digital EQUIPMENT CORPORATION
MAYNARD, MASSACHUSETTS

OPTION NUMBER

QJ900-FZ
[A00]

SOFTWARE BILL OF MATERIAL

MAINTAINER: SOFTWARE DISTRIBUTION CENTER
(MAYNARD, MASSACHUSETTS 01754)

Page: 01

Date: 18-Mar-77

OPTION TITLE: BASIC/PTS LISTING KIT

SOFTWARE ASSEMBLY POINT: NB GA MR WM
CHECKLIST DISTRIBUTION : NB GA MR WM

Sel Chk	Qty	Program Code	Program Title
-	1	DEC-11-LPTBA-B-LA	LISTING OF BASIC/PTS

KIT CONTENTS:
1 SYS Listing

IDENTIFICATION

AB-2045B-SC

PRODUCT CODE	DEC-11-LPTBA-B-LA
PRODUCT NAME	LISTING OF BASIC/PTS
DATE CREATED	MAY 1974
MAINTAINER	DEVELOPMENT
AUTHOR	A. STANKARD

```
                        .NOSTR=1
                        .EOT
```

```
                        ;
                        ; BASIC/PTS  PART1-BASICL
                        ;
                        ; DEC-11-LPTBA-A-L1
                        ;
                        ; COPYRIGHT 1973
                        ;
                        ; DIGITAL EQUIPMENT CORPORATION
                        ; MAYNARD,MASSACHUSETTS 01754
                        ;
```



```
                            .TITLE  BASICL  V001A EDIT #61 (REF) 03/21/73
                        ;                               SOURCE FILE #1
000000                      .ASECT
                        ;
                        ;
                        ;
                        ; BASIC CONSISTS OF THREE MODULES:
                        ;       1) BASICL - INTERPRETER WITH SYSTEM I/O
                        ;       2) FPMP-11 - MATH PACKAGE CONDITIONALIZED FOR USE IN BASIC
                        ;       3) BASICH - USER AREA AND ONCE-ONLY CODE
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ; ORIGINAL PAPER TAPE VERSION   BY: LEN LEKMAN
                        ;
                        ; MODIFICATION AND COMMENTS     BY: ROY FOLK
                        ;
                        ; COMPLETION AND FURTHER
                        ; COMMENTS                      BY: ANN STANKARD
                        ;
```

```
                        ;*****************************************************
                        ;*****************************************************
                        ;*************                             *************
                        ;*************  GLOBALS( ASSY, PARAMS,)    *************
                        ;*************  DESCRIPTION( LOW CORE)     *************
                        ;*************  GENERAL TABLES,            *************
                        ;*************  CONSTANTS, STORAGE         *************
                        ;*************                             *************
                        ;*****************************************************
                        ;*****************************************************



                        ;
                        ; GLOBALS
                        ;
                        ; --FPMP=11
                        ;
                                ,GLOBL  $POLSH
                                ,GLOBL  $IR
                                ,GLOBL  $MLR
                                ,GLOBL  $OVR
                                ,GLOBL  $AOR,$SBR
                                ,GLOBL  SIN,COS,SQRT,ALOG,ATAN,EXP
                                ,GLOBL  $ERVEC
                        ;
                        ; --BASIC2
                        ;
                                ,GLOBL  TPB, TKS
                                ,GLOBL  START
                                ,GLOBL  USRAREA
                                ,GLOBL  FILLCO
                                ,GLOBL  LIMIT, POL, ARRAYS, PDSIZE
                                ,GLOBL  ERRPDL, ERRMIX, ERRSYN, ERRARG
                                ,GLOBL  TABLES, TBL5EN
                                ,GLOBL  EVAL, GETVAR, STOVAR, MSG
                                ,GLOBL  COLUMN, FAC1, FAC2, ,RPAR, ,LPAR
                                ,GLOBL  INT, MAKEST, OPRATO, SOPRAT
                                ,GLOBL  ,COMMA, T1, T2, T3, ,EOL
                                ,GLOBL  NUMOUT, NUMSGN, ARGB, STPRO
                                ,GLOBL  SAVCHAR,VAL,NORM
                                ,GLOBL  RND1, RND2
                                ,GLOBL  $STKSZ, ERRFPU
```



```
                        ;
                        ; ASSEMBLY PARAMS,
                        ;
                                ,IFNDF  $STKSZ  ;SIZE OF STACK
                        $STKSZ  =200            ;BYTES
                                ,ENDC
                        ;
                                ,IFNDF  $PRORG  ;PROG, ORIGIN AFTER VECTOR SPACE
                        $PRORG  =400
                                ,ENDC
                        ;
                                ,IFNDF  $PPBSZ  ;PP BUFF, SIZE
                        $PPBSZ  =30
                                ,ENDC
                        ;
                                ,IFNDF  $PRBSZ  ;PAPER-TAPE READER BUFF, SIZE
                        $PRBSZ  =30
                                ,ENDC
                        ;
                                ,IFNDF  $LPBSZ  ;LINE-PRINTER BUFF, SIZE
                        $LPBSZ  =40
                                ,ENDC
                        ;
                        ;  $NOPTP = NO PAPER TAPE; DEFINE TO ELIMINATE CODE FOR HIGH SPEED
                        ;   PAPER TAPE
                        ;
                        ;  $NOLPT = NO LINE PRINTER; DEFINE TO ELIMINATE LINE PRINTER CODE
                        ;
                        ;  $LONGER = LONG ERROR MESSAGES; DEFINE TO GET LONG,
                        ;       EXPLANATORY ERROR MESSAGES,
                        ;
                        ;  $NOSTR = NO STRINGS; DEFINE TO ELIMINATE STRING VARIABLE CODE
                        ;
                        ;  $NOPOW = NO POWER FAIL; DEFINE TO ASSEMBLE WITHOUT
                        ;       POWER FAIL/RESTART ROUTINE,
                        ;
                        ;
                        ;
```

```
123                                 ;
124                                 ; HOT POOP!!
125                                 ;
126                                 ; --USER AREA MAP
127                                 ;
128                                 ; HIGH ADDRESSES+----------------+
129                                 ;               ! GOSUB POINTERS! <--LIMIT(R5) NEEDED
130                                 ;               ! 26 FN POINTERS!
131                                 ;               ! READ POINTER  ! <--PDL(R5) NEEDED
132                                 ;               +----------------+
133                                 ;               !  PUSH          !
134                                 ;               !  DOWN          !    PDSIZE(R5) NEEDED  =
135                                 ;               !  LIST          !  (PDL(R5)-ARRAYS(R5))/2-STACK(INTERRUPT)
136                                 ;               +----------------+
137                                 ;               !              ! <--ARRAYS(R5) NEEDED
138                                 ;               !  ARRAYS      !
139                                 ;               !              !
140                                 ;               !            \!/
141                                 ;               !             V
142                                 ;               +----------------+
143                                 ;               !   STRINGS      !
144                                 ;               +----------------+
145                                 ;               !              A
146                                 ;               !            /!\
147                                 ;               !             !
148                                 ;               ! SYMBOLS     !
149                                 ;               +----------------+
150                                 ;               !             A
151                                 ;               !            /!\
152                                 ;               ! INTERP.     !
153                                 ;               ! CODE        ! <--CODE(R5) NEEDED
154                                 ;               +----------------+
155                                 ;               !  USER        !
156                                 ;               !  LINE        !
157                                 ;               !  BUFFER      ! <--LINE(R5) NEEDED
158                                 ;               +----------------+
159                                 ;               ! USER I/O BUFF.!
160                                 ;               ! AND BUFF. HDR.!
161                                 ;               ! SPACE (TTY)   !
162                                 ;               +----------------+
163                                 ;               !  USER AREA    !
164                                 ;               ! STORAGE CELLS !
165                                 ; R5 POINTS-->  !              !
166                                 ;               +----------------+
167                                 ;
168
```



```
169                                 ;
170                                 ; --
171                                 ;
172                                 ; CODE AREA CONTAINS STRING OF CODE BYTES WHICH COMPRISE
173                                 ;  THE COMPILED CODE TO BE INTERPETED.  THE SYNTAX SCAN OF THESE
174                                 ;  CODE BYTES IS FACILITATED BY THE JUDICIOUS PLACEMENT
175                                 ;  OF CERTAIN OF THESE CODE BYTES TO BE CALLED 'TOKENS',
176                                 ;  IF THE HIGH BIT OF A TOKEN IS ON
177                                 ;  IT IS A SYSTEM SYMBOL (S.SYM), (SEE TABLE, BELOW,)
178                                 ;  HIGH BIT OFF SIGNALS TOKEN AS THE HIGH ORDER BYTE OF A WORD
179                                 ;  (LOW ORDER BYTE FOLLOWS TOKEN) USED AS AN ADDRESS OFFSET INTO
180                                 ;  THE USER SYMBOL TABLE (USPTR).
181                                 ;
182                                 ; --SYMBOL TABLE ENTRIES
183                                 ;
184                                 ; ARE MADE FOR LINE NUMBERS, SCALARS, NUMERIC ARRAYS, AND
185                                 ;  STRINGS, AS SHOWN (IN ORDER):
186                                 ;
187                                 ;       LINENO  177775  177776  177777
188                                 ;       ADDRESS VALUE   ADDRESS ADDRESS
189                                 ;               VALUE   MAXSS1  MAXSS1
190                                 ;               0       MAXSS2  MAXSS2
191                                 ;               VARNAME VARNAME STRINGNAME
192                                 ;
193                                 ; --GENERAL REGISTER USAGE
194                                 ;
195                                 ; R1 IS BASIC EXECUTION PC
196                                 ; R4 IS THE EXECUTION STACK DEPTH
197                                 ; R5 IS THE POINTER TO THE USER AREA
198
```

```
                                        ;
                                        ; INTERUPT VECTORS AND LOWEST CODE
                                        ;
                        000000          .       =0
000000  000167  001030                  JMP     START           ;RESTART AT LOC, #0
                        000004          .       =4
000004  000006  000000                  .       .+2,0           ;TIME OUT, BUSS ERROR
                        000010          .       =10
000010  000012  000000                  .       .+2,0           ;RESERVED INSTRUCTION
                        000014          .       =14
000014  000016  000000                  .       .+2,0           ;DEBUG, TRAP VEC,
                        000020          .       =20
000020  007742  000000                  .       BOMB,0          ;IOT TRAP VEC,

                        000024          .       =24

                                        ;IFNDF  $NOPOW
000024  007330  000000                  .       POWDWN,0        ;POWER FAIL/RESTART
                                        ;ENDC

                                        ;IFDF   $NOPOW
                                        .       .+2,0   ;HALT ON POWER FAIL
                                        ;ENDC

                        000030          .       =30
000030  000032  000000                  .       .+2,0           ;EMT TRAP
                        000034          .       =34
000034  000036  000000                  .       .+2,0           ;'TRAP' TRAP
                                        ;
                                        ; +-'SYSTEM' AREA
                                        ;
                        000040          .       =40
000040     001     101          000000  VERNUM: .BYTE   001,'A  ;VERSION NUMBER
000042  000000  000000  000000                  .       0,0,0,0,0,0,0
000050  000000  000000  000000
000056  000000
                                        ;
                                        ; +-INTERUPT VECTORS
                                        ;
                        000060          .       =60
000060  006470  000200                  .       KBINT,PR4       ;KEYBOARD
                        000064          .       =64
000064  006642  000200                  .       TPINT,PR4       ;TELEPRINTER
                        000070          .       =70
                                        ;IFNDF  $NOPTP
000070  007104  000200                  .       PTRINT,PR4      ;HSP, PT, READER
                        000074          .       =74
000074  007204  000200                  .       PPINT,PR4       ;HSP, PT, PUNCH
                                        ;ENDC
                                        ;IFDF   $NOPTP
                                        .       0,0,0,0
                                        ;ENDC
                        000200          .       =200
                                        ;IFNDF  $NOLPI
000200  007256  000200                  .       LPINT,PR4
```



```
                                        ;ENDC
                                        ;IFDF   $NOLPI
                                        .       0,0
                                        ;ENDC
                        000240          .       =240
000240  000242  000000                  .       .+2,0           ;PIRQ
                        000244          .       =244
000244  0000006 000000                  .       ERRFPV,0        ;FLOATING POINT TRAP

                        000400          .       =$PRORG         ;START OF CODE (MAY HAVE TO CHANGE,
                                                                ;  DEPENDING UPON ATTACHED DEVS.)
```

```
                        ;
                        ; GENERAL ASSIGNMENTS, VARIABLES, GLOBALS, ETC;
                        ;

                        ; --REGISTER ASSIGNMENTS
                        ;
000000                  R0      =%0
000001                  R1      =%1
000002                  R2      =%2
000003                  R3      =%3
000004                  R4      =%4
000005                  R5      =%5
000006                  SP      =%6
000007                  PC      =%7
                        ;
                        ; --DEVICE REGISTER CONSTANTS
                        ;
177560                  TKS     =177560
177562                  TKB     =177562
177564                  TPS     =177564
177566                  TPB     =177566
177550                  PRS     =177550
177552                  PRB     =177552
177554                  PPS     =177554
177556                  PPB     =177556
177514                  LPS     =177514
177516                  LPB     =177516
                        ;
                        ; --GENERAL CONSTANTS
                        ;
000011                  TAB     =11
000012                  LF=     012
000014                  FF      =14
000015                  CR=     015
000040                  BL=     040
177776                  PS      =177776
000140                  PR3     =140            ;PRIORITY LEVEL 3
000200                  PR4     =200
000340                  PR7     =340
177775                  .SCALAR =177775         ;USER SYM, TABLE FLAG FOR SCALAR
177776                  .NVAR   =177776         ;FLAG FOR NUMERIC ARRAY VARIABLE
177777                  .SVAR   =177777         ;FLAG FOR STRING VAR.
```



```
                        ;
                        ; --USER AREA STORAGE CELL OFFSETS
                        ;
000000                  SYMBOLS =       0               ;CONTAINS ADDR OF THE FIRST SYMBOL
000002                  LIMIT   =       SYMBOLS+2       ;CONTAINS ADDR OF HIGHEST WD OF USER AREA
000004                  POL     =       LIMIT+2         ;CONTAINS ADDR OF EMPTY STACK
000006                  POSIZE  =       POL+2           ;CONTAINS LOW LIMIT OF STACK
000010                  ARRAYS  =       POSIZE+2        ;CONTAINS ADDR OF HIGHEST WORD OF ARRAYS
000012                  HIFREE  =       ARRAYS+2        ;CONTAINS ADDR OF HIGHEST FREE WORD
000014                  LOFREE  =       HIFREE+2        ;CONTAINS ADDR OF LOWEST FREE WORD
000016                  CODE    =       LOFREE+2        ;CONTAINS ADDR OF INTERPRETIVE CODE
000020                  LINE    =       CODE+2          ;CONTAINS ADDR OF USER LINE BUFFER
000022                  VARSAV  =       LINE+2          ;VAR SAVE FOR ASSIGNMENT
000024                  SS1SAVE =       VARSAV+2        ;SS1 SAVE FOR ASSIGNMENT
000026                  SS2SAVE =       SS1SAVE+2       ;SS2 SAVE FOR ASSIGNMENT
000030                  LINENO  =       SS2SAVE+2       ;CONTAINS THE LINE NUMBER
000032                  GSBCTR  =       LINENO+2        ;CONTAINS 33*DEPTH OF ACTIVE GOSUBS
000034                  COLUMN  =       GSBCTR+2        ;HAS ADDR OF COL CT FOR CURRENT DEV
000036                  CLMNTTY =       COLUMN+2        ;LAST TTY COLUMN TYPED
000040                  FAC1    =       CLMNTTY+2       ;HIGH ORDER FLOATING VALUE
000042                  FAC2    =       FAC1+2          ;LOW ORDER FLOATING VALUE
000044                  R0SAVE  =       FAC2+2          ;PLACE FOR R0 WHILE IN FPMP11
000046                  R1SAVE  =       R0SAVE+2        ;DITTO R1
000050                  R2SAVE  =       R1SAVE+2        ;DITTO R2
000052                  R3SAVE  =       R2SAVE+2        ;DITTO R3
000054                  R4SAVE  =       R3SAVE+2        ;DITTO R4
000056                  T1      =       R4SAVE+2        ;SHORT TERM TEMPORARY
000060                  T2      =       T1+2            ;DITTO
000062                  T3      =       T2+2            ;DITTO
000064                  RND1    =       T3+2            ;HISTORY OF RND
000066                  RND2    =       RND1+2          ;DITTO
000070                  RNDCT   =       RND2+2          ;RANDOMIZER
000072                  LOSTR   =       RNDCT+2         ;LOW LIMIT OF STRING STORAGE
000074                  HISTR   =       LOSTR+2         ;HIGH LIMIT OF STRING STORAGE
                        ; --THESE NEXT TWO MUST BE LO BYTE AND HI BYTE, RESPECTIVELY,
                        ;   OF SAME WORD
000076                  ODEV    =       HISTR+2         ;OUTPUT DEV CODE (TTY=0,PT=2,LP=4)
000077                  IDEV    =       ODEV+1          ;INPUT DEV CODE (TTY=0,PR=2)
000100                  TPHD    =       ODEV+2          ;HOLDS START OF TELEPRT BUFF HDR
000102                  KBHD    =       TPHD+2          ;HOLDS START OF KBD BUFF HDR
000104                  ECHOSP  =       KBHD+2          ;ADDR OF NEXT ECHO CHAR (IF NON-0)
000106                  CNCFLG  =       ECHOSP+2        ;'+C' FLAG, NON=0 IF PENDING
000107                  CNOFLG  =       CNCFLG+1        ;'+O' FLAG, DITTO
000110                  FILLCO  =       CNOFLG+1        ;FILL-COUNT
000111                  FILLNO  =       FILLCO+1        ;NO OF FILL CHARS
000112                  FILLCH  =       FILLNO+1        ;CHAR BEFORE FILL
000113                  SPARE   =       FILLCH+1        ; [SPARE BYTE]
                        ;
                        ; --I/O BUFFER HEADER OFFSETS
                        ;
000000                  BSTRT   =0      ;(R1)   START OF BUFF,
000002                  BEND    =2      ;BEND(R1)       END OF BUFFER
000004                  BGET1   =4      ;BGET1(R1)      FIRST GET POINTER
000006                  BGET2   =6      ;BGET2(R1)      SECOND GET PTR;
000010                  BPUT    =10     ;BPUT(R1)       PUT PTR.
```

```
                        000012          BFSPEC  =12    ;BFSPEC(R1)     SPECIAL WORD = USE PARTIC. TO DEV.
                                        ;
                                        ; **SYSTEM VARIABLES
                                        ;
000400  000000          CLMNPP: .WORD   0       ;PAPER TAPE PUNCH COLUMN POSITION
000402  000000          CLMNLP: .WORD   0       ;LINE PRINTER COLUMN POSITION
```



```
                                        ;
                                        ; -- SYSTEM TOKEN DEFINITIONS
                                        ;
                                        ;
                        000201          .EOL=   201     ;BEGINNING OF SEQUENCE USED BY 'KEYWDS' AND 'TABLE1'
                        000202          .END=   .EOL+1
                        000203          .FOR=   .END+1
                        000204          .GOSUB= .FOR+1
                        000205          .GOTO=  .GOSUB+1
                        000206          .IF=    .GOTO+1
                        000207          .INPUT= .IF+1
                        000210          .LET=   .INPUT+1
                        000211          .NEXT=  .LET+1
                        000212          .PRINT= .NEXT+1
                        000213          .RETURN=.PRINT+1
                        000214          .STOP=  .RETURN+1
                        000215          .DIM=   .STOP+1
                        000216          .RANDOM=.DIM+1
                        000217          .RESTOR=.RANDOM+1
                        000220          .REM=   .RESTOR+1
                        000221          .DEF=   .REM+1
                        000222          .READ=  .DEF+1
                        000223          .DATA=  .READ+1
                        000224          .CALL=  .DATA+1
                        000225          .EOF=   .CALL+1 ;END OF SEQUENCE USED BY 'TABLE1'
                        000226          .AMPERS=.EOF+1
                        000227          .UPARRO=.AMPERS+1;BEGINNING OF SEQUENCE USED BY 'TABLE2' AND OPR PREC.
                        000230          .STAR=  .UPARRO+1
                        000231          .SLASH= .STAR+1
                        000232          .PLUS=  .SLASH+1
                        000233          .UNARY= .PLUS+1
                        000234          .MINUS= .UNARY+1
                        000235          .TERM=  .MINUS+1;END OF SEQUENCE USED BY 'TABLE2'
                        000236          .SEMI=  .TERM+1
                        000237          .RPAR=  .SEMI+1
                        000240          .TO=    .RPAR+1
                        000241          .STEP=  .TO+1
                        000242          .THEN=  .STEP+1
                        000243          .COMMA= .THEN+1
                        000244          .LE=    .COMMA+1
                        000245          .EL=    .LE+1
                        000246          .GE=    .EL+1
                        000247          .EG=    .GE+1
                        000250          .NE=    .EG+1
                        000251          .EN=    .NE+1
                        000252          .LT=    .EN+1
                        000253          .GT=    .LT+1
                        000254          .EQ=    .GT+1   ;END OF SEQUENCE USED BY OPERATOR PRECEDENCE
                        000255          .LPAR=  .EQ+1
                        000256          .DQUOT= .LPAR+1
                        000257          .SQUOT= .DQUOT+1
                        000260          .COLON  =.SQUOT+1
                        000261          .POUND= .COLON+1
                        000262          .FN=    .POUND+1
                        000263          .RNDL=  .FN+1   ;BEGINNING OF SEQUENCE USED BY 'TABLE5'
```

```
                000264                  ,RND=   ,RNDL+1
                000265                  ,SIN=   ,RND+1
                000266                  ,COS=   ,SIN+1
                000267                  ,SQR=   ,COS+1
                000270                  ,ATN=   ,SQR+1
                000271                  ,EXP=   ,ATN+1
                000272                  ,LOG=   ,EXP+1
                000273                  ,ABS=   ,LOG+1
                000274                  ,INT=   ,ABS+1
                000275                  ,SGN=   ,INT+1
                000276                  ,TAB=   ,SGN+1
                                                ,IFNDF  $NOSTR
                                        ,LEN=   ,TAB+1
                                        ,ASC=   ,LEN+1
                                        ,CHRS=  ,ASC+1
                                        ,POS=   ,CHRS+1
                                        ,SEG=   ,POS+1
                                        ,VAL=   ,SEG+1
                                        ,STR=   ,VAL+1  ;END OF SEQUENCE USED BY 'TABLE5'
                                        ,LIST=  ,STR+1  ;BEGINNING OF SEQUENCE USED BY 'TABLE4'
                                                ,ENDC
                                                ,IFDF   $NOSTR
                000277                  ,LIST=  ,TAB+1
                                                ,ENDC
                000300                  ,RUN=   ,LIST+1
                000301                  ,SAVE=  ,RUN+1
                000302                  ,OLD=   ,SAVE+1
                000303                  ,SCR=   ,OLD+1
                000304                  ,CLEAR= ,SCR+1  ;END OF SEQUENCE USED BY 'TABLE4' AND 'KEYWDS'
                000374                  ,FLIT=  374
                000375                  ,ILIT1= 375
                000376                  ,ILIT2= 376
                000377                  ,TEXT   =377
```



```
                                        ;
                                        ; *-KEYWORD TABLE
                                        ;
                                        ;       ENTRIES ARE ORDERED BY THEIR FREQUENCY OF
                                        ;       OCCURRENCE,  ALPHA KEYWORDS START AT 'KEYA',
                                        ;
000404     053                  KEYWDS: ,ASCII  '+'
000405     232                          ,BYTE   ,PLUS
000406     055                          ,ASCII  '-'
000407     234                          ,BYTE   ,MINUS
000410     052                          ,ASCII  '*'
000411     230                          ,BYTE   ,STAR
000412     057                          ,ASCII  '/'
000413     231                          ,BYTE   ,SLASH
000414     136                          ,ASCII  '^'
000415     227                          ,BYTE   ,UPARRO
000416     050                          ,ASCII  '('
000417     255                          ,BYTE   ,LPAR
000420     051                          ,ASCII  ')'
000421     237                          ,BYTE   ,RPAR
000422     134                          ,ASCII  '\'
000423     201                          ,BYTE   ,EOL
000424     046                          ,ASCII  '&'
000425     226                          ,BYTE   ,AMPERS
000426     073                          ,ASCII  ';'
000427     236                          ,BYTE   ,SEMI
000430     054                          ,ASCII  ','
000431     243                          ,BYTE   ,COMMA
000432  036474                          ,ASCII  '<='
000434     244                          ,BYTE   ,LE
000435     075     074                  ,ASCII  '=<'
000437     244                          ,BYTE   ,LE
000440  036476                          ,ASCII  '>='
000442     246                          ,BYTE   ,GE
000443     075     076                  ,ASCII  '=>'
000445     246                          ,BYTE   ,GE
000446  037074                          ,ASCII  '<>'
000450     250                          ,BYTE   ,NE
000451     076     074                  ,ASCII  '><'
000453     250                          ,BYTE   ,NE
000454     074                          ,ASCII  '<'
000455     252                          ,BYTE   ,LT
000456     076                          ,ASCII  '>'
000457     253                          ,BYTE   ,GT
000460     075                          ,ASCII  '='
000461     254                          ,BYTE   ,EQ
000462     042                          ,ASCII  '"'
000463     256                          ,BYTE   ,DQUOT
000464     047                          ,ASCII  "'"
000465     257                          ,BYTE   ,SQUOT
000466     072                          ,ASCII  ':'
000467     260                          ,BYTE   ,COLON
000470     043                          ,ASCII  '#'
000471     261                          ,BYTE   ,POUND
000472     133                          ,ASCII  '['
```

```
 516  000473     255                          ,BYTE   ,LPAR
 517  000474     135                          ,ASCII  /)/
 518  000475     237                          ,BYTE   ,RPAR
 519  000476  042514  020124          KEYA:   ,ASCII  /LET /          ;START OF ALPHA KEYWORDS
 520
 521  000502     210                          ,BYTE   ,LET
 522  000503     111  020106                  ,ASCII  /IF /
 523  000506     206                          ,BYTE   ,IF
 524  000507     107  020117  047524          ,ASCII  /GO TO /
      000514     040
 525  000515     205                          ,BYTE   ,GOTO
 526  000516  047506  020122                  ,ASCII  /FOR /
 527  000522     203                          ,BYTE   ,FOR
 528  000523     040  047524     040          ,ASCII  / TO /
 529  000527     240                          ,BYTE   ,TO
 530  000530  042516  052130     040          ,ASCII  /NEXT /
 531  000535     211                          ,BYTE   ,NEXT
 532  000536  052040  042510  020116          ,ASCII  / THEN /
 533  000544     242                          ,BYTE   ,THEN
 534  000545     040  052123  050105          ,ASCII  / STEP /
      000552     040
 535  000553     241                          ,BYTE   ,STEP
 536  000554  047507  052523  020102          ,ASCII  /GOSUB /
 537  000562     204                          ,BYTE   ,GOSUB
 538  000563     122  052105  051125          ,ASCII  /RETURN/
      000570     116
 539  000571     213                          ,BYTE   ,RETURN
 540  000572  047111  052520  020124          ,ASCII  /INPUT /
 541  000600     207                          ,BYTE   ,INPUT
 542  000601     120  044522  052116          ,ASCII  /PRINT /
      000606     040
 543  000607     212                          ,BYTE   ,PRINT
 544  000610  042522     115                  ,ASCII  /REM/
 545  000613     220                          ,BYTE   ,REM
 546  000614  042504  020106                  ,ASCII  /DEF /
 547  000620     221                          ,BYTE   ,DEF
 548  000621     122  040505  020104          ,ASCII  /READ /
 549  000626     222                          ,BYTE   ,READ
 550  000627     104  052101  020101          ,ASCII  /DATA /
 551  000634     223                          ,BYTE   ,DATA
 552  000635     103  046101  020114          ,ASCII  /CALL /
 553  000642     224                          ,BYTE   ,CALL
 554  000643     106     116                  ,ASCII  /FN/
 555  000645     262                          ,BYTE   ,FN
 556  000646  047122  024104                  ,ASCII  /RND(/
 557  000652     263                          ,BYTE   ,RNDL
 558  000653     122  042116                  ,ASCII  /RND/
 559  000656     264                          ,BYTE   ,RND
 560  000657     123  047111     050          ,ASCII  /SIN(/
 561  000663     265                          ,BYTE   ,SIN
 562  000664  047503  024123                  ,ASCII  /COS(/
 563  000670     266                          ,BYTE   ,COS
 564  000671     123  051121     050          ,ASCII  /SQR(/
 565  000675     267                          ,BYTE   ,SQR
 566  000676  052101  024116                  ,ASCII  /ATN(/
```



```
 567  000702     270                          ,BYTE   ,ATN
 568  000703     105  050130     050          ,ASCII  /EXP(/
 569  000707     271                          ,BYTE   ,EXP
 570  000710  047514  024107                  ,ASCII  /LOG(/
 571  000714     272                          ,BYTE   ,LOG
 572  000715     101  051502     050          ,ASCII  /ABS(/
 573  000721     273                          ,BYTE   ,ABS
 574  000722  047111  024124                  ,ASCII  /INT(/
 575  000726     274                          ,BYTE   ,INT
 576  000727     123  047107     050          ,ASCII  /SGN(/
 577  000733     275                          ,BYTE   ,SGN
 578  000734  040524  024102                  ,ASCII  /TAB(/
 579  000740     276                          ,BYTE   ,TAB
 580
 581                                          ,IFNDF  $NOSTR
 582                                          ,ASCII  /LEN(/
 583                                          ,BYTE   ,LEN
 584                                          ,ASCII  /ASC(/
 585                                          ,BYTE   ,ASC
 586                                          ,ASCII  /CHR$(/
 587                                          ,BYTE   ,CHR$
 588                                          ,ASCII  /POS(/
 589                                          ,BYTE   ,POS
 590                                          ,ASCII  /SEG$(/
 591                                          ,BYTE   ,SEG$
 592                                          ,ASCII  /VAL(/
 593                                          ,BYTE   ,VAL
 594                                          ,ASCII  /STR$(/
 595                                          ,BYTE   ,STR$
 596                                          ,ENDC
 597
 598  000741     104  046511     040          ,ASCII  /DIM /
 599  000745     215                          ,BYTE   ,DIM
 600  000746  040522  042116  046517          ,ASCII  /RANDOMIZE/
      000754  055111     105
 601  000757     216                          ,BYTE   ,RANDOM
 602  000760  042522  052123  051117          ,ASCII  /RESTORE/
      000766     105
 603  000767     217                          ,BYTE   ,RESTORE
 604  000770  052123  050117                  ,ASCII  /STOP/
 605  000774     214                          ,BYTE   ,STOP
 606  000775     105  042116                  ,ASCII  /END/
 607  001000     272                          ,BYTE   ,END
 608  001001     114  051511     124          ,ASCII  /LIST/
 609  001005     277                          ,BYTE   ,LIST
 610  001006  052522     116                  ,ASCII  /RUN/
 611  001011     300                          ,BYTE   ,RUN
 612  001012  040523  042526                  ,ASCII  /SAVE/
 613  001016     301                          ,BYTE   ,SAVE
 614  001017     117  042114                  ,ASCII  /OLD/
 615  001022     302                          ,BYTE   ,OLD
 616  001023     123  051103                  ,ASCII  /SCR/
 617  001026     303                          ,BYTE   ,SCR
 618  001027     103  042514                  ,ASCII  /CLE/
 619  001032     304                          ,BYTE   ,CLEAN
```

```
 620
 621 001033  000                    .BYTE   0               ;END OF TABLE FLAG
 622                                .EVEN
 623                                .EOT
 624
```



```
 625                                ;                                   SOURCE FILE #2
 626                                ;*****************************************************
 627                                ;*****************************************************
 628                                ;***********                               ***********
 629                                ;***********   MAIN FLOW ROUTINES          ***********
 630                                ;***********                               ***********
 631                                ;*****************************************************
 632                                ;*****************************************************
 633
 634
 635
 636
 637                                ;
 638                                ; START = PROGRAM STARTING POINT
 639                                ;
 640 001034  016705  000000G        START:  MOV     USRAREA,R5
 641 001040  016506  000004                 MOV     POL(R5),SP
 642 001044  005065  000104                 CLR     ECHOSP(R5)
 643 001050  004767  007056                 JSR     PC,BUFCLR       ;CLEAR OUT ALL I/O RING BUFFERS
 644 001054  016506  000004        SCRATCH:MOV     POL(R5),SP
 645 001060  004767  012700                 JSR     PC,INITSCR
 646 001064  004767  007334        CLEAR:  JSR     PC,CLRVARS
 647 001070  000402                        BR      READY
 648
```

```
                                        ;+-----------------------------------------------------------
                                        ;
                                        ; READY - PRINT 'READY'
                                        ;
001072  004767  007034          READY0: JSR     PC,BUFCLR       ;CLEAR ALL I/O RING BUFFERS
001076  016506  000004          READY:  MOV     PDL(R5),SP
001102  005065  000106                  CLR     CNCFLG(R5)      ;'C' AND 'O' FLAGS
001106  005065  000076                  CLR     ODEV(R5)
001112  012765  000036  000034          MOV     #CLMNITY,COLUMN(R5)     ;SET TO COUNT TTY COLUMNS
001120  060565  000034                  ADD     R5,COLUMN(R5)   ; NOW
001124  004167  014344                  JSR     R1,MSG
001130     015     012                  .BYTE   CR,LF
001132  042522  042101     131          .ASCII  'READY'
001137     000                          .BYTE 0
                                        .EVEN
001140  016506  000004          READY2: MOV     PDL(R5),SP
001144  004167  014324                  JSR     R1,MSG
001150     015     012     012          .BYTE   CR,LF,LF
001153     000                          .BYTE 0
                                        .EVEN
001154  005065  000076          IOINIT: CLR     ODEV(R5)        ;CLR ODEV AND IDEV BYTES IN USER AREA
```



```
                                        ;+-----------------------------------------------------------
                                        ;
                                        ; EDIT -
                                        ;
                                        ;  GET LINE, TRANSLATE INTO TOKENS, AND MOVE LINE INTO
                                        ;   USER CODE SPACE, WITH ALL ENTRIES MADE INTO USER
                                        ;   SYMBOL TABLE, ARRAY SPACE, AND STRING SPACE,
                                        ;
001160  004767  013160          EDIT:   JSR     PC,LINGET
001164  103006                          BCC     EDITL           ;NORMAL INPUT LINE
001166  127527  000016  000225          CMPB    @CODE(R5),#,EOF ;CHECK EMPTY PROGRAM
001174  001336                          BNE     READY0          ;NO, RETURN TO TTY INPUT
001176  000167  015642                  JMP     DEVERR          ;YES, DEVICE NOT READY
001202  004767  016146          EDITL1: JSR     PC,TRAN
001206  005201                          INC     R1
001210  010146                          MOV     R1,-(SP)
001212  016501  000020                  MOV     LINE(R5),R1
001216  160116                          SUB     R1,(SP)
001220  112100                          MOVB    (R1)+,R0
001222  100417                          BMI     EDIMMED
001224  000300                          SWAB    R0
001226  152100                          BISB    (R1)+,R0
001230  061500                          ADD     (R5),R0
001232  021027  177775                  CMP     (R0),#,SCALAR
001236  103011                          BHIS    EDIMMED
001240  016501  000016                  MOV     CODE(R5),R1
001244  021627  000003                  CMP     (SP),#3         ;IF THERE ARE ONLY 3 BYTES THEN THE LINE
001250  001013                          BNE     EDISRCH         ;CONATINS LINENO,CR AND MEANS DELETE,
001252  005016                          CLR     (SP)
001254  005060  000002                  CLR     2(R0)
001260  000407                          BR      EDISRCH
001262  105765  000077          EDIMMED:TSTB    IDEV(R5)
001266  001334                          BNE     EDIT            ;IGNORE IMM STMTS FROM NON-TTY
001270  000167  000454                  JMP     IMMED
001274  004767  015660          EDISKIP:JSR     PC,SKIPEOL
001300  010104          EDISRCH:MOV     R1,R4           ;R0 POINTS TO LINENO OF THE NEW LINE,
001302  121127  000225                  CMPB    (R1),#,EOF      ;@(SP) CONTAINS LENGTH OF THE NEW LINE,
001306  001452                          BEQ     EDIPUT
001310  111103                          MOVB    (R1),R3
001312  100770                          BMI     EDISKIP
001314  005201                          INC     R1
001316  000303                          SWAB    R3
001320  152103                          BISB    (R1)+,R3
001322  061503                          ADD     (R5),R3
001324  021327  177775                  CMP     (R3),#,SCALAR
001330  103361                          BHIS    EDISKIP
001332  021013                          CMP     (R0),(R3)
001334  101357                          BHI     EDISKIP
001336  001036                          BNE     EDIPUT
001340  004767  015614          EDIPASS:JSR     PC,SKIPEOL
001344  121127  000225                  CMPB    (R1),#,EOF
001350  001427                          BEQ     EDIOVER
001352  111103                          MOVB    (R1),R3
001354  100771                          BMI     EDIPASS
001356  005201                          INC     R1
```

```
001360 000303                   SWAB    R3
001362 152103                   BISB    (R1)+,R3
001364 061503                   ADD     (R5),R3
001366 021327 177775            CMP     (R3),#,SCALAR
001372 103362                   BHIS    EDIPASS
001374 162701 000002            SUB     #2,R1
001400 016500 000004            MOV     PDL(R5),R0      ;IF A LINE IS CHANGED WHICH IS
001404 016502 000032            MOV     GSBCTR(R5),R2   ;REFERENCED BY THE READ POINTER,AN FN
001410 021004           EDICHG: CMP     (R0),R4         ;POINTER OR A GOSUB POINTER THEN THAT
001412 101403                   BLOS    ,+10            ;POINTER MUST BE CLEARED
001414 021001                   CMP     (R0),R1
001416 101001                   BHI     ,+4
001420 005010                   CLR     (R0)
001422 005720                   TST     (R0)+
001424 005302                   DEC     R2
001426 003370                   BGT     EDICHG
001430 160401           EDIOVER:SUB     R4,R1
001432 000401                   BR      ,+4
001434 005001           EDIPUT: CLR     R1
001436 161601                   SUB     (SP),R1         ;R1 NOW = #CHARS TO CONTRACT,
001440 016500 000004            MOV     PDL(R5),R0      ;ALL REFERENCES BY THE READ POINTER
001444 016502 000032            MOV     GSBCTR(R5),R2   ;WHICH REFER TO LINES WHICH ARE MOVED
001450 021004           EDIMODF:CMP     (R0),R4         ;BECAUSE OF EDITING MUST BE RELOCATED
001452 103401                   BLO     ,+4
001454 160110                   SUB     R1,(R0)
001456 005720                   TST     (R0)+
001460 005302                   DEC     R2
001462 003372                   BGT     EDIMODF
001464 005701                   TST     R1
001466 100433                   BMI     EDIGROW         ;R4 NOW = ADDRESS AT WHICH TO INSERT,
001470 001473                   BEQ     EDISAME
001472 010402           EDICONT:MOV     R4,R2
001474 061602                   ADD     (SP),R2
001476 010203                   MOV     R2,R3
001500 060103                   ADD     R1,R3
001502 112322           EDIMOVE:MOVB    (R3)+,(R2)+     ;MOVE DOWN THE CODE,
001504 020315                   CMP     R3,(R5)
001506 103775                   BLO     EDIMOVE
001510 105742                   TSTB    -(R2)
001512 001401                   BEQ     ,+4
001514 005202                   INC     R2              ;R2 NOW POINTS TO BYTE AFTER THE ,EOF,
001516 030227 000001            BIT     R2,#1
001522 001401                   BEQ     ,+4
001524 105022                   CLRB    (R2)+           ;R2 NOW HAS THE NEW START OF THE SYMTAB,
001526 010215                   MOV     R2,(R5)
001530 012322                   MOV     (R3)+,(R2)+     ;MOVE DOWN THE SYMBOLS,
001532 020365 000014            CMP     R3,LOFREE(R5)
001536 103774                   BLO     ,-6
001540 010265 000014            MOV     R2,LOFREE(R5)
001544 000401                   BR      ,+4
001546 005022                   CLR     (R2)+           ;CLEAR VACATED STRING STORAGE TO ZEROES,
001550 020203                   CMP     R2,R3
001552 103775                   BLO     ,-4
001554 000441                   BR      EDISAME
001556 011500           EDIGROW:MOV     (R5),R0
```



```
001560 105740                   TSTB    -(R0)
001562 001401                   BEQ     ,+4
001564 005200                   INC     R0
001566 160100                   SUB     R1,R0           ;R0 NOW POINTS PAST THE NEW HIGHEST BYTE
001570 010003                   MOV     R0,R3           ;OF CODE,
001572 005203                   INC     R3
001574 042703 000001            BIC     #1,R3           ;R3 NOW HAS THE NEW START OF SYMTAB,
001600 161503                   SUB     (R5),R3
001602 066503 000014            ADD     LOFREE(R5),R3   ;R3 NOW HAS THE NEW VALUE OF LOFREE,
001606 020365 000012            CMP     R3,HIFREE(R5)   ;CHECK TO SEE THAT THERE IS ENOUGH
001612 101402                   BLOS    ,+6
001614 000167 016524    ERROV1: JMP     ERROV2

                                ,IFNDF  $NOSTR
                                MOV     R3,R2           ;CHECK TO SEE THAT THE SPACE THAT WILL
                                TST     (R2)+           ;BE USED IS NOT OCCUPIED BY STRINGS
                                CMP     R2,LOSTR(R5)
                                BLO     EDIROOM
                                JSR     PC,UPPACK       ;NO, MOVE THEM UP
                                CMP     R2,LOSTR(R5)    ;TRY AGAIN
                                BHIS    ERROV1
                                ,ENDC

001620 016502 000014    EDIROOM:MOV     LOFREE(R5),R2
001624 010365 000014            MOV     R3,LOFREE(R5)
001630 000401                   BR      ,+4
001632 011243           EDIMVUP:MOV     -(R2),-(R3)     ;MOVE UP THE SYMTAB,
001634 020215                   CMP     R2,(R5)
001636 101375                   BHI     EDIMVUP
001640 010315                   MOV     R3,(R5)
001642 105043                   CLRB    -(R3)           ;CLEAR THE POSSIBLY USED FILLER
001644 010002                   MOV     R0,R2
001646 060100                   ADD     R1,R0           ;R0 NOW POINTS PAST OLD HIGHEST BYTE
001650 000401                   BR      ,+4             ;OF SYMTAB,
001652 114042           MOVB:   MOVB    -(R0),-(R2)     ;MOVE UP THE CODE,
001654 020024                   CMP     R0,R4
001656 101375                   BHI     ,-4
001660 016500 000020    EDISAME:MOV     LINE(R5),R0     ;MOVE THE NEW LINE INTO THE CODE,
001664 010401                   MOV     R4,R1
001666 012602                   MOV     (SP)+,R2
001670 000401                   BR      ,+4
001672 112024                   MOVB    (R0)+,(4)+
001674 005302                   DEC     R2
001676 002375                   BGE     ,-4
001700 121127 000225    EDIRLC8:CMPB    (R1),#,EOF
001704 001417                   BEQ     EDIJMP
001706 111102                   MOVB    (R1),R2
001710 100412                   BMI     EDIRLOC
001712 005201                   INC     R1
001714 000302                   SWAB    R2
001716 152102                   BISB    (R1)+,R2
001720 061502                   ADD     (R5),R2
001722 022227 177775            CMP     (R2)+,#,SCALAR
001726 103003                   BHIS    EDIRLOC
001730 010112                   MOV     R1,(R2)
```

```
   837 001732 162712 000002                 SUB     #2,(R2)
   838 001736 004767 015216           EDIRLOC:JSR     PC,SKIPEOL
   839 001742 000756                          BR      EDIRLC8
   840 001744 000167 177210           EDIJMP: JMP     EDIT
   841
```



```
   842                                      ;-------------------------------------------------------
   843                                      ;
   844                                      ; IMMED - IMMEDIATE MODE COMMANDS
   845                                      ;
   846                                      ;  DO IMMEDIATE ONLY COMMANDS (RUN, LIST, SCRATCH, CLEAR,
   847                                      ;   SAVE, AND OLD) AND SET UP OTHERS (UN-NUMBERED) FOR
   848                                      ;   EXECUTE.
   849                                      ;
   850 001750 012600                  IMMED:  MOV     (SP)+,R0                ;R0 NOW HAS # BYTES ON INPUT LINE.
   851 001752 005065 000076                   CLR     ODEV(R5)        ;TAKE ONLY THIS IMMED. COMM. FROM NON-TTY
   852 001756 016504 000006                   MOV     POSIZE(R5),R4
   853 001762 016501 000020                   MOV     LINE(R5),R1
   854 001766 005002                          CLR     R2
   855 001770 151102                          BISB    (R1),R2
   856 001772 162702 000277                   SUB     #,LIST,R2
   857 001776 006302                          ASL     R2
   858 002000 020227 000012                   CMP     R2,#TBL4END-TABLE4
   859 002004 101012                          BHI-    IMMSTMT
   860 002006 062702 000002                   ADD     #2,R2
   861 002012 060702                          ADD     PC,R2
   862 002014 061207                          ADD     (R2),PC
   863 002016 000162                  TABLE4: ,WORD   LIST-TABLE4
   864 002020 000206                          ,WORD   RUN-TABLE4
   865 002022 000124                          ,WORD   SAVE-TABLE4
   866 002024 000062                          ,WORD   OLD-TABLE4
   867 002026 177036                          ,WORD   SCRATCH-TABLE4  ;(IN 'START' CODE, ABOVE)
   868 002030 177046                  TBL4END:,WORD   CLEAR-TABLE4    ;(IN 'START' CODE, ABOVE)
   869 002032 060100                  IMMSTMT:ADD     R1,R0
   870 002034 020065 000016                   CMP     R0,CODE(R5)     ;SEE IF AN 'EOF' CAN BE FIT ON THE LINE.
   871 002040 103402                          BLO     .+6
   872 002042 000167 015770                   JMP     ERRTRN
   873 002046 112710 000225                   MOVB    #,EOF,(R0)
   874 002052 012765 177775 000030            MOV     #,SCALAR,LINEND(R5)
   875 002060 005065 000036                   CLR     CLMNTTY(R5)     ;RE-SET ALL COLMN COUNTS
   876 002064 005067 176310                   CLR     CLMNPP
   877 002070 005067 176306                   CLR     CLMNLP
   878 002074 000167 000554                   JMP     EXECUTE         ;START RUNNING.
   879
```

```
                                        ; /OLD/ COMMAND
002100  016506  000004         OLD:    MOV     POL(R5),SP      ;FLUSH OUT SYSTEM
002104  004767  011654                 JSR     PC,INITSCR      ;SCRATCH
002110  004767  006310                 JSR     PC,CLRVARS
002114  004767  015004                 JSR     PC,SCANPND      ;SCAN OFF /#/
002120  004767  006200                 JSR     PC,CHKISET      ;CHECK AND SET INPUT
002124  122127  000201                 CMPB    (R1)+,#,EOL     ;MUST END LINE RIGHT
002130  001402                         BEQ     OLD001
002132  000167  007512                 JMP     ERRSX5
002136  000167  177016         OLD001: JMP     EDIT            ;THIS DOES THE REST

                                        ; /SAVE/ COMMAND
002142  004767  014756         SAVE:   JSR     PC,SCANPND      ;SCAN OFF /#/
002146  004767  006160                 JSR     PC,CHKOSET      ;CHECK AND SET-UP OUTPUT
002152  120227  000002                 CMPB    R2,#2           ;LINE PRINTER?
002156  003404                         BLE     LISTSV          ;NO! LIST TAKES OVER
002160  004167  013336                 JSR     R1,MSGODEV      ;FOR LPT, START W/ 2 FORM-FEEDS
002164      014      014     000        .BYTE   FF,FF,0
002170                                  .EVEN
002170  004767  012606         LISTSV: JSR     PC,LSTPROG
002174  000167  176676                 JMP     READY

                                        ; /LIST/ COMMAND
002200  005201         LIST:   INC     R1
002202  004767  011046                 JSR     PC,FLINE
002206  103770                         BCS     LISTSV
002210  011201                         MOV     (R2),R1         ;GET ADDR OF CODE LINE
002212  001766                         BEQ     LISTSV          ;IF UNDEF, NORMAL LIST
002214  004767  012566                 JSR     PC,LSTLOOP      ;LIST PARTIAL PROGR
002220  000167  176652                 JMP     READY
```



```
                                        ; /RUN/ COMMAND
002224  004767  006174         RUN:    JSR     PC,CLRVARS
002230  016501  000016                 MOV     CODE(R5),R1
002234  112102         DIMLOOP:MOVB    (R1)+,R2
002236  100406                         BMI     DIMNONO
002240  000302                         SWAB    R2
002242  152102                         BISB    (R1)+,R2
002244  061502                         ADD     (R5),R2
002246  011265  000030                 MOV     (R2),LINENO(R5)
002252  112102                         MOVB    (R1)+,R2
002254  120227  000215         DIMNONO:CMPB    R2,#,DIM
002260  001415                         BEQ     DIMGOT
002262  120227  000221                 CMPB    R2,#,DEF
002266  001530                         BEQ     DEFGOT
002270  120227  000216                 CMPB    R2,#,RANDOM
002274  001504                         BEQ     RNDGOT
002276  120227  000225                 CMPB    R2,#,EOF
002302  001537                         BEQ     DIMDONE
002304  005301                         DEC     R1
002306  004767  014646         DEFDUN: JSR     PC,SKIPEOL
002312  000750                         BR      DIMLOOP
002314  112102         DIMGOT: MOVB    (R1)+,R2
002316  100506                         BMI     ERRDIM
002320  000302                         SWAB    R2
002322  152102                         BISB    (R1)+,R2
002324  061502                         ADD     (R5),R2
002326  122127  000255                 CMPB    (R1)+,#,LPAR
002332  001100                         BNE     ERRDIM
002334  005003                         CLR     R3
002336  121127  000375                 CMPB    (R1),#,ILIT1
002342  001406                         BEQ     ALLOC1
002344  122127  000376                 CMPB    (R1)+,#,ILIT2
002350  001071                         BNE     ERRDIM
002352  111103                         MOVB    (R1),R3
002354  100467                         BMI     ERRDIM
002356  000303                         SWAB    R3
002360  005201         ALLOC1: INC     R1
002362  152103                         BISB    (R1)+,R3
002364  012704  177777                 MOV     #-1,R4
002370  122127  000237                 CMPB    (R1)+,#,RPAR
002374  001423                         BEQ     DOALLOC
002376  124127  000243                 CMPB    -(R1),#,COMMA
002402  001054                         BNE     ERRDIM
002404  005201                         INC     R1
002406  005004                         CLR     R4
002410  121127  000375                 CMPB    (R1),#,ILIT1
002414  001406                         BEQ     ALLOC2
002416  122127  000376                 CMPB    (R1)+,#,ILIT2
002422  001044                         BNE     ERRDIM
002424  111104                         MOVB    (R1),R4
002426  100442                         BMI     ERRDIM
002430  000304                         SWAB    R4
002432  005201         ALLOC2: INC     R1
002434  152104                         BISB    (R1)+,R4
002436  122127  000237                 CMPB    (R1)+,#,RPAR
```

```
  966  002442  001034                        BNE     ERRDIM
  967  002444  021227  177776          DOALLOC:CMP     (R2),#,NVAR
  968  002450  001431                        BEQ     ERRDIM
  969  002452  002403                        BLT     DIMSCLR
  970  002454  005762  000004                TST     4(R2)
  971  002460  100025                        BPL     ERRDIM
  972  002462  004767  005070          DIMSCLR:JSR     PC,ALLOC        ;VAR(R2),SS1(R3),SS2(4)
  973  002466  122127  000243                CMPB    (R1)+,#,COMMA
  974  002472  001710                        BEQ     DIMGOT
  975  002474  124127  000201                CMPB    -(R1),#,EOL
  976  002500  001015                        BNE     ERRDIM
  977  002502  005201                        INC     R1
  978  002504  000653                        BR      DIMLOOP
  979                                  ;RANDOMIZE STATEMENT
  980  002506  016565  000070  000064 RNDGOT: MOV     RNDCT(R5),RND1(R5)
  981  002514  152765  000001  000064        BISB    #1,RND1(R5)
  982  002522  122127  000201                CMPB    (R1)+,#,EOL
  983  002526  001642                        BEQ     DIMLOOP
  984  002530  000167  007114                JMP     ERRSYN
  985  002534  000004          ERRDIM: IOT
  986                                        .IFNDF  $LONGER
  987  002536  042111     115                .ASCII  'IDM'
  988                                        .ENDC
  989                                        .IFDF   $LONGER
  990                                        .ASCII  'ILLEGAL DIM'
  991                                        .ENDC
  992  002541     000                        .BYTE 0
  993                                        .EVEN
  994  002542  000004          ERRDEF: IOT
  995                                        .IFNDF  $LONGER
  996  002544  042111     106                .ASCII  \IDF\
  997                                        .ENDC
  998                                        .IFDF   $LONGER
  999                                        .ASCII  'ILLEGAL DEF'
 1000                                        .ENDC
 1001  002547     000                        .BYTE 0
 1002                                        .EVEN
 1003  002550  122127  000262          DEFGOT: CMPB    (R1)+,#,FN
 1004  002554  001372                        BNE     ERRDEF
 1005  002556  112102                        MOVB    (R1)+,R2
 1006  002560  066502  000004                ADD     POL(R5),R2
 1007  002564  005712                        TST     (R2)
 1008  002566  001365                        BNE     ERRDEF
 1009  002570  122127  000255                CMPB    (R1)+,#,LPAR
 1010  002574  001362                        BNE     ERRDEF
 1011  002576  010112                        MOV     R1,(R2)
 1012  002600  000642                        BR      DEFDUN
 1013  002602  005065  000036          DIMDONE:CLR     CLMNTTY(R5)     ;RE-SET ALL COLUMN COUNTS
 1014  002606  005067  175566                CLR     CLMNPP
 1015  002612  005067  175564                CLR     CLMNLP
 1016  002616  016504  000006                MOV     POSIZE(R5),R4
 1017  002622  016501  000016                MOV     CODE(R5),R1
 1018  002626  121127  000225                CMPB    (R1),#,EOF
 1019  002632  001010                        BNE     EXECUTE         ;START RUNNING;
 1020  002634  004167  012626          ERRRUN: JSR     R1,MSGERR
```



```
 1021                                        .IFNDF  $LONGER
 1022  002640  050116     122                .ASCII  \NPR\
 1023                                        .ENDC
 1024                                        .IFDF   $LONGER
 1025                                        .ASCII  'NO PROGRAM'
 1026                                        .ENDC
 1027  002643     000                        .BYTE 0
 1028                                        .EVEN
 1029  002644  000167  176270                JMP     READY2
 1030
```

```
                                ;*********************************************
                                ;
                                ; EXECUTE - EXECUTE COMMAND LINE
                                ;
002650  004767  014304          IGNORE: JSR     PC,SKIPEOL
002654  105765  000106          EXECUTE:TSTB    CNCFLG(R5)      ;^C' HIT?
002660  001404                          BEQ     NOCNC
002662  105065  000106                  CLRB    CNCFLG(R5)
002666  000167  176204                  JMP     READY
002672  112102          NOCNC:  MOVB    (R1)+,R2
002674  002035                          BGE     NOTSYM          ;ITS A POINTER,
002676  042702  177000                  BIC     #^C200,R2
002702  006302                          ASL     R2
002704  020227  000052                  CMP     R2,#TBL1END-TABLE1+2
002710  101127                          BHI     ERRSX1          ;NOT A STATEMENT WORD,
002712  060702                          ADD     PC,R2
002714  061207                          ADD     (R2),PC         ;BRANCH THRU TABLE1,
002716  177736          TABLE1: .WORD   EXECUTE-TABLE1
002720  000336                  .WORD   END-TABLE1
002722  001216                  .WORD   FOR-TABLE1
002724  001036                  .WORD   GOSUB-TABLE1
002726  001036                  .WORD   GOTO-TABLE1
002730  000400                  .WORD   IF-TABLE1
002732  002732                  .WORD   INPUT-TABLE1
002734  000272                  .WORD   LET-TABLE1
002736  001726                  .WORD   NEXT-TABLE1
002740  002240                  .WORD   PRINT-TABLE1
002742  001142                  .WORD   RETURN-TABLE1
002744  000360                  .WORD   STOP-TABLE1
002746  177732                  .WORD   IGNORE-TABLE1
002750  177732                  .WORD   IGNORE-TABLE1
002752  001024                  .WORD   RESTORE-TABLE1
002754  177732                  .WORD   IGNORE-TABLE1
002756  177732                  .WORD   IGNORE-TABLE1
002760  003330                  .WORD   READ-TABLE1
002762  177732                  .WORD   IGNORE-TABLE1
002764  000074                  .WORD   CALL-TABLE1
002766  000336          TBL1END:.WORD   EOF-TABLE1
002770  000302          NOTSYM: SWAB    R2
002772  152102                  BISB    (R1)+,R2
002774  061502                  ADD     (R5),R2
002776  021227  177775                  CMP     (R2),#,SCALAR
003002  103107                          BHIS    ASSIGN          ;NOT A LINENO,
003004  011265  000030                  MOV     (R2),LINENO(R5) ;SAVE THE LINENO,
003010  000721                  BR      EXECUTE
```



```
                                ; 'CALL' STATEMENT
                                ;       CALL "PROG" (,,,)
                                CALL:
                                
                                        .IFNDF  $NOSTK
                                        JSR     PC,EVAL         ;EVALUATE FUNCTION NAME
                                        BCC     ERRSX1
                                        MOV     (SP)+,R0
                                        CMP     R0,#177777      ;CHECK NULL STRING
                                        BEQ     ERRFN1          ;YES, ERROR
                                        MOVB    (R0)+,R3        ;R3 IS STRING LENGTH
                                        MOVB    (R0)+,(R0)+     ;R0 IS ADDR OF CHARS
                                        .ENDC
                                
                                        .IFDF   $NOSTK
003012  112102                  MOVB    (R1)+,R2        ;R2 IS QUOTE CHAR
003014  120227  000257                  CMPB    R2,#'QUOT
003020  001403                          BEQ     CALL1
003022  120227  000256                  CMPB    R2,#"QUOT
003026  001060                          BNE     ERRSX1
003030  122127  000377          CALL1:  CMPB    (R1)+,#',TEXT
003034  001055                          BNE     ERRSX1
003036  010100                  MOV     R1,R0           ;R0 IS ADDR OF CHARS
003040  005003                  CLR     R3
003042  005203          CALLP:  INC     R3              ;COUNT CHARS IN R3
003044  105721                  TSTB    (R1)+
003046  001375                  BNE     CALLP
003050  122102                  CMPB    (R1)+,R2        ;CHECK CLOSING QUOTE
003052  001046                  BNE     ERRSX1
003054  005303                  DEC     R3
003056  001442                  BEQ     ERRFN1
                                        .ENDC
                                
                                ;       INIT TABLE SEARCH
003060  013702  000046                  MOV     @#46,R2
003064  001437                          BEQ     ERRFN1          ;NO FUNCTIONS DEFINED
003066  010246                  MOV     R2,-(SP)        ;SAVE ON STACK
                                ;       CHECK END TABLE
003070  105712          CALLCK: TSTB    (R2)
003072  001434                  BEQ     ERRFN1
                                ;       CHECK THAT (R3) CHARS MATCH
003074  010046                  MOV     R0,-(SP)
003076  010346                  MOV     R3,-(SP)
003100  122022          CALLM1: CMPB    (R0)+,(R2)+
003102  001034                  BNE     CALLNM
003104  005303                  DEC     R3
003106  001374                  BNE     CALLM1
                                ;       CHECK THAT 4-(R3) CHARS ARE 0 IN TABLE
003110  161603                  SUB     (SP),R3
003112  062703  000004                  ADD     #4,R3
003116  002422                  BLT     ERRFN1
003120  001404                  BEQ     CALLX
003122  105722          CALLM2: TSTB    (R2)+
003124  001023                  BNE     CALLNM
003126  005303                  DEC     R3
```

```
1133  003130  001374                        BNE     CALLM2
1134                                  ;       RETURN ANSWER
1135  003132  062706  000006          CALLX:  ADD     #6,SP           ;POP STACK
1136  003136  011202                        MOV     (R2),R2
1137  003140  001411                        BEQ     ERRFN1
1138  003142  010146                        MOV     R1,-(SP)
1139  003144  010446                        MOV     R4,-(SP)
1140  003146  010546                        MOV     R5,-(SP)
1141
1142  003150  004712                        JSR     PC,(R2)
1143
1144  003152  012605                        MOV     (SP)+,R5
1145  003154  012604                        MOV     (SP)+,R4
1146  003156  012601                        MOV     (SP)+,R1
1147  003160  000167  177464                JMP     IGNORE
1148  003164  000167  007026          ERRFN1: JMP     ERRUFN
1149  003170  000167  006454          ERRSX1: JMP     ERRSYN
1150                                  ;       ADVANCE TO NEXT TABLE ENTRY
1151  003174  012603                  CALLNM: MOV     (SP)+,R3
1152  003176  012600                        MOV     (SP)+,R0
1153  003200  062716  000006                ADD     #6,(SP)
1154  003204  011602                        MOV     (SP),R2
1155  003206  000730                        BR      CALLCK
1156
1157                                  ; 'LET' STATEMENT ROUTINE
1158  003210  112102                  LET:    MOVB    (R1)+,R2
1159  003212  100766                        BMI     ERRSX1          ;NOT A POINTER,
1160  003214  000302                        SWAB    R2
1161  003216  152102                        BISB    (R1)+,R2
1162  003220  061502                        ADD     (R5),R2
1163  003222  004767  010356          ASSIGN: JSR     PC,GETVAR
1164  003226  122127  000254                CMPB    (R1)+,#'=,EQ
1165  003232  001356                        BNE     ERRSX1          ;NOT EQSIGN,
1166  003234  004767  005356                JSR     PC,EVAL
1167
1168                                          .IFNDF  $NOSTR
1169                                          BCS     ASSSTR
1170                                          .ENDC
1171
1172  003240  122127  000201                CMPB    (R1)+,#',EOL
1173  003244  001351                        BNE     ERRSX1
1174  003246  004767  013772                JSR     PC,STOVAR
1175  003252  000600                        BR      EXECUTE
1176
1177                                          .IFNDF  $NOSTR
1178                                  ASSSTR: CMPB    (R1)+,#',EOL
1179                                          BNE     ERRSX1
1180                                          JSR     PC,STOSVAR
1181                                          BR      EXECUTE
1182                                          .ENDC
1183
1184                                  ; 'EOF' OR 'END' STATEMENT
1185                                  END:
1186  003254  020165  000016          EOF:    CMP     R1,CODE(R5)
1187  003260  103014                        BHIS    JMPRDY
```



```
1188  003262  004167  012206                JSR     R1,MSG
1189  003266     015     012                ,BYTE   CR,LF
1190  003270     000                        ,BYTE   0
1191          003272                        ,EVEN
1192  003272  000167  175662                JMP     EDIT
1193                                  ; 'STOP' STATEMENT
1194  003276  004167  012172          STOP:   JSR     R1,MSG
1195  003302     015     012                ,BYTE   CR,LF
1196  003304  052123  050117                ,ASCII  'STOP!
1197  003310     000                        ,BYTE   0
1198          003312                        ,EVEN
1199  003312  000167  175560          JMPRDY: JMP     READY
1200
```

```
 1311  003712  001060                          BNE     ERRSX0
 1312  003714  120027  000246                  CMPB    R0,#,GE
 1313  003720  001415                          BEQ     GOTO
 1314  003722  120027  000253                  CMPB    R0,#,GT
 1315  003726  001412                          BEQ     GOTO
 1316  003730  120027  000250                  CMPB    R0,#,NE
 1317  003734  001407                          BEQ     GOTO
 1318  003736  000167  176706                  JMP     IGNORE
 1319  003742  012775  177777  000004  RESTORE:MOV     #-1,@POL(R5)
 1320  003750  000167  176700                  JMP     EXECUTE
 1321
```



```
 1322                                  ; 'GOSUB' AND 'GOTO' STATEMENTS
 1323                                  GOSUB:
 1324  003754  004767  007274          GOTO:   JSR     PC,FLINE
 1325  003760  103435                          BCS     ERRSX0
 1326  003762  122127  000201                  CMPB    (R1)+,#,EOL
 1327  003766  001032                          BNE     ERRSX0
 1328  003770  005712                          TST     (R2)
 1329  003772  001425                          BEQ     ERRGO
 1330  003774  126127  177774  000204          CMPB    -4(R1),#,GOSUB
 1331  004002  001013                          BNE     GONOSAV
 1332  004004  016503  000032                  MOV     GSBCTR(R5),R3
 1333  004010  006303                          ASL     R3
 1334  004012  066503  000004                  ADD     POL(R5),R3
 1335  004016  020365  000002                  CMP     R3,LIMIT(R5)
 1336  004022  101006                          BHI     ERRDEEP
 1337  004024  010113                          MOV     R1,(R3)
 1338  004026  005265  000032                  INC     GSBCTR(R5)
 1339  004032  011201                  GONOSAV:MOV     (R2),R1
 1340  004034  000167  176614                  JMP     EXECUTE
 1341  004040  000004                  ERRDEEP:IOT
 1342                                          .IFNDF  $LONGER
 1343  004042  047107      104                 .ASCII  \GND\
 1344                                          .ENDC
 1345                                          .IFDF   $LONGER
 1346                                          .ASCII  'GOSUBS NESTED TOO DEEPLY'
 1347                                          .ENDC
 1348  004045      000                         .BYTE 0
 1349                                          .EVEN
 1350  004046  000004                  ERRGO:  IOT
 1351                                          .IFNDF  $LONGER
 1352  004050  046125      116                 .ASCII  \ULN\
 1353                                          .ENDC
 1354                                          .IFDF   $LONGER
 1355                                          .ASCII  'UNDEFINED LINE NUMBER'
 1356                                          .ENDC
 1357  004053      000                         .BYTE 0
 1358                                          .EVEN
 1359  004054  000167  005570          ERRSX6: JMP     ERRSX5
 1360
```

```
 1201                                        ; 'IF' STATEMENT ROUTINE
 1202  003316  004767  005274          IF:    JSR     PC,EVAL
 1203  003322  103113                         BCC     IFNUMER
 1204  003324  121127  000244                 CMPB    (R1)+,#LE
 1205  003330  103717                         BLO     ERRSX1
 1206  003332  121127  000254                 CMPB    (R1)+,#EQ
 1207  003336  101314                         BHI     ERRSX1
 1208  003340  020604                         CMP     SP,R4
 1209  003342  103551                         BLO     ERRPD3
 1210  003344  112146                         MOVB    (R1)+,-(SP)
 1211  003346  004767  005244                 JSR     PC,EVAL
 1212
 1213                                         .IFNDF  $NOSTM
 1214                                         BCC     ERRMX4
 1215                                         .ENDC
 1216
 1217  003352  012603                         MOV     (SP)+,R3
 1218  003354  012600                         MOV     (SP)+,R0
 1219  003356  012602                         MOV     (SP)+,R2
 1220  003360  010046                         MOV     R0,-(SP)
 1221  003362  005046                         CLR     -(SP)
 1222  003364  020227  177777                 CMP     R2,#-1
 1223  003370  001401                         BEQ     .+4
 1224  003372  151216                         BISB    (R2),(SP)
 1225  003374  005000                         CLR     R0
 1226  003376  020327  177777                 CMP     R3,#-1
 1227  003402  001401                         BEQ     .+4
 1228  003404  151300                         BISB    (R3),R0
 1229  003406  062702  000003                 ADD     #3,R2
 1230  003412  062703  000003                 ADD     #3,R3
 1231  003416  000402                         BR      .+6
 1232  003420  122322          IFLOOP: CMPB    (R3)+,(R2)+
 1233  003422  001047                         BNE     IFCOMP
 1234  003424  005300                         DEC     R0
 1235  003426  002410                         BLT     IFLEND
 1236  003430  005316                         DEC     (SP)
 1237  003432  002372                         BGE     IFLOOP
 1238  003434  122327  000040                 CMPB    (R3)+,#BL
 1239  003440  001040                         BNE     IFCOMP
 1240  003442  005300                         DEC     R0
 1241  003444  002373                         BGE     .-10
 1242  003446  000407                         BR      IFSEQ
 1243  003450  005716          IFLEND: TST     (SP)
 1244  003452  003405                         BLE     IFSEQ
 1245  003454  122722  000040                 CMPB    #BL,(R2)+
 1246  003460  001030                         BNE     IFCOMP
 1247  003462  005316                         DEC     (SP)
 1248  003464  003373                         BGT     .-10
 1249  003466  005726          IFSEQ:  TST     (SP)+
 1250  003470  012600                         MOV     (SP)+,R0
 1251  003472  112102          IFLEQR: MOVB    (R1)+,R2
 1252  003474  120227  000242                 CMPB    R2,#,THEN
 1253  003500  001403                         BEQ     .+10
 1254  003502  120227  000205                 CMPB    R2,#,GOTO
 1255  003506  001013                         BNE     ERRSX7
```



```
 1256  003510  120027  000244                 CMPB    R0,#,LE
 1257  003514  001517                         BEQ     GOTO
 1258  003516  120027  000246                 CMPB    R0,#,GE
 1259  003522  001514                         BEQ     GOTO
 1260  003524  120027  000254                 CMPB    R0,#,EQ
 1261  003530  001511                         BEQ     GOTO
 1262  003532  000167  177112                 JMP     IGNORE
 1263  003536  000167  006106          ERRSX7: JMP     ERRSX
 1264  003542  002453          IFCOMP: BLT     IFSGT
 1265  003544  005726                         TST     (SP)+
 1266  003546  012600                         MOV     (SP)+,R0
 1267  003550  000424                         BR      IFLLTR
 1268  003552  121127  000244          IFNUMER:CMPB    (R1),#,LE
 1269  003556  103767                         BLO     ERRSX7
 1270  003560  121127  000254                 CMPB    (R1),#,EQ
 1271  003564  101364                         BHI     ERRSX7
 1272  003566  020604                         CMP     SP,R4
 1273  003570  103436                         BLO     ERRPD5
 1274  003572  016546  000042                 MOV     FAC2(R5),-(SP)
 1275  003576  016546  000040                 MOV     FAC1(R5),-(SP)
 1276  003602  112146                         MOVB    (R1)+,-(SP)
 1277  003604  004767  005006                 JSR     PC,EVAL
 1278
 1279                                         .IFNDF  $NOSTM
 1280                                         BCS     ERRMX4
 1281                                         .ENDC
 1282
 1283  003610  012600                         MOV     (SP)+,R0
 1284  003612  004767  013474                 JSR     PC,SUBSTM
 1285  003616  001725                         BEQ     IFLEQM
 1286  003620  002426                         BLT     IFLGTM
 1287  003622  112102          IFLLTR: MOVB    (R1)+,R2
 1288  003624  120227  000242                 CMPB    R2,#,THEN
 1289  003630  001403                         BEQ     .+10
 1290  003632  120227  000205                 CMPB    R2,#,GOTO
 1291  003636  001126                         BNE     ERRSX6
 1292  003640  120027  000244                 CMPB    R0,#,LE
 1293  003644  001443                         BEQ     GOTO
 1294  003646  120027  000252                 CMPB    R0,#,LT
 1295  003652  001440                         BEQ     GOTO
 1296  003654  120027  000250                 CMPB    R0,#,NE
 1297  003660  001435                         BEQ     GOTO
 1298  003662  000167  176762                 JMP     IGNORE
 1299  003666  000167  005226          ERRPD3: JMP     ERRPDL
 1300
 1301                                         .IFNDF  $NOSTM
 1302                                ERRMX4:  JMP     ERRMIX
 1303                                         .ENDC
 1304
 1305  003672  005726                 IFSGT:  TST     (SP)+
 1306  003674  012600                         MOV     (SP)+,R0
 1307  003676  112102                 IFLGTR: MOVB    (R1)+,R2
 1308  003700  120227  000242                 CMPB    R2,#,THEN
 1309  003704  001403                         BEQ     .+10
 1310  003706  120227  000205                 CMPB    R2,#,GOTO
```

```
                                        ; /RETURN/ STATEMENT
004060 122127 000201            RETURN: CMPB    (R1)+,#,EOL
004064 001373                           BNE     ERRSX0
004066 016503 000032                    MOV     GSBCTR(R5),R3
004072 020327 000033                    CMP     R3,#33
004076 001413                           BEQ     ERRRET
004100 005303                           DEC     R3
004102 010365 000032                    MOV     R3,GSBCTR(R5)
004106 006303                           ASL     R3
004110 066503 000004                    ADD     POL(R5),R3
004114 005713                           TST     (R3)
004116 001403                           BEQ     ERRRET
004120 011301                           MOV     (R3),R1
004122 000167 176526                    JMP     EXECUTE

                                        .IFNDF  $NOSTK
                                ERRMX5: JMP     ERRMIX
                                        .ENDC

004126 000004                   ERRRET: IOT
                                        .IFNDF  $LONGER
004130 041122     107                   .ASCII  \NBG\
                                        .ENDC
                                        .IFDF   $LONGER
                                        .ASCII  /RETURN BEFORE GOSUB/
                                        .ENDC
004133     000                          .BYTE 0
                                        .EVEN

                                        ; /FOR/ STATEMENT
004134 010146                   FOR:    MOV     R1,-(SP)
004136 005316                           DEC     (SP)            ;(LOC OF THE 'FOR')
004140 112102                           MOVB    (R1)+,R2
004142 100744                           BMI     ERRSX0
004144 000302                           SWAB    R2
004146 152102                           BISB    (R1)+,R2
004150 061502                           ADD     (R5),R2
004152 021227 177777                    CMP     (R2),#,SVAR
004156 001736                           BEQ     ERRSX0
004160 122127 000254                    CMPB    (R1)+,#,EQ
004164 001333                           BNE     ERRSX0
004166 010265 000022                    MOV     R2,VARSAV(R5)   ;(VAR OF THE FOR)
004172 004767 004420                    JSR     PC,EVAL

                                        .IFNDF  $NOSTK
                                        BCS     ERRMX5
                                        .ENDC

004176 012765 177777 000024             MOV     #-1,SS1SAV(R5)  ;FORCE TO SCALAR,
004204 004767 013034                    JSR     PC,STOVAR
004210 122127 000240                    CMPB    (R1)+,#,TO
004214 001317                           BNE     ERRSX0
004216 004767 004374                    JSR     PC,EVAL

                                        .IFNDF  $NOSTK
```



```
                                        BCS     ERRMX5
                                        .ENDC

004222 016565 000040 000024             MOV     FAC1(R5),SS1SAV(R5)
004230 016565 000042 000026             MOV     FAC2(R5),SS2SAV(R5);(LIMIT OF THE FOR)
004236 005065 000040                    CLR     FAC1(R5)
004242 012765 000001 000042             MOV     #1,FAC2(R5)     ;ASSUMED STEP IS 1.0
004250 121127 000201                    CMPB    (R1),#,EOL
004254 001410                           BEQ     FORONE
004256 122127 000241                    CMPB    (R1)+,#,STEP
004262 001274                           BNE     ERRSX0
004264 004767 004326                    JSR     PC,EVAL         ;(FAC HAS THE STEP)

                                        .IFNDF  $NOSTK
                                        BCS     ERRMX5
                                        .ENDC

004270 121127 000201                    CMPB    (R1),#,EOL
004274 001267                           BNE     ERRSX0
004276 005201                   FORONE: INC     R1
004300 016503 000030                    MOV     LINENO(R5),R3
004304 000402                           BR      FORLOOK
004306 004767 012646            FORSKIP:JSR     PC,SKIPEOL
004312 111102                   FORLOOK:MOVB    (R1),R2
004314 100410                           BMI     FORNOL
004316 005201                           INC     R1
004320 000302                           SWAB    R2
004322 152102                           BISB    (R1)+,R2
004324 061502                           ADD     (R5),R2
004326 021227 177775                    CMP     (R2),#,SCALAR
004332 103001                           BHIS    FORNOL
004334 011203                           MOV     (R2),R3         ;(THE LINENO WHILE LOOKING)
004336 121127 000225            FORNOL: CMPB    (R1),#,EOF
004342 001420                           BEQ     ERR4WO
004344 121127 000211                    CMPB    (R1),#,NEXT     ;THE CODE IS NOW BEING SEARCHED
004350 001420                           BEQ     FORNEXT         ;FOREWARD FROM THE 'FOR' FOR A NEXT WITH
004352 121127 000203                    CMPB    (R1),#,FOR      ;THE SAME VAR, IF 'EOF' OR ANOTHER 'FOR'
004356 001353                           BNE     FORSKIP         ;WITH THE SAME VAR IS FOUND FIRST|ERROR.
004360 005201                           INC     R1
004362 111102                           MOVB    (R1),R2
004364 100750                           BMI     FORSKIP
004366 005201                           INC     R1
004370 000302                           SWAB    R2
004372 152102                           BISB    (R1)+,R2
004374 061502                           ADD     (R5),R2
004376 020265 000022                    CMP     R2,VARSAV(R5)
004402 001341                           BNE     FORSKIP
004404 000004                   ERR4WO: IOT
                                        .IFNDF  $LONGER
004406 053506     116                   .ASCII  /FWN/
                                        .ENDC
                                        .IFDF   $LONGER
                                        .ASCII  /FOR WITHOUT NEXT/
                                        .ENDC
004411     000                          .BYTE 0
```

```
1471                                    .EVEN
1472  004412  062701  000013        FORNEXT:ADD     #13,R1
1473  004416  111132                        MOVB    (R1),R2
1474  004420  100732                        BMI     FORSKIP
1475  004422  005201                        INC     R1
1476  004424  000302                        SWAB    R2
1477  004426  152102                        BISB    (R1)+,R2
1478  004430  061502                        ADD     (R5),R2
1479  004432  020265  000022                CMP     R2,VARSAV(R5)
1480  004436  001323                        BNE     FORSKIP
1481  004440  121127  000201                CMPB    (R1),#,EOL
1482  004444  001050                        BNE     ERSX10
1483  004446  162701  000014                SUB     #14,R1
1484  004452  116621  000001                MOVB    1(SP),(R1)+     ;PUT ADDR OF THE 'FOR' AFTER THE 'NEXT',
1485  004456  111621                        MOVB    (SP),(R1)+      ;(THATS WHAT THE 2+4+4 BYTES ARE FOR)
1486  004460  116521  000025                MOVB    SS1SAV+1(R5),(R1)+;PUT THE LIMIT AFTER THAT,
1487  004464  116521  000024                MOVB    SS1SAV(R5),(R1)+
1488  004470  116521  000027                MOVB    SS2SAV+1(R5),(R1)+
1489  004474  116521  000026                MOVB    SS2SAV(R5),(R1)+
1490  004500  116521  000041                MOVB    FAC1+1(R5),(R1)+        ;PUT THE STEP AFTER THAT,
1491  004504  116521  000040                MOVB    FAC1(R5),(R1)+
1492  004510  116521  000043                MOVB    FAC2+1(R5),(R1)+
1493  004514  116521  000042                MOVB    FAC2(R5),(R1)+
1494  004520  010316                        MOV     R3,(SP)         ;PUT THE SEARCH LINENO ON THE STACK,
1495  004522  016546  000040                MOV     FAC1(R5),-(SP)  ;PUT SGN(STEP) ON THE STACK,
1496  004526  001002                        BNE     .+6
1497  004530  016516  000042                MOV     FAC2(R5),(SP)
1498  004534  016546  000026                MOV     SS2SAV(R5),-(SP)        ;PUT THE LIMIT ON THE STACK,
1499  004540  016546  000024                MOV     SS1SAV(R5),-(SP)
1500  004544  004767  004046                JSR     PC,EVAL         ;GET THE CURRENT VALUE OF THE INDEX,
1501  004550  004767  012536                JSR     PC,SUBSTK       ;SUBTRACT THE LIMIT,
1502  004554  001421                        BEQ     FORGOGO
1503  004556  100405                        BMI     FORLTLM
1504  004560  005726              FORGTLMI: TST     (SP)+
1505  004562  100417                        BMI     FORGO
1506  004564  000404                        BR      FOREERO
1507  004566  000167  005056        ERSX10: JMP     ERRSX5
1508  004572  005726              FORLTLMI: TST     (SP)+
1509  004574  100012                        BPL     FORGO
1510  004576  012665  000030        FOREERO:MOV     (SP)+,LINENO(R5)
1511  004602  105061  177764                CLRB    -14(R1)
1512  004606  105061  177765                CLRB    -13(R1)
1513  004612  005201                        INC     R1
1514  004614  000167  176034                JMP     EXECUTE
1515  004620  005726              FORGOGO:TST     (SP)+
1516  004622  005726              FORGO:  TST     (SP)+
1517  004624  116146  177765                MOVB    -13(R1),-(SP)
1518  004630  116166  177764  000001        MOVB    -14(R1),1(SP)
1519  004636  012601                        MOV     (SP)+,R1
1520  004640  000167  176004                JMP     IGNORE
1521                                    .EOT
1522
```



```
1523                                    ;                               SOURCE FILE #3
1524                                    ; 'NEXT' STATEMENT
1525  004644  005002                NEXT:   CLR     R2
1526  004646  152102                        BISB    (R1)+,R2
1527  004650  000302                        SWAB    R2
1528  004652  152102                        BISB    (R1)+,R2
1529  004654  020265  000014                CMP     R2,LOFREE(R5)   ;JUST TO PREVENT AN NXM TRAP,
1530  004660  103131                        BHIS    ERRNEXT
1531  004662  122227  000203                CMPB    (R2)+,#,FOR
1532  004666  001126                        BNE     ERRNEXT
1533  004670  062701  000010                ADD     #10,R1
1534  004674  112103                        MOVB    (R1)+,R3
1535  004676  100525                        BMI     ERRSX2
1536  004700  000303                        SWAB    R3
1537  004702  151103                        BISB    (R1),R3
1538  004704  061503                        ADD     (R5),R3
1539  004706  005301                        DEC     R1
1540  004710  122122                        CMPB    (R1)+,(R2)+
1541  004712  001114                        BNE     ERRNEXT
1542  004714  121122                        CMPB    (R1),(R2)+
1543  004716  001112                        BNE     ERRNEXT
1544  004720  121227  000254                CMPB    (R2),#,EQ
1545  004724  001107                        BNE     ERRNEXT
1546  004726  010365  000022                MOV     R3,VARSAV(R5)   ;SAVE VARIABLE ADDRESS,
1547  004732  021327  177777                CMP     (R3),#,SVAR
1548  004736  001505                        BEQ     ERRSX2
1549  004740  022327  177775                CMP     (R3)+,#,SCALAR
1550  004744  001401                        BEQ     .+4
1551  004746  111303                        MOV     (R3),R3
1552  004750  012365  000040                MOV     (R3)+,FAC1(R5)  ;PUT THE FOR VAR VALUE INTO FAC,
1553  004754  011365  000042                MOV     (R3),FAC2(R5)
1554  004760  005301                        DEC     R1
1555  004762  114146                        MOVB    -(R1),-(SP)     ;PUT STEP ON THE STACK,
1556  004764  114166  000001                MOVB    -(R1),1(SP)
1557  004770  114146                        MOVB    -(R1),-(SP)
1558  004772  114166  000001                MOVB    -(R1),1(SP)
1559  004776  011665  000026                MOV     (SP),SS2SAV(R5) ;SAVE SGN(STEP),
1560  005002  001003                        BNE     .+10
1561  005004  016665  000002  000026        MOV     2(SP),SS2SAV(R5)
1562  005012  004767  002352                JSR     PC,ADDSTK
1563  005016  016546  000042                MOV     FAC2(R5),-(SP)
1564  005022  016546  000040                MOV     FAC1(R5),-(SP)
1565  005026  114146                        MOVB    -(R1),-(SP)
1566  005030  114166  000001                MOVB    -(R1),1(SP)
1567  005034  114146                        MOVB    -(R1),-(SP)
1568  005036  114166  000001                MOVB    -(R1),1(SP)
1569  005042  004767  012244                JSR     PC,SUBSTK
1570  005046  001417                        BEQ     NEXGO
1571  005050  100404                        BMI     NEXLTLM
1572  005052  005765  000026        NEXGTLMI:TST    SS2SAV(R5)
1573  005056  100413                        BMI     NEXGO
1574  005060  000403                        BR      NEXEND
1575  005062  005765  000026        NEXLTLMI:TST    SS2SAV(R5)
1576  005066  100007                        BPL     NEXGO
1577  005070  105041                NEXEND: CLRB    -(R1)
```

```
 1578  005072  105041                          CLRB    -(R1)
 1579  005074  062701  000015                  ADD     #15,R1
 1580  005100  022620                          CMP     (SP)+,(SP)+     ;THROW AWAY INCREMENTED INDEX,
 1581  005102  000167  175946                  JMP     EXECUTE
 1582  005106  114146                  NEXGO:  MOVB    -(R1),-(SP)
 1583  005110  114166  000001                  MOVB    -(R1),1(SP)
 1584  005114  012601                          MOV     (SP)+,R1
 1585  005116  012665  000040                  MOV     (SP)+,FAC1(R5)  ;STORE THE INCREMENTED INDEX,
 1586  005122  012665  000042                  MOV     (SP)+,FAC2(R5)
 1587  005126  012765  177777  000024          MOV     #-1,SS1SAV(R5)
 1588  005134  004767  012104                  JSR     PC,STOVAR
 1589  005140  000167  175504                  JMP     IGNORE
 1590  005144  000004                  ERRNEXT:: IOT
 1591                                          ,IFNDF  $LONGER
 1592  005146  041116     100                  ,ASCII  \NBF\
 1593                                          ,ENDC
 1594                                          ,IFDF   $LONGER
 1595                                          ,ASCII  /NEXT BEFORE FOR/
 1596                                          ,ENDC
 1597  005151     000                          ,BYTE 0
 1598                                          ,EVEN
 1599  005152  000167  004472          ERRSX2: JMP     ERRSX3
 1600
```



```
 1601                                  ; /PRINT/ STATEMENT
 1602  005156  121127  000201          PRINT:: CMPB    (R1)+,#,POUND   ;IS IT /PRINT #N[ ,../?
 1603  005162  001016                          BNE     PRNT01
 1604  005164  005201                          INC     R1
 1605  005166  004767  003140                  JSR     PC,CHKOSET      ;CHECK CHANNEL AND SET-UP I/O
 1606  005172  005702                          TST     R2              ;DEVICE CODE
 1607  005174  001403                          BEQ     COLONCH
 1608  005176  016265  005320  000034          MOV     TABL5(R2),COLUMN(R5)    ;NON-TTY COLUMN COUNT
 1609  005204  121127  000201          COLONCH:CMPB    (R1)+,#,EOL     ;DON'T NEED COLON IF NOTHING
 1610  005210  001403                          BEQ     PRNT01          ;  FOLLOWS NUMBER
 1611  005212  122127  000260                  CMPB    (R1)+,#,COLON
 1612  005216  001355                          BNE     ERRSX2
 1613  005220  121127  000243          PRNT01: CMPB    (R1)+,#,COMMA
 1614  005224  001440                          BEQ     PRICM
 1615  005226  121127  000236                  CMPB    (R1)+,#,SEMI
 1616  005232  001440                          BEQ     PRIBOTH
 1617  005234  121127  000201                  CMPB    (R1)+,#,EOL
 1618  005240  001435                          BEQ     PRIBOTH
 1619
 1620                                          ,IFDF   $NOSTR
 1621  005242  121127  000257                  CMPB    (R1)+,#,SQUOT   ;CHECK PRINT STRING
 1622  005246  001513                          BEQ     PRISTR
 1623  005250  121127  000256                  CMPB    (R1)+,#,DQUOT
 1624  005254  001510                          BEQ     PRISTR
 1625  005256  121127  000276                  CMPB    (R1)+,#,TAB     ;OR TAB FUNCTION
 1626  005262  001532                          BEQ     PRITB
 1627                                          ,ENDC
 1628
 1629  005264  004767  003326                  JSR     PC,EVAL
 1630
 1631                                          ,IFNDF  $NOSTR
 1632                                          BCS     PRISTR
 1633                                          ,ENDC
 1634
 1635  005270  027527  000034  000074          CMP     @COLUMN(R5),#74
 1636  005276  101402                          BLOS    .+6
 1637  005300  004767  000326                  JSR     PC,PRINCR
 1638  005304  004767  010450                  JSR     PC,NUMSGN
 1639  005310  016676                          ,WORD   PUTCHAR
 1640  005312  004167  010204                  JSR     R1,MSGODEV
 1641  005316     040                          ,ASCII  / /
 1642  005317     000                          ,BYTE 0
 1643                                          ,EVEN
 1644  005320  000405                          BR      PRIBOTH
 1645
 1646          005320                  TABL5   .,.+2
 1647  005322  000400                          +       CLMNPP
 1648  005324  000402                          +       CLMNLP
 1649
 1650
 1651
 1652                                          ,IFNDF  $NOSTR
 1653                                  PRISTR: MOV     (SP)+,R2
 1654                                          INC     R2
 1655                                          BEQ     PRIBOTH
```

```
                                        CLR     R3
                                        BISB    -(R2),R3
                                        ADD     #3,R2
                                PRILOOP:CMP     @COLUMN(R5),#110
                                        BLO     .+6
                                        JSR     PC,PRINCR
                                        MOVB    (R2)+,R0
                                        JSR     PC,PUTCHAR
                                        DEC     R3
                                        BGT     PRILOOP
                                        BR      PRIBOTH
                                        .ENDC

005326  004167  010170          PRICMI: JSR     R1,MSGODEV
005332     040     000                  .BYTE   BL,0
005334  121127  000201          PRIBOTH:CMPB    (R1),#,EOL
005340  001003                          BNE     PRIMORE
005342  004767  000264                  JSR     PC,PRINCR
005346  000441                          BR      PRIJMP
005350  122127  000243          PRIMORE:CMPB    (R1)+,#,COMMA
005354  001027                          BNE     PRISEMI
005356  317500  000034          PRICOMMI:MOV    @COLUMN(R5),R0
005362  001430                          BEQ     PRITEST
005364  020027  000070                  CMP     R0,#70
005370  001425                          BEQ     PRITEST
005372  103403                          BLO     .+10
005374  004767  000232                  JSR     PC,PRINCR
005400  000421                          BR      PRITEST
005402  020027  000052                  CMP     R0,#52
005406  001416                          BEQ     PRITEST
005410  020027  000034                  CMP     R0,#34
005414  001413                          BEQ     PRITEST
005416  020027  000016                  CMP     R0,#16
005422  001410                          BEQ     PRITEST
005424  004167  010072                  JSR     R1,MSGODEV
005430     040                          .ASCII  ' '
005431     000                          .BYTE 0
                                        .EVEN
005432  000751                          BR      PRICOMM
005434  124127  000236          PRISEMI:CMPB    -(R1),#,SEMI
005440  001267                          BNE     PRNT01
005442  005201                          INC     R1
005444  121127  000201          PRITEST:CMPB    (R1),#,EOL
005450  001263                          BNE     PRNT01
005452  005201                  PRIJMP: INC     R1
005454  012765  000036  000034          MOV     #CLMNTTY,COLUMN(R5)     ;RE-SET TO TTY COLUMN COUNT
005462  060565  000034                  ADD     R5,COLUMN(R5)
005466  105065  000076                  CLRB    ODEV(R5)
005472  000167  175156                  JMP     EXECUTE

                                        .IFDF   $NOSTR
005476  112103                  PRISTR: MOVB    (R1)+,R3        ;SAVE OPEN QUOTE CHAR
005500  122127  000377                  CMPB    (R1)+,#,TEXT    ;TEST FOR TEXT BYTE
005504  001222                  PRISJ:  BNE     ERRSX2
005506  112102                  PRIST1: MOVB    (R1)+,R2        ;GET NEXT CHAR
```



```
005510  001776                          BEQ     PRIST1
005512  120203                          CMPB    R2,R3           ;CHECK CLOSE QUOTE
005514  001707                          BEQ     PRIBOTH
005516  120227  000201                  CMPB    R2,#,EOL        ;CHECK END LINE
005522  001613                          BEQ     ERRSX2
005524  027527  000034  000110          CMP     @COLUMN(R5),#110
005532  103402                          BLO     .+6
005534  004767  000072                  JSR     PC,PRINCR
005540  010200                          MOV     R2,R0
005542  004767  011130                  JSR     PC,PUTCHAR
005546  000757                          BR      PRIST1

005550  105721                  PRITBI: TSTB    (R1)+
005552  004767  003040                  JSR     PC,EVAL
005556  004767  006222                  JSR     PC,INT
005562  122127  000237                  CMPB    (R1)+,#,RPAR
005566  001346                          BNE     PRISJ
005570  116502  000042                  MOVB    FAC2(R5),R2     ;GET FN VALUE
005574  167502  000034                  SUB     @COLUMN(R5),R2  ;COMPUTE # SPACES
005600  220227  000110          PRITB0: CMP     R2,#72.
005604  103403                          BLO     PRITB1
005606  162702  000110                  SUB     #72.,R2
005612  000772                          BR      PRITB0
005614  005302                  PRITB1: DEC     R2
005616  100646                          BMI     PRIBOTH
005620  012708  000040                  MOV     #BL,R0
005624  004767  011046                  JSR     PC,PUTCHAR
005630  000771                          BR      PRITB1
                                        .ENDC
                                )
005632  004167  007664          PRINCR: JSR     R1,MSGODEV
005636     015     012                  .BYTE   CR,LF
005640     000                          .BYTE 0
                005642                  .EVEN
005642  005075  000034                  CLR     @COLUMN(R5)
005646  000237                          RTS     PC
```

```
 1859  006234  000167  003410          ERRSX8: JMP     ERRSX5
 1860
 1861                                          ,IFNDF  $NOSTR
 1862                                  INPSTR: MOV     LINE(R5),R3
 1863                                          MOV     R3,R2
 1864                                          CMPB    (R3)+,#CR
 1865                                          BNE     .+4
 1866                                          MOV     R2,-(SP)
 1867                                          SUB     #3,(SP)
 1868                                          MOV     R3,-(SP)
 1869                                          SUB     R2,(SP)
 1870                                          DEC     (SP)
 1871                                          BEQ     INPNULL
 1872                                          MOV     SP,R2
 1873                                          ADD     #2,R2
 1874                                          JSR     PC,MAKESTR
 1875                                  INPNNUL:JSR     PC,STOSVAR
 1876                                          TST     (SP)+
 1877                                          CMPB    (R1)+,#,COMMA
 1878                                          BNE     INPEND
 1879                                          TST     (SP)+
 1880                                          JMP     INPY01
 1881                                  INPNULL:MOV     #-1,(SP)
 1882                                          BR      INPNNUL
 1883                                          ,ENDC
 1884
 1885  006240  000201                  INPEOL: ,WORD   ,EOL
 1886  006242  000167  011570          ERRTR8: JMP     ERRTRN
 1887
```



```
 1888                                  ; READ STATEMENT
 1889  006246  112102                  READ:   MOVB    (R1)+,R2
 1890  006250  100771                          BMI     ERRSX8
 1891  006252  000302                          SWAB    R2
 1892  006254  152102                          BISB    (R1)+,R2
 1893  006256  061502                          ADD     (R5),R2
 1894  006260  004767  005320                  JSR     PC,GETVAR
 1895  006264  017503  000004                  MOV     @PDL(R5),R3
 1896  006270  001444                          BEQ     ERRDATA
 1897  006272  020327  177777                  CMP     R3,#-1
 1898  006276  001451                          BEQ     READSRCH
 1899  006300  121327  000201                  CMPB    (R3),#,EOL
 1900  006304  001450                          BEQ     READFND
 1901  006306  122327  000243                  CMPB    (R3)+,#,COMMA
 1902  006312  001036                          BNE     READBAD
 1903  006314  111302                  READGOT:MOVB    (R3),R2
 1904
 1905                                          ,IFNDF  $NOSTR
 1906                                          CMPB    R2,#DQUOTE
 1907                                          BEQ     READQT
 1908                                          CMPB    R2,#SQUOTE
 1909                                          BEQ     READQT
 1910                                          ,ENDC
 1911
 1912  006316  004767  006142                  JSR     PC,LITEVAL
 1913  006322  000432                          BR      READBAD
 1914  006324  010346                          MOV     R3,-(SP)
 1915  006326  004767  010712                  JSR     PC,STUVAR
 1916  006332  012603                  READDUN:MOV     (SP)+,R3
 1917  006334  121327  000201                  CMPB    (R3),#,EOL
 1918  006340  001403                          BEQ     .+10
 1919  006342  121327  000243                  CMPB    (R3),#,COMMA
 1920  006346  001020                          BNE     READBAD
 1921  006350  010375  000004                  MOV     R3,@PDL(R5)
 1922  006354  122127  000243                  CMPB    (R1)+,#,COMMA
 1923  006360  001732                          BEQ     READ
 1924  006362  126127  177777  000201          CMPB    -1(R1),#,EOL
 1925  006370  001321                          BNE     ERRSX8
 1926  006372  000167  174256                  JMP     EXECUTE
 1927  006376  005075  000004          READOUT:CLR     @PDL(R5)
 1928  006402  000004                  ERRDATA:IOT
 1929                                          ,IFNDF  $LONGER
 1930  006404  047517     104                  ,ASCII  \OOD\
 1931                                          ,ENDC
 1932                                          ,IFDF   $LONGER
 1933                                          ,ASCII  'OUT OF DATA'
 1934                                          ,ENDC
 1935  006407     000                          ,BYTE 0
 1936                                          ,EVEN
 1937  006410  005075  000004          READBAD:CLR     @PDL(R5)
 1938  006414  000004                          IOT
 1939                                          ,IFNDF  $LONGER
 1940  006416  042102     122                  ,ASCII  \BOR\
 1941                                          ,ENDC
 1942                                          ,IFDF   $LONGER
```

```
                                        ; 'INPUT' STATEMENT
005650  020165  000016          INPUT:  CMP     R1,CODE(R5)
005654  101006                          BHI     INPYES
005656  004167  007004                  JSR     R1,MSGERR
                                        .IFNDF  $LONGER
005662  046111     116                  .ASCII  \ILN\
                                        .ENDC
                                        .IFDF   $LONGER
                                        .ASCII  'ILLEGAL NOW'
                                        .ENDC
005665     000                          .BYTE 0
                                        .EVEN
005666  000167  173246                  JMP     READY2
005672  105065  000062          INPYES: CLRB    T3(R5)
005676  105065  000077                  CLRB    IDEV(R5)        ;RESET INPUT TO TTY
005702  121127  000201                  CMPB    (R1),#'POUND    ;IS IT 'INPUT #N' ...?
005706  001010                          BNE     INPY01
005710  105265  000062                  INCB    T3(R5)
005714  005201                          INC     R1
005716  004767  002402                  JSR     PC,CHKISET      ;CHECK CHANNEL AND SET-UP INPUT
005722  122127  000200                  CMPB    (R1)+,#',COLON
005726  001142                          BNE     ERRSX8
005730  010746          INPY01: MOV     PC,-(SP)
005732  062716  000306          INPPC:  ADD     #INPEOL-INPPC,(SP)
005736  112102          INPLOOP:MOVB    (R1)+,R2
005740  100535                          BMI     ERRSX8
005742  000302                          SWAB    R2
005744  152102                          BISB    (R1)+,R2
005746  061502                          ADD     (R5),R2
005750  004767  005030                  JSR     PC,GETVAR
005754  011603          INPRTRY:MOV     (SP),R3
005756  121327  000201                  CMPB    (R3),#,EOL
005762  001050                          BNE     INPOK
005764  105765  000062          INPNEW: TSTB    T3(R5)
005770  001003                          BNE     NOQM
005772  004167  007476                  JSR     R1,MSG
005776     077                          .ASCII  '?'
005777     000                          .BYTE 0
                                        .EVEN
006000  010146          NOQM:   MOV     R1,-(SP)
006002  004767  006336                  JSR     PC,LINGET
006006  103002                          BCC     NOQM1
006010  000167  000366                  JMP     ERRDATA
006014  012601          NOQM1:  MOV     (SP)+,R1
006016  016502  000022                  MOV     VARSAV(R5),R2

                                        .IFNDF  $NOSTR
                                        CMP     (R2),#,SVAR
                                        BEQ     INPSTR
                                        .ENDC

006022  010146                          MOV     R1,-(SP)
006024  010446                          MOV     R4,-(SP)
006026  016501  000020                  MOV     LINE(R5),R1
006032  122127  000015                  CMPB    (R1)+,#CR
```



```
006036  001375                          BNE     .-4
006040  010104                          MOV     R1,R4
006042  005204                          INC     R4
006044  020465  000016                  CMP     R4,CODE(R5)
006050  101074                          BHI     ERRTR8
006052  114144                          MOVB    -(R1),-(4)
006054  020165  000020                  CMP     R1,LINE(R5)
006060  101374                          BHI     .-6
006062  112711  000054                  MOVB    #',,(R1)
006066  004767  011202                  JSR     PC,TRAN
006072  012604                          MOV     (SP)+,R4
006074  012601                          MOV     (SP)+,R1
006076  016516  000020                  MOV     LINE(R5),(SP)
006102  000724                          BR      INPRTRY
006104  016502  000022          INPOK:  MOV     VARSAV(R5),R2

                                        .IFNDF  $NOSTR
                                        CMP     (R2),#,SVAR
                                        BEQ     INPNEW
                                        .ENDC

006110  005203                          INC     R3
006112  005065  000040                  CLR     FAC1(R5)
006116  005065  000042                  CLR     FAC2(R5)
006122  121327  000243                  CMPB    (R3),#,COMMA
006126  001403                          BEQ     INPSTO
006130  004767  006330                  JSR     PC,LITEVAL
006134  000431                          BR      INPNGUD
006136  010316          INPSTO: MOV     R3,(SP)
006140  004767  011100                  JSR     PC,STOVAR
006144  011603                          MOV     (SP),R3
006146  121327  000243                  CMPB    (R3),#,COMMA
006152  001403                          BEQ     INPGOOD
006154  121327  000201                  CMPB    (R3),#,EOL
006160  001017                          BNE     INPNGUD
006162  122127  000243          INPGOOD:CMPB    (R1)+,#,COMMA
006166  001663                          BEQ     INPLOOP
006170  126127  177777  000201  INPEND: CMPB    -1(R1),#,EOL
006176  001016                          BNE     ERRSX8
006200  005726                          TST     (SP)+
006202  105765  000077                  TSTB    IDEV(R5)        ;IF TTY INPUT, CLR COLUMN COUNT
006206  001002                          BNE     GOEXEC
006210  005065  000036                  CLR     CLMNTTY(R5)     ;  FOR SAME
006214  000167  174434          GOEXEC: JMP     EXECUTE
006220  004167  007242          INPNGUD:JSR     R1,MSGERR
                                        .IFNDF  $LONGER
006224  051102     124                  .ASCII  \BRT\
                                        .ENDC
                                        .IFDF   $LONGER
                                        .ASCII  'BAD DATA-RETYPE FROM ERROR,'
                                        .ENDC
006227     015     012                  .BYTE   CR,LF
006231     000                          .BYTE 0
                                        .EVEN
006232  000654                          BR      INPNEW
```

```
 1943                                           ,ASCII  'BAD DATA READ'
 1944                                           ,ENDC
 1945  006421     000                           ,BYTE 0
 1946                                           ,EVEN
 1947  006422  016503  000016          READSRCH:MOV     CODE(R5),R3
 1948  006426  105713                  READFIND:TSTB    (R3)
 1949  006430  100402                           BMI     ,+6
 1950  006432  062703  000002                   ADD     #2,R3
 1951  006436  122327  000223                   CMPB    (R3)+,#,DATA
 1952  006442  001724                           BEQ     READGOT
 1953  006444  124327  000225                   CMPB    -(R3),#,EOF
 1954  006450  001752                           BEQ     READOUT
 1955  006452  010146                           MOV     R1,-(SP)
 1956  006454  010301                           MOV     R3,R1
 1957  006456  004767  010476                   JSR     PC,SKIPEOL
 1958  006462  010103                           MOV     R1,R3
 1959  006464  012601                           MOV     (SP)+,R1
 1960  006466  000757                           BR      READFIND
 1961
 1962                                           ,IFNDF  $NOSTR
 1963                                  READQT1: INC     R3
 1964                                           CMPB    (R3)+,#,TEXT
 1965                                           BNE     READBAD
 1966                                           MOV     R3,R0
 1967                                           TSTB    (R3)+
 1968                                           BNE     ,+2
 1969                                           CMPB    (R3)+,R2
 1970                                           BNE     READBAD
 1971                                           MOV     R3,-(SP)
 1972                                           MOV     R0,-(SP)
 1973                                           SUB     #3,(SP)
 1974                                           MOV     SP,R2
 1975                                           MOV     R3,-(SP)
 1976                                           SUB     R0,(SP)
 1977                                           SUB     #2,(SP)
 1978                                           BEQ     READNULL
 1979                                           JSR     PC,MAKESTR
 1980                                           MOV     (SP)+,R3
 1981                                           MOV     R3,(SP)
 1982                                           MOV     SP,R0
 1983                                           ADD     #3,R3
 1984                                           MOVB    R0,-(R3)
 1985                                           SWAB    R0
 1986                                           MOVB    R0,-(R3)
 1987                                           BR      READSTR
 1988                                  READNULL:DEC     (SP)
 1989                                           MOV     (SP)+,(SP)
 1990                                  READSTR: JSR     PC,STOSVAR
 1991                                           BR      READDUN
 1992                                           ,ENDC
 1993
 1994
```



```
 1995                                  ;*****************************************************
 1996                                  ;*****************************************************
 1997                                  ;************                         ************
 1998                                  ;************   INTERUPT HANDLERS     ************
 1999                                  ;************                         ************
 2000                                  ;*****************************************************
 2001                                  ;*****************************************************
 2002
 2003
 2004
 2005                                  ;
 2006                                  ; KBINT - KEYBOARD INTERUPT SERVICE
 2007                                  ;
 2008                                  ;  PASS ANY CHAR, TO BUFF, EXCEPT 000 (IGNORE)
 2009                                  ;   AND 003 ('C') SET FLAG FOR LATER ACTION,)
 2010                                  ;   IF BUFF, FULL, IGNORE CHAR BUT ECHO BELL,
 2011                                  ;
 2012  006470  004567  010356          KBINT:  JSR     R5,SAVRGI       ;CAUSE RTI ON REG, RESTORE
 2013  006474  016705  0000006                  MOV     USRAREA,R5
 2014  006500  016501  000102                   MOV     KBWD(R5),R1     ;KB BUFF, WDR,
 2015  006504  012704  177564                   MOV     #TPS,R4
 2016  006510  113720  177562                   MOVB    @#TKB,R0        ;GET THAT CHAR,
 2017  006514  042700  177600                   BIC     #177600,R0
 2018  006520  001442                           BEQ     KBIDUN          ;IGNORE 000
 2019  006522  120027  000003                   CMPB    R0,#003         ;'C'?
 2020  006526  001005                           BNE     KBCO
 2021  006530  105265  000106                   INCB    CNCFLG(R5)      ;MARK 'C' HIT
 2022  006534  004767  001372                   JSR     PC,BUFCLR       ;CLEAR ALL BUFFERS
 2023  006540  000410                           BR      DOPUT
 2024  006542  120027  000017          KBCO:   CMPB    R0,#17          ;CHECK 'O'
 2025  006546  001005                           BNE     DOPUT
 2026  006550  105165  000107                   COMB    CNOFLG(R5)
 2027  006554  001402                           BEQ     DOPUT           ;IF PRINT RESTORED,BYPASS
 2028  006556  004767  001424                   JSR     PC,BUFTP        ;CLEAR TELEPRINTER BUFFER
 2029  006562  004767  010016          DOPUT:  JSR     PC,PUTBYT
 2030  006566  103017                           BCC     KBIDUN          ;GOT IN
 2031  006570  105761  000012                   TSTB    BFSPEC(R1)
 2032  006574  001003                           BNE     S,0101
 2033  006576  110061  000012                   MOVB    R0,BFSPEC(R1)
 2034  006602  000207                           RTS     PC
 2035  006604  105714          S,0101: TSTB    (R4)            ;NO ROOM! DO BELL
 2036  006606  100376                           BPL     S,0101
 2037  006610  042714  000100                   BIC     #100,(R4)       ;TEMP, CLR, INTER,
 2038  006614  112737  000007  177566          MOVB    #007,@#TPB      ;INSERT BELL
 2039  006622  105714          S,0102: TSTB    (R4)            ;WAIT UNTIL DONE
 2040  006624  100376                           BPL     S,0102
 2041  006626  012737  000101  177560  KBIDUN: MOV     #101,@#TKS      ;RE-INIT, READER
 2042  006634  012714  000100                   MOV     #100,(R4)       ;ENABLE TPTR, FOR ECHO
 2043  006640  000207                           RTS     PC              ;WILL RESTORE REGS, AND RTI
 2044
```

```
                                        ;
                                        ; TPINT - TELEPRINTER INTERUPT HANDLER
                                        ;
                                        ;  USED TO PRINT OUT TEXT AND ECHO TYPE-IN,
                                        ;  TO HANDLE CASE OF PENDING REQUESTS TO DO TEXT OUTPUT
                                        ;    AND ECHO, DOES THE ORDERLY THING BY FOLLOWING THIS
                                        ;    PRIORITY:
                                        ;         IF ECHOSP(R5) SHOWS NEED TO DO 2ND HALF OF PAIRED
                                        ;          CHARS, FOR ECHO (^U, <CR><LF>), DO FIRST,
                                        ;         IF TEXT MESS, IN PROGRESS, FINISH IT (TO <LF>),
                                        ;         ELSE, ECHO,
                                        ;         ELSE, TURN OFF INTERUPTS.
                                        ;
006642  004567  010204          TPINT:  JSR     R5,SAVRGI       ;CAUSE RTI ON REG, RESTORE
006646  016705  000000                  MOV     USRAREA,R5
006652  105765  000110                  TSTB    FILLCO(R5)
006656  001404                          BEQ     TPNXT
006660  105365  000110                  DECB    FILLCO(R5)
006664  005000                          CLR     R0
006666  000416                          BR      TYPE1
006670  012704  000104          TPNXT:  MOV     #ECHOSP,R4
006674  060504                          ADD     R5,R4           ;SAVED HERE THRU-OUT ROUTINE
006676  005714                          TST     (R4)            ;SPECIAL ECHO IN PROGRESS?
006700  001415                          BEQ     TRYECHO         ;NO
006702  113400                  SPECMES:MOVB    @(R4)+,R0       ;NEXT SP, ECHO CHAR,
006704  001412                          BEQ     CLRECHO
006706  005244                          INC     -(R4)           ;PT, TO NEXT CHAR,
006710  120065  000112          TYPE1:  CMPB    R0,FILLCH(R5)
006714  001003                          BNE     TYPE1
006716  116565  000111  000110          MOVB    FILLNO(R5),FILLCO(R5)
006724  110037  177566          TYPE11: MOVB    R0,@#TPB
006730  000207                          RTS     PC              ;RESTORE REGS, AND RETURN
006732  005044          CLRECHO:CLR     -(R4)
006734  016501  000102          TRYECHO:MOV     KBWD(R5),R1     ;NO MESS, + ECHO?

006740  004767  004424                  JSR     PC,GET2BYT
006744  103011                          BCC     GOTECHO
006746  016501  000100          MSGSUP: MOV     TPWD(R5),R1     ;TRY MESSAGE
006752  004767  004412                  JSR     PC,GET2BYT
006756  103354                          BCC     TYPE            ;GOT ONE! GO DO IT
006760  042737  000100  177564  OFFTYPE:BIC     #100,@#TPS      ;CLEAR INTERUPT
006766  000207                          RTS     PC

006770  004367  001230          GOTECHO:JSR     R3,CHKCHR       ;CHECK THAT ECHO CHAR,
006774  000420                          BR      PRNTDEL         ;LINE DELETE
006776  000422                          BR      SETLFE          ;<CR>: <LF> AS ECHOSP
007000  000424                          BR      ECHOBA          ;RUB OUT: DO "\"
007002  000401                          BR      TPCN
007004  000741                          BR      TYPE
007006  120027  000003          TPCN:   CMPB    R0,#3           ;CHEC /^C/
007012  001003                          BNE     TPCO
007014  012714  007073                  MOV     #CCMES,(R4)
007020  000730                          BR      SPECMES
007022  120027  000017          TPCO:   CMPB    R0,#17          ;CHECK /^O/
007026  001342                          BNE     TRYECHO
```



```
007030  012714  007076                  MOV     #COMES,(R4)
007034  000722                          BR      SPECMES
                                        ;
007036  012714  007060          PRNTDEL:MOV     #DELMSG,(R4)    ;ST, ADDR, INTO ECHOSP(R5)
007042  000717                          BR      SPECMES
                                        ;
007044  012714  007070          SETLFE: MOV     #CRLFMES,(R4)
007050  000714                          BR      SPECMES
                                        ;
007052  112700  000137          ECHOBA: MOVB    #'\,R0
007056  000714                          BR      TYPE
                                        ;
                                DELMSG:
007060  042040  046105  052105          .ASCII  ' DELETED'
        007066  042105
007070     015     012     000  CRLFMES:.BYTE   015,012,000
007073     136     103          CCMES:  .ASCII  '^C'
007075     000                          .BYTE 0
007076  047536                  COMES:  .ASCII  '^O'
007100     015     012     000          .BYTE   015,012,000
        007104                          .EVEN
```

```
 2121                                           .IFNDF  $NOPTP
 2122                                   ;
 2123                                   ; PTRINT - PAPER-TAPE READER INTERUPT HANDLER
 2124                                   ;
 2125                                   ;  CLEARS PARITY BIT,  FLUSHES 000, RUB-OUT, AND <LF>;
 2126                                   ;  IF CHAR, WON'T FIT IN BUFF,, SAVE IN BFSPEC (LO BYTE),
 2127                                   ;  ON EOT OR OTHER ERROR, SET HI BIT OF BFSPEC,
 2128                                   ;
 2129 007104  004567  007742           PTRINT: JSR     R5,SAVRGI       ;CAUSE RTI ON REG, RESTORE
 2130 007110  012701  021242                   MOV     #PRBFHD,R1      ;FOR PUTBYT
 2131 007114  005737  177550                   TST     @#PRS           ;ERRORS?
 2132 007120  100425                           BMI     PREOT           ;YES: TREAT AS EOT
 2133 007122  113700  177552                   MOVB    @#PRB,R0        ;CHARACTER
 2134 007126  042700  177600                   BIC     #177600,R0
 2135 007132  001414                           BEQ     EXIT02          ;IGNORE 000 (CAN'T HAPPEN!)
 2136 007134  122700  000177                   CMPB    #177,R0         ;RUB OUT?
 2137 007140  001411                           BEQ     EXIT02          ;IGNORE
 2138 007142  122700  000012                   CMPB    #LF,R0
 2139 007146  001406                           BEQ     EXIT02          ;IGNORE LF'S
 2140 007150  004767  007430                   JSR     PC,PUTBYT
 2141 007154  103003                           BCC     EXIT02          ;GOT IN
 2142 007156  110061  000012                   MOVB    R0,BFSPEC(R1)   ;SAVE THIS CHAR,
 2143 007162  000207                           RTS     PC              ;RESTORE REGS, AND RTI
 2144 007164  012737  000101  177550  EXIT02: MOV     #101,@#PRS      ;RE-ENABLE READER
 2145 007172  000207                           RTS     PC
 2146 007174  052761  100000  000012  PREOT:  BIS     #100000,BFSPEC(R1)      ;SHOW EOT
 2147 007202  000207                           RTS     PC
 2148                                           .ENDC
 2149
```



```
 2150                                           .IFNDF  $NOPTP
 2151                                   ;
 2152                                   ; PPINT - PAPER TAPE PUNCH INTERUPT HANDLER
 2153                                   ;
 2154                                   ;  ON ERROR, MAKE BFSPEC NON-0,
 2155                                   ;
 2156 007204  004567  007642           PPINT:  JSR     R5,SAVRGI       ;CAUSE RTI AFTER REG, RESTORE
 2157 007210  016705  000000G                  MOV     USRAREA,R5
 2158 007214  012701  021306                   MOV     #PPBFHD,R1
 2159 007220  005737  177554                   TST     @#PPS
 2160 007224  100406                           BMI     PPERR
 2161 007226  004767  004130                   JSR     PC,GET1BYT
 2162 007232  103406                           BCS     NOCHR2          ;NOTHING TO GET
 2163 007234  110037  177556                   MOVB    R0,@#PPB
 2164 007240  000207                           RTS     PC              ;RESTORE REGS, AND RTI
 2165 007242  052761  000001  000012  PPERR:  BIS     #1,BFSPEC(R1)   ;INDIC, ERROR
 2166 007250  005037  177554           NOCHR2: CLR     @#PPS           ;CLEAR INTERUPT ENABLE
 2167 007254  000207                           RTS     PC
 2168                                   ;
 2169                                           .ENDC
 2170
```

```
2171                                            .IFNDF  $NOLPI
2172                                    ;
2173                                    ; LPINT - LINE PRINTER INTERUPT HANDLER
2174                                    ;
2175  007256  004567  007570           LPINT:  JSR     R5,SAVRGI       ;CAUSE RTI AFTER REG. RESTORE
2176  007262  016705  000000G                   MOV     USRAREA,R5
2177  007266  012701  021352                    MOV     #LPBFHD,R1
2178  007272  005737  177514                    TST     @#LPS
2179  007276  100406                           BMI     LPERR
2180  007300  004767  004056                    JSR     PC,GET1BYT
2181  007304  103406                           BCS     LPOFF           ;NOTHING THERE
2182  007306  110037  177516                    MOVB    R0,@#LPB        ;WRITE CHAR
2183  007312  000207                           RTS     PC
2184                                    ;
2185  007314  052761  000001  000012  LPERR:  BIS     #1,BFSPEC(R1)   ;RECORD ERROR
2186  007322  005037  177514          LPOFF:  CLR     @#LPS           ;TURN IT OFF FOR AWHILE
2187  007326  000207                           RTS     PC
2188                                            .ENDC
2189
2190
```



```
2191                                            .IFNDF  $NOPOW
2192                                    ;
2193                                    ; POWDWN - POWER FAIL/RESTART ROUTINE
2194                                    ;
2195  007330  012737  007340  000024  POWDWN: MOV     #POWUP,@#24
2196  007336  000000                           HALT
2197                                    ;
2198  007340  005000                   POWUP:  CLR     R0      ;INITIALIZE LOOP COUNT
2199  007342  016705  000000G          POWLP:  MOV     USRAREA,R5
2200  007346  012737  007330  000024           MOV     #POWDWN,@#24
2201  007354  016506  000004                   MOV     POL(R5),SP      ;WASTE TIME (AND INIT STACK!)
2202  007360  005300                           DEC     R0              ;COUNT TILL DONE
2203  007362  001367                           BNE     POWLP
2204  007364  000167  171502                   JMP     READY@
2205                                            .ENDC
2206                                            .EOT
2207
```

```
2208                                    ;                                       SOURCE FILE #4
2209                                    ;*****************************************************
2210                                    ;*****************************************************
2211                                    ;***********                                 ***********
2212                                    ;***********     CLOSED SUBROUTINES          ***********
2213                                    ;***********                                 ***********
2214                                    ;*****************************************************
2215                                    ;*****************************************************
2216
2217
2218
2219
2220
2221                                    ;-------------------------------------------------------
2222                                    ;  SUBROUTINE 'ADDSTK'  CALLED BY JSR PC
2223                                    ;                       ADDS (SP)+,(SP)+ TO FAC1(R5),FAC2(R5) WATCHING TYPE
2224                                    ;                       R0,R1  UNUSED
2225                                    ;                       R2,R3  DESTROYED
2226                                    ;                       R4  MODIFIED TO REFLECT STACK USAGE
2227                                    ;                       R5  MUST POINT TO USER AREA
2228                                    ;                       SP  GOES ?? DEEPER AFTER JSR
2229  007370  012603                   ADDSTK: MOV     (SP)+,R3
2230  007372  004767  003576                   JSR     PC,FIXUP
2231  007376  001414                           BEQ     ADDINT
2232  007400  012767  011666  000000G          MOV     #ERADD,$ERVEC
2233  007406  004467  003722                   JSR     R4,FPPSAV
2234  007412  014256                           .WORD   PUSH
2235  007414  000000G                          .WORD   $ADR
2236  007416  014270                           .WORD   POP
2237  007420  013306                           .WORD   FPPRES
2238  007422  005765  000040                   TST     FAC1(R5)
2239  007426  000113                           JMP     (R3)            ;COND CODES=SGN(RESULT)
2240  007430  005726                   ADDINT: TST     (SP)+
2241  007432  012602                           MOV     (SP)+,R2
2242  007434  066502  000042                   ADD     FAC2(R5),R2
2243  007440  102403                           BVS     ADDOVF
2244  007442  010265  000042                   MOV     R2,FAC2(R5)
2245  007446  000113                           JMP     (R3)            ;COND CODES=SGN(RESULT)
2246  007450  100425                   ADDOVF: BMI     ADDPOS
2247  007452  001416                           BEQ     ADDZERO
2248  007454  005402                           NEG     R2
2249  007456  105065  000042                   CLRB    FAC2(R5)
2250  007462  110265  000043                   MOVB    R2,FAC2+1(R5)
2251  007466  000302                           SWAB    R2
2252  007470  110265  000040                   MOVB    R2,FAC1(R5)
2253  007474  105065  000041                   CLRB    FAC1+1(R5)
2254  007500  062765  143600  000040          ADD     #143600,FAC1(R5)
2255  007506  000113                           JMP     (R3)            ;COND CODES=SGN(RESULT)
2256  007510  005065  000042           ADDZERO:CLR     FAC2(R5)
2257  007514  012765  144200  000040          MOV     #144200,FAC1(R5)
2258  007522  000113                           JMP     (R3)            ;COND CODES=SGN(RESULT)
2259  007524  105065  000042           ADDPOS: CLRB    FAC2(R5)
2260  007530  110265  000043                   MOVB    R2,FAC2+1(R5)
2261  007534  000302                           SWAB    R2
2262  007536  110265  000040                   MOVB    R2,FAC1(R5)
```



```
2263  007542  105065  000041                   CLRB    FAC1+1(R5)
2264  007546  062765  043600  000040          ADD     #43600,FAC1(R5)
2265  007554  000113                           JMP     (R3)            ;COND CODES=SGN(RESULT)
2266
```

```
                        ;+-----------------------------------------------
                        ;  SUBROUTINE 'ALLOC'   CALLED BY JSR PC
                        ;                      ALLOCATES ARRAY SPACE FROM FREESPACE
                        ;                      R0,R1  PRESERVED
                        ;                      R2  MUST POINT TO VAR    GETS DESTROYED
                        ;                      R3  MUST CONTAIN SS1MAX   GETS DESTROYED
                        ;                      R4  MUST CONTAIN SS2MAX    GETS DESTROYED
                        ;                      R5  MUST POINT TO USER AREA
                        ;                      SP  GOES ?? DEEPER AFTER JSR
007556 010046           ALLOC:  MOV     R0,-(SP)
007560 010146                   MOV     R1,-(SP)
007562 005203                   INC     R3
007564 010401                   MOV     R4,R1
007566 005201                   INC     R1
007570 001776                   BEQ     .-2
007572 005000                   CLR     R0                      ;****
007574 012746 000020            MOV     #20.,-(SP)              ;*  *
007600 006301           ALLLOOP:ASL     R1                      ;*  *
007602 103002                   BCC     ALLNOAD                 ;*  *
007604 060300                   ADD     R3,R0                   ;*  *  MULTIPLY (R1)*(R3)*2
007606 103447                   BCS     ERRARAY                 ;*  *  RESULT IN R0
007610 006300           ALLNOAD:ASL     R0                      ;*  *  WATCH FOR OVERFLOWS
007612 103445                   BCS     ERRARAY                 ;*  *
007614 005316                   DEC     (SP)                    ;*  *
007616 003370                   BGT     ALLLOOP                 ;*  *
007620 005726                   TST     (SP)+                   ;****
007622 062700 000002            ADD     #2,R0
007626 103437                   BCS     ERRARAY         ;R0 NOW = 2 TIMES # ELEMENTS NEEDED,
                        ;
                                .IFNDF  $NOSTR
                                CMP     (R2),#.SVAR
                                BEQ     .+6
                                .ENDC
                        ;
007630 006300                   ASL     R0
007632 103435                   BCS     ERRARAY         ;R0 NOW = # BYTES NEEDED,
007634 016501 000012            MOV     HIFREE(R5),R1
007640 160001                   SUB     R0,R1
007642 103431                   BCS     ERRARAY
007644 020165 000074            CMP     R1,HISTR(R5)
007650 103426                   BLO     ERRARAY
007652 005721                   TST     (R1)+
007654 000401                   BR      .+4
007656 005021                   CLR     (R1)+
007660 020165 000012            CMP     R1,HIFREE(R5)
007664 101774                   BLOS    .-6
007666 160001                   SUB     R0,R1
                        ;
                                .IFNDF  $NOSTR
                                CMP     (R2),#.SVAR
                                BEQ     ALLSTRN
                                .ENDC
                        ;
007670 012722 177776            MOV     #-NVAR,(R2)+
007674 012221                   MOV     (R2)+,(R1)+
```



```
007676 012221                   MOV     (R2)+,(R1)+
007700 010412                   MOV     R4,(R2)
007702 005303                   DEC     R3
007704 010342                   MOV     R3,-(R2)
007706 010142                   MOV     R1,-(R2)
007710 162712 000004            SUB     #4,(R2)
                        ;
                                .IFNDF  $NOSTR
                                BR      ALLEXIT
                        ALLSTRN:TST     (R2)+
                                MOV     (R2)+,(R1)+
                                DEC     R3
                                MOV     R3,(R2)+
                                MOV     R4,(R2)
                                MOV     R1,-4(R2)
                                SUB     #2,-4(R2)
                                MOV     #-1,(R1)+
                                CMP     R1,HIFREE(R5)
                                BLOS    .-10
                                SUB     R0,R1
                                MOV     (R1),R2
                                INC     R2
                                BEQ     ALLEXIT
                                SWAB    R1
                                MOVB    R1,(R2)+
                                SWAB    R1
                                MOVB    R1,(R2)
                                .ENDC
                        ;
007714 160005 000012    ALLEXIT:SUB     R0,HIFREE(R5)
007720 012601                   MOV     (SP)+,R1
007722 012600                   MOV     (SP)+,R0
007724 000207                   RTS     PC
007726 000004           ERRARAY:IOT
                                .IFNDF  $LONGER
007730 052101     114           .ASCII  \ATL\
                                .ENDC
                                .IFDF   $LONGER
                                .ASCII  'ARRAYS TOO LARGE'
                                .ENDC
007733     000                  .BYTE 0
                                .EVEN
```

```
                                        .IFNDF  $NOSTM
                                ;+-------------------------------------------
                                ;  'ARGB' SUBROUTINE
                                ;                       CALLED BY JSR R7
                                ;                       CALLS EVAL TO GET A 1-BYTE
                                ;                       ARGUMENT, BRANCHES TO ERRARG
                                ;                       IF ARG NO GOOD
                                ARGB:   JSR     PC,EVAL
                                        BCS     ERRARG
                                        JSR     PC,INT
                                        TST     FAC1(R5)
                                        BNE     ERRARG
                                        TSTB    FAC2+1(R5)
                                        BNE     ERRARG
                                        RTS     PC
                                        .ENDC

                                ERSQR:
                                ERLOG:
007734  000004                  ERRARG: IOT
                                        .IFDF   $LONGER
                                        .ASCII  'ARGUMENT ERROR'
                                        .ENDC
                                        .IFNDF  $LONGER
007736  051101  000107                 .ASCII  'ARG'
                                        .ENDC
007741  000000                          .BYTE   0
                                        .EVEN
```



```
                                ;+-------------------------------------------
                                ;  SUBROUTINE 'BOMB'    CALLED BY IOT
                                ;                       PRINTS ERROR MESSAGE FROM AFTER IOT
                                ;                       R0  DESTROYED
                                ;                       R1  PRESERVED
                                ;                       R2,R3,R4  UNUSED
                                ;                       R5  MUST POINT TO USER AREA
                                ;                       SP  RESET TO EMPTY STACK
                                ;                       PC  DOES NOT RETURN
                                ;
007742  011601                  BOMB:   MOV     (SP),R1
007744  016705  000000G                 MOV     USRAREA,R5
007750  016506  000004                  MOV     PDL(R5),SP
007754  005065  000076                  CLR     ODEV(R5)        ;CAUSE MES OUTPUT TO TTY
007760  012765  000036  000034          MOV     #CLMNTTY,COLUMN(R5)     ;SET-UP TTY FOR COL, COUNT
007766  060565  000034                  ADD     R5,COLUMN(R5)
007772  004167  005476                  JSR     R1,MSG
007776  000015  000012                  .BYTE   CR,LF
                                        .IFNDF  $LONGER
010000  000045                          .ASCII  '%'
                                        .ENDC
010001  000000                          .BYTE 0
                                        .EVEN
010002  112100                  BOMBNX: MOVB    (R1)+,R0
010004  001403                          BEQ     BOMBDON
010006  004767  006664                  JSR     PC,PUTCHAR
010012  000773                          BR      BOMBNX
010014  026527  000030  177775  BOMBDON:CMP     LINENO(R5),#32767.
010022  103041                          BHIS    BOMBJMP
010024  004167  005444                  JSR     R1,MSG
010030  040440  020124  044514          .ASCII  ' AT LINE '
010036  042516  000040
010041  000000                          .BYTE 0
                                        .EVEN
010042  016500  000030                  MOV     LINENO(R5),R0
010046  002405                          BLT     BOMBNEG
010050  010065  000042                  MOV     R0,FAC2(R5)
010054  005065  000040                  CLR     FAC1(R5)
010060  000414                          BR      BOMBOK
010062  110065  000043          BOMBNEG:MOVB    R0,FAC2+1(R5)
010066  105065  000042                  CLRB    FAC2(R5)
010072  000300                          SWAB    R0
010074  110065  000040                  MOVB    R0,FAC1(R5)
010100  105065  000041                  CLRB    FAC1+1(R5)
010104  062765  043600  000040          ADD     #43600,FAC1(R5)
010112  004767  005630          BOMBOK: JSR     PC,NUMOUT
010116  004167  005352                  JSR     R1,MSG
010122  000015  000012  000            .BYTE   CR,LF,0
                                        .EVEN
010126  000167  170744          BOMBJMP:JMP     READY
```

```
                                        ;
                                        ; BUFCLR--INIT I/O DEV, BUFFER HEADERS
                                        ;
                                        ; CALL: JSR     PC,BUFCLR
                                        ;
                                        ; USES R2 AND R3
                                        ;
010132  016502  000102          BUFCLR: MOV     KBHD(R5),R2     ;KEYBOARD
010136  012203                          MOV     (R2)+,R3        ;BUFF, START
010140  005722                          TST     (R2)+           ;SKIP TO BGET1
010142  010322                          MOV     R3,(R2)+
010144  010322                          MOV     R3,(R2)+
010146  010322                          MOV     R3,(R2)+
010150  004767  000032                  JSR     PC,BUFTP        ;CLEAR TELEPRINTER BUFFER
                                
                                        ,IFNDF  $NOPTP
010154  012702  021306                  MOV     #PPBFHD,R2      ;PAPER TAPE PUNCH
010160  004767  003502                  JSR     PC,INITBF
010164  012702  021242                  MOV     #PRBFHD,R2      ;PAPER TAPE READER
010170  004767  003552                  JSR     PC,INITBF
                                        ,ENDC
                                        ,IFNDF  $NOLPT
010174  012702  021352                  MOV     #LPBFHD,R2
010200  004767  003542                  JSR     PC,INITBF
                                        ,ENDC
                                
010204  000207                          RTS     PC
                                
                                
                                        ;
                                        ; BUFTP--INIT TELEPRINTER BUFFER HEADER
                                        ;
                                        ; CALL: JSR     PC,BUFTP
                                        ;
                                        ; USES R2 AND R3
                                        ;
010206  016502  000100          BUFTP:  MOV     TPHD(R5),R2     ;TELEPRINTER
010212  012203                          MOV     (R2)+,R3        ;BUFF, START
010214  022222                          CMP     (R2)+,(R2)+     ;PT, TO BGETZ
010216  010322                          MOV     R3,(R2)+
010220  010322                          MOV     R3,(R2)+
010222  000207                          RTS     PC
```



```
                                        ;
                                        ;
                                        ; CHKCHR - CHECK ASCII CHAR, FOR SPECIAL CASES
                                        ;
                                        ; CALL: MOV     [CHAR,],R0
                                        ;       JSR     R3,CHKCHR
                                        ;       INSTRUC,        ;EXECUTE IF LINE DEL, (<ALT MODE>,'@,'U)
                                        ;       INSTRUC,        ;EXEC, IF <CR>
                                        ;       INSTRUC,        ;EXEC, IF <RUB OUT> (OR '?)
                                        ;       INSTRUC,        ;EXEC, IF BELOW 40 OR ABOVE 137
                                        ;       INSTRUC,        ;EXEC, IF GOOD STUFF
                                        ;
                                        ;  USES R0
                                        ;
010224  120027  000176          CHKCHR: CMPB    R0,#176         ;ALT MODE
010230  001434                          BEQ     CCEXIT
010232  120027  000175                  CMPB    R0,#175         ;ALT MODE
010236  001431                          BEQ     CCEXIT
010240  120027  000033                  CMPB    R0,#33          ;ALT MODE
010244  001426                          BEQ     CCEXIT
010246  120027  000025                  CMPB    R0,#25          ;'+U'
010252  001423                          BEQ     CCEXIT
010254  005723                          TST     (R3)+           ;TO <CR> RETURN
010256  120027  000015                  CMPB    R0,#CH
010262  001417                          BEQ     CCEXIT
010264  005723                          TST     (R3)+           ;TO RUB OUT RETURN
010266  120027  000177                  CMPB    R0,#177         ;RUB OUT
010272  001413                          BEQ     CCEXIT
010274  120027  000137                  CMPB    R0,#'_
010300  001410                          BEQ     CCEXIT
010302  005723                          TST     (R3)+           ;TO LIMIT RETURN
010304  120027  000040                  CMPB    R0,#40
010310  103404                          BLO     CCEXIT
010312  120027  000137                  CMPB    R0,#137
010316  101001                          BHI     CCEXIT
010320  005723                          TST     (R3)+           ;REG, CHAR, RETURN
010322  000203          CCEXIT: RTS     R3
```

```
2524                                    ;
2525                                    ; CHKISET - CHECK I/O CHANNEL AND SET-UP INPUT (IDEV(R5))
2526                                    ; CHKOSET - CHECK I/O CHANNEL AND SET-UP OUTPUT (ODEV(R5))
2527                                    ;
2528                                    ; CALL: MOV     CADDR; OF CODE LINE PTR, AFTER '#'],R1
2529                                    ;       JSR     PC,CHKISET (CHKOSET)
2530                                    ;
2531                                    ;  USES: R0,R1,R2,R3,R5
2532                                    ;
2533                                    ;  CHECK IF SPECIFIED CHANNEL LEGAL (NUMBER IN RANGE AND I/O
2534                                    ;   DEVICE ASSEMBLED IN),  RETURN CODE FOR SPECIFIED DEVICE
2535                                    ;   (SAME AS VALUE OF IDEV (ODEV)) IN R2,  ON EXIT, R1 PTS'
2536                                    ;   TO BYTE AFTER CHANNEL NUMBER.
2537                                    ;
2538 010324  012700  010421            CHKISET:MOV     #ITABLE,R0
2539 010330  000402                            BR      COMCHK
2540 010332  012700  010416            CHKOSET:MOV     #OTABLE,R0
2541 010336  004767  000254            COMCHK: JSR     PC,EVAL
2542 010342  103420                            BCS     ERSX12
2543 010344  005765  000040                    TST     FAC1(R5)
2544 010350  001017                            BNE     ERCHAN          ;MUST BE INTEGER
2545 010352  016502  000042                    MOV     FAC2(R5),R2
2546 010356  001411                            BEQ     OUT012          ;TTY = EVERYTHING SET-UP
2547 010360  111003                            MOVB    (R0),R3         ;FIRST TABLE ENTRY
2548 010362  060503                            ADD     R5,R3           ;PTS, TO ACTUAL BYTE
2549 010364  020227  000002                    CMP     R2,#2           ;MAX CHANNEL NUMBER (CURRENTLY)
2550 010370  101007                            BHI     ERCHAN
2551 010372  060200                            ADD     R2,R0           ;ENTRY IN TABLE OF DEV; CODE
2552 010374  111013                            MOVB    (R0),(R3)       ;CODE INTO IDEV OR ODEV
2553 010376  001404                            BEQ     ERCHAN          ;MEANS NO DEVICE THERE
2554 010400  111002                            MOVB    (R0),R2
2555 010402  000207                    OUT012: RTS     PC
2556                                    ;
2557 010404  000167  001240            ERSX12: JMP     ERRSX5
2558                                    ;
2559 010410  000004                    ERCHAN: IOT
2560                                            .IFNDF  $LONGER
2561 010412  041504      105                   .ASCII  \DCE\
2562                                            .ENDC
2563                                            .IFDF   $LONGER
2564                                            .ASCII  'DEVICE CHANNEL ERROR'
2565                                            .ENDC
2566 010415      000                           .BYTE   0
2567                                            .EVEN
2568                                    ;
2569                                    ;
2570 010416      076                   OTABLE: .BYTE   ODEV
2571                                            .IFDF   $NOPTP
2572                                            .BYTE   0
2573                                            .ENDC
2574                                            .IFNDF  $NOPTP
2575 010417      002                           .BYTE   2
2576                                            .ENDC
2577                                            .IFDF   $NOLPI
2578                                            .BYTE   0
```



```
2579                                            .ENDC
2580                                            .IFNDF  $NOLPI
2581 010420      004                           .BYTE   4
2582                                            .ENDC
2583                                    ;
2584 010421      077                   ITABLE: .BYTE   IDEV
2585                                            .IFDF   $NOPTP
2586                                            .BYTE   0
2587                                            .ENDC
2588                                            .IFNDF  $NOPTP
2589 010422      002                           .BYTE   2
2590                                            .ENDC
2591 010423      000                           .BYTE   0
2592
```

```
2593                                    ;+-----------------------------------------------
2594                                    ;  SUBROUTINE 'CLRVARS' CALLED BY JSR PC
2595                                    ;                       CLEARS VARS TO SCALAR ZEROS,NULL STRINGS
2596                                    ;                       ALSO INITIALIZES RANDOM NO GEN
2597                                    ;                       R0  DESTROYED
2598                                    ;                       R1,R2,R3,R4  UNUSED
2599                                    ;                       R5  MUST POINT TO USER AREA
2600                                    ;                       SP  GOES NO DEEPER AFTER JSR
2601  010424  012765  032331  000064  CLRVARS:MOV     #32331,RND1(R5)
2602  010432  012765  163251  000066          MOV     #163251,RND2(R5)
2603  010440  011500                  CLRLOOP:MOV     (R5),R0
2604  010442  020065  000014                  CMP     R0,LOFREE(R5)
2605  010446  103014                          BHIS    CLRFREE
2606  010450  021027  177775                  CMP     (R0),#,SCALAR
2607  010454  103405                          BLO     CLROK
2608
2609                                          ,IFNDF  $NOSTR
2610                                          CMP     (R0),#,SVAR
2611                                          BEQ     CLRSVAR
2612                                          ,ENDC
2613
2614  010456  012720  177775                  MOV     #,SCALAR,(R0)+
2615  010462  005010                          CLR     (R0)
2616
2617                                          ,IFNDF  $NOSTR
2618                                          BR      ,+4
2619                                  CLRSVAR:MOV     (R0)+,(R0)
2620                                          ,ENDC
2621
2622  010464  012010                          MOV     (R0)+,(R0)
2623  010466  012010                          MOV     (R0)+,(R0)
2624  010470  062700  000004          CLROK:  ADD     #4,R0
2625  010474  000762                          BR      CLRLOOP
2626  010476  005020                          CLR     (R0)+
2627  010500  020065  000010          CLRFREE:CMP     R0,ARRAYS(R5)
2628  010504  101774                          BLOS    ,-6
2629  010506  016565  000010  000012          MOV     ARRAYS(R5),HIFREE(R5)
2630  010514  016565  000014  000072          MOV     LOFREE(R5),LOSTR(R5)
2631  010522  016565  000014  000074          MOV     LOFREE(R5),HISTR(R5)    ;EMPTY STRING STORAGE
2632  010530  012765  000033  000032          MOV     #33,GSBCTR(R5)
2633  010536  016500  000004                  MOV     @0L(R5),R0
2634  010542  012720  177777                  MOV     #-1,(R0)+
2635  010546  005020                          CLR     (R0)+
2636  010550  020065  000002                  CMP     R0,LIMIT(R5)
2637  010554  101774                          BLOS    ,-6
2638  010556  000207                          RTS     PC
2639
```



```
2640                                    ;+-----------------------------------------------
2641                                    ;  SUBROUTINE 'DIVTEN'  CALLED BY JSR PC
2642                                    ;                       DIVIDES R3,R4 BY DECIMAL 10
2643                                    ;                       R0,R1,R2  UNUSED
2644                                    ;                       R3,R4  IS THE 31 BIT UNSIGNED INTEGER TO DIVIDE
2645                                    ;                       R5  UNUSED
2646                                    ;                       SP  GOES 2 DEEPER AFTER JSR
2647  010560  012746  000034          DIVTEN: MOV     #34,-(SP)
2648  010564  022703  050000          DIVLOOP:CMP     #50000,R3
2649  010570  101002                          BHI     ,+6
2650  010572  062703  130000                  ADD     #130000,R3
2651  010576  006104                          ROL     R4
2652  010600  006103                          ROL     R3
2653  010602  005316                          DEC     (SP)
2654  010604  003367                          BGT     DIVLOOP
2655  010606  042703  170000                  BIC     #170000,R3
2656  010612  005726                          TST     (SP)+
2657  010614  000207                          RTS     PC
2658
```

```
2659                                    .IFNDF  $NOSTK
2660                                ;-----------------------------------------------
2661                                ;  SUBROUTINE 'DNPACK'  CALLED BY JSR PC
2662                                ;                      PACKS STRING STORAGE TOWARD LOW CORE
2663                                ;                      R0  UNUSED
2664                                ;                      R1,R2,R3  PRESERVED
2665                                ;                      R4  UNUSED
2666                                ;                      R5  MUST POINT TO USER AREA
2667                                ;                      SP  GOES 1# DEEPER AFTER JSR
2668                                DNPACK: MOV     R1,-(SP)        ;TO UNDERSTAND THESE ROUTINES IT IS HELPFUL TO
2669                                        MOV     R2,-(SP)        ;KNOW THAT NON-NULL STRINGS ARE STORED AS
2670                                        MOV     R3,-(SP)        ;(LENGTH,2 BYTE BACKPTR,N BYTE STRING,LENGTH)
2671                                        CLR     -(SP)   ;WHERE LENGTH IS THE VALUE OF N AND IF BACKPTR
2672                                        MOV     LOSTR(R5),R2    ;IS ODD,THEN IT IS RELATIVE TO SYMBOLS,
2673                                        MOV     LOFREE(R5),R1
2674                                        MOV     R1,LOSTR(R5)
2675                                DNPLOOP:CLR     (SP)
2676                                        BISB    (R2)+,(SP)
2677                                        BNE     DNPNZRO
2678                                DNPBAD: CMP     R2,HISTR(R5)
2679                                        BLO     DNPLOOP
2680                                        MOV     R1,HISTR(R5)    ;SAVE HIGH STRING ADDRESS
2681                                        TST     (SP)+
2682                                        MOV     (SP)+,R3
2683                                        MOV     (SP)+,R2
2684                                        MOV     (SP)+,R1
2685                                        RTS     PC
2686                                DNPNZRO:CLR     R3
2687                                        BISB    (R2)+,R3
2688                                        SWAB    R3
2689                                        BISB    (R2)+,R3
2690                                        ADD     (SP),R2
2691                                        INC     R2
2692                                        BIT     R3,#1
2693                                        BEQ     .+6
2694                                        DEC     R3
2695                                        ADD     (R5),R3
2696                                        CMP     R3,PDL(R5)
2697                                        BHIS    DNPBAD
2698                                        CMP     R3,SP
2699                                        BHIS    DNPGOOD
2700                                        CMP     R3,ARRAYS(R5)
2701                                        BHI     DNPBAD
2702                                        CMP     R3,HIFREE(R5)
2703                                        BHI     DNPGOOD
2704                                        CMP     R3,LOFREE(R5)
2705                                        BHIS    DNPBAD
2706                                        CMP     R3,(R5)
2707                                        BLO     DNPBAD
2708                                DNPGOOD:ADD     #4,(SP)
2709                                        SUB     (SP),R2
2710                                        CMP     R2,(R3)
2711                                        BNE     DNPIGNO
2712                                        MOV     R1,(R3)
2713                                        MOVB    (R2)+,(R1)+
```



```
2714                                        DEC     (SP)
2715                                        BGT     .+4
2716                                        BR      DNPBAD
2717                                DNPIGNO:ADD     (SP),R2
2718                                        BR      DNPBAD
2719                                        .ENDC
2720                                        .EOT
2721
```

```
                                ;                                       SOURCE FILE #5
                                ;=======================================================
                                ;
                                ; EVAL - EVALUATE EXPRESSION
                                ;
                                ; CALL: MOV     [USER-AREA]
                                ;       MOV     [STACK SIZE],R4
                                ;       MOV     [CHAR. PTR.],R1
                                ;       JSR     PC,EVAL
                                ;
                                ;  USES ALL REGS.
                                ;
                                ;  RETURNS 'C' BIT SET IF STRING VAL,
                                ;                 CLR IF NOT STRING
                                ;
                                ;  RETURNS VALUE IN FAC1(R5), FAC2(R5)
                                ;
010616  020604                 EVAL:   CMP     SP,R4
010620  103406                         BLO     BPDL
010622  012746  000235                 MOV     #,TERM,-(SP)
010626  000404                         BR      OPERAND
010630  012746  000233         UMINUS: MOV     #,UNARY,-(SP)
010634  020604                         CMP     SP,R4
010636  103530                 BPDL:   BLO     ERRPDL
010640  112102                 OPERAND:MOVB    (R1)+,R2
010642  002113                         BGE     VARBLE
010644  120227  000375                 CMPB    R2,#,ILIT1
010650  001455                         BEQ     ILIT1
010652  120227  000376                 CMPB    R2,#,ILIT2
010656  001462                         BEQ     ILIT2
010660  120227  000374                 CMPB    R2,#,FLIT
010664  001462                         BEQ     FLIT
010666  120227  000295                 CMPB    R2,#,LPAR
010672  001470                         BEQ     LPAR
010674  120227  000234                 CMPB    R2,#,MINUS
010700  001753                         BEQ     UMINUS
010702  120227  000232                 CMPB    R2,#,PLUS
010706  001754                         BEQ     OPERAND
010710  120227  000262                 CMPB    R2,#,FN
010714  001504                         BEQ     GOTOFN

                                        ,IFNDF  $NOSTR
                                        CMPB    R2,#,$QUOT
                                        BEQ     GOTOQT
                                        CMPB    R2,#,DQUOT
                                        BNE     NOQUOTE
                                GOTOQT: JMP     QUOTE
                                        ,ENDC

010716  042702  177400         NOQUOTE:BIC     #177400,R2
010722  162702  000003                 SUB     #,RNDL,R2
010726  006302                         ASL     R2
010730  020227  000024                 CMP     R2,#TBL5END-TABLE5
010734  101054                         BHI     ERRSX4
010736  062702  000012                 ADD     #12,R2
```



```
010742  060702                         ADD     PC,R2
010744  005712                         TST     (R2)
010746  003002                         BGT     .+6
010750  000167  002042                 JMP     ERRUFN
010754  061207                         ADD     (R2),PC
010756  000000                 TABLE5: .WORD   0                       ; RND(
010760  000000                         .WORD   0                       ; RND
010762  000716                         .WORD   SINFN-TABLE5
010764  000724                         .WORD   COSFN-TABLE5
010766  000732                         .WORD   SQRFN-TABLE5
010770  000746                         .WORD   ATNFN-TABLE5
010772  000754                         .WORD   EXPFN-TABLE5
010774  000770                         .WORD   LOGFN-TABLE5
010776  000000                         .WORD   0                       ; ABS
011000  001122                         .WORD   INTFN-TABLE5
011002  000000                         .WORD   0                       ; SGN

                                        ,IFNDF  $NOSTR
                                        .WORD   0                       ; TAB
                                        .WORD   0                       ; LEN
                                        .WORD   0                       ; ASC
                                        .WORD   0                       ; CHR$
                                        .WORD   0                       ; POS
                                        .WORD   0                       ; SEG
                                        .WORD   0                       ; VAL
                                        .WORD   0                       ; STR
                                        ,ENDC

011002                          TBL5END =.-2
011004  005065  000040         ILIT1:  CLR     FAC1(R5)
011010  105065  000043                 CLRB    FAC2+1(R5)
011014  112165  000042                 MOVB    (R1)+,FAC2(R5)
011020  000167  000246                 JMP     OPRATOR
011024  005065  000040         ILIT2:  CLR     FAC1(R5)
011030  000404                         BR      ILITCOM
011032  112165  000041         FLIT:   MOVB    (R1)+,FAC1+1(R5)
011036  112165  000040                 MOVB    (R1)+,FAC1(R5)
011042  112165  000043         ILITCOM:MOVB    (R1)+,FAC2+1(R5)
011046  112165  000042                 MOVB    (R1)+,FAC2(R5)
011052  000507                         BR      OPRATOR
011054  004767  177536         LPAR:   JSR     PC,EVAL

                                        ,IFNDF  $NOSTR
                                        BCS     LPSTRNG
                                        ,ENDC

011060  122127  000237                 CMPB    (R1)+,#,RPAR
011064  001502                         BEQ     OPRATOR
011066  000167  000556         ERRSX4: JMP     ERRSX5

                                        ,IFNDF  $NOSTR
                                LPSTRNG:CMPB    (R1)+,#,RPAR
                                        BNE     ERRSX4
                                        JMP     SOPRATR
                                        ,ENDC
```

```
 2832
 2833  011072  000332                VARBLE: SWAB    R2
 2834  011074  152102                        BISB    (R1)+,R2
 2835  011076  261502                        ADD     (R5),R2         ;COLLECT OFFSET
 2836  011100  121127  000255                CMPB    (R1),#LPAR
 2837  011104  001412                        BEQ     VARSS
 2838
 2839                                        .IFNDF  $NOSTR
 2840                                        CMP     (R2),#SVAR
 2841                                        BEQ     STRGJMP
 2842                                        .ENDC
 2843
 2844  011106  022227  177775                CMP     (R2)+,#SCALAR
 2845  011112  001463                        BEQ     VARNOSS
 2846  011114  011202                        MOV     (R2),R2
 2847  011116  000461                        BR      VARNOSS
 2848
 2849                                        .IFNDF  $NOSTR
 2850                                STRGJMP:JMP     STRGVAR
 2851                                        .ENDC
 2852
 2853  011120  000004                ERRPDL: IOT
 2854                                        .IFNDF  $LONGER
 2855  011122  052105     103                .ASCII  \ETC\
 2856                                        .ENDC
 2857                                        .IFDF   $LONGER
 2858                                        .ASCII  /EXPRESSION TOO COMPLEX/
 2859                                        .ENDC
 2860  011125     000                     .BYTE 0
 2861                                        .EVEN
 2862  011126  000167  001540        GOTOFN: JMP     FNFN
 2863
 2864                                        .IFNDF  $NOSTR
 2865                                ERRMX2: JMP     ERRMIX
 2866                                        .ENDC
 2867
 2868  011132  005201                VARSS:  INC     R1
 2869  011134  003771                        BLE     ERRPDL
 2870  011136  010246                        MOV     R2,-(SP)
 2871  011140  004767  177452                JSR     PC,EVAL
 2872
 2873                                        .IFNDF  $NOSTR
 2874                                        BCS     ERRMX2
 2875                                        .ENDC
 2876
 2877  011144  004767  002634                JSR     PC,INT
 2878  011150  005765  000040                TST     FAC1(R5)
 2879  011154  001027                        BNE     ERRSS2
 2880  011156  121127  000237                CMPB    (R1),#RPAR
 2881  011162  001426                        BEQ     VONESS
 2882  011164  122127  000243                CMPB    (R1)+,#COMMA
 2883  011170  001336                        BNE     ERRSX4
 2884  011172  016546  000042                MOV     FAC2(R5),-(SP)
 2885  011176  004767  177414                JSR     PC,EVAL
 2886
```



```
 2887                                        .IFNDF  $NOSTR
 2888                                        BCS     ERRMX2
 2889                                        .ENDC
 2890
 2891  011202  004767  002576                JSR     PC,INT
 2892  011206  005765  000040                TST     FAC1(R5)
 2893  011212  001010                        BNE     ERRSS2
 2894  011214  122127  000237                CMPB    (R1)+,#RPAR
 2895  011220  001322                        BNE     ERRSX4
 2896  011222  016503  000042                MOV     FAC2(R5),R3
 2897  011226  100402                        BMI     ERRSS2
 2898  011230  012600                        MOV     (SP)+,R0
 2899  011232  000407                        BR      VTWOSS
 2900  011234  000167  003534        ERRSS2: JMP     ERRSS3
 2901  011240  005201                VONESS: INC     R1
 2902  011242  012703  177777                MOV     #-1,R3
 2903  011246  016500  000042                MOV     FAC2(R5),R0
 2904  011252  100770                VTWOSS: BMI     ERRSS2
 2905  011254  012602                        MOV     (SP)+,R2
 2906
 2907                                        .IFNDF  $NOSTR
 2908                                        CMP     (R2),#SVAR
 2909                                        BNE     .+6
 2910                                        JMP     STRGARR
 2911                                        .ENDC
 2912
 2913  011256  004767  003336                JSR     PC,LOCGET
 2914  011262  012265  000040        VARNOSS:MOV     (R2)+,FAC1(R5)
 2915  011266  011265  000042                MOV     (R2),FAC2(R5)
 2916  011272  111103                OPRATOR:MOVB    (R1),R3         ;NEXT CHARACTER,
 2917  011274  120327  000201                CMPB    R3,#,EOL
 2918  011300  001437                        BEQ     DOITNOW
 2919  011302  120327  000205                CMPB    R3,#,GOTO
 2920  011306  001434                        BEQ     DOITNOW
 2921  011310  120327  000227                CMPB    R3,#,UPARRO
 2922  011314  103664                        BLO     ERRSX4
 2923  011316  011602                        MOV     (SP),R2         ;PREVIOUS OPERATOR,
 2924  011320  120227  000227                CMPB    R2,#,UPARRO
 2925  011324  101425                        BLOS    DOITNOW         ;JUMP IF ^,
 2926  011326  120227  000231                CMPB    R2,#,SLASH
 2927  011332  101417                        BLOS    PREC2           ;JUMP IF * OR /,
 2928  011334  120227  000234                CMPB    R2,#,MINUS
 2929  011340  101411                        BLOS    PREC3           ;JUMP IF + - OR UNARY,
 2930  011342  120227  000254                CMPB    R2,#,EQ
 2931  011346  101403                        BLOS    PREC4           ;JUMP IF = <> <= >= < OR >,
 2932  011350  120327  000254        PREC5:  CMPB    R3,#,EQ
 2933  011354  101427                        BLOS    NOTNOW          ;JUMP IF ANY OPERATOR,
 2934  011356  120327  000234        PREC4:  CMPB    R3,#,MINUS
 2935  011362  101424                        BLOS    NOTNOW          ;JUMP IF + - UNARY * /  OR ^,
 2936  011364  120327  000231        PREC3:  CMPB    R3,#,SLASH
 2937  011370  101421                        BLOS    NOTNOW          ;JUMP IF * / OR ^,
 2938  011372  120327  000227        PREC2:  CMPB    R3,#,UPARRO
 2939  011376  101416                        BLOS    NOTNOW          ;JUMP IF ^,
 2940  011400  005002                DOITNOW:CLR     R2              ;(E,G,  A*B+ ) DO THE * NOW,
 2941  011402  152602                        BISB    (SP)+,R2
```

```
 2942  011404  162702  000226                  SUB     #,UPARRO-1,R2
 2943  011410  060202                          ADD     R2,R2
 2944  011412  060702                          ADD     PC,R2           ;JUMP TO DO THE APPROPRIATE OPERATION,
 2945  011414  061207                          ADD     (R2),PC         ;LEFT OPERAND IS @(SP),2(SP),
 2946  011416  000510                  TABLE2: .WORD   UPARRO-TABLE2   ;RIGHT OPERAND IS FAC1(R5),FAC2(R5),
 2947  011420  000124                          .WORD   STAR-TABLE2
 2948  011422  000154                          .WORD   SLASH-TABLE2
 2949  011424  000116                          .WORD   PLUS-TABLE2
 2950  011426  000056                          .WORD   UNARY-TABLE2
 2951  011430  000052                          .WORD   MINUS-TABLE2
 2952  011432  000042                          .WORD   TERMIN-TABLE2
 2953  011434  016546  000042          NOTNOW: MOV     FAC2(R5),-(SP)  ;(E,G,  A+B* ) DONT DO THE + YET,
 2954  011440  020604                          CMP     SP,R4
 2955  011442  103410                          BLO     ERRPD5
 2956  011444  016546  000040                  MOV     FAC1(R5),-(SP)
 2957  011450  010346                          MOV     R3,-(SP)
 2958  011452  005201                          INC     R1
 2959  011454  000167  177100                  JMP     OPERAND
 2960  011460  000241                  TERMIN: CLC                     ;CARRY OFF MEANS NUMERIC RESULT,
 2961  011462  000207                          RTS     PC
 2962  011464  000167  177430          ERRPD5: JMP     ERRPDL
 2963  011470  004767  005610          MINUS:  JSR     PC,SUBSTK
 2964  011474  005765  000040          UNARY:  TST     FAC1(R5)
 2965  011500  001404                          BEQ     UINTEG
 2966  011502  062765  100000  000040          ADD     #100000,FAC1(R5)
 2967  011510  000670                          BR      OPRATOR
 2968  011512  005465  000042          UINTEG: NEG     FAC2(R5)
 2969  011516  102265                          BVC     OPRATOR
 2970  011520  012765  044000  000040          MOV     #44000,FAC1(R5)
 2971  011526  005065  000042                  CLR     FAC2(R5)
 2972  011532  000657                          BR      OPRATOR
 2973  011534  004767  175630          PLUS:   JSR     PC,ADDSTK
 2974  011540  000654                          BR      OPRATOR
 2975  011542  012767  011606  000000G STAR:   MOV     #ERSTAR,SERVEC
 2976  011550  004467  001500                  JSR     R4,FPPSAV
 2977  011554  011620                          .WORD   TSTSTK          ;TEST TOP OF STACK AND FLOAT IF INT
 2978  011556  011634                          .WORD   PUSHF           ;PUSH FAC AND FLOAT IF NEC
 2979  011560  000000G                         .WORD   $MLR            ;PERFORM MULT
 2980  011562  014270                          .WORD   POP             ;POP RESULT
 2981  011564  013306                          .WORD   FPPRES
 2982  011566  000167  177500          STAR1:  JMP     OPRATOR
 2983  011572  012767  012230  000000G SLASH:  MOV     #ERRDIV,SERVEC
 2984  011600  004467  001530                  JSR     R4,FPPSAV
 2985  011604  011620                          .WORD   TSTSTK
 2986  011606  011634                          .WORD   PUSHF
 2987  011610  000000G                         .WORD   $OVR
 2988  011612  014270                          .WORD   POP
 2989  011614  013306                          .WORD   FPPRES
 2990  011616  000763                          BR      STAR1
 2991  011620  005716                  TSTSTK: TST     (SP)
 2992  011622  001003                          BNE     TSTS2
 2993  011624  005726                  TSTS1:  TST     (SP)+
 2994  011626  000167  000000G                 JMP     $IR
 2995  011632  000134                  TSTS2:  JMP     @(R4)+          ;RETURN
 2996  011634  016546  000042          PUSHF:  MOV     FAC2(R5),-(SP)
```



```
 2997  011640  016546  000040                  MOV     FAC1(R5),-(SP)
 2998  011644  001767                          BEQ     TSTS1           ;FLOAT IF INTEGER
 2999  011646  000134                          JMP     @(R4)+          ;RETURN
 3000                                          .IFDF   $NOSTR
 3001                                  ;DUMMY ENTRY POINTS FOR $NOSTR
 3002                                  SAVCHAR:
 3003                                  ARGB:
 3004                                  MAKEST:
 3005                                  SOPRAT:
 3006                                  STOSVAR:
 3007                                  ERRMIX:
 3008                                  STPRO:
 3009                                          .ENDC
 3010
 3011                                  ERRSYN:
 3012  011650  000004                  ERRSX5: IOT
 3013                                          .IFNDF  $LONGER
 3014  011652  054523     116                 .ASCII  'SYN'
 3015                                          .ENDC
 3016                                          .IFDF   $LONGER
 3017                                          .ASCII  'SYNTAX ERROR'
 3018
 3019                                          .ENDC
 3020  011655     000                         .BYTE 0
 3021                                          .EVEN
 3022
 3023                                          .IFNDF  $NOSTR
 3024                                  QUOTE:  CMPB    (R1)+,#,TEXT
 3025                                          BNE     ERRSX5
 3026                                          MOV     R2,R3           ;SAVE QUOTE CHAR,
 3027                                          MOV     R1,R2
 3028                                          TSTB    (R1)+
 3029                                          BNE     .-2
 3030                                          CMP     SP,R4
 3031                                          BLO     ERRPD5
 3032                                          MOV     R2,-(SP)
 3033                                          SUB     #3,(SP)
 3034                                          MOV     R1,-(SP)
 3035                                          SUB     R2,(SP)
 3036                                          CMPB    (R1)+,R3                ;SAME AS STARTING QUOTE CHAR,?
 3037                                          BNE     ERRSX5
 3038                                          DEC     (SP)
 3039                                          BEQ     QUNULL
 3040                                          MOV     SP,R2
 3041                                          ADD     #2,R2
 3042                                          JSR     PC,MAKESTR
 3043                                          MOV     (SP)+,(SP)
 3044                                          JMP     QUOTBUM
 3045                                  QUNULL: CMP     (SP)+,(SP)+
 3046                                  GOTVAL: MOV     #1,-(SP)
 3047                                          BR      SOPRATR
 3048                                          .ENDC
 3049
 3050  011656  000167  000000          ERRD02: JMP     ERRD02
 3051
```

```
3052                                    ,IFNDF  $NOSTR
3053                            STRGARR:JSR     PC,LOCGET
3054                                    BR      STRGBTH
3055                            STRGVAR:TST     (R2)+
3056                                    CMP     2(R2),#-1
3057                                    BEQ     STRGBTH
3058                                    MOV     (R2),R2
3059                            STRGBTH:MOV     (R2),R3
3060                                    CMP     R3,#-1
3061                                    BEQ     GOTVAL
3062                                    CMP     SP,R4
3063                                    BLO     ERRD02
3064                                    CLR     -(SP)
3065                                    MOVB    (R3),(SP)
3066                                    JSR     PC,MAKESTR
3067                            SOPRATR:CMPB    2(SP),#,AMPERS
3068                                    BEQ     CONCAT
3069                                    CMPB    (R1),#,AMPERS
3070                                    BEQ     AMPWAIT
3071                                    CMPB    2(SP),#,TERM
3072                                    BNE     ERRSX5
3073                                    MOV     (SP)+,R0
3074                                    TST     (SP)+
3075                                    MOV     (SP),R2
3076                                    MOV     R0,(SP)
3077                                    INC     R0
3078                                    BEQ     SOPRX           ;CHECK NULL STRING
3079                                    ADD     #2,R0
3080                                    MOV     SP,R3
3081                                    MOVB    R3,-(R0)
3082                                    SWAB    R3
3083                                    MOVB    R3,-(R0)
3084                            SOPRX:  SEC
3085                                    JMP     (R2)
3086                            AMPWAIT:MOVB    (R1)+,-(SP)
3087                                    JSR     PC,EVAL
3088                                    BCS     SOPRATR
3089                            ERRMIX: IOT
3090                                    ,IFNDF  $LONGER
3091                                    ,ASCII  \NSM\
3092                                    ,ENDC
3093                                    ,IFDF   $LONGER
3094                                    ,ASCII  'NUMBERS AND STRINGS MIXED'
3095                                    ,ENDC
3096                                    ,BYTE 0
3097                                    ,EVEN
3098                            CONCAT: CMP     (SP),#-1
3099                                    BNE     CATNOT
3100                                    CMP     (SP)+,(SP)+
3101                                    BR      SOPRATR
3102                            CATNOT: MOV     4(SP),R2
3103                                    CMP     R2,#-1
3104                                    BNE     CATLONG
3105                                    MOV     (SP)+,R0
3106                                    TST     (SP)+
```



```
3107                                    MOV     R0,(SP)
3108                                    BR      CATCOM
3109                            CATLONG:CLR     R0
3110                                    BISB    (R2),R0
3111                                    MOV     (SP),R3
3112                                    CMP     SP,R4
3113                                    BLO     ERRD02
3114                                    CLR     -(SP)
3115                                    MOVB    (R3),(SP)
3116                                    ADD     R0,(SP)
3117                                    CMP     (SP),#377
3118                                    BHI     ERRSTR
3119                                    MOV     SP,R2
3120                                    ADD     #6,R2
3121                                    JSR     PC,MAKESTR
3122                                    MOV     (SP)+,R3
3123                                    MOV     4(SP),R2
3124                                    CLR     R0
3125                                    BISB    (R2),R0
3126                                    MOV     R3,4(SP)
3127                                    ADD     R0,R3
3128                                    ADD     #3,R3
3129                                    MOV     (SP)+,R2
3130                                    TST     (SP)+
3131                                    CLR     R0
3132                                    BISB    (R2),R0
3133                                    ADD     #3,R2
3134                                    MOVB    (R2)+,(R3)+
3135                                    DEC     R0
3136                                    BGT     .-4
3137                            STPRO:
3138                            QUOTBUM:MOV     (SP),R0
3139                            CATCOM: MOV     SP,R2
3140                                    ADD     #3,R0
3141                                    MOVB    R2,-(R0)
3142                                    SWAB    R2
3143                                    MOVB    R2,-(R0)
3144                                    BR      SOPRATR
3145                                    ,ENDC
3146
3147  011662  000167  177232           ERRPD2: JMP     ERRPDL
3148
3149                                    ,IFNDF  $NOSTR
3150                            ERRSTR: IOT
3151                                    ,IFNDF  $LONGER
3152                                    ,ASCII  \STL\
3153                                    ,ENDC
3154                                    ,IFDF   $LONGER
3155                                    ,ASCII  'STRING TOO LONG'
3156                                    ,ENDC
3157                                    ,BYTE 0
3158                                    ,EVEN
3159                                    ,ENDC
3160
3161                            EREXP:
```

```
                                ERSTAR:
                                ERADD:
011666  000004                 ERROVR: IOT
                                        .IFNDF  $LONGER
011670  053117     106                 .ASCII  /OVF/
                                        .ENDC
                                        .IFDF   $LONGER
                                        .ASCII  /OVERFLOW/
                                        .ENDC
011673     000                          .BYTE 0
                                        .EVEN

011674  012746  000000G         SINFN:  MOV     #SIN,-(SP)
011700  000432                         BR      FUNCT:
011702  012746  000000G         COSFN:  MOV     #COS,-(SP)
011706  000427                         BR      FUNCT:
011710  012746  000000G         SQRFN:  MOV     #SQRT,-(SP)
011714  012767  007734  000000G         MOV     #ERSQR,SERVEC
011722  000421                         BR      FUNCT:
011724  012746  000000G         ATNFN:  MOV     #ATAN,-(SP)
011730  000416                         BR      FUNCT:
011732  012746  0000000         EXPFN:  MOV     #EXP,-(SP)
011736  012767  011666  000000G         MOV     #EREXP,SERVEC
011744  000410                         BR      FUNCT:
011746  012746  000000G         LOGFN:  MOV     #ALOG,-(SP)
011752  012767  007734  000000G         MOV     #ERLOG,SERVEC
011760  000402                         BR      FUNCT:

                                        .IFNDF  $NOSTR
                                ERRMX9: JMP     ERRMIX
                                        .ENDC

011762  000167  177674         ERRD11: JMP     ERRDZ
011766  004767  176624         FUNCT:: JSR     PC,EVAL

                                        .IFNDF  $NOSTR
                                        BCC     FUNCTJ
                                        JMP     ERRARG
                                FUNCTJ:
                                        .ENDC

011772  012603                         MOV     (SP)+,R3
011774  001002                         BNE     .+6
011776  000167  001014                 JMP     ERRUFN
012002  122127  000237                 CMPB    (R1)+,#,RPAR
012006  001045                         BNE     ERRSX9
012010  004467  001320                 JSR     R4,FPPSAV
012014  011634                         .WORD   PUSHF           ;PUSH FAC, FLOAT IF NECESSARY
012016  012024                         .WORD   FUNOK           ;GO DO FUNCTION
012020  013306                         .WORD   FPPRES
012022  000435                         BR      OPRFN
012024  010601          FUNOK:  MOV     SP,R1           ;SAVE ADDRESS OF PUSHED ARG
012026  010446                         MOV     R4,-(SP)        ;SAVE R4
012030  012746  012056                 MOV     #FUNRET,-(SP)   ;DUMMY RETURN ADDRESS
012034  012746  000137                 MOV     #137,-(SP)      ;DUMMY JUMP INSTR,
```



```
012040  010146                         MOV     R1,-(SP)        ;ADDR OF ARGUMENT
012042  012746  000401                 MOV     #401,-(SP)      ;DUMMY BW
012046  010500                         MOV     R5,R0           ;SAVE R5 VALUE
012050  010605                         MOV     SP,R5           ;RETURN ADDRESS (TO DUMMY PROG IN STK)
012052  004070  000052                 JSR     R0,@R5SAVE(R0)
012056  062706  000010         FUNRET: ADD     #8.,SP          ;POP DUMMY PROGRAM
012062  012604                         MOV     (SP)+,R4        ;RESTORE R4
012064  022626                         CMP     (SP)+,(SP)+     ;REMOVE OLD FAC FROM STACK
012066  010065  000040                 MOV     R0,FAC1(R5)     ;STORE RESULT IN FAC
012072  010165  000042                 MOV     R1,FAC2(R5)
012076  000134                         JMP     @(R4)+          ;AND RETURN

                                        .IFNDF  $NOSTR
                                ERRMX8: JMP     ERRMIX
                                        .ENDC

012100  004767  176512         INTFN:  JSR     PC,EVAL

                                        .IFNDF  $NOSTR
                                        BCS     ERRMX8
                                        .ENDC

012104  122127  000237                 CMPB    (R1)+,#,RPAR
012110  001004                         BNE     ERRSX9
012112  004767  001666                 JSR     PC,INT
012116  000167  177150         OPRFN:  JMP     OPRATOR
012122  000167  177522         ERRSX9: JMP     ERRSXS



                                ;       IF A IS NOT 0,FLOAT IT
012126  005716                 UPARRO: TST     (SP)            ;IS UPPER A 0
012130  001010                         BNE     UA1
012132  005766  000002                 TST     2(SP)           ;IS LOWER A 0
012136  001405                         BEQ     UA1
012140  005726                         TST     (SP)+
012142  004467  001166                 JSR     R4,FPPSAV
012146  000000G                         .WORD   $IR
012150  013306                         .WORD   FPPRES

012152  012767  012516  000000G UA1:   MOV     #ERREXP,SERVEC
012160  005765  000040                 TST     FAC1(R5)        ;IS UPPER B = 0
012164  001036                         BNE     UA10
012166  005765  000042                 TST     FAC2(R5)        ;IS LOWER B = 0
012172  001006                         BNE     UA5
012174  012765  000001  000042         MOV     #1,FAC2(R5)     ;SET RESULT 1
012202  022626                 UA4:    CMP     (SP)+,(SP)+
012204  000167  177062                 JMP     OPRATOR
                                ; B IS INTEGER
012210  005716                 UA5:    TST     (SP)            ;IS A = 0
012212  001077                         BNE     UA17
012214  005765  000042                 TST     FAC2(R5)        ;IS LOWER B > 0
012220  100013                         BPL     UA9             ;NO, ERROR MESSAGE
012222  000004                 UA8:    IOT
                                        .IFNDF  $LONGER
```

```
 3272  012224  053104  000          ,ASCII  'DV0'
 3273                               ,ENDC
 3274                               ,IFDF   $LONGER
 3275                               ,ASCII  'DIVISION BY 0'
 3276                               ,ENDC
 3277  012227  000                  ,BYTE   0
 3278                               ,EVEN
 3279  012230  020027  004000  ERRDIV: CMP    R0,#4000
 3280  012234  103372               BHIS    UA8
 3281  012236  000167  177424       JMP     ERSTAK
 3282  012242  103367               BHIS    UA8             ;DIV BY 0
 3283  012244  000167  177416       JMP     ERROVR          ;OVERFLOW
 3284  012250  005065  000040  UA9:  CLR     FAC1(R5)        ;SET RESULT 0
 3285  012254  005065  000042       CLR     FAC2(R5)
 3286  012260  000750               BR      UA4
 3287
 3288                               ;FIX B IF INTEGER < 256
 3289  012262  005765  000042  UA10:  TST     FAC2(R5)
 3290  012266  001036               BNE     UA10,5
 3291  012270  016500  000040       MOV     FAC1(R5),R0
 3292  012274  010002               MOV     R0,R2
 3293  012276  042700  100177       BIC     #100177,R0      ;EXTRACT EXPONENT
 3294  012302  020027  040200       CMP     R0,#40200       ;CHECK TOO SMALL
 3295  012306  103426               BLO     UA10,5
 3296  012310  042702  177600       BIC     #177600,R2
 3297  012314  052702  000200       BIS     #200,R2         ;R2 IS MANTISSA
 3298  012320  020027  042000  UA10A: CMP    R0,#42000
 3299  012324  001406               BEQ     UA10B
 3300  012326  101016               BHI     UA10,5
 3301  012330  006202               ASR     R2
 3302  012332  103414               BCS     UA10,5          ;IF BIT CLEARED,NO GOOD
 3303  012334  062700  000200       ADD     #200,R0         ;UPDATE EXPONENT
 3304  012340  000767               BR      UA10A
 3305  012342  005765  000040  UA10B: TST     FAC1(R5)
 3306  012346  100001               BPL     UA10C
 3307  012350  005402               NEG     R2
 3308  012352  005065  000040  UA10C: CLR     FAC1(R5)
 3309  012356  010265  000042       MOV     R2,FAC2(R5)
 3310  012362  000712               BR      UA5
 3311                               ;B IS REAL
 3312  012364  005716  UA10,5: TST     (SP)            ;IS A = 0
 3313  012366  001405               BEQ     UA12
 3314  012370  004467  000740       JSR     R4,FPPSAV
 3315  012374  014256               ,WORD   PUSH
 3316  012376  012536  012600       ,WORD   UAJMP,UA23
 3317
 3318  012402  005765  000040  UA12:  TST     FAC1(R5)        ;IS UPPER B> 0
 3319  012406  100320               BPL     UA9             ;YES
 3320  012410  000704               BR      UA8             ;NO, ERROR
 3321
 3322                               ;B INTEGER, A REAL
 3323  012412  016500  000042  UA17:  MOV     FAC2(R5),R0
 3324  012416  010002               MOV     R0,R2           ;SAVE FOR SIGN TEST
 3325  012420  002001               BGE     UA17A
 3326  012422  005400               NEG     R0
```



```
 3327  012424  020027  000256  UA17A: CMP     R0,#256         ;R0 IS ABS(B) IF TOO BIG, FLOAT
 3328  012430  103057               BHIS    UA20
 3329  012432  004467  000676       JSR     R4,FPPSAV       ;GO INTO POLISH MODE
 3330  012436  014270               ,WORD   POP             ;CURRENT PRODUCT IS A
 3331  012440  014302               ,WORD   PUSH1           ;CURRENT ANSWER IS 1 (ON STK)
 3332  012442  012524  012452  UA17B: ,WORD   UAASR,UA17D     ;SHIFT B, BRANCH IF BIT IS 0
 3333  012446  014256               ,WORD   PUSH            ;GET CURRENT PRODUCT ONT OSTACK
 3334  012450  000000G              ,WORD   $MLR            ;MULTIPLY INTO CURRENT ANSWER
 3335  012452  012542  012472  UA17D: ,WORD   UATST0,UA17F    ;TEST IF DONE, BRANCH IF SO
 3336  012456  014256  014256       ,WORD   PUSH,PUSH       ;TWO COPIES OF CURRENT PRODUCT
 3337  012462  000000G              ,WORD   $MLR            ;SQUARE IT
 3338  012464  014270               ,WORD   POP             ;AND PUT RESULT BACK
 3339  012466  012536  012442       ,WORD   UAJMP,UA17B     ;POLISH JUMP TO 17B
 3340  012472  014270  UA17F: ,WORD   POP             ;STORE ANSWER INTO FAC
 3341  012474  012552  012510       ,WORD   UATST2,UA19     ;TEST SIGN B, IF + JUMP
 3342  012500  014302               ,WORD   PUSH1           ;PUSH 1 ONTO STACK
 3343  012502  014256               ,WORD   PUSH
 3344  012504  000000G              ,WORD   $DVR            ;ANS = 1/ANS IF B NEG
 3345  012506  014270               ,WORD   POP     ;POP ANSWER TO FAC
 3346  012510  013306  UA19:  ,WORD   FPPRES
 3347  012512  000167  176554       JMP     OPRATOR         ;AND GET OUT
 3348
 3349                               ERREXP:
 3350  012516  000004  UA21:  IOT
 3351                               ,IFNDF  $LONGER
 3352  012520  042536     122       ,ASCII  \?ER\
 3353                               ,ENDC
 3354                               ,IFDF   $LONGER
 3355                               ,ASCII  /? ERROR/
 3356                               ,ENDC
 3357  012523  000                  ,BYTE   0
 3358                               ,EVEN
 3359
 3360  012524  006265  000044  UAASR:  ASR     R0SAVE(R5)
 3361  012530  103002               BCC     UAJMP
 3362  012532  005724  UANJMP: TST     (R4)+           ;SKIP OVER JUMP ADDRESS
 3363  012534  000134               JMP     @(R4)+
 3364  012536  011404  UAJMP:  MOV     @R4,R4          ;POLISH MODE JUMP
 3365  012540  000134               JMP     @(R4)+
 3366  012542  005765  000044  UATST0: TST     R0SAVE  (R5)
 3367  012546  001773               BEQ     UAJMP
 3368  012550  000770               BR      UANJMP
 3369  012552  005765  000050  UATST2: TST     R2SAVE(R5)      ;TEST SIGN OF B
 3370  012556  100367               BPL     UAJMP           ;POLISH JUMP IF B +
 3371  012560  000764               BR      UANJMP
 3372  012562  005716  UATSTA: TST     @SP
 3373  012564  100754               BMI     UA21
 3374  012566  000134               JMP     @(R4)+
 3375
 3376  012570  010246  UA20:  MOV     R2,-(SP)        ;PUSH INT B ON STACK
 3377  012572  004467  000536       JSR     R4,FPPSAV
 3378  012576  000000G              ,WORD   $IR             ;FLOAT B
 3379                               UA23:
 3380  012600  012622               ,WORD   REVRSE          ;REVERSE TOP TWO ON STK
 3381  012602  012562               ,WORD   UATSTA          ;TEST TOP STK, BR TO ERR IF -
```

```
3382  012604  012644                          ,WORD   CALOG           ;CALL ALOG FUNCTION ON A, RESULT TO FAC
3383  012606  014256                          ,WORD   PUSH            ;PUSH LOG(A) ONTO STACK
3384  012610  000000G                         ,WORD   $ALR            ;CALC B*LOG(A)
3385  012612  012656                          ,WORD   CEXP            ;CALC EXP(B*LOG(A)), RESULT TO FAC
3386  012614  013306                          ,WORD   FPPRES
3387  012616  000167  176450                  JMP     OPRATOR
3388  012622  012620                  REVRSE: MOV     (SP)+,R0        ;ROUTINE TO REVERSE TOP 2
3389  012624  012601                          MOV     (SP)+,R1        ;FLTPT NUMBERS ON STACK
3390  012626  012602                          MOV     (SP)+,R2
3391  012630  012603                          MOV     (SP)+,R3
3392  012632  010146                          MOV     R1,-(SP)
3393  012634  010046                          MOV     R0,-(SP)
3394  012636  010346                          MOV     R3,-(SP)
3395  012640  010246                          MOV     R2,-(SP)
3396  012642  000134                          JMP     @(R4)+
3397  012644  012765  000000G 000052  CALOG:  MOV     #ALOG,R3SAVE(R5)
3398  012652  000167  177146                  JMP     FUNOK
3399  012656  012765  000000G 000052  CEXP:   MOV     #EXP,R3SAVE(R5)
3400  012664  000167  177134                  JMP     FUNOK
3401  012670  000167  176224          ERRPD4: JMP     ERRPDL
3402
3403                                          ;.IFNDF $NOSTR
3404                                  ERRMX1: JMP     ERRMIX
3405                                          ;.ENDC
3406
3407  012674  000167  176750          ERRSXA: JMP     ERRSX%
3408  012700  112102                  FNFN:   MOVB    (R1)+,R2
3409  012702  122127  000255                  CMPB    (R1)+,#,LPAR
3410  012706  001372                          BNE     ERRSXA
3411  012710  066502  000004                  ADD     PDL(R5),R2
3412  012714  020604                          CMP     SP,R4
3413  012716  103764                          BLO     ERRPD4
3414  012720  011200                          MOV     (R2),R0         ;R0 NOW CONTAINS THE DEF POINTER,
3415  012722  001435                          BEQ     ERRUFN
3416  012724  010046                  FNSWAP: MOV     R0,-(SP)
3417  012726  004767  175664                  JSR     PC,EVAL
3418
3419                                          ;.IFNDF $NOSTR
3420                                          BCS     FNSTR
3421                                          ;.ENDC
3422
3423  012732  012600                          MOV     (SP)+,R0                ;RESTORE THE DEF POINTER,
3424  012734  112002                          MOVB    (R0)+,R2                ;GET THE VARIABLE FROM THE DEF,
3425  012736  100425                          BMI     ERRAG1
3426  012740  000302                          SWAB    R2
3427  012742  152002                          BISB    (R0)+,R2
3428  012744  061502                          ADD     (R5),R2
3429
3430                                          ;.IFNDF $NOSTR
3431                                          CMP     (R2),#,SVAR     ;IF THE DEF VARIABLE IS STRING AND
3432                                          BEQ     ERRMX1          ;THE VALUE ISNT THEN ITS AN ERROR,
3433                                          ;.ENDC
3434
3435  012746  022227  177775                  CMP     (R2)+,#,SCALAR
3436  012752  001401                          BEQ     .+4
```



```
3437  012754  011202                          MOV     (R2),R2
3438  012756  020604                          CMP     SP,R4
3439  012760  103743                          BLO     ERRPD4
3440  012762  011246                          MOV     (R2),-(SP)      ;STACK THE OLD VALUE AND REPLACE IT
3441  012764  016522  000040                  MOV     FAC1(R5),(R2)+  ;WITH THE NEW VALUE
3442  012770  011246                          MOV     (R2),-(SP)
3443  012772  016512  000042                  MOV     FAC2(R5),(R2)
3444  012776  000400                          BR      FNREPL
3445
3446
3447                                  FNSTR:  MOV     2(SP),R0                ;RESTORE THE DEF POINTER,
3448                                          MOVB    (R0)+,R2                ;GET THE VARIABLE FROM THE DEF,
3449                                          BMI     ERRAG1
3450                                          SWAB    R2
3451                                          BISB    (R0)+,R2
3452                                          ADD     (R5),R2
3453                                          CMP     (R2)+,#,SVAR    ;IF THE DEF VARIABLE IS NUMERIC AND
3454                                          BNE     ERRMX1          ;THE VALUE ISNT THEN ITS AN ERROR,
3455                                          CMP     2(R2),#-1       ;IF THE VAR ISNT A STRING SCALAR
3456                                          BEQ     .+4
3457                                          MOV     (R2),R2         ;THEN DO THIS,
3458                                          MOV     (R2),R3         ;R3 NOW CONTAINS THE OLD VALUE,
3459                                          MOV     (SP)+,(R2)      ;0(R2) NOW CONTAINS THE NEW VALUE,
3460                                          MOV     R3,(SP)         ;0(SP) NOW CONTAINS THE OLD VALUE,
3461                                          MOV     R2,-(SP)                ;TOP(STK) IS A TEMP FOR VAR ADDRESS,
3462                                          MOV     (R2),R2         ;R2 NOW CONTAINS THE NEW VALUE,
3463                                          INC     R2
3464                                          BEQ     FNNEWNL         ;BRANCH IF THE NEW VALUE IS NULL,
3465                                          INC     R3
3466                                          BNE     FNOLNN          ;BRANCH IF THE OLD VALUE IS NONNULL,
3467                                          MOVB    1(SP),(R2)+     ;OLD VALUE IS NULL, NEW IS NOT; SO:
3468                                          MOVB    (SP)+,(R2)      ;MAKE AN ABS BACKPTR TO THE VAR,
3469                                          BR      FNREPL          ;(THE NEW VALUE IS NOW PROTECTED)
3470                                  FNOLNN: MOVB    (R3)+,(R2)+     ;OLD AND NEW ARE BOTH NONNULL SO:
3471                                          MOVB    (R3),(R2)       ;PUT OLD BACKPTR INTO THE NEW VALUE,
3472                                          SUB     #2,R3
3473                                  FNNEWNL:TST     (SP)+           ;GET RID OF THE TEMP,
3474                                          MOV     SP,R2
3475                                          INC     R3
3476                                          BEQ     FNREPL          ;    (CHECK NULL STRING!)
3477                                          SWAB    R2              ;SET THE BACKPTR OF THE OLD VALUE
3478                                          MOVB    R2,(R3)+        ;TO ITS CURRENT STACK ADDRESS,
3479                                          SWAB    R2
3480                                          MOVB    R2,(R3)
3481                                          ;.ENDC
3482
3483  013000  121027  000243          FNREPL: CMPB    (R0),#,COMMA
3484  013004  001007                          BNE     FNNOCOM
3485  013006  122021                          CMPB    (R0)+,(R1)+
3486  013010  001745                          BEQ     FNSWAP
3487  013012  000167  174716          ERRAG1: JMP     ERRARG
3488  013016  000004                  ERRUFN: IOT
3489                                          ;.IFNDF $LONGER
3490  013020  043125      116                 ;.ASCII 'UFN'
3491                                          ;.ENDC
```

```
                                        ,IFDF   $LONGER
                                        ,ASCII  'UNDEFINED FUNCTION'
                                        ,ENDC
013023     000                          ,BYTE 0
                                        ,EVEN
013024  121027  000237          FNNOCOM:CMPB    (R0),#,RPAR
013030  001370                          BNE     ERRAG1
013032  010046                          MOV     R0,-(SP)
013034  122021                          CMPB    (R0)+,(R1)+
013036  001365                          BNE     ERRAG1
013040  122027  000254                  CMPB    (R0)+,#,EQ
013044  001362                          BNE     ERRAG1
013046  010146                          MOV     R1,-(SP)
013050  010001                          MOV     R0,R1
013052  004767  175540                  JSR     PC,EVAL
013056  103402                          BCS     .+6
013060  005003                          CLR     R3
013062  000401                          BR      .+4
013064  012603                          MOV     (SP)+,R3
013066  121127  000201                  CMPB    (R1),#,EOL
013072  001300                          BNE     ERRSXA
013074  012601                          MOV     (SP)+,R1
013076  012600                          MOV     (SP)+,R0
013100  114046          FNRLUP: MOVB    -(R0),-(SP)
013102  114066  000001                  MOVB    -(R0),1(SP)
013106  012602                          MOV     (SP)+,R2
013110  061502                          ADD     (R5),R2

                                        ,IFNDF  $NOSTR
                                        CMP     (R2),#,SVAR
                                        BEQ     FNRSTR
                                        ,ENDC

013112  022227  177775                  CMP     (R2)+,#,SCALAR
013116  001401                          BEQ     .+4
013120  011202                          MOV     (R2),R2
013122  012662  000002                  MOV     (SP)+,2(R2)
013126  012612                          MOV     (SP)+,(R2)
013130  000414                          BR      FNCHK
013132  012612          FNRSSC: MOV     (SP)+,(R2)
013134  010046                          MOV     R0,-(SP)
013136  011200                          MOV     (R2),R0
013140  161502                          SUB     (R5),R2
013142  005202                          INC     R2

                                        ,IFNDF  $NOSTR
                                        BR      FNRCOM
                                FNRSTR: TST     (R2)+
                                        CMP     2(R2),#-1
                                        BEQ     FNRSSC
                                        MOV     (R2),R2
                                        MOV     (SP)+,(R2)
                                        MOV     R0,-(SP)
                                        MOV     (R2),R0
                                        ,ENDC
```



```
013144  005200          FNRCOM: INC     R0
013146  001404                  BEQ     FNRSNUL
013150  000302                  SWAB    R2
013152  110220                  MOVB    R2,(R0)+
013154  000302                  SWAB    R2
013156  110220                  MOVB    R2,(R0)+
013160  012600          FNRSNUL:MOV     (SP)+,R0
013162  124027  000255  FNCHK:  CMPB    -(R0),#,LPAR
013166  001344                  BNE     FNRLUP

                                ,IFNDF  $NOSTR
                                TST     R3
                                BEQ     FNNUMER
                                MOV     R3,-(SP)
                                CMP     R3,#177777      ;CHECK NULL STRING
                                BNE     .+6
                                JMP     $OPRATR
                                JMP     STPRO
                                ,ENDC

013170  000167  176076  FNNUMER:JMP     OPRATOR
                                ,EOT
```

```
                                        ;                                SOURCE FILE #6
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                        ;  SUBROUTINE 'FIXUP'   CALLED BY JSR PC
                                        ;                       MATCHES TYPES OF @(SP),2(SP) AND FAC1(R5),FAC2(R5)
                                        ;                       R0,R1  UNUSED
                                        ;                       R2  DESTROYED
                                        ;                       R3,R4  UNUSED
                                        ;                       R5  MUST POINT TO USER AREA
                                        ;                       SP  GOES NO DEEPER AFTER JSR
013174  012602                  FIXUP:  MOV     (SP)+,R2
013176  005716                          TST     (SP)
013200  001413                          BEQ     FIXTST
013202  005765  000040                  TST     FAC1(R5)
013206  001021                          BNE     FIXRET          ;COND CODES=NONZERO MEANS FLOATING,
013210  016546  000042                  MOV     FAC2(R5),@(SP)          ;****
013214  004467  000114                  JSR     R4,FPPSAV
013220  000000G                         ,WORD   SIR
013222  014270                          ,WORD   POP
013224  013306                          ,WORD   FPPRES
013226  000410                          BR      FIXCLE
013230  005765  000040          FIXTST: TST     FAC1(R5)
013234  001406                          BEQ     FIXRET          ;COND CODES=0 MEANS INTEGER,
013236  005726                          TST     (SP)+           ;****
013240  004467  000070                  JSR     R4,FPPSAV
013244  000000G                         ,WORD   SIR
013246  013306                          ,WORD   FPPRES
013250  000244                  FIXCLE: CLZ                     ;COND CODES=NONZERO MEANS FLOATING,
013252  000112                  FIXRET: JMP     (R2)
```



```
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                        ;  SUBROUTINE 'FLINE'   CALLED BY JSR PC
                                        ;                       FINDS THE ADDRESS OF THE LINE NO
                                        ;                       REFERENCED BY (R1) IN R2,
                                        ;                       IF THE SYMBOL TABLE REFERENCE IS NOT A LINE NO,
                                        ;                       SETS CARRY, OTHERWISE, THE CARRY IS CLEAR,
013254  112102                  FLINE:  MOVB    (R1)+,R2
013256  100407                          BMI     FLE
013260  000302                          SWAB    R2
013262  152102                          BISB    (R1)+,R2
013264  061502                          ADD     (R5),R2
013266  012200                          MOV     (R2)+,R0
013270  020027  177775                  CMP     R0,#,SCALAR
013274  103402                          BLO     FLX
013276  000261                  FLE:    SEC
013300  000207                          RTS     PC
013302  000241                  FLX:    CLC
013304  000207                          RTS     PC
```

```
                                   ;--------------------------------------------------------
                                   ;  ROUTINE 'FPPRES'      CALLED IN POLISH MODE
                                   ;                        RESTORES R0 THRU R4 SAVED BY FPPSAV
                                   ;                        AND RETURNS IN NORMAL EXEC MODE
013306  016500  000044             FPPRES: MOV     R0SAVE(R5),R0
013312  016501  000046                     MOV     R1SAVE(R5),R1
013316  016502  000050                     MOV     R2SAVE(R5),R2
013322  016503  000052                     MOV     R3SAVE(R5),R3
013326  016546  000054                     MOV     R4SAVE(R5),-(SP)
013332  000204                             RTS     R4


                                   ;----------------------------------------------------------
                                   ;  ROUTINE 'FPPSAV'      CALLED BY JSR R4
                                   ;                        SAVES R0 THRU R4, AND RETURNS IN
                                   ;                        POLISH MODE
013334  010065  000044             FPPSAV: MOV     R0,R0SAVE(R5)
013340  010165  000046                     MOV     R1,R1SAVE(R5)
013344  010265  000050                     MOV     R2,R2SAVE(R5)
013350  010365  000052                     MOV     R3,R3SAVE(R5)
013354  012665  000054                     MOV     (SP)+,R4SAVE(R5)
013360  000134                             JMP     @(R4)+          ;ENTER POLISH MODE

