a (CTRL-C) has been struck on the ASR-33, FWR types the compiler output. Striking any other character turns this typing off.

PLACE Takes the contents of the AC and places them into the output buffer. LOC is the buffer pointer and is assumed to always be pointing to a valid buffer location. After the contents of the AC have been placed, LOC is incremented by one and tested to see if the next value is a proper buffer address. If it is a normal return is made. If LOC points outside of the buffer then a call is made to WBUF which writes the current buffer onto tape, advances the tape pointer and resets LOC. A normal return is then made.

WBUF Writes the current buffer onto tape and resets LOC to point at the first word in the output buffer. WBUF starts writing tapes at block 1 of unit 0. WTAPE is used for the actual tape operation.

FMID Control is returned to here when the compiler realizes it has found the END statement and it is at "mid-pass." The current output buffer is terminated with a 7703 [(CTRL-C), assuming the last stored character was a carriage return.] The buffer is then written onto tape. The file pointers in the CA are set up with two file engries being constructed. The just completed output segment is set up as the second entry. The output segment to follow (storage allocations) will be used as the first entry. Control is returned to the compiler.

FEEDF The compiler comes here seeking ASCII. If the RDRSW is 0 then RKBD is used to obtain the ASCII. Otherwise a call is made on DCODE.

GETWOR This routine is used by BITS to fetch a packed word from the input buffer. RL is the input buffer pointer and is checked prior to fetching a word to see if it points into the current buffer. If it does, the next word is fetched and RL is incremented. If it doesn't, a call is made to RTAPE to read in two additional tape blocks (two blocks are always read ... this shouldn't cause problems), RL is reset to point to the start of the input buffer and the next word is fetched in the normal fashion.

COMPRES Takes ASCII from the compiler and determines how it should be coded in order to place it into *ED format. PUTOUT is used to do the required packing and to cause the packed output to be placed into the output buffer. Line feeds are ignored.

PUTOUT Strips the contents of the AC to six bits and depending on RLSW places the result in the left or right hand side of a computer word. If it goes into the right hand side, the result is placed into the output buffer through a call on PLACE.

DCODE Obtains six bit characters using the BITS routine and maps these back into ASCII assuming the packed input uses *ED format. A line feed is supplied after every carriage return. If a (CTRL-C) is encountered a call is made to SETUP to set up the next input file and DCODE is then restarted.

BITS — Furnishes DCODE with 6 bit words for use in reconstructing the source ASCII. Uses GETWOR to obtain its packed words and uses HALF as a switch to determine which side of a word should be used next.

RTAPE
WTAPE — A one page tape routine for reading and writing 128 word blocks on LINC tape. See the enclosed listing for details.

Assembly Instructions

The following procedure is used in constructing the core-image file *FORTRAN.

```
RUN *ASM C1+C2+C3+C4+C5+C6 2=C7+$

SAVE -B FORTSUP

RUN *ASM F1TAPE

SAVE -B BF1TAPE

LOAD A

GET FORTV4

GET FORTSUP

GET BF1TAPE @15000

BUILD   (the starting address is 15600)
```

When control returns to the file system, -B will contain the desired coreimage file.

Files C1 through C7 contain the symbolics for the patches and the support code required to interface the compiler with CPS. F1TAPE is a one page LINC tape read-write routine and is the only tape routine used other than the one in the system boot. FORTV4 is a binary file which contains DEC's FORTRAN compiler, unmodified. FORTV4 is generated by reading the source binary paper tape into a file using PAPERBIN.

FORTRAN Highlights

The source files named in the invoking RUN command are moved into -B.

Compilation proceeds from -B into -S.

Two output files are produced. These are set up so that SABR operates on them in the correct order.

SABR immediately follows the FORTRAN compiler on tape and the loader provides the location of SABR in the CPS file system.

The RUN loader is accessible to the user via the boot read only tape routine.

Input and output files use the *ED format.

All line feeds are ignored on output.

The input text is contained in 6 or less files.

The standard CPS conventions for the system communication area are observed.

The parameters specified in the P=... construct are passed directly to SABR on pass 2.

If no input files are specified in the invoking RUN command, the ASR-33 is used as the active input device.

The input buffer is two pages long. The output buffer is four pages long.

```
                    /FIELD 0 PATCHES ON THE COMPILER
                    /
                    FIELD 0
                    /
                    *352
0352   7000           NOP /INIT START PATCH
0353   7000           NOP
0354   7200           CLA
                    /
                    *357
0357   5177           5177 /GO TO II
                    /
                    *547
0547   6212           CIF 10 /READER PATCH
0550   4751           JMS I .+1
0551   3315           FEEDF
                    /
                    *5340
5340   6212           CIF 10 /PUNCH PATCH
5341   4742           JMS I .+1
5342   3200           FWR
                    /
                    *5355
5355   7000           NOP /FINAL END PATCH
5356   7200           CLA
5357   6212           CIF 10
5360   4761           JMS I .+1
5361   1635           END
                    /
                    *3163
3163   7200           CLA /MIDPASS FIX
3164   6212           CIF 10
3165   4766           JMS I .+1
3166   3263           FMID
3167   7000           NOP
                    /
                    *7170
7170   6046           TLS /CHANGING SEQ OF TLS TSF IN ERROR ROUTINE
7171   6041           TSF .
7172   5371           JMP .-1
                    /
                    FIELD 1
                    /
                    *1000
1000   7000           NOP /REAL START
1001   7200           CLA
1002   6046           TLS /CHATTER
                      PAGE
```

```
                FIELD 1
                /
                /UTILITY ROUTINES FOR OUTPUT
                /DECIMAL PRINT AND MESAGE
                /
                *2200
                /
2200    0000    DECOUT, 0 /AC=NUM TO TYPE, 0 TO 4097
2201    3243      DCA DTEMP /SUPPRESSES LEADING 0'S
2202    1233      TAD DECFIX /TAD INSTR
2203    3212      DCA DECFXL
2204    1241      TAD M4DEC /MAX OF 4 DIGITS
2205    3240      DCA DECCNT
2206    7410      SKP /IGNORE NXT INS
2207    3243    DECLPA, DCA DTEMP
2210    1243      TAD DTEMP
2211    7100      CLL /USE LINK FOR OVRFLW TST
2212    1234    DECFXL, TAD MTENS
2213    7420      SNL /L=0 RES WAS NEG
2214    5217      JMP DECHAV /HAVE DIGIT
2215    2242      ISZ DECVAL
2216    5207      JMP DECLPA
2217    7300    DECHAV, CLA CLL
2220    1242      TAD DECVAL
2221    7450      SNA /SUP LDNG 0'S
2222    5225      JMP .+3
2223    1244      TAD DEC260
2224    4245      JMS DECTYP
2225    7300      CLA CLL
2226    3242      DCA DECVAL /ZERO NXT DGT
2227    2212      ISZ DECFXL /DIVIDE BY 10
2230    2240      ISZ DECCNT /DONE?
2231    5210      JMP DECLPA+1 / NO
2232    5600      JMP I DECOUT /GO HOME
                /
2233    1234    DECFIX, TAD MTENS
2234    6030    MTENS, -1750 /KEEP IN ORDER, -1000
2235    7634      -144
2236    7766      -12
2237    7777      -1
2240    0000    DECCNT, 0
2241    7774    M4DEC, -4
2242    0000    DECVAL, 0
2243    0000    DTEMP, 0
2244    0260    DEC260, 260
                /
2245    0000    DECTYP, 0
2246    6046      TLS
2247    6041      TSF
2250    5247      JMP .-1
2251    5645      JMP I DECTYP
                /
                /
2252    0000    MESAGE, 0 /AC=ADDR OF TEXT
```

```
PAGE 03

2253  1335     TAD MESN1 /-1
2254  3010     DCA 10 /USES AUTO-INDEX
2255  1410     TAD I 10
2256  3267     DCA MSRGHT /SAVE PACKED WORD
2257  1267     TAD MSRGHT
2260  7012     RTR
2261  7012     RTR
2262  7012     RTR
2263  4270     JMS TYPECH /TYPE LH
2264  1267     TAD MSRGHT
2265  4270     JMS TYPECH /TYPE RH
2266  5255     JMP MESAGE+3
2267  0000   MSRGHT, 0 /TEMP
2270  0000   TYPECH, 0 /TYPES
2271  0326     AND MASK77
2272  7450     SNA /0 TERMINATES
2273  5652     JMP I MESAGE /RETURN
2274  1327     TAD MESN40 /-40
2275  7500     SMA /<40?
2276  5301     JMP .+3
2277  1330     TAD C340MES /340
2300  5314     JMP MTP
2301  1331     TAD M3MES /-3
2302  7440     SZA /LF?
2303  5306     JMP .+3
2304  1332     TAD C212MES
2305  5314     JMP MTP
2306  1336     TAD M2MES /-2
2307  7440     SZA /CR?
2310  5313     JMP .+3
2311  1333     TAD C215MES
2312  5314     JMP MTP
2313  1334     TAD C245MES
2314  4245   MTP, JMS DECTYP
2315  7200     CLA
2316  5670     JMP I TYPECH /RET
             /
2317  0000   GCDF, 0
2320  7200     CLA
2321  6214     RDF
2322  1325     TAD C6201
2323  6211     CDF 10
2324  5717     JMP I GCDF
             /
2325  6201   C6201, 6201
             /
2326  0077   MASK77, 77
2327  7740   MESN40, -40
2330  0340   C340MES, 340
2331  7775   M3MES, -3
2332  0212   C212MES, 212
2333  0215   C215MES, 215
2334  0245   C245MES, 245
2335  7777   MESN1, -1
2336  7776   M2MES, -2
```

PAGE

```
              FIELD 1
              /
              /CODE TO START UP THE COMPILER
              /
              *5600
5600   6036   START, KRB /CLR FLAG
5601   7300     CLA CLL
5602   1377     TAD (7773 /FIX UP POINIERS IN TOP PAGE
5603   3010     DCA 10
5604   1376     TAD (7402
5605   7041     CIA
5606   3410     DCA I 10
5607   7240     CMA CLA
5610   3410     DCA I 10
5611   7201     CLA IAC
5612   3410     DCA I 10
5613   1375     TAD (7400
5614   3410     DCA I 10
5615   1374     TAD (-6 /MAX NUMBER OF FILES ALLOWED
5616   3773     DCA GCNT
5617   1372     TAD (4001 /START OF -3
5620   3254     DCA PWHERE
5621   1371     TAD (7600
5622   3277     DCA TAP /PTS TO FILE DATA IN TOP PAGE
5623   1370     TAD (3 /SEE IF READER IPT
5624   1677     TAD I TAP /-3=RDR
5625   7650     SNA CLA
5626   5274     JMP SETRDR
5627   1677   TRLOOP, TAD I TAP /GET BLN
5630   7450     SNA
5631   5263     JMP DONE
5632   3252     DCA GBLK /SET UP TO MOVE TO UNIT1
5633   1254     TAD PWHERE /NEW LOC ON TAPE
5634   3677     DCA I TAP /NEW BLN
5635   2277     ISZ TAP /GET # BLKS
5636   1677     TAD I TAP
5637   0367     AND (77 /GET RID OF SOURCE SINK BIT
5640   7450     SNA /SHOULD NOT =0
5641   5263     JMP DONE
5642   3253     DCA GN /GET THIS MANY
5643   1254     TAD PWHERE /SEE IF ROOM
5644   0366     AND (1777
5645   1253     TAD GN
5646   1365     TAD (-150
5647   7700     SMA CLA
5650   5300     JMP NOROOM /ERROR MESS
5651   4764     JMS TRANS /XFER IT
5652   0000   GBLK, 0
5653   0000   GN, 0
5654   0000   PWHERE, 0
5655   1254     TAD PWHERE /ADV POINTER
5656   1253     TAD GN
5657   3254     DCA PWHERE
5660   2277     ISZ TAP /NEXT ENTRY
```

```
PAGE 06

5661  2773    ISZ GCNT /6 XET ?
5662  5227    JMP TRLOOP /NO
5663  1371  DONE, TAD (7600 /DONE
5664  3763    DCA TA /FOR USE WHEN COMPILING
5665  1362    TAD (6 /HOW MANY?
5666  1773    TAD GCNT /# IN AC
5667  7041    CIA
5670  3773    DCA GCNT
5671  4761    JMS SETUP /SET UP INPUT TAPE CALLS
5672  5673  STARTC, JMP I .+1
5673  1000    1000 /FORTRAN ENTRY PT
            /
5674  7300  SETRDR, CLA CLL
5675  3760    DCA RDRSW
5676  5272    JMP STARTC /GET GOING
            /
5677  0000  TAP, 0
            /
5700  1303  NORCOM, TAD ML2
5701  4757    JMS MESAGE
5702  5756    JMP ENERT
            /
5703  5704  ML2, M2A
5704  4543  M2A, TEXT :%#
5705  2417  TO
5706  1740  O
5707  1525  MU
5710  0310  CH
5711  4011   I
5712  1620  NP
5713  2524  UT
5714  0000  :
5756  1707    PAGE
5757  2252
5760  3357
5761  1600
5762  0006
5763  1634
5764  5400
5765  7330
5766  1777
5767  0077
5770  0003
5771  7600
5772  4001
5773  1633
5774  7772
5775  7400
5776  7402
5777  7773
```

```
              /TAPE BUFFER CHANGER,RKBD,ERRORMESS,END
              /
              FIELD 1
              /
              *1600
1600  0000    SETUP, 0 /SET UP TAPE UNIT CALLS
1601  7300      CLA CLL
1602  1233      TAD GCNT
1603  7650      SNA CLA
1604  5302      JMP TOMANY
1605  1634      TAD I TA /BLN
1606  0377      AND (1777
1607  3776      DCA ITAB
1610  1634      TAD I TA /GET UNIT
1611  7006      RTL
1612  0375      AND (1
1613  3774      DCA IUN /INPT CALL UNIT
1614  7240      CLA CMA
1615  3773      DCA RL /FORCE INITIAL READ
1616  2233      ISZ GCNT /ONE LESS BUFF
1617  7000      NOP
1620  3772      DCA HALF /SW FOR SIDE TO GET FROM
1621  2234      ISZ TA
1622  2234      ISZ TA /IGNORE BLK CNT
1623  5600      JMP I SETUP
              /
              /
1624  0000    RKBD, 0
1625  6031      KSF
1626  5225      JMP .-1
1627  6036      KRB
1630  0371      AND (177
1631  1370      TAD (200
1632  5624      JMP I RKBD
              /
              /
1633  0000    GCNT, 0
1634  0000    TA, 0
              /
              /
1635  0000    END, 0 /COME HERE WHEN ALL DONE
1636  6213      CDF CIF 10
1637  7300      CLA CLL
1640  1767      TAD LOC
1641  3234      DCA TA
1642  1366      TAD (7703
1643  3634      DCA I TA
1644  4765      JMS WBUF /PUT OUT REST OF BUFFER
1645  1747      TAD I L7600 /FIX BL LEN
1646  7041      CIA
1647  1764      TAD BLK
1650  3750      DCA I L7601 /HAVE PROPER LEN
1651  6201      CDF
1652  1751      TAD I L75 /SEE IF ERROR SW=1
```

```
1653  6211     CDF 10
1654  7640     SZA CLA
1655  5307     JMP ENERT
              /
              /NOW LOAD SABR
              /
1656  1752     TAD I L7773 /ADDRES OF SABR
1657  6201     CDF
1660  3745     DCA I L134
1661  3746     DCA I L135 /0 LENGHT TO FOOL ROUTINE
1662  1363     TAD (7371 /*7372
1663  3010     DCA 10    /7621
                         /JMS I .-1
1664  1362     TAD (7621 /450
1665  3410     DCA I 10  /1
1666  1361     TAD (4772 /0
1667  3410     DCA I 10  /7400
1670  1360     TAD (450
1671  3410     DCA I 10
1672  7201     CLA IAC
1673  3410     DCA I 10
1674  3410     DCA I 10
1675  1357     TAD (7400
1676  3410     DCA I 10
1677  6203     CDF CIF
1700  5701     JMP I .+1
1701  7373     7373
              /
              /
              /
1702  1314  TOMANY, TAD ML1 /NO END STATEMENT
1703  4756    JMS MESAGE
1704  5307    JMP ENERT
1705  1324  TOFULL, TAD ML3
1706  4756    JMS MESAGE
1707  7200  ENERT, CLA
1710  1336    TAD ML4
1711  4756    JMS MESAGE
1712  6203    CDF CIF
1713  5747    JMP I L7600
              /
1714  1715  ML1, M1A
1715  4543  M1A, TEXT :%#
1716  0516  EN
1717  0440  D
1720  2324  ST
1721  1516  MN
1722  2477  T?
1723  0000  :
1724  1725  ML3, M3A
1725  4543  M3A, TEXT :%#
1726  1725  OU
1727  2420  TP
1730  2524  UT
1731  4024   T
```

```
PAGE 09

1732  1717  OO
1733  4022   R
1734  1107  IG
1735  0000  :
1736  1737  ML4, M4A
1737  4543  M4A, TEXT :%#
1740  0522  ER
1741  2217  RO
1742  2240  R
1743  2205  RE
1744  2400  T:
1745  0134  L134, 134
1746  0135  L135, 135
1747  7600  L7600, 7600
1750  7601  L7601, 7601
1751  0075  L75, 75
1752  7773  L7773, 7773
1753  2252    PAGE
1757  7400
1760  0450
1761  4772
1762  7621
1763  7371
1764  3250
1765  3240
1766  7703
1767  3262
1770  0200
1771  0177
1772  2143
1773  3355
1774  3343
1775  0001
1776  3341
1777  1777
```

```
               /TRANSFER UTILITY
               /
               FIELD 1
               /
               *5400
5400  0000     TRANS, 0
5401  7300       CLA CLL
5402  1600       TAD I TRANS
5403  0377       AND (1777 /GET BLN
5404  3251       DCA FBLK /FROM
5405  1600       TAD I TRANS /GET UNIT
5406  0376       AND (4000
5407  7106       RTL CLL
5410  3253       DCA FUNT /F UNIT
5411  2200       ISZ TRANS
5412  1600       TAD I TRANS
5413  3271       DCA AMT /THIS MANY BLKS
5414  2200       ISZ TRANS
5415  1600       TAD I TRANS
5416  0377       AND (1777
5417  3256       DCA TOBLK /TO HERE
5420  1600       TAD I TRANS
5421  0376       AND (4000
5422  7106       RTL CLL
5423  3260       DCA TOUNT /TO UNIT
5424  2200       ISZ TRANS /FOR RET
5425  1252       ~~TAD FNUM~~
5426  3272       DCA NUM
5427  7200     SCRND, CLA
5430  1271       TAD AMT
5431  7650       SNA CLA
5432  5600       JMP I TRANS /ALL DONE
5433  1272       TAD NUM
5434  7040       CMA
5435  1271       TAD AMT
5436  7510       SPA /S IF FULL AMT TOGO
5437  5242       JMP LESS
5440  7001       IAC
5441  5245       JMP LST
5442  7200     LESS, CLA
5443  1271       TAD AMT
5444  3272       DCA NUM
5445  3271     LST, DCA AMT /RDMAINING
5446  1272       TAD NUM
5447  3257       DCA TONUM
5450  4775       JMS RTAPE
5451  0000     FBLK, 0
5452  0005     FNUM, 5
5453  0000     FUNT, 0
5454  6400       6400 /START OF BUFFER
5455  4774       JMS WTAPE
5456  0000     TOBLK, 0
5457  0000     TONUM, 0
5460  0000     TOUNT, 0
```