```



```
                                   ;
                                   ; GET1BYT (GET2BYT) - GET BYTE OUT OF RING BUFFER USING
                                   ;  GET POINTER 1 (R2).
                                   ;
                                   ; CALL: #[BUFF. HDR. BASE],R1
                                   ;        #[BASE OFFSET OF GET PTR.],R3
                                   ;        JSR     PC,GETBYT
                                   ;
                                   ;  USES R0,R1,R2,R3
                                   ;
                                   ;  RETURN CHAR. IN R0.
                                   ;  'C' BIT IS: CLR IF CHAR. OBTAINED
                                   ;              SET IF BUFF. WAS EMPTY (NO CHAR.)
                                   ;
013362  012703  000004             GET1BYT:MOV     #BGET1,R3
013366  000402                             BR      GETBYT
013370  012703  000006             GET2BYT:MOV     #BGET2,R3
013374  012702  177776             GETBYT: MOV     #PS,R2          ;STATUS
013400  011246                             MOV     (R2),-(SP)
013402  012712  000340                     MOV     #PR7,(R2)       ;HIGHEST PRIORITY
013406  060103                             ADD     R1,R3
013410  021361  000010                     CMP     (R3),BPUT(R1)   ;ANYTHING IN BUFF?
013414  001412                             BEQ     BUFEMP
013416  113300                             MOVB    @(R3)+,R0       ;GET BYTE
013420  005243                             INC     -(R3)           ;PT. TO NEXT BYTE
013422  021361  000002                     CMP     (R3),BEND(R1)   ;WRAP AROUND?
013426  101402                             BLOS    NOWRP
013430  016113  000000                     MOV     BSTRT(R1),(R3)  ;PT. TO NEXT BYTE TO GET
013434  012612                     NOWRP:  MOV     (SP)+,(R2)      ;STATUS BACK
013436  000241                             CLC                     ;SIGNAL SUCCESS
013440  000207                             RTS     PC
013442  012612                     BUFEMP: MOV     (SP)+,(R2)      ;STATUS BACK
013444  000261                             SEC                     ;SHOW FAILURE
013446  000207                             RTS     PC