TAD (5
DCA FNUM
TAD (5

TAD AMT
DCA FNUM

```
PAGE 11

5461  6400     6400 /BUFF
5462  1251     TAD FBLK
5463  1252     TAD FNUM
5464  3251     DCA FBLK
5465  1256     TAD TOBLK
5466  1257     TAD TONUM
5467  3256     DCA TOBLK
5470  5227     JMP GORND
             /
5471  0000   AMT, 0
5472  0000   NUM, 0
5574  5006     PAGE
5575  5000
5576  4000
5577  1777
```

```
                /CODE TO SUPPORT COMPILER DURING EXECUTION
                /
                FIELD 1
                /
                *3200
                /
3200   0000     FWR, 0 /TAKES ASCII FROM FORTRAN
3201   3356       DCA IPT
3202   7004       RAL /SAVE LINK
3203   3223       DCA TLINK
3204   4777       JMS GCDF /GET CDF AND DO CDF 10
3205   3215       DCA FWRRET
3206   2200       ISZ FWR /MISS PATCH
3207   1356       TAD IPT
3210   4776       JMS COMPRES /PACKIT
3211   6034       KRS /TYPER FOR DEBUGGING, CONT C TURNS ON, ANYTHING OFF
3212   1375       TAD (-203
3213   7650       SNA CLA
3214   5224       JMP PRNT
3215   0000     FWRRET, 0
3216   6202       CIF
3217   7300       CLA CLL
3220   1223       TAD TLINK /RESTORE LINK
3221   7010       R@R
3222   5600       JMP I FWR /GO BACK
                /
3223   0000     TLINK, 0
                /
3224   1356     PRNT, TAD IPT
3225   4774       JMS DECTYP
3226   5215       JMP FWRRET /CLEARS ON RET TO FIELD 0
                /
                /
3227   0000     PLACE, 0 /PUT IN BUFF
3230   3662       DCA I LOC /C(AC)
3231   2262       ISZ LOC
3232   1262       TAD LOC
3233   7100       CLL /OVERFLOW SETS TO 1
3234   1373       TAD (-4600
3235   7630       SZL CLA /L=1 FULL
3236   4240       JMS WBUF /W IT
3237   5627       JMP I PLACE
                /
3240   0000     WBUF, 0 /WRITE OPT BUFF
3241   7300       CLA CLL
3242   1250       TAD BLK
3243   1251       TAD BLK+1
3244   1372       TAD (-256 /TOO FULL?
3245   7700       SMA CLA
3246   5771       JMP TOFULL
3247   4770       JMS WTAPE
3250   0001     BLK, 1
3251   0004       4
3252   0000       0
```

```
PAGE 13

3253  3600  LOCI, 3600 /STRT OF OPT BUFF
3254  1250    TAD BLK
3255  1251    TAD BLK+1
3256  3250    DCA BLK
3257  1253    TAD LOCI
3260  3262    DCA LOC
3261  5640    JMP I WBUF
            /
3262  3600  LOC, 3600 /FOR START UP
            /
3263  0000  FMID, 0 /MIDPASS ALTERATION
3264  4777    JMS GCDF
3265  3312    DCA FMDRET
3266  2263    ISZ FMID /MIS PATCH
3267  1367    TAD (7703
3270  3662    DCA I LOC /TERM BUFFER
3271  4240    JMS WBUF /W IT
3272  1010    TAD 10
3273  3777    DCA GCDF /USING AS A TEMP
3274  1366    TAD (7577
3275  3010    DCA 10 /SET UP BLK CALLS FOR SABR
3276  1250    TAD BLK
3277  3410    DCA I 10
3300  3410    DCA I 10 /WILL FIX LATER WHEN ENDING
3301  7201    CLA IAC
3302  3410    DCA I 10
3303  7240    CLA CMA
3304  1250    TAD BLK
3305  3410    DCA I 10
3306  3410    DCA I 10 /ZERO FOLLOWING ENTRIES
3307  3410    DCA I 10
3310  1777    TAD GCDF
3311  3010    DCA 10 /RESTORE FOR COMPILER
3312  7000  FMDRET, NOP
3313  6202    CIF
3314  5663    JMP I FMID
            /
3315  0000  FEEDF, 0 /FEED COMP ASCII
3316  2315    ISZ FEEDF /MISS PATCH
3317  4777    JMS GCDF
3320  3325    DCA FDRET
3321  1357    TAD RDRSW /SEE IF RDR
3322  7650    SNA CLA /0=YES
3323  5330    JMP RDRIN
3324  4765    JMS DCODE /GET CHAR
3325  7000  FDRET, NOP
3326  6202    CIF
3327  5715    JMP I FEEDF
            /
3330  4764  RDRIN, JMS RKBD
3331  5325    JMP FDRET
            /
3332  0000  GETWOR, 0 /GET WORD FROM BUFFER
3333  7300    CLA CLL
3334  1355    TAD RL
```

```
PAGE 14

3335  1363    TAD (-5600 /TOP
3336  7620    SNL CLA
3337  5352    JMP INTHER
3340  4762    JMS RTAPE
3341  0001  ITAB, 1
3342  0002  INUM, 2
3343  0001  IUN,  1
3344  5200  ILOC, 5200
3345  1341    TAD ITAB
3346  1342    TAD INUM
3347  3341    DCA ITAB
3350  1344    TAD ILOC
3351  3355    DCA RL
3352  1755  INTHER, TAD I RL
3353  2355    ISZ RL
3354  5732    JMP I GETWOR
            /
3355  6001  RL, 6001 /FORCE READ
            /
3356  0000  IPT, 0
3357  0001  RDRSW, 1 /OFF
3362  5000    PAGE
3363  2200
3364  1624
3365  2056
3366  7577
3367  7703
3370  5006
3371  1705
3372  7522
3373  3200
3374  2245
3375  7575
3376  2000
3377  2317
```

```
PAGE 15


                  RTAPE=5000
                  WTAPE=5006
                  /
                  /SUBROUTINE TO COMPRESS ASCII
                  /INTO EDITOR FORMAT
                  /FULL RANGE PACK!!!!
                  /
                  *2000
                  /
2000    0000      COMPRES, 0
2001    3350        DCA IPTB
2002    1350        TAD IPTB
2003    1377        TAD (-212
2004    7450        SNA
2005    5600        JMP I COMPRES /IGNORE LF
2006    1376        TAD (-66 /-300 NOW
2007    7510        SPA /S IF 300 CODE
2010    5230        JMP T200
2011    1375        TAD (-40
2012    7710        SPA CLA /+ARE 77XX
2013    5216        JMP REGOUT
2014    7240      PUT77, CLA CMA /77 PREFIX
2015    4237        JMS PUTOUT
2016    1350      REGOUT, TAD IPTB /CHAR 6 BITS
2017    4237        JMS PUTOUT
2020    1350        TAD IPTB /CR?
2021    1374        TAD (-215
2022    7640        SZA CLA
2023    5600        JMP I COMPRES
2024    1351        TAD RLSW /WHICH SIDE?
2025    7710        SPA CLA
2026    4237        JMS PUTOUT /1500 TO TERMINATE
2027    5600        JMP I COMPRES
2030    1373      T200, TAD (40 /+ARE REG EXCEPI FOR ?
2031    7510        SPA
2032    5214        JMP PUT77 /PRDFIX IT
2033    1372        TAD (-37 /A 277 ?
2034    7650        SNA CLA
2035    5214        JMP PUT77 /YES
2036    5216        JMP REGOUT /ALL DONE
                  /
2037    0000      PUTOUT, 0
2040    0371        AND (77 /TO PUT IN BUFFER
2041    2351        ISZ RLSW /WHICH SIDE?
2042    5246        JMP LEFT
2043    1352        TAD PWORD /RT SIDE ADDED IN
2044    4770        JMS PLACE /PUT IN BUFF
2045    5255        JMP PUTEXT
2046    7106      LEFT, CLL RTL /GET INTO LHF
2047    7006        RTL
2050    7006        RTL
2051    0367        AND (7700 /JUST IN CASE
2052    3352        DCA PWORD
2053    7240        CLA CMA
```

```
PAGE 16

2054  3351     DCA NLSW
2055  5637   PUTEXT, JMP I PUTOUT
             /
             /SUBROUTINE TO DECODE PACKED EDITOR BUFFERS
             /FULL RANGE DECODING!!!!
             /
2056  0000   DCODE, 0
2057  7300     CLA CLL
2060  2345     ISZ LFSW /-1=LFEED NEEDED
2061  7410     SKP
2062  5312     JMP LF
2063  4326     JMS BITS /GET 6 BITS
2064  1366     TAD (-77 /SPECIAL CODE?
2065  7450     SNA
2066  5274     JMP T77 /YEP
2067  1365     TAD (37
2070  7510     SPA
2071  1364   K300, TAD (100
2072  1363   K200, TAD (240
2073  5656     JMP I DCODE /HAVE ASCII
             /
2074  4326   T77, JMS BITS /77 WHAT?
2075  1362     TAD (-3 /END OF BUFFER?
2076  7450     SNA
2077  5324     JMP GETMOR /YES
2100  1361     TAD (-12 /CR?
2101  7440     SZA /YES
2102  5314     JMP REG /NORMAL
2103  1346     TAD HALF /WILL NEXT BE NEW WORD?
2104  7710     SPA CLA
2105  4326     JMS BITS /PULL IN 00 AFTER 15
2106  7240     CLA CMA
2107  3345     DCA LFSW /GIVE LF
2110  1360     TAD (-23 /FOR CR
2111  5272     JMP K200
             /
2112  1357   LF, TAD (-26 /LFEED
2113  5272     JMP K200
2114  1360   REG, TAD (-23 /DOWN 40 NOW
2115  7510     SPA
2116  5272     JMP K200
2117  1372     TAD (-37 /A ? ?
2120  7450     SNA
2121  5267     JMP K300-2 /CLEVER ?
2122  1365     TAD (37
2123  5271     JMP K300
             /
2124  4756   GETMOR, JMS SETUP
2125  5257     JMP DCODE+1
             /
2126  0000   BITS, 0
2127  2346     ISZ HALF /+=LH
2130  5334     JMP LHF
2131  1347     TAD WORD
2132  0371   MASK, AND (77
```

PAGE 17

```
2133  5726    JMP I BITS
2134  7240  LHF, CLA CMA /SET LH SW=-1
2135  3346    DCA HALF
2136  4755    JMS GETWCA
2137  3347    DCA WORD
2140  1347    TAD WORD
2141  7012    RTR
2142  7012    RTR
2143  7012    RTR
2144  5332    JMP MASK
            /
            /
2145  0000  LFSW, 0
2146  0000  HALF, 0
2147  0000  WORD, 0
2150  0000  IPTR, 0
2151  0000  RLSW, 0
2152  0000  PWORD, 0
2155  3332    PAGE
2156  1600
2157  7752
2160  7755
2161  7766
2162  7775
2163  0240
2164  0100
2165  0037
2166  7701
2167  7700
2170  3227
2171  0077
2172  7741
2173  0040
2174  7563
2175  7740
2176  7712
2177  7566
```

CHAPTER 2

CPS SABR ASSEMBLER

The DEC SABR assembler has been incorporated into CPS and provisions for listing control have been added. The file *SABR contains version 13 of SABR along with the associated CPS support code.

*SABR is used to assemble files containing programs written using the SABR language. The resulting relocatable binary output is stored in the binary working area -B. *SABR is also used as the second pass of DEC's 8-K FORTRAN compiler.

A total of 67 locations in the original DEC version have been patched in order to allow it to interface with CPS.

Operation proceeds in the following manner:

1. The command

```
RUN *SABR FILE1+...+FILE6  2=FILE7+...+FILE12
     3=FILE13+...+FILE18  P=...
```

is used to load and start SABR. A maximum of 18 source files can be used to form a single SABR program. If no file names are supplied it is assumed that the input is to come from the ASR-33. SABR recognizes the following parameters in the P=... construct:

B, NB	binary, no binary
L, NL	listing, no listing
S, NS	symbol table, no symbol table

The default set is BNLNS.

2. The coreimage file *SABR is loaded by the RUN loader and started at location 11200 in the support code. The parameter field is scanned by the subroutine, PARTST. This subroutine sets various switches depending on the specified parameters. Illegal characters are ignored and a blank terminates the scan. Next, the input file list is checked to see if the source program is contained in CPS files or if it is to come from the ASR-33. If it is to come from the reader (the contents of 17600 is -3) then RDRSW is set to 0. If the input is to come from CPS files, BLKPT is set to point to the first set of file parameters in the communication area (CA). This has the effect of initializing the SETIPT routine (which is used to initialize the input buffering routines and handle concatenated files). SETIPT is then called so that when SABR is started the buffer routines will be initialized. Finally, the file update switches in the CA are set (no file specifications are made at this time) and SABR is then started at location 00200.

3. Each time SABR requests an input character it does a JMS to the GET subroutine. GET fetches its characters through INSCAN.

INSCAN checks the input text stream for listing control commands which are flagged by the occurrence of a (CTRL-H). INSCAN deletes these commands from the text fed to SABR. When a (CTRL-H) is encountered, the trailing characters are checked for command interpretation. The listing control commands are described below.

INSCAN gets its characters from INPUT. INPUT checks RDRSW to see if the input is coming from the ASR-33 or from CPS files. If the ASR-33 is being used INPUT reads in the next character and forces the parity channel to a 1. If the input is coming from CPS files INPUT makes a call to DCODE.

DCODE is used to convert the *ED 6 bit packed data format into ASCII. It also supplies a line feed after each carriage return. When DCODE encounters a (CTRL-C) (*ED end of buffer symbol) it makes a call to SETIPT to set up the next input file (if there is one). The subroutine BITS is used to obtain 6 bit characters from the input buffer.

BITS returns successive half words to DCODE, a half word per call. The full word packed code is obtained from GETWOR.

GETWOR first checks to see if the next word is in the input buffer. It does this by checking to see if GLOC is less than 1600. (The input buffer is two pages long and lies between 11200 and 11577.) If GLOC is in the desired range the next packed word is extracted from the buffer by doing an indirect on GLOC. GLOC is then incremented by one and a return is made to the caller. If GLOC is equal to 1600, RBUF is used to read a new segment into the buffer and to reset GLOC to 1200. The next word is then obtained by making an indirect on GLOC and GLOC is then incremented by one.

RBUF uses a one page LINC tape routine for the actual tape operations.

4. Each time SABR outputs a binary frame it makes a JMS to PUT. The PUT routine packs the output frames, 3 frames per two computer words. Each time a new location is required in the output buffer (2 pages long, located in 17200 through 17577) PUT makes a call to TSTPLC.

TSTPLC advances PLOC by one and then checks to see if the new value is in the buffer. If it is, a normal return is made. If PLOC points outside the buffer then the current buffer contents are written into -B and PLOC is set to 7200 (WBUF does this).

5. When SABR completes its first pass it makes a JMS to MIDP. This routine first writes out the contents of the current buffer thus terminating the binary file being constructed in -B. MIDP checks to see whether 1 or 2 blocks are required out of the 2 page output buffer. SETIPT is reinitialized in case a listing has been requested and the output routine in SABR is set to feed PSCAN instead of PUT. Control is then returned to SABR.

6. SABR now proceeds to output the symbol table. This output goes to PSCAN which checks to see if the user has requested that it be typed. A line buffer is used starting at location 17200 in the old output buffer area. Upon forming an output line, PSCAN types this line if requested by the user. Otherwise the line is checked for the occurrence of an undefined symbol. If the symbol on the current line is defined, the line is ignored. If the symbol is undefined, the line is typed out. Thus the user is always notified of undefined symbols.

7. When SABR finishes outputting the symbol table it has been patched to output a 377 code. This output flags PSCAN that the symbol table has been output and that if SABR is returned to, the output listing will follow.

The listing control switch is now checked. If the user did not request a listing, control is passed to FINISH. This routine sets up the file pointers for -B and returns to the file system.

If a listing has been requested then a return is made to SABR.

8. Listing control is obtained by scanning the input line and setting switches which are checked by the output routines. Facilities have been included for turning the printing on and off and for forcing a page eject. The output from pass 2 is typed using a standard 8-1/2 by 11 inch format with 57 lines typed per page. Listing control is invoked through the occurrence of a (CTRL-H) in the input text.

listing command ::= CTRL-H<command character>

command character ::= N<command character> |E|P|
<other character>

other character ::= any ASCII character not N or E or P

Other characters have no effect and are ignored. The above approach can cause problems if a (CTRL-H) is followed by a carriage return.

The command characters have the following effect:

N negate the following command character string.

P turn the print on.

E start a new page .. page eject.

Memory Organization

The upper 4-K of memory is used as indicated below. The lower 4-K only contains patches. The top of the SABR symbol table has been moved down to allow the inclusion of the support routines.

Locations	Contents
11000-11177	SETIPT, RBUF, GETWOR, CHTYP
11200-11377	SABR, SETRDR, also the first half of the input buffer
11400-15577	the rest of the input buffer
16000-16177	PSCAN, TSTLST, SCNRET, PRTLIN, BFIX, DPOST, ENDSYM, PLN, CRLF
16200-16377	EJECT, INSCAN

Locations	Contents
16400-16577	DCODE, BITS, INPUT, FINISH
16600-16777	GET, PUT, TSTPLC, WBUF, MIDP, GCDF
17000-17177	RTAPE, WTAPE
17200-17577	output buffer

Subroutine Descriptions

SETIPT Used to set up the input buffer routines whenever a new file is to be accessed. It also checks to see if a file is available. BLKPT must be initialized to 7600 in order to initialize SETIPT. This routine sets GLOC to 1777 and HALF to 0 so that the next data request forces the input buffer to be filled.

RBUF This·routine calls RTAPE to read 2 blocks of text into the input buffer. It then resets GLOC to 1200 (the bottom of the input buffer).

GETWOR Extracts a word from the input buffer. First it checks GLOC to see if the next data location is in the input buffer. If it is, GETWOR does an indirect access using GLOC, increments GLOC by 1 and returns. If the next location is not in the input buffer, RBUF is called which reloads the buffer and resets GLOC. The next word is extracted from the buffer, GLOC is incremented by 1 and GETWOR returns to the caller.

CHTYP Used to type ASCII characters on the ASR-33. Returns with a clear AC and link.

SABR The file system starts *SABR here. The parameter string is scanned using PARTST, the file update switches are set, RDRSW is set depending on whether or not the input is to be on the ASR-33, and control is passed to the SABR assembler at location 00200.

SETRDR If the ASR-33 is to be used for input, SETRDR sets the reader switch RDRSW equal to 0.

PSCAN — Used to monitor the output of SABR during the symbol table and listing operations. PSCAN places the SABR output into a line buffer using DPOST. A line is terminated by a carriage return with line feeds being ignored. Once a line has been formed MODE is checked to see if SABR is listing a program or dumping the symbol table.

If MODE=0 the symbol table is being dumped. If SYMSW if 0 the user has requested that it be typed. If SYMSW is 1 the line is checked to see if the symbol it describes is undefined. If it is defined the line is ignored. If the symbol is undefined then the current line is typed as an error message to the user.

TSTLST — Tests to see if the listing switch is on. If not, the buffer is reset and no line is typed. If the switch is on, control is transferred to PRTLIN. LSTSW is set and cleared by the listing commands, P and NP.

SCNRET — Used to make the return from PSCAN.

PRTLIN — Causes the current line to be typed and keeps count of the number of lines on the current page. Generates page ejects at the end of each page.

BFIX — Resets the line buffer used in typing the symbol table and the listing. Also clears the location where a U for an undefined symbol would be found.

DPOST — Places the output characters from SABR into the line buffer. Checks for the occurrence of a 377 code which flags the end of the symbol table output. The 377 code causes control to be passed to ENDSYM.

ENDSYM — Terminates the symbol table listing (if it is being typed) and checks to see if a listing has been requested by the user. If LSTSW=1 control is passed to FINISH. If LSTSW=0 then the listing switch LSTSWT is set to allow typing, the mode is set to listing and control is passed to PRTLIN forcing a new output page. Control is then returned to SABR for the second pass.

PLN — Prints the contents of the line buffer.

CRLF Types a carriage return followed by the number of line feeds specified by minus the contents of the AC when CRLF was entered.

EJECT This is the page eject routine. It is also responsible for typing the page headings and page numbers.

INSCAN Scans the code being fed to SABR in order to intercept listing commands and delete them from the text. When a listing command is encountered, INSCAN also sets the appropriate switches.

DCODE Obtains six bit characters using the BITS routine and maps these back into ASCII assuming that the packed input uses the *ED format. A line feed is supplied after every carriage return. If a (CTRL-C) is encountered a call is made to SETIPT to set up the next input file and DCODE is restarted.

BITS Furnishes DCODE with 6 bit words for use in reconstructing the source ASCII. Uses GETWOR to obtain its packed words and uses HALF as a switch to determine which side of a word should be used next.

INPUT Checks RDRSW to see if the input text is to come from the ASR-33 of from CPS files. If it comes from the ASR-33 then a character is read and its parity channel is forced to a 1. If the input is to come from CPS files then a call is made to DCODE.

FINISH Updates the -B pointers, forces the terminal page eject, and returns to the CPS file system.

GET SABR comes here for an input character. GET goes to INSCAN to get the desired character.

PUT SABR comes here to get rid of its binary paper tape frames. This routine packs these frames 3 per 2 words. PUT uses TSTPLC to insure that the buffer location currently in use is valid.

TSTPLC Advances the output buffer pointer PLOC and checks to see if the indicated location is in core. If it is, a return is made. If it is not in core then a call is made to WBUF to write the current buffer onto tape and to reset PLOC.

WBUF Writes the contents of the output buffer onto tape and resets the output buffer pointer PLOC. Uses the one page tape routine for the actual tape operation.

MIDP SABR comes here after completing the desired assembly and before outputting the symbol table. The output buffer is written into -B and SETIPT is reset and called in case a listing pass is required (this does not involve a tape operation). The SABR output routine is set to PSCAN and SABR is restarted.

GCDF This routine sets the data field to 1 and does a RDF with a CDF in the AC. Return is made with a data field resetting CDF in the AC.

RTAPE }
WTAPE } A one page tape routine for reading and writing 128 word blocks on LINC tape. See the enclosed listing for details.

Assembly Instructions

The following procedure is used in constructing the coreimage file *SABR.

```
RUN *ASM S1+S2+S3+S4+S5+S6    2=$

SAVE -B SABRSUP

RUN *ASM F1TAPE

SAVE -B BF1TAPE

LOAD A

GET SABRV13

GET SABRSUP

GET BF1TAPE @17000

BUILD      (the starting address is 11200)
```

When control returns to the file system, -B will contain the desired coreimage.

Files S1 through S6 contain the symbolic version of the patches and support code required to interface SABR with CPS. F1TAPE is a one page tape routine for reading and writing LINC tapes using a 128 word format. It is the only tape routine used by *SABR. SABRV13 is a binary file which contains the binary paper tape of DEC's SABR assembler. This file is generated using the program PAPERBIN.

SABR Comments

The listing support code is somewhat more complicated than it has to be.

The input buffer is two pages long and is always filled by reading two blocks off of tape. This can cause problems if the page following an input file does not have the correct checksum (note that LINC tapes have 8 extra blocks at the end of a tape so that end of tape encounters are not a problem).

The file pointers for -B are not updated until after all of the user requested options have been supplied. In particular, if an impatient user manually terminates a listing and returns to the file system the -B parameters will not be those associated with its contents. A more reasonable approach would be to set the -B parameters when control returns to the support code before the symbol table is dumped.

SABR Highlights

Input is assumed to use the *ED format.

The input can contain listing control commands.

The output is placed in -B.

Undefined symbols are listed even if the user has not requested a symbol table listing.

The input buffer is 2 pages long. The output buffer is 2 pages long.

The output consists of binary paper tape frames 3 packed per 2 computer words.

Assembly proceeds from source files into -B.

The standard conventions for the CPS communication area are observed.

If no input files are specified the input is assumed to come from the ASR-33.

```
               FIELD 0
               /
               /PATCHES FOR SABR ASSEMBLER V(03)
               /
               CORE1=6000
               /
               RTAPE=7000
               WTAPE=7006
               /
               *6
0006   5777    CORE1-1
               /
               *560
0560   5367      JMP 567 /JMP ENDEND-GETTING RID OF THE HALT
               /
               *566
0566   5773      5773 /JMP I REE
0567   4455      4455 /JMS I WLNP
0570   6213      CDF CIF 10
0571   5772      JMP I .+1
0572   6505      FINISH
               /
               *2762
2762   5763      JMP I .+1 /FATAL ERROR ROUTIN
2763   6422      FATERRET
               /
               *3162
3162   6000      CORE1
               /
               *3165
3165   6212      CIF 10
3166   4767      JMS I .+1
3167   6610      PUT    /THIS IS PUNCH OVER RIDE
               /
               *4171
4171   4772      JMS I .+1 /END OF SYMBOL TABLE
4172   4661      SETLIST
               /
               *4332
4332   7000      NOP /CRLF
4333   7000      NOP /CRLF
               /
               *4355
4355   7000      NOP /L.T.
               /
               *4357
4357   5777      CORE1-1
               /
               *4657 /SUPPRESS SABR BUFFER
4657   4702      4702 /JMS I INDEV
4660   5656      5656 /JMP I R
               /
4661   0000    SETLIST, 0 /FEEDS A 377 TO PSCAN
4662   4424      4424 /JMS I CRLF
```

```
4663  6212     CIF 10
4664  1271     TAD RUBOUT
4665  4666     JMS I .+1
4666  6000     PSCAN
4667  2261     ISZ SETLIST /MISS PATCH
4670  5661     JMP I SETLIST
4671  0377   RUBOUT, 377
               /
               /
               *4722 /LDR ROUTINE
4722  6212     CIF 10
4723  4724     JMS I .+1
4724  6715     MIDP
4725  5721     JMP I 4721
               /
               *6421
6421  5617     5617 /JMP I IOINIT
               /
6422  0000   FATERRET, 0 /IN IOINIT AREA
6423  6203     CDF CIF
6424  1634     TAD I FATML
6425  7450     SNA /DONE?
6426  5232     JMP .+4
6427  4454     JMS I 54 /TYPE IT
6430  2234     ISZ FATML
6431  5224     JMP .-5
6432  5633     JMP I .+1
6433  7600     7600
6434  6435   FATML, .+1
6435  0241   "!
6436  0306   "F
6437  0301   "A
6440  0324   "T
6441  0301   "A
6442  0314   "L
6443  0241   "!
6444  0000   0 /TERMINATOR
               /
               *6627 /HSR PATCH
6627  6212     CIF 10
6630  4631     JMS I .+1
6631  6600     GET
6632  5626     JMP I 6626
               /
               *6656 /ASR INPUT FIX
6656  6031     KSF
6657  5256     JMP .-1
6660  6036     KRB
6661  0264     AND .+3
6662  1265     TAD .+3
6663  5655     JMP I .-6
6664  0177     177
6665  0200     200
               /
               *6527
```

```
6527  0001    1 /LIST ON HSP
            /
              PAGE
```

```
               /SABR STARTS HERE!!
               /
               FIELD 1
               /
               *1200
               /
1200   7300    SABR, CLA CLL
1201   4777      JMS PARTST /SCAN PAR LIST
1202   1776      TAD 7600 /SEE IF RDR INPUT
1203   1375      TAD (3
1204   7650      SNA CLA
1205   5232      JMP SETRDR /YES, TURN ON
1206   1376      TAD (7600
1207   3774      DCA BLKPT /SET UP BLOCK POINTER
1210   4773      JMS SETIPT /SET UP READ BLOCKS
1211   7300    GOSABR, CLA CLL
1212   1372      TAD (7773
1213   3010      DCA 10
1214   1371      TAD (-7402
1215   3410      DCA I 10
1216   7240      CLA CMA
1217   3410      DCA I 10
1220   7201      CLA IAC
1221   3410      DCA I 10
1222   1370      TAD (7402
1223   3410      DCA I 10
1224   6203      CDF CIF /GO START SABR NOW
1225   6046      TLS
1226   6036      KRB
1227   7300      CLA CLL
1230   5631      JMP I .+1
1231   0200      200
               /
1232   7300    SETRDR, CLA CLL
1233   3767      DCA RDRSW /TURN ON
1234   5211      JMP GOSABR
1367   6504      PAGE
1370   7402
1371   0376
1372   7773
1373   1100
1374   1167
1375   0003
1376   7600
1377   1400
```

```
                *1100
                /
1100   0000     SETIPT, 0 /SET UP CONCATS ETC
1101   7300       CLA CLL
1102   1767       TAD I BLKPT
1103   7450       SNA /NOT 0 DONT TERM
1104   5355       JMP TOMANY
1105   7500       SMA /SEE IF -1,-2,-3
1106   5313       JMP .+5
1107   1377       TAD (10
1110   7500       SMA /NO VALID BLN CAN GOOF IT
1111   5355       JMP TOMANY
1112   1376       TAD (-10 /RESOTRE
1113   0375       AND (1777
1114   3331       DCA BLK
1115   1767       TAD I BLKPT /GET UNIT
1116   7006       RTL
1117   0374       AND (1
1120   3333       DCA LOCI-1
1121   2367       ISZ BLKPT
1122   2367       ISZ BLKPT /IGNORE LENGHT
1123   1375       TAD (1777 /FORCE FIRST READ
1124   3354       DCA GLOC
1125   3773       DCA HALF
1126   5700       JMP I SETIPT
                /
1127   0000     RBUF, 0
1130   4772       JMS RTAPE
1131   0000     BLK, 0
1132   0002       2 /BLOCKS
1133   0000       0 /NORMALLY 0
1134   1200     LOCI, 1200
1135   1331       TAD BLK
1136   1332       TAD BLK+1
1137   3331       DCA BLK
1140   1334       TAD LOCI
1141   3354       DCA GLOC
1142   5727       JMP I RBUF
                /
1143   0000     GETWOR, 0 /GET WORD FROM INPUT BUFF
1144   7300       CLA CLL
1145   1371       TAD (-1600 /ARE WE IN BUFFER?
1146   1354       TAD GLOC /L=1 WE ARE NOT
1147   7630       SZL CLA
1150   4327       JMS RBUF /GET MORE
1151   1754       TAD I GLOC
1152   2354       ISZ GLOC
1153   5743       JMP I GETWOR
                /
1154   1700     GLOC, 1700 /TO FORCE FIRST READ
                /
1155   6203     TOMANY, CDF CIF
1156   5757       JMP I .+1 /USE SABR'S ROUTINE
1157   2702       2702
```

PAGE 06

```
                /
1160   0000     CHTYP, 0 /CHARACTER TYPER, RETS WI AC=L=0
1161   6046       TLS
1162   6041       TSF
1163   5362       JMP .-1
1164   7300       CLA CLL
1165   5760       JMP I CHTYP
1166   2702       2702
                /
1167   0000     BLKPT, 0
1171   6200       PAGE
1172   7000
1173   6532
1174   0001
1175   1777
1176   7770
1177   0010
```

```
                /PARAMETER TESTING ROUTINE
                /
                FIELD 1
                /
                *1400
                /
1400    0000    PARTST, 0
1401    7300      CLA CLL
1402    3236      DCA NEGSW /0=YES
1403    1377      TAD (7707
1404    3237      DCA PARFLD /PT OT LIST
1405    2237    PLPB, ISZ PARFLD
1406    1240      TAD PTAB /CHAR TAB
1407    3241      DCA PTABL
1410    1242      TAD PDISP /DISPATCH TABLE
1411    3243      DCA PDISPL
1412    1244      TAD NENT /# ENTRIES
1413    7041      CIA
1414    3245      DCA PCNT
1415    1637    PLPA, TAD I PARFLD /GET TEST CHAR
1416    7450      SNA
1417    5600      JMP I PARTST /0 TERMINATES
1420    7041      CIA
1421    1641      TAD I PTABL
1422    7650      SNA CLA
1423    5232      JMP MATCH
1424    2243      ISZ PDISPL
1425    2241      ISZ PTABL
1426    3236      DCA NEGSW /SET TO YES
1427    2245      ISZ PCNT
1430    5215      JMP PLPA
1431    5205      JMP PLPB
                /
1432    1643    MATCH, TAD I PDISPL
1433    3246      DCA MTEM
1434    4646      JMS I MTEM
1435    5205      JMP PLPB
                /
1436    0000    NEGSW, 0
1437    0000    PARFLD, 0
1440    1447    PTAB, PARTAB
1441    0000    PTABL, 0
1442    1453    PDISP, DISPATCH
1443    0000    PDISPL, 0
1444    0004    NENT, 4 /4 ENTRIES
1445    0000    PCNT, 0
1446    0000    MTEM, 0
1447    0316    PARTAB, "N
1450    0323     "S
1451    0314     "L
1452    0240      240 /BLANK
                /
1453    1457    DISPATCH, NEG
1454    1465      SYMTAB
```

```
1455  1472     LISTING
1456  1477     BLANK
              /
1457  0000    NEG, 0
1460  1236     TAD NEGSW
1461  7650     SNA CLA
1462  7001     IAC
1463  3236     DCA NEGSW
1464  5657     JMP I NEG
              /
1465  0000    SYMTAB, 0
1466  1236     TAD NEGSW
1467  3776     DCA SYMSW
1470  3236     DCA NEGSW /SET TO YES
1471  5665     JMP I SYMTAB
              /
1472  0000    LISTING, 0
1473  1236     TAD NEGSW
1474  3775     DCA LSTSW /EXTERNAL LIST SWITCH
1475  3236     DCA NEGSW /TO YES
1476  5672     JMP I LISTING
              /
1477  0000    BLANK, 0
1500  7300     CLA CLL
1501  5600     JMP I PARTST
              /
1575  6152     PAGE
1576  6150
1577  7707
```

```
                /SABR I-0 SUPPORT
                /
                FIELD 1
                /
                *6600
                /
6600    0000    GET, 0 /GET A CHARACTER FOR SABR
6601    2200      ISZ GET /MISS PATCH
6602    4340      JMS GCDF
6603    3205      DCA RETG
6604    4777      JMS INSCAN /GET CHAR AND USE FOR CONTROL
6605    7000    RETG, NOP
6606    6202      CIF
6607    5600      JMP I GET
                /
6610    0000    PUT, 0 /PUT AND PACK SABR BINARY
6611    0376      AND (377
6612    3314      DCA IPT
6613    2210      ISZ PUT /MISS PATCH
6614    4340      JMS GCDF
6615    3260      DCA PRET /FOR RET
6616    1351      TAD PUTSW /WHICH THIRD?, INIT = 0
6617    7450      SNA /0=LEFT
6620    5251      JMP PART1
6621    7700      SMA CLA /-1=RIGHT
6622    5231      JMP PART2 /+1=MID
6623    1314      TAD IPT /DO RIGHT SIDE
6624    1713      TAD I PLOC
6625    3713      DCA I PLOC /PACKED IN
6626    4263      JMS TSTPLC /GET NEXT WORD ADVANCE
6627    3351      DCA PUTSW /SET FOR LSIDE
6630    5260      JMP PRET
6631    1314    PART2, TAD IPT /SPLITS TWO WORDS
6632    7112      RTR CLL
6633    7012      RTR
6634    0375      AND (17
6635    1713      TAD I PLOC
6636    3713      DCA I PLOC /PACKED HIGH 4 BITS
6637    4263      JMS TSTPLC /ADVANCE TO THE NEXT WORD
6640    1314      TAD IPT
6641    7112      RTR CLL
6642    7012      RTR
6643    7010      RAR
6644    0374      AND (7400
6645    3713      DCA I PLOC
6646    7240      CLA CMA
6647    3351      DCA PUTSW
6650    5260      JMP PRET
6651    1314    PART1, TAD IPT
6652    7006      RTL
6653    7006      RTL
6654    0373      AND (7760
6655    3713      DCA I PLOC
6656    7201      CLA IAC
```

```
6657  3351     DCA PUTSW
6660  7000   PRET, NOP
6661  6202     CIF
6662  5610     JMP I PUT
             /
6663  0000   TSTPLC, 0 /ADVANCE PLOC AND SEE IF IN CORE
6664  2313     ISZ PLOC
6665  1372     TAD (-7600
6666  1313     TAD PLOC
6667  7700     SMA CLA /-IS IN CORE
6670  4272     JMS WBUF
6671  5663     JMP I TSTPLC
             /
6672  0000   WBUF, 0 /WRITE BUFFER
6673  1301     TAD PBLK /TEST FOR END OF BUFFER
6674  1302     TAD PBLK+1
6675  1371     TAD (-150
6676  7700     SMA CLA
6677  5346     JMP TOOFAR
6700  4770     JMS WTAPE
6701  0001   PBLK, 1
6702  0002     2
6703  0001   PUNIT, 1
6704  7200   PLOCI, 7200
6705  1301     TAD PBLK
6706  1302     TAD PBLK+1
6707  3301     DCA PBLK
6710  1304     TAD PLOCI
6711  3313     DCA PLOC
6712  5672     JMP I WBUF
             /
6713  7200   PLOC, 7200 /INIT VALUE TO START
6714  0000   IPT, 0
             /
6715  0000   MIDP, 0 /MID PASS HANDLING FOR POSS LISTING
6716  2315     ISZ MIDP /MISS PATCH
6717  4340     JMS GCDF
6720  3335     DCA MRET
6721  1313     TAD PLOC /HALF?
6722  1367     TAD (-7400
6723  7710     SPA CLA
6724  7240     CLA CMA /FOR LATER USE
6725  3350     DCA PART
6726  4272     JMS WBUF
6727  1366     TAD (7600
6730  3765     DCA BLKPT
6731  4764     JMS SETIPT /RESET TAPE ACCESS ORDER
6732  1352     TAD PPATCH /SET PUNCH TO PSCAN
6733  6201     CDF          /FOR POSSIBLE LISTING
6734  3763     DCA 3167
6735  7000   MRET, NOP /WILL BE A CDF X
6736  6202     CIF
6737  5715     JMP I MIDP
             /
6740  0000   GCDF, 0
```

```
6741   7300     CLA CLL
6742   1362     TAD (6201
6743   6214     RDF
6744   6211     CDF 10
6745   5740     JMP I GCDF
               /
6746   7402    TOOFAR, HLT /SHOULD NEVER GET HERE, TOO MUCH BINARY
6747   5346     JMP .-1 /THIS IS PROTECTION
               /
6750   0000    PART, 0
6751   0000    PUTSW, 0
6752   6000    PPATCH, PSCAN
6762   6201     PAGE
6763   3167
6764   1100
6765   1167
6766   7600
6767   0400
6770   7006
6771   7630
6772   0200
6773   7760
6774   7400
6775   0017
6776   0377
6777   6246
```

```
               /DECODING ROUTINE FOR SABR
               /FULL RANGE!
               /
               FIELD 1
               /
               *6400
               /
6400   0000    DCODE, 0
6401   7300      CLA CLL
6402   2334      ISZ LFSW /-1 NEED L FEED
6403   7410      SKP
6404   5234      JMP LF
6405   4250      JMS BITS
6406   1377      TAD (-77 /SEE IF SPEC CODE
6407   7450      SNA
6410   5216      JMP T77 /YES, BUT WHICH
6411   1376      TAD (37 /-40 NOW
6412   7510      SPA
6413   1375    K300, TAD (100
6414   1374    K200, TAD (240
6415   5600      JMP I DCODE
               /
6416   4250    T77, JMS BITS /GET NEXT 6 BITS
6417   1373      TAD (-3 /TERMINATOR?
6420   7450      SNA
6421   5246      JMP GETMOR
6422   1372      TAD (-12
6423   7440      SZA /CR?
6424   5236      JMP REG /NORMAL
6425   1332      TAD HALF
6426   7710      SPA CLA
6427   4250      JMS BITS /GET TRAILING 00
6430   7240      CLA CMA /-1
6431   3334      DCA LFSW /FOR LF TO FOLLOW
6432   1371      TAD (-23 /GET CR
6433   5214      JMP K200
6434   1370    LF, TAD (-26 /FOR LF
6435   5214      JMP K200
6436   1371    REG, TAD (-23 /NOW -40
6437   7510      SPA
6440   5214      JMP K200
6441   1367      TAD (-37 /A? ?
6442   7450      SNA
6443   5211      JMP K300-2 /CLEVER ?
6444   1376      TAD (37
6445   5213      JMP K300
               /
6446   4766    GETMOR, JMS SETIPT
6447   5201      JMP DCODE+1
               /
6450   0000    BITS, 0 /GET 6 BITS
6451   2332      ISZ HALF /+=FETCH WORD
6452   5256      JMP LHF
6453   1333      TAD WORD
```

```
6454  0365  MASK, AND (77
6455  5650    JMP I BITS
6456  7240  LHF, CLA CMA /SET TO RHF NEXT
6457  3332    DCA HALF
6460  4764    JMS GETWOR
6461  3333    DCA WORD
6462  1333    TAD WORD
6463  7012    RTR
6464  7012    RTR
6465  7012    RTR
6466  5254    JMP MASK /STRIP IT
            /
6467  0000  INPUT, 0
6470  7300    CLA CLL
6471  1304    TAD RDRSW /0=RDR
6472  7650    SNA CLA
6473  5276    JMP .+3
6474  4200    JMS DCODE
6475  5667    JMP I INPUT
6476  6031    KSF
6477  5276    JMP .-1
6500  6036    KRB
6501  0363    AND (177
6502  1362    TAD (200
6503  5667    JMP I INPUT
            /
6504  0001  RDRSW,'1 /NORMALLY OFF
            /
            /
            /FINAL WRAP UP AND GO TO CPS
            /
6505  7300  FINISH, CLA CLL
6506  1361    TAD (7767
6507  3010    DCA 10
6510  7201    CLA IAC
6511  3410    DCA I 10
6512  1760    TAD PBLK
6513  1357    TAD (-1
6514  1756    TAD PART
6515  3410    DCA I 10
6516  1355    TAD (1000 /REL BIN CODE
6517  3410    DCA I 10
            /
6520  1754    TAD PACNT /EJECT IF NON 0
6521  7650    SNA CLA
6522  5327    JMP .+5
6554  6157    TAD (7
6555  1000
6556  6750
6557  7777
6560  6701
6561  7767
6562  0200
6563  0177
6564  1143
```

```
6565  0077
6566  1100
6567  7741
6570  7752
6571  7755
6572  7766
6573  7775
6574  0240
6575  0100
6576  0037
6577  7701
6523  1353   1 /AT PAGE TOP?
6524  1752     TAD LNCNT
6525  7640     SZA CLA
6526  4751     JMS EJECT
6527  6203     CIF CDF
6530  5731     JMP I .+1 /GO TO CPS
6531  7600     7600
             /
6532  0000   HALF, 0
6533  0000   WORD, 0
6534  0000   LFSW, 0
6551  6200     PAGE
6552  6244
6553  0071
```

```
            FIELD 1
            *6000
            /
            /OUTPUT MONITOR AND UNDEF SYM CHECK
            /
6000  0000  PSCAN, 0
6001  2200    ISZ PSCAN /MISS PATCH
6002  3345    DCA IPTCH /IPT CHAR
6003  7004    RAL /SAVE LINK
6004  3346    DCA IPTLK
6005  4777    JMS GCDF /GET RET FLD
6006  3240    DCA PSCDF /SAVE
6007  1345    TAD IPTCH
6010  1376    TAD (-212 /LF?
6011  7450    SNA
6012  5237    JMP SCNRET /YES, IGNORE
6013  1375    TAD (-3 /CR?
6014  7640    SZA CLA
6015  5270    JMP DPOST /SAVE IT
6016  1347    TAD MODE /1=LIST, 0=SYM TAB
6017  7640    SZA CLA
6020  5233    JMP TSTLST /TEST IF LIST IS ON
6021  1350    TAD SYMSW /0=PRINT
6022  7650    SNA CLA
6023  5245    JMP PRTLIN
6024  1374    TAD (-325 /UNDEF?
6025  1773    TAD 7214
6026  7640    SZA CLA
6027  5262    JMP BFIX /NOTHING TO GO
6030  1351    TAD BUFPT /YEP, PRINT UNDEF SYM
6031  4315    JMS PLN
6032  5262    JMP BFIX /GET OUT
            /
6033  1353  TSTLST, TAD LSTSWT /LIST ON?
6034  7650    SNA CLA
6035  5245    JMP PRTLIN /YES
6036  5262    JMP BFIX /NO
            /
6037  7300  SCNRET, CLA CLL
6040  7000  PSCDF, NOP /CDF GOES HERE
6041  1346    TAD IPTLK /RESTORE LINK
6042  7010    RAR
6043  6202    CIF
6044  5600    JMP I PSCAN
            /
6045  1357  PRTLIN, TAD PAGNT /IF 0 STARTING
6046  7650    SNA CLA
6047  4772    JMS EJECT /HEAD PAGE
6050  1371    TAD (7200
6051  3354    DCA TLOC
6052  1351    TAD BUFPT
6053  4315    JMS PLN /PRINT LINE
6054  2770    ISZ LNCNT /COUNT IT
6055  7610    SKP CLA
```

```
6056  4772     JMS EJECT /PAGE DONE
6057  1767     TAD ESWT /EXTERNAL PAGE EJECT?
6060  7640     SZA CLA
6061  4772     JMS EJECT
             /
6062  1371   BFIX, TAD (7200
6063  3351     DCA BUFPT /RESET BUFFER
6064  1371     TAD (7200
6065  3354     DCA TLOC
6066  3773     DCA 7214 /CLEAR U
6067  5237     JMP SCNRET /GO BACK
             /
6070  1366   DPOST, TAD (-377 /TEST FOR RUBOUT
6071  1345     TAD IPTCH
6072  7650     SNA CLA
6073  5302     JMP ENDSYM /END OF SYM TAB
6074  1345     TAD IPTCH
6075  3751     DCA I BUFPT /SAVE
6076  2351     ISZ BUFPT
6077  5237     JMP SCNRET /ALL DONE
6100  7402     HLT
6101  5300     JMP .-1 /SHOULDN'T BE ABLE TO GET HERE
             /
6102  7240   ENDSYM, CLA CMA /SET LISTING
6103  3360     DCA HDRSW
6104  1352     TAD LSTSW
6105  7640     SZA CLA
6106  5765     JMP FINISH
6107  3353     DCA LSTSWT /ALLOW LOCAL CONTROL
6110  7201     CLA IAC
6111  3347     DCA MODE
6112  7201     CLA IAC
6113  3767     DCA ESWT
6114  5245     JMP PRTLIN /FORCE NEW PAGE
             /
6115  0000   PLN, 0 /PRINTS LINE
6116  7041     CIA /AC =END OF LINE
6117  1354     TAD TLOC /AC=NUM CHAR
6120  3355     DCA CHCNT
6121  7240     CLA CMA /1 CRLF
6122  4334     JMS CRLF
6123  1355     TAD CHCNT
6124  7700     SMA CLA
6125  5715     JMP I PLN /NO CHARS
6126  1754     TAD I TLOC
6127  4764     JMS CHTYP
6130  2354     ISZ TLOC
6131  2355     ISZ CHCNT
6132  5326     JMP .-4
6133  5715     JMP I PLN
             /
6134  0000   CRLF, 0
6135  3356     DCA CRLFCN
6136  1363     TAD (215
6137  4764     JMS CHTYP
```

```
6140  1362     TAD (212
6141  4764     JMS CHTYP
6142  2356     ISZ CRLFCN
6143  5340     JMP .-3
6144  5734     JMP I CRLF
              /
6145  0000    IPTCH, 0
6146  0000    IPTLK, 0
6147  0000    MODE, 0
6150  0001    SYMSW, 1
6151  7200    BUFPT, 7200
6152  0001    LSTSW, 1
6153  0000    LSTSWT, 0
6154  7200    TLOC, 7200
6155  0000    CHCNT, 0
6156  0000    CRLFCN, 0
6157  0000    PACNT, 0
6160  7777    HDRSW, -1
              /
6162  0212    PAGE
6163  0215
6164  1160
6165  6505
6166  7401
6167  6245
6170  6244
6171  7200
6172  6200
6173  7214
6174  7453
6175  7775
6176  7566
6177  6740
```

```
              *6200 /MORE I-O STUFF
              /
6200   0000   EJECT, 0 /PAGE EJECT
6201   1244     TAD LNCNT /AC=0
6202   1377     TAD (-4
6203   4776     JMS CRLF
6204   1375     TAD (255
6205   4774     JMS CHTYP /-
6206   1373     TAD (-2
6207   4776     JMS CRLF
6210   3243     DCA ECNT /FOR PAGE NUM
6211   2772     ISZ PACNT /ADV PAGE NUMBER
6212   1772     TAD PACNT /GET PAGE NUM
6213   1371   TENL, TAD (-12
6214   7510     SPA
6215   5220     JMP .+3
6216   2243     ISZ ECNT
6217   5213     JMP TENL
6220   1370     TAD (12
6221   1367     TAD (260
6222   3320     DCA NU2 /LOWER DIGIT
6223   1243     TAD ECNT
6224   1367     TAD (260
6225   3317     DCA NU1 /TENS DIGIT
6226   1366     TAD (-71
6227   3244     DCA LNCNT
6230   3245     DCA ESWT /EXTERNAL EJECT OFF
6231   1310     TAD PMES /PAGE
6232   3765     DCA TLOC
6233   1311     TAD PMES+1
6234   2764     ISZ HDRSW
6235   7410     SKP
6236   1346     TAD ADDTNL /SEND-SBEG+1
6237   4763     JMS PLN /PRINT
6240   1373     TAD (-2
6241   4776     JMS CRLF
6242   5600     JMP I EJECT
              /
6243   0000   ECNT, 0
6244   7775   LNCNT, -3 /FOR FIRST CALL
6245   0000   ESWT, 0
              /
              /INPUT SCANNER
              /
6246   0000   INSCAN, 0 /SEE IF CTRL-H
6247   4762     JMS INPUT
6250   1361     TAD (-210
6251   7450     SNA
6252   5255     JMP CTRLH
6253   1360     TAD (210 /RESTORE
6254   5646     JMP I INSCAN
6255   4762   CTRLH, JMS INPUT /WHAT CONTROL?
6256   1357     TAD (-305 /E?
6257   7450     SNA
6260   5272     JMP EFX /SET UP EJECT
```

```
6261  1356     TAD (-13 /P?
6262  7450     SNA
6263  5277     JMP PFX
6264  1355     TAD (2 /N?
6265  7650     SNA CLA
6266  5302     JMP NFX /NEGATE
6267  3307  ALRET, DCA NEGSW2 /SET YES
6270  4762     JMS INPUT /GET VALID CHAR
6271  5646     JMP I INSCAN
            /
6272  1307  EFX, TAD NEGSW2 /BACKWARDS
6273  7650     SNA CLA
6274  7001     IAC
6275  3245     DCA ESWT
6276  5267     JMP ALRET
            /
6277  1307  PFX, TAD NEGSW2
6300  3754     DCA LSTSWT
6301  5275     JMP EFX+3 /TURN EJECT OFF
            /
6302  1307  NFX, TAD NEGSW2
6303  7650     SNA CLA
6304  7001     IAC
6305  3307     DCA NEGSW2
6354  6153     JMP CTRLH /GET ANOTHER CHAR
6355  0002
6356  7765
6357  7473
6360  0210
6361  7570
6362  6467
6363  6115
6364  6160
6365  6154
6366  7707
6367  0260
6370  0012
6371  7766
6372  6157
6373  7776
6374  1160
6375  0255
6376  6134
6377  7774
6306  5255
            /
            /
6307  0000  NEGSW2, 0
            /
            /
6310  6312  PMES, P1
6311  6321         NU2+1 /FOR HEADINGS
6312  0320  P1, "P
6313  0301     "A
6314  0307     "G
```

```
6315  0305     "E
6316  0240     "
6317  0260  NU1, "0
6320  0260  NU2, "0
6321  0240  SBEG, "
6322  0323     "S
6323  0301     "A
6324  0302     "B
6325  0322     "R
6326  0240     "
6327  0301     "A
6330  0323     "S
6331  0323     "S
6332  0305     "E
6333  0315     "M
6334  0302     "B
6335  0314     "L
6336  0305     "E
6337  0322     "R
6340  0240     "
6341  0326     "V
6342  0250     "(
6343  0261     "1
6344  0263     "3
6345  0251  SEND, ")
            /
6346  0025  ADDTNL, SEND-SBEG+1 /ADD L FOR SABR HEADING
            /
            PAGE
```

CHAPTER 3

CPS RELOCATABLE LOADER

The relocatable loader used in CPS is version 06 of DEC's relocatable loader. A total of 31 locations have been patched and extensive support code has been added in order to permit keyboard control of the loading process. The user operation of the relocatable loader is very similar to that of the absolute loader. The relocatable loader does not operate under interrupts.

The relocatable loader uses the (.) as its prefix character on input command lines. Line editing facilities are also provided. The editing control characters are:

(RUBOUT)	delete the last typed character . . . echos a \
(BACKARROW)	delete the entire line
(LINE FEED)	echo the current command line
(CTRL-P)	accept the next character as is . . . literal next
(-) (CR)	continue this command on the next line
(CR)	end of command line

Up to 255 characters can be used to form a command line.

The loading procedure consists of taking CPS SABR produced binary files and loading their contents into a core image maintained in the -S working area. The loader has been restricted to a maximum of 8-K of memory. The area used on unit 0 for the coreimage consists of tape blocks 200 through 277. Location 0 of block 200 corresponds to location 0 in memory and location 127 of block 277 corresponds to location 17777 of memory. The field 1 run time is located in block 241. The field 0 run time is kept in core during the loading procedure so that the required linkage lists can be constructed with a minimum of fuss. The field 0 run time is written into the coreimage when the BUILD command is given by the user. The finalized coreimage module is left in -B and the proper pointers in the CPS communications area (CA) are set to indicate the current state of -B to the file system.