```

```
                                        ;-----------------------------------------------------
                                        ;
                                        ; GETCHAR - RETURN A CHAR, IN R0 OR WAIT UNTIL YOU CAN
                                        ;
                                        ; CALL: #[BUFF, HDR, BASE],R1
                                        ;       JSR     PC,GETCHAR
                                        ;
                                        ;  USES R3, (R0,R1,R2)
                                        ;
                                        ;  RETURNS CHAR, IN R0
                                        ;  IF PAPER TAPE READER EOT FOUND, EXIT TO 'READY'.
                                        ;
013450  010246                          GETCHAR:MOV     R2,-(SP)
013452  004767  177704                  GETCH1: JSR     PC,GET1BYT
013456  103405                                  BCS     NOCHAR          ;NOTHING THERE YET
013460  016102  000012                          MOV     BFSPEC(R1),R2   ;CHECK IF READER STOPPED
013464  003011                                  BGT     XCHAR           ;0 OR + IS OK, BRANCH IF CHAR
013466  012602                          GETX:   MOV     (SP)+,R2
013470  000207                                  RTS     PC
013472  016100  000012                  NOCHAR: MOV     BFSPEC(R1),R0   ;CHECK EOF OR EXTRA CHAR
013476  001423                                  BEQ     REENAB
013500  005061  000012                          CLR     BFSPEC(R1)
013504  000261                                  SEC
013506  000767                                  BR      GETX
013510  010046                          XCHAR:  MOV     R0,-(SP)        ;PUSH OLD CHAR
013512  010200                                  MOV     R2,R0           ;POSITION NEXT
013514  004767  003064                          JSR     PC,PUTBYT
013520  103001                                  BCC     .+4             ;MAKE SURE IT GOT IN!
013522  000000                                  HALT
013524  012600                                  MOV     (SP)+,R0        ;GET BACK OLD CHAR
013526  005061  000012                          CLR     BFSPEC(R1)
013532  116502  000077                          MOVB    IDEV(R5),R2
013536  012772  000101  013600                  MOV     #101,@ITAB(R2)  ;RE-ENABLE INPUT DEVICE
013544  000750                                  BR      GETX            ; (CARRY IS CLEAR)
                                        ;       RE-ENABLE FROM A DEAD START
013546  116502  000077                  REENAB: MOVB    IDEV(R5),R2
013552  032772  004000  013600                  BIT     #004000,@ITAB(R2)       ;CHECK BUSY BIT
013560  001003                                  BNE     GCWAIT
013562  012772  000101  013600                  MOV     #101,@ITAB(R2)
013570  000001                          GCWAIT: WAIT    .               ;WAIT FOR AN INTERRUPT
013572  004767  000514                          JSR     PC,IOWAIT       ;WASTE TIME
013576  000725                                  BR      GETCH1

                                        ;       TABLE OF INPUT DEVICE STATUS REGISTERS
013600  177560                          ITAB:   .WORD   TKS
013602  177550                                  .WORD   PRS
```



```
                                        ;-----------------------------------------------------
                                        ; 'GETVAR' SUBROUTINE
                                        ;               THIS SUBROUTINE READS A VARIABLE NAME OR
                                        ;               AN ARRAY ELEMENT, AND ADDRESSES THAT
                                        ;               VARIABLE IN CORE, SO THAT A SUBSEQUENT 'STOVAR'
                                        ;               CALL WILL STORE THE FAC IN THE VARIABLE.
                                        ;               DESTROYS FAC1 AND FAC2.
                                        ;               CALLED BY JSR PC,GETVAR
                                        ;                       R1 POINTS TO THE CODE NAMING THE VARIABLE
                                        ;
013604  010265  000022                  GETVAR: MOV     R2,VARSAV(R5)
013610  012702  177777                          MOV     #-1,R2
013614  010265  000024                          MOV     R2,SS1SAV(R5)
013620  010265  000026                          MOV     R2,SS2SAV(R5)
013624  121127  000255                          CMPB    (R1),#.LPAR
013630  001041                                  BNE     NOSUBS
013632  005201                                  INC     R1
013634  004767  174756                          JSR     PC,EVAL

                                                .IFNDF  $NOSTR
                                                BCS     ERRMX3
                                                .ENDC

013640  004767  000140                          JSR     PC,INT
013644  005765  000040                          TST     FAC1(R5)
013650  001034                                  BNE     ERRSS1
013652  016565  000042  000024                  MOV     FAC2(R5),SS1SAV(R5)
013660  100430                                  BMI     ERRSS1
013662  112102                                  MOVB    (R1)+,R2
013664  120227  000237                          CMPB    R2,#.RPAR
013670  001421                                  BEQ     NOSUBS
013672  120227  000243                          CMPB    R2,#.COMMA
013676  001017                                  BNE     ERRSX3
013700  004767  174712                          JSR     PC,EVAL

                                                .IFNDF  $NOSTR
                                                BCS     ERRMX3
                                                .ENDC

013704  004767  000074                          JSR     PC,INT
013710  005765  000040                          TST     FAC1(R5)
013714  001012                                  BNE     ERRSS1
013716  016565  000042  000026                  MOV     FAC2(R5),SS2SAV(R5)
013724  100406                                  BMI     ERRSS1
013726  122127  000237                          CMPB    (R1)+,#.RPAR
013732  001001                                  BNE     ERRSX3
013734  000207                          NOSUBS: RTS     PC
013736  000167  175706                  ERRSX3: JMP     ERRSX5

                                                .IFNDF  $NOSTR
                                        ERRMX3: JMP     ERRMIX
                                                .ENDC

013742  000167  175266                  ERRSS1: JMP     ERRSS2
```

```
3779                                    ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
3780                                    ;
3781                                    ; INITBF - INIT, BUFFER HDRS,
3782                                    ;
3783                                    ; CALL: MOV     #[HDR, ADDR,],R2
3784                                    ;       JSR     PC,INITBF
3785                                    ;
3786                                    ;  USES R2,R3
3787                                    ;
3788                                    ;  LEAVES R3 POINTING TO WORD AFTER LAST IN BUFF, HDR,
3789                                    ;
3790  013746  012203                    INITBF: MOV     (R2)+,R3        ;BUFF, START ADDR,
3791  013750  005722                            TST     (R2)+           ;PT, TO BGET1
3792  013752  010322                            MOV     R3,(R2)+
3793  013754  005722                            TST     (R2)+           ;PT, TO BPUT
3794  013756  010322                            MOV     R3,(R2)+
3795  013760  005022                            CLR     (R2)+           ;BFSPEC
3796  013762  000207                            RTS     PC
3797
3798
3799
3800
3801                                    ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
3802                                    ;
3803                                    ; INITSCR - SCRATCH USER AREA
3804                                    ;
3805                                    ; CALL: JSR     PC,INITSCR
3806                                    ;
3807                                    ;  USES R0
3808                                    ;
3809  013764  016500  000016            INITSCR:MOV     CODE(R5),R0
3810  013770  012720  000225                    MOV     #,EOF;(R0)+
3811  013774  010015                            MOV     R0,(R5)
3812  013776  012065  000014                    MOV     R0,LOFREE(R5)
3813  014002  000207                            RTS     PC
3814
```



```
3815                                    ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
3816                                    ;  SUBROUTINE FINT:     CALLED BY JSR PC
3817                                    ;                       COMPUTES INT(FAC), CONVERTING THE RESULT
3818                                    ;                       TO A 1-WORD INTEGER, IF POSSIBLE
3819  014004  016500  000040            INT:    MOV     FAC1(R5),R0     ;GET HIGH ORDER WORD IN R0
3820  014010  001452                            BEQ     INTRTS          ;ALREADY AN INTEGER
3821  014012  004567  003044                    JSR     R5,SAVREG       ;SAVE REGISTERS
3822  014016  010001                            MOV     R0,R1
3823  014020  042700  100177                    BIC     #100177,R0      ;EXTRACT EXPONENT
3824  014024  020027  040200                    CMP     R0,#40200       ;CHECK ABS VALUE < 1
3825  014030  103473                            BLO     INT7
3826  014032  020027  046000                    CMP     R0,#46000       ;CHECK INTEGER BY TRUNCATION
3827  014036  101037                            BHI     INTRTS          ;YES, RETURN
3828  014040  005002                            CLR     R2
3829  014042  162700  043600                    SUB     #43600,R0       ;CHECK MORE THAN 15 BITS INTEGER
3830  014046  003034                            BGT     INT5            ;YES, PRODUCE FLT INT ANSWER
3831  014050  000301                            SWAB    R1
3832  014052  105001                            CLRB    R1
3833  014054  156501  000043                    BISB    FAC2+1(R5),R1   ;R1 IS 16 BITS OF MANTISSA
3834  014060  052701  100000                    BIS     #100000,R1      ;SET HIDDEN BIT
3835  014064  000241                            CLC
3836  014066  006001                            ROR     R1              ;GET 15 BITS OF ABSOLUTE VALUE
3837  014070  005700                            TST     R0              ;TEST ALREADY INTEGER
3838  014072  002005                            BGE     INT1A           ;YES
3839  014074  006201            INT1:   ASR     R1              ;SHIFT MANTISSA 1 RIGHT
3840  014076  005502                            ADC     R2              ;ADD CARRY BIT TO R2
3841  014100  062700  000200                    ADD     #200,R0         ;INCREMENT EXPONENT
3842  014104  002773                            BLT     INT1            ;LOOP UNTIL DONE
3843  014106  005765  000040            INT1A:  TST     FAC1(R5)        ;ORIGINAL NUMBER NEG
3844  014112  100005                            BPL     INT2            ;NO
3845  014114  005702                            TST     R2              ;ANY BITS ZEROED?
3846  014116  001402                            BEQ     INT1B
3847  014120  005201                            INC     R1              ;ADJUST FOR NEGATIVE
3848  014122  102447                            BVS     INT9            ;OVERFLOW
3849  014124  005401            INT1B:  NEG     R1
3850  014126  010165  000042    INT2:   MOV     R1,FAC2(R5)
3851  014132  005065  000040    INT2A:  CLR     FAC1(R5)
3852  014136  000207            INTRTS: RTS     PC
3853
3854  014140  020027  002200    INT5:   CMP     R0,#2200        ;CHECK DONE
3855  014144  002005                            BGE     INT6            ;YES
3856  014146  000261                            SEC                     ;SET CARRY BIT
3857  014150  006102                            ROL     R2              ;CREATE PATTERN OF BITS TO CLEAR
3858  014152  062700  000200                    ADD     #200,R0
3859  014156  000770                            BR      INT5
3860  014160  016500  000042    INT6:   MOV     FAC2(R5),R0     ;SAVE OLD FAC2 IN CASE NEG ARG
3861  014164  040265  000042                    BIC     R2,FAC2(R5)     ;CLEAR BITS
3862  014170  005765  000040                    TST     FAC1(R5)        ;CHECK NEG ARG
3863  014174  100360                            BPL     INTRTS          ;NO, DONE
3864  014176  030200                            BIT     R2,R0           ;CHECK EXACT INTEGER ALREADY
3865  014200  001756                            BEQ     INTRTS          ;YES, RETURN
3866  014202  004467  000000G                   JSR     R4,$PULSH       ;NO, MUST SUBTRACT 1
3867  014206  014256                            ,WORD   PUSH            ;PUSH FAC
3868  014210  014302                            ,WORD   PUSH1           ;PUSH FLOATING 1
3869  014212  000000G                           ,WORD   $SUBR           ;SUBTRACT
```

```
3870  014214  014270                          ,WORD   POP             ;POP ANSWER TO FAC
3871  014216  014136                          ;WORD   INTRTS          ;AND RETURN (OUT OF POLISH MODE)
3872  014220  005701                  INT7:   TST     R1              ;ABS(ARG) < 1
3873  014222  100403                          BMI     INT8
3874  014224  005065  000042                  CLR     FAC2(R5)        ;IF + THEN ANS IS 0
3875  014230  000740                          BR      INT2A
3876  014232  012765  177777  000042  INT8:   MOV     #177777,FAC2(R5) ;IF - THEN ANS IS -1
3877  014240  000734                          BR      INT2A
3878  014242  012765  143000  000040  INT9:   MOV     #143000,FAC1(R5)
3879  014250  005065  000042                  CLR     FAC2(R5)
3880  014254  000730                          BR      INTRTS
3881
3882  014256  016546  000042          PUSH:   MOV     FAC2(R5),-(SP)
3883  014262  016546  000040                  MOV     FAC1(R5),-(SP)
3884  014266  000134                          JMP     @(R4)+
3885  014270  012665  000040          POP:    MOV     (SP)+,FAC1(R5)
3886  014274  012665  000042                  MOV     (SP)+,FAC2(R5)
3887  014300  000134                          JMP     @(R4)+
3888  014302  005046                  PUSH1:  CLR     -(SP)
3889  014304  012746  040200                  MOV     #40200,-(SP)
3890  014310  000134                          JMP     @(R4)+
3891
3892
3893
3894
3895
3896                                  ;-----------------------------------------------------
3897                                  ;
3898                                  ; IOWAIT - WASTE TIME TILL NEXT INTERRUPT
3899                                  ;
3900                                  ; CALL: JSR     PC,IOWAIT
3901                                  ;
3902  014312  005265  000070          IOWAIT: INC     RNDCT(R5)
3903  014316  105765  000106                  TSTB    CNCFLG(R5)
3904  014322  001402                          BEQ     WTRET
3905  014324  000167  164546                  JMP     READY
3906  014330  005737  000050          WTRET:  TST     @#50            ;CHECK WAIT LOOP ADDR
3907  014334  001402                          BEQ     .+6
3908  014336  000137  000050                  JMP     @#50            ;SPECIAL WAIT ROUTINE
3909  014342  000207                          RTS     PC
3910
```



```
3911                                  ;-----------------------------------------------------
3912                                  ;
3913                                  ; LINGET - EDIT A LINE INTO LINE BUFFER USING GETCHAR
3914                                  ;
3915                                  ; CALL: USER AREA BASE INTO R5
3916                                  ;       JSR     PC,LINGET
3917                                  ;
3918                                  ;  PRESERVES ALL REGISTERS
3919                                  ;
3920                                  ;  RECOGNIZES '←' AND <RUB OUT> FOR RUB OUT AND <ALT MODE>
3921                                  ;   AND '^U' FOR LINE DELETE; <CR> AS LINE TERMINATOR;
3922                                  ;   THROWS AWAY ALL OTHERS BELOW ASCII 40 AND ABOVE
3923                                  ;   ASCII 137.
3924                                  ;
3925  014344  004567  002512          LINGET: JSR     R5,SAVREG       ;SAVE REGS;
3926  014350  116500  000077          LINA:   MOVB    IDEV(R5),R0     ;GUESS, INPUT DEV, CODE
3927  014354  016001  014460                  MOV     TAB3(R0),R1     ;TO SELECT DEV;
3928  014360  005700                          TST     R0
3929  014362  001002                          BNE     LINEIN
3930  014364  060501                          ADD     R5,R1           ;USER BASE
3931  014366  011101                          MOV     (R1),R1         ;HAVE USER BUFF, HOR,
3932  014370  016502  000020          LINEIN: MOV     LINE(R5),R2
3933  014374  020265  000020          LINRUB: CMP     R2,LINE(R5)     ;DON'T RUB OUT TOO MUCH
3934  014400  001401                          BEQ     DOCHAR
3935  014402  005302                          DEC     R2
3936  014404  004767  177040          DOCHAR: JSR     PC,GETCHAR      ;DON'T COME BACK TILL YOU GOT ONE
3937  014410  103001                          BCC     GOTONE
3938  014412  000207                          RTS     PC              ;INDICATE INCOMPLETE LINE (CARRY SET)
3939  014414  112012          GOTONE: MOVB    R0,(R2)         ;PUT HER IN
3940  014416  004367  173602                  JSR     R3,CHKCHR       ;RETURN EXEC, ONE OF NEXT FIVE LOCS,
3941  014422  000752                          BR      LINA            ;IF LINE DELETE
3942  014424  000207                          RTS     PC              ;DONE (<CR>) (CARRY CLEAR)
3943  014426  000762                          BR      LINRUB
3944  014430  000765                          BR      DOCHAR          ;MEANINGLESS CHARS;
3945  014432  005202                          INC     R2              ;GOOD CHAR;
3946  014434  020265  000016                  CMP     R2,CODE(R5)     ;CHECK LINE SPACE OVERFLOW
3947  014440  103761                          BLO     DOCHAR
3948  014442  004167  001020                  JSR     R1,MSGERR
3949  014446     015     012                  ,BYTE   CR,LF
3950                                          ,IFNDF  $LONGER
3951  014450  052114     114                  ,ASCII  'LTL'
3952                                          ,ENDC
3953                                          ,IFDF   $LONGER
3954                                          ,ASCII  'LINE TOO LONG'
3955                                          ,ENDC
3956  014453     015     012                  ,BYTE   CR,LF
3957  014455     000                          ,BYTE   0
3958                                          ,EVEN
3959  014456  000744                          BR      LINEIN
3960                                  ;
3961                                  TAB3:
3962  014460  000172                          .       KBHD
3963                                          ,IFNDF  $NOPTP
3964  014462  021242                          .       PRBFHD
3965                                          ,ENDC
```

```
 3966
```



```
 3967                                   ;+++++++++++++++++++++++++++++++++++++++++++++++
 3968                                   ;  'LITEVAL' SUBROUTINE
 3969                                   ;               CALLED BY JSP PC,LITEVAL
 3970                                   ;               R1 POINTS TO THE CODE
 3971                                   ;               THIS SUBROUTINE IS CALLED TO EVALUATE A
 3972                                   ;               LITERAL IN THE CODE,
 3973  014464  005000                   LITEVAL:CLR     R0
 3974  014466  005065  000040                   CLR     FAC1(R5)
 3975  014472  005065  000042                   CLR     FAC2(R5)
 3976  014476  121327  000232                   CMPB    (R3),#'+,PLUS
 3977  014502  001404                           BEQ     LEVPLUS
 3978  014504  121327  000234                   CMPB    (R3),#'-,MINUS
 3979  014510  001002                           BNE     LEVLIT
 3980  014512  005100                           COM     R0
 3981  014514  005203                LEVPLUS:INC     R3
 3982  014516  122327  000375          LEVLIT: CMPB    (R3)+,#'1,ILIT1
 3983  014522  001416                           BEQ     LEV1LIT
 3984  014524  124327  000374                   CMPB    -(R3),#'!,FLIT
 3985  014530  001404                           BEQ     LEVFLT
 3986  014532  122327  000376                   CMPB    (R3)+,#'2,ILIT2
 3987  014536  001406                           BEQ     LEV2LIT
 3988  014540  000207                           RTS     PC
 3989  014542  005203                LEVFLT: INC     R3
 3990  014544  112365  000041                   MOVB    (R3)+,FAC1+1(R5)
 3991  014550  112365  000040                   MOVB    (R3)+,FAC1(R5)
 3992  014554  112365  000043          LEV2LIT:MOVB    (R3)+,FAC2+1(R5)
 3993  014560  112365  000042          LEV1LIT:MOVB    (R3)+,FAC2(R5)
 3994  014564  005700                           TST     R0
 3995  014566  001411                           BEQ     LEVPOS
 3996  014570  005765  000040                   TST     FAC1(R5)
 3997  014574  001003                           BNE     LEVFNEG
 3998  014576  005465  000042                   NEG     FAC2(R5)
 3999  014602  000403                           BR      LEVPOS
 4000  014604  062765  100000  000040 LEVFNEG:ADD     #100000,FAC1(R5)
 4001  014612  062716  000002          LEVPOS: ADD     #2,(SP)
 4002  014616  000207                           RTS     PC
 4003
```

```
                        ;+-----------------------------------------------------
                        ;  SUBROUTINE 'LOCGET'  CALLED BY JSR PC
                        ;                       GETS LOC OF ELEMENT IN ARRAY (CHECKS SUBSCRIPTS)
                        ;                       R0  MUST CONTAIN SS1      GETS DESTROYED
                        ;                       R1  UNUSED
                        ;                       R2  MUST POINT TO VAR     GETS SET TO RESULT
                        ;                       R3  MUST CONTAIN SS2      GETS DESTROYED
                        ;                       R4  UNUSED
                        ;                       R5  MUST POINT TO USER AREA
                        ;                       SP  GOES ?? DEEPER AFTER JSR
014620  021227  177775  LOCGET: CMP     (R2),#,SCALAR
014624  001403                  BEQ     LOCALOC
014626  005762  000004          TST     4(R2)
014632  100015                  BPL     LOCSS1
014634  010246  LOCALOC:MOV     R2,-(SP)
014636  010446                  MOV     R4,-(SP)
014640  010346                  MOV     R3,-(SP)
                        
                                ,IFNDF  $NOSTR
                                JSR     PC,DNPACK
                                ,ENDC
                        
014642  012703  000012          MOV     #12,R3
014646  011604                  MOV     (SP),R4
014650  100401                  BMI     ,+4
014652  010304                  MOV     R3,R4
014654  004767  172676          JSR     PC,ALLOC
014660  012603                  MOV     (SP)+,R3
014662  012604                  MOV     (SP)+,R4
014664  012602                  MOV     (SP)+,R2
014666  020062  000004  LOCSS1: CMP     R0,4(R2)
014672  101040                  BHI     ERRSS3
014674  005703                  TST     R3
014676  100427                  BMI     LOCNO2
014700  005762  000006          TST     6(R2)
014704  100433                  BMI     ERRSS3
014706  020362  000006          CMP     R3,6(R2)
014712  101030                  BHI     ERRSS3
014714  016246  000004          MOV     4(R2),-(SP)             ;****
014720  005216                  INC     (SP)                    ;*  *
014722  010346                  MOV     R3,-(SP)                ;*  *
014724  012746  000020          MOV     #20,-(SP)               ;*  *
014730  006303  LOCLOOP:ASL     R3                              ;*  *
014732  006366  000002          ASL     2(SP)                   ;*  *  MYULTIPLY SS2 BY SS1MAX
014736  103002                  BCC     ,+6                     ;*  *  RESULT IN R3
014740  066603  000004          ADD     4(SP),R3                ;*  *
014744  005316                  DEC     (SP)                    ;*  *
014746  003370                  BGT     LOCLOOP                 ;*  *
014750  022626                  CMP     (SP)+,(SP)+             ;*  *
014752  005726                  TST     (SP)+                   ;****
014754  060300                  ADD     R3,R0
014756  005200  LOCNO2: INC     R0
014760  006300                  ASL     R0
                        
                                ,IFNDF  $NOSTR
```



```
                                CMP     (R2),#,SVAR
                                BEQ     ,+4
                                ,ENDC
                        
014762  006300                  ASL     R0
014764  016202  000002          MOV     2(R2),R2
014770  060002                  ADD     R0,R2
014772  000207                  RTS     PC
014774  000004  ERRSS3: IOT
                                ,IFNDF  $LONGER
014776  047523      102         ,ASCII  \%OB\
                                ,ENDC
                                ,IFDF   $LONGER
                                ,ASCII  'SUBSCRIPT OUT OF BOUNDS'
                                ,ENDC
015001      000                 ,BYTE 0
                                ,EVEN
                                ,EOT
```

```
 4078                                   ;                                SOURCE FILE #7
 4079                                   ;-------------------------------------------------
 4080                                   ;  SUBROUTINE 'LSTPROG' CALLED BY JSR PC
 4081                                   ;                       LISTS THE USER PROGRAM THROUGH /PUTCHAR/
 4082                                   ;                            R0 - R4 DESTROYED
 4083                                   ;                       R5  MUST POINT TO USER AREA
 4084                                   ;                       SP  GOES ?? DEEPER AFTER JSR
 4085  015002  016501  000016          LSTPROG:MOV     CODE(R5),R1
 4086  015006  112102                  LSTLOOP:MOVB    (R1)+,R2
 4087  015010  002142                          BGE     LSTPTN
 4088  015012  120227  000201                  CMPB    R2,#,EOL
 4089  015016  001444                          BEQ     LSTEOL
 4090  015020  120227  000225                  CMPB    R2,#,EOF
 4091  015024  001001                          BNE     .+4
 4092  015026  000207                          RTS     PC
 4093  015030  120227  000374                  CMPB    R2,#,FLIT
 4094  015034  103064                          BHIS    LSTSPEC
 4095  015036  042702  177400                  BIC     #177400,R2
 4096  015042  013703                          MOV     PC,R3
 4097  015044  062703  163340          LSTPC:  ADD     #KEYWDS-LSTPC,R3
 4098  015050  010304                  LSTSRCH:MOV     R3,R4           ;R4 IS START OF KEYWORD
 4099  015052  112300                          MOVB    (R3)+,R0
 4100  015054  100376                          BPL     .-2             ;LOOP TILL GET END KEYWORD
 4101  015056  120002                          CMPB    R0,R2           ;COMPARE VALUES
 4102  015060  001373                          BNE     LSTSRCH
 4103  015062  112400                  LSTKWD: MOVB    (R4)+,R0
 4104  015064  100403                          BMI     LSTCHK
 4105  015066  004767  001004                  JSR     PC,PUTCHAR
 4106  015072  000773                          BR      LSTKWD
 4107  015074  124127  000211          LSTCHK: CMPB    @(R1)+,#,NEXT
 4108  015100  001410                          BEQ     LSTNEXT
 4109  015102  122127  000202                  CMPB    (R1)+,#,FN
 4110  015106  001337                          BNE     LSTLOOP
 4111  015110  112100                  LSTFN:  MOVB    (R1)+,R0
 4112  015112  062700  000200                  ADD     #'A-'A+2,R0
 4113  015116  006000                          ROR     R0
 4114  015120  000427                          BR      LSTCHAR
 4115  015122  062701  000013          LSTNEXT:ADD     #13,R1
 4116  015126  000727                          BR      LSTLOOP
 4117  015130  111102                  LSTEOL: MOVB    (R1),R2
 4118  015132  100010                          BPL     LSTLIN
 4119  015134  120227  000225                  CMPB    R2,#,EOF
 4120  015140  001015                          BNE     LSTBSLH
 4121  015142  004167  000354          LSTCRLF:JSR     R1,MSGODEV
 4122  015146      015     012     000         .BYTE   CR,LF,0
 4123              015152                      .EVEN
 4124  015152  000715                          BR      LSTLOOP
 4125  015154  000302                  LSTLIN: SWAB    R2
 4126  015156  005201                          INC     R1
 4127  015160  151102                          BISB    (R1),R2
 4128  015162  005301                          DEC     R1
 4129  015164  061502                          ADD     (R5),R2
 4130  015166  021227  177775                  CMP     (R2),#,SCALAR
 4131  015172  103763                          BLO     LSTCRLF
 4132  015174  012700  000134          LSTBSLH:MOV     #'\,R0
```



```
 4133  015200  004767  001472          LSTCHAR:JSR     PC,PUTCHAR
 4134  015204  000700                          BR      LSTLOOP
 4135  015206  120227  000374          LSTSPEC:CMPB    R2,#,FLIT
 4136  015212  001425                          BEQ     LSTFLIT
 4137  015214  120227  000376                  CMPB    R2,#,ILIT2
 4138  015220  103406                          BLO     LSTIL1
 4139  015222  001412                          BEQ     LSTIL2
 4140  015224  112100                  LSTTEXT:MOVB    (R1)+,R0
 4141  015226  001667                          BEQ     LSTLOOP
 4142  015230  004767  001442                  JSR     PC,PUTCHAR
 4143  015234  000773                          BR      LSTTEXT
 4144  015236  105065  000043          LSTIL1: CLRB    FAC2+1(R5)
 4145  015242  112165  000042                  MOVB    (R1)+,FAC2(R5)
 4146  015246  000434                          BR      LSTILB
 4147  015250  112165  000043          LSTIL2: MOVB    (R1)+,FAC2+1(R5)
 4148  015254  112165  000042                  MOVB    (R1)+,FAC2(R5)
 4149  015260  005065  000040          LSTILB: CLR     FAC1(R5)
 4150  015264  000410                          BR      LSTFLB
 4151  015266  112165  000041          LSTFLIT:MOVB    (R1)+,FAC1+1(R5)
 4152  015272  112165  000040                  MOVB    (R1)+,FAC1(R5)
 4153  015276  112165  000043                  MOVB    (R1)+,FAC2+1(R5)
 4154  015302  112165  000042                  MOVB    (R1)+,FAC2(R5)
 4155  015306  004767  000434          LSTFLB: JSR     PC,NUMOUT
 4156  015312  000167  177470          LSTJMP: JMP     LSTLOOP
 4157  015316  000302                  LSTPTR: SWAB    R2
 4158  015320  152102                          BISB    (R1)+,R2
 4159  015322  061502                          ADD     (R5),R2
 4160  015324  021227  177775                  CMP     (R2),#,SCALAR
 4161  015330  103030                          BHIS    LSTVAR
 4162  015332  011202                          MOV     (R2),R2
 4163  015334  002405                          BLT     LSTLNO
 4164  015336  010265  000042                  MOV     R2,FAC2(R5)
 4165  015342  005065  000040                  CLR     FAC1(R5)
 4166  015346  000414                          BR      LSTLNX
 4167  015350  110265  000043          LSTLNO: MOVB    R2,FAC2+1(R5)
 4168  015354  105065  000042                  CLRB    FAC2(R5)
 4169  015360  000302                          SWAB    R2
 4170  015362  110265  000040                  MOVB    R2,FAC1(R5)
 4171  015366  105065  000041                  CLRB    FAC1+1(R5)
 4172  015372  062765  043600  000040          ADD     #43600,FAC1(R5)
 4173  015400  004767  000342          LSTLNX: JSR     PC,NUMOUT
 4174  015404  012700  000040                  MOV     #BL,R0
 4175  015410  000673                          BR      LSTCHAR
 4176  015412  116200  000010          LSTVAR: MOVB    10(R2),R0
 4177  015416  004767  001254                  JSR     PC,PUTCHAR
 4178  015422  116200  000011                  MOVB    11(R2),R0
 4179  015426  001402                          BEQ     .+6
 4180  015430  004767  001242                  JSR     PC,PUTCHAR
 4181
 4182                                          .IFDF   $NOSTR
 4183  015434  000726                          BR      LSTJMP
 4184                                          .ENDC
 4185                                          .IFNDF  $NOSTR
 4186                                          CMP     (R2),#,SVAR
 4187                                          BNE     LSTJMP
```

```
4188                                    MOV     #1$,R0
4189                                    BR      LSTCHAR
4190                                    .ENDC
4191
```



```
4192                                    .IFNDF  $NOSTR
4193
4194                            ;+++++++++++++++++++++++++++++++++++++++++++++++++++++
4195                            ;  SUBROUTINE 'MAKESTR' CALLED BY JSR PC
4196                            ;                       MAKES A BASIC STRING FROM A STRING OF CHARACTERS
4197                            ;                       R0  DESTROYED
4198                            ;                       R1  UNUSED
4199                            ;                       R2  MUST POINT TO A WORD WHICH CONTAINS 3 LESS
4200                            ;                           THAN THE ADDRESS OF THE FIRST CHARACTER
4201                            ;                       R3  DESTROYED
4202                            ;                       R4  UNUSED
4203                            ;                       R5  MUST POINT TO USER AREA
4204                            ;                       SP  ON ENTRY 0(SP) MUST CONTAIN # CHARACTERS
4205                            ;                           ON EXIT 2(SP) CONTAINS POINTER TO THE STRING
4206                            MAKESTR:JSR     PC,MAKETRY      ;WILL NEW STRING FIT?
4207                                    BLOS    MAKGOT          ;YES
4208                                    JSR     PC,DNPACK       ;NO, GARBAGE COLLECT
4209                                    JSR     PC,MAKETRY      ;TRY AGAIN
4210                                    BLOS    MAKGOT          ;FITS THIS TIME
4211                            ERRSOV: IOT
4212                                    .IFNDF  $LONGER
4213                                    .ASCII  \$SO\
4214                                    .ENDC
4215                                    .IFDF   $LONGER
4216                                    .ASCII  'STRING STORAGE OVERFLOW'
4217                                    .ENDC
4218                                    .BYTE 0
4219                                    .EVEN
4220                            MAKCHK: JSR     PC,MAKETRY
4221                                    BHI     ERRSOV
4222                            MAKGOT: MOV     R0,HISTR(R5)    ;SAVE NEW HIGH STRING ADDR
4223                                    MOV     2(SP),R0        ;R0 NOW CONTAINS # CHARS,
4224                                    MOV     (R2),R2
4225                                    ADD     #3,R2           ;R2 NOW CONTAINS ADDRESS OF CHARS,
4226                                    MOVB    R0,(R3)+                ;R3 PUTS CHARACTERS IN
4227                                    CMPB    (R3)+,(R3)+
4228                                    MOVB    (R2)+,(R3)+
4229                                    DEC     R0
4230                                    BGT     .-4
4231                                    MOV     2(SP),R0
4232                                    MOVB    R0,(R3)
4233                                    SUB     R0,R3
4234                                    MOV     SP,R0
4235                                    TST     (R0)+
4236                                    MOVB    R0,-(R3)
4237                                    SWAB    R0
4238                                    MOVB    R0,-(R3)
4239                                    DEC     R3
4240                                    MOV     R3,2(SP)
4241                                    RTS     PC
4242
4243                            ;       SUBROUTINE TO CHECK ROOM ENOUGH FOR STRING
4244                            MAKETRY:MOV     HISTR(R5),R0
4245                                    MOV     R0,R3
4246                                    ADD     4(SP),R0
```

```
4247                                    ADD     #4,R0
4248                                    CMP     R0,HIFREE(R5)
4249                                    RTS     PC
4250                                    .ENDC
4251
4252
4253
4254
4255                            ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
4256                            ;  SUBROUTINE TO MPYTEN:  CALLED BY JSR PC
4257                            ;                         MULTIPLIES R3,R4 BY DECIMAL 10
4258                            ;                         R0,R1,R2  UNUSED
4259                            ;                         R3,R4  IS THE 32 BIT UNSIGNED INTEGER TO MULTPLY
4260                            ;                         R5  UNUSED
4261                            ;                         SP  GOES 4 DEEPER AFTER JSR
4262  015436  010346            MPYTEN: MOV     R3,-(SP)
4263  015440  010446                    MOV     R4,-(SP)
4264  015442  006304                    ASL     R4
4265  015444  006103                    ROL     R3
4266  015446  006304                    ASL     R4
4267  015450  006103                    ROL     R3
4268  015452  062604                    ADD     (SP)+,R4
4269  015454  005503                    ADC     R3
4270  015456  062603                    ADD     (SP)+,R3
4271  015460  006304                    ASL     R4
4272  015462  006103                    ROL     R3
4273  015464  000207                    RTS     PC
4274
```



```
4275                            ;
4276                            ; MSG - OUTPUT A LINE TO TTY
4277                            ; MSGODEV - OUTPUT A LINE TO CURR. OUTPUT DEV.
4278                            ;
4279                            ; CALL: JSR     R1,MSG (MSGODEV)
4280                            ;       .ASCII  "[MESSAGE]"
4281                            ;       .BYTE   0
4282                            ;       .EVEN
4283                            ;                       ;RETURN HERE
4284                            ;
4285                            ;  USES R0
4286                            ;
4287                            ;  SET UP LINE WITH NUMBER OF BYTES TERMINATED
4288                            ;   BY A BYTE OF 000.
4289                            ;
4290                            MSGERR:
4291                                    .IFNDF  SLONGER
4292  015466  112700  000045                  MOVB    #'%,R0
4293  015472  000401                    BR      MSGCOM
4294                                    .ENDC
4295  015474  112100            MSG:    MOVB    (R1)+,R0
4296  015476  105065  000107    MSGCOM: CLRB    GNOFLG(R5)
4297  015502  016546  000034            MOV     COLUMN(R5),-(SP)        ;SAVE FOR TEMP SWITCH
4298  015506  012765  000036  000034    MOV     #CLMNITY,COLUMN(R5)
4299  015514  060565  000034            ADD     R5,COLUMN(R5)
4300  015520  000404                    BR      FRSTOUT
4301  015522  016546  000034    MSGODEV:MOV     COLUMN(R5),-(SP)
4302  015526  112100            DOMSG:  MOVB    (R1)+,R0
4303  015530  001403                    BEQ     LINDUN
4304  015532  004767  001140    FRSTOUT:JSR     PC,PUTCHAR              ;PUT CHAR. TO TTY
4305  015536  000773                    BR      DOMSG           ;DO NEXT CHAR
4306  015540  005201            LINDUN: INC     R1              ;TO INSURE RETURN ON
4307  015542  006201                    ASR     R1              ;  EVEN WORD BOUNDARY
4308  015544  006301                    ASL     R1
4309  015546  012665  000034            MOV     (SP)+,COLUMN(R5)
4310  015552  000201                    RTS     R1
4311
```

```
4312                                    ;-------------------------------------------------------
4313                                    ; SUBROUTINE 'NORM'     NORMALIZES INTEGER CONTAINED IN
4314                                    ;                       R3 AND R4, MULTIPLIED BY EXPONENT
4315                                    ;                       IN R2.  ANSWER IS IN R3, R4,
4316                                    ;                       CALLED BY JSR PC
4317  015554  012746  000237           NORM:   MOV     #237,-(SP)
4318  015560  005703                           TST     R3
4319  015562  001413                           BEQ     LITTST
4320  015564  100003                           BPL     LITNORM
4321  015566  006003                           ROR     R3
4322  015570  006004                           ROR     R4
4323  015572  005216                           INC     (SP)
4324  015574  030327  040000           LITNORM:BIT     R3,#40000
4325  015600  001010                           BNE     LITOK
4326  015602  006304                           ASL     R4
4327  015604  006103                           ROL     R3
4328  015606  005316                           DEC     (SP)
4329  015610  000771                           BR      LITNORM
4330  015612  005704           LITTST: TST     R4
4331  015614  001367                           BNE     LITNORM
4332  015616  005726                           TST     (SP)+
4333  015620  000207                           RTS     PC
4334  015622  005702           LITOK:  TST     R2
4335  015624  001425                           BEQ     LITSTO
4336  015626  100416                           BMI     LITDIV
4337  015630  006203                           ASR     R3
4338  015632  006004                           ROR     R4
4339  015634  006203                           ASR     R3
4340  015636  006004                           ROR     R4
4341  015640  006203                           ASR     R3
4342  015642  006004                           ROR     R4
4343  015644  006203                           ASR     R3
4344  015646  006004                           ROR     R4
4345  015650  062716  000004                   ADD     #4,(SP)
4346  015654  004767  177556                   JSR     PC,MPYTEN
4347  015660  005302                           DEC     R2
4348  015662  000744                           BR      LITNORM
4349  015664  004767  172070           LITDIV: JSR     PC,DIVTEN
4350  015670  005202                           INC     R2
4351  015672  000740                           BR      LITNORM
4352  015674  006203           LITSHR: ASR     R3
4353  015676  006004                           ROR     R4
4354  015700  030327  177000           LITSTO: BIT     R3,#177000
4355  015704  001373                           BNE     LITSHR
4356  015706  005716                           TST     (SP)
4357  015710  003002                           BGT     .+6
4358  015712  012716  000001                   MOV     #1,(SP)
4359  015716  021627  000377                   CMP     (SP),#377
4360  015722  101402                           BLOS    .+6
4361  015724  012716  000377                   MOV     #377,(SP)
4362  015730  110366  000001                   MOVB    R3,1(SP)
4363  015734  012603                           MOV     (SP)+,R3
4364  015736  000303                           SWAB    R3
4365  015740  006003                           ROR     R3
4366  015742  006004                           ROR     R4
```



```
4367  015744  000207                           RTS     PC
4368
```

```
4369                                    ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
4370                                    ;  'NUMOUT' SUBROUTINE
4371                                    ;               CALLED BY JSR PC,NUMOUT
4372                                    ;               OUTPUTS AN UNSIGNED NUMBER FROM THE FAC
4373                                    ;               VIA THE SUBROUTINE PUTCHAR,
4374  015746  004567  001110      NUMOUT: JSR     R5,SAVREG
4375  015752  012701  016676              MOV     #PUTCHAR,R1
4376  015756  000443                      BR      NUMSTI
4377
4378
4379                                    ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
4380                                    ;  'NUMSGN' SUBROUTINE
4381                                    ;               CALLED BY JSR PC,NUMSGN
4382                                    ;                       ;WORD OUTPUT
4383                                    ;               WHERE OUTPUT IS THE OUTPUT ROUTINE,
4384                                    ;               WHICH MAY BE PUTCHAR OR SAVCHAR,
4385                                    ;               OUTPUTS A SIGNED NUMBER FROM THE
4386                                    ;               FAC VIA THE OUTPUT ROUTINE,
4387  015760  017600  000000      NUMSGN: MOV     @(SP),R0
4388  015764  062716  000002              ADD     #2,(SP)
4389  015770  004567  001066              JSR     R5,SAVREG
4390  015774  010001                      MOV     R0,R1
4391  015776  005765  000040              TST     FAC1(R5)
4392  016002  001410                      BEQ     NUMFIX
4393  016004  100025                      BPL     NUMPOS
4394  016006  062765  100000  000040      ADD     #100000,FAC1(R5)
4395  016014  001016                      BNE     NUMNEG
4396  016016  005065  000042              CLR     FAC2(R5)
4397  016022  001016                      BNE     NUMPOS
4398  016024  005765  000042      NUMFIX: TST     FAC2(R5)
4399  016030  100013                      BPL     NUMPOS
4400  016032  005465  000042              NEG     FAC2(R5)
4401  016036  102005                      BVC     NUMNEG
4402  016040  012765  044000  000040      MOV     #44000,FAC1(R5)
4403  016046  005065  000042              CLR     FAC2(R5)
4404  016052  012700  000055      NUMNEG: MOV     #'-,R0
4405  016056  000402                      BR      NUMPRS
4406                                  NUMPOS:
4407                                          .IFNDF  $NOSTR
4408                                          CMP     R1,#SAVCHAR
4409                                          BEQ     NUMSTI          ;DON'T OUTPUT BLANK FOR STRS
4410                                          .ENDC
4411
4412  016060  012700  000040              MOV     #BL,R0
4413  016064  004711              NUMPRS: JSR     PC,(R1)
4414
4415                                  NUMSTI:
4416  016066  016504  000042              MOV     FAC2(R5),R4     ;R4 = LOW ORDER MANTISSA,
4417  016072  005002                      CLR     R2              ;R2 = DECIMAL EXPONENT,
4418  016074  016503  000040              MOV     FAC1(R5),R3     ;R3 = HIGH ORDER MANTISSA,
4419  016100  001010                      BNE     NUMFLT
4420  016102  012700  000037              MOV     #37,R0          ;R0 = BINARY EXPONENT,
4421  016106  005704                      TST     R4
4422  016110  001016                      BNE     NUMSHFT
4423  016112  012700  000060              MOV     #'0,R0
```



```
4424  016116  004711                      JSR     PC,(R1)
4425  016120  000207                      RTS     PC
4426  016122  010300              NUMFLT: MOV     R3,R0
4427  016124  006300                      ASL     R0
4428  016126  105000                      CLRB    R0
4429  016130  000300                      SWAB    R0
4430  016132  162700  000171              SUB     #171,R0
4431  016136  042703  177600              BIC     #177600,R3
4432  016142  052703  000200              BIS     #200,R3
4433  016146  005300              NUMSHFT:DEC     R0
4434  016150  006304                      ASL     R4
4435  016152  006103                      ROL     R3
4436  016154  030327  040000      NUMNORM:BIT     R3,#40000
4437  016160  001772                      BEQ     NUMSHFT
4438  016162  005700                      TST     R0
4439  016164  003016                      BGT     NUMBIG
4440  016166  006203                      ASR     R3
4441  016170  006004                      ROR     R4
4442  016172  006203                      ASR     R3
4443  016174  006004                      ROR     R4
4444  016176  006203                      ASR     R3
4445  016200  006004                      ROR     R4
4446  016202  006203                      ASR     R3
4447  016204  006004                      ROR     R4
4448  016206  062700  000004              ADD     #4,R0
4449  016212  004767  177220              JSR     PC,MPYTEN
4450  016216  005302                      DEC     R2
4451  016220  000755                      BR      NUMNORM
4452  016222  020027  000004      NUMBIG: CMP     R0,#4
4453  016226  003437                      BLE     NUMOK
4454  016230  004767  172324      NUMDIV: JSR     PC,DIVTEN
4455  016234  005202                      INC     R2
4456  016236  000746                      BR      NUMNORM
4457  016240  005200              NUMALN: INC     R0
4458  016242  006203                      ASR     R3
4459  016244  006004                      ROR     R4
4460  016246  020027  000004      NUMOK:  CMP     R0,#4
4461  016252  002772                      BLT     NUMALN
4462  016254  020327  050000              CMP     R3,#50000
4463  016260  103363                      BHIS    NUMDIV
4464  016262  062704  001250              ADD     #1250,R4        ;ROUNDING,
4465  016266  103010                      BCC     NORNDOV
4466  016270  005203                      INC     R3
4467  016272  020327  050000              CMP     R3,#50000
4468  016276  103404                      BLO     NORNDOV
4469  016300  012703  004000              MOV     #4000,R3
4470  016304  005004                      CLR     R4
4471  016306  005202                      INC     R2
4472  016310  012700  000006      NORNDOV:MOV     #6,R0
4473  016314  010346              NUMDIG: MOV     R3,-(SP)
4474  016316  000316                      SWAB    (SP)
4475  016320  006016                      ROR     (SP)
4476  016322  006016                      ROR     (SP)
4477  016324  006016                      ROR     (SP)
4478  016326  042716  177760              BIC     #177760,(SP)
```

```
 4479  016332  062716  000060                ADD     #'0,(SP)
 4480  016336  042703  174000                BIC     #174000,R3
 4481  016342  004767  177070                JSR     PC,MPYTEN
 4482  016346  005300                         DEC     R0
 4483  016350  003361                         BGT     NUMDIG
 4484  016352  010603                         MOV     SP,R3
 4485  016354  062703  000014                ADD     #14,R3
 4486  016360  020227  177776                CMP     R2,#-2
 4487  016364  002404                         BLT     NUMFM1
 4488  016366  001446                         BEQ     NUMFM2
 4489  016370  020227  000006                CMP     R2,#6
 4490  016374  002451                         BLT     NUMFM3
 4491  016376  014300          NUMFM1: MOV     -(R3),R0
 4492  016400  004711                         JSR     PC,(R1)
 4493  016402  012700  000056                MOV     #'.,R0
 4494  016406  004711                         JSR     PC,(R1)
 4495  016410  012704  000005                MOV     #5,R4
 4496  016414  014300          NUMLP1: MOV     -(R3),R0
 4497  016416  004711                         JSR     PC,(R1)
 4498  016420  005304                         DEC     R4
 4499  016422  003374                         BGT     NUMLP1
 4500  016424  012700  000105                MOV     #'E,R0
 4501  016430  004711                         JSR     PC,(R1)
 4502  016432  012700  000053                MOV     #'+,R0
 4503  016436  005702                         TST     R2
 4504  016440  100003                         BPL     .+10
 4505  016442  012700  000055                MOV     #'-,R0
 4506  016446  005402                         NEG     R2
 4507  016450  004711                         JSR     PC,(R1)
 4508  016452  012700  000060                MOV     #'0,R0
 4509  016456  162702  000012                SUB     #12,R2
 4510  016462  100402                         BMI     .+6
 4511  016464  005200                         INC     R0
 4512  016466  000773                         BR      .-10
 4513  016470  004711                         JSR     PC,(R1)
 4514  016472  010200                         MOV     R2,R0
 4515  016474  062700  000072                ADD     #'0+12,R0
 4516  016500  004711                         JSR     PC,(R1)
 4517  016502  000435                         BR      NUMEXIT
 4518  016504  012700  000056          NUMFM2: MOV     #'.,R0
 4519  016510  004711                         JSR     PC,(R1)
 4520  016512  012700  000060                MOV     #'0,R0
 4521  016516  004711                         JSR     PC,(R1)
 4522  016520  012704  000006          NUMFM3: MOV     #6,R4
 4523  016524  010600                         MOV     SP,R0
 4524  016526  022027  000060                CMP     (R0)+,#'0
 4525  016532  001002                         BNE     .+6
 4526  016534  005304                         DEC     R4
 4527  016536  000773                         BR      .-10
 4528  016540  020402                         CMP     R4,R2
 4529  016542  003002                         BGT     .+6
 4530  016544  010204                         MOV     R2,R4
 4531  016546  005204                         INC     R4
 4532  016550  005202                         INC     R2
 4533  016552  005202                         INC     R2
```



```
 4534  016554  005302          NUMLP3: DEC     R2
 4535  016556  001003                  BNE     .+10
 4536  016560  012700  000056          MOV     #'.,R0
 4537  016564  004711                  JSR     PC,(R1)
 4538  016566  014300                  MOV     -(R3),R0
 4539  016570  004711                  JSR     PC,(R1)
 4540
 4541  016572  005304                  DEC     R4
 4542  016574  003367                  BGT     NUMLP3
 4543  016576  062706  000014  NUMEXIT:ADD     #14,SP
 4544  016602  000207                  RTS     PC
 4545
 4546                                  .IFNDF  $NOSTM
 4547                          SAVCHAR:
 4548                                  JSR     R5,SAVREG
 4549                                  MOV     T2(R5),R1
 4550                                  MOVB    R0,(R1)+
 4551                                  MOV     R1,T2(R5)
 4552                                  INC     T1(R5)
 4553                                  RTS     PC
 4554                                  .ENDC
 4555
```

```
4556                                    ;
4557                                    ; PUTBYT - PUT BYTE INTO RING BUFFER
4558                                    ;
4559                                    ; CALL: MOV     [BUFF, HDR],R1
4560                                    ;       MOV     [CHAR],R0
4561                                    ;       JSR     PC,PUTBYT
4562                                    ;
4563                                    ;  USES R0,R1,R2,R3
4564                                    ;
4565                                    ;  RAISE PRIORITY SINCE THIS MAY BE CALLED AT
4566                                    ;   MAINSTREAM, BUT ACCESSES POINTERS THAT MAY CHANGE AT
4567                                    ;   INTERUPT LEVEL.
4568                                    ;  RETURNS WITH 'C' BIT: CLR IF BYTE GOT IN
4569                                    ;                        SET IF NO ROOM
4570                                    ;
4571  016604  012702  177776           PUTBYT: MOV     #PS,R2
4572  016610  011246                           MOV     (R2),-(SP)      ;SAVE STATUS
4573  016612  012712  000340                   MOV     #PR7,(R2)       ;REPLACE WITH HIGHEST
4574  016616  016103  000010                   MOV     BPUT(R1),R3
4575  016622  005203                           INC     R3              ;CANDIDATE FOR NEW POSITION
4576  016624  020361  000002                   CMP     R3,BEND(R1)     ;NEED TO WRAP AROUND?
4577  016630  101402                           BLOS    NOWRAP
4578  016632  016103  000000                   MOV     BSTRT(R1),R3    ;YES: NEW POS, AT TOP
4579  016636  020361  000004           NOWRAP: CMP     R3,BGET1(R1)    ;IF EQUAL TO EITHER GET PTR,,
4580  016642  001412                           BEQ     NOROOM          ;  THEN NO ROOM
4581  016644  020361  000006                   CMP     R3,BGET2(R1)
4582  016650  001407                           BEQ     NOROOM
4583  016652  110071  000018                   MOVB    R0,@BPUT(R1)
4584  016656  010361  000010                   MOV     R3,BPUT(R1)     ;UPDATE PTR,
4585  016662  012612                           MOV     (SP)+,(R2)      ;STATUS BACK
4586  016664  000241                           CLC                     ;SIGNAL SUCCESS
4587  016666  000207                           RTS     PC
4588  016670  012612           NOROOM: MOV     (SP)+,(R2)      ;STATUS BACK
4589  016672  000261                           SEC                     ;SHOW FAILURE
4590  016674  000207                           RTS     PC
4591
```



```
4592                                    ;
4593                                    ; PUTCHAR - PUT CHAR TO CURRENT OUTPUT DEV,
4594                                    ;
4595                                    ; CALL: (LOAD ODEV(R5))
4596                                    ;       MOV     [CHAR],R0
4597                                    ;       JSR     PC,PUTCHAR
4598                                    ;
4599                                    ;  SAVES ALL REGS,
4600                                    ;
4601                                    ;  TRIES TO FIT CHAR INTO PROPER BUFFER,  GOES TO IOWAIT ON
4602                                    ;   FAILURE,
4603                                    ;
4604  016676  004567  000160           PUTCHAR:JSR     R5,SAVREG
4605  016702  012746  047777                   MOV     #47777,-(SP)    ;INIT TIME OUT COUNT
4606  016706  005275  000034                   INC     @COLUMN(R5)
4607  016712  105765  000106                   TSTB    CNCFLG(R5)
4608  016716  001402                           BEQ     PUTCCO
4609  016720  000167  162152                   JMP     READY
4610  016724  116504  000076           PUTCCO: MOVB    ODEV(R5),R4     ;OUTPUT DEVICE CODE
4611  016730  001003                           BNE     HDRSTUP         ;IF NOT TTY, CONTINUE
4612  016732  105765  000107                   TSTB    CNOFLG(R5)      ;CHECK FOR ^O?
4613  016736  001032                           BNE     OUT010          ;IF SET EXIT IMMEDIATELY
4614  016740  016401  017030           HDRSTUP:MOV     TAB1(R4),R1
4615  016744  005704                           TST     R4              ;WHICH DEVICE?
4616  016746  001002                           BNE     PUTIT           ;NOT TTY
4617  016750  060501                           ADD     R5,R1           ;NEED CURR, USER'S BUFF,
4618  016752  011101                           MOV     (R1),R1         ;NOW, ACTUAL HEADER
4619  016754  004767  177624           PUTIT:  JSR     PC,PUTBYT
4620  016760  103007                           BCC     ITSIN
4621  016762  004767  175524                   JSR     PC,IOWAIT       ;NO ROOM! WAIT
4622  016766  005316                           DEC     (SP)            ;TIME OUT
4623  016770  001371                           BNE     PUTIT
4624  016772  105061  000012                   CLRB    BFSPEC(R1)
4625  016776  000422                           BR      DEVERR
4626  017000  105061  000012           ITSIN:  CLRB    BFSPEC(R1)
4627  017004  012774  000100  017036           MOV     #100,@TAB2(R4)  ;ENABLE INTERUPT ON CHOSEN DEV;
4628  017012  005265  000070                   INC     RNDCT(R5)
4629  017016  105761  000012                   TSTB    BFSPEC(R1)      ;I/O ERROR?
4630  017022  001010                           BNE     DEVERR
4631  017024  005726           OUT010: TST     (SP)+           ;POP TIME OUT COUNT
4632  017026  000207                           RTS     PC              ;RESTORES REGS, AND RETURNS
4633                                    ;
4634                                    ; TAB1 - DEV, BUFFER HDRS, (TPHD IS OFFSET INTO USRAREA
4635                                    ;  FOR ADDR, TO TPRTR, BUFF, HDR,)
4636                                    ;
4637  017030  000120           TAB1:   +       TPHD
4638                                            ,IFNDF $NOPTP
4639  017032  021306                           +       PPBFHD
4640                                            ,ENDC
4641                                            ,IFDF $NOPTP
4642                                            0
4643                                            ,ENDC
4644
4645                                            ,IFNDF $NOLPT
4646  017034  021352                           +       LPBFHD
```

```
                                        ,ENDC
                                        ,IFDF $NOLPT
                                        @
                                        ,ENDC
                                )
                                ) TAB2 - DEV, STATUS REG,
                                )
017036 177564                   TAB2:   +       TPS
017040 177554                           +       PPS
017042 177514                           +       LPS

017044 000004                   DEVERR: IOT
                                        ,IFNDF  $LONGER
017046 047104     122                   ,ASCII  \DNR\
                                        ,ENDC
                                        ,IFDF   $LONGER
                                        ,ASCII  \DEVICE NOT READY\
                                        ,ENDC
017051     000                          ,BYTE 0
                                        ,EVEN
                                        ,EOT
```



```
                                )                               SOURCE FILE #8
                                )*******************************************************
                                )
                                ) SAVREG - SAVE ALL REGISTERS
                                )
                                ) CALL: CJSR    PC,SUBR,]       ;MUST CALL SAVREG FROM SUBRTN,
                                )       JSR     R5,SAVREG
                                )
                                )  PUSHES ALL REGS, AND EXITS WITH ADDR, ON STACK
                                )   WHICH BRINGS CONTROL BACK WHEN SUBR, DOES (RTS      PC),
                                )   THEN, RESTORES REGS, AND DOES (RTS PC) WHICH RETURNS TO JUST
                                )   AFTER CALL TO INITIAL SUBROUTINE (MODIFIED BOWERING SPECIAL)
                                )
017052 011646                   SAVRGI: MOV     (SP),-(SP)
017054 012766 017122  000002            MOV     #INTEXT,2(SP)
017062 010446                   SAVREG: MOV     R4,-(SP)
017064 010346                           MOV     R3,-(SP)
017066 010246                           MOV     R2,-(SP)
017070 010146                           MOV     R1,-(SP)
017072 010046                           MOV     R0,-(SP)
017074 010546                           MOV     R5,-(SP)        ;FOR RETURN TO SUBR,
017076 016605  000014                   MOV     12.(SP),R5      ;GET OLD R5
017102 004736                           JSR     PC,@(SP)+       ;STACK NEXT LOC,!
017104 012600                   RESREG: MOV     (SP)+,R0
017106 012601                           MOV     (SP)+,R1
017110 012602                           MOV     (SP)+,R2
017112 012603                           MOV     (SP)+,R3
017114 012604                           MOV     (SP)+,R4
017116 012605                           MOV     (SP)+,R5
017120 000207                           RTS     PC              ;BACK TO CALLER OF INIT, SUBR,
                                )
017122 000002                   INTEXT: RTI
```

```
                                )*******************************************************
                                )
                                ) SCANPND - SCAN TOKEN FOR '#'
                                )
                                ) CALL: MOV     CPTR, TO BYTE BEFORE '#' TOKEN],R1
                                )       JSR     PC,SCANPND
                                )
                                )  USES R1
                                )
017124 012765 177775  000030    SCANPND:MOV     #,SCALAR,LINENO(R5)
017132 005201                           INC     R1
017134 122127 000261                    CMPB    (R1)+,#,POUND
017140 001402                           BEQ     PNDGO
017142 000167 172502                    JMP     ERRSX5
017146 000207                   PNDGO:  RTS     PC
```

```
                                ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                ;  SUBROUTINE 'SKIPEOL' CALLED BY JSR PC
                                ;                       MOVES R1 PAST THE NEXT 'EOL'
                                ;                       R0  UNUSED
                                ;                       R1  MODIFIED TO REFLECT OPERATION
                                ;                       R2,R3,R4,R5  UNUSED
                                ;                       SP  GOES NO DEEPER AFTER JSR
017150  062701  000003          SKIP05: ADD     #3,R1
017154  005201                  SKIP02: INC     R1
017156  005201                  SKIP01: INC     R1
017160  105721                  SKIPEOL:TSTB    (R1)+
017162  002375                          BGE     SKIP01
017164  124127  000374                  CMPB    -(R1),#,FLIT
017170  001767                          BEQ     SKIP05
017172  101014                          BHI     SKIPHI
017174  122127  000201                  CMPB    (R1)+,#,EOL
017200  001420                          BEQ     SKIPRTS
017202  124127  000202                  CMPB    -(R1),#,FN
017206  001762                          BEQ     SKIP02
017210  122127  000211                  CMPB    (R1)+,#,NEXT
017214  001361                          BNE     SKIPEOL
017216  062701  000012                  ADD     #12,R1
017222  000756                          BR      SKIPEOL
                                ;       EITHER ,ILIT1(375), ,ILIT2(376) OR ,TEXT(377)
017224  122127  000376          SKIPHI: CMPB    (R1)+,#,ILIT2
017230  001751                          BEQ     SKIP02
017232  103751                          BLO     SKIP01
017234  105721                  SKIPTXT:TSTB    (R1)+
017236  001376                          BNE     SKIPTXT
017240  000747                          BR      SKIPEOL
017242  000207                  SKIPRTS:RTS     PC


                                ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                ;  'STOVAR' SUBROUTINE
                                ;                       CALLED BY JSR PC,STOVAR
                                ;                       STORES FAC1, FAC2 IN THE VARIABLE OR
                                ;                       ARRAY ELEMENT LAST ADDRESSED BY THE
                                ;                       SUBROUTINE 'GETVAR'
017244  016502  000022          STOVAR: MOV     VARSAV(R5),R2

                                        .IFNDF  $NOSTR
                                        CMP     (R2),#,SVAR
                                        BEQ     ERRMX7
                                        .ENDC

017250  016500  000024                  MOV     SS1SAV(R5),R0
017254  100405                          BMI     STONOSS
017256  016503  000026                  MOV     SS2SAV(R5),R3
017262  004767  175332                  JSR     PC,LOCGET
017266  000404                          BR      STOCOMM
017270  022227  177775          STONOSS:CMP     (R2)+,#,SCALAR
017274  001401                          BEQ     STOCOMM
017276  011202                          MOV     (R2),R2
```



```
017300  016522  000040          STOCOMM:MOV     FAC1(R5),(R2)+
017304  016512  000042                  MOV     FAC2(R5),(R2)
017310  000207                          RTS     PC

                                        .IFNDF  $NOSTR
                                ERRMX7: JMP     ERRMIX
                                        .ENDC
```

```
4786                                    .IFNDF  $NOSTK
4787                                ;+--------------------------------------------------
4788                                ;  'STOSVAR' SUBROUTINE
4789                                ;               CALLED BY JSR PC,STOSVAR
4790                                ;               STORES A STRING VARIABLE AS ADDRESSED BY
4791                                ;               VARSAV
4792                                STOSVAR:MOV     VARSAV(R5),R2
4793                                        CMP     (R2),#,SVAR
4794                                        BNE     ERRMX6
4795                                        MOV     SS1SAV(R5),R0
4796                                        BMI     STOSSNO
4797                                        MOV     SS2SAV(R5),R3
4798                                        JSR     PC,LOCGET
4799                                        BR      STOMMCO
4800                                STOSSNO:TST     (R2)+
4801                                        CMP     2(R2),#-1
4802                                        BEQ     STOVTAB
4803                                        MOV     (R2),R2
4804                                STOMMCO:MOV     (SP)+,R3
4805                                        MOV     (SP)+,R0
4806                                        MOV     R0,(R2)
4807                                        BR      STOVCOM
4808                                STOVTAB:MOV     (SP)+,R3
4809                                        MOV     (SP)+,R0
4810                                        MOV     R0,(R2)
4811                                        INC     R2
4812                                        SUB     (R5),R2
4813                                STOVCOM:INC     R0
4814                                        BEQ     STOSX           ;CHECK NULL STRING
4815                                        ADD     #2,R0
4816                                        MOVB    R2,-(R0)
4817                                        SWAB    R2
4818                                        MOVB    R2,-(R0)
4819                                STOSX:  JMP     (R3)
4820                                ERRMX6: JMP     ERRMIX
4821                                        .ENDC
4822
4823
4824
4825
4826                                ;+--------------------------------------------------
4827                                ;  SUBROUTINE 'SUBSTK'  CALLED BY JSR PC
4828                                ;                       NEGATES @(SP)+,2(SP) THEN CONTINUES IN 'ADDSTK'
4829                                ;                       R0,R1  UNUSED
4830                                ;                       R2,R3  DESTROYED
4831                                ;                       R4  MODIFIED TO REFLECT STACK USAGE
4832                                ;                       R5  MUST POINT TO USER AREA
4833                                ;                       SP  GOES ?? DEEPER AFTER JSR
4834  017312  005766  000002       SUBSTK: TST     2(SP)
4835  017316  001405                       BEQ     SUBINT
4836  017320  062766  100000  000002       ADD     #100000,2(SP)
4837  017326  000167  170036       ADDSTK: JMP     ADDSTK
4838  017332  005466  000004       SUBINT: NEG     4(SP)
4839  017336  102373                       BVC     ADDSTK
4840  017340  012766  044000  000002       MOV     #44000,2(SP)
```



```
4841  017346  005066  000004               CLR     4(SP)
4842  017352  000765                       BR      ADDSTK
4843
```

```
4844                                    ;+++++++++++++++++++++++++++++++++++++++++++++++
4845                                    ; 'TRAN' SUBROUTINE
4846                                    ;               CALLED BY JSR PC,TRAN
4847                                    ;               TRANSLATES FROM INPUT ASCII CODE
4848                                    ;               TO INTERNAL CODE, CONSISTING OF TOKENS,
4849                                    ;               LINE NUMBER REFERENCES, SYMBOL TABLE
4850                                    ;               REFERENCES, AND LITERALS,  ALWAYS RETURNS,
4851                                    ;               NO MATTER WHAT IS INPUT,
4852                                    ;               AS NEW SYMBOLS ARE ENCOUNTERED, BUILDS NEW
4853                                    ;               ENTRIES INTO THE SYMBOL TABLE,
4854                                    ;               ERRORS ARE TRANSLATED TO [;TEXT,,,,]
4855 017354 016502 000020          TRAN:  MOV     LINE(R5),R2
4856 017360 010201                        MOV     R2,R1
4857 017362 122127 000015                 CMPB    (R1)+,#CR
4858 017366 001375                        BNE     .-4
4859 017370 016500 000016                 MOV     CODE(R5),R0
4860 017374 114140                        MOVB    -(R1),-(R0)
4861 017376 020102                        CMP     R1,R2           ;R0 IS WHERE ORIGINAL TEXT IS,
4862 017400 001375                        BNE     .-4             ;R1 IS WHERE TRANSLATED TEXT GOES,
4863 017402 005065 000056          TRANLUP:CLR    T1(R5)          ;PREVIOUS ITEM WASN'T A VAR,
4864 017406 122027 000040          TRANVAR:CMPB   (R0)+,#BL
4865 017412 001775                        BEQ     TRANVAR
4866 017414 124027 000015                 CMPB    -(R0),#CR
4867 017420 001003                        BNE     TRYNUM
4868 017422 112711 000201                 MOVB    #,EOL,(R1)
4869 017426 000207                        RTS     PC
4870 017430 121027 000056          TRYNUM: CMPB    (R0),#'.
4871 017434 001406                        BEQ     NUMJMP
4872 017436 121027 000060                 CMPB    (R0),#'0
4873 017442 103405                        BLO     TRYKWD
4874 017444 121027 000071                 CMPB    (R0),#'9
4875 017450 101002                        BHI     TRYKWD
4876 017452 000167 000632          NUMJMP: JMP     I$NUM
4877
4878 017456 012704 160700          TRYKWD: MOV     #KEYWDS-TRNPC,R4
4879 017462 121027 000101                 CMPB    (R0),#'A        ;CHECK ALPHA KEYWORDS
4880 017466 103405                        BLO     COMTAB  ;TOO LOW
4881 017470 121027 000132                 CMPB    (R0),#'Z
4882 017474 101002                        BHI     COMTAB  ;TOO HIGH
4883 017476 012704 160772                 MOV     #KEYA-TRNPC,R4  ;JUST RIGHT
4884 017502 010703          COMTAB: MOV     PC,R3
4885 017504 060403          TRNPC:  ADD     R4,R3           ;ADDRESS TABLE
4886 017506 010002          TRYAGN: MOV     R0,R2
4887 017510 122223          RETRY:  CMPB    (R2)+,(R3)+
4888 017512 001776                  BEQ     RETRY
4889 017514 114304                  MOVB    -(R3),R4
4890 017516 100414                  BMI     AMATCH
4891 017520 001550                  BEQ     NOTKWD
4892 017522 005302                  DEC     R2
4893 017524 122227 000040           CMPB    (R2)+,#BL
4894 017530 001767                  BEQ     RETRY
4895 017532 005302                  DEC     R2
4896 017534 122327 000040           CMPB    (R3)+,#BL
4897 017540 001763                  BEQ     RETRY
4898 017542 105723                  TSTB    (R3)+
```



```
4899 017544 100376                  BPL     .+2
4900 017546 000757                  BR      TRYAGN
4901 017550 005302          AMATCH: DEC     R2
4902 017552 010200                  MOV     R2,R0
4903 017554 110421                  MOVB    R4,(R1)+
4904 017556 120427 000220           CMPB    R4,#,KEM
4905 017562 001540                  BEQ     REMPACK
4906 017564 120427 000262           CMPB    R4,#,FN
4907 017570 001461                  BEQ     TRNFN
4908 017572 120427 000211           CMPB    R4,#,NEXT
4909 017576 001505                  BEQ     NEXFILL
4910 017600 120427 000257           CMPB    R4,#,SQUOT
4911 017604 001403                  BEQ     QUOTPAK
4912 017606 120427 000256           CMPB    R4,#,DQUOT
4913 017612 001273                  BNE     TRANLUP
4914 017614 122704 000257          QUOTPAK:CMPB    #,SQUOT,R4
4915 017620 001402                  BEQ     .+6
4916 017622 012704 000252           MOV     #'"-''+,SQUOT,R4
4917 017626 062704 177570           ADD     #''-",SQUOT,R4
4918 017632 112721 000377           MOVB    #,TEXT,(R1)+
4919 017636 220100                  CMP     R1,R0
4920 017640 103076                  BHIS    ERRTRN
4921 017642 121004          STOSTR: CMPB    (R0),R4         ;SAME AS INITIAL QUOTE CHAR,?
4922 017644 001405                  BEQ     ENDQUOT
4923 017646 121027 000015           CMPB    (R0),#CR
4924 017652 001415                  BEQ     ENDCR
4925 017654 112021                  MOVB    (R0)+,(R1)+
4926 017656 000771                  BR      STOSTR
4927 017660 105021          ENDQUOT:CLRB    (R1)+
4928 017662 005200                  INC     R0
4929 017664 120427 000047           CMPB    R4,#''
4930 017670 001403                  BEQ     ENDSQUO
4931 017672 112721 000256           MOVB    #,DQUOTE,(R1)+
4932 017676 000641                  BR      TRANLUP
4933 017700 112721 000257          ENDSQUO:MOVB    #,SQUOTE,(R1)+
4934 017704 000636                  BR      TRANLUP
4935 017706 105021          ENDCR:  CLRB    (R1)+
4936 017710 112711 000201           MOVB    #,EOL,(R1)
4937 017714 010102                  MOV     R1,R2
4938 017716 124227 000377           CMPB    -(R2),#,TEXT
4939 017722 001375                  BNE     .-4
4940 017724 110412                  MOVB    R4,(R2)
4941 017726 112742 000377           MOVB    #,TEXT,-(R2)
4942 017732 000207                  RTS     PC
4943 017734 122027 000040          TRNFN:  CMPB    (R0)+,#BL
4944 017740 001775                  BEQ     .-4
4945 017742 124027 000101           CMPB    -(R0),#'A
4946 017746 103411                  BLO     TRFNBAD
4947 017750 121027 000132           CMPB    (R0),#'Z
4948 017754 101006                  BHI     TRFNBAD
4949 017756 112002                  MOVB    (R0)+,R2
4950 017760 006302                  ASL     R2
4951 017762 162702 000200           SUB     #'A-'A-2,R2
4952 017766 110221                  MOVB    R2,(R1)+
4953 017770 000604                  BR      TRANLUP
```

```
 4954  017772  112740  000116          TRPNBAD:MOVB    #'N,-(R0)
 4955  017776  112740  000106                  MOVB    #'F,-(R0)
 4956  020002  005301                          DEC     R1
 4957  020004  020100                          CMP     R1,R0
 4958  020006  101013                          BHI     ERRTRN
 4959  020010  000425                          BR      REMPACK
 4960  020012  020100                  NEXFILL:CMP     R1,R0
 4961  020014  103010                          BHIS    ERRTRN
 4962  020016  105021                          CLRB    (R1)+
 4963  020020  105011                          CLRB    (R1)
 4964  020022  062701  000011                  ADD     #11,R1
 4965  020026  020100                          CMP     R1,R0
 4966  020030  103002                          BHIS    ERRTRN
 4967  020032  000167  177344                  JMP     TRANLUP
 4968  020036  000167  000266          ERRTRN: JMP     ERRTR7
 4969  020042  121027  000132          NOTKWD: CMPB    (R0),#'Z
 4970  020046  101006                          BHI     REMPACK
 4971  020050  121027  000101                  CMPB    (R0),#'A
 4972  020054  103403                          BLO     REMPACK
 4973  020056  005765  000056                  TST     T1(R5)          ;2 VARS IN A ROW IS AN ERROR SO DONT
 4974  020062  001415                          BEQ     ISVAR           ;WASTE SYMBOL SPACE ON IT,
 4975  020064  112721  000377          REMPACK:MOVB    #-1,TEXT-1(R1)+
 4976  020070  020100                          CMP     R1,R0
 4977  020072  103361                          BHIS    ERRTRN
 4978  020074  121027  000015          STOTXT: CMPB    (R0),#CR
 4979  020100  001402                          BEQ     CARRET
 4980  020102  112021                          MOVB    (R0)+,(R1)+
 4981  020104  000773                          BR      STOTXT
 4982  020106  105021                  CARRET: CLRB    (R1)+
 4983  020110  112711  000201                  MOVB    #-1,EOL-1(R1)
 4984  020114  000207                          RTS     PC
 4985  020116  112002                  ISVAR:  MOVB    (R0)+,R2
 4986  020120  122027  000040                  CMPB    (R0)+,#BL
 4987  020124  001775                          BEQ     .-4
 4988  020126  124027  000071                  CMPB    -(R0),#'9
 4989  020132  101006                          BHI     NODIGIT
 4990  020134  121027  000060                  CMPB    (R0),#'0
 4991  020140  103403                          BLO     NODIGIT
 4992  020142  000302                          SWAB    R2
 4993  020144  152002                          BISB    (R0)+,R2
 4994  020146  000302                          SWAB    R2
 4995  020150  011503                  NODIGIT:MOV     (R5),R3
 4996  020152  122027  000040                  CMPB    (R0)+,#BL
 4997  020156  001775                          BEQ     .-4
 4998  020160  005300                          DEC     R0
 4999  020162  020365  000014          VARSRCH:CMP     R3,LOFREE(R5)
 5000  020166  103020                          BHIS    PUTITIN
 5001  020170  021327  177775                  CMP     (R3),#-1,SCALAR
 5002  020174  103003                          BHIS    NOTIT
 5003  020176  062703  000004                  ADD     #4,R3
 5004  020202  000767                          BR      VARSRCH
 5005  020204  062703  000010          NOTIT:  ADD     #10,R3
 5006
 5007                                          .IFNDF  $NOSTR
 5008                                          CMP     -10(R3),#-1,SVAR
```



```
 5009                                          BEQ     TRYSVAR
 5010                                          .ENDC
 5011
 5012  020210  022302                          CMP     (R3)+,R2
 5013  020212  001363                          BNE     VARSRCH
 5014  020214  121027  000044                  CMPB    (R0),#'$
 5015  020220  001760                          BEQ     VARSRCH
 5016
 5017                                          .IFNDF  $NOSTR
 5018                                          BR      FOUNDV
 5019                                  TRYSVAR:CMP     (R3)+,R2
 5020                                          BNE     VARSRCH
 5021                                          CMPB    (R0),#'$
 5022                                          BNE     VARSRCH
 5023                                          INC     R0
 5024                                          .ENDC
 5025
 5026  020222  162703  000012          FOUNDV: SUB     #12,R3
 5027  020226  000415                          BR      RETVAR
 5028                                  PUTITIN:
 5029                                          .IFNDF  $NOSTR
 5030                                          ADD     #14,R3
 5031                                          CMP     R3,LOSTR(R5)    ;MAKE SURE STRINGS ARE NOT THERE
 5032                                          BLO     PUTAOK
 5033                                          JSR     PC,UPPACK       ;IF THEY ARE, PACK THEM UPWARDS
 5034                                          CMP     R3,LOSTR(R5)    ;AND TRY AGAIN
 5035                                          BHIS    ERROV2          ;NO GOOD
 5036                                  PUTAOK: TST     -(R3)           ;R3 IS END OF ENTRY
 5037                                          .ENDC
 5038
 5039                                          .IFDF   $NOSTR
 5040  020230  062703  000012                  ADD     #12,R3          ;ADDRESS END OF ENTRY IN R3
 5041                                          .ENDC
 5042
 5043  020234  020365  000012                  CMP     R3,HIFREE(R5)
 5044  020240  101041                          BHI     ERROV2
 5045  020242  010365  000014                  MOV     R3,LOFREE(R5)
 5046  020246  010243                          MOV     R2,-(R3)
 5047  020250  005043                          CLR     -(R3)
 5048  020252  005043                          CLR     -(R3)
 5049  020254  005043                          CLR     -(R3)
 5050  020256  012743  177775                  MOV     #-1,SCALAR,-(R3)
 5051
 5052                                          .IFNDF  $NOSTR
 5053                                          CMPB    (R0),#'$
 5054                                          BNE     RETVAR
 5055                                          INC     R0
 5056                                          MOV     #-1,SVAR,(R3)+
 5057                                          COM     (R3)+
 5058                                          COM     (R3)+
 5059                                          COM     (R3)
 5060                                          SUB     #6,R3
 5061                                          .ENDC
 5062
 5063  020262  010565  000056          RETVAR: MOV     R5,T1(R5)       ;SOMETHING NONZERO,
```

```
5064  020266  161503                  RETLNO: SUB     (R5),R3
5065  020270  000303                          SWAB    R3
5066  020272  110321                          MOVB    R3,(R1)+
5067  020274  000303                          SWAB    R3
5068  020276  110321                          MOVB    R3,(R1)+
5069  020300  020100                          CMP     R1,R0
5070  020302  101255                          BHI     ERRTRN
5071  020304  000167  177076                 JMP     TRANVAR
5072                                  ISNUM:
5073  020310  010046                          MOV     R0,-(SP)        ;SAVE PTR IN CASE OF BAD NUM
5074  020312  004767  000364                 JSR     PC,VAL          ;FIND MANTISSA AND EXPONENT
5075  020316  121027  000056                 CMPB    (R0),#'.
5076  020322  001016                          BNE     EXPTEN
5077  020324  012600                          MOV     (SP)+,R0
5078  020326  000656                          BR      REMPACK
5079  020330  004167  175132         ERRTR7: JSR     R1,MSGERR
5080                                          .IFNDF  $LONGER
5081  020334  046124     124                  .ASCII  \TLT\
5082                                          .ENDC
5083                                          .IFDF   $LONGER
5084                                          .ASCII  'LINE TOO LONG TO TRANSLATE'
5085                                          .ENDC
5086  020337     000                          .BYTE 0
5087                                          .EVEN
5088  020340  000167  160574                 JMP     READY2
5089  020344  004167  175116         ERROV2: JSR     R1,MSGERR
5090                                          .IFNDF  $LONGER
5091  020350  052120     102                  .ASCII  \PTB\
5092                                          .ENDC
5093                                          .IFDF   $LONGER
5094                                          .ASCII  'PROGRAM TOO BIG'
5095                                          .ENDC
5096  020353     000                          .BYTE 0
5097                                          .EVEN
5098  020354  000167  160560                 JMP     READY2
5099  020360  020327  014630         EXPTEN: CMP     R3,#14630
5100  020364  101127                          BHI     MAKFLIT
5101  020366  005702                          TST     R2
5102  020370  003404                          BLE     EXPNEG
5103  020372  004767  175040                 JSR     PC,MPYTEN
5104  020376  005302                          DEC     R2
5105  020400  000767                          BR      EXPTEN
5106  020402  001120         EXPNEG: BNE     MAKFLIT
5107  020404  005703                          TST     R3
5108  020406  001116                          BNE     MAKFLIT
5109  020410  020427  177775                 CMP     R4,#,SCALAR
5110  020414  103113                          BHIS    MAKFLIT
5111  020416  016503  000020                 MOV     LINE(R5),R3
5112  020422  005301                          DEC     R1
5113  020424  020103                          CMP     R1,R3
5114  020426  103452                          BLO     MAKLNO
5115  020430  121127  000204                 CMPB    (R1),#,GOSUB
5116  020434  001447                          BEQ     MAKLNO
5117  020436  121127  000205                 CMPB    (R1),#,GOTO
5118  020442  001444                          BEQ     MAKLNO
```



```
5119  020444  121127  000224                 CMPB    (R1),#,CALL
5120  020450  001441                          BEQ     MAKLNO
5121  020452  121127  000242                 CMPB    (R1),#,THEN
5122  020456  001436                          BEQ     MAKLNO
5123  020460  121127  000277                 CMPB    (R1),#,LIST
5124  020464  001433                          BEQ     MAKLNO
5125  020466  005201                          INC     R1
5126  020470  005003                          CLR     R3
5127  020472  005704                          TST     R4
5128  020474  002463                          BLT     MAKFLIT
5129  020476  020427  000377                 CMP     R4,#377
5130  020502  003003                          BGT     MAKEI2
5131  020504  112721  000375         LITZERO: MOVB    #,ILII1,(R1)+
5132  020510  000407                          BR      MAKEOK
5133  020512  112721  000376         MAKEI2: MOVB    #,ILII2,(R1)+
5134  020516  000304                 LITCOMN: SWAB    R4
5135  020520  020100                          CMP     R1,R0
5136  020522  103010                          BHIS    TRNER4
5137  020524  110421                          MOVB    R4,(R1)+
5138  020526  000304                          SWAB    R4
5139  020530  110421                 MAKEOK: MOVB    R4,(R1)+
5140  020532  005726                          TST     (SP)+
5141  020534  020100                          CMP     R1,R0
5142  020536  101002                          BHI     TRNER4
5143  020540  000167  176636                 JMP     TRANLUP
5144  020544  000167  177266         TRNER4: JMP     ERRTRN
5145  020550  000167  177570         ERROV3: JMP     ERROV2
5146  020554  005201                 MAKLNO: INC     R1
5147  020556  011503                          MOV     (R5),R3
5148  020560  020365  000014         LNOSRCH: CMP     R3,LOFREE(R5)
5149  020564  103013                          BHIS    STOLNO
5150  020566  021327  177775                 CMP     (R3),#,SCALAR
5151  020572  103403                          BLO     ISLNO
5152  020574  062703  000012                 ADD     #12,R3
5153  020600  000767                          BR      LNOSRCH
5154  020602  021304         ISLNO:  CMP     (R3),R4
5155  020604  001414                          BEQ     FOUNDLN
5156  020606  062703  000004                 ADD     #4,R3
5157  020612  000762                          BR      LNOSRCH
5158                                  STOLNO:
5159                                          .IFDF   $NOSTK
5160  020614  062703  000004                 ADD     #4,R3
5161                                          .ENDC
5162
5163                                          .IFNDF  $NOSTK
5164                                          ADD     #6,R3           ;CHECK THAT THE SYMTAB SPACE
5165                                          CMP     R3,LOSTR(R5)    ;IS NOT OCCUPIED BY STRINGS
5166                                          BLO     STOAOK
5167                                          JSR     PC,UPPACK       ;IF IT IS, MOVE STRINGS UP
5168                                          CMP     R3,LOSTR(R5)    ;AND CHECK ROOM ENOUGH AGAIN
5169                                          BHIS    ERROV3
5170                                  STOAOK: TST     -(R3)           ;ADJUST SYMTAB ADDRESS
5171                                          .ENDC
5172
5173  020620  020365  000012                 CMP     R3,HIFREE(R5)
```

```
5174  020624  101351                  BHI     ERRQV3
5175  020626  710365  000014          MOV     R3,LOFREE(R5)
5176  020632  005043                  CLR     -(R3)
5177  020634  010443                  MOV     R4,-(R3)
5178  020636  005726          FOUNDLN:TST     (SP)+
5179  020640  000167  177422          JMP     RETLNO
5180  020644  005703          MAKFLIT:        TST     R3
5181  020646  001002                  BNE     CALNRM
5182  020650  005704                  TST     R4
5183  020652  001714                  BEQ     LITZERO
5184  020654  004767  174674  CALNRM: JSR     PC,NORM
5185  020660  112721  000374          MOVB    #'FLIT,(R1)+
5186  020664  000303                  SWAB    R3
5187  020666  220100                  CMP     R1,R0
5188  020670  103325                  BHIS    TRNER4
5189  020672  110321                  MOVB    R3,(R1)+
5190  020674  000303                  SWAB    R3
5191  020676  110321                  MOVB    R3,(R1)+
5192  020700  000706                  BR      LITCOMN
5193
5194
```



```
5195                                  .IFNDF  $NOSTR
5196                          ;+-----------------------------------------------
5197                          ;  SUBROUTINE TO UPPACK:  CALLED BY JSR PC
5198                          ;                        PACKS STRING STORAGE TOWARD HIGH CORE
5199                          ;                        R0  UNUSED
5200                          ;                        R1,R2,R3  PRESERVED
5201                          ;                        R4  UNUSED
5202                          ;                        R5  MUST POINT TO USER AREA
5203                          ;                        SP  GOES 1F DEEPER AFTER JSR
5204                          UPPACK: MOV     R1,-(SP)
5205                                  MOV     R2,-(SP)
5206                                  MOV     R3,-(SP)
5207                                  CLR     -(SP)
5208                                  MOV     HISTR(R5),R1
5209                                  MOV     HIFREE(R5),R2
5210                                  MOV     R2,HISTR(R5)
5211                          UPPLOOP:CLR     (SP)            ;GET END OF THE NEXT STRING
5212                                  BISB    -(R1),(SP)
5213                                  BNE     UPPNZRO         ;(LAST BYTE CONTAINS THE LENGTH)
5214                          UPPBAD: CMP     R1,LOSTR(R5)
5215                                  BHI     UPPLOOP
5216                                  MOV     R2,LOSTR(R5)
5217                                  TST     (SP)+
5218                                  MOV     (SP)+,R3
5219                                  MOV     (SP)+,R2
5220                                  MOV     (SP)+,R1
5221                                  RTS     PC
5222                          UPPNZRO:SUB     (SP),R1         ;ADDRESS BACK PTR
5223                                  CLR     R3
5224                                  BISB    -(R1),R3
5225                                  SWAB    R3
5226                                  BISB    -(R1),R3        ;GET IT IN R3
5227                                  SWAB    R3
5228                                  DEC     R1
5229                                  BIT     R3,#1           ;CHECK REL TO SYMBOLS
5230                                  BEQ     .+6
5231                                  DEC     R3
5232                                  ADD     (R5),R3         ;YES, ADD BASE OF SYM TAB
5233                                  CMP     R3,PDL(R5)      ;MAKE SURE IT'S NOT TOO HI
5234                                  BHIS    UPPBAD
5235                                  CMP     R3,SP           ;IN STACK IS OK
5236                                  BHIS    UPPGOOD
5237                                  CMP     R3,ARRAYS(R5)   ;IN ARRAYS IS GOOD
5238                                  BHI     UPPBAD
5239                                  CMP     R3,HIFREE(R5)   ;IN FREE STORAGE IS BAD
5240                                  BHI     UPPGOOD
5241                                  CMP     R3,LOFREE(R5)
5242                                  BHIS    UPPBAD
5243                                  CMP     R3,(R5)
5244                                  BLO     UPPBAD          ;BELOW SYMBOL TABLE IS BAD
5245                          UPPGOOD:ADD     #4,(SP)         ;GOOD STRING, MOVE IT UP
5246                                  CMP     (R3),R1
5247                                  BNE     UPPBAD          ;(DON'T GARBAGE COLLECT IT)
5248                                  ADD     (SP),R1
5249                                  SUB     R1,(R3)
```

```
5250                                    ADD     R2,(R3)
5251                                    MOVB    -(R1),-(R2)
5252                                    DEC     (SP)
5253                                    BGT     .-4
5254                                    BR      UPPBAD
5255                                    .ENDC
5256
```



```
5257                            ;-----------------------------------------------------------
5258                            ;  SUBROUTINE 'VAL'      CALLED BY JSR R7
5259                            ;                       CONVERTS AN ASCII STRING AT (R0)
5260                            ;                       TO A VALUE IN R3,R4, AND
5261                            ;                       AN EXPONENT IN R2
5262                            VAL:
5263 020702 005002                      CLR     R2              ;DEC PLACES+100000 OR TRAILING ZEROES.
5264 020704 005003                      CLR     R3              ;HIGH ORDER OF 32 BIT INTEGER;
5265 020706 005004                      CLR     R4              ;LOW ORDER OF 32 BIT INTEGER.
5266 020710 122027 000040       NUDIGIT:CMPB    (R0)+,#' 
5267 020714 001775                      BEQ     .-4
5268 020716 124027 000060               CMPB    -(R0),#'0
5269 020722 103427                      BLO     NOTDIG
5270 020724 121027 000071               CMPB    (R0),#'9
5271 020730 101024                      BHI     NOTDIG
5272 020732 020327 014630               CMP     R3,#14630       ;IF HIGH WORD GREATER THAN THIS
5273 020736 101405                      BLOS    CANFIT          ;THEN CANT FIT ANOTHER DIGIT IN 32 BITS,
5274 020740 005200                      INC     R0
5275 020742 005702                      TST     R2
5276 020744 002761                      BLT     NUDIGIT         ;AFTER POINT SO NEEDNT COUNT THEM,
5277 020746 005202                      INC     R2              ;CANT FIT DIGITS IN MANTISSA SO MUST
5278 020750 000757                      BR      NUDIGIT         ;KEEP TRACK OF TRAILING ZEROES;
5279 020752 004767 174460       CANFIT: JSR     PC,MPY10N
5280 020756 005046                      CLR     -(SP)
5281 020760 112016                      MOVB    (R0)+,(SP)      ;AND ADD IN THE DIGIT;
5282 020762 162716 000000               SUB     #'0,(SP)
5283 020766 062604                      ADD     (SP)+,R4
5284 020770 005503                      ADC     R3
5285 020772 005702                      TST     R2
5286 020774 001745                      BEQ     NUDIGIT         ;FITS IN MANTISSA,
5287 020776 005202                      INC     R2              ;FITS IN MANTISSA BUT AFTER POINT SO
5288 021000 000743                      BR      NUDIGIT         ;COUNT DECIMAL PLACES,
5289 021002 122027 000056       NOTDIG: CMPB    (R0)+,#'.
5290 021006 001024                      BNE     NOTDOT
5291 021010 005702                      TST     R2
5292 021012 001003                      BNE     DOTROT
5293 021014 012702 100000               MOV     #100000,R2      ;DOT COMES AFTER SHORT NUMBER SO GET
5294 021020 000733                      BR      NUDIGIT         ;READY TO COUNT DECIMAL PLACES;
5295 021022 003000              DOTROT: BGT     DOTIGNO
5296 021024 122027 000040       DOTIGNO:CMPB    (R0)+,#' 
5297 021030 001775                      BEQ     .-4
5298 021032 124027 000060               CMPB    -(R0),#'0       ;DOT COMES AFTER FULL MANTISSA WITH
5299 021036 103405                      BLO     PASTDOT         ;TRAILING ZEROES SO IGNORE MORE DIGITS,
5300 021040 121027 000071               CMPB    (R0),#'9
5301 021044 101002                      BHI     PASTDOT
5302 021046 005200                      INC     R0
5303 021050 000765                      BR      DOTIGNO
5304 021052 122027 000056       PASTDOT:CMPB    (R0)+,#'.       ;DOT COMES WHILE IGNORING TRAILING
5305 021056 001400                      BEQ     DOTBAD          ;AFTER POINT WITH FULL MANTISSA;ERROR,
5306                            DOTBAD:
5307 021060 005046              NOTDOT: CLR     -(SP)
5308 021062 124027 000105               CMPB    -(R0),#'E
5309 021066 001056                      BNE     NOEXPON         ;NO 'E' AFTER THE NUMBER
5310 021070 020165 000020               CMP     R1,LINE(R5)     ;IF ITS THE FIRST THING ON THE LINE IT
5311 021074 001453                      BEQ     NOEXPON         ;MUST BE A LINENO SO 'E' IS ILLEGAL,
```

```
 5312  021076  005200                      INC     R0
 5313  021100  121027  000040              CMPB    (R0),#BL
 5314  021104  001774                      BEQ     .-6
 5315  021106  122027  000053              CMPB    (R0)+,#'+
 5316  021112  001406                      BEQ     EXPDIG
 5317  021114  124027  000055              CMPB    -(R0),#'-
 5318  021120  001003                      BNE     EXPDIG
 5319  021122  005200                      INC     R0
 5320  021124  012716  100000              MOV     #100000,(SP)
 5321  021130  122027  000040      EXPDIG: CMPB    (R0)+,#BL
 5322  021134  001775                      BEQ     .-4
 5323  021136  124027  000060              CMPB    -(R0),#'0
 5324  021142  103423                      BLO     EXPDUN
 5325  021144  121027  000071              CMPB    (R0),#'9
 5326  021150  101020                      BHI     EXPDUN
 5327  021152  011646                      MOV     (SP),-(SP)
 5328  021154  006316                      ASL     (SP)
 5329  021156  006316                      ASL     (SP)
 5330  021160  066616  000002              ADD     2(SP),(SP)
 5331  021164  006316                      ASL     (SP)
 5332  021166  042766  077777  000002      BIC     #77777,2(SP)
 5333  021174  062616                      ADD     (SP)+,(SP)
 5334  021176  005046                      CLR     -(SP)
 5335  021200  112016                      MOVB    (R0)+,(SP)
 5336  021202  062616                      ADD     (SP)+,(SP)
 5337  021204  162716  000060              SUB     #'0,(SP)
 5338  021210  000747                      BR      EXPDIG
 5339  021212  005716              EXPDUN: TST     (SP)
 5340  021214  002003                      BGE     NOEXPON
 5341  021216  042716  100000              BIC     #100000,(SP)
 5342  021222  005416                      NEG     (SP)
 5343  021224  005702              NOEXPON:TST     R2
 5344  021226  002003                      BGE     EXPOK
 5345  021230  042702  100000              BIC     #100000,R2
 5346  021234  005402                      NEG     R2
 5347  021236  062602              EXPOK:  ADD     (SP)+,R2
 5348  021240  000207                      RTS     PC
 5349
```



```
 5350                                    ;
 5351                                    ; SYSTEM I/O BUFFERS AND HEADERS
 5352                                    ;
 5353
 5354
 5355
 5356                                            .IFNDF  $NOPTP
 5357                                    ;
 5358                                    ; PAPER TAPE READER
 5359                                    ;
 5360  021242  021256            PRBFHD: .       PRBUF   ;BSTRT
 5361  021244  021385                    .       PRBEND          ;BEND
 5362  021246  021256                    .WORD   PRBUF           ;BGET1
 5363  021250  000000                    .       0               ;BGET2
 5364  021252  021256                    .WORD   PRBUF           ;BPUT
 5365  021254  000000                    .WORD   0               ;BFSPEC
 5366          021256            PRBUF   .,
 5367          021306                    .,.+$PRBSZ
 5368          021385            PRBEND  .,.+1
 5369                                    .EVEN
 5370                                    ;
 5371                                    ; PAPER TAPE PUNCH
 5372                                    ;
 5373  021306  021322            PPBFHD: .       PPBUF           ;BSTRT
 5374  021310  021351                    .       PPBEND          ;BEND
 5375  021312  021322                    .WORD   PPBUF           ;BGET1
 5376  021314  000000                    .       0               ;BGET2
 5377  021316  021322                    .WORD   PPBUF           ;BPUT
 5378  021320  000000                    .WORD   0               ;BFSPEC
 5379          021322            PPBUF   .,
 5380          021352                    .,.+$PPBSZ
 5381          021351            PPBEND  .,.+1
 5382                                    .EVEN
 5383                                    .ENDC
 5384
 5385
 5386                                            .IFNDF  $NOLPT
 5387                                    ;
 5388                                    ; LINE PRINTER
 5389                                    ;
 5390  021352  021366            LPBFHD: .       LPBUF           ;BSTRT
 5391  021354  021425                    .       LPBEND          ;BEND
 5392  021356  021366                    .WORD   LPBUF           ;BGET1
 5393  021360  000000                    .       0               ;BGET2
 5394  021362  021366                    .WORD   LPBUF           ;BPUT
 5395  021364  000000                    .WORD   0               ;BFSPEC
 5396          021366            LPBUF   .,
 5397          021426                    .,.+$LPBSZ
 5398          021425            LPBEND  .,.+1
 5399                                    .EVEN
 5400                                    .ENDC
 5401
```

```
5402          021426                  ENDAB   =.
5403          000000'                         .CSECT
5404          000000'                 BEGIN   =.
5405          021426'                 .       =.+ENDAB
5406          000001                          .END
```



```
ADDINT  007430          ADDOVF  007450          ADDPOS  007524          ADDSTK  007370
ADDZER  007510          ADDSTK1 017326          ALLEXI  007714          ALLLOO  007600
ALLNOA  007610          ALLOC   007556          ALLOC1  002300          ALLOC2  002432
ALOG    = ****** G      AMATCH  017550          ARGB    011650 G        ARRAYS  = 000010 G
ASSIGN  003222          ATAN    = ****** G      ATNFN   011724          BEGIN   = 000000R
BEND    = 000002        BFSPEC  = 000012        BGET1   = 000004        BGET2   = 000006
BL      = 000040        BOMB    007742          BOMBDO  010014          BOMBJM  010126
BOMBNE  010062          BOMBNX  010002          BOMBOK  010112          BPDL    010636
BPUT    = 000010        BSTRT   = 000000        BUFCLR  010152          BUFEMP  013442
BUFTP   010206          CALL    003012          CALLCK  003070          CALLM1  003100
CALLM2  003122          CALLNM  003174          CALLP   003042          CALLX   003132
CALL1   003038          CALNRM  020654          CALOG   012644          CANFIT  020752
CARRET  020106          CCEXIT  010322          CCMES   007073          CEXP    012656
CHKCHR  010224          CHKISE  010324          CHKOSE  010332          CLEAR   001064
CLMNLP  000402          CLMNPP  000400          CLMNTT  = 000036        CLRECH  006732
CLRFRE  010500          CLRLOO  010442          CLROK   010470          CLRVAR  010424
CNCFLG  = 000106        CNOFLG  = 000107        CODE    = 000016        COLONC  005204
COLUMN  = 000034 G      COMCHK  010336          COMES   007076          COMTAB  017502
COS     = ****** G      COSFN   011702          CR      = 000015        CRLFME  007070
DEFDUN  002306          DEFGOT  002350          DELMSG  007060          DEVERR  017044
DIMDON  002602          DIMGOT  002314          DIMLOO  002234          DIMNON  002254
DIMSCL  002462          DIVLOO  010564          DIVTEN  010560          DOALLO  002444
DOCHAR  014404          DOITNO  011400          DOMSG   015526          DOPUT   006562
DOTBAD  021060          DOTIGN  021024          DOTROT  021022          ECHORA  007052
ECHOSP  = 000104        EDICHG  001410          EDICON  001472          EDIGRO  001556
EDIJMP  001744          EDIMME  001262          EDIMOD  001450          EDIMOV  001502
EDIMVU  001632          EDIOVE  001430          EDIPAS  001340          EDIPUT  001434
EDIRLC  001700          EDIRLO  001736          EDIROO  001620          EDISAM  001660
EDISKI  001274          EDISRC  001300          EDIT    001100          EDITL   001222
END     003254          ENDAB   = 021426        ENDCR   017700          ENDQUO  017660
ENDSQU  017700          EOF     003254          ERADD   011606          ERCHAN  010410
EREXP   011666          ERLOG   007734          ERPD02  011656          ERPD11  011762
ERRAG1  013012          ERRARA  007726          ERRARG  007734 G        ERRDAT  006402
ERRDEE  004040          ERRDEF  002542          ERRDIM  002534          ERRDIV  012230
ERREXP  012516          ERRFN1  003164          ERRFPU  = ****** G      ERRGO   004046
ERRMIX  011650 G        ERRNEX  005144          ERROVR  011606          ERROV1  001614
ERROV2  020344          ERROV3  020550          ERRPDL  011120 G        ERRPD2  011662
ERRPD3  003666          ERRPD4  012670          ERRPD5  011464          ERRRET  004126
ERRRUN  002634          ERRSS1  013742          ERRSS2  011234          ERRSS3  014774
ERRSXA  012674          ERRSX1  003170          ERRSX2  005152          ERRSX3  013736
ERRSX4  011066          ERRSX5  011650          ERRSX6  004054          ERRSX7  003536
ERRSX8  006234          ERRSX9  012122          ERRSYN  011650 G        ERRTRN  020036
ERRTR7  020330          ERRTR8  006242          ERRUFN  013016          ERR4WO  004404
ERSQR   007734          ERSTAR  011666          ERSX10  004506          ERSX12  010404
EVAL    010616 G        EXECUT  002654          EXIT02  007104          EXP     = ****** G
EXPDIG  021130          EXPDUN  021212          EXPFN   011732          EXPNEG  020402
EXPOK   021236          EXPTEN  020360          FAC1    = 000040 G      FAC2    = 000042 G
FF      = 000014        FILLCH  = 000112        FILLCO  = 000110 G      FILLNO  = 000111
FIXCLE  013250          FIXRET  013252          FIXTST  013230          FIXUP   013174
FLE     013276          FLINE   013254          FLIT    011032          FLX     013302
FNCHK   013162          FNFN    012700          FNNOCO  013024          FNNUME  013170
FNRCOM  013144          FNREPL  013000          FNRLUP  013100          FNRSNU  013160
FNRSSC  013132          FNSWAP  012724          FOR     004134          FORGO   004622
FORGOG  004620          FORGTL  004560          FORLOO  004312          FORLTL  004572
FORNEX  004412          FORNOL  004336          FORONE  004276          FORSKI  004306
```

```
FORGER  004576        FOUNDL  020636        FOUNDV  020222        FPPRES  013306
FPPSAV  013334        FRSTOU  015532        FUNCTI  011700        FUNOK   012024
FUNRET  012056        GCWAIT  013570        GETBYT  013374        GETCHA  013450
GETCH1  013452        GETVAR  013604 G      GETX    013466        GET2BY  013362
GET2BY  013370        GOEXEC  006214        GONOSA  004432        GOSUB   003754
GOTECH  006770        GOTO    003754        GOTOFN  011126        GOTONE  014414
GSBCTR = 000032       HORSTU  016740        HIFREE = 000012       HISTR  = 000074
IDEV   = 000077       IF      003316        IFCOMP  003542        IFLEND  003450
IFLEQR  003472        IFLGTR  003676        IFLLTR  003622        IFLOOP  003420
IFNUME  003552        IFSEQ   003466        IFSGT   003672        IGNORE  002650
ILITCO  011042        ILIT1   011004        ILIT2   011024        IMMED   001750
IMMSTM  002032        INITBF  013746        INITSC  013704        INPEND  006170
INPEOL  006240        INPGOO  006162        INPLOO  005736        INPNEW  005764
INPNGU  006220        INPOK   006104        INPPC   005732        INPRTR  005754
INPSTO  006136        INPUT   005650        INPYES  005672        INPYN1  005730
INT     014004 G      INTEXT  017122        INTFN   012100        INTRTS  014136
INT1    014074        INT1A   014106        INT1B   014124        INT2    014126
INT2A   014132        INT3    014140        INT6    014100        INT7    014220
INTB    014232        INT9    014242        IOINIT  001154        IOWAIT  014312
ISLNO   020602        ISNUM   020310        ISVAR   020116        ITAB    013620
ITABLE  010421        ITSIN   017000        JMPRDY  003332        KBCO    006542
KBHD   = 000102       KBIDUN  006626        KBINT   006470        KEYA    000476
KEYWDS  000404        LET     003210        LEVFLT  014542        LEVFNE  014604
LEVLIT  014516        LEVPLU  014514        LEVPOS  014612        LEV1LT  014560
LEV2LT  014554        LF     = 000012       LIMIT  = 000002 G     LINA    014350
LINDUN  015540        LINE   = 000020       LINEIN  014370        LINEND = 000030
LINGET  014344        LINRUB  014374        LIST    002200        LISTSV  002170
LITCOM  020516        LITDIV  013664        LITEVA  014404        LITNOR  015574
LITOK   015622        LITSMR  015674        LITSTO  015700        LITTST  015612
LITZER  020504        LNOSRC  020560        LOCALO  014634        LOCGET  014620
LOCLOO  014730        LOCNO2  014756        LOCSS1  014666        LOFREE = 000014
LOGFN   011746        LOSTR  = 000072       LPAR    011054        LPR    = 177516
LPBEND = 021425       LPBFHD  021352        LPBUF  = 021306        LPERR   007314
LPINT   007256        LPOFF   007322        LPS    = 177514        LSTBSL  015174
LSTCHA  015200        LSTCHK  015074        LSTCRL  015142        LSTEOL  015130
LSTFLB  015306        LSTFLI  015266        LSTFN   015110        LSTILB  015260
LSTIL1  015236        LSTIL2  015250        LSTJMP  015312        LSTKWD  015062
LSTLIN  015154        LSTLNO  015350        LSTLNX  015400        LSTLOO  015006
LSTNEX  015122        LSTPC   015044        LSTPRO  015002        LSTPTR  015316
LSTSPE  015206        LSTSRC  015050        LSTTEX  015224        LSTVAR  015412
MAKE12  020512        MAKEOK  020530        MAKEST  011650 G     MAKFLI  020644
MAKLNO  020554        MINUS   011470        MPYTEN  015436        MSG     015474 G
MSGCOM  015476        MSGERR  015466        MSGODE  015522        MSGSUP  006746
NEXEND  005070        NEXFIL  020012        NEXGO   005106        NEXGTL  005052
NEXLTL  005062        NEXT    004644        NOCHAR  013472        NOCHR2  007250
NOCNC   002672        NODIGI  020150        NOEXPO  021224        NOOM    006000
NOQM1   006014        NOQUOT  010716        NORM    015594 G      NORNOO  016310
NOROOM  016670        NOSUBS  013734        NOTDIG  021002        NOTOOT  021060
NOTIT   020204        NOTKWD  020042        NOTNOW  011434        NOTSYM  002770
NOWRAP  016636        NOWRP   013434        NUDIGI  020710        NUMALN  016240
NUMBIG  016222        NUMDIG  016314        NUMDIV  016230        NUMEXI  016576
NUMFIX  016024        NUMFLT  016122        NUMFM1  016376        NUMFM2  016504
NUMFM3  016520        NUMJMP  017452        NUMLP1  016414        NUMLP3  016554
NUMNEG  016052        NUMNOR  016154        NUMOK   016240        NUMOUT  015746 G
NUMPOS  016060        NUMPRS  016064        NUMSGN  015700 G     NUMSHF  016146
```



```
NUMSTT  016066        ODEV   = 000076       OFFTYP  006700        OLD     002100
OLD001  002136        OPERAN  010640        OPRATO  011272 G      OPRFN   012116
OTABLE  010416        OUT010  017024        OUT012  010402        PASTOO  021052
PC     =X000007       POL    = 000004 G     POSIZE = 000006 G     PLUS    011934
PNOGO   017146        POP     014270        POWDWN  007330        POWLP   007342
POWUP   007340        PPB    = 177556       PPBEND = 021351        PPBFHD  021306
PPBUF  = 021322       PPERR   007242        PPINT   007204        PPS    = 177554
PRB    = 177552       PRBEND = 021305       PRBFHD  021242        PRBUF  = 021256
PREC2   011372        PREC3   011364        PREC4   011356        PREC5   011350
PREOT   007174        PRIBOT  005334        PRICM   005326        PRICOM  005356
PRIJMP  005452        PRIMOR  005350        PRINCR  005632        PRINT   005156
PRISEM  005434        PRISJ   005504        PRISTR  005476        PRIST1  005506
PRITB   005550        PRITB0  005600        PRITB1  005614        PRITES  005444
PRNTDE  007036        PRNT01  005220        PRS    = 177550       PRS    = 000140
PR4    = 000200       PR7    = 000340       PS     = 177776       PTRINT  007104
PUSH    014256        PUSHF   011634        PUSH1   014302        PUTBYT  016604
PUTCCO  016724        PUTCHA  016676        PUTIT   016754        PUTIT1  020230
QUOTPA  017614        READ    006246        READBA  006410        READOU  006332
READGO  006314        READOU  006376        READY   001076        READY0  001072
READY2  001140        REAFIN  006426        REASRC  006422        REENAB  013346
REMPAC  020064        RESREG  017104        RESTOR  003742        RETLNO  020266
RETRY   017510        RETURN  004060        RETVAR  020202        REVRSE  012622
RNDCT  = 000070       RNDGOT  002506        RNO1   = 000004 G     RNO2   = 000006 G
RUN     002224        R0     =X000000       R0SAVE = 000044        R1     =X000001
R1SAVE = 000046       R2     =X000002       R2SAVE = 000050        R3     =X000003
R3SAVE = 000052       R4     =X000004       R4SAVE = 000054        R5     =X000005
SAVCHA  011650 G      SAVE    002142        SAVREG  017002        SAVRGI  017052
SCANPN  017124        SCRATC  001054        SETLFE  007044        SIN    = ****** G
SINFN   011674        SKIPEO  017160        SKIPHI  017224        SKIPRT  017242
SKIPTX  017234        SKIP01  017156        SKIP02  017154        SKIP35  017150
SLASH   011972        SOPRAT  011650 G      SP     =X000006       SPARE  = 000113
SPECME  006702        SQRFN   011710        SQRT   = ****** G     SS1SAV = 000024
SS2SAV = 000026       STAR    011542        START   001034 G      STAR1   011566
STOCOM  017300        STOLNO  020614        STONOS  017270        STOP    003276
STOSTR  017642        STOSVA  011650        STOTXT  020074        STOVAR  017244 G
STPRO   011650 G      SUBINT  017332        SUBSTK  017312        SYMBOL = 000000
S,0101  006604        S,0102  006622        TAB    = 000011       TABLE1  002710
TABLE2  011416        TABLE4  002016        TABLE5  010750 G      TABL5  = 005320
TAB1    017030        TAB2    017036        TAB3    014460        TBL1EN  002766
TBL4EN  002030        TBL5EN = 011002 G     TERMIN  011400        TKB    = 177562
TKS    = 177560 G     TPB    = 177566 G     TPCN    007006        TPCO    007022
TPHD   = 000100       TPINT   006642        TPNXT   006670        TPS    = 177564
TRAN    017354        TRANLU  017402        TRANVA  017406        TRENRA  017772
TRNER4  020544        TRNFN   017734        TRNPC   017504        TRVACN  017506
TRYECH  006734        TRYKWD  017456        TRYNUM  017430        TSTSTK  011620
TSTS1   011624        TSTS2   011632        TYPE    006710        TYPE1   006724
T1     = 000056 G     T2     = 000060 G     T3     = 000062 G     UAASR   012524
UAJMP   012536        UANJMP  012532        UATSTA  012502        UATST0  012542
UATST2  012552        UA1     012152        UA10    012202        UA10A   012320
UA10B   012342        UA10C   012352        UA10,5  012364        UA12    012402
UA17    012412        UA17A   012424        UA17B   012442        UA17D   012452
UA17F   012472        UA19    012510        UA20    012570        UA21    012516
UA23    012600        UA4     012202        UA5     012210        UAR     012222
UA9     012250        UINTEG  011512        UMINUS  010630        UNARY   011474
UPARRO  012126        USRARE = ****** G     VAL     020702 G      VARBLE  011072
```

```
VARNOS  011262        VARSAV = 000022        VARSRC  020162        VARSS   011132
VERNUM  000040        VONESS  011240        VTWOSS  011252        WTRET   014330
XCHAR   013510        $ADR   = ******  G     $DVR   = ******  G     $ERVEC = ******  G
$IR    = ******  G     $LPBSZ = 000040        $HLR   = ******  G     $NOSTR = 000001
$POLSH = ******  G     $PPBSZ = 000030        $PRBSZ = 000030        $PPORG = 000400
$SBR   = ******  G     $STKSZ = 000200  G     ,ABS   = 000273        ,AMPER = 000226
,ATN   = 000270        ,CALL  = 000224        ,CLEAR = 000304        ,COLON = 000260
,COMMA = 000243  G     ,COS   = 000266        ,DATA  = 000223        ,DEF   = 000221
,DIM   = 000215        ,DQUOT = 000256        ,EQ    = 000247        ,EL    = 000245
,EN    = 000251        ,END   = 000202        ,EOF   = 000225        ,EOL   = 000201  G
,EQ    = 000254        ,EXP   = 000271        ,FLIT  = 000374        ,FN    = 000262
,FOR   = 000203        ,GE    = 000246        ,GOSUB = 000204        ,GOTO  = 000205
,GT    = 000253        ,IF    = 000206        ,ILIT1 = 000375        ,ILIT2 = 000376
,INPUT = 000207        ,INT   = 000274        ,LE    = 000244        ,LET   = 000210
,LIST  = 000277        ,LOG   = 000272        ,LPAR  = 000255  G     ,LT    = 000252
,MINUS = 000234        ,NE    = 000250        ,NEXT  = 000211        ,NVAR  = 177776
,OLD   = 000302        ,PLUS  = 000232        ,POUND = 000201        ,PRINT = 000212
,RANDO = 000216        ,READ  = 000222        ,REM   = 000220        ,RESTO = 000217
,RETUR = 000213        ,RND   = 000264        ,RNDL  = 000203        ,RPAR  = 000237  G
,RUN   = 000300        ,SAVE  = 000301        ,SCALA = 177775        ,SCR   = 000303
,SEMI  = 000236        ,SGN   = 000275        ,SIN   = 000265        ,SLASH = 000231
,SQR   = 000267        ,SQUOT = 000257        ,STAR  = 000230        ,STEP  = 000241
,STOP  = 000214        ,SVAR  = 177777        ,TAB   = 000276        ,TERM  = 000235
,TEXT  = 000377        ,THEN  = 000242        ,TO    = 000240        ,UNARY = 000233
,UPARR = 000227        ,      = 021426R
```