Loading Procedure

The relocatable loader is invoked by giving the CPS file system the command

LOAD R

This should be changed to RUN *RLOAD in order to maintain a uniform command structure in CPS. The relocatable loader is already in coreimage form so that this change should be a minor one.

The RUN loader loads the entire relocatable loader file into the lower 4-K of memory. This is done in order to preserve the CPS indexes contained in memory field 1. The program is then started at location 05600.

The first order of business consists of searching the unit 1 CPS index for all relocatable binary file entries and moving these into the area in memory field 1 which is to serve as the relocatable loader index. Only the names and file locations for files on tape unit 1 are retained. Each entry in the new index consists of 5 locations, the first 4 locations consist of the file name and the fifth, the file location.

After the new index has been constructed the loader code is moved up into field 1, the field 0 run time is placed where it belongs, control is transferred to field 1 and the field 1 run time is written onto tape. At this point control is transferred to the restart point CRES.

CRES enters the DEC relocatable loader (DRL) for initialization at location 6200. DRL returns to CRES at RLSTRTR. Portions of low core are zeroed and the coreimage area on tape is sprayed with zeros as a "core constant". Next the page reference list is zeroed and the run time pages referenced. Finally, the file input routines are initialized in order to force an initial tape read and control is passed to the internal re-entry point, ENTER.

Loader Command Input

ENTER sets the prefix character to (.) and goes to PLIB (Place Line In Buffer). The PLIB routine uses a 2 page line buffer to accumulate the input command line. The buffer holds straight ASCII with the parity channel forced to 1. All line editing operations are carried out in PLIB. When a terminal carriage return is supplied by the user, PLIB returns to the caller.

The output routine is initialized and the Get Word From Line Buffer routine (GWFLB) is called in order to transfer the first input word into the word area W1 through W4. The word is constructed using simple 6 bit stripped ASCII with two characters stored per computer word. Trailing blanks are provided if needed.

The contents of W1 - W4 are reduced to 3 characters and SLRV is used to do a table search for the specified command (SLRV is also used in performing index searches). If the command is valid, control is passed to the appropriate module. If an invalid command has been entered, an error message is typed out and control is again passed to ENTER.

Loader Operation

Because of the complexity of the loader, its operation will now be described on a command by command basis.

GET

The GET command is used to cause user specified files to be loaded. Up to 6 files can be loaded with a single GET command.

Control is transferred to CGET. CGET first transfers the file names following the GET command into a get list, CGETL. No names or more than six names cause an error message to be typed and control returned to ENTER. The get list is terminated by a 0 and is then used in searching the index for the named files and for constructing a block number list which will be used in the actual loading operation. SLRV is used in searching the index. When the block number list has been constructed control is passed to LOADP.

LOADP is the routine used to initialize the input routines and cause the user supplied list of programs to be loaded by DRL. SETIPT is used to set up the input routines. Control is finally passed to DRL at location 7251. After the loading of a program control returns to LOADP. If another program is to be loaded LOADP recycles. If not, control is returned to ENTER.

The DRL obtains binary paper tape frames by making calls on GETIPT. This routine unpacks the frames stored in SABR binary files and returns them to DRL. Before trying to extract a frame from the input buffer (locations 01000 through 02777) GETIPT checks to see if the next input word is in core by making a call to GNXLOC.

GNXLOC checks to see if the next data word is in core. If it is, a normal return is made with GETLO pointing at the desired location. If the next word is not in core then a call is made to RTAPE in order to fill the buffer and GETLO is reset to point at the buffer bottom. Note that the actual size of a SABR file is not retained, thus the buffer is always filled using 8 blocks off of tape. This can cause problems if the blocks following a SABR file do not contain the correct checksum. It is strongly recommended that the relocatable loader support code be rewritten so that this problem is eliminated.

When the DRL is ready to deposit a program word into core it instead makes a call to RLOAD.

RLOAD sets up the desired address using DRL pointers and makes a call on GETLOC. GETLOC checks to see if the desired address is in the coreimage segment currently in core (the coreimage buffer occupies locations 03000 through 07000). If it isn't, GETLOC writes out the current buffer and loads in a portion of the coreimage which contains the desired location. When the current location is in core, GETLOC determines the buffer address of this location and sets a page pointer. Control is then returned to RLOAD. Next PLACE is called. This routine stores the contents furnished by DRL into the coreimage buffer and returns to RLOAD. The memory page just stored into (in the coreimage, that is) is then referenced through a call to REFER. Finally, RLOAD returns to DRL.

If during the loading process DRL encounters an error, control is transferred to RLERROR. This routine first checks to see if the current memory field has been filled. If it has and the current field is field 0, the memory bank used by DRL is set to 1 and LOADP is re-entered at LOAD2 in order to make a second try at loading the current file. If field 1 has also been filled, the appropriate error message is typed and control transferred to ENTER. All other errors cause an error message to be typed and control to be transferred to ENTER.

After successfully loading a program DRL returns control to LOADP. Any additional files are then loaded. Control eventually returns to ENTER when the input file list has been exhausted.

BUILD

This command is used to take the current contents of the coreimage on tape unit 0 and form a coreimage module in -B on unit 1.

The command dispatch routine transfers control to the BUILD module at CBUI. The first order of business is to check whether or not the user has supplied a starting address to be used with the coreimage module being constructed. If he has, it is used. If he hasn't then the entry point of the first loaded SABR file is used. This address is extracted from the field 0 run time which is located in the lower memory bank. The subroutine WPRESC is then called in order to write the contents of the current coreimage buffer onto tape. The memory field Ø run time is then written into the coreimage. Next the entire field 1 portion of the coreimage is read into the lower memory bank and DISCBL is called to pack this into the RUN loader format and write the result into the -B working area. The same procedure is followed for the field 0 portion of the coreimage. Once both fields have been packed and transferred into -B a coreimage descriptor block is appended and a return is made to the file system with the -B update switches set properly.

It is felt that the BUILD command can be re-coded in a much more efficient manner.

DUMP

This command is used to allow the user to examine locations within the coreimage file under construction.

The command dispatcher enters this module at CDUM. If the user has only specified one location to be dumped then the following sequence of operations occurs: Control goes to CDUM1, a carriage return line feed is typed, an = is typed, calls are made in order to GETLOC, EXTRAC, and OCTOUT, control is finally returned to ENTER.

If two locations are specified by the user then the contents of the locations inclusively contained between them are dumped on the teletype, 8 values per line. The arithmetic is somewhat funny in this section of code with only the lower four octal digits being used in the calculations. The resulting output uses the same format as the Absolute Loader. Provision has been provided for the user to terminate the dumping by merely typing a (CTRL-C). Control returns to ENTER when the dump terminates.

REPLACE

This command is provided in order to allow the user to replace selected locations in the current coreimage with values entered from the keyboard.

The command dispatcher enters the replace module at CREP. The first octal number following the REP command is used as the starting address for the requested replacements. The remaining octal numbers are used as contents until the list is exhausted or until an invalid value is encountered (5 octal digits or more). Replacement proceeds by successive calls on GETLOC, PLACE, and REFER. Control is returned to ENTER.

EXIT

This command allows the user to return to the CPS file system without modifying the contents of -B.

The command dispatcher goes to CEXI. This routine simply does a JMP to location 07600 which is the start of the CPS system boot routine.

MAP

This command is used to list the names of the unloaded program entry points. If the letter A follows this command (or any word starting with an A such as ALL) then all defined program entry points are listed.

This command module is entered at CMAP. GWFLB is used to check for the presence of an A. UNLSW is set to -1 if only the unloaded file names are to be typed. A call is made to INITIB in order to initialize the output line buffer which will be used to hold the DRL output until a full output line has been accumulated. DRL is then started with a 2000 in the AC, faking the original OSR instruction used by DRL. DRL outputs its lines to RLTYP. This routine stores up a line and then checks to see if it should be typed depending upon whether the entry point has been defined and upon the setting of UNLSW. The command line buffer is used as the MAP buffer. When all of the entry points have been output by DRL control returns to the CMAP portion of the code. At this time UNLSW is checked. If only the unloaded entry points have been listed then a call to the SPACE command is faked in order to cause typing of the remaining number of unused pages of memory. Control eventually ends up at ENTER.

SPACE

This command is used to type out the number of pages of memory remaining available for loading programs into.

This module is entered at CSPA. The output buffer is initialized, UNLSW is set to 0, 4000 is placed in the AC and DRL is restarted. DRL passes its output to RLTYP which types the two line which DRL outputs. Control is finally passed to ENTER.

LIBRARY

This command is used to cause the commonly used FORTRAN support routines to be loaded without fuss, muss, bother or pain. The modules loaded are: IOH, FLOAT, INTEGER, UTILITY, ERROR and SUBSC. (These modules have their names preceded with a (.) in order to set them off from other file names. It is suggested that these names be preceded with an (*) instead. This means that they won't be listed every time someone wants a CPS index listing of non-system file names. The required modification is minor.)

The library module is entered at CLIB. The input program count PRGCNT is set equal to 6 and the GET command code is entered in such a way to cause a special list of file names to be used.

RESTART

Control is transferred to CRES. The resulting operation is described in the section titled, Loading Procedure.

COMMENT

This is a "do nothing" command. Control is passed directly back to ENTER via CCOM.

UNREFERENCE

This command is used to remove pages from the coreimage under construction. Pages are removed simply by clearing the appropriate location in the page reference table. The unreferenced page is not sprayed with a core constant (0 for the relocatable loader).

The command dispatcher transfers control to CUNR. DPOI is used to obtain a location on the page to be unreferenced. The appropriate location in the page reference table is calculated and zeroed. Control is then passed back to ENTER.

INDEX

This command causes a list of all loadable relocatable binary file names to be typed out.

This command starts at CIND. MVTW is used to transfer the names to be typed into W1 - W4 and MESAGE is used to type them. Control is returned to ENTER.

Memory Organization

The memory storage allocation after the relocatable loader has been loaded and started is as follows:

Locations	Contents
00000-00777	field 0 run time
01000-02777	input buffer
03000-06777	coreimage buffer
07000-07577	not used
07600-07777	CPS boot loader

Locations (Cont.)	Contents (Cont.)
10000-10177	constants and pointers
10200-11377	index
11400-11777	input line buffer
12000-12177	magnetic tape routine
12200-12377	PLIB, DPOSIT, TYPECH, CRLF, CCOM
12400-12577	GCNFB, OBI, EXCHR, GWFLB, SPRSPA, TERMTST
12600-12777	ENTER, SLRV, CGET, INDSRH
13000-13177	MESAGE, DPOI, OCTOUT
13200-13377	CDUM, CREP
13400-13577	LOADP, RLERROR, RLTYP, INITIB, SPACER, RDRIN
13600-13777	GETLOC, PLACE, EXTRAC, RLOAD, WPRESC, GTOUT2
14000-14177	GETIPT, GNXLOC, CIND
14200-14377	SETIPT, CMAP, CSPA, CLIB, NOTHER, MVTW, CEXI
14400-14577	REFER, CUNR, reference table
14600-14777	command dispatch table, library list
15000-15177	messages
15200-15377	CRES, CBUI
15400-15577	DISCBL
15600-15777	system entry point . . . gets clobbered, PACKIN

In the actual coreimage file the field 0 run time is found in locations 00200 through 00777. The field 1 run time is stored in locations 01000 through 01177.

Subroutine Descriptions

RTAPE
WTAPE } This is a version of the one page tape routine used in *FORTRAN and in *SABR. However it has been modified to work into and out of memory field 0 and to start the tapes moving in the direction determined by the last accessed tape block. This tape routine is kludged up and should be replaced by one which also provides the capability of leaving the tapes in motion after a tape operation. The tape settle time out delay segment would then have to check to see if the tapes are moving in order to see if the delay should be provided.

PLIB Used to enter and edit user typed command lines. Upon entry the input character store routine DPOSIT is initialized, a carriage return-line feed provided and the prefix character typed. This routine also recognizes the editing control characters and implements them.

DPOSIT This routine places input characters into the two page command line buffer. If the buffer capacity is exceeded an error message is typed and the current contents of the line buffer are lost.

TYPECH Used to type single ASCII characters. Returns with a clear AC.

CRLF Used to put out a single carriage return-line feed pair.

CCOM The code implementing the COMMENT command. Simply does a JMP to ENTER.

GCNFB Extracts characters from the input line buffer. It has three return points depending on whether the buffer is empty, whether an alphabetic (defined as non-octal digit) character has been obtained or whether an octal digit has been found. The extracted character is left in the AC upon return.

OBI Initializes the character count and buffer pointer for the EXCHR routine.

EXCHR Used to pull character off of the command line buffer. Has two return points. One for an end of buffer, the other for when a character has been found and is in the AC.

GWFLB Used to pull whole words off of the input line buffer. This routine has two returns, one for a successful extraction and the other for an end of line buffer. The extracted word is left in the registers W1 through W4 and is constructed using stripped 6 bit ASCII with two characters contained in each computer word. Trailing blanks are supplied for words of less than 8 characters and words longer than 8 characters are truncated to 8 characters. Words are strings of ASCII characters terminated by (blank) or (+). The terminator is not considered to be part of a word. Blanks preceding a word are ignored.

SPRSPA Sprays 4 words containing packed (blanks) wherever the user wants. Kind of a senseless routine.

TERMTST Used to see if the user has struck a (CTRL-C). Used wherever an attention interrupt feature has been incorporated. The code for this routine should be changed so that it forces the parity channel to 1.

ENTER This is the main internal re-entry point. The prefix character is set to the (.) and PLIB is called. PLIB returns when the user terminates his command line. The output routines are initialized by a call on OBI and the first word in the command line buffer is extracted using GWFLB. This word is contained in W1 - W4 and is next truncated to 3 characters. SLRV is used to search the command dispatch table and return with the address of the specified command in the AC. If the proper command was not found in the table an error message is typed and a jump to ENTER is made. After saving the command address, an indirect jump is made to the specified command module.

SLRV This routine is used to scan the list specified by the contents of the AC for the first occurrence of the contents of locations W1 - W4. The specified list is assumed to contain entries 5 words long. The first four words are compared against W1 - W4 and the contents of the fifth word are returned in the AC if a match is found. A special return is made if no match is found in the list. The list search is terminated when a zero is found as the first word in a list entry.

CGET This code implements the GET command. Calls are made on GWFLB in order to transfer the user supplied list of file names from the command line buffer into a name list. A maximum of 6 names can be transferred. More than 6 names or no names cause an error message to be typed. The location of the file name list is placed in the AC and control is transferred to INDSRH.

INDSRH This routine is used to search the index for names contained in the list specified by the contents of the AC when this routine was entered. If a name is not located in the index NOTHER is used to request a replacement name from the user. As each name is located its location on tape is stored on another list for later use in the loading operation. When the block number list has been constructed control is passed to LOADP in order to start loading the specified files.

MESAGE This is a modification of DEC's message type out routine. The address of the text to be typed is assumed to be in the AC when this routine is entered. The TEXT pseudo-op in MACRO-8 is used to generate the text for use with this routine.

DPOI This is a double precision octal input routine. Octal numbers in the range of 0 through 17777 are read from the input command line buffer using this routine. There are two return points. One is used when there are no remaining octal numbers in the current buffer. The other is used whenever a value has successfully been extracted. The resulting 13 bit value is returned in the link and AC. Numbers falling outside of the valid range are flagged as errors.

OCTOUT This routine is used to type out 4 digit octal numbers.

CDUM This implements the DUMP command. Two output formats are provided. If the user only requests one location to be dumped, a new line is started and the desired value is typed out. If several locations are to be typed then each line in the dump is provided an address header and up to 8 values are dumped per line.

The desired output location is forced to be in the current coreimage buffer by making a call to GETLOC with the required address contained in the link-AC. GETLOC returns with the address of the required location in the coreimage buffer contained in the AC. EXTRAC is called and returns with the contents of the specified location in the AC. OCTOUT is then called which types out the contents of the AC.

This routine could be improved upon with little effort.

CREP This is the REPLACE command implementation. The first octal value contained in the command line buffer is taken as a memory address. Trailing octal numbers are used as contents of succeeding locations starting with the specified address. The replacements continue until the input buffer is exhausted or until an invalid input value is detected. Control is returned to ENTER.

The following procedure is used in storing into the coreimage. GETLOC is called with the desired address contained in the link-AC. This causes the segment of the coreimage containing the desired location to be loaded into the coreimage buffer and the location in the buffer to be determined. Next PLACE is called to place the desired contents into the buffer. Finally the page just stored into is referenced through a call on REFER.

LOADP This section of code access the file block list constructed by INDSRCH and uses this list to control the loading process. After setting up the block list pointers, LOADP sets the DRL memory bank pointer to bank 0, initializes the input routine, and transfers control to DRL.

After DRL successfully loads a program it returns to LOADP. If another file entry is present LOADP loops on LOAD2 which then places a call to SETIPT and restarts DRL. When all entries in the block list have been exhausted control is returned to ENTER.

The LOAD2 point is also used for a re-entry when memory bank 0 has been found to be full and memory bank 1 is to be tried.

RLERROR This routine handles all error returns from DRL. DRL error number 2 (current memory bank too full) is accorded special treatment. All other error messages are typed and control is passed to ENTER.

When a type 2 error is encountered the current memory bank selector is checked to see which field is being loaded into. If it is field 1 then this implies (because field 0 is always tried first) that there is insufficient room in the machine for the current program. If the current field is field 0 then the memory bank selector location in DRL is set to bank one and a second try is made. The bank resetting and second try are accomplished by making a jump to LOAD2 with a 1 in the AC.

RLTYP This routine is used to buffer and type character strings generated by DRL. The input command line buffer and the associated support routines are used in the process. Whenever a carriage return is encountered RLTYP checks to see if UNLSW is -1 or 0. If it is 0 then the current line is typed and control is returned to DRL. If UNLSW is -1 then only unloaded entry points (flagged by a U) are typed out. Control is then returned to DRL.

INITIB Used to initialize the input line buffer.

SPACER Used to type a specified number of blanks.

RDRIN The support routine for Teletype input. Ignores leader-trailer and forces the parity channel to 1.

GETLOC A utility for determining the coreimage buffer address of a memory location specified in the address formed by the link and AC upon entry. If the desired address is not in core, the current buffer is written back into the coreimage and a coreimage segment containing the desired location is loaded into the coreimage buffer. When GETLOC returns, the address of the desired location in the coreimage buffer is contained in the AC. RPAGE is set to the page number which contains the desired address.

RPAGE is used by the page reference routine, REFER. Special provision is included for properly handling the memory field 0 run time addresses.

PLACE
Uses the value of the AC upon entry as a pointer into the coreimage buffer and places the contents of HOLD into this location.

EXTRAC
Extracts the contents of the location specified by the AC from the coreimage buffer. Returns with the desired value in the AC.

RLOAD
The DRL comes here when it wishes to deposit values into the coreimage under development. DRL does not come here when modifying the run time tables. This routine extracts the desired address from DRL and makes calls to GETLOC, PLACE, and REFER before returning to DRL for another number.

WPRESC
This routine writes the present contents of the coreimage buffer onto tape.

GTOUT2
This is part of the BUILD command code which had to be segmented because of space limitations. This section sets the communication area file update switches and exits to the CPS system boot.

GETIPT
This routine is called whenever DRL desires an input frame. The input frames are contained in binary files produced by *SABR. The GNXLOC routine is used to check if the next input frame can be extracted from the current input buffer or whether the buffer has to be refilled. GETIPT returns to DRL with the resulting SABR binary frame in the AC.

GNXLOC
This routine checks to see if the input buffer has been exhausted. If not, it returns with GETLO pointing at the next available data word. If the input buffer has been exhausted, 8 additional blocks from the current input file are loaded and GETLO is reset to point at the buffer start.

It should be noted that GNXLOC always attempts to fill the input buffer with 8 blocks independent of input file size. DRL is depended upon to terminate whenever a file's contents have been exhausted. In future versions the size of each file should be included in the index and GNXLOC should manage the input buffer so that it never attempts to read past the end of a file. It should also check to see if DRL is attempting to read past the end of a file (however, there is no record of this happening).

CIND

This routine performs the operations required for the INDEX command. All of the entries in the loader index are listed on the Teletype. Striking a (CTRL-C) causes an attention interrupt and returns control to ENTER.

SETIPT

This routine is used to setup new input files so that the first data request made of the input buffer routines will cause the buffer to be loaded from a new file. SETIPT also checks to see if the file requested is presently in core. If it is, a tape read is not forced. This is useful when trying to load a program into bank 0 and having to retry loading it into bank 1.

CMAP

This subprogram handles the work for the MAP command. If this command is followed by a word starting with an A UNLSW is set to 0. Otherwise it is set to -1. Next, the input buffer routines are initialized and control is passed to DRL with the octal number 2000 in the AC.

The purpose of the 2000 in the AC is to fool DRL into behaving as if it had just read the switch register and found bit 1 set. DRL proceeds to put out all of its symbol definitions passing its output to RLTYP.

After DRL is finished it returns to the CMAP code. If UNLSW is -1 the number of pages remaining in each memory bank is typed out by faking a call to the SPACE command. Otherwise control is passed to ENTER.

CSPA

CSPA implements the SPACE command. First the line buffer is initialized, UNLSW cleared and a 4000 is placed in the AC. Finally the CMAP command is jumped into at the call to DRL. The 4000 in the AC makes DRL think that bit 0 of the switch registers is set.

CLIB The LIBRARY command support code. The number of input file names is set to 6, the address of the library table is placed in the AC and INDSRH is jumped to. All we are doing here is faking the input from a GET command and then jumping into the GET command support code. The use of a library table containing file names instead of file locations makes modifications in file positions less traumatic.

NOTHER The code used by INDSRH to request a replacement name from the user if a file name he has specified is not present in the index. No files are loaded until all designated files have been found in the index. Typing a lone carriage return kills the command and returns control to ENTER.

MVTW Used to move a block of 4 words from the address specified in the AC into W1 through W4.

CEXI The EXIT command. Simply does a jump to the CPS system boot located in the top page of memory bank 0.

REFER Maintains the page reference table. This table is used to keep track of which memory pages have been stored into and which have not. If a page has had something stored in it, it is said to be referenced. The reference table is structured, inefficiently, using 1 computer word per page of memory. An entry of 0 is associated with an unreference page, and an entry of 1 is associated with a referenced page.

CUNR This implements the UNREFERENCE command. The octal number following this command is taken to be a memory address in the page to be unreferenced. The appropriate entry in the page reference table is replaced with a 0.

CRES The RESTART command. This subprogram is also used as part of the relocatable loader startup. DRL entered at 6200 so that it initializes itself, the buffers are zeroed, the coreimage area on unit 0 is sprayed with zeros. (The coreimage area on unit 0 consists of blocks 0200 through 0377.) Finally, the page reference table is set up and control passed to ENTER.

CBUI Implements the BUILD command. The input line is checked for a starting address. If one has been supplied it is used, otherwise the first entry in the run time tables is used as the starting address (this is done as a service to FORTRAN users). The WPRESC subroutine is next called to write out the contents of the coreimage buffer onto tape. The memory bank 0 run time is written out next.

The DISCBL routine is called twice in order to build up the desired coreimage module on tape unit 1 in -B. A coreimage descriptor block is then formed and placed in front of this module. The appropriate CA file pointers are set and a return is made to CPS.

DISCBL This is used to compress the coreimage segment contained in the lower 4-K of memory. This routine is entered with the AC containing a pointer into the page reference table. The coreimage is worked on a memory bank at a time. The space held by unreferenced pages is removed and the coreimage descriptor block is built up. This routine must be initialized before being called the first time.

This routine can and should be improved upon.

SYSTEM ENTRY CPS starts the loader here. At the time it is started the entire coreimage module for the relocatable loader is contained in the lower memory bank. This is done in order to preserve the integrity of the CPS indexes contained in the upper memory bank. The unit 1 CPS index is scanned and all entries for *SABR type files are transferred to the area to be used by the relocatable loader for its index. The relocatable loader code is then transferred to the upper memory bank and activated. Next, the field 0 and field 1 run times are disposed of, with the field 1 run time being placed on tape. Control is passed to CRES.

PACKIN Used to pack the designated CPS index and place the entries for *SABR type files into the relocatable loader index area.