ERRORS DETECTED: 0

RUN-TIME: 72 SECONDS
CORE USED: 4K

```
       1895  1897  1899  1901  1903  1914  1916  1917  1919  1921  1947  1948  1950  1951
       1953  1956  1958  2088  2229  2239  2245  2255  2258  2265  2278  2286  2324  2325
       2451  2453  2454  2455  2480  2482  2483  2508  2511  2516  2521  2522  2547  2548
       2552  2648  2650  2652  2655  2896  2902  2916  2917  2919  2921  2932  2934  2936
       2938  2957  3203  3391  3394  3508  3510  3626  3638  3696  3658  3662  3663  3665
       3666  3667  3669  3798  3792  3794  3940  3976  3978  3981  3982  3984  3986  3989
       3998  3991  3992  3993  4020  4026  4029  4031  4036  4040  4044  4046  4049  4054
       4096  4097  4098  4099  4262  4265  4267  4269  4270  4272  4318  4321  4324  4327
       4337  4339  4341  4343  4352  4354  4362  4363  4364  4365  4418  4426  4431  4432
       4435  4436  4440  4442  4444  4446  4458  4462  4466  4467  4469  4473  4480  4484
       4485  4491  4496  4538  4574  4575  4578  4578  4579  4581  4584  4686  4696  4772
       4884  4885  4887  4889  4896  4898  4995  4999  5001  5003  5005  5012  5026  5040
       5043  5045  5046  5047  5048  5049  5050  5064  5065  5066  5067  5068  5099  5107
       5111  5113  5126  5147  5148  5150  5152  5154  5156  5160  5173  5175  5176  5177
       5180  5186  5189  5190  5191  5264  5272  5284
R3SAVE  331#  332  3221  3397  3399  3626  3638
R4      274#  707  735  743  749  758  817  820  852  949  955  960  962  964
       1016  1139  1145  1288  1272  1801  1805  1806  1807  1814  2015  2035  2037  2039
       2042  2065  2066  2067  2069  2071  2077  2096  2100  2103  2106  2233  2279  2323
       2651  2739  2744  2954  2976  2984  2995  2999  3208  3214  3223  3227  3253  3314
       3329  3362  3363  3364  3365  3374  3377  3396  3412  3438  3586  3594  3628  3640
       3866  3884  3887  3890  4019  4027  4029  4032  4098  4103  4263  4264  4266  4268
       4271  4322  4326  4330  4338  4340  4342  4344  4353  4366  4416  4421  4434  4441
       4443  4445  4447  4459  4464  4478  4495  4498  4522  4526  4528  4530  4531  4541
       4610  4614  4615  4627  4685  4697  4870  4883  4885  4889  4993  4994  4996  4998
       4918  4912  4914  4916  4917  4921  4929  4940  5109  5127  5129  5134  5137  5138
       5139  5154  5177  5182  5265  5283
R4SAVE  332#  333  3627  3639
R5      275#  640  641  642  644  654  655  656  657  658  664  669  682  688
        694  697  703  715  729  733  736  747  748  763  771  773  775  781
        789  790  791  885  886  889  811  819  833  891  852  853  870  874
        875  881  913  918  919  936  980  981  1006  1013  1016  1017  1036  1038
       1071  1074  1140  1144  1162  1186  1274  1275  1319  1332  1334  1335  1338  1364
       1368  1370  1397  1402  1409  1419  1420  1421  1422  1436  1444  1460  1461  1478
       1479  1486  1487  1488  1489  1490  1491  1492  1493  1495  1497  1498  1499  1510
       1529  1538  1546  1552  1553  1559  1561  1563  1564  1572  1575  1585  1586  1587
       1608  1635  1677  1701  1702  1703  1710  1728  1729  1745  1750  1742  1763  1766
       1777  1782  1793  1802  1807  1810  1810  1818  1826  1827  1844  1846  1893  1895
       1921  1927  1937  1947  2012  2013  2014  2021  2026  2058  2059  2060  2062  2066
       2072  2074  2078  2082  2129  2156  2157  2175  2176  2199  2201  2238  2242  2244
       2249  2250  2252  2253  2254  2256  2257  2259  2260  2262  2263  2264  2303  2306
       2311  2351  2404  2405  2406  2407  2408  2420  2426  2428  2429  2431  2432  2434
       2435  2436  2450  2479  2543  2545  2548  2601  2602  2603  2604  2627  2629  2630
       2631  2632  2633  2636  2806  2807  2808  2810  2812  2813  2814  2815  2835  2878
       2884  2892  2896  2903  2914  2915  2953  2956  2964  2966  2968  2970  2971  2996
       2997  3219  3220  3225  3226  3258  3260  3262  3268  3284  3285  3289  3291  3305
       3308  3309  3318  3323  3360  3366  3369  3397  3399  3411  3428  3441  3443  3518
       3534  3583  3585  3591  3610  3623  3624  3625  3626  3627  3635  3636  3637  3638
       3639  3708  3712  3734  3736  3737  3740  3750  3764  3766  3809  3811  3812  3819
       3821  3833  3843  3850  3851  3860  3861  3862  3874  3876  3878  3879  3882  3883
       3885  3886  3902  3903  3925  3926  3930  3932  3933  3946  3974  3975  3992  3991
       3992  3993  3996  3998  4000  4085  4129  4144  4145  4147  4148  4149  4151  4152
       4153  4154  4159  4164  4165  4167  4168  4170  4171  4172  4296  4297  4298  4299
```

```
       4301  4309  4374  4389  4391  4394  4396  4398  4400  4402  4403  4416  4418  4674
       4606  4607  4610  4612  4617  4628  4690  4691  4698  4716  4763  4770  4772  4778
       4779  4855  4859  4863  4973  4995  4999  5043  5045  5063  5064  5111  5147  5148
       5173  5175  5310
READ   1065  1889#  1923
READBA 1902  1913  1920  1937#
READOU 1916#
READGO 1903#  1952
READOU 1927#  1954
READY   647   654#   900   909  1039  1199  2441  3905  4629
READY0  653#   683  2204
READY2  664#  1029  1761  5088  5098
REAFIN 1900  1948#  1960
REASRC 1898  1947#
REENAB 3697  3712#
REMPAC 4905  4959  4970  4972  4975#  5078
RESREG 4693#
RESTOR 1062  1319#
RETLNO 5064#  5179
RETRY  4887#  4888  4894  4897
RETURN 1058  1362#
RETVAR 5027  5063#
REVRSE 3380  3388#
RND1     81   336#   337   980   981  2601
RND2     81   337#   338  2602
RNDCT   338#   339   980  3902  4628
RNDGOT  926   980#
RUN     864   912#
SAVCHA   80  3002#
SAVE    865   892#
SAVREG 3821  3925  4374  4389  4604  4685#
SAVRGI 2012  2058  2129  2156  2175  4683#
SCANPN  884   892  4716#
SCRATC  644#   887
SETLFE 2090  2106#
SIN      63  3174
SINFN  2784  3174#
SKIP01 4732#  4734  4749
SKIP02 4731#  4741  4748
SKIP05 4730#  4736
SKIPEO  706   721   838   930  1035  1438  1957  4733#  4743  4745  4752
SKIPH1 4737  4747#
SKIPRT 4739  4753#
SKIPTX 4750#  4751
SLASH  2948  2983#
SOPRAT   77  3005#
SP      276#   641   644   654   664   687   689   698   700   746   759   821   850   881
       1114  1119  1120  1126  1135  1138  1139  1140  1144  1145  1146  1151  1152  1153
       1154  1208  1210  1217  1218  1219  1220  1221  1224  1236  1243  1247  1249  1250
       1265  1266  1272  1274  1275  1276  1283  1305  1306  1391  1392  1484  1485  1494
       1495  1497  1498  1499  1504  1508  1510  1515  1516  1517  1518  1519  1555  1556
       1557  1558  1559  1561  1563  1564  1565  1566  1567  1568  1580  1582  1583  1584
       1585  1586  1771  1772  1779  1788  1792  1800  1801  1814  1815  1816  1832  1834
```

```
$ERVEC      64   2232   2975   2983   3179   3184   3187   3257
$IR         59   2994   3254   3378   3587   3595
$LONGE     986    989    995    998   1021   1024   1342   1345   1351   1354   1381   1384   1464   1467
          1591   1594   1793   1796   1849   1852   1929   1932   1939   1942   2356   2359   2385   2388
          2411   2560   2563   2854   2857   3013   3019   3165   3168   3271   3274   3351   3354   3489
          3492   3950   3953   4068   4071   4291   4660   4663   5080   5083   5090   5093
$LPBSZ     103   5397
$MLR        60   2979   3334   3337   3384
$NOLPT     250    253   2171   2464   2577   2580   4645   4649   5386
$NOPOW     215    219   2191
$NOPTP     241    246   2121   2150   2458   2571   2574   2585   2588   3983   4638   4641   5356
$NOSTR      2#    436    447    581    795   1082   1092   1168   1177   1213   1279   1371   1376   1485
          1415   1429   1620   1631   1652   1706   1795   1820   1861   1925   1962   2206   2315   2329
          2365   2609   2617   2659   2763   2794   2819   2827   2839   2849   2864   2873   2887   2947
          3000   3023   3052   3149   3198   3197   3229   3235   3403   3419   3430   3446   3520   3537
          3558   3743   3759   3773   4022   4058   4182   4185   4192   4407   4546   4745   4782   4786
          5007   5017   5029   5039   5052   5159   5163   5195
$POLSH      58   3866
$PPBSZ      95   5380
$PRBSZ      99   5367
$PRORG      91    261
$SBR        62   3869
$STKSZ      82     87
```

```
BASICL  MACX11   V021   22-MAR-73   15:41   PAGE 2

      2                               ;  BASIC/PTS  PART1-BASICL
      3                               ;
      4                               ;  DEC-11-LPTBA-A-LA1
      5                               ;
      6                               ;  COPYRIGHT 1973
      7                               ;
      8                               ;  DIGITAL EQUIPMENT CORPORATION
      9                               ;  MAYNARD,MASSACHUSETTS 01754
     10
```

```
                        .TITLE  BASICL  V001A EDIT #61 (REF) 03/01/73
                                                SOURCE FILE #1
000000                  .ASECT
;
;
;
; BASIC CONSISTS OF THREE MODULES:
;       1) BASICL - INTERPRETER WITH SYSTEM I/O
;       2) FPMP-11 - MATH PACKAGE CONDITIONALIZED FOR USE IN BASIC
;       3) BASICH - USER AREA AND ONCE-ONLY CODE
;
;
;
;
;
;
;
; ORIGINAL PAPER TAPE VERSION   BY: LEN ELEKMAN
;
; MODIFICATION AND COMMENTS     BY: ROY FOLK
;
; COMPLETION AND FURTHER
; COMMENTS                      BY: ANN STANKARD
;
```



```
;*************************************************
;*************************************************
;*************                          **********
;*************  GLOBALS: ASSY, PARAMS,  **********
;*************  DESCRIPTION: LOW CORE:  **********
;*************  GENERAL TABLES,         **********
;*************  CONSTANTS, STORAGE      **********
;*************                          **********
;*************************************************
;*************************************************
;
;
;
;
; GLOBALS
;
; --FPMP-11
;
        .GLOBL  $POLSH
        .GLOBL  $IR
        .GLOBL  $MLR
        .GLOBL  $DVR
        .GLOBL  $ADR,$SBR
        .GLOBL  $IN,COS,SQRT,ALOG,ATAN,EXP
        .GLOBL  $ERVEC
;
; --BASIC2
;
        .GLOBL  TPB, IKS
        .GLOBL  START
        .GLOBL  USRAREA
        .GLOBL  FILLCO
        .GLOBL  LIMIT, POL, ARRAYS, PDSIZE
        .GLOBL  ERRPOL, ERRMIX, ERRSYN, ERRAMG
        .GLOBL  TABLES, TBLSEN
        .GLOBL  EVAL, GETVAR, STOVAR, MSG
        .GLOBL  COLUMN, FAC1, FAC2, .RPAR, .LPAR
        .GLOBL  INT, MAKEST, OPRATO, SCPMAT
        .GLOBL  .COMMA, T1, T2, T3, .EOL
        .GLOBL  NUMOUT, NUMSGN, ARGB, STPRO
        .GLOBL  SAVCHAR,VAL,NORM
        .GLOBL  RND1, RND2
        .GLOBL  $STKSZ, ERRFPU
```

```
                        ;
                        ; ASSEMBLY PARAMS,
                        ;
                                ,IFNDF  $STKSZ  ;SIZE OF STACK
                        $STKSZ  =200            ;BYTES
                                ,ENDC
                        ;
                                ,IFNDF  $PRORG  ;PROG, ORIGIN AFTER VECTOR SPACE
                        $PRORG  =400
                                ,ENDC
                        ;
                                ,IFNDF  $PPBSZ  ;PP BUFF, SIZE
                        $PPBSZ  =38
                                ,ENDC
                        ;
                                ,IFNDF  $PRBSZ  ;PAPER-TAPE READER BUFF, SIZE
                        $PRBSZ  =38
                                ,ENDC
                        ;
                                ,IFNDF  $LPBSZ  ;LINE-PRINTER BUFF, SIZE
                        $LPBSZ  =40
                                ,ENDC
                        ;
                        ;  $NOPTP = NO PAPER TAPE; DEFINE TO ELIMINATE CODE FOR HIGH SPEED
                        ;   PAPER TAPE
                        ;
                        ;  $NOLPT = NO LINE PRINTER; DEFINE TO ELIMINATE LINE PRINTER CODE
                        ;
                        ;  $LONGER = LONG ERROR MESSAGES; DEFINE TO GET LONG,
                        ;       EXPLANATORY ERROR MESSAGES,
                        ;
                        ;  $NOSTR = NO STRINGS; DEFINE TO ELIMINATE STRING VARIABLE CODE
                        ;
                        ;  $NOPOW = NO POWER FAIL; DEFINE TO ASSEMBLE WITHOUT
                        ;       POWER FAIL/RESTART ROUTINE,
                        ;
                        ;
                        ;
```



```
                        ;
                        ; HOT POOP!!
                        ;
                        ; **USER AREA MAP
                        ;
                        ; HIGH ADDRESSES**+----------------+
                        ;                | GOSUB POINTERS| <--LIMIT(R5) NEEDED
                        ;                | 26 FN POINTERS|
                        ;                | READ POINTER  | <--PDL(R5) NEEDED
                        ;                +----------------+
                        ;                |  PUSH          |
                        ;                |  DOWN          |    PDSIZE(R5) NEEDED =
                        ;                |  LIST          |  (PDL(R5)-ARRAYS(R5))/2+STACK(INTERRUPT)
                        ;                +----------------+
                        ;                |                | <--ARRAYS(R5) NEEDED
                        ;                |  ARRAYS        |
                        ;                |                |
                        ;                |          \!/   |
                        ;                |           V    |
                        ;                +----------------+
                        ;                |   STRINGS      |
                        ;                +----------------+
                        ;                |           A    |
                        ;                |          /!\   |
                        ;                |                |
                        ;                | SYMBOLS        |
                        ;                +----------------+
                        ;                |           A    |
                        ;                |          /!\   |
                        ;                | INTERP,        |
                        ;                | CODE           | <--CODE(R5) NEEDED
                        ;                +----------------+
                        ;                |   USER         |
                        ;                |   LINE         |
                        ;                |   BUFFER       | <--LINE(R5) NEEDED
                        ;                +----------------+
                        ;                | USER I/O BUFF,|
                        ;                | AND BUFF, WOR,|
                        ;                | SPACE (TTY)    |
                        ;                +----------------+
                        ;                |  USER AREA     |
                        ;                | STORAGE CELLS |
                        ; R5 POINTS-->   |                |
                        ;                +----------------+
                        ;
```

```
                                ;
                                ; *-
                                ;
                                ; CODE AREA CONTAINS STRING OF CODE BYTES WHICH COMPRISE
                                ;  THE COMPILED CODE TO BE INTERPETED.  THE SYNTAX SCAN OF THESE
                                ;  CODE BYTES IS FACILITATED BY THE JUDICIOUS PLACEMENT
                                ;  OF CERTAIN OF THESE CODE BYTES TO BE CALLED 'TOKENS'.
                                ;  IF THE HIGH BIT OF A TOKEN IS ON
                                ;  IT IS A SYSTEM SYMBOL (S,SYM), (SEE TABLE, BELOW.)
                                ;  HIGH BIT OFF SIGNALS TOKEN AS THE HIGH ORDER BYTE OF A WORD
                                ;  (LOW ORDER BYTE FOLLOWS TOKEN) USED AS AN ADDRESS OFFSET INTO
                                ;  THE USER SYMBOL TABLE (USPTR).
                                ;
                                ; *-SYMBOL TABLE ENTRIES
                                ;
                                ; ARE MADE FOR LINE NUMBERS, SCALARS, NUMERIC ARRAYS, AND
                                ;  STRINGS, AS SHOWN (IN ORDER):
                                ;
                                ;       LINENO  177775  177776  177777
                                ;       ADDRESS VALUE   ADDRESS ADDRESS
                                ;               VALUE   MAXSS1  MAXSS1
                                ;               0       MAXSS2  MAXSS2
                                ;               VARNAME VARNAME STRINGNAME
                                ;
                                ; *-GENERAL REGISTER USAGE
                                ;
                                ; R1 IS BASIC EXECUTION PC
                                ; R4 IS THE EXECUTION STACK DEPTH
                                ; R5 IS THE POINTER TO THE USER AREA
```



```
                                ;
                                ; INTERUPT VECTORS AND LOWEST CORE
                                ;
000000                          .       =0
000000  000167  001076                  JMP     START           ;RESTART AT LOC. #0
000004                          .       =4
000004  000006  000000                  +       ,+2,0           ;TIME OUT, BUSS ERROR
000010                          .       =10
000010  000012  000000                  +       ,+2,0           ;RESERVED INSTRUCTION
000014                          .       =14
000014  000016  000000                  +       ,+2,0           ;DEBUG, TRAP VEC,
000020                          .       =20
000020  010320  000000                  +       BOMB,0          ;IOT TRAP VEC,
                                ;
000024                          .       =24
                                ;
                                ;
                                        ;IFNDF  $NOPOW
                                        +       POWDWN,0        ;POWER FAIL/RESTART
                                        ;ENDC
                                ;
                                        ;IFDF   $NOPOW
                                        +       ,+2,0   ;HALT ON POWER FAIL
                                        ;ENDC
                                ;
000030                          .       =30
000030  000032  000000                  +       ,+2,0           ;EMT TRAP
000034                          .       =34
000034  000036  000000                  +       ,+2,0           ;'TRAP' TRAP
                                ;
                                ;  *-'SYSTEM' AREA
                                ;
000040                          .       =40
000040    001    101            VERNUM: .BYTE   001,'A          ;VERSION NUMBER
000042  000000  000000  000000          +       0,0,0,0,0,0,0,0
000050  000000  000000  000000
000056  000000
                                ;
                                ; *-INTERUPT VECTORS
                                ;
000060                          .       =60
000060  006722  000200                  +       KBINT,PR4       ;KEYBOARD
000064                          .       =64
000064  007074  000200                  +       TPINT,PR4       ;TELEPRINTER
000070                          .       =70
                                        ;IFNDF  $NOPTP
000070  007336  000200                  +       PTRINT,PR4      ;HSP, PT, READER
000074                          .       =74
000074  007436  000200                  +       PPINT,PR4       ;HSP, PT, PUNCH
                                        ;ENDC
                                        ;IFDF   $NOPTP
                                        +       0,0,0,0
                                        ;ENDC
000200                          .       =200
                                        ;IFNDF  $NOLPT
000200  007510  000200                  +       LPINT,PR4
```

```
                                        .ENDC
                                        .IFDF   $NOLPT
                                        *       0,0
                                        .ENDC
                000240                  .       =240
000240  000240  000242  000000          *       .+2,0           ;PIRQ
                000244                  .       =244
000244  000244  000000G 000000          *       ERRFPU,0        ;FLOATING POINT TRAP

                000400                  .       =$PRORG                 ;START OF CODE (MAY HAVE TO CHANGE,
                                                                        ;  DEPENDING UPON ATTACHED DEVS,)
```



```
                                        ;
                                        ; GENERAL ASSIGNMENTS, VARIABLES, GLOBALS, ETC,
                                        ;

                                        ; --REGISTER ASSIGNMENTS
                                        ;
                000000                  R0      =%0
                000001                  R1      =%1
                000002                  R2      =%2
                000003                  R3      =%3
                000004                  R4      =%4
                000005                  R5      =%5
                000006                  SP      =%6
                000007                  PC      =%7
                                        ;
                                        ; --DEVICE REGISTER CONSTANTS
                                        ;
                177560                  TKS     =177560
                177562                  TKB     =177562
                177564                  TPS     =177564
                177566                  TPB     =177566
                177550                  PRS     =177550
                177552                  PRB     =177552
                177554                  PPS     =177554
                177556                  PPB     =177556
                177514                  LPS     =177514
                177516                  LPB     =177516
                                        ;
                                        ; --GENERAL CONSTANTS
                                        ;
                000011                  TAB     =11
                000012                  LF=     012
                000014                  FF      =14
                000015                  CR=     015
                000040                  BL=     040
                177776                  PS      =177776
                000140                  PR3     =140            ;PRIORITY LEVEL 3
                000200                  PR4     =200
                000340                  PR7     =340
                177775                  .SCALAR =177775         ;USER SYM, TABLE FLAG FOR SCALAR
                177776                  .NVAR   =177776         ;FLAG FOR NUMERIC ARRAY VARIABLE
                177777                  .SVAR   =177777         ;FLAG FOR STRING VAR,
```

```
                                ;
                                ; **-USER AREA STORAGE CELL OFFSETS
                                ;
000000          SYMBOLS =       0               ;CONTAINS ADDR OF THE FIRST SYMBOL
000002          LIMIT   =       SYMBOLS+2       ;CONTAINS ADDR OF HIGHEST WD OF USER AREA
000004          PDL     =       LIMIT+2         ;CONTAINS ADDR OF EMPTY STACK
000006          PDSIZE  =       PDL+2           ;CONTAINS LOW LIMIT OF STACK
000010          ARRAYS  =       PDSIZE+2        ;CONTAINS ADDR OF HIGHEST WORD OF ARRAYS
000012          HIFREE  =       ARRAYS+2        ;CONTAINS ADDR OF HIGHEST FREE WORD
000014          LOFREE  =       HIFREE+2        ;CONTAINS ADDR OF LOWEST FREE WORD
000016          CODE    =       LOFREE+2        ;CONTAINS ADDR OF INTERPRETIVE CODE
000020          LINE    =       CODE+2          ;CONTAINS ADDR OF USER LINE BUFFER
000022          VARSAV  =       LINE+2          ;VAR SAVE FOR ASSIGNMENT
000024          SS1SAVE =       VARSAV+2        ;SS1 SAVE FOR ASSIGNMENT
000026          SS2SAVE =       SS1SAVE+2       ;SS2 SAVE FOR ASSIGNMENT
000030          LINENO  =       SS2SAVE+2       ;CONTAINS THE LINE NUMBER
000032          GSBCTR  =       LINENO+2        ;CONTAINS 33*DEPTH OF ACTIVE GOSUBS
000034          COLUMN  =       GSBCTR+2        ;HAS ADDR OF COL CT FOR CURRENT DEV
000036          CLMNTTY =       COLUMN+2        ;LAST TTY COLUMN TYPED
000040          FAC1    =       CLMNTTY+2       ;HIGH ORDER FLOATING VALUE
000042          FAC2    =       FAC1+2          ;LOW ORDER FLOATING VALUE
000044          R0SAVE  =       FAC2+2          ;PLACE FOR R0 WHILE IN FPMP11
000046          R1SAVE  =       R0SAVE+2        ;DITTO R1
000050          R2SAVE  =       R1SAVE+2        ;DITTO R2
000052          R3SAVE  =       R2SAVE+2        ;DITTO R3
000054          R4SAVE  =       R3SAVE+2        ;DITTO R4
000056          T1      =       R4SAVE+2        ;SHORT TERM TEMPORARY
000060          T2      =       T1+2            ;DITTO
000062          T3      =       T2+2            ;DITTO
000064          RND1    =       T3+2            ;HISTORY OF RND
000066          RND2    =       RND1+2          ;DITTO
000070          RNDCT   =       RND2+2          ;RANDOMIZER
000072          LOSTR   =       RNDCT+2         ;LOW LIMIT OF STRING STORAGE
000074          HISTR   =       LOSTR+2         ;HIGH LIMIT OF STRING STORAGE
                                ; **-THESE NEXT TWO MUST BE LO BYTE AND HI BYTE, RESPECTIVELY,
                                ;   OF SAME WORD
000076          ODEV    =       HISTR+2         ;OUTPUT DEV CODE (TTY=0,PT=2,LP=4)
000077          IDEV    =       ODEV+1          ;INPUT DEV CODE (TTY=0,PR=2)
000100          TPWD    =       ODEV+2          ;HOLDS START OF TELEPRT BUFF HDR
000102          KBWD    =       TPWD+2          ;HOLDS START OF KBD BUFF HDR
000104          ECHOSP  =       KBWD+2          ;ADDR OF NEXT ECHO CHAR (IF NON-0)
000106          CNCFLG  =       ECHOSP+2        ;'*C' FLAG, NON-0 IF PENDING
000107          CNOFLG  =       CNCFLG+1        ;'*O' FLAG, DITTO
000110          FILLCO  =       CNOFLG+1        ;FILL-COUNT
000111          FILLNO  =       FILLCO+1        ;NO OF FILL CHARS
000112          FILLCH  =       FILLNO+1        ;CHAR BEFORE FILL
000113          SPARE   =       FILLCH+1        ; (SPARE BYTE)
                                ;
                                ; **-I/O BUFFER HEADER OFFSETS
                                ;
000000          BSTRT   =0      ;(R1)   START OF BUFF,
000002          BEND    =2      ;BEND(R1)       END OF BUFFER
000004          BGET1   =4      ;BGET1(R1)      FIRST GET POINTER
000006          BGET2   =6      ;BGET2(R1)      SECOND GET PTR,
000010          BPUT    =10     ;BPUT(R1)       PUT PTR,
```



```
000012          BFSPEC  =12     ;BFSPEC(R1)     SPECIAL WORD - USE PARTIC. TO DEV.
                                ;
                                ; **-SYSTEM VARIABLES
                                ;
000400  000000  CLMNPP: ,WORD   0       ;PAPER TAPE PUNCH COLUMN POSITION
000402  000000  CLMNLP: ,WORD   0       ;LINE PRINTER COLUMN POSITION
```

```
                                ;
                                ; ** SYSTEM TOKEN DEFINITIONS
                                ;
                                ;
000201                          ,EOL=   201     ;BEGINNING OF SEQUENCE USED BY 'KEYWDS' AND 'TABLE1'
000202                          ,END=   ,EOL+1
000203                          ,FOR=   ,END+1
000204                          ,GOSUB= ,FOR+1
000205                          ,GOTO=  ,GOSUB+1
000206                          ,IF=    ,GOTO+1
000207                          ,INPUT= ,IF+1
000210                          ,LET=   ,INPUT+1
000211                          ,NEXT=  ,LET+1
000212                          ,PRINT= ,NEXT+1
000213                          ,RETURN=,PRINT+1
000214                          ,STOP=  ,RETURN+1
000215                          ,DIM=   ,STOP+1
000216                          ,RANDOM=,DIM+1
000217                          ,RESTOR=,RANDOM+1
000220                          ,REM=   ,RESTOR+1
000221                          ,DEF=   ,REM+1
000222                          ,READ=  ,DEF+1
000223                          ,DATA=  ,READ+1
000224                          ,CALL=  ,DATA+1
000225                          ,EOF=   ,CALL+1 ;END OF SEQUENCE USED BY 'TABLE1'
000226                          ,AMPERS=,EOF+1
000227                          ,UPARRO=,AMPERS+1;BEGINNING OF SEQUENCE USED BY 'TABLE2' AND OPR PREC,
000230                          ,STAR=  ,UPARRO+1
000231                          ,SLASH= ,STAR+1
000232                          ,PLUS=  ,SLASH+1
000233                          ,UNARY= ,PLUS+1
000234                          ,MINUS= ,UNARY+1
000235                          ,TERM=  ,MINUS+1;END OF SEQUENCE USED BY 'TABLE2'
000236                          ,SEMI=  ,TERM+1
000237                          ,RPAR=  ,SEMI+1
000240                          ,TO=    ,RPAR+1
000241                          ,STEP=  ,TO+1
000242                          ,THEN=  ,STEP+1
000243                          ,COMMA= ,THEN+1
000244                          ,LE=    ,COMMA+1
000245                          ,EL=    ,LE+1
000246                          ,GE=    ,EL+1
000247                          ,EG=    ,GE+1
000250                          ,NE=    ,EG+1
000251                          ,EN=    ,NE+1
000252                          ,LT=    ,EN+1
000253                          ,GT=    ,LT+1
000254                          ,EQ=    ,GT+1   ;END OF SEQUENCE USED BY OPERATOR PRECEDENCE
000255                          ,LPAR=  ,EQ+1
000256                          ,DQUOT= ,LPAR+1
000257                          ,SQUOT= ,DQUOT+1
000260                          ,COLON  =,SQUOT+1
000261                          ,POUND= ,COLON+1
000262                          ,FN=    ,POUND+1
000263                          ,RNDL=  ,FN+1   ;BEGINNING OF SEQUENCE USED BY 'TABLE5'
```



```
000264                          ,RND=   ,RNDL+1
000265                          ,SIN=   ,RND+1
000266                          ,COS=   ,SIN+1
000267                          ,SQR=   ,COS+1
000270                          ,ATN=   ,SQR+1
000271                          ,EXP=   ,ATN+1
000272                          ,LOG=   ,EXP+1
000273                          ,ABS=   ,LOG+1
000274                          ,INT=   ,ABS+1
000275                          ,SGN=   ,INT+1
000276                          ,TAB=   ,SGN+1
                                ;
                                        ,IFNDF  $NOSTR
000277                          ,LEN=   ,TAB+1
000300                          ,ASC=   ,LEN+1
000301                          ,CHR$=  ,ASC+1
000302                          ,POS=   ,CHR$+1
000303                          ,SEG=   ,POS+1
000304                          ,VAL=   ,SEG+1
000305                          ,STR=   ,VAL+1  ;END OF SEQUENCE USED BY 'TABLE5'
000306                          ,LIST=  ,STR+1  ;BEGINNING OF SEQUENCE USED BY 'TABLE4'
                                        ,ENDC
                                ;
                                        ,IFDF   $NOSTR
                                ,LIST=  ,TAB+1
                                        ,ENDC
                                ;
000307                          ,RUN=   ,LIST+1
000310                          ,SAVE=  ,RUN+1
000311                          ,OLD=   ,SAVE+1
000312                          ,SCR=   ,OLD+1
000313                          ,CLEAR= ,SCR+1  ;END OF SEQUENCE USED BY 'TABLE4' AND 'KEYWDS'
000374                          ,FLIT=  374
000375                          ,ILIT1= 375
000376                          ,ILIT2= 376
000377                          ,TEXT   =377
                                ;
```

```
                                        ;
                                        ; **KEYWORD TABLE
                                        ;
                                        ;       ENTRIES ARE ORDERED BY THEIR FREQUENCY OF
                                        ;       OCCURRENCE.  ALPHA KEYWORDS START AT 'KEYA1'.
                                        ;
000404    053                   KEYWDS1 ,ASCII  /+/
000405    232                           ,BYTE   ,PLUS
000406    055                           ,ASCII  /-/
000407    234                           ,BYTE   ,MINUS
000410    052                           ,ASCII  /*/
000411    230                           ,BYTE   ,STAR
000412    057                           ,ASCII  //
000413    231                           ,BYTE   ,SLASH
000414    136                           ,ASCII  /^/
000415    227                           ,BYTE   ,UPARRO
000416    050                           ,ASCII  /(/
000417    255                           ,BYTE   ,LPAR
000420    051                           ,ASCII  /)/
000421    237                           ,BYTE   ,RPAR
000422    134                           ,ASCII  /\/
000423    201                           ,BYTE   ,EOL
000424    046                           ,ASCII  /&/
000425    226                           ,BYTE   ,AMPERS
000426    073                           ,ASCII  /;/
000427    236                           ,BYTE   ,SEMI
000430    054                           ,ASCII  /,/
000431    243                           ,BYTE   ,COMMA
000432  036474                          ,ASCII  /<=/
000434    244                           ,BYTE   ,LE
000435    075     074                   ,ASCII  /=</
000437    244                           ,BYTE   ,LE
000440  036476                          ,ASCII  />=/
000442    246                           ,BYTE   ,GE
000443    075     076                   ,ASCII  /=>/
000445    246                           ,BYTE   ,GE
000446  037074                          ,ASCII  /<>/
000450    250                           ,BYTE   ,NE
000451    076     074                   ,ASCII  /><
000453    250                           ,BYTE   ,NE
000454    074                           ,ASCII  /</
000455    252                           ,BYTE   ,LT
000456    076                           ,ASCII  />/
000457    253                           ,BYTE   ,GT
000460    075                           ,ASCII  /=/
000461    254                           ,BYTE   ,EQ
000462    042                           ,ASCII  /"/
000463    256                           ,BYTE   ,DQUOT
000464    047                           ,ASCII  "'"
000465    257                           ,BYTE   ,SQUOT
000466    072                           ,ASCII  /:/
000467    260                           ,BYTE   ,COLON
000470    043                           ,ASCII  /#/
000471    261                           ,BYTE   ,POUND
000472    133                           ,ASCII  /[/
```



```
000473    255                           ,BYTE   ,LPAR
000474    135                           ,ASCII  /]/
000475    237                           ,BYTE   ,RPAR
000476  042514  020124          KEYA1   ,ASCII  /LET /          ;START OF ALPHA KEYWORDS
                                        ;
000502    210                           ,BYTE   ,LET
000503    111  020106                    ,ASCII  /IF /
000506    206                           ,BYTE   ,IF
000507    107  020117  047524            ,ASCII  /GO TO /
000514    040
000515    205                           ,BYTE   ,GOTO
000516  047506  020122                    ,ASCII  /FOR /
000522    203                           ,BYTE   ,FOR
000523    040  047524    040            ,ASCII  / TO /
000527    240                           ,BYTE   ,TO
000530  042516  052130    040            ,ASCII  /NEXT /
000535    211                           ,BYTE   ,NEXT
000536  052040  042510  020116            ,ASCII  / THEN /
000544    242                           ,BYTE   ,THEN
000545    040  052123  050105            ,ASCII  / STEP /
000552    040
000553    241                           ,BYTE   ,STEP
000554  047507  052523  020102            ,ASCII  /GOSUB /
000562    204                           ,BYTE   ,GOSUB
000563    122  052105  051125            ,ASCII  /RETURN/
000570    116
000571    213                           ,BYTE   ,RETURN
000572  047111  052520  020124            ,ASCII  /INPUT /
000600    207                           ,BYTE   ,INPUT
000601    120  044522  052116            ,ASCII  /PRINT /
000606    040
000607    212                           ,BYTE   ,PRINT
000610  042522    115                   ,ASCII  /REM/
000613    220                           ,BYTE   ,REM
000614  042504  020106                    ,ASCII  /DEF /
000620    221                           ,BYTE   ,DEF
000621    122  040505  020104            ,ASCII  /READ /
000626    222                           ,BYTE   ,READ
000627    104  052101  020101            ,ASCII  /DATA /
000634    223                           ,BYTE   ,DATA
000635    103  044101  020114            ,ASCII  /CALL /
000642    224                           ,BYTE   ,CALL
000643    106    116                    ,ASCII  /FN/
000645    262                           ,BYTE   ,FN
000646  047122  024104                    ,ASCII  /RND(/
000652    263                           ,BYTE   ,RNDL
000653    122  042116                    ,ASCII  /RND/
000656    264                           ,BYTE   ,RND
000657    123  047111    050            ,ASCII  /SIN(/
000663    265                           ,BYTE   ,SIN
000664  047523  024123                    ,ASCII  /COS(/
000670    266                           ,BYTE   ,COS
000671    123  051121    050            ,ASCII  /SQR(/
000675    267                           ,BYTE   ,SQR
000676  052101  024116                    ,ASCII  /ATN(/
```

```
564  000702     270                          ,BYTE   ,ATN
565  000703     105  050130      050         ,ASCII  'EXP('
566  000707     271                          ,BYTE   ,EXP
567  000710  047514  024107                  ,ASCII  'LOG('
568  000714     272                          ,BYTE   ,LOG
569  000715     101  051502      050         ,ASCII  'ABS('
570  000721     273                          ,BYTE   ,ABS
571  000722  047111  024124                  ,ASCII  'INT('
572  000726     274                          ,BYTE   ,INT
573  000727     123  047107      050         ,ASCII  'SGN('
574  000733     275                          ,BYTE   ,SGN
575  000734  040524  024102                  ,ASCII  'TAB('
576  000740     276                          ,BYTE   ,TAB
577
578                                          ,IFNDF  $NOSTR
579  000741     114  047105      050         ,ASCII  'LEN('
580  000745     277                          ,BYTE   ,LEN
581  000746  051501  024103                  ,ASCII  'ASC('
582  000752     300                          ,BYTE   ,ASC
583  000753     103  051110  024044          ,ASCII  'CHR$('
584  000760     301                          ,BYTE   ,CHR$
585  000761     120  051517      050         ,ASCII  'POS('
586  000765     302                          ,BYTE   ,POS
587  000766  042523  022107      050         ,ASCII  'SEG$('
588  000773     303                          ,BYTE   ,SEG
589  000774  040526  024114                  ,ASCII  'VAL('
590  001000     304                          ,BYTE   ,VAL
591  001001     123  051124  024044          ,ASCII  'STR$('
592  001006     305                          ,BYTE   ,STR
593                                          ,ENDC
594
595  001007     104  046511      040         ,ASCII  'DIM '
596  001013     215                          ,BYTE   ,DIM
597  001014  040522  042116  046517          ,ASCII  'RANDOMIZE'
     001022  055111     105
598  001025     216                          ,BYTE   ,RANDOM
599  001026  042522  052123  051117          ,ASCII  'RESTORE'
     001034     105
600  001035     217                          ,BYTE   ,RESTORE
601  001036  052123  050117                  ,ASCII  'STOP'
602  001042     214                          ,BYTE   ,STOP
603  001043     105  042116                  ,ASCII  'END'
604  001046     202                          ,BYTE   ,END
605  001047     114  051511      124         ,ASCII  'LIST'
606  001053     306                          ,BYTE   ,LIST
607  001054  052522     116                  ,ASCII  'RUN'
608  001057     307                          ,BYTE   ,RUN
609  001060  040523  042526                  ,ASCII  'SAVE'
610  001064     310                          ,BYTE   ,SAVE
611  001065     117  042114                  ,ASCII  'OLD'
612  001070     311                          ,BYTE   ,OLD
613  001071     123  051103                  ,ASCII  'SCR'
614  001074     312                          ,BYTE   ,SCR
615  001075     103  042514                  ,ASCII  'CLE'
616  001100     313                          ,BYTE   ,CLEAR
```



```
617
618  001101     000                          ,BYTE   0               ;END OF TABLE FLAG
619                                          ,EVEN
620                                          ,EOT
621
```

```
                                        ;                                SOURCE FILE #2
                                        ;*********************************************
                                        ;*********************************************
                                        ;*************                     *************
                                        ;*************   MAIN FLOW ROUTINES*************
                                        ;*************                     *************
                                        ;*********************************************
                                        ;*********************************************


                                        ;
                                        ; START - PROGRAM STARTING POINT
                                        ;
001102  016705  000000G         START:  MOV     USRAREA,R5
001106  016506  000004                  MOV     PDL(R5),SP
001112  005065  000104                  CLR     ECHOSP(R5)
001116  004767  007366                  JSR     PC,BUFCLR       ;CLEAR OUT ALL I/O RING BUFFERS
001122  016506  000004          SCRATCH:MOV     PDL(R5),SP
001126  004767  014422                  JSR     PC,INITSCR
001132  004767  007644          CLEAR:  JSR     PC,CLRVARS
001136  000402                          BR      READY
```



```
                                        ;-------------------------------------------------
                                        ;
                                        ; READY - PRINT 'READY'
                                        ;
001140  004767  007344          READY0: JSR     PC,BUFCLR       ;CLEAR ALL I/O RING BUFFERS
001144  016506  000004          READY:  MOV     PDL(R5),SP
001150  005065  000106                  CLR     CNCFLG(R5)      ;'^C' AND '^O' FLAGS
001154  005065  000076                  CLR     ODEV(R5)
001160  012765  000036  000034          MOV     #CLMNTTY,COLUMN(R5)     ;SET TO COUNT TTY COLUMNS
001166  060565  000034                  ADD     R5,COLUMN(R5)   ; NOW
001172  004167  016254                  JSR     R1,MSG
001176     015     012                  .BYTE   CR,LF
001200  042522  042101     131          .ASCII  'READY'
001205     000                          .BYTE 0
                                        .EVEN
001206  016506  000004          READY2: MOV     PDL(R5),SP
001212  004167  016234                  JSR     R1,MSG
001216     015     012     012          .BYTE   CR,LF,LF
001221     000                          .BYTE 0
                                        .EVEN
001222  005065  000076          IOINIT: CLR     ODEV(R5)        ;CLR ODEV AND IDEV BYTES IN USER AREA
```

```
669                                     ;+-----------------------------------------------
670                                     ;
671                                     ; EDIT -
672                                     ;
673                                     ;  GET LINE, TRANSLATE INTO TOKENS, AND MOVE LINE INTO
674                                     ;   USER CODE SPACE, WITH ALL ENTRIES MADE INTO USER
675                                     ;   SYMBOL TABLE, ARRAY SPACE, AND STRING SPACE,
676                                     ;
677  001226  004767  014702        EDIT:   JSR     PC,LINGET
678  001232  103006                        BCC     EDITL           ;NORMAL INPUT LINE
679  001234  127527  000016  000225       CMPB    @CODE(R5),#,EOF ;CHECK EMPTY PROGRAM
680  001242  001336                        BNE     READY0          ;NO, RETURN TO TTY INPUT
681  001244  000167  017604                JMP     DEVERR          ;YES, DEVICE NOT READY
682  001250  004767  020236        EDITL1: JSR     PC,TRAN
683  001254  005201                        INC     R1
684  001256  010146                        MOV     R1,-(SP)
685  001260  016501  000020                MOV     LINE(R5),R1
686  001264  160116                        SUB     R1,(SP)
687  001266  112100                        MOVB    (R1)+,R0
688  001270  100417                        BMI     EDIMMED
689  001272  000300                        SWAB    R0
690  001274  152100                        BISB    (R1)+,R0
691  001276  061500                        ADD     (R5),R0
692  001300  021027  177775                CMP     (R0),#,SCALAR
693  001304  103011                        BHIS    EDIMMED
694  001306  016501  000016                MOV     CODE(R5),R1
695  001312  021627  000003                CMP     (SP),#3         ;IF THERE ARE ONLY 3 BYTES THEN THE LINE
696  001316  001013                        BNE     EDISRCH         ;CONTAINS LINENO,CR AND MEANS DELETE,
697  001320  005016                        CLR     (SP)
698  001322  005060  000002                CLR     2(R0)
699  001326  000427                        BR      EDISRCH
700  001330  105765  000077        EDIMMED:TSTB    IDEV(R5)
701  001334  001334                        BNE     EDIT            ;IGNORE IMM STMTS FROM NON-TTY
702  001336  000167  000500                JMP     IMMED
703  001342  004767  017622        EDISKIP:JSR     PC,SKIPEOL
704  001346  010104        EDISRCH:MOV     R1,R4           ;R0 POINTS TO LINENO OF THE NEW LINE,
705  001350  121127  000225                CMPB    (R1),#,EOF      ;@(SP) CONTAINS LENGTH OF THE NEW LINE,
706  001354  001452                        BEQ     EDIPUT
707  001356  111103                        MOVB    (R1),R3
708  001360  100770                        BMI     EDISKIP
709  001362  005201                        INC     R1
710  001364  000303                        SWAB    R3
711  001366  152103                        BISB    (R1)+,R3
712  001370  061503                        ADD     (R5),R3
713  001372  021327  177775                CMP     (R3),#,SCALAR
714  001376  103361                        BHIS    EDISKIP
715  001400  021013                        CMP     (R0),(R3)
716  001402  101357                        BHI     EDISKIP
717  001404  001036                        BNE     EDIPUT
718  001406  004767  017556        EDIPASS:JSR     PC,SKIPEOL
719  001412  121127  000225                CMPB    (R1),#,EOF
720  001416  001427                        BEQ     EDIOVER
721  001420  111103                        MOVB    (R1),R3
722  001422  100771                        BMI     EDIPASS
723  001424  005201                        INC     R1
```



```
724  001426  000303                        SWAB    R3
725  001430  152103                        BISB    (R1)+,R3
726  001432  061503                        ADD     (R5),R3
727  001434  021327  177775                CMP     (R3),#,SCALAR
728  001440  103362                        BHIS    EDIPASS
729  001442  162701  000002                SUB     #2,R1
730  001446  016500  000004                MOV     POL(R5),R0      ;IF A LINE IS CHANGED WHICH IS
731  001452  016502  000032                MOV     GSBCTM(R5),R2   ;REFFRENCED BY THE READ POINTER,AN FN
732  001456  021004        EDICHG: CMP     (R0),R4         ;POINTER OR A GOSUB POINTER THEN THAT
733  001460  101403                        BLOS    .+10            ;POINTER MUST BE CLEARED
734  001462  021001                        CMP     (R0),R1
735  001464  101001                        BHI     .+4
736  001466  005010                        CLR     (R0)
737  001470  005720                        TST     (R0)+
738  001472  005302                        DEC     R2
739  001474  003370                        BGT     EDICHG
740  001476  160401        EDIOVER:SUB     R4,R1
741  001500  000401                        BR      .+4
742  001502  005001        EDIPUT: CLR     R1
743  001504  161601                        SUB     (SP),R1         ;R1 NOW = #CHARS TO CONTRACT,
744  001506  016500  000004                MOV     POL(R5),R0      ;ALL REFERENCES BY THE READ POINTER
745  001512  016502  000032                MOV     GSBCTM(R5),R2   ;WHICH REFER TO LINES WHICH ARE MOVED
746  001516  021004        EDIMODF:CMP     (R0),R4         ;BECAUSE OF EDITING MUST BE RELOCATED
747  001520  103401                        BLO     .+4
748  001522  160110                        SUB     R1,(R0)
749  001524  005720                        TST     (R0)+
750  001526  005302                        DEC     R2
751  001530  003372                        BGT     EDIMODF
752  001532  005701                        TST     R1
753  001534  100433                        BMI     EDIGROW         ;R4 NOW = ADDRESS AT WHICH TO INSERT,
754  001536  001505                        BEQ     EDISAME
755  001540  010402        EDICONT:MOV     R4,R2
756  001542  061602                        ADD     (SP),R2
757  001544  010203                        MOV     R2,R3
758  001546  060103                        ADD     R1,R3
759  001550  112322        EDIMOVE:MOVB    (R3)+,(R2)+     ;MOVE DOWN THE CODE,
760  001552  020315                        CMP     R3,(R5)
761  001554  103775                        BLO     EDIMOVE
762  001556  105742                        TSTB    -(R2)
763  001560  001401                        BEQ     .+4
764  001562  005202                        INC     R2              ;R2 NOW POINTS TO BYTE AFTER THE ,EOF,
765  001564  030227  000001                BIT     R2,#1
766  001570  001401                        BEQ     .+4
767  001572  105022                        CLRB    (R2)+           ;R2 NOW HAS THE NEW START OF THE SYMTAB,
768  001574  010215                        MOV     R2,(R5)
769  001576  012322                        MOV     (R3)+,(R2)+     ;MOVE DOWN THE SYMBOLS;
770  001600  020365  000014                CMP     R3,LOFREE(R5)
771  001604  103774                        BLO     .-6
772  001606  010265  000014                MOV     R2,LOFREE(R5)
773  001612  000401                        BR      .+4
774  001614  005022                        CLR     (R2)+           ;CLEAR VACATED STRING STORAGE TO ZEROES,
775  001616  020203                        CMP     R2,R3
776  001620  103775                        BLO     .-4
777  001622  000453                        BR      EDISAME
778  001624  011500        EDIGROW:MOV     (R5),R0
```

```
  779  001626  105740                        TSTB    @(R0)
  780  001630  001401                        BEQ     .+4
  781  001632  005200                        INC     R0
  782  001634  160100                        SUB     R1,R0           ;R0 NOW POINTS PAST THE NEW HIGHEST BYTE
  783  001636  010003                        MOV     R0,R3           ;OF CODE,
  784  001640  005203                        INC     R3
  785  001642  042703  000001                BIC     #1,R3           ;R3 NOW HAS THE NEW START OF SYMTAB,
  786  001646  161503                        SUB     (R5),R3
  787  001650  066503  000014                ADD     LOFREE(R5),R3   ;R3 NOW HAS THE NEW VALUE OF LOFREE,
  788  001654  020365  000012                CMP     R3,HIFREE(R5)   ;CHECK TO SEE THAT THERE IS ENOUGH
  789  001660  101402                        BLOS    .+6
  790  001662  000167  020712        ERROV1: JMP     ERROV2
  791
  792                                        .IFNDF  $NOSTR
  793  001666  010302                        MOV     R3,R2           ;CHECK TO SEE THAT THE SPACE THAT WILL
  794  001670  005722                        TST     (R2)+           ;BE USED IS NOT OCCUPIED BY STRINGS
  795  001672  020265  000072                CMP     R2,LOSTR(R5)
  796  001676  103405                        BLO     EDIROOM
  797  001700  004767  021254                JSR     PC,UPPACK       ;NO, MOVE THEM UP
  798  001704  020265  000072                CMP     R2,LOSTR(R5)    ;TRY AGAIN
  799  001710  103364                        BHIS    ERROV1
  800                                        .ENDC
  801
  802  001712  016502  000014        EDIROOM:MOV     LOFREE(R5),R2
  803  001716  010365  000014                MOV     R3,LOFREE(R5)
  804  001722  000401                        BR      .+4
  805  001724  014243                EDIMVUP:MOV     -(R2),-(R3)     ;MOVE UP THE SYMTAB,
  806  001726  020215                        CMP     R2,(R5)
  807  001730  101375                        BHI     EDIMVUP
  808  001732  010315                        MOV     R3,(R5)
  809  001734  105043                        CLRB    -(R3)           ;CLEAR THE POSSIBLY USED FILLER
  810  001736  010002                        MOV     R0,R2
  811  001740  060100                        ADD     R1,R0           ;R0 NOW POINTS PAST OLD HIGHEST BYTE
  812  001742  000401                        BR      .+4             ;OF SYMTAB,
  813  001744  114042                        MOVB    -(R0),-(R2)     ;MOVE UP THE CODE,
  814  001746  020004                        CMP     R0,R4
  815  001750  101375                        BHI     .-4
  816  001752  016500  000020        EDISAME:MOV     LINE(R5),R0     ;MOVE THE NEW LINE INTO THE CODE,
  817  001756  010401                        MOV     R4,R1
  818  001760  012602                        MOV     (SP)+,R2
  819  001762  000401                        BR      .+4
  820  001764  112024                        MOVB    (R0)+,(4)+
  821  001766  005302                        DEC     R2
  822  001770  002375                        BGE     .-4
  823  001772  121127  000225        EDIRLC8:CMPB    (R1),#;,EOF
  824  001776  001417                        BEQ     EDIJMP
  825  002000  111102                        MOVB    (R1),R2
  826  002002  100412                        BMI     EDIRLOC
  827  002004  005201                        INC     R1
  828  002006  000302                        SWAB    R2
  829  002010  152102                        BISB    (R1)+,R2
  830  002012  061502                        ADD     (R5),R2
  831  002014  022227  177775                CMP     (R2)+,#,SCALAR
  832  002020  103003                        BHIS    EDIRLOC
  833  002022  010112                        MOV     R1,(R2)
```



```
  834  002024  162712  000002                SUB     #2,(R2)
  835  002030  004767  017134        EDIRLOC:JSR     PC,SKIPEOL
  836  002034  000756                        BR      EDIRLC8
  837  002036  000167  177164        EDIJMP: JMP     EDIT
  838
```

```
                                ;*****************************************************
                                ;
                                ; IMMED - IMMEDIATE MODE COMMANDS
                                ;
                                ;  DO IMMEDIATE ONLY COMMANDS (RUN, LIST, SCRATCH, CLEAR,
                                ;    SAVE, AND OLD) AND SET UP OTHERS (UN-NUMBERED) FOR
                                ;    EXECUTE.
                                ;
002042  012600                  IMMED:  MOV     (SP)+,R0                ;R0 NOW HAS # BYTES ON INPUT LINE.
002044  005065  000076                  CLR     ODEV(R5)                ;TAKE ONLY THIS IMMED. COMM. FROM NON-TTY
002050  016504  000006                  MOV     PDSIZE(R5),R4
002054  016501  000020                  MOV     LINE(R5),R1
002060  005002                          CLR     R2
002062  151102                          BISB    (R1),R2
002064  162702  000306                  SUB     #,LIST,R2
002070  006302                          ASL     R2
002072  020227  000012                  CMP     R2,#TBL4END-TABLE4
002076  101012                          BHI     IMMSTMT
002100  062702  000002                  ADD     #2,R2
002104  060702                          ADD     PC,R2
002106  061207                          ADD     (R2),PC
002110  000162                  TABLE4: .WORD   LIST-TABLE4
002112  000206                          .WORD   RUN-TABLE4
002114  000124                          .WORD   SAVE-TABLE4
002116  000062                          .WORD   OLD-TABLE4
002120  177012                          .WORD   SCRATCH-TABLE4  ;(IN 'START' CODE, ABOVE)
002122  177022                  TBL4END:.WORD   CLEAR-TABLE4    ;(IN 'START' CODE, ABOVE)
002124  060100                  IMMSTMT:ADD     R1,R0
002126  020065  000016                  CMP     R0,CODE(R5)             ;SEE IF AN 'EOF' CAN BE FIT ON THE LINE,
002132  103402                          BLO     .+6
002134  000167  020034                  JMP     ERRTRN
002140  112710  000225                  MOVB    #,EOF,(R0)
002144  012765  177775  000030          MOV     #,SCALAR,LINENO(R5)
002152  005065  000036                  CLR     CLMNTY(R5)              ;RE-SET ALL COLMN COUNTS
002156  005067  176216                  CLR     CLMNPP
002162  005067  176214                  CLR     CLMNLP
002166  000167  000554                  JMP     EXECUTE                 ;START RUNNING,
```



```
                                ; 'OLD' COMMAND
002172  016506  000004          OLD:    MOV     PDL(R5),SP              ;FLUSH OUT SYSTEM
002176  004767  013352                  JSR     PC,INITSCR              ;SCRATCH
002202  004767  006574                  JSR     PC,CLRVARS
002206  004767  016722                  JSR     PC,SCANPND              ;SCAN OFF '#'
002212  004767  006464                  JSR     PC,CHKISET              ;CHECK AND SET INPUT
002216  122127  000201                  CMPB    (R1)+,#,EOL             ;MUST END LINE RIGHT
002222  001402                          BEQ     OLD001
002224  000167  010314                  JMP     ERRSX5
002230  000167  176772          OLD001: JMP     EDIT                    ;THIS DOES THE REST

                                ; 'SAVE' COMMAND
002234  004767  016674          SAVE:   JSR     PC,SCANPND              ;SCAN OFF '#'
002240  004767  006444                  JSR     PC,CHKOSET              ;CHECK AND SET-UP OUTPUT
002244  120227  000002                  CMPB    R2,#2                   ;LINE PRINTER?
002250  003404                          BLE     LISTSV                  ;NO! LIST TAKES OVER
002252  004167  015222                  JSR     R1,MSGODEV              ;FOR LPT, START W/ 2 FORM-FEEDS
002256      014     014     000         .BYTE   FF,FF,0
                002262                  .EVEN
002262  004767  014316          LISTSV: JSR     PC,LSIPROG
002266  000167  176692                  JMP     READY

                                ; 'LIST' COMMAND
002272  005201          LIST:   INC     R1
002274  004767  012534                  JSR     PC,FLINE
002300  103770                          BCS     LISTSV
002302  011201                          MOV     (R2),R1                 ;GET ADDR OF CODE LINE
002304  001766                          BEQ     LISTSV                  ;IF UNDEF, NORMAL LIST
002306  004767  014276                  JSR     PC,LSILOOP              ;LIST PARTIAL PROGR
002312  000167  176626                  JMP     READY
```

```
                                        ; /RUN/ COMMAND
002316  004767  000460          RUN:    JSR     PC,CLRVARS
002322  016501  000016                  MOV     CODE(R5),R1
002326  112102                  DIMLOOP:MOVB    (R1)+,R2
002330  100406                          BMI     DIMNOND
002332  000302                          SWAB    R2
002334  152102                          BISB    (R1)+,R2
002336  061502                          ADD     (R5),R2
002340  011265  000030                  MOV     (R2),LINENO(R5)
002344  112102                          MOVB    (R1)+,R2
002346  120227  000215          DIMNOND:CMPB    R2,#,DIM
002352  001415                          BEQ     DIMGOT
002354  120227  000221                  CMPB    R2,#,DEF
002360  001530                          BEQ     DEFGOT
002362  120227  000210                  CMPB    R2,#,RANDOM
002366  001504                          BEQ     RNDGOT
002370  120227  000225                  CMPB    R2,#,EOF
002374  001537                          BEQ     DIMDONE
002376  005301                          DEC     R1
002400  004767  016564          DEFDUN: JSR     PC,SKIPEOL
002404  000750                          BR      DIMLOOP
002406  112102                  DIMGOT: MOVB    (R1)+,R2
002410  100506                          BMI     ERRDIM
002412  000302                          SWAB    R2
002414  152102                          BISB    (R1)+,R2
002416  061502                          ADD     (R5),R2
002420  122127  000255                  CMPB    (R1)+,#,LPAR
002424  001100                          BNE     ERRDIM
002426  005003                          CLR     R3
002430  121127  000375                  CMPB    (R1),#,ILIT1
002434  001406                          BEQ     ALLOC1
002436  122127  000376                  CMPB    (R1)+,#,ILIT2
002442  001071                          BNE     ERRDIM
002444  111103                          MOVB    (R1),R3
002446  100467                          BMI     ERRDIM
002450  000303                          SWAB    R3
002452  005201                  ALLOC1: INC     R1
002454  152103                          BISB    (R1)+,R3
002456  012704  177777                  MOV     #-1,R4
002462  122127  000237                  CMPB    (R1)+,#,RPAR
002466  001423                          BEQ     DOALLOC
002470  124127  000243                  CMPB    -(R1),#,COMMA
002474  001054                          BNE     ERRDIM
002476  005201                          INC     R1
002500  005004                          CLR     R4
002502  121127  000375                  CMPB    (R1),#,ILIT1
002506  001406                          BEQ     ALLOC2
002510  122127  000376                  CMPB    (R1)+,#,ILIT2
002514  001044                          BNE     ERRDIM
002516  111104                          MOVB    (R1),R4
002520  100442                          BMI     ERRDIM
002522  000304                          SWAB    R4
002524  005201                  ALLOC2: INC     R1
002526  152104                          BISB    (R1)+,R4
002530  122127  000237                  CMPB    (R1)+,#,RPAR
```



```
002534  001034                          BNE     ERRDIM
002536  021227  177776          DOALLOC:CMP     (R2),#,NVAR
002542  001431                          BEQ     ERRDIM
002544  002403                          BLT     DIMSCLR
002546  005762  000004                  TST     4(R2)
002552  100025                          BPL     ERRDIM
002554  004767  005238          DIMSCLR:JSR     PC,ALLOC        ;VAR(R2),SS1(R3),SS2(4)
002560  122127  000243                  CMPB    (R1)+,#,COMMA
002564  001710                          BEQ     DIMGOT
002566  124127  000201                  CMPB    -(R1),#,EOL
002572  001015                          BNE     ERRDIM
002574  005201                          INC     R1
002576  000653                          BR      DIMLOOP
                                        ;RANDOMIZE STATEMENT
002600  016565  000070  000064  RNDGOT: MOV     RNDCT(R5),RND1(R5)
002606  152765  000001  000064          BISB    #1,RND1(R5)
002614  122127  000201                  CMPB    (R1)+,#,EOL
002620  001642                          BEQ     DIMLOOP
002622  000167  007716                  JMP     ERRSYN
002626  000004                  ERRDIM: IOT
                                        .IFNDF  $LONGER
002630  042111      115                 .ASCII  'IDM'
                                        .ENDC
                                        .IFDF   $LONGER
                                        .ASCII  'ILLEGAL DIM'
                                        .ENDC
002633      000                         .BYTE 0
                                        .EVEN
002634  000004                  ERRDEF: IOT
                                        .IFNDF  $LONGER
002636  042111      106                 .ASCII  \IDF\
                                        .ENDC
                                        .IFDF   $LONGER
                                        .ASCII  'ILLEGAL DEF'
                                        .ENDC
002641      000                         .BYTE 0
                                        .EVEN
002642  122127  000202          DEFGOT: CMPB    (R1)+,#,FN
002646  001372                          BNE     ERRDEF
002650  112102                          MOVB    (R1)+,R2
002652  066502  000004                  ADD     PDL(R5),R2
002656  005712                          TST     (R2)
002660  001365                          BNE     ERRDEF
002662  122127  000255                  CMPB    (R1)+,#,LPAR
002666  001362                          BNE     ERRDEF
002670  010112                          MOV     R1,(R2)
002672  000642                          BR      DEFDUN
002674  005065  000036          DIMDONE:CLR     CLMNTTY(R5)     ;RE-SET ALL COLUMN COUNTS
002700  005067  175474                  CLR     CLMNPP
002704  005067  175472                  CLR     CLMNLP
002710  016504  000006                  MOV     PDSIZE(R5),R4
002714  016501  000016                  MOV     CODE(R5),R1
002720  121127  000225                  CMPB    (R1),#,EOF
002724  001010                          BNE     EXECUTE         ;START RUNNING
002726  004167  014512          ERRRUN: JSR     R1,MSGERR
```

```
1018                                    .IFNDF  $LONGER
1019  002732  050116     122             .ASCII  \NPR\
1020                                    .ENDC
1021                                    .IFDF   $LONGER
1022                                    .ASCII  /NO PROGRAM/
1023                                    .ENDC
1024  002735     000                    .BYTE 0
1025                                    .EVEN
1026  002736  000167  176244            JMP     READYZ
1027
```



```
1028                                    ;-----------------------------------------------------------
1029                                    ;
1030                                    ; EXECUTE = EXECUTE COMMAND LINE
1031                                    ;
1032  002742  004767  016222            IGNORE: JSR     PC,SKIPEOL
1033  002746  105765  000106            EXECUTE:TSTB    CNCFLG(R5)      ;^C' HIT?
1034  002752  001404                            BEQ     NOCNC
1035  002754  105065  000106                    CLRB    CNCFLG(R5)
1036  002760  000167  176160                    JMP     READY
1037  002764  112102                    NOCNC:  MOVB    (R1)+,R2
1038  002766  002035                            BGE     NOTSYM          ;ITS A POINTER,
1039  002770  042702  177600                    BIC     #^200,R2
1040  002774  006302                            ASL     R2
1041  002776  020227  000052                    CMP     R2,#TBL1END-TABLE1+2
1042  003002  101115                            BHI     ERRSX1          ;NOT A STATEMENT WORD,
1043  003004  060702                            ADD     PC,R2
1044  003006  061207                            ADD     (R2),PC         ;BRANCH THRU TABLE1,
1045  003010  177736                    TABLE1: .WORD   EXECUTE-TABLE1
1046  003012  000330                            .WORD   END-TABLE1
1047  003014  001224                            .WORD   FOR-TABLE1
1048  003016  001040                            .WORD   GOSUB-TABLE1
1049  003020  001040                            .WORD   GOTO-TABLE1
1050  003022  000372                            .WORD   IF-TABLE1
1051  003024  002640                            .WORD   INPUT-TABLE1
1052  003026  000246                            .WORD   LET-TABLE1
1053  003030  001742                            .WORD   NEXT-TABLE1
1054  003032  002254                            .WORD   PRINT-TABLE1
1055  003034  001144                            .WORD   RETURN-TABLE1
1056  003036  000352                            .WORD   STOP-TABLE1
1057  003040  177732                            .WORD   IGNORE-TABLE1
1058  003042  177732                            .WORD   IGNORE-TABLE1
1059  003044  001026                            .WORD   RESTORE-TABLE1
1060  003046  177732                            .WORD   IGNORE-TABLE1
1061  003050  177732                            .WORD   IGNORE-TABLE1
1062  003052  003346                            .WORD   READ-TABLE1
1063  003054  177732                            .WORD   IGNORE-TABLE1
1064  003056  000074                            .WORD   CALL-TABLE1
1065  003060  000330                    TBL1END:.WORD   EOF-TABLE1
1066  003062  000302                    NOTSYM: SWAB    R2
1067  003064  152102                            BISB    (R1)+,R2
1068  003066  261502                            ADD     (R5),R2
1069  003070  021227  177775                    CMP     (R2),#,SCALAR
1070  003074  103075                            BHIS    ASSIGN          ;NOT A LINENO,
1071  003076  011265  000030                    MOV     (R2),LINENO(R5) ;SAVE THE LINENO,
1072  003102  000721                            BR      EXECUTE
1073
1074
```

```
                                ; 'CALL' STATEMENT
                                ;       CALL "PROG" (,,,)
                                CALL:

                                        .IFNDF  $NOSTM
003104  004767  000272                  JSR     PC,EVAL         ;EVALUATE FUNCTION NAME
003110  103052                          BCC     ERRSX1
003112  012600                          MOV     (SP)+,R0
003114  020027  177777                  CMP     R0,#177777      ;CHECK NULL STRING
003120  001444                          BEQ     ERRFN1          ;YES, ERROR
003122  112003                          MOVB    (R0)+,R3        ;R3 IS STRING LENGTH
003124  112020                          MOVB    (R0)+,(R0)+     ;R0 IS ADDR OF CHARS
                                        .ENDC

                                        .IFDF   $NOSTM
                                        MOVB    (R1)+,R2        ;R2 IS QUOTE CHAR
                                        CMPB    R2,#,SQUOT
                                        BEQ     CALL1
                                        CMPB    R2,#,DQUOT
                                        BNE     ERRSX1
                                CALL1:  CMPB    (R1)+,#,TEXT
                                        BNE     ERRSX1
                                        MOV     R1,R0           ;R0 IS ADDR OF CHARS
                                        CLR     R3
                                CALLP:  INC     R3              ;COUNT CHARS IN R3
                                        TSTB    (R1)+
                                        BNE     CALLP
                                        CMPB    (R1)+,R2        ;CHECK CLOSING QUOTE
                                        BNE     ERRSX1
                                        DEC     R3
                                        BEQ     ERRFN1
                                        .ENDC

                                ;       INIT TABLE SEARCH
003126  013702  000046                  MOV     @#46,R2
003132  001437                          BEQ     ERRFN1          ;NO FUNCTIONS DEFINED
003134  010246                          MOV     R2,-(SP)        ;SAVE ON STACK
                                ;       CHECK END TABLE
003136  105712                  CALLCK: TSTB    (R2)
003140  001434                          BEQ     ERRFN1
                                ;       CHECK THAT (R3) CHARS MATCH
003142  010046                          MOV     R0,-(SP)
003144  010346                          MOV     R3,-(SP)
003146  122022                  CALLM1: CMPB    (R0)+,(R2)+
003150  001034                          BNE     CALLNM
003152  005303                          DEC     R3
003154  001374                          BNE     CALLM1
                                ;       CHECK THAT 4+(R3) CHARS ARE * IN TABLE
003156  161603                          SUB     (SP),R3
003160  062703  000004                  ADD     #4,R3
003164  002422                          BLT     ERRFN1
003166  001404                          BEQ     CALLX
003170  105722                  CALLM2: TSTB    (R2)+
003172  001023                          BNE     CALLNM
003174  005303                          DEC     R3
```



```
003176  001374                          BNE     CALLM2
                                ;       RETURN ANSWER
003200  062706  000006          CALLX:  ADD     #6,SP           ;POP STACK
003204  011202                          MOV     (R2),R2
003206  001411                          BEQ     ERRFN1
003210  010146                          MOV     R1,-(SP)
003212  010446                          MOV     R4,-(SP)
003214  010546                          MOV     R5,-(SP)

003216  004712                          JSR     PC,(R2)

003220  012605                          MOV     (SP)+,R5
003222  012604                          MOV     (SP)+,R4
003224  012601                          MOV     (SP)+,R1
003226  000167  177510                  JMP     IGNORE
003232  000167  011262          ERRFN1: JMP     ERRUFN
003236  000167  007302          ERRSX1: JMP     ERRSYN
                                ;       ADVANCE TO NEXT TABLE ENTRY
003242  012603                  CALLNM: MOV     (SP)+,R3
003244  012600                          MOV     (SP)+,R0
003246  062716  000006                  ADD     #6,(SP)
003252  011602                          MOV     (SP),R2
003254  000730                          BR      CALLCK

                                ; 'LET' STATEMENT ROUTINE
003256  112102                  LET:    MOVB    (R1)+,R2
003260  100766                          BMI     ERRSX1          ;NOT A POINTER;
003262  000302                          SWAB    R2
003264  152102                          BISB    (R1)+,R2
003266  061502                          ADD     (R5),R2
003270  004767  012070          ASSIGN: JSR     PC,GETVAR
003274  122127  000254                  CMPB    (R1)+,#,EQ
003300  001356                          BNE     ERRSX1          ;NOT EQSIGN,
003302  004767  006074                  JSR     PC,EVAL

                                        .IFNDF  $NOSTM
003306  103406                          BCS     ASSSTR
                                        .ENDC

003310  122127  000201                  CMPB    (R1)+,#,EOL
003314  001350                          BNE     ERRSX1
003316  004767  015732                  JSR     PC,STOVAR
003322  000611                          BR      EXECUTE

                                        .IFNDF  $NOSTM
003324  122127  000201          ASSSTR: CMPB    (R1)+,#,EOL
003330  001342                          BNE     ERRSX1
003332  004767  015776                  JSR     PC,STOSVAR
003336  000603                          BR      EXECUTE
                                        .ENDC

                                ; 'EOF' OR 'END' STATEMENT
                                END:
003340  020165  000016          EOF:    CMP     R1,CODE(R5)
003344  103014                          BHIS    JMPRDY
```

```
 1185  003346  004167  014100                JSR     R1,MSG
 1186  003352     015     012                ,BYTE   CR,LF
 1187  003354     000                        ,BYTE 0
 1188          003356                        ,EVEN
 1189  003356  000167  175644                JMP     EDIT
 1190                                ; 'STOP' STATEMENT
 1191  003362  004167  014064        STOP:   JSR     R1,MSG
 1192  003366     015     012                ,BYTE   CR,LF
 1193  003370  052123  050117                ,ASCII  'STOP!
 1194  003374     000                        ,BYTE   0
 1195          003376                        ,EVEN
 1196  003376  000167  175542        JMPRDY: JMP     READY
 1197
```



```
 1198                                ; 'IF' STATEMENT ROUTINE
 1199  003402  004767  005774        IF:     JSR     PC,EVAL
 1200  003406  103114                        BCC     IFNUMER
 1201  003410  121127  000244                CMPB    (R1),#,LE
 1202  003414  103710                        BLO     ERRSX1
 1203  003416  121127  000254                CMPB    (R1),#,EQ
 1204  003422  101305                        BHI     ERRSX1
 1205  003424  020604                        CMP     SP,R4
 1206  003426  103553                        BLO     ERRPDJ
 1207  003430  112146                        MOVB    (R1)+,-(SP)
 1208  003432  004767  005744                JSR     PC,EVAL
 1209
 1210                                        ,IFNDF  $NOSTM
 1211  003436  103151                        BCC     ERRMX4
 1212                                        ,ENDC
 1213
 1214  003440  012603                        MOV     (SP)+,R3
 1215  003442  012600                        MOV     (SP)+,R0
 1216  003444  012602                        MOV     (SP)+,R2
 1217  003446  010046                        MOV     R0,-(SP)
 1218  003450  005046                        CLR     -(SP)
 1219  003452  020227  177777                CMP     R2,#-1
 1220  003456  001401                        BEQ     .+4
 1221  003460  151216                        BISB    (R2),(SP)
 1222  003462  005000                        CLR     R0
 1223  003464  020327  177777                CMP     R3,#-1
 1224  003470  001401                        BEQ     .+4
 1225  003472  151300                        BISB    (R3),R0
 1226  003474  062702  000003                ADD     #3,R2
 1227  003500  062703  000003                ADD     #3,R3
 1228  003504  000402                        BR      .+6
 1229  003506  122322        IFLOOP: CMPB    (R3)+,(R2)+
 1230  003510  001047                        BNE     IFCOMP
 1231  003512  005300                        DEC     R0
 1232  003514  002410                        BLT     IFLEND
 1233  003516  005316                        DEC     (SP)
 1234  003520  002372                        BGE     IFLOOP
 1235  003522  122327  000040                CMPB    (R3)+,#BL
 1236  003526  001040                        BNE     IFCOMP
 1237  003530  005300                        DEC     R0
 1238  003532  002373                        BGE     .-10
 1239  003534  000407                        BR      IFSEQ
 1240  003536  005716        IFLEND: TST     (SP)
 1241  003540  003405                        BLE     IFSEQ
 1242  003542  122722  000040                CMPB    #BL,(R2)+
 1243  003546  001030                        BNE     IFCOMP
 1244  003550  005316                        DEC     (SP)
 1245  003552  003373                        BGT     .-10
 1246  003554  005726        IFSEQ:  TST     (SP)+
 1247  003556  012600                        MOV     (SP)+,R0
 1248  003560  112102        IFLEQR: MOVB    (R1)+,R2
 1249  003562  120227  000242                CMPB    R2,#,THEN
 1250  003566  001403                        BEQ     .+10
 1251  003570  120227  000205                CMPB    R2,#,GOTO
 1252  003574  001013                        BNE     ERRSX7
```

```
 1253  003576  120027  000244                    CMPB    R0,#,LE
 1254  003602  001522                            BEQ     GOTO
 1255  003604  120027  000246                    CMPB    R0,#,GE
 1256  003610  001517                            BEQ     GOTO
 1257  003612  120027  000254                    CMPB    R0,#,EQ
 1258  003616  001514                            BEQ     GOTO
 1259  003620  000167  177110                    JMP     IGNORE
 1260  003624  000167  006714          ERRSX7: JMP     ERRSX5
 1261  003630  002456                  IFCOMP: BLT     IFSGT
 1262  003632  005726                          TST     (SP)+
 1263  003634  012600                          MOV     (SP)+,R0
 1264  003636  000425                          BR      IFLLTR
 1265  003640  121127  000244          IFNUMER:CMPB    (R1),#,LE
 1266  003644  103767                          BLO     ERRSX7
 1267  003646  121127  000254                    CMPB    (R1),#,EQ
 1268  003652  101364                          BHI     ERRSX7
 1269  003654  320604                          CMP     SP,R4
 1270  003656  103437                          BLO     ERRPD3
 1271  003660  016546  000042                  MOV     FAC2(R5),-(SP)
 1272  003664  016546  000040                  MOV     FAC1(R5),-(SP)
 1273  003670  112146                          MOVB    (R1)+,-(SP)
 1274  003672  004767  005504                  JSR     PC,EVAL
 1275
 1276                                          .IFNDF  $NOSTK
 1277  003676  103431                          BCS     ERRMX4
 1278                                          .ENDC
 1279
 1280  003700  012600                          MOV     (SP)+,R0
 1281  003702  004767  015542                  JSR     PC,SUBSTK
 1282  003706  001724                          BEQ     IFLEQR
 1283  003710  002430                          BLT     IFLGTR
 1284  003712  112102                  IFLLTR: MOVB    (R1)+,R2
 1285  003714  120227  000242                  CMPB    R2,#,THEN
 1286  003720  001403                          BEQ     .+10
 1287  003722  120227  000205                  CMPB    R2,#,GOTO
 1288  003726  001110                          BNE     ERRSX6
 1289  003730  120027  000244                  CMPB    R0,#,LE
 1290  003734  001445                          BEQ     GOTO
 1291  003736  120027  000252                  CMPB    R0,#,LT
 1292  003742  001442                          BEQ     GOTO
 1293  003744  120027  000250                  CMPB    R0,#,NE
 1294  003750  001437                          BEQ     GOTO
 1295  003752  000167  176704                  JMP     IGNORE
 1296  003756  000167  006010          ERRPD3: JMP     ERRPDL
 1297
 1298                                          .IFNDF  $NOSTK
 1299  003762  000167  007024          ERRMX4: JMP     ERRMIX
 1300                                          .ENDC
 1301
 1302  003766  005726                  IFSGT:  TST     (SP)+
 1303  003770  012600                          MOV     (SP)+,R0
 1304  003772  112102                  IFLGTR: MOVB    (R1)+,R2
 1305  003774  120227  000242                  CMPB    R2,#,THEN
 1306  004000  001403                          BEQ     .+10
 1307  004002  120227  000205                  CMPB    R2,#,GOTO
```



```
 1308  004006  001060                          BNE     ERRSX6
 1309  004010  120027  000246                  CMPB    R0,#,GE
 1310  004014  001415                          BEQ     GOTO
 1311  004016  120027  000253                  CMPB    R0,#,GT
 1312  004022  001412                          BEQ     GOTO
 1313  004024  120027  000250                  CMPB    R0,#,NE
 1314  004030  001407                          BEQ     GOTO
 1315  004032  000167  176704                  JMP     IGNORE
 1316  004036  012775  177777  000004  RESTORE:MOV     #-1,@PDL(R5)
 1317  004044  000167  176676                  JMP     EXECUTE
 1318
```

```
1319                                    ; 'GOSUB' AND 'GOTO' STATEMENTS
1320                                    GOSUB:
1321  004050  004767  010700            GOTO:   JSR     PC,FLINE
1322  004054  103435                            BCS     ERRSX0
1323  004056  122127  000201                    CMPB    (R1)+,#,EOL
1324  004062  001032                            BNE     ERRSX0
1325  004064  005712                            TST     (R2)
1326  004066  001425                            BEQ     ERRGO
1327  004070  126127  177774  000204            CMPB    -4(R1),#,GOSUB
1328  004076  001013                            BNE     GONOSAV
1329  004100  016503  000032                    MOV     GSBCTR(R5),R3
1330  004104  006303                            ASL     R3
1331  004106  066503  000004                    ADD     POL(R5),R3
1332  004112  020365  000002                    CMP     R3,LIMIT(R5)
1333  004116  101006                            BHI     ERRDEEP
1334  004120  010113                            MOV     R1,(R3)
1335  004122  005265  000032                    INC     GSBCTR(R5)
1336  004126  011201                    GONOSAV:MOV     (R2),R1
1337  004130  000167  176612                    JMP     EXECUTE
1338  004134  000004                    ERRDEEP:IOT
1339                                            .IFNDF  $LONGER
1340  004136  047105      104                   .ASCII  \GND\
1341                                            .ENDC
1342                                            .IFDF   $LONGER
1343                                            .ASCII  'GOSUBS NESTED TOO DEEPLY'
1344                                            .ENDC
1345  004141      000                           .BYTE 0
1346                                            .EVEN
1347  004142  000004                    ERRGO:  IOT
1348                                            .IFNDF  $LONGER
1349  004144  046125      116                   .ASCII  \ULN\
1350                                            .ENDC
1351                                            .IFDF   $LONGER
1352                                            .ASCII  'UNDEFINED LINE NUMBER'
1353                                            .ENDC
1354  004147      000                           .BYTE 0
1355                                            .EVEN
1356  004150  000167  006370            ERRSX6: JMP     ERRSX2
1357
```



```
1358                                    ; 'RETURN' STATEMENT
1359  004154  122127  000201            RETURN: CMPB    (R1)+,#,EOL
1360  004160  001373                            BNE     ERRSX0
1361  004162  016503  000032                    MOV     GSBCTR(R5),R3
1362  004166  020327  000033                    CMP     R3,#33
1363  004172  001415                            BEQ     ERRRET
1364  004174  005303                            DEC     R3
1365  004176  010365  000032                    MOV     R3,GSBCTR(R5)
1366  004202  006303                            ASL     R3
1367  004204  066503  000004                    ADD     POL(R5),R3
1368  004210  005713                            TST     (R3)
1369  004212  001405                            BEQ     ERRRET
1370  004214  011301                            MOV     (R3),R1
1371  004216  000167  176924                    JMP     EXECUTE
1372
1373                                            .IFNDF  $NOSTR
1374  004222  000167  006564            ERRMX5: JMP     ERRMIX
1375                                            .ENDC
1376
1377  004226  000004                    ERRRET: IOT
1378                                            .IFNDF  $LONGER
1379  004230  041122      107                   .ASCII  \RBG\
1380                                            .ENDC
1381                                            .IFDF   $LONGER
1382                                            .ASCII  'RETURN BEFORE GOSUB'
1383                                            .ENDC
1384  004233      000                           .BYTE 0
1385                                            .EVEN
1386
1387                                    ; 'FOR' STATEMENT
1388  004234  010146                    FOR:    MOV     R1,-(SP)
1389  004236  005316                            DEC     (SP)            ;(LOC OF THE 'FOR')
1390  004240  112102                            MOVB    (R1)+,R2
1391  004242  100742                            BMI     ERRSX0
1392  004244  000302                            SWAB    R2
1393  004246  152102                            BISB    (R1)+,R2
1394  004250  061502                            ADD     (R5),R2
1395  004252  021227  177777                    CMP     (R2),#,SVAR
1396  004256  001734                            BEQ     ERRSX0
1397  004260  122127  000254                    CMPB    (R1)+,#,EQ
1398  004264  201331                            BNE     ERRSX0
1399  004266  010265  000022                    MOV     R2,VARSAV(R5)   ;(VAR OF THE FOR)
1400  004272  004767  005104                    JSR     PC,EVAL
1401
1402                                            .IFNDF  $NOSTR
1403  004276  103751                            BCS     ERRMX5
1404                                            .ENDC
1405
1406  004300  012765  177777  000024            MOV     #-1,SS1SAV(R5)  ;FORCE TO SCALAR,
1407  004306  004767  014742                    JSR     PC,STOVAR
1408  004312  122127  000240                    CMPB    (R1)+,#,TO
1409  004316  001314                            BNE     ERRSX0
1410  004320  004767  005056                    JSR     PC,EVAL
1411
1412                                            .IFNDF  $NOSTR
```

```
1413  004324  103736                          BCS     ERRMX9
1414                                          .ENDC
1415
1416  004326  016565  000040  000024          MOV     FAC1(R5),SS1SAV(R5)
1417  004334  016565  000042  000026          MOV     FAC2(R5),SS2SAV(R5)!(LIMIT OF THE FOR)
1418  004342  005065  000040                  CLR     FAC1(R5)
1419  004346  012765  000001  000042          MOV     #1,FAC2(R5)     ;ASSUMED STEP IS 1.0
1420  004354  121127  000201                  CMPB    (R1),#,EOL
1421  004360  001411                          BEQ     FORONE
1422  004362  122127  000241                  CMPB    (R1)+,#,STEP
1423  004366  001270                          BNE     ERRSX9
1424  004370  004767  005006                  JSR     PC,EVAL         ;(FAC HAS THE STEP)
1425
1426                                          .IFNDF  $NOSTK
1427  004374  103712                          BCS     ERRMX9
1428                                          .ENDC
1429
1430  004376  121127  000201                  CMPB    (R1),#,EOL
1431  004402  001262                          BNE     ERRSX9
1432  004404  005201                  FORONE: INC     R1
1433  004406  016503  000038                  MOV     LINENO(R5),R3
1434  004412  000402                          BR      FORLOOK
1435  004414  004767  014558          FORSKIP:JSR     PC,SKIPEOL
1436  004420  111102                  FORLOOK:MOVB    (R1),R2
1437  004422  100410                          BMI     FORNOL
1438  004424  005201                          INC     R1
1439  004426  000302                          SWAB    R2
1440  004430  152102                          BISB    (R1)+,R2
1441  004432  061502                          ADD     (R5),R2
1442  004434  021227  177775                  CMP     (R2),#,SCALAR
1443  004440  103001                          BHIS    FORNOL
1444  004442  011203                          MOV     (R2),R3         ;(THE LINENO WHILE LOOKING)
1445  004444  121127  000225          FORNOL: CMPB    (R1),#,EOF
1446  004450  001420                          BEQ     ERR4WO
1447  004452  121127  000211                  CMPB    (R1),#,NEXT     ;THE CODE IS NOW BEING SEARCHED
1448  004456  001420                          BEQ     FORNEXT         ;FOREWARD FROM THE 'FOR' FOR A NEXT WITH
1449  004460  121127  000203                  CMPB    (R1),#,FOR      ;THE SAME VAR, IF 'EOF' OR ANOTHER 'FOR'
1450  004464  001353                          BNE     FORSKIP         ;WITH THE SAME VAR IS FOUND FIRST|ERROR,
1451  004466  005201                          INC     R1
1452  004470  111102                          MOVB    (R1),R2
1453  004472  100750                          BMI     FORSKIP
1454  004474  005201                          INC     R1
1455  004476  000302                          SWAB    R2
1456  004500  152102                          BISB    (R1)+,R2
1457  004502  061502                          ADD     (R5),R2
1458  004504  020265  000022                  CMP     R2,VARSAV(R5)
1459  004510  001341                          BNE     FORSKIP
1460  004512  000004                  ERR4WO: IOT
1461                                          .IFNDF  $LONGER
1462  004514  053506     116                  .ASCII  'FWN'
1463                                          .ENDC
1464                                          .IFDF   $LONGER
1465                                          .ASCII  'FOR WITHOUT NEXT'
1466                                          .ENDC
1467  004517     000                          .BYTE 2
```



```
1468                                          .EVEN
1469  004520  062701  000013          FORNEXT:ADD     #13,R1
1470  004524  111102                          MOVB    (R1),R2
1471  004526  100732                          BMI     FORSKIP
1472  004530  005201                          INC     R1
1473  004532  000302                          SWAB    R2
1474  004534  152102                          BISB    (R1)+,R2
1475  004536  061502                          ADD     (R5),R2
1476  004540  020265  000022                  CMP     R2,VARSAV(R5)
1477  004544  001323                          BNE     FORSKIP
1478  004546  121127  000201                  CMPB    (R1),#,EOL
1479  004552  001050                          BNE     ERSX10
1480  004554  162701  000014                  SUB     #14,R1
1481  004560  116621  000001                  MOVB    1(SP),(R1)+     ;PUT ADDR OF THE 'FOR' AFTER THE 'NEXT',
1482  004564  111621                          MOVB    (SP),(R1)+      ;(THATS WHAT THE 2+4+4 BYTES ARE FOR)
1483  004566  116521  000025                  MOVB    SS1SAV+1(R5),(R1)+;PUT THE LIMIT AFTER THAT,
1484  004572  116521  000024                  MOVB    SS1SAV(R5),(R1)+
1485  004576  116521  000027                  MOVB    SS2SAV+1(R5),(R1)+
1486  004602  116521  000026                  MOVB    SS2SAV(R5),(R1)+
1487  004606  116521  000041                  MOVB    FAC1+1(R5),(R1)+        ;PUT THE STEP AFTER THAT,
1488  004612  116521  000040                  MOVB    FAC1(R5),(R1)+
1489  004616  116521  000043                  MOVB    FAC2+1(R5),(R1)+
1490  004622  116521  000042                  MOVB    FAC2(R5),(R1)+
1491  004626  010316                          MOV     R3,(SP)         ;PUT THE SEARCH LINENO ON THE STACK,
1492  004630  016546  000040                  MOV     FAC1(R5),-(SP)  ;PUT SGN(STEP) ON THE STACK,
1493  004634  001002                          BNE     .+6
1494  004636  016516  000042                  MOV     FAC2(R5),(SP)
1495  004642  016546  000026                  MOV     SS2SAV(R5),-(SP)        ;PUT THE LIMIT ON THE STACK,
1496  004646  016546  000024                  MOV     SS1SAV(R5),-(SP)
1497  004652  004767  004524                  JSR     PC,EVAL         ;GET THE CURRENT VALUE OF THE INDEX,
1498  004656  004767  014566                  JSR     PC,SUBSTK       ;SUBTRACT THE LIMIT,
1499  004662  001421                          BEQ     FORGOGO
1500  004664  100405                          BMI     FORLTLM
1501  004666  005726                  FORGTLM:TST     (SP)+
1502  004670  100417                          BMI     FORGO
1503  004672  000404                          BR      FORZERO
1504  004674  000167  005644          ERSX10: JMP     ERRSX9
1505  004700  005726                  FORLTLM:TST     (SP)+
1506  004702  100012                          BPL     FORGO
1507  004704  012665  000038          FORZERO:MOV     (SP)+,LINENO(R5)
1508  004710  105061  177764                  CLRB    -14(R1)
1509  004714  105061  177765                  CLRB    -13(R1)
1510  004720  005201                          INC     R1
1511  004722  000167  176020                  JMP     EXECUTE
1512  004726  005726                  FORGOGO:TST     (SP)+
1513  004730  005726                  FORGO:  TST     (SP)+
1514  004732  116146  177765                  MOVB    -13(R1),-(SP)
1515  004736  116166  177764  000001          MOVB    -14(R1),1(SP)
1516  004744  012601                          MOV     (SP)+,R1
1517  004746  000167  175770                  JMP     IGNORE
1518                                          .EOT
1519
```

```
1520                                    ;                               SOURCE FILE #3
1521                                    ; 'NEXT' STATEMENT
1522  004752  005002                   NEXT:   CLR     R2
1523  004754  152102                           BISB    (R1)+,R2
1524  004756  000302                           SWAB    R2
1525  004760  152102                           BISB    (R1)+,R2
1526  004762  020265  000014                   CMP     R2,LOFREE(R5)   ;JUST TO PREVENT AN NXM TRAP,
1527  004766  103131                           BHIS    ERRNEXT
1528  004770  122227  000203                   CMPB    (R2)+,#,FOR
1529  004774  001126                           BNE     ERRNEXT
1530  004776  062701  000010                   ADD     #10,R1
1531  005002  112103                           MOVB    (R1)+,R3
1532  005004  100525                           BMI     ERRSX2
1533  005006  000303                           SWAB    R3
1534  005010  151103                           BISB    (R1),R3
1535  005012  061503                           ADD     (R5),R3
1536  005014  005301                           DEC     R1
1537  005016  122122                           CMPB    (R1)+,(R2)+
1538  005020  001114                           BNE     ERRNEXT
1539  005022  121122                           CMPB    (R1),(R2)+
1540  005024  001112                           BNE     ERRNEXT
1541  005026  121227  000254                   CMPB    (R2),#,EQ
1542  005032  001107                           BNE     ERRNEXT
1543  005034  010365  000022                   MOV     R3,VARSAV(R5)   ;SAVE VARIABLE ADDRESS,
1544  005040  021327  177777                   CMP     (R3),#,SVAR
1545  005044  001505                           BEQ     ERRSX2
1546  005046  022327  177775                   CMP     (R3)+,#,SCALAR
1547  005052  001401                           BEQ     .+4
1548  005054  011303                           MOV     (R3),R3
1549  005056  012365  000040                   MOV     (R3)+,FAC1(R5)  ;PUT THE FOR VAR VALUE INTO FAC,
1550  005062  011365  000042                   MOV     (R3),FAC2(R5)
1551  005066  005301                           DEC     R1
1552  005070  114146                           MOVB    -(R1),-(SP)     ;PUT STEP ON THE STACK,
1553  005072  114166  000001                   MOVB    -(R1),1(SP)
1554  005076  114146                           MOVB    -(R1),-(SP)
1555  005100  114166  000001                   MOVB    -(R1),1(SP)
1556  005104  011665  000026                   MOV     (SP),SS2SAV(R5) ;SAVE SGN(STEP),
1557  005110  001003                           BNE     .+10
1558  005112  016665  000002  000026           MOV     2(SP),SS2SAV(R5)
1559  005120  004767  002476                   JSR     PC,ADDSTK
1560  005124  016546  000042                   MOV     FAC2(R5),-(SP)
1561  005130  016546  000040                   MOV     FAC1(R5),-(SP)
1562  005134  114146                           MOVB    -(R1),-(SP)
1563  005136  114166  000001                   MOVB    -(R1),1(SP)
1564  005142  114146                           MOVB    -(R1),-(SP)
1565  005144  114166  000001                   MOVB    -(R1),1(SP)
1566  005150  004767  014274                   JSR     PC,SUBSTK
1567  005154  001417                           BEQ     NEXGO
1568  005156  100404                           BMI     NEXLTLM
1569  005160  005765  000026           NEXGTLM:TST     SS2SAV(R5)
1570  005164  100413                           BMI     NEXGO
1571  005166  000403                           BR      NEXEND
1572  005170  005765  000026           NEXLTLM:TST     SS2SAV(R5)
1573  005174  100007                           BPL     NEXGO
1574  005176  105041                   NEXEND: CLRB    -(R1)
```



```
1575  005200  105041                           CLRB    -(R1)
1576  005202  062701  000015                   ADD     #15,R1
1577  005206  022626                           CMP     (SP)+,(SP)+     ;THROW AWAY INCREMENTED INDEX,
1578  005210  000167  175532                   JMP     EXECUTE
1579  005214  114146                   NEXGO:  MOVB    -(R1),-(SP)
1580  005216  114166  000001                   MOVB    -(R1),1(SP)
1581  005222  012601                           MOV     (SP)+,R1
1582  005224  012665  000040                   MOV     (SP)+,FAC1(R5)  ;STORE THE INCREMENTED INDEX,
1583  005230  012665  000042                   MOV     (SP)+,FAC2(R5)
1584  005234  012765  177777  000024           MOV     #-1,SS1SAV(R5)
1585  005242  004767  014006                   JSR     PC,STOVAR
1586  005246  000167  175470                   JMP     IGNORE
1587  005252  000004                   ERRNEXT:IOT
1588                                            ;IFNDF  $LONGER
1589  005254  041116     106                  ;ASCII  \NBF\
1590                                            ;ENDC
1591                                            ;IFDF   $LONGER
1592                                            ;ASCII  /NEXT BEFORE FOR/
1593                                            ;ENDC
1594  005257     000                           ;BYTE 0
1595                                            ;EVEN
1596  005260  000167  005260           ERRSX2: JMP     ERRSX2
1597
```

```
                        ; 'PRINT' STATEMENT
005264 121127 000261    PRINT:  CMPB    (R1),#';POUND   ;IS IT 'PRINT #N ...'?
005270 001016                   BNE     PRNT01
005272 005201                   INC     R1
005274 004767 003410            JSR     PC,CHKOSET      ;CHECK CHANNEL AND SET-UP I/O
005300 005702                   TST     R2              ;DEVICE CODE
005302 001403                   BEQ     COLONCH
005304 016265 005406 000034             MOV     TABL5(R2),COLUMN(R5)    ;NON-TTY COLUMN COUNT
005312 121127 000201    COLONCH:CMPB    (R1),#':EOL     ;DON'T NEED COLON IF NOTHING
005316 001403                   BEQ     PRNT01          ;  FOLLOWS NUMBER
005320 122127 000260            CMPB    (R1)+,#'0,COLON
005324 001355                   BNE     ERRSX2
005326 121127 000243    PRNT01: CMPB    (R1),#',COMMA
005332 001453                   BEQ     PRICM
005334 121127 000236            CMPB    (R1),#';SEMI
005340 001453                   BEQ     PRIBOTH
005342 121127 000201            CMPB    (R1),#':EOL
005346 001450                   BEQ     PRIBOTH
                                ,IFDF   $NOSTR
                                CMPB    (R1),#'",SQUOT  ;CHECK PRINT STRING
                                BEQ     PRISTR
                                CMPB    (R1),#'',DQUOT
                                BEQ     PRISTR
                                CMPB    (R1),#TAB       ;OR TAB FUNCTION
                                BEQ     PRITB
                                ,ENDC
005350 004767 004026            JSR     PC,EVAL

                                ,IFNDF  $NOSTR
005354 103417                   BCS     PRISTR
                                ,ENDC

005356 027527 000034 000074             CMP     @COLUMN(R5),#74
005364 101402                   BLOS    .+6
005366 004767 000240            JSR     PC,PRINCR
005372 004767 012340            JSR     PC,NUMSGN
005376 020706                   .WORD   PUTCHAR
005400 004167 012074            JSR     R1,MSGODEV
005404     040                  .ASCII  ' '
005405     000                  .BYTE 0
                                .EVEN
005406 000430                   BR      PRIBOTH

                        TABL5   *,*+2
                                .       CLMNPP
                                .       CLMNLP


                                ,IFNDF  $NOSTR
005414 012602           PRISTR: MOV     (SP)+,R2
005416 005202                   INC     R2
005420 001423                   BEQ     PRIBOTH
```