Assembly Instructions

The following procedure is used in constructing the core-image file containing the relocatable loader.

```
RUN *ASM  L1+L2+L3+L4+L5+L6  2=L7+L8+L9+L10+$
SAVE -B LDRSUP
RUN *ASM LDRTAPE
LOAD A
GET LOADER
GET LDRSUP
GET -B
BUILD     (the starting address is 05600)
```

When control returns to the system, -B will contain the desired coreimage module.

Files L1 through L10 contain the support code, patches and run time for the desired loader module. File LDRTAPE contains a hashed up one page tape routine for use in magnetic tape operations. LOADER is a file generated using PAPERBIN and contains the source binary for the DEC relocatable loader.

```
                CSUM=RTAPE
                INTS=6147
                IAAC=6171
                IACB=6131
                ICON=6141
                /
                /
                /TAPE ROUTINE FOR READING
                /AND WRITING SIZE 128 BLOCKS ON
                /THE LINC 3
                /
                /THIS IS TO REPLACE THE ONE
                /I STOLE FROM DEC
                /
                *2000
                /
2000   0000   RTAPE, 0
2001   7300     CLA CLL
2002   1200     TAD RTAPE /MOVE POINTER
2003   3206     DCA WTAPE
2004   1143     TAD M4000 /TO FORCE FUNCT = 3
2005   5210     JMP INTO
2006   0000   WTAPE, 0
2007   7300     CLA CLL
2010   1144   INTO, TAD C4003 /WRITE WITH SWITCH
2011   3145     DCA FUNCT
2012   1146     TAD BLN /LAST BLOCK
2013   7141     CIA CLL
2014   1606     TAD I WTAPE /L=1 GO FWD
2015   7210     CLA RAR
2016   3142     DCA TDIR
2017   1606     TAD I WTAPE /BLOCK NUMBER
2020   3146     DCA BLN
2021   2206     ISZ WTAPE
2022   1606     TAD I WTAPE /NUMB OF BLOCKS
2023   7041     CIA
2024   3147     DCA NUMB
2025   2206     ISZ WTAPE
2026   1606     TAD I WTAPE /UNIT
2027   7112     RTR CLL /PUT INTO BIT 0
2030   1321     TAD C2 /TO SET SEARCH
2031   3150     DCA UNIT
2032   2206     ISZ WTAPE
2033   1606     TAD I WTAPE /CORE LOC
2034   3151     DCA LOC
2035   2152    ISZ CNTR /TIMEOUT TO ALLOW TAPE
2036   5233    JMP .-3  /DRIVE TO SETTLE DOWN
2037   2206     ISZ WTAPE
2040   7120     STL /LINC = 1, FIRST PASS
2041   2146   SERCHA, ISZ BLN /1'S COMPL ON TAPE
2042   3200   SERCHB, DCA CSUM /ZERO CHECK SUM
2043   1324     TAD C7600
2044   3152     DCA CNTR /WORD COUNT
2045   1150     TAD UNIT /NOW SET SERCH
```

```
2046  6141     ICON
2047  7201     CLA IAC /TO START MOTION BACKWARDS
2050  1142     TAD TDIR
2051  7430     SZL /SEE IF MOVING
2052  6141     ICON
2053  1347     TAD CS /TO CLEAR INTS
2054  6141   B, ICON
2055  4363   A, JMS WAIT /FOR BLK INT
2056  7500     SMA /ONLY NEG VALID
2057  7120     STL /POS WANT FWD FOR BLK 0
2060  1146     TAD BLN
2061  7650     SNA CLA /L=1, WANT FWD
2062  5276     JMP THERE /ON BLOCK
2063  6147     INTS /WANT M0
2064  7010     RAR /M0 TO L, L TO BIT 0
2065  0143     AND M4000
2066  7460     SZA SNL
2067  5255     JMP A /WANT FORE, GOT FORE
2070  7020     CML
2071  7520     SNL SMA
2072  5255     JMP A /BACKWARDS
2073  6141     ICON /STOP
2074  7001     IAC /BIT 0 IS OK HERE
2075  5254     JMP B /CHANGE MOTION
2076  6147   THERE, INTS /WANT M1
2077  7012     RTR
2100  7620     SNL CLA /ON AND FORDW
2101  5255     JMP A /ON AND GOING BACK, REVERSE
2102  1145     TAD FUNCT
2103  6141     ICON /SET BLOCK MODE
2104  7500     SMA /S IF TO WRITE
2105  5336     JMP RDATA
2106  1321     TAD C2 /TO GET 5
2107  6141     ICON /TURN WRITERS ON
2110  7200   WRITE, CLA
2111  6201   CDFA, CDF
2112  1551     TAD I LOC
2113  6211     CDF 10
2114  6161     IACB /AC TO LINC REG
2115  1200     TAD CSUM
2116  3200     DCA CSUM
2117  4363     JMS WAIT /PUT IT OUT
2120  2151     ISZ LOC
2121  0002   C2, 2 /STORAGE, THIS HAS NO EFFECT
2122  2152     ISZ CNTR /DONE?
2123  5310     JMP WRITE /NO
2124  7600   C7600, 7600 /ALSO A CLA
2125  1200     TAD CSUM
2126  6161     IACB /WRITE CHECK SUM
2127  4363     JMS WAIT
2130  4363     JMS WAIT /ALLOW ACTUAL WRITE OF CS
2131  7300   WAIT2, CLA CLL
2132  2147     ISZ NUMB /ALL BLOCKS DONE?
2133  5241     JMP SERCHA /NO
2134  6141     ICON /STOP
```

PAGE 03

```
2135  5606     JMP I WTAPE /GO HOME
2136  4363  RDATA, JMS WAIT /GUARD
2137  4363  RDTA, JMS WAIT
2140  1200     TAD CSUM
2141  3200     DCA CSUM
2142  6171     IAAC /GET AGAIN
2143  6201  CDFB, CDF
2144  3551     DCA I LOC /PUT IN CORE
2145  6211     CDF 10
2146  2151     ISZ LOC
2147  0006  C6, 6 /STORAGE, NO EFFECT
2150  2152     ISZ CNTR
2151  5337     JMP RDTA /CONTINUE
2152  4363     JMS WAIT /CSUM
2153  7041     CIA
2154  1200     TAD CSUM
2155  7650     SNA CLA /MAYBE BAD
2156  5331     JMP WAIT2 /THIS IS OK
2157  1324     TAD C7600
2160  1151     TAD LOC /FIX BACK
2161  3151     DCA LOC
2162  5242     JMP SERCHB /TRY AGAIN
2163  0000  WAIT, 0
2164  7300  W1, CLA CLL
2165  6147  C7, INTS
2166  7700     SMA CLA /TAPE?
2167  5364     JMP W1
2170  1365     TAD C7
2171  6141     ICON /CLEAR INTS
2172  7300     CLA CLL
2173  6171     IAAC /GET FROM TAPE
2174  5763     JMP I WAIT
            /
            /
            *140
0140  2111  LCDFA, CDFA
0141  2143  LCDFB, CDFB
0142  0000  TDIR, 0
0143  4000  M4000, 4000
0144  4003  C4003, 4003
0145  0000  FUNCT, 0
0146  1777  BLK, 1777
0147  0000  NUMB, 0
0150  0000  UNIT, 0
0151  0000  LOC, 0
0152  0000  CNTR, 0
```

```
RUN *ASM L1+L2+L3+L4+L5+L6 2=L7+L8+L9+L10+6 P=NNN#NSL
```

```
MACRO8 ASSEMBLER (AN229)

PAGE 01


                /PAGE 0 OF THE SABR LOADER
                /
                *20
                /
0020    0000    WHBNK, 0 /BANK FOR GETLOC
0021    0000    WHLOC, 0 /ADDRS FOR GETLOC
0022    0000    RPAGE, 0 /CURRENT PAGE BEING REFFED
0023    0004    CURSTR, 4 /CURRENT STARTING PAGE IN CORE
0024    0000    GETTEM, 0 /A VERY VOLITLE TEMP
0025    0000    HOLD, 0 /VALUE TO BE PUT IN CORE BUFFER
0026    2000    RTAPE, 2000
0027    2006    WTAPE, 2006
0030    0000    PLIST, 0 /BUFF IPT TEMP
0031    1400    BFSTRT, 1400 /START OF LINE BUFFER
0032    6400    NBFSTRT, -1400 /NEG OF BFSTRT
0033    6200    NBUFTOP, -1600 /NEG OF BUFF TOP
0034    0256    PREFIX, 256 /HOLDS BUFF IPT PREFIX CHAR
0035    0000    BCRLOC, 0 /CURRENT LOC INPUT TO BUFFER
0036    0037    WLOC, W1
0037    4040    W1, 4040
0040    4040    W2, 4040
0041    4040    W3, 4040
0042    4040    W4, 4040
0043    0000    W5, 0 /TERMINATOR
0044    0000    UNLSW, -0
0045    0000    PRGCNT, 0
0046    7251    LOADGO, RESTRT /ENTRY IN SABR LOADER
0047    0000    CSTRHG, 0
0050    0000    CSTRLO, 0
0051    0000    CENDLO, 0
0052    0000    CENDHG, 0
0053    0000    CDUMT,  0
0054    0000    CDEMPT, 0
0055    0000    OUTCNT, 0
0056    0000    WRDCNT, 0
0057    0000    INDPT1, 0
0060    0000    INDPT2, 0
0061    7700    N100, -100
0062    7774    N4, -4
0063    0004    C4, 4
                /
                PAGE
```

PAGE 02

```
                *2200
                /PLIB...PUT LINE IN BUFFER
                /
2200   0000     PLIB, 0
2201   1031       TAD BFSTRT /INIT BUFF PTR
2202   3035       DCA BCRLOC
2203   4335       JMS CRLF
2204   1034       TAD PREFIX
2205   4327       JMS TYPECH /TYPE PREFIX
2206   4777     PLIBLA, JMS RDRIN /GET CHARACTER
2207   3030       DCA PLIBT /TEMP
2210   1030       TAD PLIBT
2211   1376       TAD (-220 /CONT P ?
2212   7450       SNA
2213   5240       JMP CNTRLP
2214   1375       TAD (-117 /BACK ARROW
2215   7450       SNA
2216   5243       JMP BACARO
2217   1374       TAD (-40 /RUBOUT
2220   7450       SNA
2221   5246       JMP RUBOUT
2222   1373       TAD (165 /LF
2223   7450       SNA
2224   5260       JMP LNFEED
2225   1372       TAD (-3 /CR
2226   7450       SNA
2227   5272       JMP CARRET
2230   1374       TAD (-40 /MINUS
2231   7650       SNA CLA
2232   5277       JMP MINUS
2233   1030     PLIB2, TAD PLIBT
2234   4327       JMS TYPECH
2235   1030       TAD PLIBT
2236   4315       JMS DPOSIT
2237   5206       JMP PLIBLA
                /
2240   4777     CNTRLP, JMS RDRIN   /GET FOLLOWING CHARACTER
2241   3030       DCA PLIBT
2242   5233       JMP PLIB2
                /
2243   1371     BACARO, TAD (337   /ECHO A BACKARROW
2244   4327       JMS TYPECH
2245   5201       JMP PLIB+1          / RE-INITIALIZE AND START OVER
                /
2246   1370     RUBOUT, TAD (334
2247   4327       JMS TYPECH          /TYPE BACK-SLASH
2250   7240       CLA CMA
2251   1035       TAD BCRLOC          /DECREMENT BUFFER POINTER BY 1
2252   1032       TAD NBFSTR
2253   7510       SPA                 / SEE IF SOMETHING TO DELETE
2254   7200       CLA
2255   1031       TAD BFSTRT          / RESET BUFFER POINTER
2256   3035       DCA BCRLOC
2257   5206       JMP PLIBLA
                /
```

```
2260  4335  LNFEED, JMS CRLF        /ECHO CURRENT LINE
2261  1367    TAD (275
2262  4327    JMS TYPECH
2263  4766    JMS OBI /INIT OUT ROUTINE
2264  4765    JMS GONFB
2265  5206    JMP PLIBLA
2266  7410    SKP
2267  1364    TAD (260
2270  4327    JMS TYPECH
2271  5264    JMP .-5
              /
2272  1363  CARRET, TAD (240 /TERMINATE BUFFER WITH BLANK
2273  4315    JMS DPOSIT
2274  1362    TAD (200
2275  4327    JMS TYPECH
2276  5600    JMP I PLIB /GO BACK
              /
2277  1361  MINUS, TAD (255 /MINUS
2300  4327    JMS TYPECH         /ECHO
2301  4777    JMS RDRIN
2302  1360    TAD (-215          /TEST TO SEE IF NEXT CHAR IS CR
2303  7440    SZA                /SKIP IF IT IS
2304  5307    JMP MINUS2
2305  4335    JMS CRLF           /GET NEXT LINE
2306  5206    JMP PLIBLA
2307  1357  MINUS2, TAD (215 /CR   HAVE TO SAVE - AND CHARACTER
2310  3030    DCA PLIBT
2311  1361    TAD (255 /MINUS
2312  4315    JMS DPOSIT
2313  1030    TAD PLIBT
2314  5207    JMP PLIBLA+1
              /
2315  0000  DPOSIT, 0           /PLACE AC INTO BUFFER
2316  3435    DCA I BCRLOC
2317  2035    ISZ BCRLOC
2320  1035    TAD BCRLOC
2321  1033    TAD NBUFTOP     /CHECK FOR OVERFLOW
2322  7710    SPA CLA
2323  5715    JMP I DPOSIT
2324  1756    TAD MIDF        /TOO MANY
2325  4755    JMS MESAGE
2326  5201    JMP PLIB+1
              /
2327  0000  TYPECH, 0         /MERELY DRIVES PRINTER
2330  6046    TLS
2331  6041    TSF
2332  5331    JMP .-1
2333  7200    CLA
2334  5727    JMP I TYPECH
              /
2335  0000  CRLF, 0            /TYPE CR-LF
2336  7200    CLA
2337  1357    TAD (215
2340  4327    JMS TYPECH
2341  1354    TAD (212
PAGE 04

2342  4327     JMS TYPECH
2343  5735     JMP I CRLF
              /
              /CCOM...COMMENT COMMAND
              /
2344  5753    CCOM, JMP ENTER    / DO NOTHING
              /
2353  2600    PAGE
2354  0212
2355  3000
2356  5012
2357  0215
2360  7563
2361  0255
2362  0200
2363  0240
2364  0260
2365  2400
2366  2420
2367  0275
2370  0334
2371  0337
2372  7775
2373  0165
2374  7740
2375  7661
2376  7560
2377  3527
```

```
                *2400
                /
                /GCNFB...GET CHARACTER OR NUMBER FROM BUFFER
                /
                /     R  ET HERE IF BUFF EMPTY
                /     R  ET HERE IF ALPH AC=ASCII
                /      RET HERE IF OCTAL AC=0-7
                /
2400   0000     GCNFB, 0
2401   4231       JMS EXCHR /BASIC BUFFER OUTPUT ROUTINE
2402   5600       JMP I GCNFB /EMPTY
2403   2200       ISZ GCNFB
2404   1377       TAD (-260 /ASCII IN AC
2405   7510       SPA
2406   5216       JMP GCNFB2 /NOT OCTAL
2407   1376       TAD (-10
2410   7500       SMA
2411   5215       JMP GCNFB1 /NOT OCTAL
2412   1375       TAD (10
2413   2200       ISZ GCNFB /OCTAL RET
2414   5600       JMP I GCNFB
2415   1375     GCNFB1, TAD (10
2416   1374     GCNFB2, TAD (260
2417   5600       JMP I GCNFB
                /
2420   0000     ODI, 0 /INIT EXCHR ROUTINE
2421   7200       CLA
2422   1031       TAD BFSTRT
2423   3245       DCA OPTR
2424   1032       TAD NBFSTRT
2425   1035       TAD BCRLOC
2426   7041       CIA
2427   3244       DCA OPCTR
2430   5620       JMP I ODI
                /
2431   0000     EXCHR, 0 /EXTRACT A CHAR FROM IPT BUFF
2432   7200       CLA          /IF EMPTY ALWAYS RET, CALL+1
2433   1244       TAD OPCTR    /IF NOT EMPTY RET, CALL+2
2434   7650       SNA CLA /0=EMPTY
2435   5631       JMP I EXCHR
2436   2231       ISZ EXCHR
2437   1645       TAD I OPTR
2440   2245       ISZ OPTR
2441   2244       ISZ OPCTR
2442   7000       NOP
2443   5631       JMP I EXCHR
                /
2444   0000     OPCTR, 0
2445   0000     OPTR, 0
                /
                /GWFLB...GET WORD FROM LINE BUFFER
                /
                /  RET HERE IF EOL
                /  RET HERE W WORD IN W1-W4
                /
```

```
2446  0000  GWFLB, 0
2447  7300    CLA CLL
2450  1036    TAD WLOC
2451  4332    JMS SPRSPA /PUT SPACES IN W
2452  1036    TAD WLOC
2453  3344    DCA GWPTR
2454  1373    TAD (-4 /WORD COUNT
2455  3345    DCA GWCNT
2456  3347    DCA GWFLHS /SET SW TO LEFT SIDE
2457  4200    JMS GCNFD /GET CHARACTER
2460  5646    JMP I GWFLB /RETURN FOR EMPTY
2461  7410    SKP /ALPHA
2462  1374    TAD (260 /OCTAL
2463  1372    TAD (-240 /SPACE?
2464  7450    SNA
2465  5257    JMP .-6 /IGNORE PRECEEDING BLANKS  (correction made)
2466  2246    ISZ GWFLB /ADV RET
2467  5277    JMP GWFLB2 /JUMP INTO STORAGE LOOP
            /
2470  4200  GWFLB3, JMS GCNFD /GET CHAR
2471  7402    HLT /CANT GET HERE...! ?
2472  7410    SKP
2473  1374    TAD (260 /OCTAL RET
2474  1372    TAD (-240 /SPACES TERMINATE NOW
2475  7450    SNA
2476  5646    JMP I GWFLB
2477  1371  GWFLB2, TAD (-13 /+TERMINATES TOO
2500  7450    SNA
2501  5646    JMP I GWFLB
2502  1370    TAD (253
2503  3346    DCA GWFLBT /TEMP
2504  1345    TAD GWCNT
2505  7650    SNA CLA /S IF STILL ROOM
2506  5270    JMP GWFLB3 /IGNORE
2507  1346    TAD GWFLBT
2510  0367    AND (77 /STRIP
2511  2347    ISZ GWFLHS /L OR R
2512  5322    JMP GWLH /LEFT
2513  1366    TAD (-40 /DO RIGHT
2514  1744    TAD I GWPTR
2515  3744    DCA I GWPTR
2516  2344    ISZ GWPTR
2517  2345    ISZ GWCNT
2520  7000    NOP
2521  5270    JMP GWFLB3
2522  7106  GWLH, RTL CLL
2523  7006    RTL
2524  7006    RTL
2525  1365    TAD (40
2526  3744    DCA I GWPTR
2527  7240    CLA CMA
2530  3347    DCA GWFLHS
2531  5270    JMP GWFLB3
            /
2532  0000  SPRSPA, 0 /SPRAYS 4 BLANK WORDS
```