202


```
005422 005003                           CLR     R3
005424 154203                           BISB    @(R2),R3
005426 062702 000003                    ADD     #3,R2
005432 027527 000034 000110 PRILOOP:CMP     @COLUMN(R5),#110
005440 103402                           BLO     .+6
005442 004767 000164                    JSR     PC,PRINCR
005446 112200                           MOVB    (R2)+,R0
005450 004767 013232                    JSR     PC,PUTCHAR
005454 005303                           DEC     R3
005456 003365                           BGT     PRILOOP
005460 000403                           BR      PRIBOTH
                                        ,ENDC

005462 004167 012012            PRICM:  JSR     R1,MSGODEV
005466     040      000                 ,BYTE   BL,0
005470 121127 000201            PRIBOTH:CMPB    (R1),#':EOL
005474 001003                           BNE     PRIMORE
005476 004767 000130                    JSR     PC,PRINCR
005502 000441                           BR      PRIJMP
005504 122127 000243            PRIMORE:CMPB    (R1)+,#',COMMA
005510 001027                           BNE     PRISEMI
005512 017500 000034            PRICOMM:MOV     @COLUMN(R5),R0
005516 001430                           BEQ     PRITEST
005520 020027 000070                    CMP     R0,#70
005524 001425                           BEQ     PRITEST
005526 103403                           BLO     .+10
005530 004767 000076                    JSR     PC,PRINCR
005534 000421                           BR      PRITEST
005536 020027 000052                    CMP     R0,#52
005542 001416                           BEQ     PRITEST
005544 020027 000034                    CMP     R0,#34
005550 001413                           BEQ     PRITEST
005552 020027 000016                    CMP     R0,#16
005556 001410                           BEQ     PRITEST
005560 004167 011714                    JSR     R1,MSGODEV
005564     040                  .ASCII  ' '
005565     000                  .BYTE 0
                                .EVEN
005566 000751                           BR      PRICOMM
005570 124127 000236            PRISEMI:CMPB    @(R1)+,#';SEMI
005574 001254                           BNE     PRNT01
005576 005201                           INC     R1
005600 121127 000201            PRITEST:CMPB    (R1),#':EOL
005604 001250                           BNE     PRNT01
005606 005201                   PRIJMP: INC     R1
005610 012765 000036 000034             MOV     #CLMNITY,COLUMN(R5)     ;RE-SET TO TTY COLUMN COUNT
005616 060565 000034                    ADD     R5,COLUMN(R5)
005622 105065 000076                    CLRB    ODEV(R5)
005626 000167 175114                    JMP     EXECUTE

                                        ,IFDF   $NOSTR
                                PRISTR: MOVB    (R1)+,R3        ;SAVE OPEN QUOTE CHAR
                                        CMPB    (R1)+,#',TEXT   ;TEST FOR TEXT BYTE
                                PRISJ:  BNE     ERRSX2
                                PRIST1: MOVB    (R1)+,R2        ;GET NEXT CHAR
```

203

```
1708                                            BEQ     PRIST1
1709                                            CMPB    R2,R3           ;CHECK CLOSE QUOTE
1710                                            BEQ     PRIBOTH
1711                                            CMPB    R2,#,EOL        ;CHECK END LINE
1712                                            BEQ     ERRSX2
1713                                            CMP     @COLUMN(R5),#110
1714                                            BLO     .+6
1715                                            JSR     PC,PRINCR
1716                                            MOV     R2,R0
1717                                            JSR     PC,PUTCHAR
1718                                            BR      PRIST1
1719
1720                                    PRITB1: TSTB    (R1)+
1721                                            JSR     PC,EVAL
1722                                            JSR     PC,INT
1723                                            CMPB    (R1)+,#,RPAR
1724                                            BNE     PRISJ
1725                                            MOVB    FAC2(R5),R2     ;GET FN VALUE
1726                                            SUB     @COLUMN(R5),R2  ;COMPUTE # SPACES
1727                                    PRITB0: CMP     R2,#72.
1728                                            BLO     PRITB1
1729                                            SUB     #72.,R2
1730                                            BR      PRITB0
1731                                    PRITB1: DEC     R2
1732                                            BMI     PRIBOTH
1733                                            MOV     #8L,R0
1734                                            JSR     PC,PUTCHAR
1735                                            BR      PRITB1
1736                                            .ENDC
1737                                    ;
1738  005632  004167  011642           PRINCR: JSR     R1,MSGODEV
1739  005636     015     012                    .BYTE   CR,LF
1740  005640     000                            .BYTE 0
1741          005642                            .EVEN
1742  005642  005075  000034                    CLR     @COLUMN(R5)
1743  005646  000207                            RTS     PC
1744
1745
```



```
1746                                    ; 'INPUT' STATEMENT
1747  005650  020165  000016           INPUT:  CMP     R1,CODE(R5)
1748  005654  101006                           BHI     INPYES
1749  005656  004167  011502                   JSR     R1,MSGERR
1750                                           .IFNDF  $LONGER
1751  005662  046111     116                   .ASCII  \ILN\
1752                                           .ENDC
1753                                           .IFDF   $LONGER
1754                                           .ASCII  'ILLEGAL NOW'
1755                                           .ENDC
1756  005665     000                           .BYTE 0
1757                                           .EVEN
1758  005666  000167  173314                   JMP     READY2
1759  005672  105065  000002           INPYES: CLRB    T3(R5)
1760  005676  105065  000077                   CLRB    IDEV(R5)        ;RESET INPUT TO TTY
1761  005702  121127  000261                   CMPB    (R1),#,POUND    ;IS IT 'INPUT #N'...?
1762  005706  001010                           BNE     INPY01
1763  005710  105265  000002                   INCB    T3(R5)
1764  005714  005201                           INC     R1
1765  005716  104767  002760                   JSR     PC,CHKISET      ;CHECK CHANNEL AND SET-UP INPUT
1766  005722  122127  000260                   CMPB    (R1)+,#,COLON
1767  005726  201150                           BNE     ERRSX0
1768  005730  010746           INPY01: MOV     PC,-(SP)
1769  005732  062716  000416           INPPC:  ADD     #INPEOL-INPPC,(SP)
1770  005736  112102           INPLOOP:MOVB    (R1)+,R2
1771  005740  100543                           BMI     ERRSX0
1772  005742  000302                           SWAB    R2
1773  005744  152102                           BISB    (R1)+,R2
1774  005746  061502                           ADD     (R5),R2
1775  005750  004767  007410                   JSR     PC,GETVAR
1776  005754  011603           INPRTRY:MOV     (SP),R3
1777  005756  121327  000201                   CMPB    (R3),#,EOL
1778  005762  001053                           BNE     INPOK
1779  005764  105765  000002           INPNEW: TSTB    T3(R5)
1780  005770  001003                           BNE     NOQM
1781  005772  004167  011454                   JSR     R1,MSG
1782  005776     077                           .ASCII  '?'
1783  005777     000                           .BYTE 0
1784                                           .EVEN
1785  006000  010146           NOQM:   MOV     R1,-(SP)
1786  006002  004767  010126                   JSR     PC,LINGET
1787  006006  103002                           BCC     NOQM1
1788  006010  000167  000512                   JMP     ERRDATA
1789  006014  012601           NOQM1:  MOV     (SP)+,R1
1790  006016  016502  000022                   MOV     VARSAV(R5),R2
1791
1792                                           .IFNDF  $NOSTR
1793  006022  021227  177777                   CMP     (R2),#,SVAR
1794  006026  001512                           BEQ     INPSTR
1795                                           .ENDC
1796
1797  006030  010146                           MOV     R1,-(SP)
1798  006032  010446                           MOV     R4,-(SP)
1799  006034  016501  000020                   MOV     LINE(R5),R1
1800  006040  122127  000015                   CMPB    (R1)+,#CR
```

```
 1801  006044  001375                    BNE     .+4
 1802  006046  010104                    MOV     R1,R4
 1803  006050  005204                    INC     R4
 1804  006052  020465  000016            CMP     R4,CODE(R5)
 1805  006056  101135                    BHI     ERRTR8
 1806  006060  114144                    MOVB    -(R1),-(4)
 1807  006062  020165  000020            CMP     R1,LINE(R5)
 1808  006066  101374                    BHI     .-6
 1809  006070  112711  000054            MOVB    #',,(R1)
 1810  006074  004767  013412            JSR     PC,TRAN
 1811  006100  012604                    MOV     (SP)+,R4
 1812  006102  012601                    MOV     (SP)+,R1
 1813  006104  016516  000020            MOV     LINE(R5),(SP)
 1814  006110  000721                    BR      INPRTRY
 1815  006112  016502  000022  INPOK1:   MOV     VARSAV(R5),R2
 1816
 1817                                    .IFNDF  $NOSTR
 1818  006116  021227  177777            CMP     (R2),#-1,SVAR
 1819  006122  001720                    BEQ     INPNEW
 1820                                    .ENDC
 1821
 1822  006124  005203                    INC     R3
 1823  006126  005065  000040            CLR     FAC1(R5)
 1824  006132  005065  000042            CLR     FAC2(R5)
 1825  006136  121327  000243            CMPB    (R3)+,#',COMMA
 1826  006142  001403                    BEQ     INPSTO
 1827  006144  004767  010104            JSR     PC,LITEVAL
 1828  006150  000431                    BR      INPNGUD
 1829  006152  010316  INPSTO: MOV       R3,(SP)
 1830  006154  004767  013074            JSR     PC,STOVAR
 1831  006160  011603                    MOV     (SP),R3
 1832  006162  121327  000243            CMPB    (R3)+,#',COMMA
 1833  006166  001403                    BEQ     INPGOOD
 1834  006170  121327  000201            CMPB    (R3)+,#EOL
 1835  006174  001017                    BNE     INPNGUD
 1836  006176  122127  000243  INPGOOD: CMPB    (R1)+,#',COMMA
 1837  006202  001655                    BEQ     INPLOOP
 1838  006204  126127  177777  000201    INPEND: CMPB    -1(R1),#,EOL
 1839  006212  001016                    BNE     ERRSX8
 1840  006214  005726                    TST     (SP)+
 1841  006216  105765  000077            TSTB    IDEV(R5)        ;IF TTY INPUT, CLR COLUMN COUNT
 1842  006222  001002                    BNE     GOEXEC
 1843  006224  005065  000036            CLR     CLMNTTY(R5)     ;  FOR SAME
 1844  006230  000167  174512  GOEXEC: JMP     EXECUTE
 1845  006234  004167  011204  INPNGUD: JSR     R1,MSGERR
 1846                                    .IFNDF  $LONGER
 1847  006240  051102     124            .ASCII  \BRT\
 1848                                    .ENDC
 1849                                    .IFDF   $LONGER
 1850                                    .ASCII  'BAD DATA-RETYPE FROM ERROR,'
 1851                                    .ENDC
 1852  006243     015     012            .BYTE   CR,LF
 1853  006245     000                    .BYTE   0
 1854                                    .EVEN
 1855  006246  000646                    BR      INPNEW
```

206


```
 1856  006250  000167  004270  ERRSX8: JMP     ERRSX5
 1857
 1858                                    .IFNDF  $NOSTR
 1859  006254  016503  000020  INPSTR: MOV     LINE(R5),R3
 1860  006260  010302                    MOV     R3,R2
 1861  006262  122327  000015            CMPB    (R3)+,#CR
 1862  006266  001375                    BNE     .-4
 1863  006270  010246                    MOV     R2,-(SP)
 1864  006272  162716  000003            SUB     #3,(SP)
 1865  006276  010346                    MOV     R3,-(SP)
 1866  006300  160216                    SUB     R2,(SP)
 1867  006302  005316                    DEC     (SP)
 1868  006304  001416                    BEQ     INPNULL
 1869  006306  010602                    MOV     SP,R2
 1870  006310  062702  000002            ADD     #2,R2
 1871  006314  004767  010732            JSR     PC,MAKESTR
 1872  006320  004767  013010  INPNNUL: JSR     PC,STOSVAR
 1873  006324  005726                    TST     (SP)+
 1874  006326  122127  000243            CMPB    (R1)+,#',COMMA
 1875  006332  001324                    BNE     INPEND
 1876  006334  005726                    TST     (SP)+
 1877  006336  000167  177366            JMP     INPY01
 1878  006342  012716  177777  INPNULL: MOV     #-1,(SP)
 1879  006346  000764                    BR      INPNNUL
 1880                                    .ENDC
 1881
 1882  006350  000201  INPEOL: .WORD     EOL
 1883  006352  000167  013616  ERRTR8: JMP     ERRTRN
 1884
```

207

```
 1885                                   ; /READ/ STATEMENT
 1886  006356 112102                    READ:   MOVB    (R1)+,R2
 1887  006360 100733                            BMI     ERRSX8
 1888  006362 000302                            SWAB    R2
 1889  006364 152102                            BISB    (R1)+,R2
 1890  006366 061502                            ADD     (R5),R2
 1891  006370 004767 006770                     JSR     PC,GETVAR
 1892  006374 017503 000004                     MOV     @PDL(R5),R3
 1893  006400 001452                            BEQ     ERRDATA
 1894  006402 020327 177777                     CMP     R3,#-1
 1895  006406 001457                            BEQ     READSRCH
 1896  006410 121327 000201                     CMPB    (R3),#,EOL
 1897  006414 001456                            BEQ     READFIND
 1898  006416 122327 000243                     CMPB    (R3)+,#,COMMA
 1899  006422 001044                            BNE     READBAD
 1900  006424 111302                    READGOT:MOVB    (R3),R2
 1901
 1902                                           .IFNDF  $NOSTR
 1903  006426 120227 000256                     CMPB    R2,#DQUOTE
 1904  006432 001470                            BEQ     READQT
 1905  006434 120227 000257                     CMPB    R2,#SQUOTE
 1906  006440 001465                            BEQ     READQT
 1907                                           .ENDC
 1908
 1909  006442 004767 007006                     JSR     PC,LITEVAL
 1910  006446 000432                            BR      READBAD
 1911  006450 010346                            MOV     R3,-(SP)
 1912  006452 004767 012576                     JSR     PC,STOVAR
 1913  006456 012603                    READDUN:MOV     (SP)+,R3
 1914  006460 121327 000201                     CMPB    (R3),#,EOL
 1915  006464 001403                            BEQ     .+10
 1916  006466 121327 000243                     CMPB    (R3),#,COMMA
 1917  006472 001020                            BNE     READBAD
 1918  006474 010375 000004                     MOV     R3,@PDL(R5)
 1919  006500 122127 000243                     CMPB    (R1)+,#,COMMA
 1920  006504 001724                            BEQ     READ
 1921  006506 126127 177777 000201              CMPB    -1(R1),#,EOL
 1922  006514 001255                            BNE     ERRSX8
 1923  006516 000167 174224                     JMP     EXECUTE
 1924  006522 005075 000004             READOUT:CLR     @PDL(R5)
 1925  006526 000004                    ERRDATA:IOT
 1926                                           .IFNDF  $LONGER
 1927  006530 047517     104                    .ASCII  \OOD\
 1928                                           .ENDC
 1929                                           .IFDF   $LONGER
 1930                                           .ASCII  'OUT OF DATA'
 1931                                           .ENDC
 1932  006533     000                           .BYTE 0
 1933                                           .EVEN
 1934  006534 005075 000004             READBAD:CLR     @PDL(R5)
 1935  006540 000004                            IOT
 1936                                           .IFNDF  $LONGER
 1937  006542 042102     122                    .ASCII  \BOR\
 1938                                           .ENDC
 1939                                           .IFDF   $LONGER
```

208


```
 1940                                           .ASCII  'BAD DATA READ'
 1941                                           .ENDC
 1942  006545     000                           .BYTE 0
 1943                                           .EVEN
 1944  006546 016503 000016             READSRCH:MOV    CODE(R5),R3
 1945  006552 105713                    READFIND:TSTB   (R3)
 1946  006554 100402                            BMI     .+6
 1947  006556 062703 000002                     ADD     #2,R3
 1948  006562 122327 000223                     CMPB    (R3)+,#,DATA
 1949  006566 001716                            BEQ     READGOT
 1950  006570 124327 000225                     CMPB    -(R3),#,EOP
 1951  006574 001752                            BEQ     READOUT
 1952  006576 010146                            MOV     R1,-(SP)
 1953  006600 010301                            MOV     R3,R1
 1954  006602 004767 012302                     JSR     PC,SKIPEOL
 1955  006606 010103                            MOV     R1,R3
 1956  006610 012601                            MOV     (SP)+,R1
 1957  006612 000757                            BR      READFIND
 1958
 1959                                           .IFNDF  $NOSTR
 1960  006614 005203                    READQT: INC     R3
 1961  006616 122327 000377                     CMPB    (R3)+,#,TEXT
 1962  006622 001344                            BNE     READBAD
 1963  006624 010300                            MOV     R3,R0
 1964  006626 105723                            TSTB    (R3)+
 1965  006630 001376                            BNE     .-2
 1966  006632 122302                            CMPB    (R3)+,R2
 1967  006634 001337                            BNE     READBAD
 1968  006636 010346                            MOV     R3,-(SP)
 1969  006640 010046                            MOV     R0,-(SP)
 1970  006642 162716 000003                     SUB     #3,(SP)
 1971  006646 010602                            MOV     SP,R2
 1972  006650 010346                            MOV     R3,-(SP)
 1973  006652 160016                            SUB     R0,(SP)
 1974  006654 162716 000002                     SUB     #2,(SP)
 1975  006660 001413                            BEQ     READNULL
 1976  006662 004767 010364                     JSR     PC,MAKESTR
 1977  006666 012603                            MOV     (SP)+,R3
 1978  006670 010316                            MOV     R3,(SP)
 1979  006672 010600                            MOV     SP,R0
 1980  006674 062703 000003                     ADD     #3,R3
 1981  006700 110043                            MOVB    R0,-(R3)
 1982  006702 000300                            SWAB    R0
 1983  006704 110043                            MOVB    R0,-(R3)
 1984  006706 000402                            BR      READSTR
 1985  006710 005316                    READNULL:DEC    (SP)
 1986  006712 012616                            MOV     (SP)+,(SP)
 1987  006714 004767 012414             READSTR:JSR     PC,STOSVAR
 1988  006720 000656                            BR      READDUN
 1989                                           .ENDC
 1990
 1991
```

209

```
                                       ;************************************************
                                       ;************************************************
                                       ;************                        ************
                                       ;************   INTERUPT HANDLERS    ************
                                       ;************                        ************
                                       ;************************************************
                                       ;************************************************



                                       ;
                                       ; KBINT - KEYBOARD INTERUPT SERVICE
                                       ;
                                       ;  PASS ANY CHAR, TO BUFF, EXCEPT 000 (IGNORE)
                                       ;   AND 003 ('↑C'; SET FLAG FOR LATER ACTION,)
                                       ;   IF BUFF, FULL, IGNORE CHAR BUT ECHO BELL,
                                       ;
006722 004567 012134           KBINT:  JSR     R5,SAVRGI       ;CAUSE RTI ON REG, RESTORE
006726 016705 000000G                  MOV     USRAREA,R5
006732 016501 000102                   MOV     KBHD(R5),R1     ;KB BUFF, HDR,
006736 012704 177564                   MOV     #TPS,R4
006742 113700 177562                   MOVB    @#TKB,R0        ;GET THAT CHAR,
006746 042700 177600                   BIC     #177600,R0
006752 001442                          BEQ     KBIDUN          ;IGNORE 000
006754 120027 000003                   CMPB    R0,#003         ;'↑C'?
006760 001005                          BNE     KBCO
006762 105265 000106                   INCB    CNCFLG(R5)      ;MARK '↑C' HIT
006766 004767 001516                   JSR     PC,BUFCLR       ;CLEAR ALL BUFFERS
006772 000410                          BR      DOPUT
006774 120027 000017           KBCO:   CMPB    R0,#17          ;CHECK '↑O'
007000 001005                          BNE     DOPUT
007002 105165 000107                   COMB    CNOFLG(R5)
007006 001402                          BEQ     DOPUT           ;IF PRINT RESTORED,BYPASS
007010 004767 001550                   JSR     PC,BUFTP        ;CLEAR TELEPRINTER BUFFER
007014 004767 011574           DOPUT:  JSR     PC,PUTBYT
007020 103017                          BCC     KBIDUN          ;GOT IN
007022 105761 000012                   TSTB    BFSPEC(R1)
007026 001003                          BNE     S,0101
007030 110061 000012                   MOVB    R0,BFSPEC(R1)
007034 000207                          RTS     PC
007036 105714                  S,0101: TSTB    (R4)            ;NO ROOM: DO BELL
007040 100376                          BPL     S,0101
007042 042714 000100                   BIC     #100,(R4)       ;TEMP, CLR, INTER,
007046 112737 000007 177566            MOVB    #007,@#TPB      ;INSERT BELL
007054 105714                  S,0102: TSTB    (R4)            ;WAIT UNTIL DONE
007056 100376                          BPL     S,0102
007060 012737 000101 177560    KBIDUN: MOV     #101,@#TKS      ;RE-INIT, READER
007066 012714 000100                   MOV     #100,(R4)       ;ENAPLE TPTR, FOR ECHO
007072 000207                          RTS     PC              ;WILL RESTORE REGS, AND RTI
```

210


```
                                       ;
                                       ; TPINT - TELEPRINTER INTERUPT HANDLER
                                       ;
                                       ;  USED TO PRINT OUT TEXT AND ECHO TYPE-IN,
                                       ;  TO HANDLE CASE OF PENDING REQUESTS TO DO TEXT OUTPUT
                                       ;   AND ECHO, DOES THE ORDERLY THING BY FOLLOWING THIS
                                       ;   PRIORITY:
                                       ;       IF ECHOSP(R5) SHOWS NEED TO DO 2ND HALF OF PAIRED
                                       ;        CHARS, FOR ECHO (↑U, <CR><LF>), DO FIRST,
                                       ;       IF TEXT MESS, IN PROGRESS, FINISH IT (TO <LF>),
                                       ;       ELSE, ECHO,
                                       ;       ELSE, TURN OFF INTERUPTS,
                                       ;
007074 004567 011762           TPINT:  JSR     R5,SAVRGI       ;CAUSE RTI ON REG, RESTORE
007100 016705 000000G                  MOV     USRAREA,R5
007104 105765 000110                   TSTB    FILLCO(R5)
007110 001404                          BEQ     TPNXT
007112 105365 000110                   DECB    FILLCO(R5)
007116 005000                          CLR     R0
007120 000416                          BR      TYPE1
007122 012704 000104           TPNXT:  MOV     #ECHOSP,R4
007126 060504                          ADD     R5,R4           ;SAVED HERE THRU-OUT ROUTINE
007130 005714                          TST     (R4)            ;SPECIAL ECHO IN PROGRESS?
007132 001415                          BEQ     TRYECHO         ;NO
007134 113400                  SPECMES:MOVB    @(R4)+,R0       ;NEXT SP, ECHO CHAR,
007136 001412                          BEQ     CLRECHO
007140 005244                          INC     -(R4)           ;PT, TO NEXT CHAR,
007142 120065 000112           TYPE:   CMPB    R0,FILLCH(R5)
007146 001003                          BNE     TYPE1
007150 116565 000111 000110            MOVB    FILLNO(R5),FILLCO(R5)
007156 110037 177566           TYPE1:  MOVB    R0,@#TPB
007162 000207                          RTS     PC              ;RESTORE REGS, AND RETURN
007164 005044                  CLRECHO:CLR     -(R4)
007166 016501 000102           TRYECHO:MOV     KBHD(R5),R1     ;NO MESS, - ECHO?

007172 004767 005752                   JSR     PC,GET2BYT
007176 103011                          BCC     GOTECHO
007200 016501 000100           MSGSUP: MOV     TPHD(R5),R1     ;TRY MESSAGE
007204 004767 005740                   JSR     PC,GET2BYT
007210 103354                          BCC     TYPE            ;GOT ONE: GO DO IT
007212 042737 000100 177564    OFPTYPE:BIC     #100,@#TPS      ;CLEAR INTERUPT
007220 000207                          RTS     PC
                                       ;
007222 004367 001354           GOTECHO:JSR     R3,CHKCHR       ;CHECK THAT ECHO CHAR,
007226 000420                          BR      PRNTDEL         ;LINE DELETE
007230 000422                          BR      SETLFE          ;<CR>: <LF> AS ECHOSP
007232 000424                          BR      ECHOBA          ;RUB OUT: DO "←"
007234 000401                          BR      TPCN
007236 000741                          BR      TYPE
007240 120027 000003           TPCN:   CMPB    R0,#3           ;CHEC '↑C'
007244 001003                          BNE     TPCO
007246 012714 007325                   MOV     #CCMES,(R4)
007252 000730                          BR      SPECMES
007254 120027 000017           TPCO:   CMPB    R0,#17          ;CHECK '↑O'
007260 001342                          BNE     TRYECHO
```

211

```
 2097  007262  012714  007330                MOV     #CCMES,(R4)
 2098  007266  000722                       BR      SPECMES
 2099                                ;
 2100  007270  012714  007312        PRNTDEL:MOV     #DELMSG,(R4)    ;ST, ADDR, INTO ECHO SP(R5)
 2101  007274  000717                       BR      SPECMES
 2102                                ;
 2103  007276  012714  007322        SETLFE: MOV     #CRLFMES,(R4)
 2104  007302  000714                       BR      SPECMES
 2105                                ;
 2106  007304  112700  000137        ECHOBA: MOVB    #'-,R0
 2107  007310  000714                       BR      TYPE
 2108                                ;
 2109                                DELMSG:
 2110  007312  042040  046105  052105        .ASCII  ' DELETED'
       007320  042105
 2111  007322     015     012     000 CRLFMES:.BYTE   015,012,000
 2112  007325     136     103         CCMES:  .ASCII  '^C'
 2113  007327     000                 .BYTE 0
 2114  007330  047536                COMES:  .ASCII  '\0'
 2115  007332     015     012     000         .BYTE   015,012,000
 2116          007336                 .EVEN
 2117
```

212


```
 2118                                        .IFNDF  $NOPTP
 2119                                ;
 2120                                ; PTRINT - PAPER-TAPE READER INTERUPT HANDLER
 2121                                ;
 2122                                ;  CLEARS PARITY BIT,  FLUSHES 000, RUB-OUT, AND <LF>;
 2123                                ;  IF CHAR; WON'T FIT IN BUFF., SAVE IN BFSPEC (LO BYTE),
 2124                                ;  ON EOT OR OTHER ERROR, SET HI BIT OF BFSPEC,
 2125                                ;
 2126  007336  004567  011520        PTRINT: JSR     R5,SAVRGI       ;CAUSE RTI ON REG, RESTORE
 2127  007342  012701  023714                MOV     #PRBFHD,R1      ;FOR PUTBYT
 2128  007346  005737  177550                TST     @#PRS           ;ERRORS?
 2129  007352  100425                       BMI     PREOT           ;YES; TREAT AS EOT
 2130  007354  113700  177552                MOVB    @#PRB,R0        ;CHARACTER
 2131  007360  042700  177000                BIC     #177600,R0
 2132  007364  001414                       BEQ     EXITB2          ;IGNORE 000 (CAN'T HAPPEN!)
 2133  007366  122700  000177                CMPB    #177,R0         ;RUB OUT?
 2134  007372  001411                       BEQ     EXITB2          ;IGNORE
 2135  007374  122700  000012                CMPB    #LF,R0
 2136  007400  001406                       BEQ     EXITB2          ;IGNORE LF'S
 2137  007402  004767  011206                JSR     PC,PUTBYT
 2138  007406  103003                       BCC     EXITB2          ;GOT IN
 2139  007410  110061  000012                MOVB    R0,BFSPEC(R1)   ;SAVE THIS CHAR,
 2140  007414  000207                       RTS     PC              ;RESTORE REGS, AND RTI
 2141  007416  012737  000101  177550 EXITB2: MOV    #101,@#PRS      ;RE-ENABLE READER
 2142  007424  000207                       RTS     PC
 2143  007426  052761  100000  000012 PREOT: BIS     #100000,BFSPEC(R1)      ;SHOW EOT
 2144  007434  000207                       RTS     PC
 2145                                        .ENDC
 2146
```

213

```
                                        .IFNDF  $NOPTP
                                    ;
                                    ; PPINT = PAPER TAPE PUNCH INTERUPT HANDLER
                                    ;
                                    ;  ON ERROR, MAKE BFSPEC NON-0,
                                    ;
007436  004567  011420          PPINT:  JSR     R5,SAVRGI       ;CAUSE RTI AFTER REG, RESTORE
007442  016705  000000G                  MOV     USRAREA,R5
007446  012701  023760                   MOV     #PPBFHD,R1
007452  005737  177554                   TST     @#PPS
007456  100406                          BMI     PPERR
007460  004767  005456                   JSR     PC,GET1BYT
007464  103406                          BCS     NOCHR2          ;NOTHING TO GET
007466  110037  177556                   MOVB    R0,@#PPB
007472  000207                          RTS     PC              ;RESTORE REGS, AND RTI
007474  052761  000001  000012  PPERR:  BIS     #1,BFSPEC(R1)   ;"NODIC" ERROR
007502  005037  177554          NOCHR2: CLR     @#PPS           ;CLEAR INTERUPT ENABLE
007506  000207                          RTS     PC
                                    ;
                                        .ENDC
```

214


```
                                        .IFNDF  $NOLPI
                                    ;
                                    ; LPINT = LINE PRINTER INTERUPT HANDLER
                                    ;
007510  004567  011346          LPINT:  JSR     R5,SAVRGI       ;CAUSE RTI AFTER REG, RESTORE
007514  016705  000000G                  MOV     USRAREA,R5
007520  012701  024024                   MOV     #LPBFHD,R1
007524  005737  177514                   TST     @#LPS
007530  100406                          BMI     LPERR
007532  004767  005404                   JSR     PC,GET1BYT
007536  103406                          BCS     LPOFF           ;NOTHING THERE
007540  110037  177516                   MOVB    R0,@#LPB        ;WRITE CHAR
007544  000207                          RTS     PC
                                    ;
007546  052761  000001  000012  LPERR:  BIS     #1,BFSPEC(R1)   ;RECORD ERROR
007554  005037  177514          LPOFF:  CLR     @#LPS           ;TURN IT OFF FOR AWHILE
007560  000207                          RTS     PC
                                        .ENDC
```

215

```
                                        .IFNDF  $NOPOW
                                ;
                                ; POWDWN - POWER FAIL/RESTART ROUTINE
                                ;
007562  012737  007572  000024  POWDWN: MOV     #POWUP,@#24
007570  000000                  HALT
                                ;
007572  005000                  POWUP:  CLR     R0      ;INITIALIZE LOOP COUNT
007574  016705  000000G         POWLP:  MOV     USRAREA,R5
007600  012737  007562  000024          MOV     #POWDWN,@#24
007606  016506  000004                  MOV     POL(R5),SP      ;WASTE TIME (AND INIT STACK!)
007612  005300                  DEC     R0              ;COUNT TILL DONE
007614  001367                  BNE     POWLP
007616  000167  171316          JMP     READY@
                                .ENDC
                                .EOT
```

216


```
                                ;                               SOURCE FILE #4
                                ;*******************************************************
                                ;*******************************************************
                                ;***********                                 ***********
                                ;***********      CLOSED SUBROUTINES         ***********
                                ;***********                                 ***********
                                ;*******************************************************
                                ;*******************************************************
                                ;+-----------------------------------------------------
                                ;  SUBROUTINE 'ADDSTK'  CALLED BY JSP PC
                                ;                       ADDS (SP)+,(SP)+ TO FAC1(R5),FAC2(R5) MATCHING TYPE
                                ;                       R0,R1  UNUSED
                                ;                       R2,R3  DESTROYED
                                ;                       R4  MODIFIED TO REFLECT STACK USAGE
                                ;                       R5  MUST POINT TO USER AREA
                                ;                       SP  GOES ?? DEEPER AFTER JSR
007622  012603                  ADDSTK: MOV     (SP)+,R3
007624  004767  005124                  JSR     PC,FIXUP
007630  001414                  BEQ     ADDINT
007632  012767  013214  000000G         MOV     #ERADD,$ERVEC
007640  004467  005250                  JSR     R4,FPPSAV
007644  016046                  ,WORD   PUSH
007646  000000G                 ,WORD   $ADR
007650  016060                  ,WORD   POP
007652  015066                  ,WORD   FPPRES
007654  005765  000040          ADDINT: TST     FAC1(R5)
007660  000113                  JMP     (R3)            ;COND CODES=SGN(RESULT)
007662  005726                  ADDINT: TST     (SP)+
007664  012602                  MOV     (SP)+,R2
007666  066502  000042                  ADD     FAC2(R5),R2
007672  102403                  BVS     ADDOVF
007674  010265  000042                  MOV     R2,FAC2(R5)
007700  000113                  JMP     (R3)            ;COND CODES=SGN(RESULT)
007702  100425                  ADDOVF: BMI     ADDPOS
007704  001416                  BEQ     ADDZERO
007706  005402                  NEG     R2
007710  105065  000042                  CLRB    FAC2(R5)
007714  110265  000043                  MOVB    R2,FAC2+1(R5)
007720  000302                  SWAB    R2
007722  110265  000040                  MOVB    R2,FAC1(R5)
007726  105065  000041                  CLRB    FAC1+1(R5)
007732  062765  143600  000040          ADD     #143600,FAC1(R5)
007740  000113                  JMP     (R3)            ;COND CODES=SGN(RESULT)
007742  005065  000042          ADDZERO:CLR     FAC2(R5)
007746  012765  144200  000040          MOV     #144200,FAC1(R5)
007754  000113                  JMP     (R3)            ;COND CODES=SGN(RESULT)
007756  105065  000042          ADDPOS: CLRB    FAC2(R5)
007762  110265  000043                  MOVB    R2,FAC2+1(R5)
007766  000302                  SWAB    R2
007770  110265  000040                  MOVB    R2,FAC1(R5)
```

217

```
 2260  007774  105065  000041                CLRB    FAC1+1(R5)
 2261  010000  062765  043000  000040        ADD     #43000,FAC1(R5)
 2262  010006  000113                        JMP     (R3)            ;COND CODES=SGN(RESULT)
 2263
```

218


```
 2264                                ;----------------------------------------------
 2265                                ;  SUBROUTINE 'ALLOC'   CALLED BY JSR PC
 2266                                ;                      ALLOCATES ARRAY SPACE FROM FREESPACE
 2267                                ;                      R0,R1  PRESERVED
 2268                                ;                      R2  MUST POINT TO VAR    GETS DESTROYED
 2269                                ;                      R3  MUST CONTAIN SS1MAX   GETS DESTROYED
 2270                                ;                      R4  MUST CONTAIN SS2MAX    GETS DESTROYED
 2271                                ;                      R5  MUST POINT TO USER AREA
 2272                                ;                      SP  GOES 22 DEEPER AFTER JSR
 2273  010010  010046                ALLOC:  MOV     R0,-(SP)
 2274  010012  010146                        MOV     R1,-(SP)
 2275  010014  005203                        INC     R3
 2276  010016  010401                        MOV     R4,R1
 2277  010020  005201                        INC     R1
 2278  010022  001776                        BEQ     .-2
 2279  010024  005000                        CLR     R0                      ;****
 2280  010026  012746  000020                MOV     #20.,-(SP)              ;*  *
 2281  010032  006301                ALLLOOP:ASL     R1                      ;*  *
 2282  010034  103002                        BCC     ALLNOAD                 ;*  *
 2283  010036  060300                        ADD     R3,R0                   ;*  *  MULTIPLY (R1)*(R3)*2
 2284  010040  103505                        BCS     ERRARAY                 ;*  *  RESULT IN R0
 2285  010042  006300                ALLNOAD:ASL     R0                      ;*  *  WATCH FOR OVERFLOWS
 2286  010044  103503                        BCS     ERRARAY                 ;*  *
 2287  010046  005316                        DEC     (SP)                    ;*  *
 2288  010050  003370                        BGT     ALLLOOP                 ;*  *
 2289  010052  005726                        TST     (SP)+                   ;****
 2290  010054  062700  000002                ADD     #2,R0
 2291  010060  103475                        BCS     ERRARAY         ;R0 NOW = 2 TIMES # ELEMENTS NEEDED,
 2292
 2293                                        .IFNDF  $NOSTM
 2294  010062  021227  177777                CMP     (R2),#,SVAR
 2295  010066  001402                        BEQ     .+6
 2296                                        .ENDC
 2297
 2298  010070  006300                        ASL     R0
 2299  010072  103470                        BCS     ERRARAY         ;R0 NOW = # BYTES NEEDED,
 2300  010074  016501  000012                MOV     HIFREE(R5),R1
 2301  010100  160001                        SUB     R0,R1
 2302  010102  103464                        BCS     ERRARAY
 2303  010104  020165  000074                CMP     R1,HISTR(R5)
 2304  010110  103461                        BLO     ERRARAY
 2305  010112  005721                        TST     (R1)+
 2306  010114  000401                        BR      .+4
 2307  010116  005021                        CLR     (R1)+
 2308  010120  020165  000012                CMP     R1,HIFREE(R5)
 2309  010124  101774                        BLOS    .-6
 2310  010126  160001                        SUB     R0,R1
 2311
 2312                                        .IFNDF  $NOSTM
 2313  010130  021227  177777                CMP     (R2),#,SVAR
 2314  010134  001413                        BEQ     ALLSTMN
 2315                                        .ENDC
 2316
 2317  010136  012722  177776                MOV     #,NVAR,(R2)+
 2318  010142  012221                        MOV     (R2)+,(R1)+
```

219

```
 2319  010144  012221                          MOV     (R2)+,(R1)+
 2320  010146  010412                          MOV     R4,(R2)
 2321  010150  005303                          DEC     R3
 2322  010152  010342                          MOV     R3,-(R2)
 2323  010154  010142                          MOV     R1,-(R2)
 2324  010156  162712  000004                  SUB     #4,(R2)
 2325
 2326                                          .IFNDF  $NOSTR
 2327  010162  000427                          BR      ALLEXIT
 2328  010164  005722                  ALLSTRNG:TST    (R2)+
 2329  010166  012221                          MOV     (R2)+,(R1)+
 2330  010170  005303                          DEC     R3
 2331  010172  010322                          MOV     R3,(R2)+
 2332  010174  010412                          MOV     R4,(R2)
 2333  010176  010162  177774                  MOV     R1,-4(R2)
 2334  010202  162762  000002  177774          SUB     #2,-4(R2)
 2335  010210  012721  177777                  MOV     #-1,(R1)+
 2336  010214  020165  000012                  CMP     R1,HIFREE(R5)
 2337  010220  101773                          BLOS    .-10
 2338  010222  160001                          SUB     R0,R1
 2339  010224  011102                          MOV     (R1),R2
 2340  010226  005202                          INC     R2
 2341  010230  001404                          BEQ     ALLEXIT
 2342  010232  000301                          SWAB    R1
 2343  010234  110122                          MOVB    R1,(R2)+
 2344  010236  000301                          SWAB    R1
 2345  010240  110112                          MOVB    R1,(R2)
 2346                                          .ENDC
 2347
 2348  010242  160065  000012          ALLEXIT:SUB     R0,HIFREE(R5)
 2349  010246  012601                          MOV     (SP)+,R1
 2350  010250  012600                          MOV     (SP)+,R0
 2351  010252  000207                          RTS     PC
 2352  010254  000004                  ERRARAY:IOT
 2353                                          .IFNDF  $LONGER
 2354  010256  052101     114                  .ASCII  \AT\
 2355                                          .ENDC
 2356                                          .IFDF   $LONGER
 2357                                          .ASCII  'ARRAYS TOO LARGE'
 2358                                          .ENDC
 2359  010261     000                          .BYTE   0
 2360                                          .EVEN
 2361
```

220


```
 2362                                          .IFNDF  $NOSTR
 2363                                  ;-------------------------------------------------
 2364                                  ; 'ARGB' SUBROUTINE
 2365                                  ;                       CALLED BY JSR R7
 2366                                  ;                       CALLS EVAL TO GET A 1-BYTE
 2367                                  ;                       ARGUMENT, BRANCHES TO ERRARG
 2368                                  ;                       IF ARG NO GOOD
 2369  010262  004767  001114          ARGB:   JSR     PC,EVAL
 2370  010266  103411                          BCS     ERRARG
 2371  010270  004767  005300                  JSR     PC,INT
 2372  010274  005765  000040                  TST     FAC1(R5)
 2373  010300  001004                          BNE     ERRARG
 2374  010302  105765  000043                  TSTB    FAC2+1(R5)
 2375  010306  001001                          BNE     ERRARG
 2376  010310  000207                          RTS     PC
 2377                                          .ENDC
 2378
 2379                                  ERSOR:
 2380                                  ERLOG:
 2381  010312  000004                  ERRARG: IOT
 2382                                          .IFDF   $LONGER
 2383                                          .ASCII  'ARGUMENT ERROR'
 2384                                          .ENDC
 2385                                          .IFNDF  $LONGER
 2386  010314  051101     107                  .ASCII  'ARG'
 2387                                          .ENDC
 2388  010317     000                          .BYTE   0
 2389                                          .EVEN
 2390
```

221

```
                                ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                ;  SUBROUTINE TBOMB     CALLED BY IOT
                                ;                       PRINTS ERROR MESSAGE FROM AFTER IOT
                                ;                       R0  DESTROYED
                                ;                       R1  PRESERVED
                                ;                       R2,R3,R4  UNUSED
                                ;                       R5  MUST POINT TO USER AREA
                                ;                       SP  RESET TO EMPTY STACK
                                ;                       PC  DOES NOT RETURN
010320  011601                  BOMB:   MOV     (SP),R1
010322  016705  000000G                 MOV     USRAREA,R5
010326  016506  000004                  MOV     PDL(R5),SP
010332  005065  000076                  CLR     ODEV(R5)        ;CAUSE MES OUTPUT TO TTY
010336  012765  000036  000034          MOV     #CLMNTTY,COLUMN(R5)     ;SET-UP TTY FOR COL, COUNT
010344  060565  000034                  ADD     R5,COLUMN(R5)
010350  004167  007876                  JSR     R1,MSG
010354      015     012                 .BYTE   CR,LF
                                        .IFNDF  $LONGER
010356      045                         .ASCII  /%/
                                        .ENDC
010357      000                         .BYTE 0
                                        .EVEN
010360  112100                  BOMBNX: MOVB    (R1)+,R0
010362  001403                          BEQ     BOMBDON
010364  004767  010316                  JSR     PC,PUTCHAR
010370  000773                          BR      BOMBNX
010372  026527  000030  177775  BOMBDON:CMP     LINENO(R5),#-1,SCALAR
010400  103041                          BHIS    BOMBJMP
010402  004167  007844                  JSR     R1,MSG
010406  040440  020124  044514          .ASCII  /  AT LINE /
010414  042516      040
010417      000                         .BYTE 0
                                        .EVEN
010420  016500  000030                  MOV     LINENO(R5),R0
010424  002405                          BLT     BOMBNEG
010426  010065  000042                  MOV     R0,FAC2(R5)
010432  005065  000040                  CLR     FAC1(R5)
010436  000414                          BR      BOMBOK
010440  110065  000043          BOMBNEG:MOVB    R0,FAC2+1(R5)
010444  105065  000042                  CLRB    FAC2(R5)
010450  000300                          SWAB    R0
010452  110065  000040                  MOVB    R0,FAC1(R5)
010456  105065  000041                  CLRB    FAC1+1(R5)
010462  062765  043600  000040          ADD     #43600,FAC1(R5)
010470  004767  007230          BOMBOK: JSR     PC,NUMOUT
010474  004167  006752                  JSR     R1,MSG
010500      015     012     000         .BYTE   CR,LF,0
                        010504          .EVEN
010504  000167  170434          BOMBJMP:JMP     READY
```

222


```
                                ;
                                ; BUFCLR--INIT I/O DEV, BUFFER HEADERS
                                ;
                                ; CALL: JSR     PC,BUFCLR
                                ;
                                ; USES R2 AND R3
                                ;
010510  016502  000102          BUFCLR: MOV     KBHD(R5),R2     ;KEYBOARD
010514  012203                          MOV     (R2)+,R3        ;BUFF, START
010516  005722                          TST     (R2)+           ;SKIP TO BGET1
010520  010322                          MOV     R3,(R2)+
010522  010322                          MOV     R3,(R2)+
010524  010322                          MOV     R3,(R2)+
010526  004767  000032                  JSR     PC,BUFTP        ;CLEAR TELEPRINTER BUFFER
                                        .IFNDF  $NOPTP
010532  012702  023760                  MOV     #PPBFHD,R2      ;PAPER TAPE PUNCH
010536  004767  004774                  JSR     PC,INITBF
010542  012702  023714                  MOV     #PRBFHD,R2      ;PAPER TAPE READER
010546  004767  004764                  JSR     PC,INITBF
                                        .ENDC
                                        .IFNDF  $NOLPT
010552  012702  024024                  MOV     #LPBFHD,R2
010556  004767  004754                  JSR     PC,INITBF
                                        .ENDC
010562  000207                          RTS     PC
                                ;
                                ; BUFTP--INIT TELEPRINTER BUFFER HEADER
                                ;
                                ; CALL: JSR     PC,BUFTP
                                ;
                                ; USES R2 AND R3
                                ;
010564  016502  000100          BUFTP:  MOV     TPHD(R5),R2     ;TELEPRINTER
010570  012203                          MOV     (R2)+,R3        ;BUFF, START
010572  022222                          CMP     (R2)+,(R2)+     ;PT, TO BGET2
010574  010322                          MOV     R3,(R2)+
010576  010322                          MOV     R3,(R2)+
010600  000207                          RTS     PC
```

223

```
                                        ;
                                        ;
                                        ; CHKCHR - CHECK ASCII CHAR, FOR SPECIAL CASES
                                        ;
                                        ; CALL: MOV     CCHAR,1,R0
                                        ;       JSR     R3,CHKCHR
                                        ;       INSTRUC,        ;EXECUTE IF LINE DEL, (<ALT MODE>,^U)
                                        ;       INSTRUC,        ;EXEC, IF <CR>
                                        ;       INSTRUC,        ;EXEC, IF <RUB OUT> (OR ^H)
                                        ;       INSTRUC,        ;EXEC, IF BELOW 40 OR ABOVE 137
                                        ;       INSTRUC,        ;EXEC, IF GOOD STUFF
                                        ;
                                        ;  USES R0
                                        ;
010602  120027  000176          CHKCHR: CMPB    R0,#176         ;ALT MODE
010606  001434                          BEQ     CCEXIT
010610  120027  000175                  CMPB    R0,#175         ;ALT MODE
010614  001431                          BEQ     CCEXIT
010616  120027  000033                  CMPB    R0,#33          ;ALT MODE
010622  001426                          BEQ     CCEXIT
010624  120027  000025                  CMPB    R0,#25          ;^U
010630  001423                          BEQ     CCEXIT
010632  005723                          TST     (R3)+           ;TO <CR> RETURN
010634  120027  000015                  CMPB    R0,#CR
010640  001417                          BEQ     CCEXIT
010642  005723                          TST     (R3)+           ;TO RUB OUT RETURN
010644  120027  000177                  CMPB    R0,#177         ;RUB OUT
010650  001413                          BEQ     CCEXIT
010652  120027  000137                  CMPB    R0,#'_
010656  001410                          BEQ     CCEXIT
010660  005723                          TST     (R3)+           ;TO LIMIT RETURN
010662  120027  000040                  CMPB    R0,#40
010666  103404                          BLO     CCEXIT
010670  120027  000137                  CMPB    R0,#137
010674  101001                          BHI     CCEXIT
010676  005723                          TST     (R3)+           ;REG, CHAR, RETURN
010700  000203          CCEXIT: RTS     R3
```

224


```
                                        ;
                                        ; CHKISET - CHECK I/O CHANNEL AND SET-UP INPUT (IDEV(R5))
                                        ; CHKOSET - CHECK I/O CHANNEL AND SET-UP OUTPUT (ODEV(R5))
                                        ;
                                        ; CALL: MOV     CADDR, OF CODE LINE PTR, AFTER '#(),R1
                                        ;       JSR     PC,CHKISET (CHKOSET)
                                        ;
                                        ;  USES: R0,R1,R2,R3,R5
                                        ;
                                        ;  CHECK IF SPECIFIED CHANNEL LEGAL (NUMBER IN RANGE AND I/O
                                        ;   DEVICE ASSEMBLED IN),  RETURN CODE FOR SPECIFIED DEVICE
                                        ;   (SAME AS VALUE OF IDEV (ODEV)) IN R2,  ON EXIT, R1 PTS
                                        ;   TO BYTE AFTER CHANNEL NUMBER.
                                        ;
010702  012700  010777          CHKISET:MOV     #ITABLE,R0
010706  000402                          BR      COMCHK
010710  012700  010774          CHKOSET:MOV     #OTABLE,R0
010714  004767  000462          COMCHK: JSR     PC,EVAL
010720  103420                          BCS     ERSX12
010722  005765  000040                  TST     FAC1(R5)
010726  001017                          BNE     ERCHAN          ;MUST BE INTEGER
010730  016502  000042                  MOV     FAC2(R5),R2
010734  001411                          BEQ     OUT012          ;TTY - EVERYTHING SET-UP
010736  111003                          MOVB    (R0),R3         ;FIRST TABLE ENTRY
010740  060503                          ADD     R5,R3           ;PTS, TO ACTUAL BYTE
010742  020227  000002                  CMP     R2,#2           ;MAX CHANNEL NUMBER (CURRENTLY)
010746  101007                          BHI     ERCHAN
010750  060200                          ADD     R2,R0           ;ENTRY IN TABLE OF DEV, CODE
010752  111013                          MOVB    (R0),(R3)       ;CODE INTO IDEV OR ODEV
010754  001404                          BEQ     ERCHAN          ;MEANS NO DEVICE THERE
010756  111002                          MOVB    (R0),R2
010760  000207          OUT012: RTS     PC
                                        ;
010762  000167  001556          ERSX12: JMP     ERRSX2
                                        ;
010766  000004                  ERCHAN: IOT
                                        ;IFNDF  $LONGER
010770  041504      105                 ;ASCII  \DCE\
                                        ;ENDC
                                        ;IFDF   $LONGER
                                        ;ASCII  /DEVICE CHANNEL ERROR/
                                        ;ENDC
010773      000                         ;BYTE   0
                                        ;EVEN
                                        ;
                                        ;
010774      076                 OTABLE: ;BYTE   ODEV
                                        ;IFDF   $NOPTP
                                        ;BYTE   0
                                        ;ENDC
                                        ;IFNDF  $NOPTP
010775      002                         ;BYTE   2
                                        ;ENDC
                                        ;IFDF   $NOLPI
                                        ;BYTE   0
```

225

```
                                        ;ENDC
                                        ;IFNDF  $NOLPI
010776  004                             ;BYTE   4
                                        ;ENDC
                                ;
ITABLE: ;BYTE   IDEV
                                        ;IFDF   $NOPTR
                                        ;BYTE   8
                                        ;ENDC
                                        ;IFNDF  $NOPTR
011000  002                             ;BYTE   2
                                        ;ENDC
011001  000                             ;BYTE   0
```

226


```
                                ;+-----------------------------------------------
                                ;  SUBROUTINE 'CLRVARS' CALLED BY JSP PC
                                ;                       CLEARS VARS TO SCALAR ZEROS,NULL STRINGS
                                ;                       ALSO INITIALIZES RANDOM NO GEN
                                ;                       R0  DESTROYED
                                ;                       R1,R2,R3,R4  UNUSED
                                ;                       R5  MUST POINT TO USER AREA
                                ;                       SP  GOES NO DEEPER AFTER JSR
011002  012765  032331  000064  CLRVARS:MOV     #32331,RND1(R5)
011010  012765  163251  000066          MOV     #163251,RND2(R5)
011016  011500                  CLRLOOP:MOV     (R5),R0
011020  020065  000014                  CMP     R0,LOFREE(R5)
011024  103021                          BHIS    CLRFREE
011026  021027  177775                  CMP     (R0),#,SCALAR
011032  103412                          BLO     CLROK
                                ;
                                        ;IFNDF  $NOSTR
011034  021027  177777                  CMP     (R0),#,SVAR
011040  001404                          BEQ     CLRSVAR
                                        ;ENDC
                                ;
011042  012720  177775                  MOV     #,SCALAR,(R0)+
011046  005010                          CLR     (R0)
                                ;
                                        ;IFNDF  $NOSTR
011050  000401                          BR      .+4
011052  012010          CLRSVAR:MOV     (R0)+,(R0)
                                        ;ENDC
                                ;
011054  012010                          MOV     (R0)+,(R0)
011056  012010                          MOV     (R0)+,(R0)
011060  062700  000004          CLROK:  ADD     #4,R0
011064  000755                          BR      CLRLOOP
011066  005020                          CLR     (R0)+
011070  020065  000010          CLRFREE:CMP     R0,ARRAYS(R5)
011074  101774                          BLOS    .-6
011076  016565  000010  000012          MOV     ARRAYS(R5),HIFREE(R5)
011104  016565  000014  000072          MOV     LOFREE(R5),LOSTR(R5)
011112  016565  000014  000074          MOV     LOFREE(R5),HISTR(R5)    ;EMPTY STRING STORAGE
011120  012765  000033  000032          MOV     #33,GSBCTR(R5)
011126  016500  000004                  MOV     POL(R5),R0
011132  012720  177777                  MOV     #-1,(R0)+
011136  005020                          CLR     (R0)+
011140  020065  000002                  CMP     R0,LIMIT(R5)
011144  101774                          BLOS    .-6
011146  000207                          RTS     PC
```

227

```
                                ;*****************************************
                                ;  SUBROUTINE 'DIVTEN'  CALLED BY JSR PC
                                ;                       DIVIDES R3,R4 BY DECIMAL 10
                                ;                       R0,R1,R2  UNUSED
                                ;                       R3,R4  IS THE 31 BIT UNSIGNED INTEGER TO DIVIDE
                                ;                       R5  UNUSED
                                ;                       SP  GOES 2 DEEPER AFTER JSR
011150  012746  000034          DIVTEN: MOV     #34,-(SP)
011154  022703  050000          DIVLOOP:CMP     #50000,R3
011160  101002                          BHI     .+6
011162  062703  130000                  ADD     #130000,R3
011166  006104                          ROL     R4
011170  006103                          ROL     R3
011172  005316                          DEC     (SP)
011174  003367                          BGT     DIVLOOP
011176  042703  170000                  BIC     #170000,R3
011202  005726                          TST     (SP)+
011204  000207                          RTS     PC
```

228


```
                                        .IFNDF  $NOSTK
                                ;*****************************************
                                ;  SUBROUTINE 'DNPACK'  CALLED BY JSP PC
                                ;                       PACKS STRING STORAGE TOWARD LOW CORE
                                ;                       R0  UNUSED
                                ;                       R1,R2,R3  PRESERVED
                                ;                       R4  UNUSED
                                ;                       R5  MUST POINT TO USER AREA
                                ;                       SP  GOES 1 DEEPER AFTER JSR
011206  010146                  DNPACK: MOV     R1,-(SP)        ;TO UNDERSTAND THESE ROUTINES IT IS HELPFUL TO
011210  010246                          MOV     R2,-(SP)        ;KNOW THAT NON-NULL STRINGS ARE STORED AS
011212  010346                          MOV     R3,-(SP)        ;(LENGTH,2 BYTE BACKPTR,N BYTE STRING,LENGTH)
011214  005046                          CLR     -(SP)   ;WHERE LENGTH IS THE VALUE OF N AND IF BACKPTR
011216  016502  000072                  MOV     LOSTR(R5),R2    ;IS ODD,THEN IT IS RELATIVE TO SYMBOLS,
011222  016501  000014                  MOV     LOFREE(R5),R1
011226  010165  000072                  MOV     R1,LOSTR(R5)
011232  005016          DNPLOOP:CLR     (SP)
011234  152216                          BISB    (R2)+,(SP)
011236  001012                          BNE     DNPNZRO
011240  020265  000074          DNPBAD: CMP     R2,HISTR(R5)
011244  103772                          BLO     DNPLOOP
011246  010165  000074                  MOV     R1,HISTR(R5)    ;SAVE HIGH STRING ADDRESS
011252  005726                          TST     (SP)+
011254  012603                          MOV     (SP)+,R3
011256  012602                          MOV     (SP)+,R2
011260  012601                          MOV     (SP)+,R1
011262  000207                          RTS     PC
011264  005003          DNPNZRO:CLR     R3
011266  152203                          BISB    (R2)+,R3
011270  000303                          SWAB    R3
011272  152203                          BISB    (R2)+,R3
011274  061602                          ADD     (SP),R2
011276  005202                          INC     R2
011300  030327  000001                  BIT     R3,#1
011304  001402                          BEQ     .+6
011306  005303                          DEC     R3
011310  061503                          ADD     (R5),R3
011312  020365  000004                  CMP     R3,PDL(R5)
011316  103350                          BHIS    DNPBAD
011320  020306                          CMP     R3,SP
011322  103013                          BHIS    DNPGOOD
011324  020365  000010                  CMP     R3,ARRAYS(R5)
011330  101343                          BHI     DNPBAD
011332  020365  000012                  CMP     R3,HIFREE(R5)
011336  101005                          BHI     DNPGOOD
011340  020365  000014                  CMP     R3,LOFREE(R5)
011344  103335                          BHIS    DNPBAD
011346  020315                          CMP     R3,(R5)
011350  103733                          BLO     DNPBAD
011352  062716  000004          DNPGOOD:ADD     #4,(SP)
011356  161602                          SUB     (SP),R2
011360  020213                          CMP     R2,(R3)
011362  001005                          BNE     DNPIGNO
011364  010113                          MOV     R1,(R3)
011366  112221                          MOVB    (R2)+,(R1)+
```

229

```
 2711  011370  005316                          DEC     (SP)
 2712  011372  003375                          BGT     .+4
 2713  011374  000721                          BR      ONPBAD
 2714  011376  061602                ONPIGNO:  ADD     (SP),R2
 2715  011400  000717                          BR      ONPBAD
 2716                                          .ENDC
 2717                                          .EOT
 2718
```

230


```
 2719                                ;                                SOURCE FILE #5
 2720                                ;*****************************************************
 2721                                ;
 2722                                ; EVAL - EVALUATE EXPRESSION
 2723                                ;
 2724                                ; CALL: MOV     [USER-AREA],
 2725                                ;       MOV     [STACK SIZE],R4
 2726                                ;       MOV     [CHAR. PTR.],R1
 2727                                ;       JSR     PC,EVAL
 2728                                ;
 2729                                ;  USES ALL REGS,
 2730                                ;
 2731                                ;  RETURNS 'C' BIT SET IF STRING VAL,
 2732                                ;                  CLR IF NOT STRING
 2733                                ;
 2734                                ;  RETURNS VALUE IN FAC1(R5), FAC2(R5)
 2735                                ;
 2736  011402  020604                EVAL:   CMP     SP,R4
 2737  011404  103406                        BLO     BPDL
 2738  011406  012746  000235                MOV     #,TERM,-(SP)
 2739  011412  000404                        BR      OPERAND
 2740  011414  012746  000233        UMINUS: MOV     #,UNARY,-(SP)
 2741  011420  020604                        CMP     SP,R4
 2742  011422  103563                BPDL:   BLO     ERRPDL
 2743  011424  112102                OPERAND:MOVB    (R1)+,R2
 2744  011426  002141                        BGE     VARBLE
 2745  011430  120227  000375                CMPB    R2,#,ILIT1
 2746  011434  001475                        BEQ     ILIT1
 2747  011436  120227  000376                CMPB    R2,#,ILIT2
 2748  011442  001502                        BEQ     ILIT2
 2749  011444  120227  000374                CMPB    R2,#,FLIT
 2750  011450  001502                        BEQ     FLIT
 2751  011452  120227  000255                CMPB    R2,#,LPAR
 2752  011456  001510                        BEQ     LPAR
 2753  011460  120227  000234                CMPB    R2,#,MINUS
 2754  011464  001753                        BEQ     UMINUS
 2755  011466  120227  000232                CMPB    R2,#,PLUS
 2756  011472  001754                        BEQ     OPERAND
 2757  011474  120227  000262                CMPB    R2,#,FN
 2758  011500  001537                        BEQ     GOTOFN
 2759
 2760                                        .IFNDF  $NOSTR
 2761  011502  120227  000257                CMPB    R2,#,SQUOT
 2762  011506  001403                        BEQ     GOTOQT
 2763  011510  120227  000256                CMPB    R2,#,DQUOT
 2764  011514  001002                        BNE     NOQUOTE
 2765  011516  000167  001030        GOTOQT: JMP     QUOTE
 2766                                        .ENDC
 2767
 2768  011522  042702  177400        NOQUOTE:BIC     #177400,R2
 2769  011526  162702  000263                SUB     #,RNDL,R2
 2770  011532  006302                        ASL     R2
 2771  011534  020227  000044                CMP     R2,#TBL3END-TABLF3
 2772  011540  101065                        BHI     ERRSX4
 2773  011542  062702  000012                ADD     #12,R2
```

231

```
2774  011546  060702                        ADD     PC,R2
2775  011550  005712                        TST     (R2)
2776  011552  003002                        BGT     .+6
2777  011554  000167  002740                JMP     ERRUFN
2778  011560  061227                        ADD     (R2),PC
2779  011562  000000              TABLE5:   .WORD   0                   ; RND(
2780  011564  000000                        .WORD   0                   ; RND
2781  011566  001440                        .WORD   SINFN-TABLE5
2782  011570  001446                        .WORD   COSFN-TABLE5
2783  011572  001454                        .WORD   SQRFN-TABLE5
2784  011574  001470                        .WORD   ATNFN-TABLE5
2785  011576  001476                        .WORD   EXPFN-TABLE5
2786  011600  001512                        .WORD   LOGFN-TABLE5
2787  011602  000000                        .WORD   0                   ; ABS
2788  011604  001662                        .WORD   INTFN-TABLE5
2789  011606  000000                        .WORD   0                   ; SGN
2790
2791                                        .IFNDF  $NOSTR
2792  011610  000000                        .WORD   0                   ; TAB
2793  011612  000000                        .WORD   0                   ; LEN
2794  011614  000000                        .WORD   0                   ; ASC
2795  011616  000000                        .WORD   0                   ; CHR$
2796  011620  000000                        .WORD   0                   ; POS
2797  011622  000000                        .WORD   0                   ; SEG
2798  011624  000000                        .WORD   0                   ; VAL
2799  011626  000000                        .WORD   0                   ; STR
2800                                        .ENDC
2801
2802            011626      TBL5END =.-2
2803  011630  005065  000040  ILIT1:   CLR     FAC1(R5)
2804  011634  105065  000043          CLRB    FAC2+1(R5)
2805  011640  112165  000042          MOVB    (R1)+,FAC2(R5)
2806  011644  000167  000316          JMP     OPRATOR
2807  011650  005065  000040  ILIT2:   CLR     FAC1(R5)
2808  011654  000406                  BR      ILITCOM
2809  011656  112165  000041  FLIT:    MOVB    (R1)+,FAC1+1(R5)
2810  011662  112165  000040          MOVB    (R1)+,FAC1(R5)
2811  011666  112165  000043  ILITCOM: MOVB    (R1)+,FAC2+1(R5)
2812  011672  112165  000042          MOVB    (R1)+,FAC2(R5)
2813  011676  000533                  BR      OPRATOR
2814  011700  004767  177476  LPAR:    JSR     PC,EVAL
2815
2816                                        .IFNDF  $NOSTR
2817  011704  103405                        BCS     LPSTRNG
2818                                        .ENDC
2819
2820  011706  122127  000237                CMPB    (R1)+,#,RPAR
2821  011712  001525                        BEQ     OPRATOR
2822  011714  000167  000624  ERRSX4:  JMP     ERRSXP
2823
2824                                        .IFNDF  $NOSTR
2825  011720  122127  000237  LPSTRNG: CMPB    (R1)+,#,RPAR
2826  011724  001373                        BNE     ERRSX4
2827  011726  000167  000766                JMP     SOPRATR
2828                                        .ENDC
```

232


```
2829
2830  011732  000302              VARBLE:  SWAB    R2
2831  011734  152102                       BISB    (R1)+,R2
2832  011736  061502                       ADD     (R5),R2             ;COLLECT OFFSET
2833  011740  121127  000255               CMPB    (R1),#,LPAR
2834  011744  001421                       BEQ     VARSS
2835
2836                                        .IFNDF  $NOSTR
2837  011746  021227  177777                CMP     (R2),#,SVAR
2838  011752  001405                       BEQ     STRGJMP
2839                                        .ENDC
2840
2841  011754  022227  177775                CMP     (R2)+,#,SCALAR
2842  011760  001476                       BEQ     VARNOSS
2843  011762  011202                       MOV     (R2),R2
2844  011764  000474                       BR      VARNOSS
2845
2846                                        .IFNDF  $NOSTR
2847  011766  000167  000666  STRGJMP: JMP     STRGVAR
2848                                        .ENDC
2849
2850  011772  000004              ERRPDL:  IOT
2851                                        .IFNDF  $LONGER
2852  011774  052105      103               .ASCII  \ETC\
2853                                        .ENDC
2854                                        .IFDF   $LONGER
2855                                        .ASCII  'EXPRESSION TOO COMPLEX'
2856                                        .ENDC
2857  011777      000                       .BYTE 0
2858                                        .EVEN
2859  012000  000167  002246  GOTOFN:  JMP     FNFN
2860
2861                                        .IFNDF  $NOSTR
2862  012004  000167  001002  ERRMX2:  JMP     ERRMIX
2863                                        .ENDC
2864
2865  012010  005201              VARSS:   INC     R1
2866  012012  003767                       BLE     ERRPDL
2867  012014  010246                       MOV     R2,-(SP)
2868  012016  004767  177360                JSR     PC,EVAL
2869
2870                                        .IFNDF  $NOSTR
2871  012022  103770                       BCS     ERRMX2
2872                                        .ENDC
2873
2874  012024  004767  003544                JSR     PC,INT
2875  012030  005765  000040                TST     FAC1(R5)
2876  012034  001030                       BNE     ERRSS2
2877  012036  121127  000237                CMPB    (R1),#,RPAR
2878  012042  001427                       BEQ     VONESS
2879  012044  122127  000243                CMPB    (R1)+,#,COMMA
2880  012050  001321                       BNE     ERRSX4
2881  012052  016546  000042                MOV     FAC2(R5),-(SP)
2882  012056  004767  177320                JSR     PC,EVAL
2883
```

233

```
                                        .IFNDF  $NOSTK
012862  103758                          BCS     ERRMX2
                                        .ENDC

012864  004767  003504                  JSR     PC,INT
012870  005765  000040                  TST     FAC1(R5)
012874  001010                          BNE     ERRSS2
012876  122127  000237                  CMPB    (R1)+,#,RPAR
012102  001304                          BNE     ERRSX4
012104  016503  000042                  MOV     FAC2(R5),R3
012110  100402                          BMI     ERRSS2
012112  012600                          MOV     (SP)+,R0
012114  000407                          BR      VTWOSS
012116  000167  004454          ERRSS2: JMP     ERRSS3
012122  005201          VONESS: INC     R1
012124  012703  177777                  MOV     #-1,R3
012130  016500  000042                  MOV     FAC2(R5),R0
012134  100770          VTWOSS: BMI     ERRSS2
012136  012602                          MOV     (SP)+,R2

                                        .IFNDF  $NOSTK
012140  021227  177777                  CMP     (R2),#,SVAR
012144  001002                          BNE     .+6
012146  000167  000500                  JMP     STRGAHR
                                        .ENDC

012152  004767  004232          VARNOSS: JSR    PC,LOCGET
012156  012265  000040          VARNOS: MOV     (R2)+,FAC1(R5)
012162  011265  000042                  MOV     (R2),FAC2(R5)
012166  111103          OPRATOR: MOVB   (R1),R3         ;NEXT CHARACTER,
012170  120327  000201                  CMPB    R3,#,EOL
012174  001437                          BEQ     DOITNOW
012176  120327  000205                  CMPB    R3,#,GOTO
012202  001434                          BEQ     DOITNOW
012204  120327  000227                  CMPB    R3,#,UPARR0
012210  103641                          BLO     ERRSX4
012212  011602                          MOV     (SP),R2         ;PREVIOUS OPERATOR,
012214  120227  000227                  CMPB    R2,#,UPARR0
012220  101425                          BLOS    DOITNOW         ;JUMP IF *,
012222  120227  000231                  CMPB    R2,#,SLASH
012226  101417                          BLOS    PREC2           ;JUMP IF * OR /,
012230  120227  000234                  CMPB    R2,#,MINUS
012234  101411                          BLOS    PREC3           ;JUMP IF + - OR UNARY,
012236  120227  000254                  CMPB    R2,#,LQ
012242  101403                          BLOS    PREC4           ;JUMP IF = <> <= >= < OR >,
012244  120327  000254          PREC5:  CMPB    R3,#,LQ
012250  101427                          BLOS    NOTNOW          ;JUMP IF ANY OPERATOR,
012252  120327  000234          PREC4:  CMPB    R3,#,MINUS
012256  101424                          BLOS    NOTNOW          ;JUMP IF + - UNARY * /  OR ^,
012260  120327  000231          PREC3:  CMPB    R3,#,SLASH
012264  101421                          BLOS    NOTNOW          ;JUMP IF * / OR ^,
012266  120327  000227          PREC2:  CMPB    R3,#,UPARR0
012272  101416                          BLOS    NOTNOW          ;JUMP IF ^,
012274  005002          DOITNOW: CLR    R2                      ;(E,G,  A+B+ ) DO THE * NOW,
012276  152602                          BISB    (SP)+,R2
```

234


```
012300  162702  000226                  SUB     #,UPARR0-1,R2
012304  060202                          ADD     R2,R2
012306  060702                          ADD     PC,R2           ;JUMP TO DO THE APPROPRIATE OPERATION,
012310  061207                          ADD     (R2),PC         ;LEFT OPERAND IS 0(SP),2(SP),
012312  001102          TABLE2: .WORD   UPARR0-TABLE2   ;RIGHT OPERAND IS FAC1(R5),FAC2(R5),
012314  000124                  .WORD   STAR-TABLE2
012316  000154                  .WORD   SLASH-TABLE2
012320  000116                  .WORD   PLUS-TABLE2
012322  000056                  .WORD   UNARY-TABLE2
012324  000052                  .WORD   MINUS-TABLE2
012326  000042                  .WORD   TERMIN-TABLE2
012330  016546  000042          NOTNOW: MOV     FAC2(R5),-(SP)  ;(E,G,  A+B+ ) DONT DO THE + YET,
012334  020604                          CMP     SP,R4
012336  103410                          BLO     ERRPDS
012340  016546  000040                  MOV     FAC1(R5),-(SP)
012344  010346                          MOV     R3,-(SP)
012346  005201                          INC     R1
012350  000167  177050                  JMP     OPERAND
012354  000241          TERMIN: CLC                     ;CARRY OFF MEANS NUMERIC RESULT,
012356  000207                          RTS     PC
012360  000167  177406          ERRPDS: JMP     ERRPDL
012364  004767  007060          MINUS:  JSR     PC,SUBSTK
012370  005765  000040          UNARY:  TST     FAC1(R5)
012374  001404                          BEQ     UINTEG
012376  062765  100000  000040          ADD     #100000,FAC1(R5)
012404  000670                          BR      OPRATOR
012406  005465  000042          UINTEG: NEG     FAC2(R5)
012412  102265                          BVC     OPRATOR
012414  012765  044000  000040          MOV     #44000,FAC1(R5)
012422  005065  000042                  CLR     FAC2(R5)
012426  000657                          BR      OPRATOR
012430  004767  175166          PLUS:   JSR     PC,ADDSTK
012434  000654                          BR      OPRATOR
012436  012767  013214  000000G STAR:   MOV     #ERSTAR,SERVEC
012444  004467  002444                  JSR     R4,FPPSAV
012450  012514                          .WORD   TSTSTK          ;TEST TOP OF STACK AND FLOAT IF INT
012452  012530                          .WORD   PUSHF           ;PUSH FAC AND FLOAT IF NEC
012454  000000G                         .WORD   $MLR            ;PERFORM MULT
012456  016068                          .WORD   POP             ;POP RESULT
012460  015066                          .WORD   FPPRES
012462  000167  177500          STAR1:  JMP     OPRATOR
012466  012767  013576  000000G SLASH:  MOV     #ERRDIV,SERVEC
012474  004467  002414                  JSR     R4,FPPSAV
012500  012514                          .WORD   TSTSTK
012502  012530                          .WORD   PUSHF
012504  000000G                         .WORD   $DVR
012506  016068                          .WORD   POP
012510  015066                          .WORD   FPPRES
012512  000763                          BR      STAR1
012514  005716          TSTSTK: TST     (SP)
012516  001003                          BNE     TSTS2
012520  005726          TSTS1:  TST     (SP)+
012522  000167  000000G                 JMP     $IR
012526  000134          TSTS2:  JMP     @(R4)+          ;RETURN
012530  016546  000042          PUSHF:  MOV     FAC2(R5),-(SP)
```

235

```
2994  012534  016546  000040            MOV     FAC1(R5),-(SP)
2995  012540  001767                    BEQ     TSTSL           ;FLOAT IF INTEGER
2996  012542  000134                    JMP     @(R4)+          ;RETURN
2997                                    .IFDF   $NOSTR
2998                            ;DUMMY ENTRY POINTS FOR $NOSTR
2999                            SAVCHAR:
3000                            ARGB:
3001                            MAKEST:
3002                            SOPRAT:
3003                            STOSVAR:
3004                            ERRMIX:
3005                            STPRO:
3006                                    .ENDC
3007
3008                            ERRSYN:
3009  012544  000004            ERRSX5: IOT
3010                                    .IFNDF  $LONGER
3011  012546  054523  000116            .ASCII  'SYN'
3012                                    .ENDC
3013                                    .IFDF   $LONGER
3014                                    .ASCII  'SYNTAX ERROR'
3015
3016                                    .ENDC
3017  012551  000                       .BYTE 0
3018                                    .EVEN
3019
3020                                    .IFNDF  $NOSTR
3021  012552  122127  000377    QUOTE:  CMPB    (R1)+,#,TEXT
3022  012556  001372                    BNE     ERRSX5
3023  012560  010203                    MOV     R2,R3           ;SAVE QUOTE CHAR,
3024  012562  010102                    MOV     R1,R2
3025  012564  105721                    TSTB    (R1)+
3026  012566  001376                    BNE     .-2
3027  012570  020604                    CMP     SP,R4
3028  012572  103672                    BLO     ERRPD2
3029  012574  010246                    MOV     R2,-(SP)
3030  012576  162716  000003            SUB     #3,(SP)
3031  012602  010146                    MOV     R1,-(SP)
3032  012604  160216                    SUB     R2,(SP)
3033  012606  122103                    CMPB    (R1)+,R3                ;SAME AS STARTING QUOTE CHAR,?
3034  012610  001355                    BNE     ERRSX5
3035  012612  005316                    DEC     (SP)
3036  012614  001410                    BEQ     QUNULL
3037  012616  010602                    MOV     SP,R2
3038  012620  062702  000002            ADD     #2,R2
3039  012624  004767  004422            JSR     PC,MAKESTR
3040  012630  012616                    MOV     (SP)+,(SP)
3041  012632  000167  000324            JMP     QUOTBUM
3042  012636  022626            QUNULL: CMP     (SP)+,(SP)+
3043  012640  012746  177777    GOTVAL: MOV     #-1,-(SP)
3044  012644  000425                    BR      SOPRATR
3045                                    .ENDC
3046
3047  012646  000167  000330    ERRD02: JMP     ERRPD2
3048
```

236


```
3049
3050  012652  004767  003532            .IFNDF  $NOSTM
3051  012656  000406    STRGARR:JSR     PC,LOCGET
3052  012660  005722                    BR      STRGBTH
3053  012662  026227  000002  177777    TST     (R2)+
3054  012670  001421            STRGVAR:CMP     2(R2),#-1
3055  012672  011202                    BEQ     STRGBTH
3056  012674  011203                    MOV     (R2),R2
3057  012676  020327  177777    STRGBTH:MOV     (R2),R3
3058  012702  001756                    CMP     R3,#-1
3059  012704  020604                    BEQ     GOTVAL
3060  012706  103757                    CMP     SP,R4
3061  012710  005046                    BLO     ERRD02
3062  012712  111316                    CLR     -(SP)
3063  012714  004767  004332            MOVB    (R3),(SP)
3064  012720  126627  000002  000226    JSR     PC,MAKESTR
3065  012726  001434    SOPRATR:CMPB    2(SP),#,AMPERS
3066  012730  121127  000226            BEQ     CONCAT
3067  012734  001422                    CMPB    (R1),#,AMPERS
3068  012736  126627  000002  000235    BEQ     AMPWAIT
3069  012744  001277                    CMPB    2(SP),#,TERM
3070  012746  012600                    BNE     ERRSX5
3071  012750  005726                    MOV     (SP)+,R0
3072  012752  011602                    TST     (SP)+
3073  012754  010016                    MOV     (SP),R2
3074  012756  005200                    MOV     R0,(SP)
3075  012760  001406                    INC     R0
3076  012762  062700  000002            BEQ     SOPRX           ;CHECK NULL STRING
3077  012766  010603                    ADD     #2,R0
3078  012770  110340                    MOV     SP,R3
3079  012772  000303                    MOVB    R3,-(R0)
3080  012774  110340                    SWAB    R3
3081  012776  000261    SOPRX:  SEC     MOVB    R3,-(R0)
3082  013000  000112                    JMP     (R2)
3083  013002  112146    AMPWAIT:MOVB    (R1)+,-(SP)
3084  013004  004767  176372            JSR     PC,EVAL
3085  013010  103743                    BCS     SOPRATR
3086  013012  000004    ERRMIX: IOT
3087                                    .IFNDF  $LONGER
3088  013014  051516  000115            .ASCII  \NSM\
3089                                    .ENDC
3090                                    .IFDF   $LONGER
3091                                    .ASCII  'NUMBERS AND STRINGS MIXED'
3092                                    .ENDC
3093  013017  000                       .BYTE 0
3094                                    .EVEN
3095  013020  021627  177777    CONCAT: CMP     (SP),#-1
3096  013024  001002                    BNE     CATNOT
3097  013026  022626                    CMP     (SP)+,(SP)+
3098  013030  000733                    BR      SOPRATR
3099  013032  016602  000004    CATNOT: MOV     4(SP),R2
3100  013036  020227  177777            CMP     R2,#-1
3101  013042  001004                    BNE     CATLONG
3102  013044  012600                    MOV     (SP)+,R0
3103  013046  005726                    TST     (SP)+
```

237

```
3104  013050  010016                          MOV     R0,(SP)
3105  013052  000444                          BR      CATCOM
3106  013054  005000                  CATLONG:CLR     R0
3107  013056  151220                          BISB    (R2)+,R0
3108  013060  011603                          MOV     (SP),R3
3109  013062  020604                          CMP     SP,R4
3110  013064  103670                          BLO     ERRD02
3111  013066  005046                          CLR     -(SP)
3112  013070  111316                          MOVB    (R3),(SP)
3113  013072  060016                          ADD     R0,(SP)
3114  013074  021627  000377                  CMP     (SP),#377
3115  013100  101042                          BHI     ERRSTR
3116  013102  010602                          MOV     SP,R2
3117  013104  062702  000006                  ADD     #6,R2
3118  013110  004767  004136                  JSR     PC,MAKESTR
3119  013114  012603                          MOV     (SP)+,R3
3120  013116  016602  000004                  MOV     4(SP),R2
3121  013122  005000                          CLR     R0
3122  013124  151200                          BISB    (R2),R0
3123  013126  010366  000004                  MOV     R3,4(SP)
3124  013132  060003                          ADD     R0,R3
3125  013134  062703  000003                  ADD     #3,R3
3126  013140  012602                          MOV     (SP)+,R2
3127  013142  005726                          TST     (SP)+
3128  013144  005000                          CLR     R0
3129  013146  151200                          BISB    (R2),R0
3130  013150  062702  000003                  ADD     #3,R2
3131  013154  112223                          MOVB    (R2)+,(R3)+
3132  013156  005300                          DEC     R0
3133  013160  003375                          BGT     .-4
3134                            STPRO:
3135  013162  011600                  QUOTBUM:MOV     (SP),R0
3136  013164  010602                  CATCOM: MOV     SP,R2
3137  013166  062700  000003                  ADD     #3,R0
3138  013172  110240                          MOVB    R2,-(R0)
3139  013174  000302                          SWAB    R2
3140  013176  110240                          MOVB    R2,-(R0)
3141  013200  000647                          BR      SOPRATR
3142                                          .ENDC
3143
3144  013202  000167  176564          ERRPD2: JMP     ERRPDL
3145
3146                                          .IFNDF  $NOSTR
3147  013206  000004                  ERRSTR: IOT
3148                                          .IFNDF  $LONGER
3149  013210  052123     114                  .ASCII  \$TL\
3150                                          .ENDC
3151                                          .IFDF   $LONGER
3152                                          .ASCII  /STRING TOO LONG/
3153                                          .ENDC
3154  013213     000                          .BYTE 0
3155                                          .EVEN
3156                                          .ENDC
3157
3158                            EREXP:
```