PAGE 07

```
2533  3344     DCA GWPTR
2534  1373     TAD (-4
2535  3345     DCA GWCNT
2536  1364     TAD (4040
2537  3744     DCA I GWPTR
2540  2344     ISZ GWPTR
2541  2345     ISZ GWCNT
2542  5336     JMP .-4
2543  5732     JMP I SPRSPA
              /
2544  0000    GWPTR, 0
2545  0000    GWCNT, 0
2546  0000    GWFLST, 0
2547  0000    GWFLHS, 0
              /
2550  0000    TERMTST, 0 /SEE IF USER WANTS TO STOP.. CHECK FOR CTRL-C
2551  7200      CLA
2552  6034      KRS         AND (177
2553  1363      TAD (-203}  TAD (-3    do
2554  7640      SZA CLA
2555  5750      JMP I TERMTST
2556  6032      KCC
2557  5762      JMP ENTER
              /
2562  2600    PAGE
2563  7575
2564  4040
2565  0040
2566  7740
2567  0077
2570  0253
2571  7765
2572  7540
2573  7774
2574  0230
2575  0010
2576  7770
2577  7520
```

```
            /
            /RELCACATBLE BINARY LOADER MAIN ENTRY
            /(NOT THE SYSTEM START POINT)
            /
            *2600
            /
2600  7300  ENTER, CLA CLL
2601  1377    TAD (256 /.
2602  3034    DCA PREFIX  /RESET PREFIX TO .
2603  4776    JMS PLIB    / GET COMMAND LINE
2604  4775    JMS OBI /INIT OUTPUT ROUTINES
2605  4774    JMS GWFLB /GET COMMAND
2606  5203    JMP .-3 /NOTHING THERE
2607  1373    TAD (4040 /CUT COMMAND TO 3 CHARS
2610  3042    DCA W4
2611  1373    TAD (4040
2612  3041    DCA W3
2613  1040    TAD W2
2614  0372    AND (7700
2615  1371    TAD (40
2616  3040    DCA W2
2617  1770    TAD COMTAB /START OF COMMAND TABLE
2620  4230    JMS SLRV
2621  5225    JMP ENT2 /NOT HERE
2622  3224    DCA .+2   /AC = ADDRESS OF COMMAND MODULE
2623  5624    JMP I .+1 /GO TO IT
2624  0000    0000 /RESERVED
            /
2625  1767  ENT2, TAD MILCO
2626  4766    JMS MESAGE /ILLEGAL COMMAND
2627  5200    JMP ENTER /TRY AGAIN
            /
            /
            /SLRV...SCAN LIST RETURN VALUE
            /        RET HERE AC=0 IF CAN'T FIND
            /        RET HERE AC=NUMBER IN FIFTH LOC IF FOUND
            /
            SLRVPT=11 /AUTO INDEX
            /
2630  0000  SLRV, 0    /USED FOR COMMANDS AND FOR INDEX SEARCHES
2631  1365    TAD (-1
2632  3011    DCA SLRVPT /SAVE LIST ADDRESS
2633  1411    TAD I SLRVPT /GET 1ST WORD
2634  7450    SNA /0 TERMINATES LIST
2635  5630    JMP I SLRV /NO MATCH
2636  7041    CIA
2637  1037    TAD W1 /WORKS OUT OF W1-W4
2640  7640    SZA CLA
2641  5264    JMP SLRVA /NOPE
2642  1411    TAD I SLRVPT
2643  7041    CIA
2644  1040    TAD W2
2645  7640    SZA CLA
2646  5265    JMP SLRVB /MISSED ON 2
2647  1411    TAD I SLRVPT
```

```
2650  7041     CIA
2651  1041     TAD W3
2652  7640     SZA CLA
2653  5266     JMP SLRVC
2654  1411     TAD I SLRVPT
2655  7041     CIA
2656  1042     TAD W4
2657  7640     SZA CLA /S IF WE MATCH
2660  5267     JMP SLRVD
2661  1411     TAD I SLRVPT /GET CONSTANT
2662  2230     ISZ SLRV /ADV RET
2663  5630     JMP I SLRV /GO BACK
              /
2664  2011    SLRVA, ISZ SLRVPT
2665  2011    SLRVB, ISZ SLRVPT
2666  2011    SLRVC, ISZ SLRVPT
2667  2011    SLRVD, ISZ SLRVPT
2670  5233     JMP SLRV+3
              /
              /
              /CGET...GET COMMAND
              /
2671  7240    CGET, CLA CMA
2672  1764     TAD CGEIL /START OF NAMES LIST TO BE CONSTRUCTED
2673  3010     DCA 10 /AUTO IND
2674  3045     DCA PRGCNT /COUNT # PROGS SET TO 0 TO START
2675  4774    CGET3, JMS GWFLB /GET A NAME
2676  5320     JMP CGETD /DONE RET
2677  2045     ISZ PRGCNT /ADD 1 TO COUNT
2700  1363     TAD (-7 /STOP AT 6 NAMES
2701  1045     TAD PRGCNT
2702  7710     SPA CLA
2703  5307     JMP CGET2 /CAN FIT IN
2704  1762     TAD MTMN /TOO MANY NAMES
2705  4766     JMS MESAGE
2706  5200     JMP ENTER
              /
2707  1037    CGET2, TAD W1 /SAVE INPUT INTO OUTLIST
2710  3410     DCA I 10
2711  1040     TAD W2
2712  3410     DCA I 10
2713  1041     TAD W3
2714  3410     DCA I 10
2715  1042     TAD W4
2716  3410     DCA I 10
2717  5275     JMP CGET3 /GET MORE
              /
2720  3410    CGETD, DCA I 10 /0 TERMINATES NAME LIST
2721  1045     TAD PRGCNT
2722  7640     SZA CLA /0 IS NOT ENOUGH
2723  5330     JMP .+5
2724  7200    TOFEW, CLA
2725  1761     TAD MTOFE /TOOFEW OPERANDS
2726  4766     JMS MESAGE
2727  5200     JMP ENTER
```

```
                /
2730  1764        TAD CGETL
2731  5332        JMP INDSRH /SEARCH INDEX <- CAN REMOVE THIS
                /
                /
                /ROUTINE TO SEARCH THE INDEX FOR NAMES
                /AND TO CONSTRUCT THE LOAD LIST
                /
2732  3057      INDSRH, DCA INDPT1 /FROM LIST
2733  1764        TAD BLNLST /BLOCK # LIST TO BE USED BY LOADER
2734  3060        DCA INDPT2
2735  1457      INDSH2, TAD I INDPT1 /0 TERMINATES
2736  7650        SNA CLA
2737  5760        JMP LOADP /GO START LOADING NOW
2740  1057        TAD INDPT1
2741  4757        JMS MVTW /MOVE NAME TO W1-W4
2742  1356      INDSH3, TAD (200 /STRT OF INDEX
2743  4230        JMS SLRV   /SEARCH INDEX FOR ENTRY IN W1-W4
2744  5755        JMP NOTHER /NOT HERE
2745  3460        DCA I INDPT2 /SAVE BLK ... FOUND IT
2746  2060        ISZ INDPT2   /ADVANCE TAPE LOCATION LIST POINTER
2747  1354        TAD (4
2750  1057        TAD INDPT1 /CHANGE INDPT1 TO POINT TO
2754  0004        DCA INDPT1 /NEXT NAME IN USER SUPPLIED LIST
2755  4260
2756  0200
2757  4275
2760  3400
2761  5040
2762  5000
2763  7771
2764  4676
2765  7777
2766  3000
2767  5025
2770  4600
2771  0040
2772  7700
2773  4040
2774  2446
2775  2420
2776  2200
2777  0256
2751  3057
2752  5335        JMP INDSH2
                /
                /
                PAGE
                *3000   MESAGE IS MODIFIED DEC 8-16-0
                /
3000  0000  MESAGE, 0 /AC=ADDR OF TEXT
3001  1253    TAD MESM1 /-1
3002  3010    DCA 10 /USES AUTO-INDEX
3003  1410    TAD I 10
3004  3215    DCA MSRGHT /SAVE PACKED WORD
3005  1215    TAD MSRGHT
3006  7012    RTR
3007  7012    RTR
3010  7012    RTR
3011  4216    JMS TYPCH /TYPE LH
3012  1215    TAD MSRGHT
3013  4216    JMS TYPCH /TYPE RH
3014  5203    JMP MESAGE+3
3015  0000  MSRGHT, 0 /TEMP
3016  0000  TYPCH, 0 /TYPES
3017  0244    AND MASK77
3020  7450    SNA /0 TERMINATES
3021  5600    JMP I MESAGE /RETURN
3022  1245    TAD MESM40 /-40
3023  7500    SMA /<40?
3024  5227    JMP .+3
3025  1246    TAD C340MES /340
3026  5242    JMP MIP
3027  1247    TAD M3MES /-3
3030  7440    SZA /LFD
3031  5234    JMP .+3
3032  1250    TAD C212MES
3033  5242    JMP MIP
3034  1254    TAD M2MES /-2
3035  7440    SZA /CR?
3036  5241    JMP .+3
3037  1251    TAD C215MES
3040  5242    JMP MIP
3041  1252    TAD C245MES
3042  4777  MIP, JMS TYPECH
3043  5616    JMP I TYPCH /RET
            /
            /
3044  0077  MASK77, 77
3045  7740  MESM40, -40
3046  0340  C340MES, 340
3047  7775  M3MES, -3
3050  0212  C212MES, 212
3051  0215  C215MES, 215
3052  0245  C245MES, 245
3053  7777  MESM1, -1
3054  7776  M2MES, -2
            /
            /DPOI...DOUBLE PRECISION OCTAL INPUT
            /        RET HERE IF BUFFER EMPTY
            /        RET HERE NUMBER IN AC AND LINK
            /
3055  0000  DPOI, 0
```

```
PAGE 12

3056  7300     CLA CLL
3057  3333     DCA HIGHV /CLR HIGH
3060  3332     DCA LOWV /CLR LOW
3061  1376     TAD (-5 /MAX # DIGITS
3062  3327     DCA DPOIC1
3063  4775     JMS GCNFB /GET NUMBER
3064  5314     JMP DPOIR1 /BUFF END NO NMBER
3065  5263     JMP .-2 /IGNORE PRECEEDING ALPHA
3066  2255     ISZ DPOI /NUMBER
3067  3331   DPOIL, DCA DPOIT /TEMP
3070  1374     TAD (-3 /SHIFT 3 BITS
3071  3330     DCA DPOIC2
3072  7100   DPOIS, CLL /1 BIT AT A TIME
3073  1332     TAD LOWV
3074  7004     RAL
3075  3332     DCA LOWV
3076  1333     TAD HIGHV
3077  7004     RAL
3100  3333     DCA HIGHV
3101  2330     ISZ DPOIC2
3102  5272     JMP DPOIS /MORE TO GO
3103  1332     TAD LOWV
3104  1331     TAD DPOIT
3105  3332     DCA LOWV /NOW UPDATED
3106  4775     JMS GCNFB
3107  5314     JMP DPOIR1 /END
3110  5314     JMP DPOIR1 /END
3111  2327     ISZ DPOIC1 /NUMB IN AC
3112  5267     JMP DPOIL /NOT TOO MANY YET
3113  5323     JMP MINV /OOPS
3114  7300   DPOIR1, CLA CLL
3115  1333     TAD HIGHV
3116  7010     RAR
3117  7640     SZA CLA
3120  5323     JMP MINV
3121  1332     TAD LOWV
3122  5655     JMP I DPOI
             /
3123  7300   MINV, CLA CLL
3124  1773     TAD MINUS /BAD NUMBER
3125  4200     JMS MESAGE
3126  5772     JMP ENTER
             /
3127  0000   DPOIC1, 0
3130  0000   DPOIC2, 0
3131  0000   DPOIT, 0
3132  0000   LOWV, 0
3133  0000   HIGHV, 0
             /
             /
             /OCTAL OUTPUT ROUTINE
             /
3134  0000   OCTOUT, 0 /TYPES AC
3135  3355     DCA OCTIPT
3136  1371     TAD (-4
3137  3356    DCA OCTCNT
3140  1355  OCTLP, TAD OCTIPT
3141  7006    RTL
3142  7004    RAL
3143  3355    DCA OCTIPT
3144  1355    TAD OCTIPT
3145  7004    RAL
3146  0370    AND (7
3147  1367    TAD (260
3150  4777    JMS TYPECH
3151  7300    CLA CLL
3152  2356    ISZ OCTCNT
3153  5340    JMP OCTLP
3154  5734    JMP I OCTOUT /AC=0 L=0
            /
3155  0000  OCTIPT, 0
3156  0000  OCTCNT, 0
            /
3167  0260  PAGE
3170  0007
3171  7774
3172  2600
3173  5123
3174  7775
3175  2400
3176  7773
3177  2327
```

```
              *3200
              /
              /CDUM...DUMP COMMAND ... THIS IS A KLUDGE
              /
3200  4777    CDUM, JMS DPOI /GET NUMBER
3201  5776      JMP TOFEW /ERROR MESSAGE
3202  3050      DCA CSTRLO
3203  7004      RAL
3204  3047      DCA CSTRHG
3205  4777      JMS DPOI
3206  5302      JMP CDUM1 /ONLY 1 LOC
3207  3051      DCA CENDLO
3210  7004      RAL
3211  3052      DCA CENDHG
3212  1050      TAD CSTRLO /FIGURE NUMBER OF BLANKS
3213  0375      AND (7770
3214  3053      DCA CDUMT /TEMP
3215  1050      TAD CSTRLO
3216  0374      AND (7
3217  7041      CIA
3220  3054      DCA CDEMPT /THIS MANY
3221  1053      TAD CDUMT
3222  3050      DCA CSTRLO
3223  1051      TAD CENDLO
3224  7040      CMA
3225  1050      TAD CSTRLO
3226  7500      SMA
3227  5773      JMP NOPEM /TOO BAD
3230  3055      DCA OUTCNT
3231  4772    CDUM5, JMS CRLF
3232  1371      TAD (240
3233  4770      JMS TYPECH
3234  1047      TAD CSTRHG
3235  1367      TAD (260
3236  4770      JMS TYPECH
3237  1050      TAD CSTRLO
3240  4766      JMS OCTOUT
3241  1371      TAD (240
3242  4770      JMS TYPECH
3243  1375      TAD (-10 /OCTAL
3244  3056      DCA WRDCNT
3245  1365    CDUM4, TAD (-2
3246  4764      JMS SPACER /2 SPACES
3247  1054      TAD CDEMPT
3250  7650      SNA CLA
3251  5257      JMP CDUM2 /NO MORE BLANKS
3252  1363      TAD (-4
3253  4764      JMS SPACER /BLANK
3254  2054      ISZ CDEMPT
3255  7000      NOP
3256  5265      JMP CDUM3
              /
3257  1047    CDUM2, TAD CSTRHG
3260  7110      RAR CLL
3261  1050      TAD CSTRLO
```

PAGE 15

```
3262  4762    JMS GETLOC /GET LOC
3263  4761    JMS EXTRAC /GET WORD
3264  4766    JMS OCTOUT /PRINT
             /
3265  7301   CDUM3, CLA CLL IAC
3266  1050    TAD CSTRLO
3267  3050    DCA CSTRLO
3270  7010    RAR
3271  1047    TAD CSTRHG
3272  3047    DCA CSTRHG
3273  2055    ISZ OUTCNT
3274  7410    SKP
3275  5760    JMP ENTER /ALL DONE
3276  4757    JMS TERMTST /SEE IF USER WANTS TO STOP
3277  2056    ISZ WRDCNT
3300  5245    JMP CDUM4 /MORE ON LINE
3301  5231    JMP CDUM5
             /
3302  4772   CDUM1, JMS CRLF
3303  1356    TAD (275 /=
3304  4770    JMS TYPECH
3305  1047    TAD CSTRHG
3306  7110    RAR CLL
3307  1050    TAD CSTRLO
3310  4762    JMS GETLOC
3311  4761    JMS EXTRAC
3312  4766    JMS OCTOUT
3313  5760    JMP ENTER
             /
             /CREP...REPLACE COMMAND
             /
3314  4777   CREP, JMS DPOI    /GET ADDRESS
3315  5776    JMP TOFEW /ERROR
3316  3050    DCA CSTRLO
3317  7004    RAL
3320  3047    DCA CSTRHG
3321  3053    DCA CDUMT /COUNTS # REPS
3322  4777   CREP2, JMS DPOI    /GET VALUE
3323  5345    JMP CREPD /DONE
3324  7430    SZL /LINC SHOULD = 0
3325  5755    JMP MINV /ILLEGAL NUMBER
3326  3025    DCA HOLD / SET UP PLACE ROUTINE
3327  2053    ISZ CDUMT / COUNT THIS REP
3330  1047    TAD CSTRHG
3331  7110    RAR CLL    /SET UP HIGH ADDR
3332  1050    TAD CSTRLO /SET UP LOW ADDR
3333  4762    JMS GETLOC /GET IN CORE ADDR
3334  4764    JMS PLACE /PUT IT OUT
3354  3666    JMS REFER /REF PAGE
3355  3123
3356  0275
3357  2550
3360  2600
3361  3675
3362  3600
```

```
PAGE 16

3363  7774
3364  3517
3365  7776
3366  3134
3367  0260
3370  2327
3371  0240
3372  2335
3373  4126
3374  0007
3375  7770
3376  2724
3377  3055
3335  4753
3336  7301     CLA IAC CLL
3337  1050     TAD CSTRLO  / INCREMENT ADDRESS BY 1
3340  3050     DCA CSTRLO
3341  7004     RAL
3342  1047     TAD CSTRHG
3343  3047     DCA CSTRHG
3344  5322     JMP CREP2  / DO NEXT ONE
3345  1053   CREPD, TAD CDUMT  / See IF WE DID 4 REPLACES
3346  7650     SMA CLA
3347  5752     JMP TOFEW /ERROR
3350  5751     JMP ENTER /ALL DONE
             /
             /
3351  2600   PAGE
3352  2724
3353  4400
```

```
            *3400
            /
3400  7300  LOADP, CLA CLL /SET UP LOADING SEQUENCE
3401  1045    TAD PRGCNT
3402  7041    CIA
3403  3045    DCA PRGCNT /SET UP COUNTER
3404  1777    TAD DLNLST
3405  3217    DCA LOADT /SET UP LOADING POINTER
3406  3776  LOAD2, DCA BANK /SET BANK=0
3407  1617    TAD I LOADT /GET FILE LOCATION
3410  4775    JMS SETIPT /SET UP INPUT ROUTINES
3411  4446    JMS I LOADGO /GO DO LOADING IN DEC REL. LOADER
3412  7200    CLA
3413  2217    ISZ LOADT /LOADER GETS BACK SO ADV TO NEXT FILE
3414  2045    ISZ PRGCNT /SEE IF DONE
3415  5206    JMP LOAD2 /DO NEXT
3416  5774    JMP ENTER /ALL DONE
            /
3417  0000  LOADT, 0
            /
            /
3420  1373  RLERROR, TAD (-2 /LDR ERROR, 2 IS OK
3421  7450    SNA
3422  5232    JMP CHBANK /SIMPLY BANK FULL
3423  3237  RLER2, DCA RLERT /FATAL LOADER ERROR... YOU GOT TROUBLE DOC!
3424  1772    TAD MFATLER
3425  4771    JMS MESAGE
3426  1370    TAD (262
3427  1237    TAD RLERT
3430  4767    JMS TYPECH
3431  5774    JMP ENTER
3432  1776  CHBANK, TAD BANK /SEE WHICH BANK
3433  7640    SZA CLA
3434  5223    JMP RLER2 /ALL FULL BOTH BANK 0 AND 1
3435  7201    CLA IAC /TRY BANK 1 THIS TIME
3436  5206    JMP LOAD2 /AWAY WE GO...
            /
3437  0000  RLERT, 0
            /
            /
3440  0000  RLTYP, 0 /TYPES MAP, ETC .. BUFFERS LINES FROM DEC REL LOADER
3441  3024    DCA GETTEM
3442  6214    RDF
3443  1366    TAD (6201 /SAVE RELOCATABLE LOADER DATA FIELD
3444  3307    DCA RLTYR
3445  6211    CDF 10
3446  1024    TAD GETTEM
3447  2240    ISZ RLTYP /MIS PATCH.. INIT THIS ROUTINE
                        /BY INIT THE LINE BUFFER
3450  1365    TAD (-212 /LF>
3451  7450    SNA
3452  5307    JMP RLTYR
3453  1364    TAD (-3
3454  7450    SNA /CR?
3455  5261    JMP RLCRT /TERMINATE LINE
```

```
3456  1363     TAD (215
3457  4762     JMS DPOSIT /DON'T FORGET TO INIT..SAVING CHARACTER
3460  5307     JMP RLTYR  / RETURN
3461  1044   RLCRT, TAD UNLSW /SEE IF WE SHOULD TYPE NOW
3462  7700     SMA CLA /-1=ONLY UNLOADED ONES TO BE TYPED
3463  5270     JMP RLTYP2
3464  1761     TAD 1416 /U? ALSO TYPE JUST THOSE NOT YET LOADED
3465  1360     TAD (-325 /-U
3466  7640     SZA CLA
                                    LINE
3467  5306     JMP RLTDN /INIT BUFF AND RET
3470  1035   RLTYP2, TAD BCRLOC /IGNORE BLANK LINES
3471  1357     TAD (-1400
3472  7650     SNA CLA
3473  5306     JMP RLTDN
3474  4756     JMS ODI /INIT OUT ROUTINE
3475  4755     JMS CRLF /NEW LINE
3476  7240     CLA CMA
3477  4317     JMS SPACER /TYPE SPACE
3500  4754   RLTYPL, JMS GCNFB  /GET CHARACTER
3501  5306     JMP RLTDN    /END OF LINE RETURN POINT
3502  7410     SKP          /ASCII RET PT
3554  2400     TAD (260     /OCTAL RET PT
3555  2335
3556  2420
3557  3400
3560  7453
3561  1416
3562  2315
3563  0215
3564  7775
3565  7566
3566  6201
3567  2327
3570  0262
3571  3000
3572  5107
3573  7776
3574  2600
3575  4200
3576  3330
3577  4676
3503  1353   SE
3504  4752     JMS TYPECH  /TYPE A CHARACTER
3505  5300     JMP RLTYPL  /DO ENTIRE LINE
3506  4311   RLTDN, JMS INITIL /INIT BUFF.. FOR NEXT LINE, IF ANY
3507  7000   RLTYR, NOP /A CDF FOR RETURN TO DEC RELOC ROUTINE
3510  5640     JMP I RLTYP  /RETURN TO DEC
             /
3511  0000   INITIL, 0   / INITIALIZE INPUT BUFFER FOR MAP COMMAND
3512  7300     CLA CLL
3513  1031     TAD BFSINT
3514  3035     DCA BCRLOC
3515  3751     DCA 1416 /CLEAR U TEST .. IMPORTANT
3516  5711     JMP I INITIL
             /
```

```
3517  0000  SPACER, 0        /AC = -# SPACES TO TYPE
3520  3326    DCA SPACNT
3521  1350    TAD (240
3522  4752    JMS TYPECH
3523  2326    ISZ SPACNT
3524  5321    JMP .-3
3525  5717    JMP I SPACER
            /
3526  0000  SPACNT, 0
            /
            /
3527  0000  RDRIN, 0       / READER INPUT ROUTINE
3530  6031    KSF
3531  5330    JMP .-1
3532  6036    KRB
3533  0347    AND (177
3534  7450    SNA
3535  5330    JMP RDRIN+1  /IGNORE LEADER TRAILER
3536  1346    TAD (200
3537  5727    JMP I RDRIN
            /
3546  0200  PAGE
3547  0177
3550  0240
3551  1416
3552  2327
3553  0230
```

```
              *3600
              /
              /UTILITY FOR DETERMINING LOW CORE ADDRESS OF A LOC
              /SPECIFIED IN THE ADDRESS FORMED BY LINK AND AC
              /IF NOT IN CORE WRITES CORE OUT AND GETS LOC INTO CORE
              /STARTING AT A NEAR PAGE BOUNDARY
              /RETS WITH LOC IN AC
              /
3600  0000    GETLOC, 0
3601  3021      DCA WHLOC  /SAVE 12 BITS OF ADDRESS
3602  7004      RAL
3603  3020      DCA WHBNK  /SAVE WHICH MEMORY BANK
3604  1020      TAD WHBNK
3605  7110      RAR CLL
3606  1021      TAD WHLOC
3607  7006      RTL
3610  7006      RTL
3611  7006      RTL /GETTING PAGE #
3612  0377      AND (77
3613  3022      DCA RPAGE /0-77
3614  1022      TAD RPAGE  /SEE IF TRYING TO MODIFY RUN TIME
3615  7041      CIA
3616  1376      TAD (3 /IF POS IN RUN TIME
3617  7700      SMA CLA
3620  5263      JMP RUNTMN /HANDLE R T DIFF SINCE ITS LEFT IN CORE
3621  1023      TAD CURSTR /CURRENT STRT PAGE IN CORE
3622  7041      CIA      /SEE IF PAGE DESIRED IS IN CORE
3623  1022      TAD RPAGE /IF NEG BELOW CURENT
3624  7510      SPA
3625  5231      JMP BELOWC  /BELOW WHATS IN CORE
3626  1375      TAD (-20 /SEE IF ABOVE
3627  7710      SPA CLA
3630  5245      JMP INCORE
3631  4315    BELOWC, JMS WPRESC  /WRITE PRESENT CORE IMAGE SEGMENT OUT
3632  1022      TAD RPAGE
3633  1374      TAD (-4 /ALLOW OVERLAP TO OCCUR BETWEEN SEGMENTS
3634  3023    GETNEW, DCA CURSTR /NEW START OF IN CORE SEGMENT.
3635  1023      TAD CURSTR
3636  1373      TAD (200
3637  3241      DCA GTBLK
3640  4126      JMS I RTAPE /GET NEW SEGMENT INTO CORE
3641  0000    GTBLK, 0
3642  0020      20
3643  0000      0
3644  3000      3000
              /
3645  7300    INCORE, CLA CLL  /PROPER SEGMENT IS IN CORE
3646  1023      TAD CURSTR /CALC ADDRES IN BUFF
3647  7041      CIA
3650  1022      TAD RPAGE
3651  7106      CLL RTL
3652  7006      RTL
3653  7006      RTL
3654  7004      RAL
3655  1372      TAD (3000 /STRT OF OPT BUFF
PAGE 21

3656  3024    DCA GETTEM
3657  1021    TAD WHLOC
3660  0371    AND (177
3661  1024    TAD GETTEM
3662  5600   GETRET, JMP I GETLOC
             /
3663  1021   RUNTMN, TAD WHLOC /HANDLES ADDRESSES INTO RUN TIME
3664  7100    CLL
3665  5262    JMP GETRET
             /
             /
3666  0000   PLACE, 0 /PUT C(HOLD) IN BUFF ADD IN AC
3667  3024    DCA GETTEM
3670  1025    TAD HOLD
3671  6201    CDF
3672  3424    DCA I GETTEM
3673  6211    CDF 10
3674  5666    JMP I PLACE /AC=0
             /
3675  0000   EXTRAC, 0 /GET CONT OF LOC IN AC FROM BUFFER
3676  3024    DCA GETTEM
3677  6201    CDF
3700  1424    TAD I GETTEM
3701  6211    CDF 10
3702  5675    JMP I EXTRAC
             /
             /RLOAD...SUBROUTINE SABR USES TO DEPOSIT
             /
3703  0000   RLOAD, 0    /SABR LOADER COMES HERE TO DEPOSIT INTO CORE IMAGE
3704  2303    ISZ RLOAD /MISS PATCH
3705  3025    DCA HOLD /SAVE CONT
3706  1770    TAD BANK
3707  7110    CLL RAR
3710  1737    TAD CUR /AC = ADDRESS DESIRED
3711  4200    JMS GETLOC /AC=CORE ADDR
3712  4266    JMS PLACE /STOW IT
3713  4766    JMS REFER /REF THE PROPER PAGE
3714  5703    JMP I RLOAD
             /
3715  0000   WPRESC, 0    /WRITE CONTENTS OF PRESENT CORE IMAGE BUFFER ON TAPE
3716  7300    CLA CLL
3717  1373    TAD (200
3720  1023    TAD CURSTR
3721  3323    DCA SVEBLK
3722  4427    JMS I WTAPE
3723  0000   SVEBLK, 0
3724  0020    20          /OCTAL 20 BLOCK (PAGE) BUFFER
3725  0000    0
3726  3000   DBFST, 3000  /STARTING AT LOC 3000    (3000-6777)
3727  5715    JMP I WPRESC
             /
             /
3730  1365   GTOUT2, TAD (7773 /REST OF BUILD... HAD TO SPLIT UP BECAUSE
3731  3010    DCA 10                                ROOM GOT TIGHT
3732  1364    TAD (-7402
                        /THIS UPDATES THE -B PARAMETER FOR CPS
```

```
3733  3410    DCA I 10
3734  7240    CLA CMA
3735  3410    DCA I 10
3736  7201    CLA IAC
3737  3410    DCA I 10
3740  1363    TAD (7402
3741  3410    DCA I 10
3742  1362    TAD (7767
3743  3010    DCA 10
3744  1361    TAD (37
3745  3410    DCA I 10
3746  1100    TAD 100
3747  3410    DCA I 10
3750  1360    TAD (400
3751  3410    DCA I 10
3752  5757    JMP CEXI  /EXIT FROM LOADER BUT WITH CPS UPDATE SWITCHES SET
                /
3757  4311  PAGE
3760  0400
3761  0037
3762  7767
3763  7402
3764  0376
3765  7773
3766  4400
3767  6540
3770  6336
3771  0177
3772  3000
3773  0200
3774  7774
3775  7760
3776  0003
3777  0077
```

```
                *4000
                /
4000  0000  GETIPT, 0 /FEEDS BINARY TO LOADER .. LOADER COMES HERE FOR ITS DATA
4001  7300    CLA CLL
4002  1256    TAD THSW /0=LFT, 1=MID, -1=RT .. UNPACKING BINARY.. SEE WHICH PART
4003  7450    SNA
4004  5245    JMP LFTS   /LEFT
4005  7700    SMA CLA
4006  5222    JMP MIDS   /MID
4007  6201    CDF /-1 EXTRACT RIGHT
4010  1257    TAD I GETLO
4011  6211    CDF 10
4012  0377    AND (377
4013  3260    DCA IPTT
4014  4261    JMS GNXLOC /MAKE SURE NEXT IN CORE
4015  7200    CLA
4016  3256  GIBRET, DCA THSW
4017  1260    TAD IPTT
4020  2200    ISZ GETIPT
4021  5600    JMP I GETIPT
            /
4022  6201  MIDS, CDF
4023  1657    TAD I GETLO /LOW 4 BITS
4024  6211    CDF 10
4025  0376    AND (17
4026  7106    CLL RTL
4027  7006    RTL
4030  3260    DCA IPTT
4031  4261    JMS GNXLOC
4032  6201    CDF
4033  1657    TAD I GETLO
4034  6211    CDF 10
4035  0375    AND (7400
4036  7106    CLL RTL
4037  7006    RTL
4040  7004    RAL
4041  1260    TAD IPTT
4042  3260    DCA IPTT
4043  7240    CLA CMA
4044  5216    JMP GIBRET
            /
4045  6201  LFTS, CDF
4046  1657    TAD I GETLO
4047  6211    CDF 10
4050  0374    AND (7760
4051  7112    RTR CLL
4052  7012    RTR
4053  3260    DCA IPTT
4054  7201    CLA IAC
4055  5216    JMP GIBRET
            /
4056  0000  THSW, 0
4057  0000  GETLO, 0
4060  0000  IPTT, 0
            /
```