238


```
3159                            ERSTAR:
3160                            ERADD:
3161  013214  000004                  ERROVR: IOT
3162                                          .IFNDF  $LONGER
3163  013216  053117     109                  .ASCII  /OVF/
3164                                          .ENDC
3165                                          .IFDF   $LONGER
3166                                          .ASCII  /OVERFLOW/
3167                                          .ENDC
3168  013221     000                          .BYTE 0
3169                                          .EVEN
3170
3171  013222  012746  000000G           SINFN:  MOV     #SIN,-(SP)
3172  013226  000434                          BR      FUNCT
3173  013230  012746  000000G           COSFN:  MOV     #COS,-(SP)
3174  013234  000431                          BR      FUNCT
3175  013236  012746  000000G           SQRFN:  MOV     #SQRT,-(SP)
3176  013242  012767  010312  000000G           MOV     #ERSQR,SERVEC
3177  013250  000423                          BR      FUNCT
3178  013252  012746  000000G           ATNFN:  MOV     #ATAN,-(SP)
3179  013256  000420                          BR      FUNCT
3180  013260  012746  000000G           EXPFN:  MOV     #EXP,-(SP)
3181  013264  012767  013214  000000G           MOV     #EREXP,SERVEC
3182  013272  000412                          BR      FUNCT
3183  013274  012746  000000G           LOGFN:  MOV     #ALOG,-(SP)
3184  013300  012767  010312  000000G           MOV     #ERLOG,SERVEC
3185  013306  000404                          BR      FUNCT
3186
3187                                          .IFNDF  $NOSTR
3188  013310  000167  177476          ERRMX9: JMP     ERRMIX
3189                                          .ENDC
3190
3191  013314  000167  177662          ERRD11: JMP     ERRPDZ
3192  013320  004767  176056          FUNCT:  JSR     PC,EVAL
3193
3194                                          .IFNDF  $NOSTR
3195  013324  103002                          BCC     FUNCTJ
3196  013326  000167  174700                  JMP     ERRARG
3197                            FUNCTJ:
3198                                          .ENDC
3199
3200  013332  012603                          MOV     (SP)+,R3
3201  013334  001002                          BNE     .+6
3202  013336  000167  001156                  JMP     ERRUFN
3203  013342  122127  000237                  CMPB    (R1)+,#,RPAR
3204  013346  001050                          BNE     ERRSX9
3205  013350  004467  001540                  JSR     R4,FPPSAV
3206  013354  012538                          .WORD   PUSHF           ;PUSH FAC, FLOAT IF NECESSARY
3207  013356  013364                          .WORD   FUNOK           ;GO DO FUNCTION
3208  013360  015066                          .WORD   FPPRES
3209  013362  000440                          BR      OPRFN
3210  013364  010681          FUNOK:  MOV     SP,R1           ;SAVE ADDRESS OF PUSHED ARG
3211  013366  010446                          MOV     R4,-(SP)        ;SAVE R4
3212  013370  012746  013416                  MOV     #FUNRET,-(SP)   ;DUMMY RETURN ADDRESS
3213  013374  012746  000137                  MOV     #137,-(SP)      ;DUMMY JUMP INSTR,
```

239

```
 3214  013400  010146                          MOV     R1,-(SP)        ;ADDR OF ARGUMENT
 3215  013402  012746  000401                  MOV     #401,-(SP)      ;DUMMY BR
 3216  013406  010500                          MOV     R5,R0           ;SAVE R5 VALUE
 3217  013410  010605                          MOV     SP,R5           ;RETURN ADDRESS (TO DUMMY PROG IN STK)
 3218  013412  004070  000052                  JSR     R0,@RJSAVE(R0)
 3219  013416  062706  000010          FUNRET: ADD     #8.,SP          ;POP DUMMY PROGRAM
 3220  013422  012604                          MOV     (SP)+,R4        ;RESTORE R4
 3221  013424  022626                          CMP     (SP)+,(SP)+     ;REMOVE OLD FAC FROM STACK
 3222  013426  010065  000040                  MOV     R0,FAC1(R5)     ;STORE RESULT IN FAC
 3223  013432  010165  000042                  MOV     R1,FAC2(R5)
 3224  013436  000134                          JMP     @(R4)+          ;AND RETURN
 3225
 3226                                          .IFNDF  $NOSTR
 3227  013440  000167  177346          ERRMX8: JMP     ERRMIX
 3228                                          .ENDC
 3229
 3230  013444  004767  175732          INTFN:  JSR     PC,EVAL
 3231
 3232                                          .IFNDF  $NOSTR
 3233  013450  103773                          BCS     ERRMX8
 3234                                          .ENDC
 3235
 3236  013452  122127  000237                  CMPB    (R1)+,#,RPAR
 3237  013456  001004                          BNE     ERRSX9
 3238  013460  004767  002110                  JSR     PC,INT
 3239  013464  000167  176476          OPRFN:  JMP     OPRATOR
 3240  013470  000167  177050          ERRSX9: JMP     ERRSX5
 3241
 3242
 3243
 3244                                  ;       IF A IS NOT 0,FLOAT IT
 3245  013474  005716                  UPARRO: TST     (SP)            ;IS UPPER A 0
 3246  013476  001010                          BNE     UA1
 3247  013500  005766  000002                  TST     2(SP)           ;IS LOWER A 0
 3248  013504  001405                          BEQ     UA1
 3249  013506  005726                          TST     (SP)+
 3250  013510  004467  001400                  JSR     R4,FPPSAV
 3251  013514  0000006                         .WORD   $IR
 3252  013516  015066                          .WORD   FPPRES
 3253
 3254  013520  012767  014064  000000G UA1:    MOV     #ERREXP,SERVEC
 3255  013526  005765  000040                  TST     FAC1(R5)        ;IS UPPER B = 0
 3256  013532  001036                          BNE     UA10
 3257  013534  005765  000042                  TST     FAC2(R5)        ;IS LOWER B = 0
 3258  013540  001006                          BNE     UA5
 3259  013542  012765  000001  000042          MOV     #1,FAC2(R5)     ;SET RESULT 1
 3260  013550  022626                  UA4:    CMP     (SP)+,(SP)+
 3261  013552  000167  176410                  JMP     OPRATOR
 3262                                  ; B IS INTEGER
 3263  013556  005716                  UA5:    TST     (SP)            ;IS A = 0
 3264  013560  001077                          BNE     UA17
 3265  013562  005765  000042                  TST     FAC2(R5)        ;IS LOWER B > 0
 3266  013566  100013                          BPL     UA9             ;NO, ERROR MESSAGE
 3267  013570  000004                  UA8:    IOT
 3268                                          .IFNDF  $LONGER
```

240


```
 3269  013572  053104      000                 .ASCII  'DV0'
 3270                                          .ENDC
 3271                                          .IFDF   $LONGER
 3272                                          .ASCII  'DIVISION BY 0'
 3273                                          .ENDC
 3274  013575      000                         .BYTE   0
 3275                                          .EVEN
 3276  013576  020027  004000          ERRDIV: CMP     R0,#4000
 3277  013602  103372                          BHIS    UA8
 3278  013604  000167  177404                  JMP     ERSTAK
 3279  013610  103367                          BHIS    UA8             ;DIV BY 0
 3280  013612  000167  177376                  JMP     ERROVR          ;OVERFLOW
 3281  013616  005065  000040          UA9:    CLR     FAC1(R5)        ;SET RESULT 0
 3282  013622  005065  000042                  CLR     FAC2(R5)
 3283  013626  000750                          BR      UA4
 3284
 3285                                  ;FIX B IF INTEGER < 256
 3286  013630  005765  000042          UA10:   TST     FAC2(R5)
 3287  013634  001036                          BNE     UA10,5
 3288  013636  016500  000040                  MOV     FAC1(R5),R0
 3289  013642  010002                          MOV     R0,R2
 3290  013644  042700  100177                  BIC     #100177,R0      ;EXTRACT EXPONENT
 3291  013650  020027  040200                  CMP     R0,#40200       ;CHECK TOO SMALL
 3292  013654  103426                          BLO     UA10,5
 3293  013656  042702  177600                  BIC     #177600,R2
 3294  013662  052702  000200                  BIS     #200,R2         ;R2 IS MANTISSA
 3295  013666  020027  042000          UA10A:  CMP     R0,#42000
 3296  013672  001406                          BEQ     UA10B
 3297  013674  101016                          BHI     UA10,5
 3298  013676  006202                          ASR     R2
 3299  013700  103414                          BCS     UA10,5          ;IF BIT CLEARED,NO GOOD
 3300  013702  062700  000200                  ADD     #200,R0         ;UPDATE EXPONENT
 3301  013706  000767                          BR      UA10A
 3302  013710  005765  000040          UA10B:  TST     FAC1(R5)
 3303  013714  100001                          BPL     UA10C
 3304  013716  005402                          NEG     R2
 3305  013720  005065  000040          UA10C:  CLR     FAC1(R5)
 3306  013724  010265  000042                  MOV     R2,FAC2(R5)
 3307  013730  000712                          BR      UA5
 3308                                  ;B IS REAL
 3309  013732  005716                  UA10,5: TST     (SP)            ;IS A = 0
 3310  013734  001405                          BEQ     UA12
 3311  013736  004467  001152                  JSR     R4,FPPSAV
 3312  013742  016046                          .WORD   PUSH
 3313  013744  014104  014146                  .WORD   UAJMP,UA23
 3314
 3315  013750  005765  000040          UA12:   TST     FAC1(R5)        ;IS UPPER B> 0
 3316  013754  100320                          BPL     UA9             ;YES
 3317  013756  000704                          BR      UA8             ;NO, ERROR
 3318
 3319                                  ;B INTEGER, A REAL
 3320  013760  016500  000042          UA17:   MOV     FAC2(R5),R0
 3321  013764  010002                          MOV     R0,R2           ;SAVE FOR SIGN TEST
 3322  013766  002001                          BGE     UA17A
 3323  013770  005400                          NEG     R0
```

241

```
 3324  013772  320027  000256          UAI7A:  CMP     R0,#256         ;R0 IS ABS(B) IF TOO BIG, FLOAT
 3325  013776  103057                          BHIS    UA20
 3326  014000  004467  001110                  JSR     R4,FPPSAV       ;GO INTO POLISH MODE
 3327  014004  016060                          ,WORD   POP             ;CURRENT PRODUCT IS A
 3328  014006  016072                          ,WORD   PUSH1           ;CURRENT ANSWER IS 1 (ON STK)
 3329  014010  314072  014020          UA17B:  ,WORD   UAASR,UA17D     ;SHIFT B, BRANCH IF BIT IS 0
 3330  014014  016046                          ,WORD   PUSH            ;GET CURRENT PRODUCT ONT OSTACK
 3331  014016  000000G                         ,WORD   $MLR            ;MULTIPLY INTO CURRENT ANSWER
 3332  014020  014110  014040          UA17D:  ,WORD   UATST0,UA19F    ;TEST IF DONE, BRANCH IF SO
 3333  014024  016046  016046                  ,WORD   PUSH,PUSH       ;TWO COPIES OF CURRENT PRODUCT
 3334  014030  000000G                         ,WORD   $MLR            ;SQUARE IT
 3335  014032  016060                          ,WORD   POP             ;AND PUT RESULT BACK
 3336  014034  014104  014010                  ,WORD   UAJMP,UA17B     ;POLISH JUMP TO 17B
 3337  014040  016060                  UA17F:  ,WORD   POP             ;STORE ANSWER INTO FAC
 3338  014042  014120  014056                  ,WORD   UATST2,UA19     ;TEST SIGN B, IF + JUMP
 3339  014046  016072                          ,WORD   PUSH1           ;PUSH 1 ONTO STACK
 3340  014050  016046                          ,WORD   PUSH
 3341  014052  000000G                         ,WORD   $DVR            ;ANS = 1/ANS IF B NEG
 3342  014054  016060                          ,WORD   POP     ;POP ANSWER TO FAC
 3343  014056  015066                  UA19:   ,WORD   FPPRES
 3344  014060  000167  176102                  JMP     OPRATOR         ;AND GET OUT
 3345
 3346                                  ERREXP:
 3347  014064  000004                  UA21:   IOT
 3348                                          ,IFNDF  $LONGER
 3349  014066  042536     122                  ,ASCII  \?ER\
 3350                                          ,ENDC
 3351                                          ,IFDF   $LONGER
 3352                                          ,ASCII  /? ERROR/
 3353                                          ,ENDC
 3354  014071     000                          ,BYTE   0
 3355                                          ,EVEN
 3356
 3357  014072  006265  000044          UAASR:  ASR     R0SAVE(R5)
 3358  014076  103002                          BCC     UAJMP
 3359  014100  005724                  UANJMP: TST     (R4)+           ;SKIP OVER JUMP ADDRESS
 3360  014102  000134                          JMP     @(R4)+
 3361  014104  011404                  UAJMP:  MOV     @R4,R4          ;POLISH MODE JUMP
 3362  014106  000134                          JMP     @(R4)+
 3363  014110  005765  000044          UATST0: TST     R0SAVE  (R5)
 3364  014114  001773                          BEQ     UAJMP
 3365  014116  000770                          BR      UANJMP
 3366  014120  005765  000050          UATST2: TST     R2SAVE(R5)      ;TEST SIGN OF B
 3367  014124  100367                          BPL     UAJMP           ;POLISH JUMP IF B +
 3368  014126  000764                          BR      UANJMP
 3369  014130  005716                  UATSTA: TST     @SP
 3370  014132  100754                          BMI     UA21
 3371  014134  000134                          JMP     @(R4)+
 3372
 3373  014136  010246                  UA20:   MOV     R2,-(SP)        ;PUSH INT B ON STACK
 3374  014140  004467  000750                  JSR     R4,FPPSAV                                       242
 3375  014144  000000G                         ,WORD   $IR             ;FLOAT B
 3376                                  UA23:
 3377  014146  014170                          ,WORD   REVRSE          ;REVERSE TOP TWO ON STK
 3378  014150  014130                          ,WORD   UATSTA          ;TEST TOP STK, BR TO ERR IF -
```



```
 3379  014152  014212                          ,WORD   CALOG           ;CALL ALOG FUNCTION ON A, RESULT TO FAC
 3380  014154  016046                          ,WORD   PUSH            ;PUSH LOG(A) ONTO STACK
 3381  014156  000000G                         ,WORD   $MLR            ;CALC B*LOG(A)
 3382  014160  014224                          ,WORD   CEXP            ;CALC EXP(B*LOG(A)), RESULT TO FAC
 3383  014162  015066                          ,WORD   FPPRES
 3384  014164  000167  175776                  JMP     OPRATOR
 3385  014170  012600                  REVRSE: MOV     (SP)+,R0        ;ROUTINE TO REVERSE TOP 2
 3386  014172  012601                          MOV     (SP)+,R1        ;FLTPT NUMBERS ON STACK
 3387  014174  012602                          MOV     (SP)+,R2
 3388  014176  012603                          MOV     (SP)+,R3
 3389  014200  010146                          MOV     R1,-(SP)
 3390  014202  010046                          MOV     R0,-(SP)
 3391  014204  010346                          MOV     R3,-(SP)
 3392  014206  010246                          MOV     R2,-(SP)
 3393  014210  000134                          JMP     @(R4)+
 3394  014212  012765  000000G 000052  CALOG:  MOV     #ALOG,R3SAVE(R5)
 3395  014220  000167  177140                  JMP     FUNOK
 3396  014224  012765  000000G 000052  CEXP:   MOV     #EXP,R3SAVE(R5)
 3397  014232  000167  177126                  JMP     FUNOK
 3398  014236  000167  175530          ERRPD4: JMP     ERRPDL
 3399
 3400
 3401  014242  000167  176544          ERRMX1: JMP     ERRMIX
 3402                                          ,ENDC
 3403
 3404  014246  000167  176272          ERRSXA: JMP     ERRSX5
 3405  014252  112102                  FNFN:   MOVB    (R1)+,R2
 3406  014254  122127  000255                  CMPB    (R1)+,#,LPAR
 3407  014260  001372                          BNE     ERRSXA
 3408  014262  066502  000004                  ADD     PDL(R5),R2
 3409  014266  020604                          CMP     SP,R4
 3410  014270  103762                          BLO     ERRPD4
 3411  014272  011200                          MOV     (R2),R0         ;R0 NOW CONTAINS THE DEF POINTER,
 3412  014274  001511                          BEQ     ERRUFN
 3413  014276  010046                  FNSWAP: MOV     R0,-(SP)
 3414  014300  004767  175076                  JSR     PC,EVAL
 3415
 3416
 3417  014304  103426                          BCS     FNSTR
 3418                                          ,ENDC
 3419
 3420  014306  012600                          MOV     (SP)+,R0                ;RESTORE THE DEF POINTER,
 3421  014310  112002                          MOVB    (R0)+,R2                ;GET THE VARIABLE FROM THE DEF,
 3422  014312  100500                          BMI     ERRAG1
 3423  014314  000302                          SWAB    R2
 3424  014316  152002                          BISB    (R0)+,R2
 3425  014320  061502                          ADD     (R5),R2
 3426
 3427
 3428  014322  021227  177777                  CMP     (R2),#,SVAR     ;IF THE DEF VARIABLE IS STRING AND
 3429  014326  001745                          BEQ     ERRMX1          ;THE VALUE ISNT THEN ITS AN ERROR,      243
 3430                                          ,ENDC
 3431
 3432  014330  022227  177775                  CMP     (R2)+,#,SCALAR
 3433  014334  001401                          BEQ     .+4
```

```
3434  014336  011202                        MOV     (R2),R2
3435  014340  020604                        CMP     SP,R4
3436  014342  103735                        BLO     ERRPD4
3437  014344  011246                        MOV     (R2),-(SP)      ;STACK THE OLD VALUE AND REPLACE IT
3438  014346  016522  000040                MOV     FAC1(R5),(R2)+  ;WITH THE NEW VALUE
3439  014352  011246                        MOV     (R2),-(SP)
3440  014354  016512  000042                MOV     FAC2(R5),(R2)
3441  014360  000458                        BR      FNREPL
3442
3443                                        ;IFNDF  $NOSTR
3444  014362  016600  000002        FNSTR:  MOV     2(SP),R0                ;RESTORE THE DEF POINTER,
3445  014366  112002                        MOVB    (R0)+,R2                ;GET THE VARIABLE FROM THE DEF,
3446  014370  100451                        BMI     ERRAG1
3447  014372  000302                        SWAB    R2
3448  014374  152002                        BISB    (R0)+,R2
3449  014376  061502                        ADD     (R5),R2
3450  014400  022227  177777                CMP     (R2)+,#,SVAR    ;IF THE DEF VARIABLE IS NUMERIC AND
3451  014404  001316                        BNE     ERRMX1          ;THE VALUE ISNT THEN ITS AN ERROR,
3452  014406  026227  000002  177777        CMP     2(R2),#-1       ;IF THE VAR ISNT A STRING SCALAR
3453  014414  001401                        BEQ     .+4
3454  014416  011202                        MOV     (R2),R2         ;THEN DO THIS,
3455  014420  011203                        MOV     (R2),R3         ;R3 NOW CONTAINS THE OLD VALUE;
3456  014422  012612                        MOV     (SP)+,(R2)      ;@(R2) NOW CONTAINS THE NEW VALUE,
3457  014424  010316                        MOV     R3,(SP)         ;@(SP) NOW CONTAINS THE OLD VALUE,
3458  014426  010246                        MOV     R2,-(SP)                ;TOP(STK) IS A TEMP FOR VAR ADDRESS,
3459  014430  011202                        MOV     (R2),R2         ;R2 NOW CONTAINS THE NEW VALUE;
3460  014432  005202                        INC     R2
3461  014434  001412                        BEQ     FNNEWNL         ;BRANCH IF THE NEW VALUE IS NULL,
3462  014436  005203                        INC     R3
3463  014440  001004                        BNE     FNOLNN          ;BRANCH IF THE OLD VALUE IS NONNULL,
3464  014442  116622  000001                MOVB    1(SP),(R2)+     ;OLD VALUE IS NULL, NEW IS NOT; SO:
3465  014446  112612                        MOVB    (SP)+,(R2)      ;MAKE AN ABS BACKPTR TO THE VAR,
3466  014450  000414                        BR      FNREPL          ;(THE NEW VALUE IS NOW PROTECTED)
3467  014452  112322        FNOLNN: MOVB    (R3)+,(R2)+     ;OLD AND NEW ARE BOTH NONNULL SO:
3468  014454  111312                        MOVB    (R3),(R2)       ;PUT OLD BACKPTR INTO THE NEW VALUE,
3469  014456  162703  000002                SUB     #2,R3
3470  014462  005726        FNNEWNL:TST     (SP)+           ;GET RID OF THE TEMP,
3471  014464  010602                        MOV     SP,R2
3472  014466  005203                        INC     R3
3473  014470  001404                        BEQ     FNREPL          ;     (CHECK NULL STRING!)
3474  014472  000302                        SWAB    R2              ;SET THE BACKPTR OF THE OLD VALUE
3475  014474  110223                        MOVB    R2,(R3)+        ;TO ITS CURRENT STACK ADDRESS,
3476  014476  000302                        SWAB    R2
3477  014500  110213                        MOVB    R2,(R3)
3478                                        ;ENDC
3479
3480  014502  121027  000243        FNREPL: CMPB    (R0),#,COMMA
3481  014506  001007                        BNE     FNNOCOM
3482  014510  122021                        CMPB    (R0)+,(R1)+
3483  014512  001671                        BEQ     FNSWAP
3484  014514  000167  173572        ERRAG1: JMP     ERRARG
3485  014520  000004        ERRUFN: IOT
3486                                        ;IFNDF  $LONGER
3487  014522  043125    116                 ;ASCII  'UFN'
3488                                        ;ENDC
```

244


```
3489                                        ;IFDF   $LONGER
3490                                        ;ASCII  'UNDEFINED FUNCTION'
3491                                        ;ENDC
3492  014525      000                       ;BYTE 0
3493                                        ;EVEN
3494  014526  121027  000237        FNNOCOM:CMPB    (R0),#,RPAR
3495  014532  001370                        BNE     ERRAG1
3496  014534  010046                        MOV     R0,-(SP)
3497  014536  122021                        CMPB    (R0)+,(R1)+
3498  014540  001365                        BNE     ERRAG1
3499  014542  122027  000254                CMPB    (R0)+,#,EQ
3500  014546  001362                        BNE     ERRAG1
3501  014550  010146                        MOV     R1,-(SP)
3502  014552  010001                        MOV     R0,R1
3503  014554  004767  174622                JSR     PC,EVAL
3504  014560  103402                        BCS     .+6
3505  014562  005003                        CLR     R3
3506  014564  000401                        BR      .+4
3507  014566  012603                        MOV     (SP)+,R3
3508  014570  121127  000201                CMPB    (R1),#,EOL
3509  014574  001224                        BNE     ERR$XA
3510  014576  012601                        MOV     (SP)+,R1
3511  014600  012600                        MOV     (SP)+,R0
3512  014602  114046        FNRLUP: MOVB    -(R0),-(SP)
3513  014604  114066  000001                MOVB    -(R0),1(SP)
3514  014610  012602                        MOV     (SP)+,R2
3515  014612  061502                        ADD     (R5),R2
3516
3517                                        ;IFNDF  $NOSTR
3518  014614  021227  177777                CMP     (R2),#,SVAR
3519  014620  001416                        BEQ     FNRSTR
3520                                        ;ENDC
3521
3522  014622  022227  177775                CMP     (R2)+,#,SCALAR
3523  014626  001401                        BEQ     .+4
3524  014630  011202                        MOV     (R2),R2
3525  014632  012662  000002                MOV     (SP)+,2(R2)
3526  014636  012612                        MOV     (SP)+,(R2)
3527  014640  000426                        BR      FNCHK
3528  014642  012612        FNRSSC: MOV     (SP)+,(R2)
3529  014644  010046                        MOV     R0,-(SP)
3530  014646  011200                        MOV     (R2),R0
3531  014650  161502                        SUB     (R5),R2
3532  014652  005202                        INC     R2
3533
3534                                        ;IFNDF  $NOSTR
3535  014654  000411                        BR      FNRCOM
3536  014656  005722        FNRSTR: TST     (R2)+
3537  014660  026227  000002  177777        CMP     2(R2),#-1
3538  014666  001765                        BEQ     FNRSSC
3539  014670  011202                        MOV     (R2),R2
3540  014672  012612                        MOV     (SP)+,(R2)
3541  014674  010046                        MOV     R0,-(SP)
3542  014676  011200                        MOV     (R2),R0
3543                                        ;ENDC
```

245

```
 3544
 3545  014700  005200                FNRCOM: INC     R0
 3546  014702  001404                        BEQ     FNRSNUL
 3547  014704  000302                        SWAB    R2
 3548  014706  110220                        MOVB    R2,(R0)+
 3549  014710  000302                        SWAB    R2
 3550  014712  110220                        MOVB    R2,(R0)+
 3551  014714  012600                FNRSNUL:MOV     (SP)+,R0
 3552  014716  124027  000255        FNCHK:  CMPB    -(R0),#,LPAR
 3553  014722  001327                        BNE     FNRLUP
 3554
 3555                                        ;;FNDF  $NOSTK
 3556  014724  005703                        TST     R3
 3557  014726  001410                        BEQ     FNNUMER
 3558  014730  010346                        MOV     R3,-(SP)
 3559  014732  020327  177777                CMP     R3,#177777      ;CHECK NULL STRING
 3560  014736  001002                        BNE     .+6
 3561  014740  000167  175754                JMP     $OPRATR
 3562  014744  000167  176212                JMP     $IPRO
 3563                                        ;ENDC
 3564
 3565  014750  000167  175212        FNNUMER:JMP     OPRATOR
 3566                                        ;EOT
 3567
```

246


```
 3568                                ;                               SOURCE FILE #6
 3569                                ;-------------------------------------------------------
 3570                                ;  SUBROUTINE 'FIXUP'   CALLED BY JSR PC
 3571                                ;                      MATCHES TYPES OF 0(SP),2(SP) AND FAC1(R5),FAC2(R5)
 3572                                ;                      R0,R1  UNUSED
 3573                                ;                      R2  DESTROYED
 3574                                ;                      R3,R4  UNUSED
 3575                                ;                      R5  MUST POINT TO USER AREA
 3576                                ;                      SP  GOES NO DEEPER AFTER JSR
 3577  014754  012602                FIXUP:  MOV     (SP)+,R2
 3578  014756  005716                        TST     (SP)
 3579  014760  001413                        BEQ     FIXTST
 3580  014762  005765  000040                TST     FAC1(R5)
 3581  014766  001021                        BNE     FIXRET          ;COND CODES=NONZERO MEANS FLOATING,
 3582  014770  016546  000042                MOV     FAC2(R5),-(SP)          !****
 3583  014774  004467  000114                JSR     R4,FPPSAV
 3584  015000  000000G                       ,WORD   $IR
 3585  015002  016060                        ,WORD   POP
 3586  015004  015066                        ,WORD   FPPRES
 3587  015006  000410                        BR      FIXCLC
 3588  015010  005765  000040        FIXTST: TST     FAC1(R5)
 3589  015014  001406                        BEQ     FIXRET          ;COND CODES=0 MEANS INTEGER,
 3590  015016  005726                        TST     (SP)+                   !****
 3591  015020  004467  000070                JSR     R4,FPPSAV
 3592  015024  000000G                       ,WORD   $IR
 3593  015026  015066                        ,WORD   FPPRES
 3594  015030  000244                FIXCLC: CLC                     ;COND CODES=NONZERO MEANS FLOATING,
 3595  015032  000112                FIXRET: JMP     (R2)
 3596
```

247

```
 3597                                   ;*****************************************
 3598                                   ;  SUBROUTINE 'FLINE'    CALLED BY JSR PC
 3599                                   ;                        FINDS THE ADDRESS OF THE LINE NO
 3600                                   ;                        REFERENCED BY (R1) IN R2,
 3601                                   ;                        IF THE SYMBOL TABLE REFERENCE IS NOT A LINE NO,
 3602                                   ;                        SETS CARRY, OTHERWISE, THE CARRY IS CLEAR,
 3603  015034  112102                  FLINE:  MOVB    (R1)+,R2
 3604  015036  100407                          BMI     FLE
 3605  015040  000302                          SWAB    R2
 3606  015042  152102                          BISB    (R1)+,R2
 3607  015044  061502                          ADD     (R5),R2
 3608  015046  012200                          MOV     (R2)+,R0
 3609  015050  020027  177775                  CMP     R0,#,$CALAR
 3610  015054  103402                          BLO     FLX
 3611  015056  000261                  FLE:    SEC
 3612  015060  000207                          RTS     PC
 3613  015062  000241                  FLX:    CLC
 3614  015064  000207                          RTS     PC
 3615
```

248


```
 3616                                   ;*****************************************************
 3617                                   ;  ROUTINE 'FPPRES'       CALLED IN POLISH MODE
 3618                                   ;                        RESTORES R0 THRU R4 SAVED BY FPPSAV
 3619                                   ;                        AND RETURNS IN NORMAL EXEC MODE
 3620  015066  016500  000044          FPPRES: MOV     R0SAVE(R5),R0
 3621  015072  016501  000046                  MOV     R1SAVE(R5),R1
 3622  015076  016502  000050                  MOV     R2SAVE(R5),R2
 3623  015102  016503  000052                  MOV     R3SAVE(R5),R3
 3624  015106  016546  000054                  MOV     R4SAVE(R5),-(SP)
 3625  015112  000204                          RTS     R4
 3626
 3627
 3628                                   ;*****************************************************
 3629                                   ;  ROUTINE 'FPPSAV'        CALLED BY JSR R4
 3630                                   ;                        SAVES R0 THRU R4, AND RETURNS IN
 3631                                   ;                        POLISH MODE
 3632  015114  010065  000044          FPPSAV: MOV     R0,R0SAVE(R5)
 3633  015120  010165  000046                  MOV     R1,R1SAVE(R5)
 3634  015124  010265  000050                  MOV     R2,R2SAVE(R5)
 3635  015130  010365  000052                  MOV     R3,R3SAVE(R5)
 3636  015134  012665  000054                  MOV     (SP)+,R4SAVE(R5)
 3637  015140  000134                          JMP     @(R4)+          ;ENTER POLISH MODE
 3638
```

249

```
                        ;
                        ; GET1BYT (GET2BYT) - GET BYTE OUT OF RING BUFFER USING
                        ;  GET POINTER 1 (R2),
                        ;
                        ; CALL: #[BUFF, HDR, BASE],R1
                        ;       #[BASE OFFSET OF GET PTR],R3
                        ;       JSR     PC,GET1BYT
                        ;
                        ;  USES R0,R1,R2,R3
                        ;
                        ;  RETURN CHAR, IN R0,
                        ;  'C' BIT IS: CLR IF CHAR' OBTAINED
                        ;              SET IF BUFF' WAS EMPTY (NO CHAR')
                        ;
015142  012703  000004  GET1BYT: MOV     #BGET1,R3
015146  000402                  BR      GETBYT
015150  012703  000006  GET2BYT: MOV     #BGET2,R3
015154  012702  177776  GETBYT:  MOV     #PS,R2          ;STATUS
015160  011246                  MOV     (R2),-(SP)
015162  012712  000340          MOV     #PR7,(R2)       ;HIGHEST PRIORITY
015166  060103                  ADD     R1,R3
015170  021361  000010          CMP     (R3),BPUT(R1)   ;ANYTHING IN BUFF?
015174  001412                  BEQ     BUFEMP
015176  113300                  MOVB    @(R3)+,R0       ;GET BYTE
015200  005243                  INC     -(R3)           ;PT, TO NEXT BYTE
015202  021361  000002          CMP     (R3),BEND(R1)   ;WRAP AROUND?
015206  101402                  BLOS    NOWRP
015210  016113  000000          MOV     BSTRT(R1),(R3)  ;PT, TO NEXT BYTE TO GET
015214  012612          NOWRP:  MOV     (SP)+,(R2)      ;STATUS BACK
015216  000241                  CLC                     ;SIGNAL SUCCESS
015220  000207                  RTS     PC
015222  012612          BUFEMP: MOV     (SP)+,(R2)      ;STATUS BACK
015224  000261                  SEC                     ;SHOW FAILURE
015226  000207                  RTS     PC
```

250


```
                        ;+-----------------------------------------------------------
                        ;
                        ; GETCHAR - RETURN A CHAR, IN R0 OR WAIT UNTIL YOU CAN
                        ;
                        ; CALL: #[BUFF, HDR, BASE],R1
                        ;       JSR     PC,GETCHAR
                        ;
                        ;  USES R3, (R0,R1,R2)
                        ;
                        ;  RETURNS CHAR, IN R0
                        ;  IF PAPER TAPE READER EOT FOUND, EXIT TO 'READY',
                        ;
015230  010246          GETCHAR: MOV     R2,-(SP)
015232  004767  177704  GETCH1: JSR     PC,GET1BYT
015236  103405                  BCS     NOCHAR          ;NOTHING THERE YET
015240  016102  000012          MOV     BFSPEC(R1),R2   ;CHECK IF READER STOPPED
015244  003011                  BGT     XCHAR           ;0 OR - IS OK, BRANCH IF CHAR
015246  012602          GETX:   MOV     (SP)+,R2
015250  000207                  RTS     PC
015252  016100  000012  NOCHAR: MOV     BFSPEC(R1),R0   ;CHECK EOF OR EXTRA CHAR
015256  001423                  BEQ     REENAB
015260  005061  000012          CLR     BFSPEC(R1)
015264  000261                  SEC
015266  000767                  BR      GETX
015270  010046          XCHAR:  MOV     R0,-(SP)        ;PUSH OLD CHAR
015272  010200                  MOV     R2,R0           ;POSITION NEXT
015274  004767  003314          JSR     PC,PUTBYT
015300  103001                  BCC     .+4             ;MAKE SURE IT GOT IN!
015302  000000                  HALT
015304  012600                  MOV     (SP)+,R0        ;GET BACK OLD CHAR
015306  005061  000012          CLR     BFSPEC(R1)
015312  116502  000077          MOVB    IDEV(R5),R2
015316  012772  000101  015360  MOV     #101,@ITAB(R2)  ;RE-ENABLE INPUT DEVICE
015324  000750                  BR      GETX            ; (CARRY IS CLEAR)
                        ;       RE-ENABLE FROM A DEAD START
015326  116502  000077  REENAB: MOVB    IDEV(R5),R2
015332  032772  004000  015360          BIT     #004000,@ITAB(R2)       ;CHECK BUSY BIT
015340  001003                  BNE     GCWAIT
015342  012772  000101  015360          MOV     #101,@ITAB(R2)
015350  000001          GCWAIT: WAIT                    ;WAIT FOR AN INTERRUPT
015352  004767  000524          JSR     PC,IOWAIT       ;WASTE TIME
015356  000725                  BR      GETCH1
                        ;
                        ;       TABLE OF INPUT DEVICE STATUS REGISTERS
015360  177560          ITAB:   .WORD   TKS
015362  177550                  .WORD   PRS
```

251

```
                                        ;*********************************************************
                                        ; 'GETVAR' SUBROUTINE
                                        ;               THIS SUBROUTINE READS A VARIABLE NAME OR
                                        ;               AN ARRAY ELEMENT, AND ADDRESSES THAT
                                        ;               VARIABLE IN CORE, SO THAT A SUBSEQUENT 'STOVAR'
                                        ;               CALL WILL STORE THE FAC IN THE VARIABLE,
                                        ;               DESTROYS FAC1 AND FAC2',
                                        ;               CALLED BY JSR PC,GETVAR
                                        ;                       R1 POINTS TO THE CODE NAMING THE VARIABLE
                                        ;
015364  010265  000022          GETVAR: MOV     R2,VARSAV(R5)
015370  012702  177777                  MOV     #-1,R2
015374  010265  000024                  MOV     R2,SS1SAV(R5)
015400  010265  000026                  MOV     R2,SS2SAV(R5)
015404  121127  000295                  CMPB    (R1),#,LPAR
015410  001043                          BNE     NOSUBS
015412  005201                          INC     R1
015414  004767  173702                  JSR     PC,EVAL

                                        .IFNDF  $NOSTM
015420  103442                          BCS     ERRMX3
                                        .ENDC

015422  004767  000146                  JSR     PC,INT
015426  005765  000040                  TST     FAC1(R5)
015432  001037                          BNE     ERRSS1
015434  016565  000042  000024          MOV     FAC2(R5),SS1SAV(R5)
015442  100433                          BMI     ERRSS1
015444  112102                          MOVB    (R1)+,R2
015446  120227  000237                  CMPB    R2,#,RPAR
015452  001422                          BEQ     NOSUBS
015454  120227  000243                  CMPB    R2,#,COMMA
015460  001020                          BNE     ERRSX3
015462  004767  173714                  JSR     PC,EVAL

                                        .IFNDF  $NOSTM
015466  103417                          BCS     ERRMX3
                                        .ENDC

015470  004767  000100                  JSR     PC,INT
015474  005765  000040                  TST     FAC1(R5)
015500  001014                          BNE     ERRSS1
015502  016565  000042  000026          MOV     FAC2(R5),SS2SAV(R5)
015510  100410                          BMI     ERRSS1
015512  122127  000237                  CMPB    (R1)+,#,RPAR
015516  001001                          BNE     ERRSX3
015520  000207          NOSUBS: RTS     PC
015522  000167  175016  ERRSX3: JMP     ERRSX5

                                        .IFNDF  $NOSTM
015526  000167  175200  ERRMX3: JMP     ERRMIX
                                        .ENDC

015532  000167  174360  ERRSS1: JMP     ERRSS2
```

252


```
                                        ;*********************************************************
                                        ;
                                        ; INITBF = INIT, BUFFER HDRS,
                                        ;
                                        ; CALL: MOV     #[WDR, ADDR,],R2
                                        ;       JSR     PC,INITBF
                                        ;
                                        ;  USES R2,R3
                                        ;
                                        ;  LEAVES R3 POINTING TO WORD AFTER LAST IN BUFF, HDR,
                                        ;
015536  012203          INITBF: MOV     (R2)+,R3        ;BUFF, START ADDR,
015540  005722                  TST     (R2)+           ;PT, TO BGET1
015542  010322                  MOV     R3,(R2)+
015544  005722                  TST     (R2)+           ;PT, TO BPUT
015546  010322                  MOV     R3,(R2)+
015550  005022                  CLR     (R2)+           ;BFSPEC
015552  000207                  RTS     PC


                                        ;*********************************************************
                                        ;
                                        ; INITSCR = SCRATCH USER AREA
                                        ;
                                        ; CALL: JSR     PC,INITSCR
                                        ;
                                        ;  USES R0
                                        ;
015554  016500  000016          INITSCR:MOV     CODE(R5),R0
015560  012720  000225                  MOV     #,EOF,(R0)+
015564  010015                          MOV     R0,(R5)
015566  010065  000014                  MOV     R0,LOFREE(R5)
015572  000207                          RTS     PC
```

253

```
                                        ;-----------------------------------------------------
                                        ;  SUBROUTINE 'INT'     CALLED BY JSR PC
                                        ;                       COMPUTES INT(FAC), CONVERTING THE RESULT
                                        ;                       TO A 1-WORD INTEGER, IF POSSIBLE
015574  016500  000040          INT:    MOV     FAC1(R5),R0     ;GET HIGH ORDER WORD IN R0
015600  001452                          BEQ     INTRTS          ;ALREADY AN INTEGER
015602  004567  003264                  JSR     R5,SAVREG       ;SAVE REGISTERS
015606  010001                          MOV     R0,R1
015610  042700  100177                  BIC     #100177,R0      ;EXTRACT EXPONENT
015614  020027  040200                  CMP     R0,#40200       ;CHECK ABS VALUE < 1
015620  103473                          BLO     INT7
015622  020027  046000                  CMP     R0,#46000       ;CHECK INTEGER BY TRUNCATION
015626  101037                          BHI     INTRTS          ;YES, RETURN
015630  005002                          CLR     R2
015632  162700  043600                  SUB     #43600,R0       ;CHECK MORE THAN 15 BITS INTEGER
015636  003034                          BGT     INT5            ;YES, PRODUCE FLT INT ANSWER
015640  000301                          SWAB    R1
015642  105001                          CLRB    R1
015644  156501  000043                  BISB    FAC2+1(R5),R1   ;R1 IS 16 BITS OF MANTISSA
015650  052701  100000                  BIS     #100000,R1      ;SET HIDDEN BIT
015654  000241                          CLC
015656  006001                          ROR     R1              ;GET 15 BITS OF ABSOLUTE VALUE
015660  005700                          TST     R0              ;TEST ALREADY INTEGER
015662  002005                          BGE     INT1A           ;YES
015664  006201                  INT1:   ASR     R1              ;SHIFT MANTISSA 1 RIGHT
015666  005502                          ADC     R2              ;ADD CARRY BIT TO R2
015670  062700  000200                  ADD     #200,R0         ;INCREMENT EXPONENT
015674  002773                          BLT     INT1            ;LOOP UNTIL DONE
015676  005765  000040          INT1A:  TST     FAC1(R5)        ;ORIGINAL NUMBER NEG
015702  100005                          BPL     INT2            ;NO
015704  005702                          TST     R2              ;ANY BITS ZEROED?
015706  001402                          BEQ     INT1B
015710  005201                          INC     R1              ;ADJUST FOR NEGATIVE
015712  102447                          BVS     INT9            ;OVERFLOW
015714  005401                  INT1B:  NEG     R1
015716  010165  000042          INT2:   MOV     R1,FAC2(R5)
015722  005065  000040          INT2A:  CLR     FAC1(R5)
015726  000207                  INTRTS: RTS     PC

015730  020027  002200          INT5:   CMP     R0,#2200        ;CHECK DONE
015734  002005                          BGE     INT6            ;YES
015736  000261                          SEC                     ;SET CARRY BIT
015740  006102                          ROL     R2              ;CREATE PATTERN OF BITS TO CLEAR
015742  062700  000200                  ADD     #200,R0
015746  000770                          BR      INT5
015750  016500  000042          INT6:   MOV     FAC2(R5),R0     ;SAVE OLD FAC2 IN CASE NEG ARG
015754  040265  000042                  BIC     R2,FAC2(R5)     ;CLEAR BITS
015760  005765  000040                  TST     FAC1(R5)        ;CHECK NEG ARG
015764  100360                          BPL     INTRTS          ;NO, DONE
015766  030200                          BIT     R2,R0           ;CHECK EXACT INTEGER ALREADY
015770  001756                          BEQ     INTRTS          ;YES, RETURN
015772  004467  000000G                 JSR     R4,$POLSH       ;NO, MUST SUBTRACT 1
015776  016046                          .WORD   PUSH            ;PUSH FAC
016000  016072                          .WORD   PUSH1           ;PUSH FLOATING 1
016002  000000G                         .WORD   $SBR            ;SUBTRACT
```



```
016004  016060                          .WORD   POP             ;POP ANSWER TO FAC
016006  015726                          .WORD   INTRTS          ;AND RETURN (OUT OF POLISH MODE)
016010  005701                  INT7:   TST     R1              ;ABS(ARG) < 1
016012  100403                          BMI     INT8
016014  005065  000042                  CLR     FAC2(R5)        ;IF + THEN ANS IS 0
016020  000740                          BR      INT2A
016022  012765  177777  000042  INT8:   MOV     #177777,FAC2(R5) ;IF - THEN ANS IS -1
016030  000734                          BR      INT2A
016032  012765  143600  000040  INT9:   MOV     #143600,FAC1(R5)
016040  005065  000042                  CLR     FAC2(R5)
016044  000730                          BR      INTRTS

016046  016546  000042          PUSH:   MOV     FAC2(R5),-(SP)
016052  016546  000040                  MOV     FAC1(R5),-(SP)
016056  000134                          JMP     @(R4)+
016060  012665  000040          POP:    MOV     (SP)+,FAC1(R5)
016064  012665  000042                  MOV     (SP)+,FAC2(R5)
016070  000134                          JMP     @(R4)+
016072  005046                  PUSH1:  CLR     -(SP)
016074  012746  040200                  MOV     #40200,-(SP)
016100  000134                          JMP     @(R4)+





                                        ;-----------------------------------------------------
                                        ;
                                        ; IOWAIT - WASTE TIME TILL NEXT INTERRUPT
                                        ;
                                        ; CALL: JSR     PC,IOWAIT
                                        ;
016102  005265  000070          IOWAIT: INC     RNDCT(R5)
016106  105765  000106                  TSTB    CNCFLG(R5)
016112  001402                          BEQ     WTRET
016114  000167  163024                  JMP     READY
016120  005737  000050          WTRET:  TST     @#50            ;CHECK WAIT LOOP ADDR
016124  001402                          BEQ     .+6
016126  000137  000050                  JMP     @#50            ;SPECIAL WAIT ROUTINE
016132  000207                          RTS     PC
```

```
3908                                    ;-------------------------------------------------------
3909                                    ;
3910                                    ; LINGET - EDIT A LINE INTO LINE BUFFER USING GETCHAR
3911                                    ;
3912                                    ; CALL: USER AREA BASE INTO R5
3913                                    ;       JSR     PC,LINGET
3914                                    ;
3915                                    ;  PRESERVES ALL REGISTERS
3916                                    ;
3917                                    ;  RECOGNIZES '@' AND <RUB OUT> FOR RUB OUT AND <ALT MODE>
3918                                    ;   AND '^U' FOR LINE DELETE; <CR> AS LINE TERMINATOR,
3919                                    ;   THROWS AWAY ALL OTHERS BELOW ASCII 40 AND ABOVE
3920                                    ;   ASCII 137.
3921                                    ;
3922  016134  004567  002732           LINGET: JSR     R5,SAVREG       ;SAVE REGS,
3923  016140  116500  000077           LINA:   MOVB    IDEV(R5),R0     ;CURR. INPUT DEV. CODE
3924  016144  016001  016250                   MOV     TAB3(R0),R1     ;TO SELECT DEV,
3925  016150  005700                           TST     R0
3926  016152  001002                           BNE     LINEIN
3927  016154  060501                           ADD     R5,R1           ;USER BASE
3928  016156  011101                           MOV     (R1),R1         ;HAVE USER BUFF. ADR.
3929  016160  016502  000020           LINEIN: MOV     LINE(R5),R2
3930  016164  020265  000020           LINRUB: CMP     R2,LINE(R5)     ;DON'T RUB OUT TOO MUCH
3931  016170  001401                           BEQ     DOCHAR
3932  016172  005302                           DEC     R2
3933  016174  004767  177030           DOCHAR: JSR     PC,GETCHAR      ;DON'T COME BACK 'TILL YOU GOT ONE
3934  016200  103001                           BCC     GOTONE
3935  016202  000207                           RTS     PC              ;INDICATE INCOMPLETE LINE (CARRY SET)
3936  016204  110012           GOTONE: MOVB    R0,(R2)         ;PUT HER IN
3937  016206  004367  172370                   JSR     R3,CHKCHR       ;RETURN EXEC, ONE OF NEXT FIVE LOCS,
3938  016212  000752                           BR      LINA            ;IF LINE DELETE
3939  016214  000207                           RTS     PC              ;DONE (<CR>) (CARRY CLEAR)
3940  016216  000762                           BR      LINRUB
3941  016220  000765                           BR      DOCHAR          ;MEANINGLESS CHARS,
3942  016222  005202                           INC     R2              ;GOOD CHAR,
3943  016224  020265  000016                   CMP     R2,CODE(R5)     ;CHECK LINE SPACE OVERFLOW
3944  016230  103761                           BLO     DOCHAR
3945  016232  004167  001206                   JSR     R1,MSGERR
3946  016236     015     012                   .BYTE   CR,LF
3947                                          .IFNDF  $LONGER
3948  016240  052114     114                   .ASCII  'LTL'
3949                                          .ENDC
3950                                          .IFDF   $LONGER
3951                                          .ASCII  'LINE TOO LONG'
3952                                          .ENDC
3953  016243     015     012                   .BYTE   CR,LF
3954  016245     000                           .BYTE   0
3955                                          .EVEN
3956  016246  000744                           BR      LINEIN
3957                                    ;
3958                                    TAB3:
3959  016250  000102                           .       KBHD
3960                                          .IFNDF $NOPTP
3961  016252  023714                           .       PRBFHD
3962                                          .ENDC
```

256


```
3963
```

257

```
                                        ;*****************************************************
                                        ;  'LITEVAL' SUBROUTINE
                                        ;               CALLED BY JSR PC,LITEVAL
                                        ;               R1 POINTS TO THE CODE
                                        ;               THIS SUBROUTINE IS CALLED TO EVALUATE A
                                        ;               LITERAL IN THE CODE,
016254  005000                  LITEVAL:CLR     R0
016256  005065  000040                  CLR     FAC1(R5)
016262  005065  000042                  CLR     FAC2(R5)
016266  121327  000232                  CMPB    (R3),#,PLUS
016272  001404                          BEQ     LEVPLUS
016274  121327  000234                  CMPB    (R3),#,MINUS
016300  001002                          BNE     LEVLIT
016302  005100                          COM     R0
016304  005203                  LEVPLUS:INC     R3
016306  122327  000375          LEVLIT: CMPB    (R3)+,#,ILIT1
016312  001416                          BEQ     LEV1LT
016314  124327  000374                  CMPB    -(R3),#,FLIT
016320  001404                          BEQ     LEVFLT
016322  122327  000376                  CMPB    (R3)+,#,ILIT2
016326  001406                          BEQ     LEV2LT
016330  000207                          RTS     PC
016332  005203                  LEVFLT: INC     R3
016334  112365  000041                  MOVB    (R3)+,FAC1+1(R5)
016340  112365  000040                  MOVB    (R3)+,FAC1(R5)
016344  112365  000043          LEV2LT: MOVB    (R3)+,FAC2+1(R5)
016350  112365  000042          LEV1LT: MOVB    (R3)+,FAC2(R5)
016354  005700                          TST     R0
016356  001411                          BEQ     LEVPOS
016360  005765  000040                  TST     FAC1(R5)
016364  001003                          BNE     LEVFNEG
016366  005465  000042                  NEG     FAC2(R5)
016372  000403                          BR      LEVPOS
016374  062765  100000  000040  LEVFNEG:ADD     #100000,FAC1(R5)
016402  062716  000002          LEVPOS: ADD     #2,(SP)
016406  000207                          RTS     PC
```

258


```
                                        ;*****************************************************
                                        ;  SUBROUTINE 'LOCGET'  CALLED BY JSR PC
                                        ;                       GETS LOC OF ELEMENT IN ARRAY (CHECKS SUBSCRIPTS)
                                        ;                       R0  MUST CONTAIN SS1     GETS DESTROYED
                                        ;                       R1  UNUSED
                                        ;                       R2  MUST POINT TO VAR    GETS SET TO RESULT
                                        ;                       R3  MUST CONTAIN SS2     GETS DESTROYED
                                        ;                       R4  UNUSED
                                        ;                       R5  MUST POINT TO USER AREA
                                        ;                       SP  GOES ?? DEEPER AFTER JSR
016410  021227  177775          LOCGET: CMP     (R2),#,SCALAR
016414  001403                          BEQ     LOCALOC
016416  005762  000004                  TST     4(R2)
016422  100017                          BPL     LOCSS1
016424  010246                  LOCALOC:MOV     R2,-(SP)
016426  010446                          MOV     R4,-(SP)
016430  010346                          MOV     R3,-(SP)
                                        
                                        ,IFNDF  $NOSTR
016432  004767  172550                  JSR     PC,UNPACK
                                        ,ENDC
                                        
016436  012703  000012                  MOV     #12,R3
016442  011604                          MOV     (SP),R4
016444  100401                          BMI     .+4
016446  010304                          MOV     R3,R4
016450  004767  171334                  JSR     PC,ALLOC
016454  012603                          MOV     (SP)+,R3
016456  012604                          MOV     (SP)+,R4
016460  012602                          MOV     (SP)+,R2
016462  020062  000004          LOCSS1: CMP     R0,4(R2)
016466  101043                          BHI     ERRSS3
016470  005703                          TST     R3
016472  100427                          BMI     LOCNO2
016474  005762  000006                  TST     6(R2)
016500  100436                          BMI     ERRSS3
016502  020362  000006                  CMP     R3,6(R2)
016506  101033                          BHI     ERRSS3
016510  016246  000004                  MOV     4(R2),-(SP)             ;****
016514  005216                          INC     (SP)                    ;*  *
016516  010346                          MOV     R3,-(SP)                ;*  *
016520  012746  000020                  MOV     #20,-(SP)               ;*  *
016524  006303                  LOCLOOP:ASL     R3                      ;*  *
016526  006366  000002                  ASL     2(SP)                   ;*  *  MYULTIPLY SS2 BY SS1MAX
016532  103002                          BCC     .+6                     ;*  *  RESULT IN R3
016534  066603  000004                  ADD     4(SP),R3                ;*  *
016540  005316                          DEC     (SP)                    ;*  *
016542  003370                          BGT     LOCLOOP                 ;*  *
016544  022626                          CMP     (SP)+,(SP)+             ;*  *
016546  005726                          TST     (SP)+                   ;****
016550  060300                          ADD     R3,R0
016552  005200                  LOCNO2: INC     R0
016554  006300                          ASL     R0
                                        
                                        ,IFNDF  $NOSTR
```

259

```
4056  016556  021227  177777          CMP     (R2),#,SVAR
4057  016562  001401                  BEQ     ,+4
4058                                  ,ENDC
4059
4060  016564  006320                  ASL     R0
4061  016566  016202  000002          MOV     2(R2),R2
4062  016572  060002                  ADD     R0,R2
4063  016574  000207                  RTS     PC
4064  016576  000004          ERRSS3: IOT
4065                                  ,IFNDF  $LONGER
4066  016600  047523    102           ,ASCII  \SUB\
4067                                  ,ENDC
4068                                  ,IFDF   $LONGER
4069                                  ,ASCII  /SUBSCRIPT OUT OF BOUNDS/
4070                                  ,ENDC
4071  016603    000                   ,BYTE 0
4072                                  ,EVEN
4073                                  ,EOT
4074
```

260


```
4075                                  ;                               SOURCE FILE #7
4076                                  ;+++++++++++++++++++++++++++++++++++++++++++++++
4077                                  ;  SUBROUTINE 'LSTPROG' CALLED BY JSR PC
4078                                  ;                               LISTS THE USER PROGRAM THROUGH 'PUTCHAR'
4079                                  ;                                       R0 - R4 DESTROYED
4080                                  ;                               R5  MUST POINT TO USER AREA
4081                                  ;                               SP  GOES ?? DEEPER AFTER JSR
4082  016604  016501  000016          LST$PROG:MOV    CODE(R5),R1
4083  016610  112102                  LST$LOOP:MOVB   (R1)+,R2
4084  016612  002142                          BGE     LST$PTH
4085  016614  120227  000201                  CMPB    R2,#,EOL
4086  016620  001444                          BEQ     LST$EOL
4087  016622  120227  000225                  CMPB    R2,#,EOF
4088  016626  001001                          BNE     ,+4
4089  016630  000207                          RTS     PC
4090  016632  120227  000374                  CMPB    R2,#,FLIT
4091  016636  103004                          BHIS    LST$SPEC
4092  016640  042702  177400                  BIC     #177400,R2
4093  016644  010703                          MOV     PC,R3
4094  016646  062703  161536          LST$PC: ADD     #KEYWDS-LST$PC,R3
4095  016652  010304                  LST$SRCH:MOV    R3,R4           ;R4 IS START OF KEYWORD
4096  016654  112300                          MOVB    (R3)+,R0
4097  016656  100376                          BPL     ,-2             ;LOOP TILL GET END KEYWORD
4098  016660  120002                          CMPB    R0,R2           ;COMPARE VALUES
4099  016662  001373                          BNE     LST$SRCH
4100  016664  112400                  LST$KWD:MOVB    (R4)+,R0
4101  016666  100403                          BMI     LST$CHK
4102  016670  004767  002012                  JSR     PC,PUTCHAR
4103  016674  000773                          BR      LST$KWD
4104  016676  124127  000211          LST$CHK:CMPB    -(R1),#,NEXT
4105  016702  001410                          BEQ     LST$NEXT
4106  016704  122127  000202                  CMPB    (R1)+,#,FN
4107  016710  001337                          BNE     LST$LOOP
4108  016712  112100                  LST$FN: MOVB    (R1)+,R0
4109  016714  062700  000200                  ADD     #'A-'A+2,R0
4110  016720  006000                          ROR     R0
4111  016722  000427                          BR      LST$CHAR
4112  016724  062701  000013          LST$NEXT:ADD    #13,R1
4113  016730  000727                          BR      LST$LOOP
4114  016732  111102                  LST$EOL:MOVB    (R1),R2
4115  016734  100010                          BPL     LST$LIN
4116  016736  120227  000225                  CMPB    R2,#,EOF
4117  016742  001015                          BNE     LST$BSLH
4118  016744  004167  000530          LST$CRLF:JSR    R1,MSGODEV
4119  016750     015     012     000          ,BYTE   CR,LF,0
4120          016754                          ,EVEN
4121  016754  000715                          BR      LST$LOOP
4122  016756  000302                  LST$LIN:SWAB    R2
4123  016760  005201                          INC     R1
4124  016762  151102                          BISB    (R1),R2
4125  016764  005301                          DEC     R1
4126  016766  061502                          ADD     (R5),R2
4127  016770  021227  177775                  CMP     (R2),#,SCALAR
4128  016774  103763                          BLO     LST$CRLF
4129  016776  012700  000134          LST$BSLH:MOV    #'\,R0
```

261

```
 4130  017002  004767  001700          LST%CHAR:JSR     PC,PUTCHAR
 4131  017006  000708                  BR      LSTLOOP
 4132  017010  120227  000374          LST%SPEC:CMPB    R2,#,FLIT
 4133  017014  001425                  BEQ     LST%FLIT
 4134  017016  120227  000376                  CMPB    R2,#,ILITZ
 4135  017022  103406                  BLO     LST%IL1
 4136  017024  001412                  BEQ     LST%IL2
 4137  017026  112100          LST%TEXT:MOVB    (R1)+,R0
 4138  017030  001667                  BEQ     LSTLOOP
 4139  017032  004767  001650                  JSR     PC,PUTCHAR
 4140  017036  000773                  BR      LST%TEXT
 4141  017040  105065  000043          LST%IL1: CLRB    FAC2+1(R5)
 4142  017044  112165  000042                  MOVB    (R1)+,FAC2(R5)
 4143  017050  000404                  BR      LST%IL0
 4144  017052  112165  000043          LST%IL2: MOVB    (R1)+,FAC2+1(R5)
 4145  017056  112165  000042                  MOVB    (R1)+,FAC2(R5)
 4146  017062  005065  000040          LST%IL0: CLR     FAC1(R5)
 4147  017066  000410                  BR      LST%FL0
 4148  017070  112165  000041          LST%FLIT:MOVB    (R1)+,FAC1+1(R5)
 4149  017074  112165  000040                  MOVB    (R1)+,FAC1(R5)
 4150  017100  112165  000043                  MOVB    (R1)+,FAC2+1(R5)
 4151  017104  112165  000042                  MOVB    (R1)+,FAC2(R5)
 4152  017110  004767  000610          LST%FL0: JSR     PC,NUMOUT
 4153  017114  000167  177470          LST%JMP: JMP     LSTLOOP
 4154  017120  000302                  LST%PTR: SWAB    R2
 4155  017122  152102                  BISB    (R1)+,R2
 4156  017124  061502                  ADD     (R5),R2
 4157  017126  021227  177775                  CMP     (R2),#,SCALAR
 4158  017132  103038                  BHIS    LST%VAR
 4159  017134  011202                  MOV     (R2),R2
 4160  017136  002405                  BLT     LST%LNO
 4161  017140  010265  000042                  MOV     R2,FAC2(R5)
 4162  017144  005065  000040                  CLR     FAC1(R5)
 4163  017150  000414                  BR      LST%LNX
 4164  017152  110265  000043          LST%LNO: MOVB    R2,FAC2+1(R5)
 4165  017156  105065  000042                  CLRB    FAC2(R5)
 4166  017162  000302                  SWAB    R2
 4167  017164  110265  000040                  MOVB    R2,FAC1(R5)
 4168  017170  105065  000041                  CLRB    FAC1+1(R5)
 4169  017174  062765  043600  000040          ADD     #43600,FAC1(R5)
 4170  017202  004767  000516          LST%LNX: JSR     PC,NUMOUT
 4171  017206  012700  000040                  MOV     #BL,R0
 4172  017212  000673                  BR      LST%CHAR
 4173  017214  116200  000010          LST%VAR: MOVB    10(R2),R0
 4174  017220  004767  001402                  JSR     PC,PUTCHAR
 4175  017224  116200  000011                  MOVB    11(R2),R0
 4176  017230  001402                  BEQ     .+6
 4177  017232  004767  001450                  JSR     PC,PUTCHAR
 4178
 4179                                          .IFDF   $NOSTR
 4180                                          BR      LST%JMP
 4181                                          .ENDC
 4182                                          .IFNDF  $NOSTR
 4183  017236  021227  177777                  CMP     (R2),#,SVAR
 4184  017242  001324                  BNE     LST%JMP
```

262


```
 4185  017244  012700  000044                  MOV     #'$,R0
 4186  017250  000654                  BR      LST%CHAR
 4187                                          .ENDC
 4188
```

263

```
                                        .IFNDF  $NOSTR

                                ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                ;  SUBROUTINE TMAKESTR  CALLED BY JSR PC
                                ;                       MAKES A BASIC STRING FROM A STRING OF CHARACTERS
                                ;                       R0  DESTROYED
                                ;                       R1  UNUSED
                                ;                       R2  MUST POINT TO A WORD WHICH CONTAINS 3 LESS
                                ;                           THAN THE ADDRESS OF THE FIRST CHARACTER
                                ;                       R3  DESTROYED
                                ;                       R4  UNUSED
                                ;                       R5  MUST POINT TO USER AREA
                                ;                       SP  ON ENTRY 0(SP) MUST CONTAIN # CHARACTERS
                                ;                           ON EXIT 0(SP) CONTAINS POINTER TO THE STRING
017252  004767  000112          MAKESTR:JSR     PC,MAKETRY      ;WILL NEW STRING FIT?
017256  101413                          BLOS    MAKGOT          ;YES
017260  004767  171722                  JSR     PC,DNPACK       ;NO, GARBAGE COLLECT
017264  004767  000100                  JSR     PC,MAKETRY      ;TRY AGAIN
017270  101406                          BLOS    MAKGOT          ;FITS THIS TIME
017272  000004          ERRSOV: IOT
                                        .IFNDF  $LONGER
017274  051523     117                  .ASCII  \$SO\
                                        .ENDC
                                        .IFDF   $LONGER
                                        .ASCII  /STRING STORAGE OVERFLOW/
                                        .ENDC
017277     000                          .BYTE 0
                                        .EVEN
017300  004767  000064          MAKCHK: JSR     PC,MAKETRY
017304  101372                          BHI     ERRSOV
017306  010065  000074          MAKGOT: MOV     R0,HISTR(R5)    ;SAVE NEW HIGH STRING ADDR
017312  016600  000002                  MOV     2(SP),R0        ;R0 NOW CONTAINS # CHARS,
017316  011202                          MOV     (R2),R2
017320  062702  000003                  ADD     #3,R2           ;R2 NOW CONTAINS ADDRESS OF CHARS,
017324  110023                          MOVB    R0,(R3)+                ;R3 PUTS CHARACTERS IN
017326  122323                          CMPB    (R3)+,(R3)+
017330  112223                          MOVB    (R2)+,(R3)+
017332  005300                          DEC     R0
017334  003375                          BGT     .-4
017336  016600  000002                  MOV     2(SP),R0
017342  110013                          MOVB    R0,(R3)
017344  160003                          SUB     R0,R3
017346  010600                          MOV     SP,R0
017350  005720                          TST     (R0)+
017352  110043                          MOVB    R0,-(R3)
017354  000300                          SWAB    R0
017356  110043                          MOVB    R0,-(R3)
017360  005303                          DEC     R3
017362  010366  000002                  MOV     R3,2(SP)
017366  000207                          RTS     PC

                                ;       SUBROUTINE TO CHECK ROOM ENOUGH FOR STRING
017370  016500  000074          MAKETRY:MOV     HISTR(R5),R0
017374  010003                          MOV     R0,R3
017376  066600  000004                  ADD     4(SP),R0
```

264


```
017402  062700  000004                  ADD     #4,R0
017406  020065  000012                  CMP     R0,HIFREE(R5)
017412  000207                          RTS     PC
                                        .ENDC


                                ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                ;  SUBROUTINE TMPYTEN1  CALLED BY JSR PC
                                ;                       MULTIPLIES R3,R4 BY DECIMAL 10
                                ;                       R0,R1,R2  UNUSED
                                ;                       R3,R4  IS THE 32 BIT UNSIGNED INTEGER TO MULTPLY
                                ;                       R5  UNUSED
                                ;                       SP  GOES 4 DEEPER AFTER JSR
017414  010346          MPYTEN1:MOV     R3,-(SP)
017416  010446                  MOV     R4,-(SP)
017420  006304                  ASL     R4
017422  006103                  ROL     R3
017424  006304                  ASL     R4
017426  006103                  ROL     R3
017430  062604                  ADD     (SP)+,R4
017432  005503                  ADC     R3
017434  062603                  ADD     (SP)+,R3
017436  006304                  ASL     R4
017440  006103                  ROL     R3
017442  000207                  RTS     PC
```

265

```
                                        ;
                                        ; MSG - OUTPUT A LINE TO TTY
                                        ; MSGODEV - OUTPUT A LINE TO CURR, OUTPUT DEV,
                                        ;
                                        ; CALL: JSR     R1,MSG (MSGODEV)
                                        ;       .ASCII  "[MESSAGE]"
                                        ;       .BYTE   0
                                        ;       .EVEN
                                        ;                       ;RETURN HERE
                                        ;
                                        ;  USES R0
                                        ;
                                        ;  SET UP LINE WITH NUMBER OF BYTES TERMINATED
                                        ;   BY A BYTE OF 000.
                                        ;
                                        MSGERR:
                                                .IFNDF  $LONGER
017444  112700  000045                          MOVB    #'%,R0
017450  000401                                  BR      MSGCOM
                                                .ENDC
017452  112100                          MSG:    MOVB    (R1)+,R0
017454  105065  000107                  MSGCOM: CLRB    CNOFLG(R5)
017460  016546  000034                          MOV     COLUMN(R5),-(SP)        ;SAVE FOR TEMP SWITCH
017464  012765  000036  000034                  MOV     #CLMNTTY,COLUMN(R5)
017472  060565  000034                          ADD     R5,COLUMN(R5)
017476  000404                                  BR      FRSTOUT
017500  016546  000034                  MSGODEV:MOV     COLUMN(R5),-(SP)
017504  112100                          DOMSG:  MOVB    (R1)+,R0
017506  001403                                  BEQ     LINDUN
017510  004767  001172                  FRSTOUT:JSR     PC,PUTCHAR              ;PUT CHAR, TO TTY
017514  000773                                  BR      DOMSG           ;DO NEXT CHAR
017516  005201                          LINDUN: INC     R1              ;TO INSURE RETURN ON
017520  006201                                  ASR     R1              ;  EVEN WORD BOUNDARY
017522  006301                                  ASL     R1
017524  012665  000034                          MOV     (SP)+,COLUMN(R5)
017530  000201                                  RTS     R1
```

266


```
                                        ;*****************************************************
                                        ;  SUBROUTINE 'NORM'    NORMALIZES INTEGER CONTAINED IN
                                        ;                       R3 AND R4, MULTIPLIED BY EXPONENT
                                        ;                       IN R2,  ANSWER IS IN R3, R4,
                                        ;                       CALLED BY JSR PC
017532  012746  000237                  NORM:   MOV     #237,-(SP)
017536  005703                                  TST     R3
017540  001413                                  BEQ     LITTST
017542  100003                                  BPL     LITNORM
017544  006003                                  ROR     R3
017546  006004                                  ROR     R4
017550  005216                                  INC     (SP)
017552  030327  040000                  LITNORM:BIT     R3,#40000
017556  001010                                  BNE     LITOK
017560  006304                                  ASL     R4
017562  006103                                  ROL     R3
017564  005316                                  DEC     (SP)
017566  000771                                  BR      LITNORM
017570  005704                          LITTST: TST     R4
017572  001367                                  BNE     LITNORM
017574  005726                                  TST     (SP)+
017576  000207                                  RTS     PC
017600  005702                          LITOK:  TST     R2
017602  001425                                  BEQ     LITSTO
017604  100416                                  BMI     LITDIV
017606  006203                                  ASR     R3
017610  006004                                  ROR     R4
017612  006203                                  ASR     R3
017614  006004                                  ROR     R4
017616  006203                                  ASR     R3
017620  006004                                  ROR     R4
017622  006203                                  ASR     R3
017624  006004                                  ROR     R4
017626  062716  000004                          ADD     #4,(SP)
017632  004767  177556                          JSR     PC,MPYTEN
017636  005302                                  DEC     R2
017640  000744                                  BR      LITNORM
017642  004767  171302                  LITDIV: JSR     PC,DIVTEN
017646  005202                                  INC     R2
017650  000740                                  BR      LITNORM
017652  006203                          LITSHR: ASR     R3
017654  006004                                  ROR     R4
017656  030327  177000                  LITSTO: BIT     R3,#177000
017662  001373                                  BNE     LITSHR
017664  005716                                  TST     (SP)
017666  003002                                  BGT     .+6
017670  012716  000001                          MOV     #1,(SP)
017674  021627  000377                          CMP     (SP),#377
017700  101402                                  BLOS    .+6
017702  012716  000377                          MOV     #377,(SP)
017706  110366  000001                          MOVB    R3,1(SP)
017712  012603                                  MOV     (SP)+,R3
017714  000303                                  SWAB    R3
017716  006003                                  ROR     R3
017720  006004                                  ROR     R4
```

267

```
 4364  017722  000207                        RTS     PC
 4365
```

268


```
 4366                                ;-----------------------------------------------------
 4367                                ;  'NUMOUT' SUBROUTINE
 4368                                ;               CALLED BY JSR PC,NUMOUT
 4369                                ;               OUTPUTS AN UNSIGNED NUMBER FROM THE FAC
 4370                                ;               VIA THE SUBROUTINE PUTCHAR;
 4371  017724  004567  001142        NUMOUT: JSR     R5,SAVREG
 4372  017730  012701  020706                MOV     #PUTCHAR,R1
 4373  017734  000446                        BR      NUMSTI
 4374
 4375
 4376                                ;-----------------------------------------------------
 4377                                ;  'NUMSGN' SUBROUTINE
 4378                                ;               CALLED BY JSR PC,NUMSGN
 4379                                ;                       ;WORD OUTPUT
 4380                                ;               WHERE OUTPUT IS THE OUTPUT ROUTINE,
 4381                                ;               WHICH MAY BE PUTCHAR OR SAVCHAR,
 4382                                ;               OUTPUTS A SIGNED NUMBER FROM THE
 4383                                ;               FAC VIA THE OUTPUT ROUTINE,
 4384  017736  017600  000000        NUMSGN: MOV     @(SP),R0
 4385  017742  062716  000002                ADD     #2,(SP)
 4386  017746  004567  001120                JSR     R5,SAVREG
 4387  017752  010001                        MOV     R0,R1
 4388  017754  005765  000040                TST     FAC1(R5)
 4389  017760  001418                        BEQ     NUMFIX
 4390  017762  100025                        BPL     NUMPOS
 4391  017764  062765  100000  000040        ADD     #100000,FAC1(R5)
 4392  017772  001016                        BNE     NUMNEG
 4393  017774  005065  000042                CLR     FAC2(R5)
 4394  020000  001016                        BNE     NUMPOS
 4395  020002  005765  000042        NUMFIX: TST     FAC2(R5)
 4396  020006  100013                        BPL     NUMPOS
 4397  020010  005465  000042                NEG     FAC2(R5)
 4398  020014  102005                        BVC     NUMNEG
 4399  020016  012765  044000  000040        MOV     #44000,FAC1(R5)
 4400  020024  005065  000042                CLR     FAC2(R5)
 4401  020030  012700  000055        NUMNEG: MOV     #'-,R0
 4402  020034  000405                        BR      NUMPRS
 4403                                NUMPOS:
 4404                                        .IFNDF  $NOSTR
 4405  020036  020127  020570                CMP     R1,#SAVCHAR
 4406  020042  001403                        BEQ     NUMSTI          ;DON'T OUTPUT BLANK FOR STRS
 4407                                        .ENDC
 4408
 4409  020044  012700  000040                MOV     #BL,R0
 4410  020050  004711                NUMPRS: JSR     PC,(R1)
 4411
 4412                                NUMSTI:
 4413  020052  016504  000042                MOV     FAC2(R5),R4     ;R4 = LOW ORDER MANTISSA,
 4414  020056  005002                        CLR     R2              ;R2 = DECIMAL EXPONENT;
 4415  020060  016503  000040                MOV     FAC1(R5),R3     ;R3 = HIGH ORDER MANTISSA,
 4416  020064  001010                        BNE     NUMFLT
 4417  020066  012700  000037                MOV     #37,R0          ;R0 = BINARY EXPONENT,
 4418  020072  005704                        TST     R4
 4419  020074  001016                        BNE     NUMSHFT
 4420  020076  012700  000060                MOV     #'0,R0
```

269

```
 4421  020102  004711                        JSR     PC,(R1)
 4422  020104  000207                        RTS     PC
 4423  020106  010300              NUMFLT:   MOV     R3,R0
 4424  020110  006300                        ASL     R0
 4425  020112  105000                        CLRB    R0
 4426  020114  000300                        SWAB    R0
 4427  020116  162700  000171              SUB     #171,R0
 4428  020122  042703  177000              BIC     #177600,R3
 4429  020126  052703  000200              BIS     #200,R3
 4430  020132  005300              NUMSHFT:  DEC     R0
 4431  020134  006304                        ASL     R4
 4432  020136  006103                        ROL     R3
 4433  020140  030327  040000        NUMNORM:  BIT     R3,#40000
 4434  020144  001772                        BEQ     NUMSHFT
 4435  020146  005700                        TST     R0
 4436  020150  003016                        BGT     NUMBIG
 4437  020152  006203                        ASR     R3
 4438  020154  006004                        ROR     R4
 4439  020156  006203                        ASR     R3
 4440  020160  006004                        ROR     R4
 4441  020162  006203                        ASR     R3
 4442  020164  006004                        ROR     R4
 4443  020166  006203                        ASR     R3
 4444  020170  006004                        ROR     R4
 4445  020172  062700  000004              ADD     #4,R0
 4446  020176  004767  177212              JSR     PC,MPYTEN
 4447  020202  005302                        DEC     R2
 4448  020204  000755                        BR      NUMNORM
 4449  020206  020027  000004        NUMBIG:   CMP     R0,#4
 4450  020212  003407                        BLE     NUMOK
 4451  020214  004767  170730        NUMDIV:   JSR     PC,DIVTEN
 4452  020220  005202                        INC     R2
 4453  020222  000746                        BR      NUMNORM
 4454  020224  005200              NUMALN:   INC     R0
 4455  020226  006203                        ASR     R3
 4456  020230  006004                        ROR     R4
 4457  020232  020027  000004        NUMOK:    CMP     R0,#4
 4458  020236  002772                        BLT     NUMALN
 4459  020240  020327  050000                CMP     R3,#50000
 4460  020244  103363                        BHIS    NUMDIV
 4461  020246  062704  001250              ADD     #1250,R4        ;ROUNDING;
 4462  020252  103010                        BCC     NORNDOV
 4463  020254  005203                        INC     R3
 4464  020256  020327  050000                CMP     R3,#50000
 4465  020262  103404                        BLO     NORNDOV
 4466  020264  012703  004000              MOV     #4000,R3
 4467  020270  005004                        CLR     R4
 4468  020272  005202                        INC     R2
 4469  020274  012700  000006        NORNDOV:  MOV     #6,R0
 4470  020300  010346              NUMDIG:   MOV     R3,-(SP)
 4471  020302  000316                        SWAB    (SP)
 4472  020304  006016                        ROR     (SP)
 4473  020306  006016                        ROR     (SP)
 4474  020310  006016                        ROR     (SP)
 4475  020312  042716  177760              BIC     #177760,(SP)
```

270


```
 4476  020316  062716  000060              ADD     #60,(SP)
 4477  020322  042703  174000              BIC     #174000,R3
 4478  020326  004767  177062              JSR     PC,MPYTEN
 4479  020332  005300                        DEC     R0
 4480  020334  003361                        BGT     NUMDIG
 4481  020336  010603                        MOV     SP,R3
 4482  020340  062703  000014              ADD     #14,R3
 4483  020344  020227  177776              CMP     R2,#-2
 4484  020350  002404                        BLT     NUMFM1
 4485  020352  001446                        BEQ     NUMFM2
 4486  020354  020227  000006              CMP     R2,#6
 4487  020360  002451                        BLT     NUMFM3
 4488  020362  014300              NUMFM1:   MOV     -(R3),R0
 4489  020364  004711                        JSR     PC,(R1)
 4490  020366  012700  000056              MOV     #'.,R0
 4491  020372  004711                        JSR     PC,(R1)
 4492  020374  012704  000005              MOV     #5,R4
 4493  020400  014300              NUMLP1:   MOV     -(R3),R0
 4494  020402  004711                        JSR     PC,(R1)
 4495  020404  005304                        DEC     R4
 4496  020406  003374                        BGT     NUMLP1
 4497  020410  012700  000105              MOV     #'E,R0
 4498  020414  004711                        JSR     PC,(R1)
 4499  020416  012700  000053              MOV     #'+,R0
 4500  020422  005702                        TST     R2
 4501  020424  100003                        BPL     .+10
 4502  020426  012700  000055              MOV     #'-,R0
 4503  020432  005402                        NEG     R2
 4504  020434  004711                        JSR     PC,(R1)
 4505  020436  012700  000060              MOV     #'0,R0
 4506  020442  162702  000012              SUB     #12,R2
 4507  020446  100402                        BMI     .+6
 4508  020450  005200                        INC     R0
 4509  020452  000773                        BR      .-10
 4510  020454  004711                        JSR     PC,(R1)
 4511  020456  010200                        MOV     R2,R0
 4512  020460  062700  000072              ADD     #'0+12,R0
 4513  020464  004711                        JSR     PC,(R1)
 4514  020466  000435                        BR      NUMEXIT
 4515  020470  012700  000056        NUMFM2:   MOV     #'.,R0
 4516  020474  004711                        JSR     PC,(R1)
 4517  020476  012700  000060              MOV     #'0,R0
 4518  020502  004711                        JSR     PC,(R1)
 4519  020504  012704  000006        NUMFM3:   MOV     #6,R4
 4520  020510  010600                        MOV     SP,R0
 4521  020512  122027  000060              CMP     (R0)+,#'0
 4522  020516  001002                        BNE     .+6
 4523  020520  005304                        DEC     R4
 4524  020522  000773                        BR      .-10
 4525  020524  020402                        CMP     R4,R2
 4526  020526  003002                        BGT     .+6
 4527  020530  010204                        MOV     R2,R4
 4528  020532  005204                        INC     R4
 4529  020534  005202                        INC     R2
 4530  020536  005202                        INC     R2
```

270

```
 4531  020540  005302                NUMLP3: DEC     R2
 4532  020542  001003                        BNE     .+10
 4533  020544  012700  000030                MOV     #'0,R0
 4534  020550  004711                        JSR     PC,(R1)
 4535  020552  214300                        MOV     -(R3),R0
 4536  020554  004711                        JSR     PC,(R1)
 4537
 4538  020556  005304                        DEC     R4
 4539  020560  003367                        BGT     NUMLP3
 4540  020562  062706  000014        NUMEXIT:ADD     #14,SP
 4541  020566  000207                        RTS     PC
 4542
 4543                                        .IFNDF  $NOSTR
 4544                                SAVCHAR:
 4545  020570  004567  000276                JSR     R5,SAVREG
 4546  020574  016501  000000                MOV     T2(R5),R1
 4547  020600  110021                        MOVB    R0,(R1)+
 4548  020602  010165  000000                MOV     R1,T2(R5)
 4549  020606  005265  000000                INC     T1(R5)
 4550  020612  000207                        RTS     PC
 4551                                        .ENDC
 4552
```

272


```
 4553                                ;
 4554                                ; PUTBYT - PUT BYTE INTO RING BUFFER
 4555                                ;
 4556                                ; CALL: MOV     [BUFF, HDR'],R1
 4557                                ;       MOV     [CHAR],R0
 4558                                ;       JSR     PC,PUTBYT
 4559                                ;
 4560                                ;  USES R0,R1,R2,R3
 4561                                ;
 4562                                ;  RAISE PRIORITY SINCE THIS MAY BE CALLED AT
 4563                                ;   MAINSTREAM, BUT ACCESSES POINTERS THAT MAY CHANGE AT
 4564                                ;   INTERUPT LEVEL
 4565                                ;  RETURNS WITH 'C' BIT: CLR IF BYTE GOT IN
 4566                                ;                        SET IF NO ROOM
 4567                                ;
 4568  020614  012702  177776        PUTBYT: MOV     #PS,R2
 4569  020620  011246                        MOV     (R2),-(SP)      ;SAVE STATUS
 4570  020622  012712  000340                MOV     #PR7,(R2)       ;REPLACE WITH HIGHEST
 4571  020626  016103  000010                MOV     BPUT(R1),R3
 4572  020632  005203                        INC     R3              ;CANDIDATE FOR NEW POSITION
 4573  020634  020361  000002                CMP     R3,BEND(R1)     ;NEED TO WRAP AROUND?
 4574  020640  101402                        BLOS    NOWRAP
 4575  020642  016103  000000                MOV     BSTRT(R1),R3    ;YES! NEW POS, AT TOP
 4576  020646  020361  000004        NOWRAP: CMP     R3,BGET1(R1)    ;IF EQUAL TO EITHER GET PTR,,
 4577  020652  001412                        BEQ     NOROOM          ;  THEN NO ROOM
 4578  020654  020361  000006                CMP     R3,BGET2(R1)
 4579  020660  001407                        BEQ     NOROOM
 4580  020662  110071  000010                MOVB    R0,@BPUT(R1)
 4581  020666  010361  000010                MOV     R3,BPUT(R1)     ;UPDATE PTR,
 4582  020672  012612                        MOV     (SP)+,(R2)      ;STATUS BACK
 4583  020674  000241                        CLC                     ;SIGNAL SUCCESS
 4584  020676  000207                        RTS     PC
 4585  020700  012612        NOROOM: MOV     (SP)+,(R2)      ;STATUS BACK
 4586  020702  000261                        SEC                     ;SHOW FAILURE
 4587  020704  000207                        RTS     PC
 4588
```

273

```
                                        ;
                                        ; PUTCHAR = PUT CHAR TO CURRENT OUTPUT DEV,
                                        ;
                                        ; CALL: (LOAD ODEV(R5))
                                        ;       MOV     [CHAR],R0
                                        ;       JSR     PC,PUTCHAR
                                        ;
                                        ;  SAVES ALL REGS,
                                        ;
                                        ;  TRIES TO FIT CHAR INTO PROPER BUFFER.  GOES TO IOWAIT ON
                                        ;   FAILURE,
                                        ;
020706  004567  000160          PUTCHAR:JSR     R5,SAVREG
020712  012746  047777                  MOV     #47777,-(SP)    ;INIT TIME OUT COUNT
020716  005275  000034                  INC     @COLUMN(R5)
020722  105765  000100                  TSTB    CNCFLG(R5)
020726  001402                          BEQ     PUTCCO
020730  000167  160210                  JMP     READY
020734  116504  000076          PUTCCO: MOVB    ODEV(R5),R4     ;OUTPUT DEVICE CODE
020740  001003                          BNE     HDRSTUP         ;IF NOT TTY, CONTINUE
020742  105765  000107                  TSTB    CNOFLG(R5)      ;CHECK FOR ^O?
020746  001032                          BNE     OUT010          ;IF SET EXIT IMMEDIATELY
020750  016401  021040          HDRSTUP:MOV     TAB1(R4),R1
020754  005704                          TST     R4              ;WHICH DEVICE?
020756  001002                          BNE     PUTIT           ;NOT TTY
020760  060501                          ADD     R5,R1           ;NEED CURR; USER'S BUFF,
020762  011101                          MOV     (R1),R1         ;NOW, ACTUAL HEADER
020764  004767  177624          PUTIT:  JSR     PC,PUTBYT
020770  103007                          BCC     ITSIN
020772  004767  175104                  JSR     PC,IOWAIT       ;NO ROOM! WAIT
020776  005316                          DEC     (SP)            ;TIME OUT
021000  001371                          BNE     PUTIT
021002  105061  000012                  CLRB    BFSPEC(R1)
021006  000422                          BR      DEVERR
021010  105061  000012          ITSIN:  CLRB    BFSPEC(R1)
021014  012774  000100  021046          MOV     #100,@TAB2(R4)  ;ENABLE INTERUPT ON CHOSEN DEV;
021022  005265  000070                  INC     RNDCT(R5)
021026  105761  000012                  TSTB    BFSPEC(R1)      ;I/O ERROR?
021032  001010                          BNE     DEVERR
021034  005726          OUT010: TST     (SP)+           ;POP TIME OUT COUNT
021036  000207                  RTS     PC              ;RESTORES REGS, AND RETURNS
                                        ;
                                        ; TAB1 = DEV, BUFFER HDRS, (TPHD IS OFFSET INTO USRAREA
                                        ;  FOR ADDR, TO TPRTR, BUFF, HDR,)
                                        ;
021040  000100          TAB1:   .       TPHD
                                        .IFNDF $NOPTP
021042  023760                  .       PPBFHD
                                        .ENDC
                                        .IFDF $NOPTP
                                        0
                                        .ENDC
                                        .IFNDF $NOLPT
021044  024024                  .       LPBFHD
```



```
                                        .ENDC
                                        .IFDF $NOLPT
                                        0
                                        .ENDC
                                        ;
                                        ; TAB2 = DEV, STATUS REG,
                                        ;
021046  177564          TAB2:   .       TPS
021050  177554                  .       PPS
021052  177514                  .       LPS
021054  000004          DEVERR: IOT
                                        .IFNDF  $LONGER
021056  047104  122             .ASCII  \ONR\
                                        .ENDC
                                        .IFDF   $LONGER
                                        .ASCII  \DEVICE NOT READY\
                                        .ENDC
021061  000             .BYTE 0
                                        .EVEN
                                        .EOT
```

```
                                        ;                                SOURCE FILE #8
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                        ;
                                        ; SAVREG - SAVE ALL REGISTERS
                                        ;
                                        ; CALL: [JSR    PC,SUBR,]       ;MUST CALL SAVREG FROM SUBRTN,
                                        ;       JSR     R5,SAVREG
                                        ;
                                        ;  PUSHES ALL REGS, AND EXITS WITH ADDR, ON STACK
                                        ;   WHICH BRINGS CONTROL BACK WHEN SUBR, DOES /RTS      PC/,
                                        ;   THEN, RESTORES REGS, AND DOES /RTS PC/ WHICH RETURNS TO JUST
                                        ;   AFTER CALL TO INITIAL SUBROUTINE (MODIFIED BOWERING SPECIAL)
                                        ;
021062  011646                 SAVRG1: MOV     (SP),-(SP)
021064  012766  021132  000002          MOV     #INTEXT,2(SP)
021072  010446                 SAVREG: MOV     R4,-(SP)
021074  010346                         MOV     R3,-(SP)
021076  010246                         MOV     R2,-(SP)
021100  010146                         MOV     R1,-(SP)
021102  010046                         MOV     R0,-(SP)
021104  010546                         MOV     R5,-(SP)        ;FOR RETURN TO SUBR,
021106  016605  000014                 MOV     12.(SP),R5      ;GET OLD R5
021112  004736                         JSR     PC,@(SP)+       ;STACK NEXT LOC,!
021114  012600                 RESREG: MOV     (SP)+,R0
021116  012601                         MOV     (SP)+,R1
021120  012602                         MOV     (SP)+,R2
021122  012603                         MOV     (SP)+,R3
021124  012604                         MOV     (SP)+,R4
021126  012605                         MOV     (SP)+,R5
021130  000207                         RTS     PC              ;BACK TO CALLER OF INIT, SUBR,
                                        ;
021132  000002                 INTEXT: RTI
```

```
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                        ;
                                        ; SCANPND - SCAN TOKEN FOR '#'
                                        ;
                                        ; CALL: MOV     [PTR, TO BYTE BEFORE '#' TOKEN],R1
                                        ;       JSR     PC,SCANPND
                                        ;
                                        ;  USES R1
                                        ;
021134  012765  177775  000030  SCANPND:MOV     #,SCALAR,LINENO(R5)
021142  005201                         INC     R1
021144  122127  000201                 CMPB    (R1)+,#,POUND
021150  001402                         BEQ     PNDGO
021152  000167  171306                 JMP     ERRSX5
021156  000207                 PNDGO:  RTS     PC
```

276


```
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                        ;  SUBROUTINE 'SKIPEOL' CALLED BY JSR PC
                                        ;                       MOVES R1 PAST THE NEXT 'EOL'
                                        ;                       R0  UNUSED
                                        ;                       R1  MODIFIED TO REFLECT OPERATION
                                        ;                       R2,R3,R4,R5  UNUSED
                                        ;                       SP  GOES NO DEEPER AFTER JSR
021160  062701  000003          SKIP05: ADD     #3,R1
021164  005201                  SKIP02: INC     R1
021166  005201                  SKIP01: INC     R1
021170  105721                  SKIPEOL:TSTB    (R1)+
021172  002375                          BGE     SKIP01
021174  124127  000374                  CMPB    -(R1),#,FLIT
021200  001767                          BEQ     SKIP05
021202  101014                          BHI     SKIPHI
021204  122127  000201                  CMPB    (R1)+,#,EOL
021210  001420                          BEQ     SKIPRTS
021212  124127  000202                  CMPB    -(R1),#,FN
021216  001762                          BEQ     SKIP02
021220  122127  000211                  CMPB    (R1)+,#,NEXT
021224  001361                          BNE     SKIPEOL
021226  062701  000012                  ADD     #12,R1
021232  000756                          BR      SKIPEOL
                                        ;       EITHER ,ILIT1(375), ,ILIT2(376) OR ,TEXT(377)
021234  122127  000376          SKIPHI: CMPB    (R1)+,#,ILIT2
021240  001751                          BEQ     SKIP02
021242  103751                          BLO     SKIP01
021244  105721                  SKIPTXT:TSTB    (R1)+
021246  001376                          BNE     SKIPTXT
021250  000747                          BR      SKIPEOL
021252  000207                  SKIPRTS:RTS     PC
```

```
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                        ;  'STOVAR' SUBROUTINE
                                        ;               CALLED BY JSR PC,STOVAR
                                        ;               STORES FAC1, FAC2 IN THE VARIABLE OR
                                        ;               ARRAY ELEMENT LAST ADDRESSED BY THE
                                        ;               SUBROUTINE 'GETVAR'
021254  016502  000022          STOVAR: MOV     VARSAV(R5),R2
                                        ;
                                        ,IFNDF  $NOSTR
021260  021227  177777                  CMP     (R2),#,SVAR
021264  001421                          BEQ     ERRMX7
                                        ,ENDC
                                        ;
021266  016500  000024                  MOV     SS1SAV(R5),R0
021272  100405                          BMI     STONOSS
021274  016503  000026                  MOV     SS2SAV(R5),R3
021300  004767  175104                  JSR     PC,LOCGET
021304  000404                          BR      STOCOMM
021306  022227  177775          STONOSS:CMP     (R2)+,#,SCALAR
021312  001401                          BEQ     STOCOMM
021314  011202                          MOV     (R2),R2
```

277

```
 4775  021316  016522  000040          STOCOMMI:MOV     FAC1(R5),(R2)+
 4776  021322  016512  000042                   MOV     FAC2(R5),(R2)
 4777  021326  000207                           RTS     PC
 4778
 4779                                           .IFNDF  $NOSTR
 4780  021330  000167  171456          ERRMX7:  JMP     ERRMIX
 4781                                           .ENDC
 4782
```

278


```
 4783                                           .IFNDF  $NOSTR
 4784                                   ;*******************************************************
 4785                                   ;  'STOSVAR' SUBROUTINE
 4786                                   ;               CALLED BY JSR PC,STOSVAR
 4787                                   ;               STORES A STRING VARIABLE AS ADDRESSED BY
 4788                                   ;               VARSAV
 4789  021334  016502  000022          STOSVAR:MOV     VARSAV(R5),R2
 4790  021340  021227  177777                   CMP     (R2),#,SVAR
 4791  021344  001037                           BNE     ERRMX6
 4792  021346  016500  000024                   MOV     SS1SAV(R5),R0
 4793  021352  100405                           BMI     STOSSNO
 4794  021354  016503  000026                   MOV     SS2SAV(R5),R3
 4795  021360  004767  175024                   JSR     PC,LOCGET
 4796  021364  000406                           BR      STOMMCO
 4797  021366  005722                  STOSSNO: TST     (R2)+
 4798  021370  026227  000002  177777          CMP     2(R2),#-1
 4799  021376  001435                           BEQ     STOVTAB
 4800  021400  011202                           MOV     (R2),R2
 4801  021402  012603                  STOMMCO: MOV     (SP)+,R3
 4802  021404  012600                           MOV     (SP)+,R0
 4803  021406  010012                           MOV     R0,(R2)
 4804  021410  000405                           BR      STOVCOM
 4805  021412  012603                  STOVTAB: MOV     (SP)+,R3
 4806  021414  012600                           MOV     (SP)+,R0
 4807  021416  010012                           MOV     R0,(R2)
 4808  021420  005202                           INC     R2
 4809  021422  161502                           SUB     (R5),R2
 4810  021424  005200                  STOVCOM: INC     R0
 4811  021426  001405                           BEQ     STOSX           ;CHECK NULL STRING
 4812  021430  062700  000002                   ADD     #2,R0
 4813  021434  110240                           MOVB    R2,-(R0)
 4814  021436  000302                           SWAB    R2
 4815  021440  110240                           MOVB    R2,-(R0)
 4816  021442  000113                  STOSX:   JMP     (R3)
 4817  021444  000167  171342          ERRMX6:  JMP     ERRMIX
 4818                                           .ENDC
 4819
 4820
 4821
 4822
 4823                                   ;*******************************************************
 4824                                   ;  SUBROUTINE 'SUBSTK'  CALLED BY JSR PC
 4825                                   ;                       NEGATES 2(SP),2(SP) THEN CONTINUES IN 'ADDSTK'
 4826                                   ;                       R0,R1  UNUSED
 4827                                   ;                       R2,R3  DESTROYED
 4828                                   ;                       R4  MODIFIED TO REFLECT STACK USAGE
 4829                                   ;                       R5  MUST POINT TO USER AREA
 4830                                   ;                       SP  GOES ?? DEEPER AFTER JSR
 4831  021450  005766  000002          SUBSTK:  TST     2(SP)
 4832  021454  001405                           BEQ     SUBINT
 4833  021456  062766  100000  000002          ADD     #100000,2(SP)
 4834  021464  000167  166132          ADDSTK:  JMP     ADDSTK
 4835  021470  005466  000004          SUBINT:  NEG     4(SP)
 4836  021474  102373                           BVC     ADDSTK
 4837  021476  012766  044000  000002          MOV     #44000,2(SP)
```

279

```
 4838  021504  005066  000004            CLR     4(SP)
 4839  021510  000765                    BR      ADSTK1
 4840
```

280


```
 4841                                  ;+-----------------------------------------------
 4842                                  ;  'TRAN' SUBROUTINE
 4843                                  ;               CALLED BY JSR PC,TRAN
 4844                                  ;               TRANSLATES FROM INPUT ASCII CODE
 4845                                  ;               TO INTERNAL CODE, CONSISTING OF TOKENS,
 4846                                  ;               LINE NUMBER REFERENCES, SYMBOL TABLE
 4847                                  ;               REFERENCES, AND LITERALS,  ALWAYS RETURNS,
 4848                                  ;               NO MATTER WHAT IS INPUT,
 4849                                  ;               AS NEW SYMBOLS ARE ENCOUNTERED, BUILDS NEW
 4850                                  ;               ENTRIES INTO THE SYMBOL TABLE,
 4851                                  ;               ERRORS ARE TRANSLATED TO [,TEXT,,,']
 4852  021512  016502  000020  TRAN:   MOV     LINE(R5),R2
 4853  021516  010201                  MOV     R2,R1
 4854  021520  122127  000015          CMPB    (R1)+,#CR
 4855  021524  001375                  BNE     .-4
 4856  021526  016500  000016          MOV     CODE(R5),R0
 4857  021532  114140                  MOVB    -(R1),-(R0)
 4858  021534  020102                  CMP     R1,R2           ;R0 IS WHERE ORIGINAL TEXT IS,
 4859  021536  001375                  BNE     .-4             ;R1 IS WHERE TRANSLATED TEXT GOES,
 4860  021540  005065  000056  TRANLUP:CLR     T1(R5)          ;PREVIOUS ITEM WASN'T A VAR,
 4861  021544  122027  000040  TRANVAR:CMPB    (R0)+,#BL
 4862  021550  001775                  BEQ     TRANVAR
 4863  021552  124027  000015          CMPB    -(R0),#CR
 4864  021556  001003                  BNE     TRYNUM
 4865  021560  112711  000201          MOVB    #EOL,(R1)
 4866  021564  000207                  RTS     PC
 4867  021566  121027  000056  TRYNUM: CMPB    (R0),#'.
 4868  021572  001406                  BEQ     NUMJMP
 4869  021574  121027  000060          CMPB    (R0),#'0
 4870  021600  103405                  BLO     TRYKWD
 4871  021602  121027  000071          CMPB    (R0),#'9
 4872  021606  101002                  BHI     TRYKWD
 4873  021610  000167  000730  NUMJMP: JMP     ISNUM
 4874
 4875  021614  012704  156542  TRYKWD: MOV     #KEYWDS-TRNPC,R4
 4876  021620  121027  000101          CMPB    (R0),#'A        ;CHECK ALPHA KEYWORDS
 4877  021624  103405                  BLO     COMTAB  ;TOO LOW
 4878  021626  121027  000132          CMPB    (R0),#'Z
 4879  021632  101002                  BHI     COMTAB  ;TOO HIGH
 4880  021634  012704  156634          MOV     #KEYA-TRNPC,R4  ;JUST RIGHT
 4881  021640  010703  COMTAB: MOV     PC,R3
 4882  021642  060403  TRNPC:  ADD     R4,R3           ;ADDRESS TABLE
 4883  021644  010002  TRYAGN: MOV     R0,R2
 4884  021646  122223  RETRY:  CMPB    (R2)+,(R3)+
 4885  021650  001776                  BEQ     RETRY
 4886  021652  114304                  MOVB    -(R3),R4
 4887  021654  100414                  BMI     AMATCH
 4888  021656  001550                  BEQ     NOTKWD
 4889  021660  005302                  DEC     R2
 4890  021662  122227  000040          CMPB    (R2)+,#BL
 4891  021666  001767                  BEQ     RETRY
 4892  021670  005302                  DEC     R2
 4893  021672  122327  000040          CMPB    (R3)+,#BL
 4894  021676  001763                  BEQ     RETRY
 4895  021700  105723                  TSTB    (R3)+
```

281

```
 4896  021702  100376                        BPL     .-2
 4897  021704  000757                        BR      TRYAGN
 4898  021706  005302                AMATCH: DEC     R2
 4899  021710  010200                        MOV     R2,R0
 4900  021712  110421                        MOVB    R4,(R1)+
 4901  021714  120427  000220                CMPB    R4,#'M
 4902  021720  001540                        BEQ     REMPACK
 4903  021722  120427  000262                CMPB    R4,#'N
 4904  021726  001461                        BEQ     TRNFN
 4905  021730  120427  000211                CMPB    R4,#,NEXT
 4906  021734  001505                        BEQ     NEXFILL
 4907  021736  120427  000257                CMPB    R4,#,SQUOT
 4908  021742  001403                        BEQ     QUOTPAK
 4909  021744  120427  000256                CMPB    R4,#,DQUOT
 4910  021750  001273                        BNE     TRANLUP
 4911  021752  122704  000257        QUOTPAK:CMPB    #,SQUOT,R4
 4912  021756  001402                        BEQ     .+6
 4913  021760  012704  000252                MOV     #'*-''+,SQUOT,R4
 4914  021764  062704  177570                ADD     #''-,SQUOT,R4
 4915  021770  112721  000377                MOVB    #,TEXT,(R1)+
 4916  021774  020100                        CMP     R1,R0
 4917  021776  103076                        BHIS    ERRTRN
 4918  022000  121004                STOSTR: CMPB    (R0),R4         ;SAME AS INITIAL QUOTE CHAR,?
 4919  022002  001405                        BEQ     ENDQUOT
 4920  022004  121027  000015                CMPB    (R0),#CR
 4921  022010  001415                        BEQ     ENDCR
 4922  022012  112021                        MOVB    (R0)+,(R1)+
 4923  022014  000771                        BR      STOSTR
 4924  022016  105021                ENDQUOT:CLRB    (R1)+
 4925  022020  005200                        INC     R0
 4926  022022  120427  000047                CMPB    R4,#''
 4927  022026  001403                        BEQ     ENDSQUO
 4928  022030  112721  000256                MOVB    #,DQUOTE,(R1)+
 4929  022034  000641                        BR      TRANLUP
 4930  022036  112721  000257        ENDSQUO:MOVB    #,SQUOTE,(R1)+
 4931  022042  000636                        BR      TRANLUP
 4932  022044  105021                ENDCR:  CLRB    (R1)+
 4933  022046  112711  000201                MOVB    #,EOL,(R1)
 4934  022052  010102                        MOV     R1,R2
 4935  022054  124227  000377                CMPB    -(R2),#,TEXT
 4936  022060  001375                        BNE     .-4
 4937  022062  110412                        MOVB    R4,(R2)
 4938  022064  112742  000377                MOVB    #,TEXT,-(R2)
 4939  022070  000207                        RTS     PC
 4940  022072  122027  000040        TRNFN:  CMPB    (R0)+,#BL
 4941  022076  001775                        BEQ     .-4
 4942  022100  124027  000101                CMPB    -(R0),#'A
 4943  022104  103411                        BLO     TRFNBAD
 4944  022106  121027  000132                CMPB    (R0),#'Z
 4945  022112  101006                        BHI     TRFNBAD
 4946  022114  112002                        MOVB    (R0)+,R2
 4947  022116  006302                        ASL     R2
 4948  022120  162702  000200                SUB     #'A+'A-2,R2
 4949  022124  110221                        MOVB    R2,(R1)+
 4950  022126  000604                        BR      TRANLUP
```

282


```
 4951  022130  112740  000116        TRFNBAD:MOVB    #'N,-(R0)
 4952  022134  112740  000106                MOVB    #'F,-(R0)
 4953  022140  005301                        DEC     R1
 4954  022142  020100                        CMP     R1,R0
 4955  022144  101013                        BHI     ERRTRN
 4956  022146  000425                        BR      REMPACK
 4957  022150  020100        NEXFILL:CMP     R1,R0
 4958  022152  103010                        BHIS    ERRTRN
 4959  022154  105021                        CLRB    (R1)+
 4960  022156  105011                        CLRB    (R1)
 4961  022160  062701  000011                ADD     #11,R1
 4962  022164  020100                        CMP     R1,R0
 4963  022166  103002                        BHIS    ERRTRN
 4964  022170  000167  177344                JMP     TRANLUP
 4965  022174  000167  000364        ERRTRN: JMP     ERRTR7
 4966  022200  121027  000132        NOTKWD: CMPB    (R0),#'Z
 4967  022204  101006                        BHI     REMPACK
 4968  022206  121027  000101                CMPB    (R0),#'A
 4969  022212  103403                        BLO     REMPACK
 4970  022214  005765  000056                TST     T1(R5)          ;2 VARS IN A ROW IS AN ERROR SO DONT
 4971  022220  001415                        BEQ     ISVAR           ;WASTE SYMBOL SPACE ON IT,
 4972  022222  112721  000377        REMPACK:MOVB    #,TEXT,(R1)+
 4973  022226  020100                        CMP     R1,R0
 4974  022230  103361                        BHIS    ERRTRN
 4975  022232  121027  000015        STOTXT: CMPB    (R0),#CR
 4976  022236  001402                        BEQ     CARRET
 4977  022240  112021                        MOVB    (R0)+,(R1)+
 4978  022242  000773                        BR      STOTXT
 4979  022244  105021                CARRET: CLRB    (R1)+
 4980  022246  112711  000201                MOVB    #,EOL,(R1)
 4981  022252  000207                        RTS     PC
 4982  022254  112002                ISVAR:  MOVB    (R0)+,R2
 4983  022256  122027  000040                CMPB    (R0)+,#BL
 4984  022262  001775                        BEQ     .-4
 4985  022264  124027  000071                CMPB    -(R0),#'9
 4986  022270  101006                        BHI     NODIGIT
 4987  022272  121027  000060                CMPB    (R0),#'0
 4988  022276  103403                        BLO     NODIGIT
 4989  022300  000302                        SWAB    R2
 4990  022302  152002                        BISB    (R0)+,R2
 4991  022304  000302                        SWAB    R2
 4992  022306  011503                NODIGIT:MOV     (R5),R3
 4993  022310  122027  000040                CMPB    (R0)+,#BL
 4994  022314  001775                        BEQ     .-4
 4995  022316  005300                        DEC     R0
 4996  022320  020365  000014        VARSRCH:CMP     R3,LOFREE(R5)
 4997  022324  103033                        BHIS    PUTITIN
 4998  022326  021327  177775                CMP     (R3),#,SCALAR
 4999  022332  103003                        BHIS    NOTIT
 5000  022334  062703  000004                ADD     #4,R3
 5001  022340  000767                        BR      VARSRCH
 5002  022342  062703  000010        NOTIT:  ADD     #10,R3
 5003
 5004                                        .IFNDF  $NOSTM
 5005  022346  026327  177770  177777        CMP     -10(R3),#,SVAR
```

283

```
 5006  022354  001436                        BEQ     TRYSVAR
 5007                                        ,ENDC
 5008
 5009  022356  022302                        CMP     (R3)+,R2
 5010  022360  001357                        BNE     VARSRCH
 5011  022362  121027  000044                CMPB    (R0),#'$
 5012  022366  001754                        BEQ     VARSRCH
 5013
 5014                                        ,IFNDF  $NOSTR
 5015  022370  000406                        BR      FOUNDV
 5016  022372  022302              TRYSVAR:  CMP     (R3)+,R2
 5017  022374  001351                        BNE     VARSRCH
 5018  022376  121027  000044                CMPB    (R0),#'$
 5019  022402  001346                        BNE     VARSRCH
 5020  022404  005200                        INC     R0
 5021                                        ,ENDC
 5022
 5023  022406  162703  000012          FOUNDV: SUB     #12,R3
 5024  022412  000441                        BR      RETVAR
 5025                                  PUTITIN:
 5026                                        ,IFNDF  $NOSTR
 5027  022414  062703  000014                ADD     #14,R3
 5028  022420  020365  000072                CMP     R3,LOSTR(R5)    ;MAKE SURE STRINGS ARE NOT THERE
 5029  022424  103405                        BLO     PUTAOK
 5030  022426  004767  000526                JSR     PC,UPPACK       ;IF THEY ARE, PACK THEM UPWARDS
 5031  022432  020365  000072                CMP     R3,LOSTR(R5)    ;AND TRY AGAIN
 5032  022436  103060                        BHIS    ERROV2          ;NO GOOD
 5033  022440  005743          PUTAOK: TST     -(R3)           ;R3 IS END OF ENTRY
 5034                                        ,ENDC
 5035
 5036                                        ,IFDF   $NOSTR
 5037                                        ADD     #12,R3          ;ADDRESS END OF ENTRY IN R3
 5038                                        ,ENDC
 5039
 5040  022442  020365  000012                CMP     R3,HIFREE(R5)
 5041  022446  101054                        BHI     ERROV2
 5042  022450  010365  000014                MOV     R3,LOFREE(R5)
 5043  022454  010243                        MOV     R2,-(R3)
 5044  022456  005043                        CLR     -(R3)
 5045  022460  005043                        CLR     -(R3)
 5046  022462  005043                        CLR     -(R3)
 5047  022464  012743  177775                MOV     #,SCALAR,-(R3)
 5048
 5049                                        ,IFNDF  $NOSTR
 5050  022470  121027  000044                CMPB    (R0),#'$
 5051  022474  001010                        BNE     RETVAR
 5052  022476  005200                        INC     R0
 5053  022500  012723  177777                MOV     #,SVAR,(R3)+
 5054  022504  005123                        COM     (R3)+
 5055  022506  005123                        COM     (R3)+
 5056  022510  005113                        COM     (R3)
 5057  022512  162703  000000                SUB     #0,R3
 5058                                        ,ENDC
 5059
 5060  022516  010565  000056          RETVAR: MOV     R5,T1(R5)       ;SOMETHING NONZERO,
```

284


```
 5061  022522  161503          RETLNO: SUB     (R5),R3
 5062  022524  000303                  SWAB    R3
 5063  022526  110321                  MOVB    R3,(R1)+
 5064  022530  000303                  SWAB    R3
 5065  022532  110321                  MOVB    R3,(R1)+
 5066  022534  020100                  CMP     R1,R0
 5067  022536  101216                  BHI     ERRTRN
 5068  022540  000167  177000          JMP     TRANVAR
 5069                                  ISNUM:
 5070  022544  010046                  MOV     R0,-(SP)        ;SAVE PTR IN CASE OF BAD NUM
 5071  022546  004767  000602          JSR     PC,VAL          ;FIND MANTISSA AND EXPONENT
 5072  022552  121027  000056          CMPB    (R0),#'.
 5073  022556  001016                  BNE     EXPTEN
 5074  022560  012600                  MOV     (SP)+,R0
 5075  022562  000617                  BR      REMPACK
 5076  022564  004167  174654          JSR     R1,MSGERR
 5077                                  ,IFNDF  $LONGER
 5078  022570  046124     124          ,ASCII  \TLT\
 5079                                  ,ENDC
 5080                                  ,IFDF   $LONGER
 5081                                  ,ASCII  'LINE TOO LONG TO TRANSLATE'
 5082                                  ,ENDC
 5083  022573     000                  ,BYTE 0
 5084                                  ,EVEN
 5085  022574  000167  156406          JMP     READY2
 5086  022600  004167  174640  ERROV2: JSR     R1,MSGERR
 5087                                  ,IFNDF  $LONGER
 5088  022604  052120     102          ,ASCII  \PTB\
 5089                                  ,ENDC
 5090                                  ,IFDF   $LONGER
 5091                                  ,ASCII  'PROGRAM TOO BIG'
 5092                                  ,ENDC
 5093  022607     000                  ,BYTE 0
 5094                                  ,EVEN
 5095  022610  000167  156372          JMP     READY2
 5096  022614  020327  014630  EXPTEN: CMP     R3,#14630
 5097  022620  101140                  BHI     MAKFLIT
 5098  022622  005702                  TST     R2
 5099  022624  003404                  BLE     EXPNEG
 5100  022626  004767  174562          JSR     PC,MPYTEN
 5101  022632  005302                  DEC     R2
 5102  022634  000767                  BR      EXPTEN
 5103  022636  001131          EXPNEG: BNE     MAKFLIT
 5104  022640  005703                  TST     R3
 5105  022642  001127                  BNE     MAKFLIT
 5106  022644  020427  177775          CMP     R4,#,SCALAR
 5107  022650  103124                  BHIS    MAKFLIT
 5108  022652  016503  000020          MOV     LINE(R5),R3
 5109  022656  005301                  DEC     R1
 5110  022660  020103                  CMP     R1,R3
 5111  022662  103452                  BLO     MAKLNO
 5112  022664  121127  000204          CMPB    (R1),#,GOSUB
 5113  022670  001447                  BEQ     MAKLNO
 5114  022672  121127  000205          CMPB    (R1),#,GOTO
 5115  022676  001444                  BEQ     MAKLNO
```

285

```
 5116  022700  121127  000224                  CMPB    (R1),#;CALL
 5117  022704  001441                          BEQ     MAKLNO
 5118  022706  121127  000242                  CMPB    (R1),#;THEN
 5119  022712  001436                          BEQ     MAKLNO
 5120  022714  121127  000300                  CMPB    (R1),#;LIST
 5121  022720  001433                          BEQ     MAKLNO
 5122  022722  005201                          INC     R1
 5123  022724  005003                          CLR     R3
 5124  022726  005704                          TST     R4
 5125  022730  002474                          BLT     MAKFLIT
 5126  022732  020427  000377                  CMP     R4,#377
 5127  022736  003003                          BGT     MAKE12
 5128  022740  112721  000375          LITZERO:MOVB    #;ILIT1,(R1)+
 5129  022744  000407                          BR      MAKEOK
 5130  022746  112721  000376          MAKE12: MOVB    #;ILIT2,(R1)+
 5131  022752  000304          LITCOMN:SWAB    R4
 5132  022754  020100                          CMP     R1,R0
 5133  022756  103010                          BHIS    TRNER4
 5134  022760  110421                          MOVB    R4,(R1)+
 5135  022762  000304                          SWAB    R4
 5136  022764  110421          MAKEOK: MOVB    R4,(R1)+
 5137  022766  005726                          TST     (SP)+
 5138  022770  020100                          CMP     R1,R0
 5139  022772  101002                          BHI     TRNER4
 5140  022774  000167  176540                  JMP     TRANLUP
 5141  023000  000167  177170          TRNER4: JMP     ERRTRN
 5142  023004  000167  177570          ERROV3: JMP     ERROV2
 5143  023010  005201          MAKLNO: INC     R1
 5144  023012  011503                          MOV     (R5),R3
 5145  023014  020365  000014          LNOSRCH:CMP     R3,LOFREE(R5)
 5146  023020  103013                          BHIS    STOLNO
 5147  023022  021327  177775                  CMP     (R3),#;SCALAR
 5148  023026  103403                          BLO     ISLNO
 5149  023030  062703  000012                  ADD     #12,R3
 5150  023034  000767                          BR      LNOSRCH
 5151  023036  021304          ISLNO:  CMP     (R3),R4
 5152  023040  001425                          BEQ     FOUNDLN
 5153  023042  062703  000004                  ADD     #4,R3
 5154  023046  000762                          BR      LNOSRCH
 5155                          STOLNO:
 5156                                  ;IFDF   $NOSTK
 5157                                  ADD     #4,R3
 5158                                  ;ENDC
 5159
 5160                                  ;IFNDF  $NOSTK
 5161  023050  062703  000006                  ADD     #6,R3           ;CHECK THAT THE SYMTAB SPACE
 5162  023054  020365  000072                  CMP     R3,LOSTR(R5)    ;IS NOT OCCUPIED BY STRINGS
 5163  023060  103405                          BLO     STOAOK
 5164  023062  004767  000072                  JSR     PC,UPPACK       ;IF IT IS, MOVE STRINGS UP
 5165  023066  020365  000072                  CMP     R3,LOSTR(R5)    ;AND CHECK ROOM ENOUGH AGAIN
 5166  023072  103344                          BHIS    ERROV3
 5167  023074  005743          STOAOK: TST     -(R3)           ;ADJUST SYMTAB ADDRESS
 5168                                  ;ENDC
 5169
 5170  023076  020365  000012                  CMP     R3,HIFREE(R5)
```

286


```
 5171  023102  101340                          BHI     ERROV3
 5172  023104  010365  000014                  MOV     R3,LOFREE(R5)
 5173  023110  005043                          CLR     -(R3)
 5174  023112  010443                          MOV     R4,-(R3)
 5175  023114  005726          FOUNDLN:TST     (SP)+
 5176  023116  000167  177400                  JMP     RETLNO
 5177  023122  005703          MAKFLIT:        TST     R3
 5178  023124  001002                          BNE     CALNRM
 5179  023126  005704                          TST     R4
 5180  023130  001703                          BEQ     LITZERO
 5181  023132  004767  174374          CALNRM: JSR     PC,NORM
 5182  023136  112721  000374                  MOVB    #;FLIT,(R1)+
 5183  023142  000303                          SWAB    R3
 5184  023144  020100                          CMP     R1,R0
 5185  023146  103314                          BHIS    TRNER4
 5186  023150  110321                          MOVB    R3,(R1)+
 5187  023152  000303                          SWAB    R3
 5188  023154  110321                          MOVB    R3,(R1)+
 5189  023156  000675                          BR      LITCOMN
 5190
 5191
```

287

```
                                .IFNDF  $NOSTR
                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                        ;  SUBROUTINE 'UPPACK'  CALLED BY JSR PC
                        ;                       PACKS STRING STORAGE TOWARD HIGH CORE
                        ;                       R0  UNUSED
                        ;                       R1,R2,R3  PRESERVED
                        ;                       R4  UNUSED
                        ;                       R5  MUST POINT TO USER AREA
                        ;                       SP  GOES 10 DEEPER AFTER JSR
023160  010146          UPPACK: MOV     R1,-(SP)
023162  010246                  MOV     R2,-(SP)
023164  010346                  MOV     R3,-(SP)
023166  005046                  CLR     -(SP)
023170  016501  000074          MOV     HISTR(R5),R1
023174  016502  000012          MOV     HIFREE(R5),R2
023200  010265  000074          MOV     R2,HISTR(R5)
023204  005016          UPPLOOP:CLR     (SP)            ;GET END OF THE NEXT STRING
023206  154116                  BISB    -(R1),(SP)
023210  001012                  BNE     UPPNZRO         ;(LAST BYTE CONTAINS THE LENGTH)
023212  020165  000072  UPPBAD: CMP     R1,LOSTR(R5)
023216  101372                  BHI     UPPLOOP
023220  010265  000072          MOV     R2,LOSTR(R5)
023224  005726                  TST     (SP)+
023226  012603                  MOV     (SP)+,R3
023230  012602                  MOV     (SP)+,R2
023232  012601                  MOV     (SP)+,R1
023234  000207                  RTS     PC
023236  161601          UPPNZRO:SUB     (SP),R1         ;ADDRESS BACK PTR
023240  005003                  CLR     R3
023242  154103                  BISB    -(R1),R3
023244  000303                  SWAB    R3
023246  154103                  BISB    -(R1),R3        ;GET IT IN R3
023250  000303                  SWAB    R3
023252  005301                  DEC     R1
023254  030327  000001          BIT     R3,#1           ;CHECK REL TO SYMBOLS
023260  001402                  BEQ     .+6
023262  005303                  DEC     R3
023264  061503                  ADD     (R5),R3         ;YES, ADD BASE OF SYM TAB
023266  020365  000004          CMP     R3,POL(R5)      ;MAKE SURE IT'S NOT TOO HI
023272  103347                  BHIS    UPPBAD
023274  020306                  CMP     R3,SP           ;IN STACK IS OK
023276  103013                  BHIS    UPPGOOD
023300  020365  000010          CMP     R3,ARRAYS(R5)   ;IN ARRAYS IS GOOD
023304  101342                  BHI     UPPBAD
023306  020365  000012          CMP     R3,HIFREE(R5)   ;IN FREE STORAGE IS BAD
023312  101005                  BHI     UPPGOOD
023314  020365  000014          CMP     R3,LOFREE(R5)
023320  103334                  BHIS    UPPBAD
023322  020315                  CMP     R3,(R5)
023324  103732                  BLO     UPPBAD          ;BELOW SYMBOL TABLE IS BAD
023326  062716  000004  UPPGOOD:ADD     #4,(SP)         ;GOOD STRING, MOVE IT UP
023332  021301                  CMP     (R3),R1
023334  001326                  BNE     UPPBAD          ;(DON'T GARBAGE COLLECT IT)
023336  061601                  ADD     (SP),R1
023340  160113                  SUB     R1,(R3)
```

288


```
023342  060213                  ADD     R2,(R3)
023344  114142                  MOVB    -(R1),-(R2)
023346  005316                  DEC     (SP)
023350  003375                  BGT     .-4
023352  000717                  BR      UPPBAD
                                .ENDC
```

289

```
                                ;-----------------------------------------------------------
                                ;  SUBROUTINE 'VAL'     CALLED BY JSR R7
                                ;                       CONVERTS AN ASCII STRING AT (R0)
                                ;                       TO A VALUE IN R3,R4, AND
                                ;                       AN EXPONENT IN R2
                                VAL:
023354  005002                          CLR     R2              ;DEC PLACES+100000 OR TRAILING ZEROES,
023356  005003                          CLR     R3              ;HIGH ORDER OF 32 BIT INTEGER,
023360  005004                          CLR     R4              ;LOW ORDER OF 32 BIT INTEGER,
023362  122027  000040          NUDIGIT:CMPB    (R0)+,#BL
023366  001775                          BEQ     .-4
023370  124027  000060                  CMPB    -(R0),#'0
023374  103427                          BLO     NOTDIG
023376  121027  000071                  CMPB    (R0),#'9
023402  101024                          BHI     NOTDIG
023404  020327  014630                  CMP     R3,#14630       ;IF HIGH WORD GREATER THAN THIS
023410  101405                          BLOS    CANFIT          ;THEN CANT FIT ANOTHER DIGIT IN 32 BITS,
023412  005200                          INC     R0
023414  005702                          TST     R2              ;CANT FIT DIGITS IN MANTISSA, BIT ITS
023416  002761                          BLT     NUDIGIT         ;AFTER POINT SO NEEDNT COUNT THEM,
023420  005202                          INC     R2              ;CANT FIT DIGITS IN MANTISSA SO MUST
023422  000757                          BR      NUDIGIT         ;KEEP TRACK OF TRAILING ZEROES,
023424  004767  173764          CANFIT: JSR     PC,MPYTEN
023430  005046                          CLR     -(SP)
023432  112016                          MOVB    (R0)+,(SP)      ;AND ADD IN THE DIGIT,
023434  162716  000060                  SUB     #'0,(SP)
023440  062604                          ADD     (SP)+,R4
023442  005503                          ADC     R3
023444  005702                          TST     R2
023446  001745                          BEQ     NUDIGIT         ;FITS IN MANTISSA,
023450  005202                          INC     R2              ;FITS IN MANTISSA BUT AFTER POINT SO
023452  000743                          BR      NUDIGIT         ;COUNT DECIMAL PLACES,
023454  122027  000056          NOTDIG: CMPB    (R0)+,#'.
023460  001024                          BNE     NOTDOT
023462  005702                          TST     R2
023464  001003                          BNE     DOTROT
023466  012702  100000                  MOV     #100000,R2      ;DOT COMES AFTER SHORT NUMBER SO GET
023472  000733                          BR      NUDIGIT         ;READY TO COUNT DECIMAL PLACES,
023474  003000                  DOTROT: BGT     DOTIGNO
023476  122027  000040          DOTIGNO:CMPB    (R0)+,#BL
023502  001775                          BEQ     .-4
023504  124027  000060                  CMPB    -(R0),#'0       ;DOT COMES AFTER FULL MANTISSA WITH
023510  103405                          BLO     PASTDOT         ;TRAILING ZEROES SO IGNORE MORE DIGITS,
023512  121027  000071                  CMPB    (R0),#'9
023516  101002                          BHI     PASTDOT
023520  005200                          INC     R0
023522  000765                          BR      DOTIGNO
023524  122027  000056          PASTDOT:CMPB    (R0)+,#'.       ;DOT COMES WHILE IGNORING TRAILING
023530  001400                          BEQ     DOTBAD          ;AFTER POINT WITH FULL MANTISSA:ERROR,
                                DOTBAD:
023532  005046                  NOTDOT: CLR     -(SP)
023534  124027  000105                  CMPB    -(R0),#'E
023540  001056                          BNE     NOEXPON         ;NO 'E' AFTER THE NUMBER
023542  020165  000020                  CMP     R1,LINE(R5)     ;IF ITS THE FIRST THING ON THE LINE IT
023546  001453                          BEQ     NOEXPON         ;MUST BE A LINENO SO 'E' IS ILLEGAL,
```

290


```
023550  005200                          INC     R0
023552  121027  000040                  CMPB    (R0),#BL
023556  001774                          BEQ     .-6
023560  122027  000053                  CMPB    (R0)+,#'+
023564  001406                          BEQ     EXPDIG
023566  124027  000055                  CMPB    -(R0),#'-
023572  001003                          BNE     EXPDIG
023574  005200                          INC     R0
023576  012716  100000                  MOV     #100000,(SP)
023602  122027  000040          EXPDIG: CMPB    (R0)+,#BL
023606  001775                          BEQ     .-4
023610  124027  000060                  CMPB    -(R0),#'0
023614  103423                          BLO     EXPDUN
023616  121027  000071                  CMPB    (R0),#'9
023622  101020                          BHI     EXPDUN
023624  011646                          MOV     (SP),-(SP)
023626  006316                          ASL     (SP)
023630  006316                          ASL     (SP)
023632  066616  000002                  ADD     2(SP),(SP)
023636  006316                          ASL     (SP)
023640  042766  077777  000002          BIC     #77777,2(SP)
023646  062616                          ADD     (SP)+,(SP)
023650  005046                          CLR     -(SP)
023652  112016                          MOVB    (R0)+,(SP)
023654  062616                          ADD     (SP)+,(SP)
023656  162716  000060                  SUB     #'0,(SP)
023662  000747                          BR      EXPDIG
023664  005716                  EXPDUN: TST     (SP)
023666  002003                          BGE     NOEXPON
023670  042716  100000                  BIC     #100000,(SP)
023674  005416                          NEG     (SP)
023676  005702                  NOEXPON:TST     R2
023700  002003                          BGE     EXPOK
023702  042702  100000                  BIC     #100000,R2
023706  005402                          NEG     R2
023710  062602                  EXPOK:  ADD     (SP)+,R2
023712  000207                          RTS     PC
```

291

```
                                        ;
                                        ; SYSTEM I/O BUFFERS AND HEADERS
                                        ;


                                                .IFNDF  $NOPTP
                                        ;
                                        ; PAPER TAPE READER
                                        ;
023714  023730                          PRBFHD: +       PRBUF   ;BSTRT
023716  023757                                  +       PRBEND          ;BEND
023720  023730                                  .WORD   PRBUF           ;BGET1
023722  000000                                  +       0               ;BGET2
023724  023730                                  .WORD   PRBUF           ;BPUT
023726  000000                                  .WORD   0               ;BFSPEC
        023730                          PRBUF   =.
                                        .       =.+$PRBSZ
        023760                          PRBEND  =.-1
                                                .EVEN
                                        ;
                                        ; PAPER TAPE PUNCH
                                        ;
023760  023774                          PPBFHD: +       PPBUF           ;BSTRT
023762  024023                                  +       PPBEND          ;BEND
023764  023774                                  .WORD   PPBUF           ;BGET1
023766  000000                                  +       0               ;BGET2
023770  023774                                  .WORD   PPBUF           ;BPUT
023772  000000                                  .WORD   0               ;BFSPEC
        023774                          PPBUF   =.
                                        .       =.+$PPBSZ
        024023                          PPBEND  =.-1
                                                .EVEN
                                                .ENDC


                                                .IFNDF  $NOLPI
                                        ;
                                        ; LINE PRINTER
                                        ;
024024  024040                          LPBFHD: +       LPBUF           ;BSTRT
024026  024077                                  +       LPBEND          ;BEND
024030  024040                                  .WORD   LPBUF           ;BGET1
024032  000000                                  +       0               ;BGET2
024034  024040                                  .WORD   LPBUF           ;BPUT
024036  000000                                  .WORD   0               ;BFSPEC
        024040                          LPBUF   =.
                                        .       =.+$LPBSZ
        024100                          LPBEND  =.-1
                                                .EVEN
                                                .ENDC
```

292


```
        024100                          ENDAB   =.
000000'                                         .CSECT
000000'                                 BEGIN   =.
        024100'                         .       =.+ENDAB
        000001                                  .END
```

293

```
ADDINT  007662        ADDOVF  007702        ADDPOS  007756        ADDSTK  007622
ADDZER  007742        ADSTK1  021464        ALLEXI  010242        ALLLOO  010032
ALLNOA  010042        ALLOC   010010        ALLOC1  002452        ALLOC2  002524
ALLSTR  010164        ALOG    = ******  G   AMATCH  021706        AMPWAI  013002
ARGB    010262  G     ARRAYS  = 000010  G   ASSIGN  003270        ASSSTR  003324
ATAN    = ******  G   ATNFN   013252        BEGIN   = 000000R       BEND    = 000002
BFSPEC  = 000012      BGET1   = 000004        BGET2   = 000006        BL      = 000040
BOMB    010320        BOMBOO  010372        BOMBJM  010504        BOMBNE  010440
BOMBNX  010360        BOMBOK  010470        BPDL    011422        BPUT    = 000010
BSTRT   = 000000      BUFCLR  010510        BUFEMP  015222        BUFTP   010564
CALL    003104        CALLCK  003136        CALLM1  003146        CALLM2  003170
CALLNM  003242        CALLX   003200        CALNRM  023132        CALOG   014212
CANFIT  023424        CARRET  022244        CATCOM  013104        CATLON  013054
CATNOT  013032        CCEXIT  010700        CCMES   007325        CEXP    014224
CHKCHR  010602        CHKISE  010782        CHKOSE  010710        CLEAR   001132
CLMNLP  000402        CLMNPP  000400        CLMNTT  = 000036      CLRECH  007164
CLRFRE  011070        CLRLOO  011020        CLROK   011000        CLRSVA  011052
CLRVAR  011002        CNCFLG  = 000106      CNOFLG  = 000107      CODE    = 000016
COLONC  005312        COLUMN  = 000034  G   COMCHK  010714        COMES   007330
COMTAB  021640        CONCAT  013020        COS     = ******  G   COSFN   013230
CR      = 000015      CRLFME  007322        DEFDUN  002400        DEFGOT  002642
DELMSG  007312        DEVERR  021854        DIMDON  002674        DIMGOT  002400
DIMLOO  002326        DIMNON  002346        DIMSCL  002594        DIVLOO  011154
DIVTEN  011150        DNPACK  011206        DNPBAD  011240        DNPGOO  011352
DNPIGN  011376        DNPLOO  011232        DNPNZR  011264        DOALLO  002536
DOCHAR  016174        DOITNO  012274        DOMSG   017504        DOPUT   007014
DOTBAD  023532        DOTIGN  023476        DOTROT  023474        ECHORA  007304
ECHOSP  = 000104      EDICHG  001456        EDICON  001540        EDIGRO  001624
EDIJMP  002036        EDIMME  001330        EDIMOD  001516        EDIMOV  001950
EDIMVU  001724        EDIOVE  001476        EDIPAS  001406        EDIPUT  001902
EDIRLC  001772        EDIRLO  002030        EDIROO  001712        EDISAM  001752
EDISKI  001342        EDISRC  001346        EDIT    001226        EDITL   001250
END     003340        ENDAB   = 024100        ENDCR   022044        ENDQUO  022016
ENDSQU  022036        EOF     003340        ERADD   013214        ERCHAN  010766
EREXP   013214        ERLOG   010312        ERPD02  012646        ERPD11  013314
ERRAG1  014514        ERRARA  010254        ERRARG  010312  G     ERRDAT  006526
ERRDEE  004134        ERRDEF  002634        ERRDIM  002626        ERRDIV  013976
ERREXP  014064        ERRFN1  003232        ERRFPU  = ******  G   ERRGO   004142
ERRMIX  013012  G     ERRMX1  014242        ERRMX2  012004        ERRMX3  015926
ERRMX4  003762        ERRMX5  004222        ERRMX6  021444        ERRMX7  021330
ERRMX8  013440        ERRMX9  013310        ERRNEX  005252        ERROVR  013214
ERROV1  001662        ERROV2  022600        ERROV3  023004        ERRPDL  011772  G
ERRPD2  013202        ERRPD3  003756        ERRPD4  014236        ERRPD5  012360
ERRRET  004226        ERRRUN  002726        ERRSOV  017272        ERRSS1  015932
ERRSS2  012116        ERRSS3  016576        ERRSTR  013206        ERRSXA  014246
ERRSX1  003236        ERRSX2  005260        ERRSX3  015522        ERRSX4  011714
ERRSX5  012544        ERRSX6  004150        ERRSX7  003624        ERRSX8  006250
ERRSX9  013470        ERRSYN  012544  G     ERRTRN  022174        ERRTR7  022564
ERRTR8  006352        ERRUFN  014520        ERR4WO  004512        ERSQR   010312
ERSTAR  013214        ERSX10  004674        ERSX12  010702        EVAL    011402  G
EXECUT  002746        EXIT02  007416        EXP     = ******  G   EXPDIG  023602
EXPDUN  023664        EXPFN   013260        EXPNEG  022636        EXPOK   023710
EXPTEN  022614        FAC1    = 000040  G   FAC2    = 000042  G   FF      = 000014
FILLCH  = 000112      FILLCO  = 000110  G   FILLNO  = 000111        FIXCLZ  015030
FIXRET  015032        FIXTST  015010        FIXUP   014754        FLE     015056
```

294


```
FLINE   015034        FLIT    011650        FLX     015002        FNCHK   014716
FNFN    014252        FNNEWN  014462        FNNOCO  014526        FNNUME  014750
FNOLNN  014452        FNRCOM  014700        FNREPL  014502        FNRLUP  014602
FNRSNU  014714        FNRSSC  014642        FNRSTR  014636        FNSTR   014362
FNSWAP  014276        FOR     004234        FORGO   004730        FORGOG  004726
FORGTL  004666        FORLOO  004420        FORLTL  004700        FORNEX  004920
FORNOL  004444        FORONE  004404        FORSKI  004414        FORZER  004704
FOUNDL  023114        FOUNDV  022406        FPPRES  015066        FPPSAV  015114
FRSTOU  017510        FUNCTI  013320        FUNCTJ  013332        FUNOK   013364
FUNRET  013416        GCWAIT  015350        GETBYT  015154        GETCHA  015230
GETCH1  015232        GETVAR  015364  G     GETX    015246        GET1BY  015142
GET2BY  015150        GOEXEC  006230        GONOSA  004126        GOSUB   004050
GOTECH  007222        GOTO    004050        GOTOFN  012000        GOTONE  016204
GOTOQT  011516        GOTVAL  012640        GSBCTR  = 000032      HORSTU  020750
HIFREE  = 000012      HISTR   = 000074        IDEV    = 000077        IF      003402
IFCOMP  003630        IFLEND  003536        IFLEQR  003560        IFLGTR  003772
IFLLTR  003712        IFLOOP  003506        IFNUME  003640        IFSEQ   003554
IFSGT   003766        IGNORE  002742        ILITCO  011606        ILIT1   011630
ILIT2   011650        IMMED   002042        IMMSTM  002124        INITBF  015536
INITSC  015554        INPEND  006204        INPEOL  006350        INPGOO  006176
INPLOO  005736        INPNEW  005764        INPNGU  006234        INPNNU  006320
INPNUL  006342        INPOK   006112        INPPC   005732        INPRTR  005754
INPSTO  006152        INPSTR  006254        INPUT   005650        INPYES  005672
INPY01  005730        INT     015574  G     INTEXT  021132        INTFN   013444
INTRTS  015726        INT1    015664        INT1A   015676        INT10   015714
INT2    015716        INT2A   015722        INT5    015730        INT6    015750
INT7    016010        INT8    016022        INT9    016002        IOINIT  001222
IOWAIT  016102        ISLNO   023036        ISNUM   022544        ISVAR   022254
ITAB    015360        ITABLE  010777        ITSIN   021010        JMPRNY  003376
KBCO    006774        KBWD    = 000102      KBIDUN  007000        KBINT   006722
KEYA    000476        KEYWDS  000464        LET     003256        LEVFLT  016332
LEVFNE  016374        LEVLIT  016306        LEVPLU  016304        LEVPOS  016402
LEV1LT  016350        LEV2LT  016344        LF      = 000012      LIMIT   = 000002  G
LINA    016140        LINDUN  017516        LINE    = 000020      LINEIN  016160
LINENO  = 000030      LINGET  016134        LINRUB  016104        LIST    002272
LISTSV  002262        LITCOM  022752        LITDIV  017642        LITEVA  016254
LITNOR  017552        LITOK   017600        LITSWR  017652        LITSTO  017656
LITTST  017570        LITZER  022740        LNOSRC  023014        LOCALO  016424
LOCGET  016410        LOCLOO  016524        LOCNO2  016552        LOCSS1  016462
LOFREE  = 000014      LOGFN   013274        LOSTR   = 000072      LPAR    011700
LPB     = 177516      LPBEND  = 024077        LPBFWD  024024        LPBUF   = 024040
LPERR   007546        LPINT   007510        LPOFF   007534        LPS     = 177514
LPSTRN  011720        LSTBSL  016776        LSTCHA  017002        LSTCHK  016676
LSTCRL  016744        LSTEOL  016732        LSTFLB  017110        LSTFLI  017070
LSTFN   016712        LSTILB  017062        LSTIL1  017040        LSTIL2  017052
LSTJMP  017114        LSTKWD  016664        LSTLIN  016756        LSTLNO  017152
LSTLNX  017202        LSTLOO  016610        LSTNEX  016724        LSTPC   016646
LSTPRO  016604        LSTPTR  017120        LSTSPE  017010        LSTSRC  016652
LSTTEX  017026        LSTVAR  017214        MAKCHK  017300        MAKEI2  022746
MAKEOK  022764        MAKEST  017252  G     MAKETR  017370        MAKFLI  023122
MAKGOT  017306        MAKLNO  023010        MINUS   012304        MPYTEN  017414
MSG     017452  G     MSGCOM  017454        MSGERR  017444        MSGODE  017500
MSGSUP  007200        NEXEND  005176        NEXFIL  022150        NEXGO   005214
NEXGTL  005160        NEXLTL  005170        NEXT    004792        NOCHAR  015252
NOCHR2  007502        NOCNC   002764        NODIGI  022306        NOEXPO  023676
```

295

```
NOQM     006000        NOQM1    006014        NOQUOT   011522        NORM     017532 G
NORNDQ   020274        NOROOM   020700        NOSUBS   015520        NOTDIG   023454
NOTDOT   023532        NOTIT    022342        NOTKWD   022200        NOTNOW   012330
NOTSYM   003062        NOWRAP   020646        NOWRP    015214        NUDIGI   023362
NUMALN   020224        NUMBIG   020206        NUMDIG   020300        NUMDIV   020214
NUMEXI   020562        NUMFIX   020002        NUMFLT   020106        NUMFM1   020362
NUMFM2   020470        NUMFM3   020504        NUMJMP   021610        NUMLP1   020400
NUMLP3   020540        NUMNEG   020030        NUMNOR   020140        NUMOK    020232
NUMOUT   017724 G      NUMPOS   020036        NUMPRS   020050        NUMSGN   017736 G
NUMSHF   020132        NUMSTT   020052        ODEV   = 000076        OFFTYP   007212
OLD      002172        OLD001   002230        OPERAN   011424        OPRATO   012166 G
OPRFN    013464        OTABLE   010774        OUT010   021034        OUT012   010760
PASTOO   023524        PC     =%000007        POL    = 000004 G      POSIZE = 000006 G
PLUS     012430        PNOGO    021156        POP      016000        POWDWN   007562
POWLP    007574        POWUP    007572        PP8    = 177556        PPBEND = 024023
PPBFWD   023760        PPBUF  = 023774        PPERR    007474        PPINT    007436
PPS    = 177554        PRB    = 177552        PRBEND = 023757        PRBFWD   023714
PRBUF  = 023730        PREC2    012266        PRECJ    012200        PREC4    012252
PREC5    012244        PREQT    007426        PRIBOT   005470        PRICM    005462
PRICOM   005512        PRIJMP   005606        PRILOO   005432        PRIMOR   005504
PRINCR   005632        PRINT    005264        PRISEM   005570        PRISTR   005414
PRITES   005600        PRNTOE   007270        PRNT01   005326        PRS    = 177550
PR3    = 000140        PR4    = 000200        PR7    = 000340        PS     = 177776
PTRINT   007336        PUSH     016046        PUSHF    012530        PUSH1    016072
PUTAOK   022440        PUTBYT   020614        PUTCCO   020734        PUTCHA   020700
PUTIT    020764        PUTITI   022414        QUNULL   012636        QUOTPU   013102
QUOTE    012552        QUOTPA   021752        READ     006356        READRA   006934
READDU   006456        READGO   006424        READOU   006522        READOT   006614
READST   006714        READY    001144        READY0   001140        READY2   001206
REAFIN   006552        REANUL   006710        REASRC   006546        REENAB   015326
REMPAC   022222        RESREG   021114        RESTOR   004036        RETLNO   022922
RETRY    021646        RETURN   004154        RETVAR   022516        REVRSE   014170
RNDCT  = 000070        RNDGOT   002600        RND1   = 000004 G      RND2   = 000006 G
RUN      002316        R0     =%000000        R0SAVE = 000044        R1     =%000001
R1SAVE = 000046        R2     =%000002        R2SAVE = 000050        R3     =%000003
R3SAVE = 000052        R4     =%000004        R4SAVE = 000054        R5     =%000005
SAVCHA   020570 G      SAVE     002234        SAVREG   021072        SAVRGI   021062
SCANPN   021134        SCRATC   001122        SETLFE   007276        SIN    = ****** G
SINFN    013222        SKIPEO   021170        SKIPHI   021234        SKIPRT   021252
SKIPTX   021244        SKIP01   021166        SKIP02   021164        SKIP05   021160
SLASH    012466        SOPRAT   012720 G      SOPRX    012776        SP     =%000006
SPARE  = 000113        SPECME   007134        SQRFN    013230        SQRT   = ****** G
SS1SAV = 000024        SS2SAV = 000026        STAR     012436        START    001102 G
STAR1    012462        STOAOK   023074        STOCOM   021316        STOLNO   023050
STOMMC   021402        STONOS   021306        STOP     003302        STOSSN   021366
STOSTR   022000        STOSVA   021334        STOSX    021442        STOTXT   022232
STOVAR   021254 G      STOVCO   021424        STOVTA   021412        STPRO    013162 G
STRGAR   012652        STRGBT   012674        STRGJM   011706        STRGVA   012660
SUBINT   021470        SUBSTK   021450        SYMBOL = 000000        S,0121   007036
S,0102   007054        TAB    = 000011        TABLE1   003010        TABLE2   012312
TABLE4   002110        TABLE5   011562 G      TABL5  = 005406        TAB1     021040
TAB2     021046        TAB3     014250        TBL1EN   003000        TBL4EN   002122
TBL5EN = 011626 G      TERMIN   012354        TKB    = 177562        TKS    = 177560 G
TPB    = 177566 G      TPCN     007240        TPCO     007254        TPHD   = 000120
TPINT    007074        TPNXT    007122        TPS    = 177564        TRAN     021512
```

296


```
TRANLU   021540        TRANVA   021544        TRFNBA   022130        TRNER4   023000
TRNFN    022072        TRNPC    021642        TRYAGN   021644        TRYECH   007166
TRYKWD   021614        TRYNUM   021566        TRYSVA   022372        TSTSTK   012514
TSTS1    012520        TSTS2    012526        TYPE     007142        TYPE1    007156
T1     = 000056 G      T2     = 000060 G      T3     = 000062 G      UAASR    014072
UAJMP    014104        UANJMP   014100        UATSTA   014130        UATST0   014110
UATST2   014120        UA1      013520        UA10     013630        UA10A    013666
UA10B    013710        UA10C    013720        UA10,5   013732        UA12     013750
UA17     013760        UA17A    013772        UA17B    014010        UA17D    014020
UA17F    014040        UA19     014056        UA20     014136        UA21     014064
UA23     014146        UA4      013550        UA5      013556        UA8      013572
UA9      013616        UINTEG   012406        UMINUS   011414        UNARY    012370
UPARRO   013474        UPPACK   023160        UPPBAD   023212        UPPGOO   023326
UPPLOO   023204        UPPNER   023236        USRARE = ****** G      VAL      023354 G
VARBLE   011732        VARNOS   012156        VARSAV = 000022        VARSRC   022320
VARSS    012010        VERNUM   000040        VONESS   012122        VTWOSS   012134
WTRET    016120        XCHAR    015270        $ADR   = ****** G      $DVR   = ****** G
$ERVEC = ****** G      $IR    = ****** G      $LPBSZ = 000040        $MLR   = ****** G
$POLSH = ****** G      $PPBSZ = 000030        $PRBSZ = 000030        $PRORG = 000400
$SBR   = ****** G      $STKSZ = 000200 G      ,ABS   = 000273        ,AMPER = 000226
,ASC   = 000300        ,ATN   = 000270        ,CALL  = 000224        ,CHR$  = 000301
,CLEAR = 000313        ,COLON = 000260        ,COMMA = 000243 G      ,COS   = 000266
,DATA  = 000223        ,DEF   = 000221        ,DIM   = 000215        ,DQUOT = 000256
,EG    = 000247        ,EL    = 000245        ,EN    = 000251        ,END   = 000202
,EOF   = 000225        ,EOL   = 000201 G      ,EQ    = 000254        ,EXP   = 000271
,FLIT  = 000374        ,FN    = 000262        ,FOR   = 000203        ,GE    = 000246
,GOSUB = 000204        ,GOTO  = 000205        ,GT    = 000253        ,IF    = 000206
,ILIT1 = 000375        ,ILIT2 = 000376        ,INPUT = 000207        ,INT   = 000274
,LE    = 000244        ,LEN   = 000277        ,LET   = 000210        ,LIST  = 000306
,LOG   = 000272        ,LPAR  = 000255 G      ,LT    = 000252        ,MINUS = 000234
,NE    = 000250        ,NEXT  = 000211        ,NVAR  = 177776        ,OLD   = 000311
,PLUS  = 000232        ,POS   = 000302        ,POUND = 000261        ,PRINT = 000212
,RANDO = 000216        ,READ  = 000222        ,REM   = 000220        ,RESTO = 000217
,RETUR = 000213        ,RND   = 000264        ,RNDL  = 000263        ,RPAR  = 000237 G
,RUN   = 000307        ,SAVE  = 000310        ,SCALA = 177775        ,SCR   = 000312
,SEG   = 000303        ,SEMI  = 000236        ,SGN   = 000275        ,SIN   = 000265
,SLASH = 000231        ,SQR   = 000267        ,SQUOT = 000257        ,STAR  = 000230
,STEP  = 000241        ,STOP  = 000214        ,STR   = 000305        ,SVAR  = 177777
,TAB   = 000276        ,TERM  = 000235        ,TEXT  = 000377        ,THEN  = 000242
,TO    = 000240        ,UNARY = 000233        ,UPARR = 000227        ,VAL   = 000304
,      = 024100R
```

ERRORS DETECTED: 0

297

RUN-TIME: 69 SECONDS
CORE USED: 4K

```
ADDINT   2228   2237#
ADDOVF   2240   2243#
ADDPOS   2243   2256#
ADDSTK   1559   2226#  2970   4834
ADDZER   2244   2253#
ADSTK1   4834#  4836   4839
ALLEXI   2327   2341   2348#
ALLLOO   2281#  2288
ALLNOA   2282   2285#
ALLOC     969   2273#  4827
ALLOC1    938    944#
ALLOC2    954    960#
ALLSTR   2314   2328#
ALOG       60   3183   3394
AMATCH   4887   4898#
AMPWAI   3067   3083#
ARGB       76   2369#
ARRAYS     69    311#   312   2624   2626   2697   5234
ASSIGN   1070   1100#
ASSSTR   1166   1175#
ATAN       60   3178
ATNFN    2784   3178#
BEGIN    5401#
BEND      355#  3664   4573
BFSPEC    359#  2028   2030   2139   2143   2162   2182   3689   3693   3695   3724   4621   4623   4626
BGET1     356#  3653   4576
BGET2     357#  3655   4578
BL        295#  1235   1242   1667   4171   4409   4861   4890   4893   4940   4983   4993   5263   5293
          5310  5318
BOMB       208  2400#
BOMBDO   2414   2417#
BOMBJM   2418   2438#
BOMBNE   2424   2428#
BOMBNX   2413#  2416
BOMBOK   2427   2434#
BPOL     2737   2742#
BPUT      358#  3660   4571   4580   4581
BSTRT     354#  3666   4575
BUFCLR    640    650   2019   2447#
BUFEMP   3661   3670#
BUFTP    2025   2453   2476#
CALL     1064   1077#
CALLCK   1113#  1152
CALLM1   1118#  1121
CALLM2   1127#  1130
CALLNM   1119   1128   1148#
CALLX    1126   1132#
CALNRM   5178   5181#
CALOG    3379   3394#
CANFIT   5270   5276#
CARRET   4976   4979#
CATCOM   3105   3136#
CATLON   3101   3106#
```

300

301

302

303

304

305

306

307

308

309

	847	866	867	870	1082	1083	1085	1086	1116	1118	1149	1215	1217	1222
	1225	1231	1237	1247	1253	1255	1257	1263	1280	1289	1291	1293	1303	1309
	1311	1313	1659	1674	1676	1681	1683	1685	1963	1969	1973	1979	1981	1982
	1983	2013	2014	2016	2021	2030	2060	2066	2069	2072	2091	2095	2176	2130
	2131	2133	2135	2139	2160	2179	2199	2199	2273	2279	2283	2285	2290	2298
	2301	2310	2338	2348	2350	2413	2423	2425	2428	2430	2431	2497	2499	2501
	2503	2506	2509	2511	2514	2516	2535	2537	2544	2548	2549	2551	2600	2601
	2603	2607	2611	2612	2616	2619	2620	2621	2623	2624	2630	2631	2632	2633
	2895	2900	3070	3073	3074	3076	3078	3080	3102	3104	3106	3107	3113	3121
	3122	3124	3128	3129	3132	3135	3137	3138	3140	3216	3218	3272	3276	3288
	3289	3290	3291	3295	3300	3320	3321	3323	3324	3385	3390	3411	3413	3420
	3421	3424	3444	3445	3448	3480	3482	3494	3496	3497	3499	3502	3511	3512
	3513	3529	3530	3541	3542	3545	3548	3550	3551	3592	3608	3609	3622	3632
	3662	3693	3698	3699	3703	3806	3807	3808	3809	3816	3819	3820	3821	3823
	3826	3834	3838	3891	3855	3857	3861	3923	3924	3925	3936	3970	3977	3991
	4031	4051	4052	4053	4060	4062	4096	4098	4170	4188	4189	4110	4129	4137
	4171	4173	4175	4185	4219	4220	4223	4226	4228	4229	4230	4231	4232	4233
	4234	4235	4241	4242	4243	4244	4245	4289	4292	4299	4384	4387	4471	4489
	4417	4420	4423	4424	4425	4426	4427	4430	4435	4445	4449	4454	4457	4469
	4479	4488	4490	4493	4497	4499	4502	4505	4508	4511	4512	4515	4517	4520
	4521	4533	4535	4547	4580	4686	4690	4767	4792	4802	4803	4806	4807	4810
	4812	4813	4815	4856	4857	4861	4863	4867	4869	4871	4876	4878	4883	4899
	4916	4918	4920	4922	4925	4940	4942	4944	4946	4951	4952	4954	4957	4962
	4966	4968	4973	4975	4977	4982	4983	4985	4987	4990	4993	4995	5011	5018
	5028	5050	5052	5066	5070	5072	5074	5132	5138	5184	5263	5265	5267	5271
	5278	5286	5293	5295	5297	5299	5301	5305	5309	5310	5312	5314	5316	5318
	5320	5322	5332											
R0SAVE	325#	326	3357	3363	3620	3632								
R1	268#	656	662	683	684	685	686	687	690	694	704	705	707	709
	711	719	721	723	725	729	734	740	742	743	748	752	758	782
	811	817	823	825	827	829	833	850	852	866	883	893	900	903
	910	911	914	917	926	929	932	934	937	939	941	944	945	947
	949	951	953	955	957	960	961	962	970	972	974	979	1000	1002
	1006	1008	1014	1015	1017	1037	1067	1135	1143	1155	1158	1161	1169	1175
	1183	1185	1191	1201	1203	1207	1246	1265	1267	1273	1284	1304	1323	1327
	1334	1336	1359	1370	1388	1390	1393	1397	1408	1420	1422	1430	1432	1436
	1438	1440	1445	1447	1449	1451	1452	1454	1456	1469	1470	1472	1474	1478
	1480	1481	1482	1483	1484	1485	1486	1487	1488	1489	1490	1506	1509	1510
	1514	1515	1516	1523	1525	1530	1531	1534	1536	1537	1539	1551	1552	1553
	1554	1555	1562	1563	1564	1565	1574	1575	1576	1579	1580	1581	1599	1601
	1606	1608	1610	1612	1614	1637	1660	1668	1672	1687	1692	1694	1695	1697
	1738	1747	1749	1761	1764	1766	1770	1773	1781	1785	1789	1797	1799	1800
	1802	1806	1807	1809	1812	1836	1838	1845	1874	1886	1889	1919	1921	1952
	1953	1955	1956	2011	2028	2038	2075	2079	2127	2139	2143	2155	2162	2174
	2182	2274	2276	2277	2281	2300	2301	2303	2305	2307	2308	2310	2318	2319
	2323	2329	2333	2335	2336	2338	2339	2342	2343	2344	2345	2349	2400	2406
	2413	2419	2435	2665	2670	2671	2677	2681	2719	2710	2743	2805	2809	2810
	2811	2812	2820	2825	2831	2833	2865	2877	2879	2891	2898	2913	2955	3021
	3024	3025	3031	3033	3066	3083	3203	3210	3214	3223	3236	3386	3389	3425
	3486	3482	3497	3501	3502	3508	3510	3603	3606	3621	3633	3659	3660	3664
	3666	3689	3693	3695	3704	3735	3737	3749	3765	3819	3828	3829	3830	3831
	3833	3836	3844	3846	3847	3869	3924	3927	3928	3945	4082	4083	4104	4106
	4108	4112	4114	4118	4123	4124	4125	4137	4142	4144	4145	4148	4149	4150

310

	4151	4155	4292	4299	4303	4304	4305	4307	4372	4387	4405	4410	4421	4449
	4491	4494	4498	4504	4510	4513	4516	4518	4534	4536	4546	4547	4548	4571
	4573	4575	4576	4578	4580	4581	4611	4614	4615	4621	4623	4626	4685	4691
	4714	4715	4727	4728	4729	4730	4732	4735	4737	4739	4741	4744	4747	4853
	4854	4857	4858	4865	4900	4915	4916	4922	4924	4928	4930	4932	4933	4934
	4949	4953	4954	4957	4959	4960	4961	4962	4972	4973	4977	4979	4980	5063
	5065	5066	5076	5086	5109	5110	5112	5114	5116	5118	5120	5122	5128	5130
	5132	5134	5136	5138	5143	5182	5184	5186	5188	5201	5205	5209	5211	5217
	5219	5221	5223	5225	5243	5245	5246	5248	5307					
R1SAVE	326#	327	3621	3633										
R2	269#	731	738	745	750	755	756	757	759	762	764	765	767	768
	769	772	774	775	793	794	795	798	802	805	806	810	813	818
	821	825	828	829	830	831	833	834	851	852	853	854	855	857
	858	859	891	903	911	913	914	915	916	917	918	920	922	924
	929	931	932	933	964	967	1002	1003	1004	1008	1037	1039	1040	1041
	1043	1044	1066	1067	1068	1069	1071	1109	1111	1113	1118	1127	1133	1139
	1151	1155	1157	1158	1159	1216	1219	1221	1226	1229	1242	1248	1249	1251
	1284	1285	1287	1304	1305	1307	1325	1336	1390	1392	1393	1394	1395	1399
	1436	1439	1440	1441	1442	1444	1452	1455	1456	1457	1458	1470	1473	1474
	1475	1476	1522	1523	1524	1525	1528	1528	1537	1539	1541	1603	1605	1650
	1651	1654	1655	1659	1770	1772	1773	1774	1790	1793	1815	1818	1860	1863
	1866	1869	1870	1886	1888	1889	1890	1900	1903	1905	1966	1971	2238	2239
	2241	2245	2247	2248	2249	2257	2258	2259	2294	2313	2317	2318	2319	2320
	2322	2323	2324	2328	2329	2331	2332	2333	2334	2339	2340	2343	2345	2447
	2448	2449	2450	2451	2452	2456	2458	2462	2476	2477	2478	2479	2480	2542
	2546	2548	2551	2666	2669	2673	2675	2680	2684	2686	2687	2688	2706	2707
	2710	2714	2743	2745	2747	2749	2751	2753	2755	2757	2761	2763	2768	2769
	2770	2771	2773	2774	2775	2778	2830	2831	2832	2837	2841	2843	2867	2902
	2905	2911	2912	2920	2921	2923	2925	2927	2937	2938	2939	2940	2941	2942
	3023	3024	3029	3032	3037	3038	3052	3053	3055	3056	3072	3082	3099	3100
	3107	3116	3117	3120	3122	3126	3129	3130	3131	3136	3138	3139	3140	3289
	3293	3294	3298	3304	3306	3321	3373	3387	3392	3405	3408	3411	3421	3423
	3424	3425	3428	3432	3434	3437	3438	3439	3440	3445	3447	3448	3449	3450
	3452	3454	3455	3456	3458	3459	3460	3464	3465	3467	3468	3471	3474	3475
	3476	3477	3514	3515	3518	3522	3524	3525	3526	3528	3530	3531	3532	3536
	3537	3539	3540	3542	3547	3548	3549	3550	3577	3595	3603	3605	3606	3607
	3608	3622	3634	3636	3657	3658	3667	3670	3686	3689	3691	3699	3705	3706
	3709	3710	3712	3731	3732	3733	3734	3749	3750	3752	3787	3788	3789	3790
	3791	3792	3825	3837	3842	3854	3856	3861	3929	3930	3932	3976	3942	3943
	4011	4013	4015	4030	4031	4035	4037	4039	4056	4061	4062	4083	4085	4087
	4090	4092	4098	4114	4116	4122	4124	4126	4127	4132	4134	4154	4155	4156
	4157	4159	4161	4164	4166	4167	4173	4175	4183	4221	4222	4225	4331	4344
	4347	4414	4447	4452	4468	4483	4488	4500	4503	4506	4511	4525	4527	4529
	4530	4531	4568	4569	4570	4582	4585	4684	4692	4760	4763	4772	4774	4775
	4776	4789	4790	4797	4798	4800	4803	4807	4808	4809	4813	4814	4815	4852
	4853	4858	4883	4884	4889	4890	4892	4898	4899	4934	4935	4937	4938	4946
	4947	4948	4949	4982	4989	4990	4991	5009	5016	5043	5098	5171	5272	5276
	5287	5213	5216	5247	5248	5260	5272	5274	5282	5284	5288	5290	5340	5342
	5343	5344												
R2SAVE	327#	328	3566	3622	3634									
R3	270#	707	710	711	712	713	715	721	724	725	726	727	757	768
	759	760	769	770	775	783	784	785	786	787	788	793	803	805
	808	809	936	941	943	945	1085	1117	1120	1123	1124	1129	1148	1214

311

```
        1223  1225  1227  1229  1235  1329  1330  1331  1332  1334  1361  1362  1364  1365
        1366  1367  1368  1370  1433  1444  1491  1531  1533  1534  1535  1543  1544  1546
        1548  1549  1550  1653  1654  1661  1770  1777  1822  1825  1829  1831  1832  1834
        1859  1860  1861  1865  1892  1894  1896  1898  1900  1911  1913  1914  1916  1918
        1944  1945  1947  1948  1950  1953  1955  1960  1961  1963  1964  1966  1968  1972
        1977  1978  1980  1981  1983  2085  2220  2236  2242  2292  2255  2262  2275  2293
        2321  2322  2330  2331  2448  2450  2451  2452  2477  2479  2480  2505  2508  2513
        2518  2519  2544  2545  2549  2645  2647  2649  2652  2667  2679  2683  2684  2685
        2686  2689  2691  2692  2693  2695  2697  2699  2701  2703  2707  2709  2893  2899
        2913  2914  2916  2918  2929  2931  2933  2935  2954  3023  3033  3056  3057  3062
        3077  3078  3079  3080  3108  3112  3119  3123  3124  3125  3131  3200  3388  3391
        3455  3457  3462  3467  3468  3469  3472  3475  3477  3505  3507  3556  3558  3559
        3623  3635  3653  3655  3659  3660  3662  3663  3664  3666  3787  3789  3791  3937
        3973  3975  3978  3979  3981  3983  3980  3987  3988  3989  3990  4017  4023  4026
        4028  4033  4037  4041  4043  4046  4051  4093  4094  4095  4096  4223  4224  4225
        4229  4230  4233  4235  4236  4237  4242  4259  4262  4264  4266  4267  4269  4315
        4318  4321  4324  4334  4336  4338  4340  4349  4351  4359  4360  4361  4362  4415
        4423  4428  4429  4432  4433  4437  4439  4441  4443  4455  4459  4463  4464  4466
        4470  4477  4481  4482  4488  4493  4535  4571  4572  4573  4575  4576  4578  4581
        4683  4693  4769  4794  4801  4805  4810  4881  4882  4884  4886  4893  4895  4992
        4996  4998  5000  5002  5005  5009  5010  5023  5027  5028  5031  5033  5040  5042
        5043  5044  5045  5046  5047  5053  5054  5055  5056  5057  5061  5062  5063  5064
        5065  5096  5104  5108  5110  5123  5144  5145  5147  5149  5151  5153  5161  5162
        5165  5167  5170  5172  5173  5174  5177  5183  5186  5187  5188  5203  5215  5220
        5221  5222  5223  5224  5226  5228  5229  5230  5232  5234  5236  5238  5240  5243
        5246  5247  5261  5269  5281
R3SAVE   328#   329  3218  3394  3396  3623  3635
R4       271#   704   732   748   746   755   814   817   849   946   952   957   959   961
        1013  1136  1142  1205  1269  1798  1802  1803  1804  1811  2012  2032  2034  2036
        2039  2062  2063  2064  2066  2068  2074  2093  2097  2130  2133  2230  2276  2320
        2332  2648  2736  2741  2951  2973  2981  2992  2996  3027  3059  3149  3205  3211
        3220  3224  3250  3311  3326  3359  3360  3361  3362  3371  3374  3393  3409  3435
        3583  3591  3625  3637  3863  3881  3884  3887  4016  4024  4026  4029  4095  4100
        4260  4261  4263  4265  4268  4319  4323  4327  4335  4337  4339  4341  4350  4363
        4413  4418  4431  4438  4440  4442  4444  4456  4461  4467  4492  4495  4519  4523
        4525  4527  4528  4538  4607  4611  4612  4624  4682  4694  4875  4880  4892  4896
        4900  4901  4903  4905  4907  4909  4911  4913  4914  4918  4926  4937  5176  5124
        5126  5131  5134  5135  5136  5151  5174  5179  5262  5280
R4SAVE   329#   330  3024  3636
R5       272#   637   638   639   641   651   652   653   654   655   661   666   679   685
         691   694   700   712   726   730   731   744   745   760   768   770   772   778
         786   787   788   795   798   802   803   806   808   816   830   848   849   850
         867   871   872   878   910   915   916   933   977   978  1003  1010  1013  1014
        1033  1035  1068  1071  1137  1141  1159  1183  1271  1272  1316  1329  1331  1332
        1335  1361  1365  1367  1394  1399  1400  1416  1417  1418  1419  1433  1441  1457
        1458  1475  1476  1483  1484  1485  1488  1487  1488  1489  1490  1492  1494  1495
        1496  1507  1526  1535  1543  1549  1550  1556  1558  1560  1561  1569  1572  1582
        1583  1584  1605  1632  1656  1674  1698  1699  1700  1742  1747  1759  1760  1763
        1774  1779  1790  1799  1804  1807  1813  1815  1823  1824  1841  1843  1859  1890
        1892  1918  1924  1934  1944  2009  2010  2011  2018  2023  2055  2056  2057  2059
        2063  2069  2071  2075  2079  2126  2153  2154  2172  2173  2196  2198  2235  2239
        2241  2246  2247  2249  2250  2251  2253  2254  2256  2257  2259  2260  2261  2300
        2303  2308  2336  2348  2372  2374  2401  2402  2403  2404  2405  2417  2423  2425
```

312

```
        2426  2428  2429  2431  2432  2433  2447  2476  2540  2542  2545  2598  2599  2600
        2601  2624  2626  2627  2628  2629  2630  2633  2669  2670  2671  2675  2677  2692
        2693  2697  2699  2701  2703  2803  2804  2805  2807  2809  2810  2811  2812  2832
        2875  2881  2889  2893  2900  2911  2912  2950  2953  2961  2963  2965  2967  2968
        2993  2994  3216  3217  3222  3223  3255  3257  3259  3265  3281  3282  3286  3288
        3302  3305  3306  3315  3320  3357  3363  3366  3394  3396  3408  3425  3438  3440
        3449  3515  3531  3580  3582  3588  3607  3620  3621  3622  3623  3624  3632  3633
        3634  3635  3636  3705  3709  3731  3733  3734  3745  3747  3761  3763  3806  3808
        3809  3816  3818  3830  3840  3847  3848  3857  3858  3859  3871  3873  3875  3876
        3879  3880  3882  3883  3899  3900  3922  3923  3927  3929  3930  3943  3971  3972
        3987  3988  3989  3990  3993  3995  3997  4082  4126  4141  4142  4144  4145  4146
        4148  4149  4150  4151  4156  4161  4162  4164  4165  4167  4168  4169  4219  4241
        4245  4293  4294  4295  4296  4298  4300  4371  4386  4388  4391  4393  4395  4397
        4399  4400  4413  4415  4545  4546  4548  4549  4601  4603  4624  4647  4609  4614
        4625  4687  4688  4695  4713  4760  4767  4769  4775  4776  4789  4792  4794  4809
        4852  4856  4860  4970  4992  4996  5028  5031  5040  5042  5060  5061  5108  5144
        5145  5162  5165  5170  5172  5205  5200  5207  5211  5213  5229  5230  5234  5236
        5238  5240  5307
READ     1062  1886#  1920
READBA   1899  1910  1917  1934#  1962  1967
READDU   1913#  1988
READGO   1900#  1949
READOU   1924#  1951
READQT   1904  1906  1900#
READST   1984  1987#
READY     644   651#   897   906  1036  1196  2438  3902  4676
READY0    650#   680  2201
READY2    661#  1026  1758  5085  5095
REAFIN   1897  1945#  1957
REANUL   1975  1985#
REASRC   1895  1944#
REENAB   3694  3709#
REMPAC   4902  4956  4967  4969  4972#  5075
RESREG   4690#
RESTOR   1059  1316#
RETLNO   5061#  5176
RETRY    4884#  4885  4891  4894
RETURN   1055  1359#
RETVAR   5024  5051  5000#
REVRSE   3377  3385#
RND1       78   333#   334   977   978  2598
RND2       78   334#   335  2599
RNDCT    335#   336   977  3899  4625
RNDGOT   923   977#
RUN      861   909#
SAVCHA    77  4405  4544#
SAVE     862   889#
SAVREG  3818  3922  4371  4386  4545  4601  4682#
SAVRG1  2009  2055  2126  2153  2172  4680#
SCANPN   881   889  4713#
SCRATC   641#   864
SETLFE  2087  2103#
SIN       60  3171
```

313

314

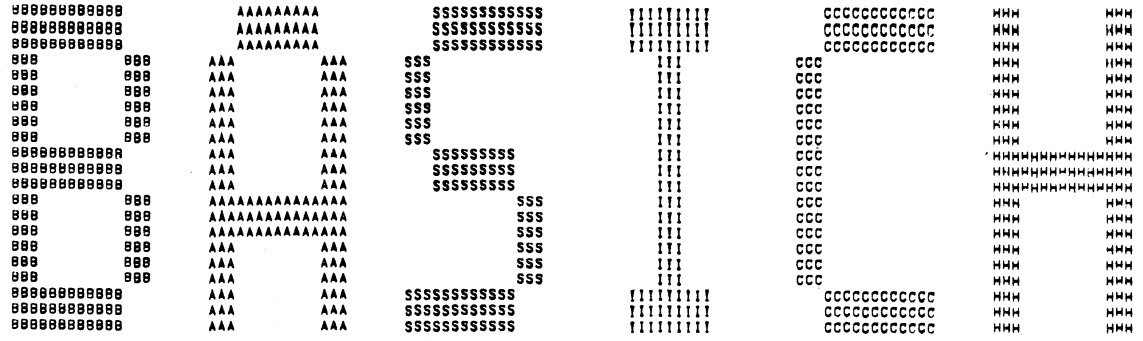
315

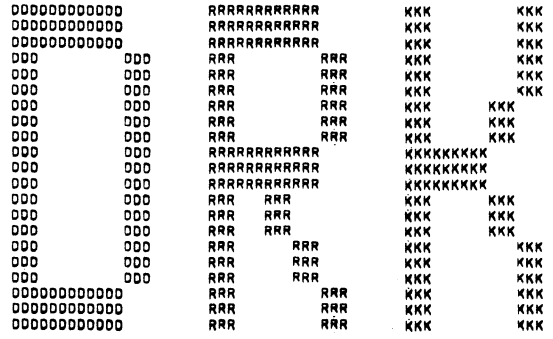
316

317

318

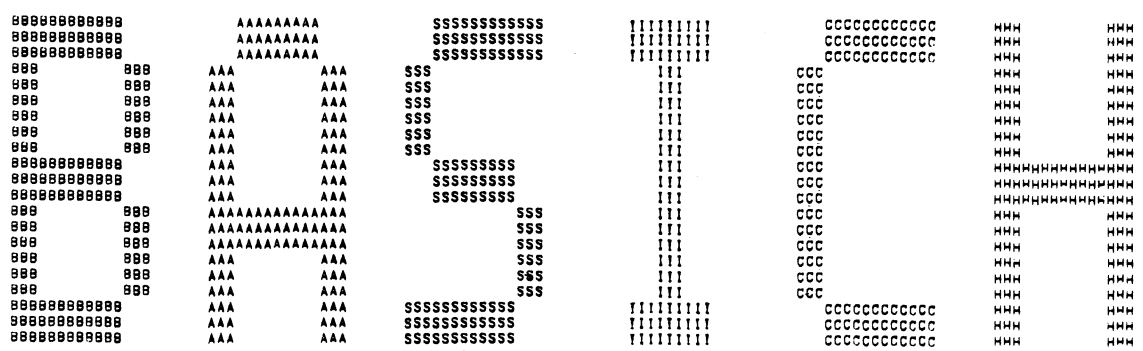
319

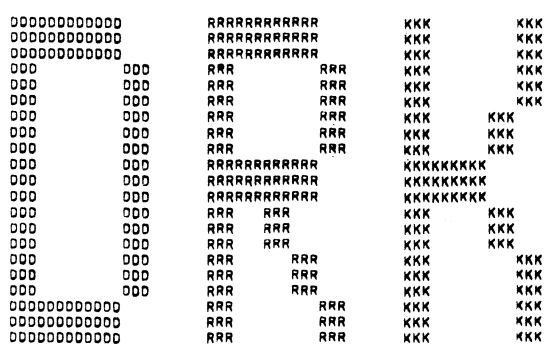




```
LPTSPL Version 4(136)  Running on LPT1
 •START• User BEAN,ROBERT [205,1065] Job  BASICL Seq, 4943 Date 23-Mar-73 00:20:51 Monitor     PF547G #40+135 •START• Note!BEAN
Request created: 22-Mar-73  23:57:28
File: DSK:BIBAS:CH,DRK[205,1065] Created: 22-Mar-73 00:00:00 <157>  Printed: 23-Mar-73 00:40:50
QUEUE Switches:  /PRINT:ARROW /FILE:ASCII /COPIES:2 /SPACING:1 /LIMIT:1746
File will be RENAMEd to <057> protection
```

320





```
LPTSPL Version 4(136)  Running on LPT1
 •START• User BEAN,ROBERT [205,1065] Job  BASICL Seq, 4943 Date 23-Mar-73 00:20:51 Monitor     PF547G #40+135 •START• Note!BEAN
Request created: 22-Mar-73  23:57:28
File: DSK:BIBAS:CH,DRK[205,1065] Created: 22-Mar-73 00:00:00 <157>  Printed: 23-Mar-73 00:40:54
QUEUE Switches:  /PRINT:ARROW /FILE:ASCII /COPIES:2 /SPACING:1 /LIMIT:1746
File will be RENAMEd to <057> protection
```

321

```
BASICH  MACX11   V021   22-MAR-73   15:45   PAGE 1

                        ;
                        ; BASIC/PTS  PART3-BASICH
                        ;
                        ; DEC-11-LPTBA-A-LA3
                        ;
                        ; COPYRIGHT 1973
                        ;
                        ; DIGITAL EQUIPMENT CORPORATION
                        ; MAYNARD, MASSACHUSETTS 01754
                        ;
```



```
BASICH  MACX11   V021   22-MAR-73   15:45   PAGE 2

                        .TITLE  BASICH V001A EDIT #020 (REF) 01/31/73
                        ;                               BASIC SOURCE FILE #15
                        ;
                        ;
                        ;
                        ;
                        ; USER AREA AND ONCE-ONLY INIT. CODE,
                        ;
                        ; TO BE LINKED LAST AFTER BASICL AND PDMP-11 TO FORM BASIC,
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
```

```
                                ;
                                ; GLOBALS
                                ;
                                        .GLOBL  TPB, TKS
                                        .GLOBL  START, FILLCO
                                        .GLOBL  USRAREA
                                        .GLOBL  LIMIT, FOL, ARRAYS, FOSIZE
                                        .GLOBL  TABLES, TBLSEND
                                        .GLOBL  COLUMN, FAC1, FAC2, .RPAR
                                        .GLOBL  ERRMIX, ERRPDL, ERRSYN, ERRARG
                                        .GLOBL  EVAL, INT, MAKEST, OPRATO, SOPRAT
                                        .GLOBL  .COMMA, T1, T2, T3
                                        .GLOBL  ARGB, SAVCHAR, NUMSGN, STPRO
                                        .GLOBL  NORM, VAL
                                        .GLOBL  RND1, RND2
                                        .GLOBL  $STKS$ ,IFPMP
                                ;
                                ; ASSEMBLY PARAMETERS
                                ;
                                        .IFNDF  $TPBS$  ;TELEPRINTER BUFFER SIZE
                                STPBSZ  =20             ;CHARACTERS
                                        .ENDC
                                ;
                                        .IFNDF  $KBBS$  ;KEYBOARD BUFF. SIZE
                                $KBBSZ  =20             ;CHARACTERS
                                        .ENDC
                                ;
                                        .IFNDF  $ULNSP  ;USER LINE SPACE
                                $ULNSP  =120            ;BYTES
                                        .ENDC
```

324


```
000000'                         .CSECT
000000                  R0      =%0
000001                  R1      =%1
000002                  R2      =%2
000003                  R3      =%3
000004                  R4      =%4
000005                  R5      =%5
000006                  SP      =%6
000007                  PC      =%7
000040                  BL      =40

000000' 000000                  USRAREA:.WORD   0
                                BASICH:
                                ;RND FUNCTION -- GENERATE RANDOM NUMBER
                                ;       SCAN PAST ARG IF PRESENT
000002' 004737  000000G RNDLFN: JSR     PC,@#EVAL
000006' 103452                  BCS     ERRNDA
000010' 122127  000000G         CMPB    (R1)+,#.RPAR
000014' 001045                  BNE     ERRNDS
                        RNDFN:
000016' 010500          RNDFN2: MOV     R5,R0
000020' 062700  000000G         ADD     #RND2,R0
000024' 011003                  MOV     @R0,R3
000026' 014002                  MOV     -(R0),R2
000030' 006303                  ASL     R3              ;MULT BY 2
000032' 006102                  ROL     R2
000034' 062002                  ADD     (R0)+,R2        ;NOW BY 3
000036' 061003                  ADD     @R0,R3
000040' 005502                  ADC     R2
000042' 061002                  ADD     @R0,R2          ;NOW BY 2**16+3
000044' 100002                  BPL     RPLUS
000046' 062702  100000          ADD     #100000,R2      ;GET 2**32+G
000052' 010310          RPLUS:  MOV     R3,@R0          ;STORE NEW GENERATORS
000054' 010240                  MOV     R2,-(R0)
000056' 012700  000201          MOV     #201,R0         ;INITIAL EXPONENT
000062' 006303          RNORM:  ASL     R3              ;FLOAT RESULT
000064' 006102                  ROL     R2
000066' 103402                  BCS     REXP            ;JUMP WHEN LEADING BIT FOUND
000070' 005300                  DEC     R0              ;ADJUST EXPONENT FOR SHIFT
000072' 000773                  BR      RNORM
000074' 105003          REXP:   CLRB    R3
000076' 150203                  BISB    R2,R3
000100' 000303                  SWAB    R3
000102' 105002                  CLRB    R2
000104' 150002                  BISB    R0,R2           ;INSERT EXPONENT INTO RESULT
000106' 000302                  SWAB    R2
000110' 006002                  ROR     R2
000112' 006003                  ROR     R3              ;INSERT + SIGN
000114' 010265  000000G         MOV     R2,FAC1(R5)
000120' 010365  000000G         MOV     R3,FAC2(R5)
000124' 000137  000000G         JMP     @#OPRATOR
000130' 000137  000000G ERRNDS: JMP     @#ERRSYN
000134' 000137  000000G ERRNDA: JMP     @#ERRARG
                        RNDFNE:
```

325

```
                                        ;ABS FUNCTION ROUTINE
000140' 004737 000000G          ABSFN:  JSR     PC,@#EVAL
000144' 103430                          BCS     ERABSA
000146' 122127 000000G                  CMPB    (R1)+,#,RPAR
000152' 001027                          BNE     ERABSS
000154' 005765 000000G                  TST     FAC1(R5)
000160' 001405                          BEQ     ABSINT
000162' 100017                          BPL     ABSX
000164' 042765 100000 000000G           BIC     #100000,FAC1(R5)
000172' 000413                          BR      ABSX
000174' 005765 000000G          ABSINT: TST     FAC2(R5)
000200' 100010                          BPL     ABSX
000202' 005465 000000G                  NEG     FAC2(R5)
000206' 102005                          BVC     ABSX
000210' 012765 044000 000000G           MOV     #44000,FAC1(R5)
000216' 005065 000000G                  CLR     FAC2(R5)
000222' 000137 000000G          ABSX:   JMP     @#OPRATOR
000226' 000137 000000G          ERABSA: JMP     @#ERRARG
000232' 000137 000000G          ERABSS: JMP     @#ERRSYN
                                ABSFNE:
```

326


```
                                        ;SGN FUNCTION ROUTINE
000236' 004737 000000G          SGNFN:  JSR     PC,@#EVAL
000242' 103424                          BCS     ERSGNA
000244' 122127 000000G                  CMPB    (R1)+,#,RPAR
000250' 001023                          BNE     ERSGNS
000252' 005000                          CLR     R0
000254' 005765 000000G                  TST     FAC1(R5)
000260' 001003                          BNE     SGNFLT
000262' 005765 000000G                  TST     FAC2(R5)
000266' 001410                          BEQ     SGNX
000270' 100002                  SGNFLT: BPL     SGNPOS
000272' 005300                          DEC     R0
000274' 000401                          BR      .+4
000276' 005200                  SGNPOS: INC     R0
000300' 010065 000000G                  MOV     R0,FAC2(R5)
000304' 005065 000000G                  CLR     FAC1(R5)
000310' 000137 000000G          SGNX:   JMP     @#OPRATOR
000314' 000137 000000G          ERSGNA: JMP     @#ERRARG
000320' 000137 000000G          ERSGNS: JMP     @#ERRSYN
                                SGNFNE:
```

327

```
                                ;TAB FUNCTION ROUTINE
000324' 004737 000000G          TABFN:  JSR     PC,@#ARGB
000330' 122127 000000G                  CMPB    (R1)+,#,RPAR
000334' 001036                          BNE     ERTABS
000336' 005046                          CLR     -(SP)
000340' 116516 000000G                  MOVB    FAC2(R5),(SP)
000344' 021627 000110           TABB:   CMP     (SP),#72.
000350' 103403                          BLO     TABC
000352' 162716 000110                   SUB     #72.,(SP)
000356' 000772                          BR      TABB
000360' 167516 000000G          TABC:   SUB     @COLUMN(R5),(SP)
000364' 100001                          BPL     .+4
000366' 005016                          CLR     (SP)
000370' 001002                          BNE     TABNON
000372' 005316                          DEC     (SP)
000374' 000414                          BR      TABNULL
000376' 010502                  TABNON: MOV     R5,R2
000400' 004737 000000G                  JSR     PC,@#MAKESTR
000404' 011600                          MOV     (SP),R0
000406' 005002                          CLR     R2
000410' 151002                          BISB    (R0),R2
000412' 062700 000003                   ADD     #3,R0
000416' 112720 000040                   MOVB    #BL,(R0)+
000422' 005302                          DEC     R2
000424' 003374                          BGT     .-6
000426' 000137 000000G          TABNULL:JMP     @#SOPRATR
000432' 000137 000000G          ERTABS: JMP     @#ERRSYN
                                TABFNE:
```

328


```
                                ; LEN FUNCTION ROUTINE
000436' 004737 000000G          LENFN:  JSR     PC,@#EVAL
000442' 103016                          BCC     ERLENA
000444' 122127 000000G                  CMPB    (R1)+,#,RPAR
000450' 001015                          BNE     ERLENS
000452' 005065 000000G                  CLR     FAC1(R5)
000456' 005065 000000G                  CLR     FAC2(R5)
000462' 012602                          MOV     (SP)+,R2
000464' 005202                          INC     R2
000466' 001402                          BEQ     .+6
000470' 114265 000000G                  MOVB    -(R2),FAC2(R5)
000474' 000137 000000G                  JMP     @#OPRATOR
000500' 000137 000000G          ERLENA: JMP     @#ERRARG
000504' 000137 000000G          ERLENS: JMP     @#ERRSYN
                                LENFNE:
```

329

```
                                        ; ASC FUNCTION ROUTINE
000510' 004737 000000G          ASCFN:  JSR     PC,@#EVAL
000514' 103023                          BCC     ERASCA
000516' 122127 000000G                  CMPB    (R1)+,#,RPAR
000522' 001022                          BNE     ERASCS
000524' 012602                          MOV     (SP)+,R2
000526' 020227 177777                   CMP     R2,#177777
000532' 001414                          BEQ     ERASCA
000534' 121227 000001                   CMPB    (R2),#1
000540' 001011                          BNE     ERASCA
000542' 005065 000000G                  CLR     FAC1(R5)
000546' 005065 000000G                  CLR     FAC2(R5)
000552' 116265 000003 000000G           MOVB    3(R2),FAC2(R5)
000560' 000137 000000G                  JMP     @#OPRATOR
000564' 000137 000000G          ERASCA: JMP     @#ERRARG
000570' 000137 000000G          ERASCS: JMP     @#ERRSYN
                                ASCFNE:
                                        ; CHR$ FUNCTION ROUTINE
000574' 004737 000000G          CHR$FN: JSR     PC,@#ARGB
000600' 122127 000000G                  CMPB    (R1)+,#,RPAR
000604' 001016                          BNE     ERCHR$
000606' 142765 000200 000000G           BICB    #200,FAC2(R5)   ;TAKE CHAR MOD 128
000614' 012746 000001                   MOV     #1,-(SP)
000620' 010502                          MOV     R5,R2
000622' 004737 000000G                  JSR     PC,@#MAKESTR
000626' 011600                          MOV     (SP),R0
000630' 116560 000000G 000003           MOVB    FAC2(R5),3(R0)
000636' 000137 000000G                  JMP     @#SOPRATR
000642' 000137 000000G          ERCHR$: JMP     @#ERRSYN
                                CHRFNE:
```



```
                                        ; POS FUNCTION ROUTINE
000646' 004737 000000G          POSFN:  JSR     PC,@#EVAL
000652' 103133                          BCC     ERPOSA
000654' 122127 000000G                  CMPB    (R1)+,#,COMMA
000660' 001132                          BNE     ERPOSS
000662' 004737 000000G                  JSR     PC,@#EVAL
000666' 103125                          BCC     ERPOSA
000670' 122127 000000G                  CMPB    (R1)+,#,COMMA
000674' 001124                          BNE     ERPOSS
000676' 004737 000000G                  JSR     PC,@#EVAL
000702' 103517                          BCS     ERPOSA
000704' 004737 000000G                  JSR     PC,@#INT
000710' 122127 000000G                  CMPB    (R1)+,#,RPAR
000714' 001114                          BNE     ERPOSS
000716' 005000                          CLR     R0
000720' 005765 000000G                  TST     FAC1(R5)
000724' 001077                          BNE     POSF
000726' 005765 000000G                  TST     FAC2(R5)
000732' 003474                          BLE     POSF
000734' 156500 000000G                  BISB    FAC2(R5),R0     ;R0 IS N
                                ;CHECK NULL X$
000740' 026627 000002 177777            CMP     2(SP),#177777
000746' 001002                          BNE     POSX
000750' 005000                          CLR     R0
000752' 000464                          BR      POSF
                                ;COMPUTE LENGTH OF X$
000754' 005002                  POSX:   CLR     R2
000756' 157602 000002                   BISB    @2(SP),R2
                                ;CHECK NULL Y$
000762' 021627 177777                   CMP     (SP),#177777
000766' 001004                          BNE     POSY
                                ;NULL Y$, ANS IS MIN(N,LEN(X$))
000770' 020002                          CMP     R0,R2
000772' 003454                          BLE     POSF
000774' 010200                          MOV     R2,R0
000776' 000452                          BR      POSF
                                ;SAVE ADDR OF END OF Y$ IN T2
001000' 005003                  POSY:   CLR     R3
001002' 157603 000000                   BISB    @(SP),R3
001006' 061603                          ADD     (SP),R3
001010' 062703 000003                   ADD     #3,R3
001014' 010365 000000G                  MOV     R3,T2(R5)
001020' 012603                          MOV     (SP)+,R3
001022' 062703 000003                   ADD     #3,R3
                                ;SAVE ADDR OF END OF X$ IN T1
001026' 061602                          ADD     (SP),R2
001030' 062702 000003                   ADD     #3,R2
001034' 010265 000000G                  MOV     R2,T1(R5)
001040' 012602                          MOV     (SP)+,R2
001042' 060002                          ADD     R0,R2
001044' 062702 000002                   ADD     #2,R2
                                ;FIND MATCHING FIRST CHARACTER
001050' 121322                  POSFST: CMPB    (R3),(R2)+
001052' 001406                          BEQ     POSREM
001054' 005200                  POSTRY: INC     R0
```

```
  298 001056' 020265  000000G            CMP     R2,T1(R5)
  299 001062' 001372                     BNE     POSFST
  300 001064' 005000                     CLR     R0
  301 001066' 000417                     BR      POSF2
  302                                ;CHECK THAT REMAINING CHARS MATCH
  303 001070' 010246                 POSREM: MOV     R2,-(SP)
  304 001072' 010346                     MOV     R3,-(SP)
  305 001074' 005203                     INC     R3
  306 001076' 020365  000000G        POSNXT: CMP     R3,T2(R5)
  307 001102' 001410                     BEQ     POSF
  308 001104' 020265  000000G            CMP     R2,T1(R5)
  309 001110' 001402                     BEQ     POSNO
  310 001112' 122223                     CMPB    (R2)+,(R3)+
  311 001114' 001770                     BEQ     POSNXT
  312 001116' 012603                 POSNO: MOV     (SP)+,R3
  313 001120' 012602                     MOV     (SP)+,R2
  314 001122' 000754                     BR      POSTRY
  315                                ;RETURN FROM FUNCTION
  316 001124' 022626                 POSF:  CMP     (SP)+,(SP)+
  317 001126' 010065  000000G        POSF2: MOV     R0,FAC2(R5)
  318 001132' 005065  000000G            CLR     FAC1(R5)
  319 001136' 000137  000000G            JMP     @#OPRATOR
  320 001142' 000137  000000G        ERPOSA: JMP     @#ERRARG
  321 001146' 000137  000000G        ERPOSS: JMP     @#ERRSYN
  322                                POSFNE:
  323
  324
```

332


```
  325                                ; SEG FUNCTION ROUTINE
  326 001152' 004737  000000G        SEGFN: JSR     PC,@#EVAL
  327 001156' 103114                     BCC     ERSEGA
  328 001160' 122127  000000G            CMPB    (R1)+,#,COMMA
  329 001164' 001113                     BNE     ERSEGS
  330 001166' 004737  000000G            JSR     PC,@#EVAL
  331 001172' 103506                     BCS     ERSEGA
  332 001174' 004737  000000G            JSR     PC,@#INT
  333 001200' 016546  000000G            MOV     FAC1(R5),-(SP)
  334 001204' 016546  000000G            MOV     FAC2(R5),-(SP)
  335 001210' 122127  000000G            CMPB    (R1)+,#,COMMA
  336 001214' 001077                     BNE     ERSEGS
  337 001216' 004737  000000G            JSR     PC,@#EVAL
  338 001222' 103472                     BCS     ERSEGA
  339 001224' 004737  000000G            JSR     PC,@#INT
  340 001230' 122127  000000G            CMPB    (R1)+,#,RPAR
  341 001234' 001067                     BNE     ERSEGS
  342                                ;GET X VALUE IN R2
  343 001236' 012602                     MOV     (SP)+,R2
  344 001240' 012600                     MOV     (SP)+,R0
  345 001242' 100403                     BMI     SEGX0
  346 001244' 001055                     BNE     SEGNUL
  347 001246' 005702                     TST     R2
  348 001250' 003002                     BGT     SEGL
  349 001252' 012702  000001         SEGX0: MOV     #1,R2
  350                                ;COMPUTE LENGTH OF A$ IN R0
  351 001256' 005000                 SEGL:  CLR     R0
  352 001260' 021627  177777             CMP     (SP),#177777    ;CHECK NULL STRING
  353 001264' 157600  000000             BISB    @(SP),R0
  354                                ;GET Y VALUE IN R3
  355 001270' 005765  000000G        SEGTY: TST     FAC1(R5)
  356 001274' 100441                     BMI     SEGNUL
  357 001276' 001402                     BEQ     SEGTY2
  358 001300' 010003                     MOV     R0,R3
  359 001302' 000403                     BR      SEGRNG
  360 001304' 016503  000000G        SEGTY2: MOV     FAC2(R5),R3
  361 001310' 003433                     BLE     SEGNUL
  362                                ;CHECK 0<=R2<=R3<=R0
  363 001312' 020003                 SEGRNG: CMP     R0,R3
  364 001314' 101001                     BHI     SEGR2
  365 001316' 010003                     MOV     R0,R3
  366 001320' 020203                 SEGR2: CMP     R2,R3
  367 001322' 101026                     BHI     SEGNUL
  368                                ;COMPUTE LENGTH OF OUTPUT STRING
  369 001324' 160203                     SUB     R2,R3
  370 001326' 005203                     INC     R3
  371                                ;MAKE NEW STRING OF THE CORRECT LENGTH
  372 001330' 010246                     MOV     R2,-(SP)        ;SAVE START CHAR
  373 001332' 010502                     MOV     R5,R2
  374 001334' 010346                     MOV     R3,-(SP)
  375 001336' 004737  000000G            JSR     PC,@#MAKESTR
  376                                ;FIX STACK AND MOVE IN CHARS FROM OLD STRING
  377 001342' 012602                     MOV     (SP)+,R2        ;NEW STRING POINTER
  378 001344' 012600                     MOV     (SP)+,R0        ;START CHAR
  379 001346' 061600                     ADD     (SP),R0         ;ADD OLD STR POINTER
```

333

```
   380  001350' 010216                          MOV     R2,(SP)         ;SAVE NEW STRING
   381  001352' 005003                          CLR     R3
   382  001354' 151203                          BISB    (R2),R3         ;R3 IS NEW STRING LENGTH
   383  001356' 062702  000003                  ADD     #3,R2           ;ADDR FIRST CHAR IN NEW STR
   384  001362' 062700  000002                  ADD     #2,R0           ;ADDR START CHAR IN OLD STR
   385  001366' 112022                  SEGLP:  MOVB    (R0)+,(R2)+     ;FILL IN NEW STRING
   386  001370' 005303                          DEC     R3
   387  001372' 001375                          BNE     SEGLP
   388  001374' 000137  000000G                 JMP     @#STPRO
   389                                  ;OUTPUT NULL STRING
   390  001400' 012716  177777          SEGNUL: MOV     #177777,(SP)
   391  001404' 000137  000000G         SEGX:   JMP     @#OPRATR
   392  001410' 000137  000000G         ERSEGA: JMP     @#ERRARG
   393  001414' 000137  000000G         ERSEGS: JMP     @#ERRSYN
   394                                  SEGFNE:
   395
   396
```

334


```
   397                                  ; VAL FUNCTION ROUTINE
   398  001420' 004737  000000G         VALFN:  JSR     PC,@#EVAL
   399  001424' 103056                          BCC     ERVALA
   400  001426' 122127  000000G                 CMPB    (R1)+,#,RPAR
   401  001432' 001051                          BNE     ERVALS
   402                                  ;READ STRING
   403  001434' 011600                          MOV     (SP),R0
   404  001436' 020027  177777                  CMP     R0,#177777      ;CHECK NULL STRING
   405  001442' 001006                          BNE     VALR
   406  001444' 005065  000000G                 CLR     FAC1(R5)
   407  001450' 005065  000000G                 CLR     FAC2(R5)
   408  001454' 005726                          TST     (SP)+
   409  001456' 000435                          BR      VALJ
   410  001460' 005002                  VALR:   CLR     R2
   411  001462' 151002                          BISB    (R0),R2
   412  001464' 010216                          MOV     R2,(SP)
   413  001466' 062700  000003                  ADD     #3,R0
   414  001472' 060002                          ADD     R0,R2
   415  001474' 105012                          CLRB    (R2)
   416  001476' 010446                          MOV     R4,-(SP)
   417  001500' 010146                          MOV     R1,-(SP)
   418  001502' 010246                          MOV     R2,-(SP)
   419                                  ;READ ASCII NUMBER AND EXPONENT
   420  001504' 004737  000000G                 JSR     PC,@#VAL
   421                                  ;NORMALIZE TO FORM FLT PT NUMBER
   422  001510' 004737  000000G                 JSR     PC,@#NORM
   423  001514' 010365  000000G                 MOV     R3,FAC1(R5)
   424  001520' 010465  000000G                 MOV     R4,FAC2(R5)
   425                                  ;CHECK END OF STRING
   426  001524' 105710                  VALCE:  TSTB    (R0)
   427  001526' 001403                          BEQ     VALX
   428  001530' 122027  000040                  CMPB    (R0)+,#BL
   429  001534' 001773                          BEQ     VALCE
   430                                  ;RESTORE REGS AND RETURN TO EVAL
   431  001536' 020026                  VALX:   CMP     R0,(SP)+
   432  001540' 001010                          BNE     ERVALA
   433  001542' 012601                          MOV     (SP)+,R1
   434  001544' 012604                          MOV     (SP)+,R4
   435  001546' 012602                          MOV     (SP)+,R2
   436  001550' 110218                          MOVB    R2,(R0)         ;RESTORE STRING LENGTH
   437  001552' 000137  000000G         VALJ:   JMP     @#OPRATOR
   438  001556' 000137  000000G         ERVALS: JMP     @#ERRSYN
   439  001562' 000137  000000G         ERVALA: JMP     @#ERRARG
   440                                  VALFNE:
   441
   442
```