PAGE 24

```
4061  0000  GNXLOC, 0         / ROUTINE TO GET NEXT LOCATION IN INPUT BUFFER
4062  7300    CLA CLL
4063  1373    TAD (-3000  /SEE IF NEXT LOC IS IN BUFFER
4064  2257    ISZ GETLO
4065  1257    TAD GETLO
4066  7620    SNL CLA /+MEANS REFIL
4067  5661    JMP I GNXLOC
4070  4423    JMS I RTAPE  /READ MORE IN
4071  0000  IGTBLK, 0 /MUST BE SET
4072  0010    10               / 10 PAGE BUFFER
4073  0001  GUNT, 1 /MUST BE SET
4074  1000  GINTL, 1000      / LOCS 1000 - 2777 IN FIELD 1 MISSES RUN TIME
4075  1271    TAD IGTBLK
4076  1272    TAD IGTBLK+1
4077  3271    DCA IGTBLK /ADV TAPE BLOCK
4100  1274    TAD GINTL
4101  3257    DCA GETLO
4102  5661    JMP I GNXLOC
            /
            /CIND...INDEX COMMAND
            /
4103  7300  CIND, CLA CLL
4104  1372    TAD (200
4105  3325    DCA CINDT /SET UP INDEX POINTER
4106  1325  CINDL, TAD CINDT
4107  4771    JMS MVTW  /GET NAME
4110  1037    TAD W1
4111  7650    SNA CLA  /TEST FOR INDEX END
4112  5324    JMP CINDD
4113  4770    JMS CRLF  /NEW LINE
4114  7240    CLA CMA
4115  4767    JMS SPACER /INDENT 1 SPACE
4116  1036    TAD WLOC
4117  4766    JMS MESAGE /TYPE NAME
4120  1325    TAD CINDT
4121  1365    TAD (5     /ADVANCE INDEX POINTER
4122  3325    DCA CINDT
4123  5306    JMP CINDL
            /
4124  5764  CINDD, JMP ENTER
            /
4125  0000 .CINDT, 0
            /
4126  7300  NOPEN, CLA CLL  / TYPES   NO!
4127  1763    TAD NOM
4130  4766    JMS MESAGE
4131  5764    JMP ENTER
            /
            /
4163  5150  PAGE
4164  2300
4165  0005
4166  3000
4167  3517
4170  2335
```

```
PAGE 25

4171  4275
4172  0200
4173  5000
4174  7760
4175  7400
4176  0017
4177  0377
```

```
            *4200
            /
4200  0000  SETIPT, 0 /SET UP BINARY READING
4201  3252    DCA SETT /TEMP
4202  1252    TAD SETT
4203  0377    AND (1777
4204  1776    TAD IGTBLK+1 /SEE IF DESIRED INPUT IS IN CORE
4205  7041    CIA          /USED WHEN BNK∅ IS FULL SAVING A TAPE READ
4206  1775    TAD IGTBLK
4207  7650    SNA CLA
4210  5225    JMP INHERE /BINARY IS IN CORE
4211  1252    TAD SETT  /SET UP ROUTINES TO FORCE FIRST DATA
4212  0377    AND (1777  /REQUEST TO READ TAPE
4213  3775    DCA IGTBLK
4214  1252    TAD SETT
4215  7106    RTL CLL
4216  0374    AND (1
4217  3773    DCA GUNI
4220  1372    TAD (3001 /FORCE READ
4221  3771    DCA GETLO
4222  4770    JMS GNXLOC /FORCE READ
4223  3767    DCA THSW
4224  5600    JMP I SETIPT
4225  1366  INHERE, TAD (1000 /DONT FORCE READ
4226  3771    DCA GETLO    /COULD DO JMP INHERE-2
4227  3767    DCA THSW
4230  5600    JMP I SETIPT
            /
            /CMAP...MAP COMMAND
            /
4231  4765  CMAP, JMS GNFLD /DO WE HAVE A PARAMETER?
4232  5237    JMP CMAP2 /NO
4233  1037    TAD W1 /SOMETHING
4234  0364    AND (7700
4235  1364    TAD (-0100 /A ?
4236  7640    SZA CLA
4237  7240  CMAP2, CLA CMA /-1 ONLY UNLOADED
4240  3044    DCA UNLSW
4241  4763    JMS INITIB
4242  1362    TAD (2000
4243  4746  CMAP3, JMS I LOADGO /TO SAPR LDR
4244  2044    ISZ UNLSW /IF -1 DO PAGES
4245  5761    JMP ENTER
            /
            /CSPA...SPACE COMMAND
            /
4246  4763  CSPA, JMS INITIB /INITIALIZE LINE BUFFER
4247  3044    DCA UNLSW
4250  1360    TAD (4000
4251  5243    JMP CMAP3   /USE PART OF CMAP
            /
4252  0000  SETT, 0
            /
            /CLIB...LIBRARY COMMAND
            /
```

```
4253  7300  CLIB, CLA CLL
4254  1357    TAD (6
4255  3045    DCA PRGCNT /SET PRGM CNT TO 6
4256  1756    TAD CLIBL  /GET SET LST TO LIBRARY LIST
4257  5755    JMP INDSRH /ALL DONE HERE
            /
4260  1754  NOTHER, TAD MPART1 /CRLF" ASKS FOR REPLACEMENT NAME
4261  4753    JMS MESAGE
4262  1036    TAD WLOC
4263  4753    JMS MESAGE
4353  3000    TAD MPART2 /"DOES NOT EXIST ...
4354  5053
4355  2732
4356  4730
4357  0006
4360  4000
4361  2600
4362  2000
4363  3511
4364  7700
4365  2446
4366  1000
4367  4056
4370  4061
4371  4057
4372  3001
4373  4073
4374  0001
4375  4071
4376  4072
4377  1777
4264  1752
4265  4751    JMS MESAGE
4266  1350    TAD (277 /?
4267  3034    DCA PREFIX  /CHANGE PREFIX TO ?
4270  4747    JMS PLIB  /GET NEW LINE
4271  4746    JMS OBI   /SET UP EXTRACTION ROUTINE
4272  4745    JMS GWFLB /GET A NEW NAME
4273  5744    JMP ENTER /CR ONLY
4274  5743    JMP INDSH3 /NEW NAME SO TEST IT TOO
            /
4275  0000  MVTW, 0   /MOVES CONTENTS OF ADDRESS IN AC INTO W1-W4
4276  1342    TAD (-1 /WILL AUTO-INDEX
4277  3010    DCA 10
4300  1410    TAD I 10
4301  3037    DCA W1
4302  1410    TAD I 10
4303  3040    DCA W2
4304  1410    TAD I 10
4305  3041    DCA W3
4306  1410    TAD I 10
4307  3042    DCA W4
4310  5675    JMP I MVTW
            /
            /
```

```
               /CEXI...EXIT COMMAND
               /
4311  7300     CEXI, CLA CLL / JUMPS TO CPS BOOT IN FIELD Ø
4312  6203       CDF CIF
4313  5714       JMP I .+1
4314  7600       7600
               /
4342  7777     PAGE
4343  2742
4344  2600
4345  2446
4346  2420
4347  2200
4350  0277
4351  3000
4352  5057
```

```
                *4400
                /
                /REFER...REFERENCE ROUTINE
                /
4400   0000     REFER, 0    / SETS THE ENTRY OF PAGE REFERENCE TABLE TO 1
4401   7200       CLA
4402   1022       TAD RPAGE / REFERENCE THIS PAGE
4403   1223       TAD PAGEB /BASE OF PAGE TABLE
4404   3024       DCA GETTEM
4405   7201       CLA IAC
4406   3424       DCA I GETTEM
4407   5600       JMP I REFER
                /
                /CUNR...UNREFERENCE COMMAND
                /
4410   7300     CUNR, CLA CLL
4411   4777       JMS DPOI   /GET THE LOCATION
4412   5776       JMP TOFEW  / USER DIDNT GIVE ONE
4413   7006       RTL            /EXTRACT THE PAGE #
4414   7006       RTL
4415   7006       RTL
4416   0375       AND (77     /MASK FOR 8-K
4417   1223       TAD PAGEB / ADD BASE OF TABLE
4420   3024       DCA GETTEM
4421   3424       DCA I GETTEM /CLEAR TABLE ENTRY
4422   5774       JMP ENTER
                /
4423   4424     PAGEB, .+1 /START OF TABLE
                /
4574   2600     PAGE         / INEFFICIENT USE OF MEMORY, USING 1 LOC PER PAGE
4575   0077                  / REQUIRE 64 DECIMAL LOCATIONS
4576   2724
4577   3055
```

```
                *4600
                /
                /COMTAB...COMMAND TABLE
                /FOR SLRV USE
                /
4600   4601     COMTAB, .+1 /START
4601   0705     0705 /GET
4602   2440     2440
4603   4040     4040
4604   4040     4040
4605   2671     CGET
4606   0225     0225 /BUILD
4607   1140     1140
4610   4040     4040
4611   4040     4040
4612   5270     CBUI
4613   0425     0425 /DUMP
4614   1540     1540
4615   4040     4040
4616   4040     4040
4617   3200     CDUM
4620   2205     2205 /REPLACE
4621   2040     2040
4622   4040     4040
4623   4040     4040
4624   3314     CREP
4625   0530     0530 /EXIT
4626   1140     1140
4627   4040     4040
4630   4040     4040
4631   4311     CEXI
4632   1501     1501 /MAP
4633   2040     2040
4634   4040     4040
4635   4040     4040
4636   4231     CMAP
4637   2320     2320 /SPACE
4640   0140     0140
4641   4040     4040
4642   4040     4040
4643   4246     CSPA
4644   1411     1411 /LIBRARY
4645   0240     0240
4646   4040     4040
4647   4040     4040
4650   4253     CLIB
4651   2205     2205 /RESTART
4652   2340     2340
4653   4040     4040
4654   4040     4040
4655   5200     CRES
4656   0317     0317 /COMMENT
4657   1540     1540
4660   4040     4040
4661   4040     4040
```

```
4662  2344  CCOM
4663  2516  2516 /UNREFFERENCE
4664  2240  2240
4665  4040  4040
4666  4040  4040
4667  4410  CUNR
4670  1116  1116 /INDEX
4671  0440  0440
4672  4040  4040
4673  4040  4040
4674  4103  CIND
            /
4675  0000  0 /TERMINATES
            /
            /
4676  4677  CGETL, .+1
            /
            BLNLST=CGETL /DOUBLE DUTY
            /
4677  0000  0       /USED FOR FORMING LIST OF NAMES FOR THE GET COMMAND
4700  0000  0
4701  0000  0       /ALSO USED TO STORE POINTERS TO NAMED FILES AS NAMES
4702  0000  0       /ARE LOCATED IN THE INDEX
4703  0000  0
4704  0000  0
4705  0000  0
4706  0000  0
4707  0000  0
4710  0000  0
4711  0000  0
4712  0000  0
4713  0000  0
4714  0000  0
4715  0000  0
4716  0000  0
4717  0000  0
4720  0000  0
4721  0000  0
4722  0000  0
4723  0000  0
4724  0000  0
4725  0000  0
4726  0000  0
4727  0000  0
4730  4731  CLIBL, .+1   /LIST OF NAMES FOR THE LIBRARY COMMAND
4731  5611  5611 /.IOH  /NEXT TIME USE * INPLACE OF . AS A PREFIX!
4732  1710  1710
4733  4040  4040
4734  4040  4040
4735  5606  5606 /.FLOAT
4736  1417  1417
4737  0124  0124
4740  4040  4040
4741  5611  5611 /.INTEGER
4742  1624  1624
```

```
PAGE 32

4743  0507  0507
4744  0522  0522
4745  5625  5625 /.UTILITY
4746  2411  2411
4747  1411  1411
4750  2431  2431
4751  5605  5605 /.ERROR
4752  2222  2222
4753  1722  1722
4754  4040  4040
4755  5623  5623 /.SUBSC
4756  2502  2502
4757  2303  2303
4760  4040  4040
            /
4761  0000  0 /TERMINATES
            /
            PAGE
```

```
                *5000
                /MESSAGES
                /
5000    5001    MITMN, .+1
5001    4543    TEXT :%#
5002    5624    .T
5003    1717    OO
5004    4015     M
5005    0116    AN
5006    3140    Y
5007    0611    FI
5010    1405    LE
5011    2300    S:
5012    5013    MIDF, .+1
5013    4543    TEXT :%#
5014    5614    .L
5015    1116    IN
5016    0540    E
5017    0225    BU
5020    0606    FF
5021    0522    ER
5022    4006     F
5023    2514    UL
5024    1400    L:
5025    5026    MILCO, .+1
5026    4543    TEXT :%#
5027    5611    .I
5030    1414    LL
5031    0507    EG
5032    0114    AL
5033    4003     C
5034    1715    OM
5035    1501    MA
5036    1604    ND
5037    0000    :
5040    5041    MIOFE, .+1
5041    4543    TEXT :%#
5042    5624    .T
5043    1717    OO
5044    4006     F
5045    0527    EW
5046    4017     O
5047    2005    PE
5050    2201    RA
5051    1604    ND
5052    2300    S:
5053    5054    MPART1, .+1
5054    4543    TEXT :%#
5055    5640    .
5056    0000    :
5057    5060    MPART2, .+1
5060    4240    TEXT :
5061    0417    DO
5062    0523    ES
5063    4016     N
```

```
PAGE 34

5064  1724  OT
5065  4005   E
5066  3011  XI
5067  2324  ST
5070  4543  %#
5071  5605  .E
5072  1624  NT
5073  0522  ER
5074  4022   R
5075  0520  EP
5076  1401  LA
5077  0305  CE
5100  1505  ME
5101  1624  NT
5102  4017   O
5103  2240  R
5104  7403  <C
5105  2276  R>
5106  0000  :
5107  5110  MFATLER, .+1
5110  4543  TEXT :%#
5111  5606  .F
5112  0124  AT
5113  0114  AL
5114  4014   L
5115  1701  OA
5116  0440  D
5117  0522  ER
5120  2217  RO
5121  2240  R
5122  0000  :
5123  5124  MINUM, .+1
5124  4543  TEXT :%#
5125  5611  .I
5126  1414  LL
5127  0507  EG
5130  0114  AL
5131  4017   O
5132  0324  CT
5133  0114  AL
5134  4016   N
5135  2515  UM
5136  0205  BE
5137  2200  R:
5140  5141  NORTM, .+1
5141  5616  TEXT :.N
5142  1740  O
5143  2225  RU
5144  1640  N
5145  2411  TI
5146  1505  ME
5147  7700  ?:
5150  5151  NOM, .+1
5151  4543  TEXT :%#
5152  5616  .
```

```
PAGE 35

5153  1741  0!
5154  0000  :
            /
            PAGE
            *5200
            /
            /CRES...RESTART COMMAND
            /
5200  7300  CRES, CLA CLL        / this to reinitialize everything
5201  5777    JMP 6200 /START W SABR loader
5202  7240  RLSTRTR, CLA CMA   / Dec loader comes here
5203  3776    DCA COML /BECAUSE OF PATCH
5204  1375    TAD (777 /0 LOW CORE except for run time in 0-777
5205  3010    DCA 10
5206  1374    TAD (-6600
5207  3267    DCA CRESCN
5210  6201    CDF
5211  3410    DCA I 10
5212  2267    ISZ CRESCN
5213  5211    JMP .-2
5214  6211    CDF 10       / now core locs 1000-7577 are now 0
5215  4427    JMS I WTAPE /NOW FIX TAPES
5216  0200     200       / 1st half of field 1 core image
5217  0020     20
5220  0000     0
5221  1000     1000
5222  4427    JMS I WTAPE
5223  0220     220       / rest of field 1 core image
5224  0020     20
5225  0000     0
5226  1000     1000
5227  4427    JMS I WTAPE
5230  0241     241 /MISS RUN TIME for field 0 which was written earlier
5231  0020     20
5232  0000     0
5233  1000     1000
5234  4427    JMS I WTAPE
5235  0261    261        / finish field 1 core image
5236  0020     20
5237  0000     0
5240  1000     1000
            /NOW FIX REFERENCE LIST
5241  1061    TAD M100 /FIRST ZERO IT unkeeping all pages
5242  3267    DCA CRESCN
5243  1773    TAD PAGEL
5244  3022    DCA RPAGE
5245  3422    DCA I RPAGE
5246  2022    ISZ RPAGE
5247  2267    ISZ CRESCN
5250  5245    JMP .-3
            /REF P 0-3   THE FIELD 0 RUNTIME
5251  3022    DCA RPAGE
5252  1062    TAD M4
5253  3267    DCA CRESCN
5254  4772    JMS REFER
5255  2022    ISZ RPAGE
5256  2267    ISZ CRESCN
5257  5254    JMP .-3
5260  1371    TAD (40 /REF F 1 RT
5261  3022     DCA RPAGE
5262  4772     JMS REFER
              /
5263  3770     DCA IGTBLK /FORCE INPUT READ WHEN FIRST BINARY INPUT IS REQUESTED
5264  1063     TAD C4       / THIS CODE IS NECESSARY BECAUSE OF THE WAY BINARY RESTARTS
5265  3023     DCA CURSTR   / DON'T NECESSARILY RELOAD NEW INPUT INTO CORE
5266  5767     JMP ENTER /GOOOOO TO IT
              /
5267  0000    CRESCN, 0
              /
              /
              /CBUI...BUILD COMMAND
              /
              CBUFLD=100
              CBUSA=101
              DISLOC=102
              RBSE=103
              PLOCAT=104
              PSTART=105
              PAGCNT=106
              COREND=107
              CFLD=110
              DISCNT=111
              DISCN2=112
              LCDFA=140
              LCDFB=141
              /
5270  4766    CBUI, JMS DPOI /SEE IF THERE'S A SA
5271  5277     JMP SABRSA /NO
5272  3101     DCA CBUSA /YES SAVE 12 BIT ADDRESS
5273  7006     RTL
5274  7006     RTL
5275  3100     DCA CBUFLD /SAVE FLD LOCATION
5276  5306     JMP CBUI2
5277  6201    SABRSA, CDF /GET FROM RT LOC OF FIRST LOADED FILE
5300  1765     TAD 300   /THIS IS A TAD I NEEDED IN EFFECT
5301  3101     DCA CBUSA / 12 BIT ADDRESS
5302  1764     TAD 200   /ACTUALLY AN INDIRECT TAD
5303  6211     CDF 10
5304  0363     AND (10   /MASK OFF FIELD BITS
5305  3100     DCA CBUFLD
5306  4762    CBUI2, JMS WPRESC / WRITE PRESENT CONTENTS OF CORE IMAGE BUFFER
5307  4427     JMS I WTAPE /W RT FOR FIELD 1
5310  0200     200
5311  0004     4
5312  0000     0
5313  0000     0
5314  4426    JMS I RTAPE /RD FLD1 INTO CORE
5315  0240     240
5316  0037     37
5317  0000     0
5320  0000     0
5321  1363     TAD (10
5322  3110     DCA CFLD /SET FLD FOR PACKING
5323  1361     TAD (1400 /CORE DIS BLK IS FORMED STARTING HERE IN FIELD 1
5324  3102     DCA DISLOC / POINTS INTO CORE IMAGE DESCRIPTOR BEING FORMED
5325  1353     TAD REFMD /MID OF REF TAB
5326  4760     JMS DISCBL /PACK UP CONTENTS OF MEMORY FIELD 1, I.E. FIELD 1 CORE IMAGE
5327  7450     SNA /MAY BE EMPTY.. AC = # BLKS IN CORE IMAGE SEGMENT
5330  5337     JMP WNUM+3
5331  3334     DCA WNUM /THIS MANY BLKS IN FIELD 1 SEGMENT
5332  4427     JMS I WTAPE /W FLD 1 INTO UNIT 1 SCRATCH AREA
5333  0040      40        / WB STARTS IN BLK 37 WITH CORE IMAGE DESC. BLK
5334  0000    WNUM, 0    / ACTUAL CODE STARTS IN BLOCK 40
5335  0001      1
5336  0000      0
5337  1333     TAD WNUM-1 /SET UP NXT WRITE FOR MEMORY FIELD 0
5340  1334     TAD WNUM
5341  3757     DCA WNUM2-1
5342  3110     DCA CFLD    / SET CURRENT FIELD TO 0
5343  4426     JMS I RTAPE /GET FLD 0 INTO LOWER 4K
5344  0200     200
5345  0037     37
5346  0000     0
5347  0000     0
5350  1354     TAD REFST /STRT OF REF TAB
5351  4760     JMS DISCBL / PACK FIELD 0 CORE IMAGE
5352  5756     JMP GETOUT
              /
5353  4434    REFMD, PAGEB+41
5354  4424    REFST, PAGEB+1
              /
5356  5514    PAGE
5357  5520
5360  5400
5361  1400
5362  3715
5363  0010
5364  0200
5365  0300
5366  3055
5367  2600
5370  4071
5371  0040
5372  4400
5373  4423
5374  1200
5375  0777
5376  6340
5377  6200
```

```
                     / THIS ROUTINE IS UNNECESSARILY COMPLICATED SINCE THE SABR LOADER
                     / DOES NOT LEAVE ANY GAPS IN CORE.. ALL THAT IS NEEDED IS TO
PAGE 39              / FIND THE FIRST Ø ENTRY IN THE PAGE REF TABLE WHICH AUTOMATICALLY WILL
                     / DETERMINE THE CORE IMAGE SEGMENT SIZE.
                      *5400
                      /
5400  0000  DISCBL, 0   / SUB PACKS FIELD Ø CONTENTS.. STARTS AT PAGE IN AC
5401  3103    DCA RBSE /SETS UP BASE TABLE i.e. POINTS INTO PAGE REF TABLE
5402  1103    TAD RBSE /UN REF P 37 ALWAYS!
5403  1037    TAD 37
5404  3111    DCA DISCNT
5405  3511    DCA I DISCNT / Ø'S PROPER LOC IN PAGE REFER TABLE
5406  1377    TAD (-37
5407  3111    DCA DISCNT  / CHECK A MAX OF 37 PAGES
5410  3107    DCA COREND / SET TOP OF PACKED CORE TO Ø
5411  3104    DCA PLOCAT /PREV PAGE
5412  1503  LOOPP, TAD I RBSE /GET REF BIT
5413  7650    SNA CLA /0 IF REF
5414  5273    JMP AZERO  / ~~[illegible]~~ A SEGMENT GAP.. BETWEEN SEGMENTS
5415  1104    TAD PLOCAT
5416  3105    DCA PSTART /~~NEW~~ START OF A SEGMENT
5417  2111  DIS2, ISZ DISCNT
5420  7000    NOP
5421  2104    ISZ PLOCAT
5422  2103    ISZ RBSE
5423  1503    TAD I RBSE /GET NEXT PAGE REF WORD
5424  7640    SZA CLA /END OF CURR SEC?
5425  5217    JMP DIS2 /MORE YET SO KEEP LOOKING FOR SEGMENT END
5426  1105    TAD PSTART /HOW MANY? IN THIS SEGMENT
5427  7041    CIA
5430  1104    TAD PLOCAT
5431  3106    DCA PAGCNT
5432  1106    TAD PAGCNT /ENTER INTO ~~DSC~~ CORE IMAGE DESCRIPTOR BLOCK SIZE
5433  3502    DCA I DISLOC
5434  2102    ISZ DISLOC  /ADV POINTER INTO CORE IMAGE DESCRIPTOR
5435  1110    TAD CFLD /SET FLD
5436  3502    DCA I DISLOC / FIELD OF CURRENT SEGMENT
5437  2102    ISZ DISLOC  /ADV POINTER
5440  1105    TAD PSTART /FIND ADDR OF SECTION IN CORE
5441  7106    CLL RTL
5442  7006    RTL
5443  7006    RTL
5444  7004    RAL
5445  3502    DCA I DISLOC / STARTING ADDRESS OF SEGMENT
5446  7240    CLA CMA /MOVE CORE TO MATCH DESCRIPTOR
5447  1502    TAD I DISLOC
5450  2102    ISZ DISLOC / POINTS TO NEXT SEGMENT ENTRY IN DESCRIPTOR BLOCK
5451  3010    DCA 10 /FROM POINTER USING AUTO-INDEX
5452  7240    CLA CMA
5453  1107    TAD COREND / TOP OF PACKED SEGMENTS IN FIELD Ø
5454  3011    DCA 11 /TO  / MOVE TO HERE
5455  1106    TAD PAGCNT / NOW CALCULATE HOW MUCH TO MOVE
5456  7106    CLL RTL
5457  7006    RTL
5460  7006    RTL
5461  7004    RAL
5462  7041    CIA
5463  3112    DCA DISCN2 /~~HOW MANY~~ MOVE THIS MANY WORDS
5464  6201    CDF
```

```
PAGE 40

5465  1410      TAD I 10   / MAY JUST MOVE CENTER OF IT SELF
5466  3411      DCA I 11   / BUT THIS IS NO PROBLEM
5467  2112      ISZ DISCN2
5470  5265      JMP .-3
5471  6211      CDF 10
5472  7201      CLA IAC
5473  1011      TAD 11
5474  3107      DCA COREND /UPDATE TOP OF AVAILABLE CORE IN FIELD 1
5475  5302      JMP DONTST /SEE IF DONE
               /
5476  2104    AZERO, ISZ PLOCAT
5477  2103      ISZ NBSE
5500  2111      ISZ DISCNT
5501  7000      NOP          / LAST ISZ CAN SKIP
5502  1111    DONTST, TAD DISCNT /SEE IF ALL 37 PAGES HAVE BEEN WRITTEN
5503  7640      SZA CLA
5504  5212      JMP LOOPP  /NOT YET
5505  1107      TAD COREND / CALCULATE NUMBER OF PAGES TO BE SENT INTO -B
5506  7012      RTR
5507  7012      RTR
5510  7012      RTR
5511  7010      RAR
5512  0373      AND (37 /MASK
5513  5600      JMP I DISCBL /AC=# BLKS TO GO INTO -B
               /
5514  7450    GETOUT, SNA / REST OF BUILD COMMAND
5515  5355      JMP RTDEL / TRIED TO HAVE 0 BLOCKS IN FIELD 0 CORE IMAGE.. NOT TOO KEEN
5516  2321      DCA WNUM2
5517  4427      JMS I WTAPE
5520  0000      0 /TO BE FILLED PTS TO START OF FIELD 0 CORE IMAGE
5521  0000    WNUM2, 0 /ALSO   SIZE OF FIELD 0 CORE IMAGE
5522  0001      1
5523  0000      0
5524  1375      TAD (7000 /NOP CDF'S IN TAPE ROUTINE SO THAT WE CAN
5525  3540      DCA I LCDFA  / WRITE THE CORE IMAGE DESCRIPTOR BLOCK AT
5526  1375      TAD (7000    / LOCATION 1400 IN FIELD 1
5527  3541      DCA I LCDFB
5530  1100      TAD CBUFLD  / SET UP FIELD OF STARTING ADDRESS
5531  3502      DCA I DISLOC
5532  2102      ISZ DISLOC  /ADV PTR INTO CORE IMAGE DESCRIPTOR
5533  1101      TAD CBUSA
5534  3502      DCA I DISLOC / SET 12 BIT STARTING ADDRESS
5535  1102      TAD DISLOC  / HAVE TO INDICATE IN CORE IMAGE DESCRIPTOR
5536  1374      TAD (-4     / WHICH ENTRY IS LAST ENTRY
5537  3102      DCA DISLOC
5540  1502      TAD I DISLOC
5541  1373      TAD (4000   / THIS IS THE PROPER SWITCH
5542  3502      DCA I DISLOC
5543  4427      JMS I WTAPE / NOW WRITE DESCRIPTOR BLOCK
5544  0037      37
5545  0001      1
5546  0001      1
5547  1400      1400
5550  1772      TAD WNUM    / NOW CALCULATE THE SIZE OF -B
5551  1321      TAD WNUM2
```

```
5552  7001     IAC   /PLUS 1 FOR DESCRIPTOR BLOCK
5553  3100     DCA 100 /SAVE IN TEMP
5554  5771     JMP GTOUT2 /RAN OUT OF ROOM HERE
             /
5555  1770   RTDEL, TAD NORTN /NO RUN TIME ERROR
5556  4767     JMS MESAGE
5557  5766     JMP CEXI /THINGS ARE TOO FAR GONE TO STAY HERE
             /              /SO GO BACK TO SYSTEM!
5566  4311   PAGE
5567  3000
5570  5140
5571  0730
5572  5334
5573  4000
5574  7774
5575  7000
5576  0037
5577  7741
```

```
                *5600
                /
                /SETS UP THE LOADER IN FIELD 1
                /
                /SYSTEM STARTS HERE WITH EVERY THING IN LOWER 4K
                /
5600    4236      JMS PACKIN /PACK UP INDEX  IN UPPER 4K LEFT BY CPS
5601    0200      200  /NEW LOCATION
5602    4000      4000 /OLD LOCATION
5603    7300      CLA CLL   / FIRST MOVE PAGE Ø OF THIS LOADER INTO
5604    1377      TAD (7777 /THE UPPER 4K
5605    3010      DCA 10
5606    1377      TAD (7777
5607    3011      DCA 11
5610    1376      TAD (-200 /PAGE 0
5611    3334      DCA MVCNTR
5612    1410      TAD I 10
5613    6211      CDF 10
5614    3411      DCA I 11
5615    6201      CDF
5616    2334      ISZ MVCNTR
5617    5212      JMP .-5
5620    1375      TAD (-5600 /THE REST OF THIS LOADER CAN NOW BE MOVED
5621    3334      DCA MVCNTR
5622    1374      TAD (1777  / START AT 2000 TO MISS NEW PACKED INDEX
5623    3010      DCA 10
5624    1374      TAD (1777
5625    3011      DCA 11
5626    1410      TAD I 10
5627    6211      CDF 10
5630    3411      DCA I 11
5631    6201      CDF
5632    2334      ISZ MVCNTR
5633    5226      JMP .-5
5634    1373      TAD (217 /MV BANKØ RUN TIME TO PROPER PLACE
5635    3010      DCA 10
5636    1372      TAD (17  /MISSING THE AUTO-INDEX REGISTERS
5637    3011      DCA 11
5640    1371      TAD (-100
5641    3334      DCA MVCNTR
5642    1410      TAD I 10
5643    3411      DCA I 11
5644    2334      ISZ MVCNTR
5645    5242      JMP .-3
5646    6213      CDF CIF 10
5647    5250      JMP .+1 /GO TO FIELD 1 WHERE THE LOADER ALSO RESIDES NOW
5650    4427      JMS I WTAPE /WRITE FIELD 1 RUNTIME ONTO TAPE
5651    0240      240     / LOC OF PAGE Ø OF FIELD 1
5652    0001      1
5653    0000      0
5654    1000      1000
5655    5770      JMP CRES /GET GOING BY FAKING A SYSTEM RESTART
               /
               /
               /INDEX PACKER FOR LOADER
```

```
                /CALL:  JMS PACKIN
                /            WHERTO  FLD 1
                /          WHERFR  FLD 1
                /         RETS HERE WITH NEXT WHERTO IN AC
                /         TERMINATES IND WITH A 0
                /         ONLY KEEPS REL BIN
                /
5656  0000      PACKIN, 0          / RETAINS ONLY 4 WORD NAME AND
5657  7300        CLA CLL          / TYPE BLOCK LOCATION
5660  1656        TAD I PACKIN     / SIZE AND TYPE LOST
5661  3332        DCA WHERTO
5662  2256        ISZ PACKIN       / ONLY SABR FILES ARE RETAINED
5663  1656        TAD I PACKIN
5664  7001        IAC
5665  3331        DCA WHERFR /2ND WORD
5666  2256        ISZ PACKIN
5667  6211        CDF 10
5670  1731        TAD I WHERFR
5671  3327        DCA PACKCNT
5672  7240        CLA CMA
5673  1332        TAD WHERTO
5674  3010        DCA 10
5675  7240        CLA CMA
5676  1331      PACKL, TAD WHERFR
5677  1367        TAD (10
5700  3331        DCA WHERFR
5701  1331        TAD WHERFR
5702  1366        TAD (6
5703  3333        DCA LTYPE
5704  1733        TAD I LTYPE
5705  1365        TAD (-1000
5706  7640        SZA CLA
5707  5321        JMP PACKNM
5710  7240        CLA CMA
5711  1331        TAD WHERFR
5712  3011        DCA 11
5713  1364        TAD (-5
5714  3330        DCA PCNT2
5715  1411        TAD I 11
5716  3410        DCA I 10
5717  2330        ISZ PCNT2
5720  5315        JMP .-3
5721  2327      PACKNM, ISZ PACKCNT
5722  5276        JMP PACKL
5723  3410        DCA I 10
5724  1010        TAD 10
5725  6201        CDF
5726  5656        JMP I PACKIN
                /
5727  0000      PACKCNT, 0
5730  0000      PCNT2, 0
5731  0000      WHERFR, 0
5732  0000      WHERTO, 0
5733  0000      LTYPE, 0
5734  0000      MVCNTR, 0
```

```
PAGE 44

                 /
5764   7773   PAGE
5765   7000
5766   0006
5767   0010
5770   5200
5771   7620
5772   0017
5773   0217
5774   1777
5775   2200
5776   7600
5777   7777
```

```
                /PATCHES FOR SABR LOADER V(06)
                /
                BANK=6336
                CUR=6540
                COML=6340
                /
                *6224
6224   5625       JMP I .+1
6225   5202       RLSTRTR
                /
                *6261
6261   7200       CLA /READING SWITCHES
6262   7000       NOP
6263   7000       NOP
                /
                *6447
6447   7000       NOP
6450   7000       NOP
6451   7000       NOP
6452   7200       CLA
                /
                *6461
6461   7000       NOP
6462   1763       1763 /TAD WORD
6463   4664       JMS I .+1
6464   3703       RLOAD
6465   7200       CLA
6466   7000       NOP
6467   7000       NOP
6470   7000       NOP
6471   7000       NOP
                /
                *6652
6652   5653       JMP I .+1
6653   3420       RLERROR /ERROR NUM IN AC
                /
                *7125 /TYPE ROUTINE
7125   4726       JMS I .+1
7126   3440       RLTYP
7127   7000       NOP
                /
                *7200 /DON'T KNOW WHY
7200   7402       HLT
                /
                *7250 /WAIT
7250   5651       JMP I .+1
7251   0000     RESTRT, 0 /CALLER COMES HERE
                /
                *7422 / HSR
7422   4623       JMS I .+1
7423   4000       GETIPT
7424   5621       JMP I .-3
                /
                *7454
7454   7776       7776
```

```
PAGE 46

                    /
                    /RUN TIME FOR FIELDS 1 AND 0
                    /
                    /FIELD 0 FIRST
                    /
                    *233
                    /
0233   0000   0000
0234   6201   6201
0235   6202   6202
0236   4437   4437
0237   0602   0602
                    /
0240   0000   0000
0241   6201   6201
0242   6202   6202
0243   4444   4444
0244   0600   0600
                    /
0245   0000   0000
0246   2045   2045
0247   6201   6201
0250   5445   5445
                    /
0251   0000   0000
0252   2051   2051
0253   6211   6211
0254   5451   5451
                    /
0255   0000   0000
0256   6201   6201
0257   6202   6202
0260   4461   4461
0261   0423   0423
                    /
0262   0000   0000
0263   6201   6201
0264   6202   6202
0265   4466   4466
0266   0400   0400
                    /
0267   0000   0000
0270   6201   6201
0271   6202   6202
0272   4473   4473
0273   0451   0451
                    /
                    /
                    /NOW FIELD 1
                    /
                    *1033
                    /
1033   0000   0000
1034   6211   6211
1035   6202   6202
```

```
PAGE 47

1036  4437  4437
1037  0602  0602
            /
1040  0000  0000
1041  6211  6211
1042  6202  6202
1043  4444  4444
1044  0600  0600
            /
1045  0000  0000
1046  2045  2045
1047  6211  6211
1050  5445  5445
            /
1051  0000  0000
1052  2051  2051
1053  6211  6211
1054  5451  5451
            /
1055  0000  0000
1056  6211  6211
1057  6202  6202
1060  4461  4461
1061  0423  0423
            /
1062  0000  0000
1063  6211  6211
1064  6202  6202
1065  4466  4466
1066  0400  0400
            /
1067  0000  0000
1070  6211  6211
1071  6202  6202
1072  4473  4473
1073  0451  0451
            /
            PAGE
```

CHAPTER 4

CPS FORTRAN LIBRARY

The CPS FORTRAN Library is a collection of relocatable binary files into which the DEC FORTRAN library has been placed. These files were generated using PAPERBIN.

The following are brief descriptions of the general support programs supplied by DEC as part of the FORTRAN-SABR package. Detailed information can be obtained from the FORTRAN manual or from the program listings.

.IOH

Entry points: READ, WRITE, IOH

This subroutine processes the format specifications used in input and output operations by FORTRAN.

Support required: .FLOAT, .INTEGER, .ERROR, .UTILITY

Space required: 13 octal pages

.FLOAT

Entry points: FAD, FSB, FMP, FDV, STO, FLOT, FLOAT,
FIX, IFIX, IFAD, ISTO, CHS, CLEAR

This subroutine does all floating point operations except absolute value which is done in the integer package, .INTEGER.

Support required: .ERROR

Space required: 6 octal pages

.INTEGER

Entry points: IREM, ABS, IABS, DIV, MPY, IRDSW

This subroutine handles integer multiply and divide as well as some floating point operations.

Support required: .ERROR, also .FLOAT if ABS is used

Space required: 2 octal pages

.UTILITY

Entry points: OPEN, CKIO, TTYIN, TTYOUT, HSIN, HSOUT, EXIT

This subroutine provides the necessary I/O calls to the actual devices. It also provides the system exit.

Support required: .FLOAT, of all things

Space required: 1 octal page

.ERROR

Entry points: SETERR, CLRERR, ERROR

This subroutine handles the run time error messages.

Support required: .IOH

Space required: 2 octal pages

.SUBSC

Entry points: SUBSC

This subroutine calculates addresses for subscripted FORTRAN variables.

Support required: .INTEGER

Space required: 1 page

.POWERS

Entry points: IIPOW, IFPOW, FIPOW, FFPOW, EXP, ALOG

This subroutine handles exponentiation.

Support required: .FLOAT, .INTEGER, .ERROR

Space required: 4 octal pages

Note: EXP may overflow (unfortunately) for large negative arguments.

SQRT

Entry points: SQRT

This subroutine finds floating point square roots.

Support required: .ERROR, .FLOAT

Space required: 1 page

TRIG

Entry points: SIN, COS, TAN

This subroutine calculates the TRIG functions for radian arguments.

Support required: .ERROR, .FLOAT

Space required: 2 octal pages

ATAN

Entry points: ATAN

This subroutine finds the principal value in radians.

Support required: .ERROR, .FLOAT

Space required: 2 octal pages

APPENDIX A

F1TAPE

This appendix contains the listing of the one page tape routine used in the *FORTRAN and *SABR support code.

```
                CSUM=RTAPE
                INTS=6147
                IAAC=6171
                IACB=6161
                ICON=6141
                /
                /
                /TAPE ROUTINE FOR READING
                /AND WRITING SIZE 128 BLOCKS ON
                /THE LINC 8
                /
                /THIS IS TO REPLACE THE ONE
                /I STOLE FROM DEC
                /
                *400
                /
0400    0000    RTAPE, 0
0401    7300      CLA CLL
0402    1200      TAD RTAPE /MOVE POINTER
0403    3206      DCA WTAPE
0404    1367      TAD M4000 /TO FORCE FUNCT = 3
0405    5210      JMP INTO
0406    0000    WTAPE, 0
0407    7300      CLA CLL
0410    1370    INTO, TAD C4003 /WRITE WITH SWITCH
0411    3371      DCA FUNCT
0412    1606      TAD I WTAPE /BLOCK NUMBER
0413    3372      DCA BLN
0414    2206      ISZ WTAPE
0415    1606      TAD I WTAPE /NUMB OF BLOCKS
0416    7041      CIA
0417    3373      DCA NUMB
0420    2206      ISZ WTAPE
0421    1606      TAD I WTAPE /UNIT
0422    7112      RTR CLL /PUT INTO BIT 0
0423    1313      TAD C2 /TO SET SEARCH
0424    3374      DCA UNIT
0425    2206      ISZ WTAPE
0426    1606      TAD I WTAPE /CORE LOC
0427    3375      DCA LOC
0430    2376     ISZ CNTR /TIMEOUT TO ALLOW TAPE
0431    5226     JMP .-3  /DRIVE TO SETTLE DOWN
0432    2206      ISZ WTAPE
0433    7120      STL /LINC = 1, FIRST PASS
0434    2372    SERCHA, ISZ BLN /1'S COMPL ON TAPE
0435    3200    SERCHB, DCA CSUM /ZERO CHECK SUM
0436    1316      TAD C7600
0437    3376      DCA CNTR /WORD COUNT
0440    1374      TAD UNIT /NOW SET SERCH
0441    6141      ICON
0442    7201      CLA IAC /TO START MOTION BACKWARDS
0443    7430      SZL /SEE IF MOVING
0444    6141      ICON
0445    1341      TAD C6 /TO CLEAR INTS
```

```
0446  6141  B, ICON
0447  4355  A, JMS WAIT /FOR BLK INT
0450  7500     SMA /ONLY NEG VALID
0451  7120     STL /POS WANT FWD FOR BLK 0
0452  1372     TAD BLN
0453  7650     SNA CLA /L=1, WANT FWD
0454  5270     JMP THERE /ON BLOCK
0455  6147     INTS /WANT MO
0456  7010     RAR /MO TO L, L TO BIT 0
0457  0367     AND M4000
0460  7460     SZA SNL
0461  5247     JMP A /WANT FORE, GOT FORE
0462  7020     CML
0463  7520     SNL SMA
0464  5247     JMP A /BACKWARDS
0465  6141     ICON /STOP
0466  7001     IAC /BIT 0 IS OK HERE
0467  5246     JMP B /CHANGE MOTION
0470  6147  THERE, INTS /WANT M1
0471  7012     RTR
0472  7620     SNL CLA /ON AND FOREW
0473  5247     JMP A /ON AND GOING BACK, REVERSE
0474  1371     TAD FUNCT
0475  6141     ICON /SET BLOCK MODE
0476  7500     SMA /S IF TO WRITE
0477  5330     JMP RDATA
0500  1313     TAD C2 /TO GET 5
0501  6141     ICON /TURN WRITERS ON
0502  7200  WRITE, CLA
0503  6211     CDF 10
0504  1775     TAD I LOC
0505  6211     CDF 10
0506  6161     IACB /AC TO LINC REG
0507  1200     TAD CSUM
0510  3200     DCA CSUM
0511  4355     JMS WAIT /PUT IT OUT
0512  2375     ISZ LOC
0513  0002  C2, 2 /STORAGE, THIS HAS NO EFFECT
0514  2376     ISZ CNTR /DONE?
0515  5302     JMP WRITE /NO
0516  7600  C7600, 7600 /ALSO A CLA
0517  1200     TAD CSUM
0520  6161     IACB /WRITE CHECK SUM
0521  4355     JMS WAIT
0522  4355     JMS WAIT /ALLOW ACTUAL WRITE OF CS
0523  7300  WAIT2, CLA CLL
0524  2373     ISZ NUMB /ALL BLOCKS DONE?
0525  5234     JMP SERCHA /NO
0526  6141     ICON /STOP
0527  5606     JMP I WTAPE /GO HOME
0530  4355  RDATA, JMS WAIT /GUARD
0531  4355  RDTA, JMS WAIT
0532  1200     TAD CSUM
0533  3200     DCA CSUM
0534  6171     IAAC /GET AGAIN
```

```
0535  6211     CDF 10
0536  3775     DCA I LOC /PUT IN CORE
0537  6211     CDF 10
0540  2375     ISZ LOC
0541  0006   C6, 6 /STORAGE, NO EFFECT
0542  2376     ISZ CNTR
0543  5331     JMP RDTA /CONTINUE
0544  4355     JMS WAIT /CSUM
0545  7041     CIA
0546  1200     TAD CSUM
0547  7650     SNA CLA /MAYBE BAD
0550  5323     JMP WAIT2 /THIS IS OK
0551  1316     TAD C7600
0552  1375     TAD LOC /FIX BACK
0553  3375     DCA LOC
0554  5235     JMP SERCHB /TRY AGAIN
0555  0000   WAIT, 0
0556  7300   W1, CLA CLL
0557  6147   C7, INTS
0560  7700     SMA CLA /TAPE?
0561  5356     JMP W1
0562  1357     TAD C7
0563  6141     ICON /CLEAR INTS
0564  7300     CLA CLL
0565  6171     IAAC /GET FROM TAPE
0566  5755     JMP I WAIT
             /
             /
0567  4000   M4000, 4000
0570  4003   C4003, 4003
0571  0000   FUNCT, 0
0572  0000   BLN, 0
0573  0000   NUMB, 0
0574  0000   UNIT, 0
0575  0000   LOC, 0
0576  0000   CNTR, 0
```

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*

Cooley Electronics Laboratory
The University of Michigan
Ann Arbor, Michigan

2a. REPORT SECURITY CLASSIFICATION

UNCLASSIFIED

2b. GROUP

3. REPORT TITLE

CPS Program Logic Manual
Volume III -- CPS 8-K FORTRAN PACKAGE

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Technical Memorandum No. 102-III - 03674-23-M December 1969

5. AUTHOR(S) *(Last name, first name, initial)*

Cederquist, G. N. and Metzger, K.

6. REPORT DATE

December 1969

7a. TOTAL NO. OF PAGES

148

7b. NO. OF REFS

8a. CONTRACT OR GRANT NO.

Nonr-1224(36)

b. PROJECT NO.

NR187-200

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S)

TM102-III

9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

03674-23-M

10. AVAILABILITY/LIMITATION NOTICES

Reproduction in whole or in part is permitted for any purpose of the U. S. Government.

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

Office of Naval Research
Department of the Navy
Washington, D. C. 20360

13. ABSTRACT

CPS is a generalized programming and file management system written for use on the PDP-8 processor of Digital Equipment Corporation's LINC-8 computer. A minimum memory size of 8192 words is required. Extensive use is made of the two tape units present on every LINC-8 for both file storage and system residence. The four volumes entitled CPS System Architecture and Conventions, CPS Basic Programming Package, CPS 8-K FORTRAN Package, and CPS System Utility Programs were written in order to take a snapshot of CPS at one point in its continuing development. This version of CPS is considered to be a first generation system; successive versions are on the drawing boards and internally resemble their parent less and less every day.

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Digital Computer						
Programming System						
PDP-8						
LINC-8						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2*a*. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2*b*. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7*a*. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7*b*. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8*a*. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8*b*, 8*c*, & 8*d*. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9*a*. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9*b*. OTHER REPORT NUMBER(S): If the report has been assigned any other repcrt numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through

_______________________________________."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through

_______________________________________."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through

_______________________________________."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS)*, *(S)*, *(C)*, or *(U)*

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional.

Security Classification