335

```
                                        ; STR FUNCTION ROUTINE
001566' 004737 000000G          STRFN:  JSR     PC,@#EVAL
001572' 103432                          BCS     ERSTRA
001574' 122127 000000G                  CMPB    (R1)+,#,RPAR
001600' 001031                          BNE     ERSTRS
001602' 012746 000020                   MOV     #20,-(SP)
001606' 010502                          MOV     R5,R2
001610' 004737 0000000G                 JSR     PC,@#MAKESTR
001614' 011603                          MOV     (SP),R3
001616' 062703 000003                   ADD     #3,R3
001622' 010365 000000G                  MOV     R3,T2(R5)
001626' 005065 000000G                  CLR     T1(R5)
001632' 004737 000000G                  JSR     PC,@#NUMSGN
001636' 000000G                         ,WORD   SAVCHAR
001640' 010602                          MOV     SP,R2
001642' 016546 000000G                  MOV     T1(R5),-(SP)
001646' 004737 0000000G                 JSR     PC,@#MAKESTR
001652' 012616                          MOV     (SP)+,(SP)
001654' 000137 000000G                  JMP     @#STPRO         ;PROTECT STRING

001660' 000137 000000G          ERSTRA: JMP     @#ERRARG
001664' 000137 000000G          ERSTRS: JMP     @#ERRSYN
                                STRFNE:
                                FNEND:
```



```
                                        ;
                                        ;
                                        ; USER AREA STORAGE CELLS
                                        ;
001670' 000000                  USR:    ,WORD   0               ;SYMBOLS
001672' 000000                          ,WORD   0               ;LIMIT
001674' 000000                          ,WORD   0               ;PDL
001676' 000000                          ,WORD   0               ;PDSIZE
001700' 000000                          ,WORD   0               ;ARRAYS
001702' 000000                          ,WORD   0               ;HIFREE
001704' 000000                          ,WORD   0               ;LOFREE
001706' 002214'                         ,WORD   USRCODE         ;CODE
001710' 002074'                         ,WORD   USRLINE         ;LINE
001712' 000000                          ,WORD   0               ;VARSAV
001714' 000000                          ,WORD   0               ;SS1SAV
001716' 000000                          ,WORD   0               ;SS2SAV
001720' 000000                          ,WORD   0               ;LINENO
001722' 000000                          ,WORD   0               ;GSBCTR
001724' 000000                          ,WORD   0               ;COLUMN
001726' 000000                          ,WORD   0               ;CLMNTTY
001730' 000000                          ,WORD   0               ;FAC1
001732' 000000                          ,WORD   0               ;FAC2
001734' 000000                          ,WORD   0               ;R0SAVE
001736' 000000                          ,WORD   0               ;R1SAVE
001740' 000000                          ,WORD   0               ;R2SAVE
001742' 000000                          ,WORD   0               ;R3SAVE
001744' 000000                          ,WORD   0               ;R4SAVE
001746' 000000                          ,WORD   0               ;T1
001750' 000000                          ,WORD   0               ;T2
001752' 000000                          ,WORD   0               ;T3
001754' 000000                          ,WORD   0               ;RND1
001756' 000000                          ,WORD   0               ;RND2
001760' 000000                  RAND:   ,WORD   0               ;RNDCT
001762' 000000                          ,WORD   0               ;LOSTR
001764' 000000                          ,WORD   0               ;HISTR
001766'     000                         ,BYTE   0               ;ODEV
001767'     000                         ,BYTE   0               ;IDEV
001770' 002040'                         .       TPBFH1          ;TPHD
001772' 002034'                         .       KBBFH1          ;KBHD
001774' 000000                          ,WORD   0               ;ECHOSP
001776'     000                         ,BYTE   0               ;CNCFLG
001777'     000                         ,BYTE   0               ;CNOFLG
002000'     000                         ,BYTE   0               ;FILLCO
002001'     000                         ,BYTE   0               ;FILLNO
002002'     000                         ,BYTE   0               ;FILLCH
002003'     000                         ,BYTE   0               ; [SPARE BYTE]
```

```
                                        ;
                                        ; USER I/O BUFFER AND BUFFER HDR, SPACE (TTY)
                                        ;
002004' 002020'                 KBBFHI: .       KBBUF1          ;BSTRT
002006' 002037'                         .       KBBEN1          ;BEND
002010' 002020'                         .WORD   KBBUF1          ;BGET1
002012' 002020'                         .WORD   KBBUF1          ;BGET2
002014' 002020'                         .WORD   KBBUF1          ;BPUT
002016' 000000                          .       0               ;BFSPEC
                002020'         KBBUF1: .,
                002040'                 .+$KBBSZ                ;DEFAULT SIZE = 20 CHARS,
                002037'         KBBEN1: .-1
                                        .EVEN
                                        ;
002040' 002054'                 TPBFHI: .       TPBUF1          ;BSTRT
002042' 002073'                         .       TPBEN1          ;BEND
002044' 000000                          .       0               ;BGET1 (NOT USED)
002046' 002054'                         .WORD   TPBUF1          ;BGET2
002050' 002054'                         .WORD   TPBUF1          ;BPUT
002052' 000000                          .       0               ;BFSPEC
                002054'         TPBUF1: .,
                002074'                 .+$TPBSZ                ;DEFAULT = 20 CHARS,
                002073'         TPBEN1: .-1
                                        .EVEN
                                        USRLINE:
                002214'                 .=.+$ULNSP              ;DEFAULT = 120
                                        USRCODE:
```

338


```
                                        ;
                                        ; --ONCE-ONLY INIT, CODE
                                        ;    FIND OUT HOW MUCH CORE AND EXPAND USRAREA ACCORDINGLY
                                        ;
002214' 162704  017776          DECCORE:SUB     #17770,R4       ;COME HERE ON TRAP FOR BAD MEM,
002220' 022626                          CMP     (SP)+,(SP)+
002222' 000412                          BR      TRYCORE         ;  REF', AND DEC, CORE SIZE
                                        ;
002224' 000005                  ONCEONLY:RESET
002226' 012706  003364'                 MOV     #TMPSTCK,SP     ;TEMP, STACK
002232' 004767  000000G                 JSR     PC,IFPMP        ;INIT FOR FPMP ROUTINES
002236' 012737  002214' 000004          MOV     #DECCORE,@#4    ;SET UP TRAP ADDR,
002244' 012704  160000                  MOV     #160000,R4      ;28K
002250' 005744                  TRYCORE:TST     -(R4)           ;TRAPS TO DECCORE IF BAD ADDR
002252' 012737  000000  000004          MOV     #0,@#4          ;FALLS THRU IF OK - RESTORE TRAP ADDR,
002260' 010467  001100                  MOV     R4,HICORE
                                        ;
                                        ;INITIALIZE TOP OF BASIC
002264' 012701  000002'                 MOV     #BASIC,R1
                                        ;PRINT HEADING AND READ ANSWER
002270' 004367  000756          INIHDI: JSR     R3,PRMSG
002274'    015     012                  .BYTE   15,12
002276' 040502  044523  020103          .ASCII  'BASIC V001A'
        002304' 030126  030460     101
002311'    015     012                  .BYTE   15,12
002313'    117  052120  043040          .ASCII  'OPT ENS: A=ALL, N=NONE, I=IND'
        002320' 051516  020072  026501
        002326' 046101  026114  047040
        002334' 047055  047117  026105
        002342' 044440  044455  042116
002350'    015     012                  .BYTE   15,12
002352'    077                          .ASCII  '?'
002353'    000                          .BYTE   0
                                        .EVEN
002354' 004767  000722                  JSR     PC,RDANS
002360' 120027  000116                  CMPB    R0,#'N          ;CHECK FOR N
002364' 001512                          BEQ     INISAV
002366' 005002                          CLR     R2              ;CLEAR ALL FLAG
002370' 120027  000101                  CMPB    R0,#'A          ;CHECK FOR A
002374' 001002                          BNE     INITI
002376' 005202                          INC     R2              ;SET ALL FLAG
002400' 000415                          BR      INIALL
002402' 120027  000111          INITI:  CMPB    R0,#'I          ;CHECK FOR I
002406' 001330                          BNE     INIHD
                                        ;PRINT HEADING FOR INDIV FUNCTIONS
002410' 004367  000636                  JSR     R3,PRMSG
002414'    015     012     012          .BYTE   15,12,12
002417'    040  026531  042531          .ASCII  ' Y=YES  N=NO'
        002424' 020123  047040  047055
        002432'    117
002433'    000                          .BYTE   0
002434' 012705  003374'         INIALL: MOV     #FNTAB,R5
002440' 012704  000000G                 MOV     #TABLE5,R4
                                        ;GET NEXT FUNCTION TABLE ADDRESS
002444' 011503                  INILP:  MOV     (R5),R3
```

339

```
 593  002446' 100005                         BPL     INIFN
 594  002450' 062705  000002                 ADD     #2,R5           ;SKIP A FUNCTION
 595  002454' 062704  000002                 ADD     #2,R4
 596  002460' 000771                         BR      INILP
 597                                 ;PRINT QUESTION FOR FUNCTION, GET ANSWER
 598  002462' 020427  000000G        INIFN:  CMP     R4,#TBL5END
 599  002466' 101051                         BHI     INISAV
 600  002470' 020527  003544'                CMP     R5,#FNTABE
 601  002474' 103046                         BHIS    INISAV
 602  002476' 005722                         TST     R2
 603  002500' 001027                         BNE     INIDF
 604  002502' 016567  000004  000014         MOV     4(R5),FNNAM1
 605  002510' 016567  000006  000010         MOV     6(R5),FNNAM2
 606  002516' 004367  000530                 JSR     R3,PRMSG
 607  002522'    015     012                 .BYTE   15,12
 608  002524' 000000                 FNNAM1: .WORD   0
 609  002526' 000000                 FNNAM2: .WORD   0
 610  002530' 035040     040                 .ASCII  ' ) '
 611  002533'    000                         .BYTE   0
 612                                         .EVEN
 613  002534' 004767  000542                 JSR     PC,RDANS
 614  002540' 120027  000116                 CMPB    R0,#'N          ;CHECK FOR N
 615  002544' 001002                         BNE     INITY
 616  002546' 005724                         TST     (R4)+
 617  002550' 000413                         BR      ININXT
 618  002552' 120027  000131         INITY:  CMPB    R0,#'Y          ;CHECK FOR Y
 619  002556' 001341                         BNE     INIFN
 620                                 ;DEFINE THE FUNCTION
 621  002560' 010100                 INIDF:  MOV     R1,R0
 622  002562' 162700  000000G                SUB     #TABLES,R0
 623  002566' 010024                         MOV     R0,(R4)+        ;PLACE ADDRESS IN TABLE
 624  002570' 012321                 INIMV:  MOV     (R3)+,(R1)+
 625  002572' 020365  000002                 CMP     R3,2(R5)
 626  002576' 103774                         BLO     INIMV
 627  002600' 062705  000010         ININXT: ADD     #10,R5
 628  002604' 020427  000000G                CMP     R4,#TBL5END
 629  002610' 101715                         BLOS    INILP
 630
 631                                 ;SAVE TOP OF BASIC
 632  002612' 010167  175102         INISAV: MOV     R1,USRAREA
 633                                 ;MOVE USER AREA TO TOP OF BASIC
 634  002616' 012702  001670'                MOV     #USR,R2
 635  002622' 010203                         MOV     R2,R3           ;COMPUTE RELOCATION CONSTANT
 636  002624' 160103                         SUB     R1,R3
 637  002626' 020227  002214'        MOVLP:  CMP     R2,#USRCODE
 638  002632' 103005                         BHIS    MOVDON
 639  002634' 012221                         MOV     (R2)+,(R1)+
 640  002636' 001773                         BEQ     MOVLP
 641  002640' 160341                         SUB     R3,-(R1)        ;RELOCATE NON-ZERO ENTRIES
 642  002642' 005721                         TST     (R1)+
 643  002644' 000770                         BR      MOVLP
 644                                 MOVDON:
 645
```

340


```
 646
 647                                 ;INITIALIZE USER AREA
 648  002646' 016704  000512                 MOV     HICORE,R4       ;R4 IS HIGHEST ADDR, IN CORE
 649  002652' 162704  000300                 SUB     #300,R4
 650  002656' 016705  175116                 MOV     USRAREA,R5
 651  002662' 010465  000000G                MOV     R4,LIMIT(R5)
 652  002666' 162704  000134                 SUB     #134,R4         ;POINTER SPACE
 653  002672' 010465  000000G                MOV     R4,POL(R5)      ;BASE OF STACK
 654  002676' 162704  000000G                SUB     #$STKSZ,R4      ;MAX. STACK SIZE
 655  002702' 010465  000000G                MOV     R4,ARRAYS(R5)   ;HIEST ARRAY STOR. LOC.
 656  002706' 062704  000040                 ADD     #40,R4
 657  002712' 010465  000000G                MOV     R4,FOSIZE(R5)
 658  002716' 012765  032331  000000G        MOV     #32331,RND1(R5)
 659  002724' 012765  163251  000000G        MOV     #163251,RND2(R5)
 660
 661                                 ;DIALOGUE TO SET UP TERMINAL TYPE
 662  002732' 005003                 ASKSER: CLR     R3
 663  002734' 004367  000312                 JSR     R3,PRMSG
 664  002740'    015     012     012         .BYTE   15,12,12
 665  002743'    106  051501  020124         .ASCII  'FAST SER TERM?'
      002750' 042523  020122  042524
      002756' 046522     077
 666  002761'    000                         .BYTE   0
 667                                         .EVEN
 668  002762' 004767  000314                 JSR     PC,RDANS
 669  002766' 120027  000116                 CMPB    R0,#'N
 670  002772' 001457                         BEQ     ASKDON
 671  002774' 120027  000131                 CMPB    R0,#'Y
 672  003000' 001354                         BNE     ASKSER
 673  003002' 005203                         INC     R3
 674
 675  003004' 004367  000242         ASKLA:  JSR     R3,PRMSG
 676  003010'    015     012                 .BYTE   15,12
 677  003012' 040514  030063  024040         .ASCII  'LA30 (300 BAUD)?'
      003020' 030063  020060  040502
      003026' 042125  037451
 678  003032'    000                         .BYTE   0
 679         003034'                         .EVEN
 680  003034' 004767  000242                 JSR     PC,RDANS
 681  003040' 120027  000131                 CMPB    R0,#'Y
 682  003044' 001432                         BEQ     ASKDON
 683  003046' 120027  000116                 CMPB    R0,#'N
 684  003052' 001354                         BNE     ASKLA
 685  003054' 005203                         INC     R3
 686
 687  003056' 004367  000170         ASKVT:  JSR     R3,PRMSG
 688  003062'    015     012                 .BYTE   15,12
 689  003064' 052126  032460  024040         .ASCII  'VT05 (>=600 BAUD)?'
      003072' 036476  030066  020060
      003100' 040502  042125  037451
 690  003106'    000                         .BYTE   0
 691         003110'                         .EVEN
 692  003110' 004767  000166                 JSR     PC,RDANS
 693  003114' 120027  000131                 CMPB    R0,#'Y
 694  003120' 001404                         BEQ     ASKDON
```

341

```
695  003122' 120027  000116                  CMPB    R0,#'N
696  003126' 001353                          BNE     ASKYT
697  003130' 000700                          BR      ASKSER
698
699  003132' 010302                  ASKDON: MOV     R3,R2
700  003134' 001412                          BEQ     FILLDON
701  003136' 006303                          ASL     R3
702  003140' 060302                          ADD     R3,R2
703  003142' 062702  003303'                 ADD     #FILLTB,R2
704  003146' 010503                          MOV     R5,R3
705  003150' 062703  000000G                 ADD     #FILLCO,R3
706  003154' 112223                          MOVB    (R2)+,(R3)+
707  003156' 112223                          MOVB    (R2)+,(R3)+
708  003160' 112223                          MOVB    (R2)+,(R3)+
709                                  FILLDON:
710                                  ;PRINT OUT IF USER FUNCTIONS LOADED
711  003162' 005737  000046                  TST     @#46
712  003166' 001414                          BEQ     NOUSR
713  003170' 004367  000056                  JSR     R3,PRMSG
714  003174'    015     012     012          .BYTE   15,12,12
715  003177'    125 042523  020122          .ASCII  'USER FNS LOADED'
     003204' 047106  020123  047514
     003212' 042101  042105
716  003216'    000                          .BYTE 0
717          003220'                         .EVEN
718                                  NOUSR:
719  003220' 004367  000026                  JSR     R3,PRMSG
720  003224'    015     012     012          .BYTE   15,12,12,0
     003227'    000
721                                          .EVEN
722
723                                  ;CHECK IF RNDL LOADED
724  003230' 012702  000000G                 MOV     #TABLES,R2
725  003234' 012203                          MOV     (R2)+,R3
726  003236' 001403                          BEQ     NORND
727  003240' 062703  000014                  ADD     #RNDFN-RNDLFN,R3
728  003244' 010312                          MOV     R3,(R2)         ;DEFINE RND ALSO
729                                  NORND:
730  003246' 000167  000000G                 JMP     START
731
732                                  ;SUBROUTINE TO PRINT A MESSAGE
733  003252' 010300                  PRMSG:  MOV     R3,R0
734  003254' 105737  1777760         PRLP:   TSTB    @#TPB+2
735  003260' 100375                          BPL     PRLP
736  003262' 112037  000000G                 MOVB    (R0)+,@#TPB
737  003266' 001372                          BNE     PRLP
738  003270' 010003                          MOV     R0,R3
739  003272' 005203                          INC     R3
740  003274' 006203                          ASR     R3
741  003276' 006303                          ASL     R3
742  003300' 000203                          RTS     R3
743
744                                  ;SUBROUTINE TO READ ANSWER FROM TELETYPE
745  003302' 005267  176452          RDANS:  INC     RAND
746  003306' 105737  0000000                 TSTB    @#TKS
```

342


```
747  003312' 100373                          BPL     RDANS
748  003314' 113700  000002G                 MOVB    @#TKS+2,R0
749  003320' 042700  177600                  BIC     #177600,R0
750  003324' 010067  000004                  MOV     R0,INCH
751  003330' 004367  177716                  JSR     R3,PRMSG
752  003334' 000000                  INCH:   .WORD   0
753  003336' 016700  177772                  MOV     INCH,R0
754  003342' 000207                          RTS     PC
755          003364'                         .       =.+20
756                                  TMPSTCK:
757  003364' 000000                  HICORE: .WORD   0
758
759          003363'                 FILLTB: .=.-3
760  003366'    000     011     015          .BYTE   0,11.,15        ;LA37
761  003371'    000     004     012          .BYTE   0,04,12         ;VT05
762
763                                  FNTAB:
764  003374' 000002'                         .WORD   RNDLFN
765  003376' 001140'                         .WORD   RNDFNE
766  003400' 047122  020104                  .ASCII  'RND '
767  003404' 177777                          .WORD   -1              ; RND
768  003406' 177777                          .WORD   -1              ; SIN
769  003410' 177777                          .WORD   -1              ; COS
770  003412' 177777                          .WORD   -1              ; SQR
771  003414' 177777                          .WORD   -1              ; ATN
772  003416' 177777                          .WORD   -1              ; EXP
773  003420' 177777                          .WORD   -1              ; LOG
774  003422' 000140'                         .WORD   ABSFN           ; ABS
775  003424' 000236'                         .WORD   ABSFNE
776  003426' 041101  020123                  .ASCII  'ABS '
777  003432' 177777                          .WORD   -1              ; INT
778  003434' 000236'                         .WORD   SGNFN           ; SGN
779  003436' 000324'                         .WORD   SGNFNE
780  003440' 043523  020116                  .ASCII  'SGN '
781  003444' 000324'                         .WORD   TABFN           ; TAB
782  003446' 000436'                         .WORD   TABFNE
783  003450' 040524  020102                  .ASCII  'TAB '
784  003454' 000436'                         .WORD   LENFN           ; LEN
785  003456' 000510'                         .WORD   LENFNE
786  003460' 042514  020116                  .ASCII  'LEN '
787  003464' 000510'                         .WORD   ASCFN           ; ASC
788  003466' 000574'                         .WORD   ASCFNE
789  003470' 051501  020103                  .ASCII  'ASC '
790  003474' 000574'                         .WORD   CHRSFN          ; CHR$
791  003476' 000646'                         .WORD   CHRFNE
792  003500' 044103  022122                  .ASCII  'CHR$'
793  003504' 000646'                         .WORD   POSFN           ; POS
794  003506' 001152'                         .WORD   POSFNE
795  003510' 047520  020123                  .ASCII  'POS '
796  003514' 001152'                         .WORD   SEGFN           ; SEG
797  003516' 001420'                         .WORD   SEGFNE
798  003520' 042523  022107                  .ASCII  'SEG$'
799  003524' 001420'                         .WORD   VALFN           ; VAL
800  003526' 001566'                         .WORD   VALFNE
801  003530' 040526  020114                  .ASCII  'VAL '
```

343

```
  002  003534' 001566'                    ,WORD   STRFN           ; STR
  003  003536' 001670'                    ,WORD   STRFNE
  004  003540' 052123  022122             ,ASCII  /STRS/
  005                              FNTABE:
  006
  007          002224'                    ,END    ONCEON
```

344


```
ABSFN   000140R     ABSFNE  000236R     ABSINT  000174R     ABSX    000222R
ARGB    = ******  G ARRAYS = ******  G ASCFN   000510R     ASCFNE  000574R
ASKOON  003132R     ASKLA   003004R     ASKSER  002732R     ASKVT   003056R
BASICH  000002R     BL      = 000040    CHRFNE  000646R     CHRSFN  000574R
COLUMN  = ******  G DECCOR  002214R     ERABSA  000226R     ERABSS  000232R
ERASCA  000564R     ERASCS  000570R     ERCHRS  000642R     ERLENA  000500R
ERLENS  000504R     ERPOSA  001142R     ERPOSS  001146R     ERRARG  = ******  G
ERRMIX  = ******  G ERRNOA  000134R     ERRNOS  000130R     ERRPOL  = ******  G
ERRSYN  = ******  G ERSEGA  001410R     ERSEGS  001414R     ERSGNA  000314R
ERSGNS  000320R     ERSTRA  001660R     ERSTRS  001664R     ERVARS  000432R
ERVALA  001562R     ERVALS  001556R     EVAL    = ******  G FAC1    = ******  G
FAC2    = ******  G FILLCO  = ******  G FILLDO  003102R     FILLTB  = 003363R
FNEND   001670R     FNNAM1  002524R     FNNAM2  002526R     FNTAB   003374R
FNTABE  003544R     HICORE  003364R     IFPMP   = ******  G INCH    003334R
INIALL  002434R     INIDF   002560R     ININFN  002462R     ININO   002270R
INILP   002444R     INIMV   002570R     ININXT  002600R     INISAV  002612R
INITI   002402R     INITY   002552R     INT     = ******  G KBREN1  = 002037R
KBBFH1  002004R     KBBUF1  = 002020R     LENFN   000436R     LENFNE  000510R
LIMIT   = ******  G MAKEST  = ******  G MOVDON  002646R     MOVLP   002626R
NORM    = ******  G NORND   003246R     NOUSR   003220R     NUMSGN  = ******  G
ONCEON  002224R     OPRATO  = ******  G PC      =%000007    POL     = ******  G
POSIZE  = ******  G POSF    001124R     POSFN   000646R     POSFNE  001152R
POSFST  001050R     POSF2   001126R     POSNO   001116R     POSNXT  001076R
POSREM  001070R     POSTRY  001054R     POSX    000754R     POSY    001000R
PRLP    003254R     PRMSG   003252R     RAND    001700R     RDANS   003302R
REXP    000074R     RNDFN   000016R     RNDFNE  000140R     RNDFN2  000016R
RNDLFN  000002R     RND1    = ******  G RND2    = ******  G RNDRM   000062R
RPLUS   000052R     R0      =%000000    R1      =%000001    R2      =%000002
R3      =%000003    R4      =%000004    R5      =%000005    SAVCHA  = ******  G
SEGFN   001152R     SEGFNE  001420R     SEGL    001256R     SEGLP   001366R
SEGNUL  001400R     SEGRNG  001312R     SEGR2   001320R     SEGTY   001270R
SEGTY2  001304R     SEGX    001404R     SEGX0   001252R     SGNFLT  000270R
SGNFN   000236R     SGNFNE  000324R     SGNPOS  000276R     SGNX    000310R
SOPRAT  = ******  G SP      =%000006    START   = ******  G STPRO   = ******  G
STRFN   001566R     STRFNE  001670R     TABB    000344R     TABC    000360R
TABFN   000324R     TABFNE  000436R     TABLE5  = ******  G TARNON  000376R
TABNUL  000426R     TBL3EN  = ******  G TKS     = ******  G TMPSTC  003364R
TPB     = ******  G TPBEN1  = 002073R     TPBFH1  002040R     TPBUF1  = 002054R
TRYCOR  002250R     T1      = ******  G T2      = ******  G T3      = ******  G
USR     001670R     USRARE  000000RG    USRCOD  002214R     USRLIN  002074R
VAL     = ******  G VALCE   001524R     VALFN   001420R     VALFNE  001566R
VALJ    001552R     VALR    001460R     VALX    001536R     %KBBSZ  = 000020
$STKSZ  = ******  G $TPBSZ  = 000020    $ULNSP  = 000120    ,COMMA  = ******  G
,RPAR   = ******  G ,       = 003544R
```

ERRORS DETECTED: 0

345

RUN-TIME: 10 SECONDS
CORE USED: 3K

348

349

```
        270   272   281   282   285   288   291   303   304   312   313   316   333   334
        343   344   352   353   372   374   377   378   379   380   390   403   408   412
        416   417   418   431   433   434   435   448   451   457   458   460   550   554
START    33   730
STPRO    41   388   401
STRFN   444#  802
STRFNE  465#  803
T1       40   290   298   308   454   458
T2       40   284   306   453
T3       40
TABB    169#  172
TABC    170   173#
TABFN   164#  781
TABFNE  190#  782
TABLE5   36   590   622   724
TABNON  176   179#
TABNUL  178   188#
TBL5EN   36   598   628
TKS      32   746   748
TMPSTC  554   756#
TPB      32   734   736
TPBEN1  532   539#
TPBFH1  507   531#
TPBUF1  531   534   535   537#
TRYCOR  551   558#
USR     474#  634
USRARE   34    75#  632   650
USRCOD  481   543#  637
USRLIN  482   541#
VAL      42   420
VALCE   426#  429
VALFN   398#  799
VALFNE  440#  800
VALJ    409   437#
VALR    405   410#
VALX    427   431#
.       153   174   187   202   526   527#  528   537   538#  539   542#  755#  759
.COMMA   40   246   250   328   335
.RPAR    37    81   122   144   165   196   213   230   255   340   400   446
SKBBSZ   56   527
SSTKSZ   44   654
STPBSZ   52   538
SULNSP   60   542
```

BASICH MACX11 V021 22-MAR-73 15:45 PAGE 1

```
        ; BASIC/PTS  PART3-BASICH
        ;
        ; DEC-11-LPTBA-A-LA3
        ;
        ; COPYRIGHT 1973
        ;
        ; DIGITAL EQUIPMENT CORPORATION
        ; MAYNARD, MASSACHUSETTS 01754
```

351

-1

```
                        .TITLE  BASIC V001A EDIT #020 (REF) 01/31/73
                        )                               BASIC SOURCE FILE #15
                        )
                        )
                        )
                        )
                        )  USER AREA AND ONCE-ONLY INIT. CODE,
                        )
                        )  TO BE LINKED LAST AFTER BASIC1 AND FPMP-11 TO FORM BASIC.
                        )
                        )
                        )
                        )
                        )
                        )
                        )
```

352


```
                        )
                        ) GLOBALS
                        )
                                .GLOBL  TPB, IKS
                                .GLOBL  START, FILLCO
                                .GLOBL  USRAREA
                                .GLOBL  LIMIT, POL, ARRAYS, PDSIZE
                                .GLOBL  TABLES, TBL5END
                                .GLOBL  COLUMN, FAC1, FAC2, .RPAR
                                .GLOBL  ERRMIX, ERRPOL, ERRSYN, ERRAMG
                                .GLOBL  EVAL, INT, MAKEST, OPRATO, SOPRAT
                                .GLOBL  .COMMA, T1, T2, T3
                                .GLOBL  ARGB, SAVCWAR, NUMSGN, STPRO
                                .GLOBL  NORM, VAL
                                .GLOBL  RND1, RND2
                                .GLOBL  SSTKSZ ,IFPMP
                        )
                        ) ASSEMBLY PARAMETERS
                        )
                                .IFNDF  $TPBSZ  ;TELEPRINTER BUFFER SIZE
                        $TPBSZ  =20             ;CHARACTERS
                                .ENDC
                        )
                                .IFNDF  $KBBSZ  ;KEYBOARD BUFF. SIZE
                        $KBBSZ  =20             ;CHARACTERS
                                .ENDC
                        )
                                .IFNDF  $ULNSP  ;USER LINE SPACE
                        $ULNSP  =120            ;BYTES
                                .ENDC
```

353

```
                        .CSECT
000000  R0      =%0
000001  R1      =%1
000002  R2      =%2
000003  R3      =%3
000004  R4      =%4
000005  R5      =%5
000006  SP      =%6
000007  PC      =%7
000040  BL      =40

000000' 000000          USRAREA:.WORD   0
                        BASICH:
                        ;RND FUNCTION -- GENERATE RANDOM NUMBER
                        ;       SCAN PAST ARG IF PRESENT
000002' 004737  000000G RNOLFN: JSR     PC,@#EVAL
000006' 103432          BCS     ERRNDA
000010' 122127  000000G         CMPB    (R1)+,#,RPAR
000014' 001045          BNE     ERRNDS
                        RNOFN:
000016' 010500          RNOFN2: MOV     R5,R0
000020' 062700  000000G         ADD     #RND2,R0
000024' 011003                  MOV     @R0,R3
000026' 014002                  MOV     -(R0),R2
000030' 006303                  ASL     R3              ;MULT BY 2
000032' 006102                  ROL     R2
000034' 062002                  ADD     (R0)+,R2        ;NOW BY 3
000036' 061003                  ADD     @R0,R3
000040' 005502                  ADC     R2
000042' 061002                  ADD     @R0,R2          ;NOW BY 2**16+3
000044' 100002                  BPL     RPLUS
000046' 062702  100000          ADD     #100000,R2      ;GET 2**32+G
000052' 010310          RPLUS:  MOV     R3,@R0          ;STORE NEW GENERATORS
000054' 010240                  MOV     R2,-(R0)
000056' 012700  000201          MOV     #201,R0         ;INITIAL EXPONENT
000062' 006303          RNORM:  ASL     R3              ;FLOAT RESULT
000064' 006102                  ROL     R2
000066' 103402                  BCS     REXP            ;JUMP WHEN LEADING BIT FOUND
000070' 005300                  DEC     R0              ;ADJUST EXPONENT FOR SHIFT
000072' 000773                  BR      RNORM
000074' 105003          REXP:   CLRB    R3
000076' 150203                  BISB    R2,R3
000100' 000303                  SWAB    R3
000102' 105002                  CLRB    R2
000104' 150002                  BISB    R0,R2           ;INSERT EXPONENT INTO RESULT
000106' 000302                  SWAB    R2
000110' 006002                  ROR     R2
000112' 006003                  ROR     R3              ;INSERT + SIGN
000114' 010265  000000G         MOV     R2,FAC1(R5)
000120' 010365  000000G         MOV     R3,FAC2(R5)
000124' 000137  000000G         JMP     @#OPRATOR
000130' 000137  000000G ERRNDS: JMP     @#ERRSYN
000134' 000137  000000G ERRNDA: JMP     @#ERRARG
                        RNOFNE:
```

354


```
                        ;ABS FUNCTION ROUTINE
000140' 004737  000000G ABSFN:  JSR     PC,@#EVAL
000144' 103430          BCS     ERABSA
000146' 122127  000000G         CMPB    (R1)+,#,RPAR
000152' 001027          BNE     ERABSS
000154' 005765  000000G         TST     FAC1(R5)
000160' 001405          BEQ     ABSINT
000162' 100017          BPL     ABSX
000164' 042765  100000  000000G BIC     #100000,FAC1(R5)
000172' 000413          BR      ABSX
000174' 005765  000000G ABSINT: TST     FAC2(R5)
000200' 100010          BPL     ABSX
000202' 005465  000000G         NEG     FAC2(R5)
000206' 102005          BVC     ABSX
000210' 012765  044000  000000G MOV     #44000,FAC1(R5)
000216' 005065  000000G         CLR     FAC2(R5)
000222' 000137  000000G ABSX:   JMP     @#OPRATOR
000226' 000137  000000G ERABSA: JMP     @#ERRARG
000232' 000137  000000G ERABSS: JMP     @#ERRSYN
                        ABSFNE:
```

355

```
                                        ;SGN FUNCTION ROUTINE
000236' 004737 000000G          SGNFN:  JSR     PC,@#EVAL
000242' 103424                          BCS     ERSGNA
000244' 122127 000000G                  CMPB    (R1)+,#,RPAR
000250' 001023                          BNE     ERSGNS
000252' 005000                          CLR     R0
000254' 005765 000000G                  TST     FAC1(R5)
000260' 001003                          BNE     SGNFLT
000262' 005765 000000G                  TST     FAC2(R5)
000266' 001410                          BEQ     SGNX
000270' 100002                  SGNFLT: BPL     SGNPOS
000272' 005300                          DEC     R0
000274' 000401                          BR      .+4
000276' 005200                  SGNPOS: INC     R0
000300' 010065 000000G                  MOV     R0,FAC2(R5)
000304' 005065 000000G                  CLR     FAC1(R5)
000310' 000137 000000G          SGNX:   JMP     @#OPRATOR
000314' 000137 000000G          ERSGNA: JMP     @#ERRARG
000320' 000137 000000G          ERSGNS: JMP     @#ERRSYN
                                SGNFNE:
```

356


```
                                        ;TAB FUNCTION ROUTINE
000324' 004737 000000G          TABFN:  JSR     PC,@#ARGB
000330' 122127 000000G                  CMPB    (R1)+,#,RPAR
000334' 001036                          BNE     ERTABS
000336' 005046                          CLR     -(SP)
000340' 116516 000000G                  MOVB    FAC2(R5),(SP)
000344' 021627 000110           TABB:   CMP     (SP),#72.
000350' 103403                          BLO     TABC
000352' 162716 000110                   SUB     #72.,(SP)
000356' 000772                          BR      TABB
000360' 167516 000000G          TABC:   SUB     @COLUMN(R5),(SP)
000364' 100001                          BPL     .+4
000366' 005016                          CLR     (SP)
000370' 001002                          BNE     TABNON
000372' 005316                          DEC     (SP)
000374' 000414                          BR      TABNULL
000376' 010502                  TABNON: MOV     R5,R2
000400' 004737 000000G                  JSR     PC,@#MAKESTR
000404' 011600                          MOV     (SP),R0
000406' 005002                          CLR     R2
000410' 151002                          BISB    (R0),R2
000412' 062700 000003                   ADD     #3,R0
000416' 112720 000040                   MOVB    #BL,(R0)+
000422' 005302                          DEC     R2
000424' 003374                          BGT     .-6
000426' 000137 000000G          TABNULL:JMP     @#SOPRATR
000432' 000137 000000G          ERTABS: JMP     @#ERRSYN
                                TABFNE:
```

357

```
                                        ; LEN FUNCTION ROUTINE
000436' 004737 000000G          LENFN:  JSR     PC,@#EVAL
000442' 103016                          BCC     ERLENA
000444' 122127 000000G                  CMPB    (R1)+,#,RPAR
000450' 001015                          BNE     ERLENS
000452' 005065 000000G                  CLR     FAC1(R5)
000456' 005065 000000G                  CLR     FAC2(R5)
000462' 012602                          MOV     (SP)+,R2
000464' 005202                          INC     R2
000466' 001402                          BEQ     .+6
000470' 114265 000000G                  MOVB    -(R2),FAC2(R5)
000474' 000137 000000G                  JMP     @#OPRATOR
000500' 000137 000000G          ERLENA: JMP     @#ERRARG
000504' 000137 000000G          ERLENS: JMP     @#ERRSYN
                                LENFNE:
```



```
                                        ; ASC FUNCTION ROUTINE
000510' 004737 000000G          ASCFN:  JSR     PC,@#EVAL
000514' 103023                          BCC     ERASCA
000516' 122127 000000G                  CMPB    (R1)+,#,RPAR
000522' 001022                          BNE     ERASCS
000524' 012602                          MOV     (SP)+,R2
000526' 020227 177777                   CMP     R2,#177777
000532' 001414                          BEQ     ERASCA
000534' 121227 000001                   CMPB    (R2),#1
000540' 001011                          BNE     ERASCA
000542' 005065 000000G                  CLR     FAC1(R5)
000546' 005065 000000G                  CLR     FAC2(R5)
000552' 116265 000003 000000G           MOVB    3(R2),FAC2(R5)
000560' 000137 000000G                  JMP     @#OPRATOR
000564' 000137 000000G          ERASCA: JMP     @#ERRARG
000570' 000137 000000G          ERASCS: JMP     @#ERRSYN
                                ASCFNE:

                                        ; CHR$ FUNCTION ROUTINE
000574' 004737 000000G          CHRSFN: JSR     PC,@#ARGB
000600' 122127 000000G                  CMPB    (R1)+,#,RPAR
000604' 001016                          BNE     ERCHRS
000606' 142765 000200 000000G           BICB    #200,FAC2(R5)   ;TAKE CHAR MOD 126
000614' 012746 000001                   MOV     #1,-(SP)
000620' 010502                          MOV     R5,R2
000622' 004737 000000G                  JSR     PC,@#MAKESTR
000626' 011600                          MOV     (SP),R0
000630' 116560 000000G 000003           MOVB    FAC2(R5),3(R0)
000636' 000137 000000G                  JMP     @#SOPRATR
000642' 000137 000000G          ERCHRS: JMP     @#ERRSYN
                                CHRFNE:
```

```
                                ; POS FUNCTION ROUTINE
000646' 004737 000000G  POSFN:  JSR     PC,@#EVAL
000652' 103133                  BCC     ERPOSA
000654' 122127 000000G          CMPB    (R1)+,#,COMMA
000660' 001132                  BNE     ERPOSS
000662' 004737 000000G          JSR     PC,@#EVAL
000666' 103125                  BCC     ERPOSA
000670' 122127 000000G          CMPB    (R1)+,#,COMMA
000674' 001124                  BNE     ERPOSS
000676' 004737 000000G          JSR     PC,@#EVAL
000702' 103517                  BCS     ERPOSA
000704' 004737 000000G          JSR     PC,@#INT
000710' 122127 000000G          CMPB    (R1)+,#,RPAR
000714' 001114                  BNE     ERPOSS
000716' 005000                  CLR     R0
000720' 005765 000000G          TST     FAC1(R5)
000724' 001077                  BNE     POSF
000726' 005765 000000G          TST     FAC2(R5)
000732' 003474                  BLE     POSF
000734' 156500 000000G          BISB    FAC2(R5),R0     ;R0 IS N
                                ;CHECK NULL XS
000740' 026627 000002 177777    CMP     2(SP),#177777
000746' 001002                  BNE     POSX
000750' 005000                  CLR     R0
000752' 000464                  BR      POSF
                                ;COMPUTE LENGTH OF XS
000754' 005002          POSX:   CLR     R2
000756' 157602 000002           BISB    @2(SP),R2
                                ;CHECK NULL YS
000762' 021627 177777           CMP     (SP),#177777
000766' 001004                  BNE     POSY
                                ;NULL YS, ANS IS MIN(N,LEN(XS))
000770' 020002                  CMP     R0,R2
000772' 003454                  BLE     POSF
000774' 010200                  MOV     R2,R0
000776' 000452                  BR      POSF
                                ;SAVE ADDR OF END OF YS IN T2
001000' 005003          POSY:   CLR     R3
001002' 157603 000000           BISB    @(SP),R3
001006' 061603                  ADD     (SP),R3
001010' 062703 000003           ADD     #3,R3
001014' 010365 000000G          MOV     R3,T2(R5)
001020' 012603                  MOV     (SP)+,R3
001022' 062703 000003           ADD     #3,R3
                                ;SAVE ADDR OF END OF XS IN T1
001026' 061602                  ADD     (SP),R2
001030' 062702 000003           ADD     #3,R2
001034' 010265 000000G          MOV     R2,T1(R5)
001040' 012602                  MOV     (SP)+,R2
001042' 060002                  ADD     R0,R2
001044' 062702 000002           ADD     #2,R2
                                ;FIND MATCHING FIRST CHARACTER
001050' 121322          POSFST: CMPB    (R3),(R2)+
001052' 001406                  BEQ     POSREM
001054' 005200          POSTRY: INC     R0
```

360


```
001056' 020265 000000G          CMP     R2,T1(R5)
001062' 001372                  BNE     POSFST
001064' 005000                  CLR     R0
001066' 000417                  BR      POSF2
                                ;CHECK THAT REMAINING CHARS MATCH
001070' 010246          POSREM: MOV     R2,-(SP)
001072' 010346                  MOV     R3,-(SP)
001074' 005203                  INC     R3
001076' 020365 000000G  POSNXT: CMP     R3,T2(R5)
001102' 001410                  BEQ     POSF
001104' 020265 000000G          CMP     R2,T1(R5)
001110' 001402                  BEQ     POSNO
001112' 122223                  CMPB    (R2)+,(R3)+
001114' 001770                  BEQ     POSNXT
001116' 012603          POSNO:  MOV     (SP)+,R3
001120' 012602                  MOV     (SP)+,R2
001122' 000754                  BR      POSTRY
                                ;RETURN FROM FUNCTION
001124' 022626          POSF:   CMP     (SP)+,(SP)+
001126' 010065 000000G  POSF2:  MOV     R0,FAC2(R5)
001132' 005065 000000G          CLR     FAC1(R5)
001136' 000137 000000G          JMP     @#OPRATOR
001142' 000137 000000G  ERPOSA: JMP     @#ERRARG
001146' 000137 000000G  ERPOSS: JMP     @#ERRSYN
                        POSFNE:
```

361

```
                                ; SEG FUNCTION ROUTINE
001152' 004737  000000G         SEGFN:  JSR     PC,@#EVAL
001156' 103114                          BCC     ERSEGA
001160' 122127  000000G                 CMPB    (R1)+,#,COMMA
001164' 001113                          BNE     ERSEGS
001166' 004737  000000G                 JSR     PC,@#EVAL
001172' 103506                          BCS     ERSEGA
001174' 004737  000000G                 JSR     PC,@#INT
001200' 016546  000000G                 MOV     FAC1(R5),-(SP)
001204' 016546  000000G                 MOV     FAC2(R5),-(SP)
001210' 122127  000000G                 CMPB    (R1)+,#,COMMA
001214' 001077                          BNE     ERSEGS
001216' 004737  000000G                 JSR     PC,@#EVAL
001222' 103472                          BCS     ERSEGA
001224' 004737  000000G                 JSR     PC,@#INT
001230' 122127  000000G                 CMPB    (R1)+,#,RPAR
001234' 001067                          BNE     ERSEGS
                                ;GET X VALUE IN R2
001236' 012602                          MOV     (SP)+,R2
001240' 012600                          MOV     (SP)+,R0
001242' 100403                          BMI     SEGX0
001244' 001055                          BNE     SEGNUL
001246' 005702                          TST     R2
001250' 003002                          BGT     SEGL
001252' 012702  000001          SEGX0:  MOV     #1,R2
                                ;COMPUTE LENGTH OF A$ IN R0
001256' 005000                  SEGL:   CLR     R0
001260' 021627  177777                  CMP     (SP),#177777    ;CHECK NULL STRING
001264' 157600  000000                  BISB    @(SP),R0
                                ;GET Y VALUE IN R3
001270' 005765  000000G         SEGTY:  TST     FAC1(R5)
001274' 100441                          BMI     SEGNUL
001276' 001402                          BEQ     SEGTY2
001300' 010003                          MOV     R0,R3
001302' 000403                          BR      SEGRNG
001304' 016503  000000G         SEGTY2: MOV     FAC2(R5),R3
001310' 003433                          BLE     SEGNUL
                                ;CHECK 0<=R2<=R3<=R0
001312' 020003                  SEGRNG: CMP     R0,R3
001314' 101001                          BHI     SEGR2
001316' 010003                          MOV     R0,R3
001320' 020203                  SEGR2:  CMP     R2,R3
001322' 101026                          BHI     SEGNUL
                                ;COMPUTE LENGTH OF OUTPUT STRING
001324' 160203                          SUB     R2,R3
001326' 005203                          INC     R3
                                ;MAKE NEW STRING OF THE CORRECT LENGTH
001330' 010246                          MOV     R2,-(SP)        ;SAVE START CHAR
001332' 010502                          MOV     R5,R2
001334' 010346                          MOV     R3,-(SP)
001336' 004737  000000G                 JSR     PC,@#MAKESTR
                                ;FIX STACK AND MOVE IN CHARS FROM OLD STRING
001342' 012602                          MOV     (SP)+,R2        ;NEW STRING POINTER
001344' 012600                          MOV     (SP)+,R0        ;START CHAR
001346' 061600                          ADD     (SP),R0         ;ADD OLD STR POINTER
```

362


```
001350' 010216                          MOV     R2,(SP)         ;SAVE NEW STRING
001352' 005003                          CLR     R3
001354' 151203                          BISB    (R2),R3         ;R3 IS NEW STRING LENGTH
001356' 062702  000003                  ADD     #3,R2           ;ADDR FIRST CHAR IN NEW STR
001362' 062700  000002                  ADD     #2,R0           ;ADDR START CHAR IN OLD STR
001366' 112022                  SEGLP:  MOVB    (R0)+,(R2)+     ;FILL IN NEW STRING
001370' 005303                          DEC     R3
001372' 001375                          BNE     SEGLP
001374' 000137  000000G                 JMP     @#STPRO
                                ;OUTPUT NULL STRING
001400' 012716  177777          SEGNUL: MOV     #177777,(SP)
001404' 000137  000000G         SEGX:   JMP     @#SOPRATR
001410' 000137  000000G         ERSEGA: JMP     @#ERRARG
001414' 000137  000000G         ERSEGS: JMP     @#ERRSYN
                                SEGFNE:
```

363

```
                                ;VAL FUNCTION ROUTINE
001420' 004737 000000G  VALFN:  JSR     PC,@#EVAL
001424' 103056                  BCC     ERVALA
001426' 122127 000000G          CMPB    (R1)+,#,RPAR
001432' 001051                  BNE     ERVALS
                                ;READ STRING
001434' 011600                  MOV     (SP),R0
001436' 020027 177777           CMP     R0,#177777      ;CHECK NULL STRING
001442' 001006                  BNE     VALR
001444' 005065 000000G          CLR     FAC1(R5)
001450' 005065 000000G          CLR     FAC2(R5)
001454' 005726                  TST     (SP)+
001456' 000435                  BR      VALJ
001460' 005002          VALR:   CLR     R2
001462' 151002                  BISB    (R0),R2
001464' 010216                  MOV     R2,(SP)
001466' 062700 000003           ADD     #3,R0
001472' 060002                  ADD     R0,R2
001474' 105012                  CLRB    (R2)
001476' 010446                  MOV     R4,-(SP)
001500' 010146                  MOV     R1,-(SP)
001502' 010246                  MOV     R2,-(SP)
                                ;READ ASCII NUMBER AND EXPONENT
001504' 004737 000000G          JSR     PC,@#VAL
                                ;NORMALIZE TO FORM FLT PT NUMBER
001510' 004737 000000G          JSR     PC,@#NORM
001514' 010365 000000G          MOV     R3,FAC1(R5)
001520' 010465 000000G          MOV     R4,FAC2(R5)
                                ;CHECK END OF STRING
001524' 105710          VALCE:  TSTB    (R0)
001526' 001403                  BEQ     VALX
001530' 122027 000040           CMPB    (R0)+,#BL
001534' 001773                  BEQ     VALCE
                                ;RESTORE REGS AND RETURN TO EVAL
001536' 022026          VALX:   CMP     R0,(SP)+
001540' 001010                  BNE     ERVALA
001542' 012601                  MOV     (SP)+,R1
001544' 012604                  MOV     (SP)+,R4
001546' 012602                  MOV     (SP)+,R2
001550' 110210                  MOVB    R2,(R0)         ;RESTORE STRING LENGTH
001552' 000137 000000G  VALJ:   JMP     @#OPRATOR
001556' 000137 000000G  ERVALS: JMP     @#ERRSYN
001562' 000137 000000G  ERVALA: JMP     @#ERRARG
                        VALFNE:
```

364


```
                                ;STR FUNCTION ROUTINE
001566' 004737 000000G  STRFN:  JSR     PC,@#EVAL
001572' 103432                  BCS     ERSTRA
001574' 122127 000000G          CMPB    (R1)+,#,RPAR
001600' 001031                  BNE     ERSTRS
001602' 012746 000020           MOV     #20,-(SP)
001606' 010502                  MOV     R5,R2
001610' 004737 000000G          JSR     PC,@#MAKESTR
001614' 011603                  MOV     (SP),R3
001616' 062703 000003           ADD     #3,R3
001622' 010365 000000G          MOV     R3,T2(R5)
001626' 005065 000000G          CLR     T1(R5)
001632' 004737 000000G          JSR     PC,@#NUMSGN
001636' 000000G                 .WORD   SAVCHAR
001640' 010602                  MOV     SP,R2
001642' 016546 000000G          MOV     T1(R5),-(SP)
001646' 004737 000000G          JSR     PC,@#MAKESTR
001652' 012616                  MOV     (SP)+,(SP)
001654' 000137 000000G          JMP     @#STPRO         ;PROTECT STRING

001660' 000137 000000G  ERSTRA: JMP     @#ERRARG
001664' 000137 000000G  ERSTRS: JMP     @#ERRSYN
                        STRFNE:
                        FNEND:
```

365

```
                                        ;
                                        ;
                                        ; USER AREA STORAGE CELLS
                                        ;
001670' 000000          USR1    ,WORD   0               ;SYMBOLS
001672' 000000                  ,WORD   0               ;LIMIT
001674' 000000                  ,WORD   0               ;POL
001676' 000000                  ,WORD   0               ;PDSIZE
001700' 000000                  ,WORD   0               ;ARRAYS
001702' 000000                  ,WORD   0               ;HIFREE
001704' 000000                  ,WORD   0               ;LOFREE
001706' 002214'                 ,WORD   USRCODE         ;CODE
001710' 002074'                 ,WORD   USRLINE         ;LINE
001712' 000000                  ,WORD   0               ;VARSAV
001714' 000000                  ,WORD   0               ;SS1SAV
001716' 000000                  ,WORD   0               ;SS2SAV
001720' 000000                  ,WORD   0               ;LINENO
001722' 000000                  ,WORD   0               ;GSUBCTR
001724' 000000                  ,WORD   0               ;COLUMN
001726' 000000                  ,WORD   0               ;CLMNTTY
001730' 000000                  ,WORD   0               ;FAC1
001732' 000000                  ,WORD   0               ;FAC2
001734' 000000                  ,WORD   0               ;R0SAVE
001736' 000000                  ,WORD   0               ;R1SAVE
001740' 000000                  ,WORD   0               ;R2SAVE
001742' 000000                  ,WORD   0               ;R3SAVE
001744' 000000                  ,WORD   0               ;R4SAVE
001746' 000000                  ,WORD   0               ;T1
001750' 000000                  ,WORD   0               ;T2
001752' 000000                  ,WORD   0               ;T3
001754' 000000                  ,WORD   0               ;RND1
001756' 000000                  ,WORD   0               ;RND2
001760' 000000          RAND1   ,WORD   0               ;RNDCT
001762' 000000                  ,WORD   0               ;LOSTR
001764' 000000                  ,WORD   0               ;HISTR
001766'     000                 ,BYTE   0               ;ODEV
001767'     000                 ,BYTE   0               ;IDEV
001770' 002040'                 .       TPBFW1          ;TPHD
001772' 002004'                 .       KBBFW1          ;KBHD
001774' 000000                  ,WORD   0               ;ECHOSP
001776'     000                 ,BYTE   0               ;CNCFLG
001777'     000                 ,BYTE   0               ;CNOFLG
002000'     000                 ,BYTE   0               ;FILLCO
002001'     000                 ,BYTE   0               ;FILLNO
002002'     000                 ,BYTE   0               ;FILLCH
002003'     000                 ,BYTE   0               ; [SPARE BYTE]
```

366


```
                                        ;
                                        ; USER I/O BUFFER AND BUFFER HDR, SPACE (TTY)
                                        ;
002004' 002020'         KBBFW1: .       KBBUF1          ;BSTPT
002006' 002037'                 .       KBBEN1          ;BEND
002010' 002020'                 ,WORD   KBBUF1          ;BGET1
002012' 002020'                 ,WORD   KBBUF1          ;BGET2
002014' 002020'                 ,WORD   KBBUF1          ;BPUT
002016' 000000                  .       0               ;BFSPEC
                        KBBUF1  =.
                                =.+$KBBSZ               ;DEFAULT SIZE = 20 CHARS.
                        KBBEN1  =.-1
                                .EVEN
                                ;
002040' 002054'         TPBFW1: .       TPBUF1          ;BSTPT
002042' 002073'                 .       TPBEN1          ;BEND
002044' 000000                  .       0               ;BGET1 (NOT USED)
002046' 002054'                 ,WORD   TPBUF1          ;BGET2
002050' 002054'                 ,WORD   TPBUF1          ;BPUT
002052' 000000                  .       0               ;BFSPEC
                        TPBUF1  =.
                                =.+$TPBSZ               ;DEFAULT = 20 CHARS.
                        TPBEN1  =.-1
                                .EVEN
                        USRLINE:
                        .=.+$ULNSP                      ;DEFAULT = 120
002214'                 USRCODE:
```

367

```
                                        ;
                                        ; --ONCE-ONLY INIT; CODE
                                        ;    FIND OUT HOW MUCH CORE AND EXPAND USRAREA ACCORDINGLY
                                        ;
002214' 162704  017770          DECCORE:SUB     #17770,R4       ;COME HERE ON TRAP FOR BAD MEM;
002220' 022626                          CMP     (SP)+,(SP)+
002222' 000412                          BR      TRYCORE         ;  REF'S, AND DEC, CORE SIZE
                                        ;
002224' 000005                  ONCEONLY:RESET
002226' 012706  003364'                 MOV     #TMPSTK,SP      ;TEMP; STACK
002232' 004767  000000G                 JSR     PC,FPMP         ;INIT FOR FPMP ROUTINES
002236' 012737  002214' 000004          MOV     #DECCORE,@#4    ;SET UP TRAP ADDR;
002244' 012704  160000                  MOV     #160000,R4      ;28K
002250' 005744                  TRYCORE:TST     -(R4)           ;TRAPS TO DECCORE IF BAD ADDR
002252' 012737  000006  000004          MOV     #6,@#4          ;FALLS THRU IF OK - RESTORE TRAP ADDR;
002260' 010467  001100                  MOV     R4,HICORE
                                        ;
                                        ;INITIALIZE TOP OF BASIC
002264' 012701  000002'                 MOV     #BASICH,R1
                                        ;PRINT HEADING AND READ ANSWER
002270' 004367  000756          INIHD:  JSR     R3,PRMSG
002274'    015     012                  .BYTE   15,12
002276' 040502  044523  020103          .ASCII  'BASIC V001A'
002304' 030126  030460     101
002311'    015     012                  .BYTE   15,12
002313'    117  052120  043040          .ASCII  'OPT FNS: A=ALL, N=NONE, I=IND'
002320' 051516  020072  020501
002326' 046101  026114  047040
002334' 047055  047117  026105
002342' 044440  044455  042116
002350'    015     012                  .BYTE   15,12
002352'    077                          .ASCII  '?'
002353'    000                          .BYTE   0
                                        .EVEN
002354' 004767  000722                  JSR     PC,RDANS
002360' 120027  000116                  CMPB    R0,#'N          ;CHECK FOR N
002364' 001512                          BEQ     INISAV
002366' 005002                          CLR     R2              ;CLEAR ALL FLAG
002370' 120027  000101                  CMPB    R0,#'A          ;CHECK FOR A
002374' 001002                          BNE     INITI
002376' 005202                          INC     R2              ;SET ALL FLAG
002400' 000415                          BR      INIALL
002402' 120027  000111          INITI:  CMPB    R0,#'I          ;CHECK FOR I
002406' 001330                          BNE     INIHD
                                        ;PRINT HEADING FOR INDIV FUNCTIONS
002410' 004367  000036                  JSR     R3,PRMSG
002414'    015     012     012          .BYTE   15,12,12
002417'    040  026531  042531          .ASCII  ' V=YES  N=NO'
002424'  020123  047040  047055
002432'    117
002433'    000                          .BYTE   0
002434' 012705  003374'         INIALL: MOV     #FNTAB,R5
002440' 012704  000000G                 MOV     #TABLE5,R4
                                        ;GET NEXT FUNCTION TABLE ADDRESS
002444' 011503                  INILP:  MOV     (R5),R3
```

368


```
002446' 100005                          BPL     INIFN
002450' 062705  000002                  ADD     #2,R5           ;SKIP A FUNCTION
002454' 062704  000002                  ADD     #2,R4
002460' 000771                          BR      INILP
                                        ;PRINT QUESTION FOR FUNCTION, GET ANSWER
002462' 020427  000000G         INIFN:  CMP     R4,#TBL5END
002466' 101051                          BHI     INISAV
002470' 020527  003544'                 CMP     R5,#FNTABE
002474' 103046                          BHIS    INISAV
002476' 005702                          TST     R2
002500' 001027                          BNE     INIDF
002502' 016567  000004  000014          MOV     4(R5),FNNAM1
002510' 016567  000006  000010          MOV     6(R5),FNNAM2
002516' 004367  000530                  JSR     R3,PRMSG
002522'    015     012                  .BYTE   15,12
002524' 000000          FNNAM1: .WORD   0
002526' 000000          FNNAM2: .WORD   0
002530' 035040     040                  .ASCII  ' ? '
002533'    000                          .BYTE   0
                                        .EVEN
002534' 004767  000542                  JSR     PC,RDANS
002540' 120027  000116                  CMPB    R0,#'N          ;CHECK FOR N
002544' 001002                          BNE     INITY
002546' 005724                          TST     (R4)+
002550' 000413                          BR      ININXT
002552' 120027  000131          INITY:  CMPB    R0,#'Y          ;CHECK FOR Y
002556' 001341                          BNE     INIFN
                                        ;DEFINE THE FUNCTION
002560' 010100                  INIDF:  MOV     R1,R0
002562' 162700  000000G                 SUB     #TABLE5,R0
002566' 010024                          MOV     R0,(R4)+        ;PLACE ADDRESS IN TABLE
002570' 012321                  ININV:  MOV     (R3)+,(R1)+
002572' 020365  000002                  CMP     R3,2(R5)
002576' 103774                          BLO     ININV
002600' 062705  000010          ININXT: ADD     #10,R5
002604' 020427  000000G                 CMP     R4,#TBL5END
002610' 101715                          BLOS    INILP
                                        ;SAVE TOP OF BASIC
002612' 010167  175102          INISAV: MOV     R1,USRAREA
                                        ;MOVE USER AREA TO TOP OF BASIC
002616' 012702  001070'                 MOV     #USR,R2
002622' 010203                          MOV     R2,R3           ;COMPUTE RELOCATION CONSTANT
002624' 160103                          SUB     R1,R3
002626' 020227  002214'         MOVLP:  CMP     R2,#USRCODE
002632' 103005                          BHIS    MOVDON
002634' 012221                          MOV     (R2)+,(R1)+
002636' 001773                          BEQ     MOVLP
002640' 160341                          SUB     R3,-(R1)        ;RELOCATE NON-ZERO ENTRIES
002642' 005721                          TST     (R1)+
002644' 000770                          BR      MOVLP
                                        ;
                                MOVDON:
```

369

```
                                        ;INITIALIZE USER AREA
002646' 016704 000512                   MOV     HICORE,R4       ;R4 IS HIGHEST ADDR, IN CORE
002652' 162704 000300                   SUB     #300,R4
002656' 016705 175110                   MOV     USRAREA,R5
002662' 010465 000000G                  MOV     R4,LIMIT(R5)
002666' 162704 000134                   SUB     #134,R4         ;POINTER SPACE
002672' 010465 000000G                  MOV     R4,POL(R5)      ;BASE OF STACK
002676' 162704 000000G                  SUB     #$STKSZ,R4      ;MAX, STACK SIZE
002702' 010465 000000G                  MOV     R4,ARRAYS(R5)   ;HIEST ARRAY STOR, LOC;
002706' 062704 000040                   ADD     #40,R4
002712' 010465 000000G                  MOV     R4,POSIZE(R5)
002716' 012765 032331 000000G           MOV     #32331,RND1(R5)
002724' 012765 163291 000000G           MOV     #163291,RND2(R5)

                                        ;DIALOGUE TO SET UP TERMINAL TYPE
002732' 005003                  ASKSER: CLR     R3
002734' 004767 000312                   JSR     R3,PRMSG
002740' 015     012     012             .BYTE   15,12,12
002743' 106 051501 020124               .ASCII  'FAST SER TERM?'
002750' 042523 020122 042524
002756' 046522  077
002761' 000                             .BYTE   0
                                        .EVEN
002762' 004767 000314                   JSR     PC,RDANS
002766' 120027 000116                   CMPB    R0,#'N
002772' 001457                          BEQ     ASKDON
002774' 120027 000131                   CMPB    R0,#'Y
003000' 001354                          BNE     ASKSER
003002' 005203                          INC     R3

003004' 004767 000242           ASKLA:  JSR     R3,PRMSG
003010'     015     012                 .BYTE   15,12
003012' 040514 030063 020040            .ASCII  'LA30 (300 BAUD)?'
003020' 030063 020060 040562
003026' 042125 037451
003032' 000                             .BYTE   0
003034'                                 .EVEN
003034' 004767 000242                   JSR     PC,RDANS
003040' 120027 000131                   CMPB    R0,#'Y
003044' 001432                          BEQ     ASKDON
003046' 120027 000116                   CMPB    R0,#'N
003052' 001354                          BNE     ASKLA
003054' 005203                          INC     R3

003056' 004767 000170           ASKVT:  JSR     R3,PRMSG
003062'     015     012                 .BYTE   15,12
003064' 052126 032460 020040            .ASCII  'VT05 (>=600 BAUD)?'
003072' 036476 030066 020060
003100' 040562 042125 037451
003106' 000                             .BYTE   0
003110'                                 .EVEN
003110' 004767 000166                   JSR     PC,RDANS
003114' 120027 000131                   CMPB    R0,#'Y
003120' 001404                          BEQ     ASKDON
```

370


```
003122' 120027 000116                   CMPB    R0,#'N
003126' 001353                          BNE     ASKVT
003130' 000700                          BR      ASKSER

003132' 010302                  ASKDON: MOV     R3,R2
003134' 001412                          BEQ     FILLDON
003136' 006303                          ASL     R3
003140' 060302                          ADD     R3,R2
003142' 062702 003303'                  ADD     #FILLTB,R2
003146' 010503                          MOV     R5,R3
003150' 062703 000000G                  ADD     #FILLCO,R3
003154' 112223                          MOVB    (R2)+,(R3)+
003156' 112223                          MOVB    (R2)+,(R3)+
003160' 112223                          MOVB    (R2)+,(R3)+
                                FILLDON:
                                        ;PRINT OUT IF USER FUNCTIONS LOADED
003162' 005737 000046                   TST     @#46
003166' 001414                          BEQ     NOUSR
003170' 004367 000056                   JSR     R3,PRMSG
003174'     015     012     012         .BYTE   15,12,12
003177'     125 042523 020122           .ASCII  'USER FNS LOADED'
003204' 047106 020123 047514
003212' 042101 042105
003216' 000                             .BYTE 0
003220'                                 .EVEN
                                NOUSR:
003220' 004367 000026                   JSR     R3,PRMSG
003224'     015     012     012         .BYTE   15,12,12,0
003227'     000
                                        .EVEN

                                        ;CHECK IF RNDL LOADED
003230' 012702 000000G                  MOV     #TABLE5,R2
003234' 012203                          MOV     (R2)+,R3
003236' 001403                          BEQ     NORND
003240' 062703 000014                   ADD     #RNDFN-RNDLFN,R3
003244' 010312                          MOV     R3,(R2)         ;DEFINE RND ALSO
                                NORND:
003246' 000167 000000G                  JMP     START

                                        ;SUBROUTINE TO PRINT A MESSAGE
003252' 010300                  PRMSG:  MOV     R3,R0
003254' 105737 177776G          PRLP:   TSTB    @#TPB+2
003260' 100375                          BPL     PRLP
003262' 112037 000000G                  MOVB    (R0)+,@#TPB
003266' 001372                          BNE     PRLP
003270' 010003                          MOV     R0,R3
003272' 005203                          INC     R3
003274' 006203                          ASR     R3
003276' 006303                          ASL     R3
003300' 000203                          RTS     R3

                                        ;SUBROUTINE TO READ ANSWER FROM TELETYPE
003302' 005267 176452           RDANS:  INC     RAND
003306' 105737 000000G                  TSTB    @#TKB
```

371

```
747  003312' 100373                          BPL     ROANS
748  003314' 113700 0000020                  MOVB    @@TKS+2,R0
749  003320' 042700 177600                   BIC     #177600,R0
750  003324' 010067 000004                   MOV     R0,INCH
751  003330' 004367 177710                   JSR     R3,PRMSG
752  003334' 000000                  INCH:   .WORD   0
753  003336' 016700 177772                   MOV     INCH,R0
754  003342' 000207                          RTS     PC
755          003364'                         .=.+20
756                                  TMPSTCK:
757  003364' 000000                  HICORE: .WORD   0
758
759          003363'                 FILLTB: .=.-3
760  003366'    000     011     015          .BYTE   0,11,15         ;LA30
761  003371'    000     004     012          .BYTE   0,04,12         ;VT05
762
763                                  FNTAB:
764  003374' 000002'                         .WORD   RNDLEN
765  003376' 000140'                         .WORD   RNDFNC
766  003400' 047122  020104                  .ASCII  'RND '
767  003404' 177777                          .WORD   -1              ; RND
768  003406' 177777                          .WORD   -1              ; SIN
769  003410' 177777                          .WORD   -1              ; COS
770  003412' 177777                          .WORD   -1              ; SQR
771  003414' 177777                          .WORD   -1              ; ATN
772  003416' 177777                          .WORD   -1              ; EXP
773  003420' 177777                          .WORD   -1              ; LOG
774  003422' 000140'                         .WORD   ABSFN           ; ABS
775  003424' 000236'                         .WORD   ABSFNC
776  003426' 041101  020123                  .ASCII  'ABS '
777  003432' 177777                          .WORD   -1              ; INT
778  003434' 000236'                         .WORD   SGNFN           ; SGN
779  003436' 000324'                         .WORD   SGNFNC
780  003440' 043523  020116                  .ASCII  'SGN '
781  003444' 000324'                         .WORD   TABFN           ; TAB
782  003446' 000436'                         .WORD   TABFNC
783  003450' 040524  020102                  .ASCII  'TAB '
784  003454' 000436'                         .WORD   LENFN           ; LEN
785  003456' 000510'                         .WORD   LENFNC
786  003460' 042514  020116                  .ASCII  'LEN '
787  003464' 000510'                         .WORD   ASCFN           ; ASC
788  003466' 000574'                         .WORD   ASCFNC
789  003470' 051501  020103                  .ASCII  'ASC '
790  003474' 000574'                         .WORD   CHRSFN          ; CHR$
791  003476' 000646'                         .WORD   CHRFNC
792  003500' 044103  022122                  .ASCII  'CHR$'
793  003504' 000646'                         .WORD   POSFN           ; POS
794  003506' 001152'                         .WORD   POSFNC
795  003510' 047520  020123                  .ASCII  'POS '
796  003514' 001152'                         .WORD   SEGFN           ; SEG
797  003516' 001420'                         .WORD   SEGFNC
798  003520' 042523  022107                  .ASCII  'SEG$'
799  003524' 001420'                         .WORD   VALFN           ; VAL
800  003526' 001566'                         .WORD   VALFNC
801  003530' 040526  020114                  .ASCII  'VAL '
```

372


```
802  003534' 001566'                         .WORD   STRFN           ; STR
803  003536' 001670'                         .WORD   STRFNC
804  003540' 052123  022122                  .ASCII  'STR$'
805                                  FNTABE:
806
807          002224'                         .END    ONCEONL
```

373

```
ABSFN   000146R     ABSFNE  000236R     ABSINT  000174R     ABSX    000222R
ARGB    = ****** G  ARRAYS  = ****** G  ASCFN   000510R     ASCFNE  000574R
ASKDON  003132R     ASKLA   003004R     ASKSER  002732R     ASKVT   003056R
BASICH  000002R     BL      = 000040    CHRFNE  000646R     CHRSFN  000574R
COLUMN  = ****** G  DECCOR  002214R     ERABSA  000226R     ERABSS  000232R
ERASCA  000564R     ERASCS  000570R     ERCHRS  000642R     ERLENA  000900R
ERLENS  000504R     ERPOSA  001142R     ERPOSS  001146R     ERRARG  = ****** G
ERRMIX  = ****** G  ERRNDA  000134R     ERRNDS  000130R     ERRPDL  = ****** G
ERRSYN  = ****** G  ERSEGA  001418R     ERSEGS  001414R     ERSGNA  000314R
ERSGNS  000320R     ERSTRA  001660R     ERSTRS  001604R     ERTABS  000432R
ERVALA  001562R     ERVALS  001556R     EVAL    = ****** G  FAC1    = ****** G
FAC2    = ****** G  FILLCO  = ****** G  FILLDO  003102R     FILLTB  = 003363R
FNEND   001670R     FNNAM1  002924R     FNNAM2  002926R     FNTAB   003374R
FNTABE  003544R     HICORE  003364R     IFPMP   = ****** G  INCH    003334R
INIALL  002434R     INIOF   002560R     ININFN  002462R     ININHD  002270R
INILP   002444R     ININV   002970R     ININXT  002600R     INISAV  002612R
INITI   002402R     INITY   002552R     INT     = ****** G  KBBEN1  = 002037R
KBBFH1  002004R     KBBUF1  = 002020R     LENFN   000436R     LENFNE  000910R
LIMIT   = ****** G  MAKEST  = ****** G  MOVDON  002646R     MOVLP   002626R
NORM    = ****** G  NORND   003246R     NOUSR   003220R     NUMSGN  = ****** G
ONCEON  002224R     OPRATO  = ****** G  PC      =%000007    PDL     = ****** G
POSIZE  = ****** G  POSF    001124R     POSFN   000646R     POSFNE  001152R
POSFST  001058R     POSF2   001126R     POSNO   001116R     POSNXT  001076R
POSREM  001078R     POSTRY  001054R     POSX    000774R     POSY    001000R
PRLP    003254R     PRMSG   003252R     RAND    001700R     RDANS   003302R
REXP    000074R     RNDFN   000010R     RNDFNE  000140R     RNDFN2  000010R
RNDLFN  000002R     RND1    = ****** G  RND2    = ****** G  RNORM   000062R
RPLUS   000052R     R0      =%000000    R1      =%000001    R2      =%000002
R3      =%000003    R4      =%000004    R5      =%000005    SAVCHA  = ****** G
SEGFN   001152R     SEGFNE  001420R     SEGL    001256R     SEGLP   001366R
SEGNUL  001400R     SEGRNG  001312R     SEGTY   001278R     SEGTY2  001304R     SEGX    001404R     SEGXR   001320R     SGNFLT  000270R
SGNFN   000236R     SGNFNE  000324R     SGNPOS  000276R     SGNX    000310R
SOPRAT  = ****** G  SP      =%000006    START   = ****** G  STPRO   = ****** G
STRFN   001566R     STRFNE  001670R     TABB    000344R     TABC    000360R
TABFN   000324R     TABFNE  000436R     TABLES  = ****** G  TABNON  000370R
TABNUL  000426R     TBLSEN  = ****** G  TKS     = ****** G  TMPSTC  003364R
TPB     = ****** G  TPBEN1  = 002073R     TPBFH1  002040R     TPBUF1  = 002054R
TRYCOR  002250R     T1      = ****** G  T2      = ****** G  T3      = ****** G
USR     001670R     USRARE  000000RG    USRCOD  002214R     USRLIN  002074R
VAL     = ****** G  VALCE   001524R     VALFN   001420R     VALFNE  001360R
VALJ    001552R     VALR    001460R     VALX    001536R     %KBBSZ  = 000020
%STKSZ  = ****** G  %TPBSZ  = 000020    %ULNSP  = 000120    ,COMMA  = ****** G
,RPAR   = ****** G  .       = 003044R
```

ERRORS DETECTED: 0

374

RUN-TIME: 18 SECONDS
CORE USED: 3K

375

379

```
                                ; BASIC/PTS  PART2-FPMP
                                ;
                                ; DEC-11-LPTBA-A-LA2
                                ;
                                ; COPYRIGHT 1973
                                ;
                                ; DIGITAL EQUIPMENT CORPORATION
                                ; MAYNARD, MASSACHUSETTS 01754
```



```
                                ;                                BASIC SOURCE FILE #9
                                ; $BAS - FPMP FOR BASIC
          000001                $BAS=1
          000001                CND$2=1
          000001                CND$3=1
          000001                CND$18=1
          000001                CND$28=1
          000001                CND$27=1
          000001                CND$38=1

                                        .GLOBL  ERRFPU ,IFPMP

                                ;INITIALIZE FPMP ROUTINE
                                IFPMP:
                                        .IFDF   FPU
                                        .WORD   170127          ;LDFPS #1000
                                        .WORD   1000
                                        .ENDC
000000' 000207                          RTS     PC

                                ;FPU ERROR INTERRUPT ROUTINE
                                ERRFPU:
                                        .IFDF   FPU
                                        .WORD   170127          ;LDFPS #1000
                                        .WORD   1000
                                        JMP     @#SERVEC
                                        .ENDC

                                        .IFNDF  MIN
          000001                CND$37=1
          000001                CND$39=1
          000001                CND$41=1
                                        .ENDC

                                ;;;;;;;;;;;;;;;;;;
                                ;FROM HERE ON THIS IS THE SAME AS FPMP.PTX
                                ;;;;;;;;;;;;;;;;;;
                                ;PRODUCT CODE           DEC-11-NFPMA-A-LA
                                ;COMPUTER               PDP-11
                                ;CONFIGURATION          PAPER TAPE CONFIGURATION IS MINIMUM
                                ;                       8192 WORDS MEMORY
                                ;SOFTWARE REQUIREMENTS  PAL-11S (OR MACRO-11)
                                ;                       LINK-11S (OR LINK-11)
                                ;PROGRAM NAME           FPMP-11
                                ;VERSION                VERSION LEVEL   1
                                ;                       PATCH LEVEL     A
                                ;DESCRIPTION            FLOATING POINT MATH PACKAGE
                                ;                       PLUS TRAP HANDLER
                                ;                       (FLOATING POINT SUBROUTINES TAKEN FROM
                                ;                       DOS-11 FORTRAN IV OTS)
                                ;AUTHOR                 C. PETERS (TRAP HANDLER & PACKAGE
                                ;                                  INTEGRATION)
                                ;DATE                   AUGUST, 1972
                                ;                       COPYRIGHT 1972, DIGITAL EQUIPMENT CORP.;
                                ;                       MAYNARD, MASSACHUSETTS 01754
```

```
                                .CSECT
                                .IFDF   SINGLE
                                CND$2=1
                                CND$3=1
                                CND$4=1
                                CND$6=1
                                CND$18=1
                                CND$20=1
                                CND$36=1
                                CND$37=1
                                CND$38=1
                                CND$39=1
                                CND$41=1
                                CND$44=1
                                CND$46=1
                                .ENDC
                                .IFDF   DOUBLE
                                CND$1=1
                                CND$5=1
                                CND$10=1
                                CND$13=1
                                CND$14=1
                                CND$15=1
                                CND$16=1
                                CND$19=1
                                CND$28=1
                                CND$45=1
                                CND$47=1
                                .ENDC
                                .IFDF   SINGLE!DOUBLE
                                CND$8=1
                                CND$9=1
                                CND$31=1
                                CND$42=1
                                .ENDC
                                .IFDF   CND$38
                                CND$2=1
                                CND$18=1
                                CND$20=1
                                CND$21=1
                                CND$36=1
                                CND$32=1
                                .ENDC
                                .IFNDF  FPU
                                .IFDF   CND$3!CND$20!CND$37!CND$39
000001                          CND$2=1
000001                          CND$18=1
000001                          CND$36=1
                                .IFDF   CND$37
000001                          CND$4=1
                                .ENDC
                                .ENDC
                                .IFDF   CND$10!CND$13!CND$15!CND$19
                                CND$1=1
                                CND$16=1
```

382


```
                                CND$28=1
                                CND$33=1
                                .IFDF   CND$13
                                CND$11=1
                                .ENDC
                                .ENDC
                                .IFDF   CND$3!CND$10
000001                          CND$27=1
                                .ENDC
                                .IFDF   CND$19!CND$20
000001                          CND$27=1
000001                          CND$35=1
                                .ENDC
                                .IFDF   CND$14
                                CND$1=1
                                CND$16=1
                                .ENDC
                                .IFDF   CND$41
000001                          CND$2=1
000001                          CND$18=1
                                .ENDC
                                .ENDC
                                .IFDF   CND$23
                                CND$27=1
                                CND$33=1
                                .ENDC
                                .IFDF   CND$22!CND$26
                                CND$35=1
                                .ENDC
                                .IFDF   CND$39
000001                          CND$33=1
                                .ENDC
                                .TITLE  TRAP02
                                .IFDF   CND$42
                                .GLOBL  TRAPH,$ERRA
                                R0=%0
                                R1=%1
                                R2=%2
                                R3=%3
                                R4=%4
                                R5=%5
                                SP=%6
                                PC=%7
                        TRAPH:  BIC     #17,2(SP)
                                CLR     -(SP)
                                MOV     R5,-(SP)
                                MOV     R4,-(SP)
                                MOV     R3,-(SP)
                                MOV     R2,-(SP)
                                MOV     R1,-(SP)
                                MOV     R0,-(SP)
                                MOV     20(SP),R3
                                BIC     #20,R3
                                MOV     R3,@#177776
                                MOV     10(SP),R1
```

383

```
                                        MOV     R1,R5
                                        MOV     -(R1),R4
                                        MOV     R4,R3
                                        BIC     #177700,R4
                                        ASL     R4
                                        MOV     TBL$42(R4),R4
                                        BEQ     ERR$42
                                        MOV     R4,R2
                                        BIC     #140000,R2
                                        ADD     PC,R2
                                PTS42:  BIT     #40000,R4
                                        BEQ     NAD$42
                                        ROLB    R3
                                        BPL     PLM$42
                                        BCC     STK$42
                                        MOV     R5,R0
                                        ADD     (R5)+,R0
                                UPC$42: MOV     R5,16(SP)
                                STK$42: MOV     #FAC$42+6,R5
                                        TST     R4
                                        BLT     ST4$42
                                        CLR     @R5
                                        CLR     -(R5)
                                        TST     (R5)+
                                ST4$42: MOV     @R5,-(SP)
                                        MOV     -(R5),-(SP)
                                        MOV     -(R5),-(SP)
                                        MOV     -(R5),-(SP)
                                        TST     R4
                                        BLT     ST6$42
                                        CMP     (R0)+,(R0)+
                                        BR      OT2$42
                                ST6$42: ADD     #8.,R0
                                        MOV     -(R0),-(SP)
                                        MOV     -(R0),-(SP)
                                OT2$42: MOV     -(R0),-(SP)
                                        MOV     -(R0),-(SP)
                                        MOV     #ADR$42,R4
                                        JMP     @R2
                                ADR$42: ,WORD   .+2
                                        MOV     #FAC$42,R5
                                        MOV     (SP)+,(R5)+
                                        MOV     (SP)+,(R5)+
                                        MOV     (SP)+,(R5)+
                                        MOV     (SP)+,(R5)+
                                RET$42: MOV     @PC,R0
                                        MOV     #FAC$42,R5
                                        TST     (R5)+
                                        BLT     NEG$42
                                        BGT     PLS$42
                                        TST     (R5)+
                                        BNE     PLS$42
                                        TST     (R5)+
                                        BNE     PLS$42
                                        TST     (R5)+
```

384


```
                                        BNE     PLS$42
                                        CLR     R0
                                NEG$42: NEG     R0
                                CMP$42: BIS     @#177776,20(SP)
                                PLS$42: TST     R0
                                CMI$42: MOV     (SP)+,R0
                                        MOV     (SP)+,R1
                                        MOV     (SP)+,R2
                                        MOV     (SP)+,R3
                                        MOV     (SP)+,R4
                                        MOV     (SP)+,R5
                                        TST     (SP)+
                                        BEQ     RTI$42
                                        BPL     RT2$42
                                        MOV     (SP)+,6(SP)
                                        MOV     (SP)+,6(SP)
                                        CMP     (SP)+,(SP)+
                                        RTI
                                RT2$42: MOV     (SP)+,2(SP)
                                        MOV     (SP)+,2(SP)
                                RTI$42: RTI
                                NAD$42: JSR     R5,@R2
                                        MOV     (PC)+,R5
                                        ,WORD   FAC$42
                                        MOV     R0,(R5)+
                                        MOV     R1,(R5)+
                                        MOV     R2,(R5)+
                                        MOV     R3,(R5)+
                                        BR      RET$42
                                PLM$42: BCC     STM$42
                                        MOV     R5,R0
                                        TST     R4
                                        BGE     PLI$42
                                        ADD     #8.,R0
                                        BR      UPC$42
                                PLI$42: CMP     (R5)+,(R5)+
                                        BR      UPC$42
                                STM$42: MOV     SP,R0
                                        ADD     #22,R0
                                        INC     14(SP)
                                        TST     R4
                                        BGE     STK$42
                                        NEG     14(SP)
                                        BR      STK$42
                                ERR$42: CLR     R0
                                        JSR     R5,$ERRA
                                        BR      ERR$42
                                FAC$42: ,WORD   0,0,0,0
                                        ,IFDF   CND$6

                                CMR$42: MOV     #CAR$42,R4
                                        JMP     $CMR
                                CAR$42: ,WORD   .+2
                                        BIS     @#177776,24(SP)
                                        CMP     (SP)+,(SP)+
```

385

```
                                        BR      CM1$42
                                        .ENDC
                                        .IFDF   CND$5
                                CMD$42: MOV     #CAD$42,R4
                                        JMP     $CMD
                                CAD$42: .WORD   CMF$42
                                        .ENDC
                                        PMODE=40000
                                        DMODE=100000
                                TBL$42: .WORD   0,0,0,0,0,0,0,0
                                        .WORD   0,0
                                        .IFDF   CND$2
                                        .WORD   $ADR-PT$42+PMODE
                                        .WORD   $SBR-PT$42+PMODE
                                        .ENDC
                                        .IFNDF  CND$2
                                        .WORD   0,0
                                        .ENDC
                                        .IFDF   CND$1
                                        .WORD   $ADD-PT$42+PMODE+DMODE
                                        .WORD   $SBD-PT$42+PMODE+DMODE
                                        .ENDC
                                        .IFNDF  CND$1
                                        .WORD   0,0
                                        .ENDC
                                        .IFDF   CND$5
                                        .WORD   CMD$42-PT$42+PMODE+DMODE
                                        .ENDC
                                        .IFNDF  CND$5
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$6
                                        .WORD   CMR$42-PT$42+PMODE
                                        .ENDC
                                        .IFNDF  CND$6
                                        .WORD   0
                                        .ENDC
                                        .WORD   0
                                        .IFDF   CND$3
                                        .WORD   $MLR-PT$42+PMODE
                                        .ENDC
                                        .IFNDF  CND$3
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$2
                                        .WORD   $MLD-PT$42+PMODE+DMODE
                                        .ENDC
                                        .IFNDF  CND$2
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$1
                                        .WORD   $DVD-PT$42+PMODE+DMODE
                                        .ENDC
                                        .IFNDF  CND$1
                                        .WORD   0
```

386


```
                                        .ENDC
                                        .WORD   0
                                        .IFDF   CND$1
                                        .WORD   $QVR-PT$42+PMODE
                                        .ENDC
                                        .IFNDF  CND$1
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$4
                                        .WORD   AINT-PT$42
                                        .ENDC
                                        .IFNDF  CND$4
                                        .WORD   0
                                        .ENDC
                                        .WORD   0,0,0,0,0,0,0
                                        .IFDF   CND$37
                                        .WORD   SIN-PT$42,COS-PT$42
                                        .ENDC
                                        .IFNDF  CND$37
                                        .WORD   0,0
                                        .ENDC
                                        .IFDF   CND$13
                                        .WORD   DSIN-PT$42+DMODE
                                        .WORD   DCOS-PT$42+DMODE
                                        .ENDC
                                        .IFNDF  CND$13
                                        .WORD   0,0
                                        .ENDC
                                        .IFDF   CND$39
                                        .WORD   ATAN-PT$42
                                        .ENDC
                                        .IFNDF  CND$39
                                        .WORD   0
                                        .ENDC
                                        .WORD   0
                                        .IFDF   CND$15
                                        .WORD   DATAN-PT$42+DMODE
                                        .ENDC
                                        .IFNDF  CND$15
                                        .WORD   0
                                        .ENDC
                                        .WORD   0
                                        .IFDF   CND$41
                                        .WORD   SQRT-PT$42
                                        .ENDC
                                        .IFNDF  CND$41
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$14
                                        .WORD   DSQRT-PT$42+DMODE
                                        .ENDC
                                        .IFNDF  CND$14
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$38
```

387

```
                                        .WORD   TANH-PTS42
                                        .ENDC
                                        .IFNDF  CND$38
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$28
                                        .WORD   EXP-PTS42
                                        .ENDC
                                        .IFNDF  CND$28
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$19
                                        .WORD   DEXP-PTS42+DMODE
                                        .ENDC
                                        .IFNDF  CND$19
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$3
                                        .WORD   ALOG-PTS42
                                        .WORD   ALOG10-PTS42
                                        .ENDC
                                        .IFNDF  CND$3
                                        .WORD   0,0
                                        .ENDC
                                        .IFDF   CND$10
                                        .WORD   DLOG-PTS42+DMODE
                                        .WORD   DLOG10-PTS42+DMODE
                                        .ENDC
                                        .IFNDF  CND$10
                                        .WORD   0,0
                                        .ENDC
                                        .WORD   0,0,0,0,0,0,0,0,0,0
                                        .IFDF   CND$44
                                        .WORD   SLOR-PTS42+PMODE
                                        .ENDC
                                        .IFNDF  CND$44
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$45
                                        .WORD   SLOD-PTS42+PMODE+DMODE
                                        .ENDC
                                        .IFNDF  CND$45
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$40
                                        .WORD   $STR-PTS42+PMODE
                                        .ENDC
                                        .IFNDF  CND$40
                                        .WORD   0
                                        .ENDC
                                        .IFDF   CND$47
                                        .WORD   $STD-PTS42+PMODE+DMODE
                                        .ENDC
                                        .IFNDF  CND$47
                                        .WORD   0
```

388


```
                                        .ENDC
                                        .WORD   0,0,0
                                        .ENDC
                                        .TITLE  $ADD09
                                        .IFDF   CND$1
                                        .GLOBL  $ADD,$SBD,$ERR
                                        R0=%0
                                        R1=%1
                                        R2=%2
                                        R3=%3
                                        R4=%4
                                        R5=%5
                                        SP=%6
                                        PC=%7
                                        A1=6
                                        B1=8.
                                        C1=10.
                                        D1=12.
                                        A2=14.
                                        B2=16.
                                        C2=18.
                                        D2=20.
                                        SIGNS=0.
                                        MQ=177304
                                        NOR=177312
                                        LSH=177314
                                        ASH=177310
                                        F0=%0
                                $SBD:   ADD     #100000,@SP
                                        .IFDF   FPU
                                $ADD:   .WORD   170011
                                        .WORD   172420
                                        .WORD   172020
                                        .WORD   174040
                                        JMP     @(R4)+
                                        .ENDC
                                        .IFNDF  FPU
                                $ADD:   MOV     R4,-(SP)
                                        MOV     R5,-(SP)
                                        CLR     -(SP)
                                        CLR     R4
                                        CLR     R5
                                        ASL     D1(SP)
                                        ROL     C1(SP)
                                        ROL     B1(SP)
                                        ROL     A1(SP)
                                        BISB    A1+1(SP),R4
                                        BEQ     A1ES1
                                        ROLB    @SP
                                        ASL     D2(SP)
                                        ROL     C2(SP)
                                        ROL     B2(SP)
                                        ROL     A2(SP)
                                        BISB    A2+1(SP),R5
                                        BNE     A2NS1
```

389

```
                                        RORB    @SP
                                        ROR     A1(SP)
                                        ROR     B1(SP)
                                        ROR     C1(SP)
                                        ROR     D1(SP)
                                        MOV     A1(SP),A2(SP)
                                        MOV     B1(SP),B2(SP)
                                        MOV     C1(SP),C2(SP)
                                        MOV     D1(SP),D2(SP)
                                A1ES1:  TST     (SP)+
                                        JMP     OUTS1
                                A2NS1:  ROLB    SIGNS+1(SP)
                                        MOVB    #1,A2+1(SP)
                                        MOVB    #1,A1+1(SP)
                                        SUB     R4,R5
                                        BGT     EXAS1
                                        MOV     A2(SP),R0
                                        MOV     B2(SP),R1
                                        MOV     C2(SP),R2
                                        MOV     D2(SP),R3
                                        BR      SCKS1
                                EXAS1:  ADD     R5,R4
                                        MOV     A1(SP),R0
                                        MOV     B1(SP),R1
                                        MOV     C1(SP),R2
                                        MOV     D1(SP),R3
                                        MOV     A2(SP),A1(SP)
                                        MOV     B2(SP),B1(SP)
                                        MOV     C2(SP),C1(SP)
                                        MOV     D2(SP),D1(SP)
                                        SWAB    @SP
                                        NEG     R5
                                SCKS1:  CMPB    SIGNS+1(SP),@SP
                                        BEQ     ECKS1
                                        NEG     R3
                                        ADC     R2
                                        ADC     R1
                                        ADC     R0
                                        NEG     R2
                                        ADC     R1
                                        ADC     R0
                                        NEG     R1
                                        ADC     R0
                                        NEG     R0
                                ECKS1:  TST     R5
                                        BEQ     SFDS1
                                SFTS1:  CMP     #-27.,R5
                                        BLE     SFRS1
                                        MOV     A1(SP),R0
                                        MOV     B1(SP),R1
                                        MOV     C1(SP),R2
                                        MOV     D1(SP),R3
                                        BR      NODS1
                                SFRS1:  CMP     #-8.,R5
                                        BLE     SR8S1
```

390


```
                                        .IFNDF  MULDIV
                                        CLR     -(SP)
                                        TST     R0
                                        @PL     SF1S1
                                        COM     @SP
                                        .ENDC
                                        .IFDF   MULDIV
                                        TST     R0
                                        .WORD   @@6746
                                        .ENDC
                                SF1S1:  CMP     #-16.,R5
                                        BLT     S16S1
                                        MOV     R2,R3
                                        MOV     R1,R2
                                        MOV     R0,R1
                                        MOV     @SP,R0
                                        ADD     #16.,R5
                                        BNE     SF1S1
                                        TST     (SP)+
                                        BR      SFDS1
                                        .IFDF   EAE
                                S16S1:  CMP     #-3,R5
                                        BLE     S8AS1
                                        MOV     R4,@SP
                                        MOV     #MQ,R4
                                        MOV     R3,@R4
                                        MOV     R2,-(R4)
                                        MOV     R5,@@LSH
                                        MOV     (R4)+,R2
                                        MOV     @R4,R3
                                        CLR     @R4
                                        MOV     R1,-(R4)
                                        MOV     R5,@@LSH
                                        TST     (R4)+
                                        BIS     @R4,R2
                                        MOV     R1,@R4
                                        MOV     R0,-(R4)
                                        MOV     R5,@@ASH
                                        MOV     (R4)+,R0
                                        MOV     @R4,R1
                                        MOV     (SP)+,R4
                                        BR      SFDS1
                                        .ENDC
                                        .IFNDF  EAE&MULDIV
                                S16S1:  CMP     #-8.,R5
                                        BLE     S8AS1
                                        ADD     #16.,R5
                                SL8S1:  ASL     R3
                                        ROL     R2
                                        ROL     R1
                                        ROL     R0
                                        ROL     @SP
                                        DEC     R5
                                        BGT     SL8S1
                                        MOV     R2,R3
```

391

```
                                        MOV     R1,R2
                                        MOV     R0,R1
                                        MOV     (SP)+,R0
                                        BR      SFDS1
                                        .ENDC
                                        .IFDF   MULDIV
                                S16S1:  CMP     #-3,R5
                                        BLE     S8AS1
                                        MOV     R4,@SP
                                        MOV     R5,-(SP)
                                        MOV     R1,R4
                                        .WORD   073005
                                        MOV     R2,R5
                                        .WORD   073410
                                        MOV     R2,R4
                                        MOV     R5,R2
                                        MOV     R3,R5
                                        .WORD   073420
                                        MOV     R5,R3
                                        MOV     (SP)+,R4
                                        BR      SFDS1
                                        .ENDC
                                S8AS1:  TST     (SP)+
                                SR8S1:  ASR     R0
                                        ROR     R1
                                        ROR     R2
                                        ROR     R3
                                        INC     R5
                                        BLT     SR8S1
                                SFDS1:  ADD     D1(SP),R3
                                        ADC     R2
                                        ADC     R1
                                        ADC     R0
                                        ADD     C1(SP),R2
                                        ADC     R1
                                        ADC     R0
                                        ADD     B1(SP),R1
                                        ADC     R0
                                        ADD     A1(SP),R0
                                        CMPB    SIGNS+1(SP),@SP
                                        BNE     SUBS1
                                        BIT     R0,#1000
                                        BEQ     NODS1
                                        ASR     R0
                                        ROR     R1
                                        ROR     R2
                                        ROR     R3
                                        INC     R4
                                NODS1:  SWAB    R4
                                        BNE     OVFS1
                                NFLS1:  BISB    R0,R4
                                        ROR     (SP)+
                                        ROR     R4
                                        ROR     R1
                                        ROR     R2
```

392


```
                                        ROR     R3
                                        ADC     R3
                                        ADC     R2
                                        ADC     R1
                                        ADC     R4
                                        BVS     OVRS1
                                        BCS     OVRS1
                                        MOV     R4,A2+6-2(SP)
                                        MOV     R1,B2+6-2(SP)
                                        MOV     R2,C2+6-2(SP)
                                        MOV     R3,D2+6-2(SP)
                                OUTS1:  MOV     (SP)+,R5
                                        MOV     (SP)+,R4
                                        ADD     #8.,SP
                                        JMP     @(R4)+
                                OVFS1:  TST     (SP)+
                                OVRS1:  JSR     R5,$ERR
                                        BR      OUTS1
                                        .BYTE   3
                                        .BYTE   1
                                UTSS1:  TST     R4
                                        BGT     NODS1
                                UNFS1:  JSR     R5,$ERR
                                        BR      UNOS1
                                        .BYTE   5
                                        .BYTE   1
                                UNOS1:  CLR     R0
                                        CLR     R1
                                        CLR     R2
                                        CLR     R3
                                ZERS1:  CLR     @SP
                                        CLR     R4
                                        BR      NFLS1
                                SUBS1:  TST     R0
                                        BGT     BT9S1
                                        BEQ     ZTSS1
                                        NEG     R3
                                        ADC     R2
                                        ADC     R1
                                        ADC     R0
                                        NEG     R2
                                        ADC     R1
                                        ADC     R0
                                        NEG     R1
                                        ADC     R0
                                        SWAB    @SP
                                        NEG     R0
                                        BEQ     ZTSS1
                                BT9S1:
                                        .IFDF   EAE
                                        BIT     R0,#740
                                        BNE     B9AS1
                                        MOV     R4,-(SP)
                                        MOV     #R0,R4
                                        MOV     R1,@R4
```

393

```
                                        MOV     R0,-2(R4)
                                        CLR     @#NOR
                                        MOV     @#NOR,-(SP)
                                        SUB     #6,@SP
                                        MOV     R1,@R4
                                        MOV     R0,-(R4)
                                        MOV     @SP,@#LSH
                                        MOV     (R4)+,R0
                                        MOV     @R4,R1
                                        MOV     R2,@R4
                                        CLR     -(R4)
                                        MOV     @SP,@#LSH
                                        BIS     (R4)+,R1
                                        MOV     R3,@R4
                                        MOV     R2,-(R4)
                                        MOV     @SP,@#LSH
                                        MOV     (R4)+,R2
                                        MOV     @R4,R3
                                        SUB     (SP)+,@SP
                                        MOV     (SP)+,R4
                                        BGT     NOD$1
                                        BR      UNF$1
                                ;ENDC
                        B9A$1:  BIT     R0,#400
                                BNE     UI$$1
                                DEC     R4
                                ASL     R3
                                ROL     R2
                                ROL     R1
                                ROL     R0
                                BR      B9A$1
                        ZT3$1:  SUB     #0,,R4
                                TST     R1
                                BNE     ZT1$1
                                SUB     #16.,R4
                                MOV     R2,R1
                                BNE     ZT2$1
                                SUB     #16.,R4
                                TST     R3
                                BEQ     ZER$1
                                BISB    R3,R1
                                SWAB    R1
                                SWAB    R3
                                BISB    R3,R0
                                CLR     R3
                                BR      BT9$1
                        ZT2$1:  MOV     R3,R2
                                CLR     R3
                        ZT1$1:  SWAB    R1
                                BISB    R1,R0
                                CLRB    R1
                                SWAB    R2
                                BISB    R2,R1
                                CLRB    R2
                                SWAB    R3
```

394


```
                                        BISB    R3,R2
                                        CLRB    R3
                                        BR      BT9$1
                                ;ENDC
                                ;ENDC
                                ;TITLE  $ADR04
                                ;IFDF   GND$2
                                ;GLOBL  $ADR,$$BR,$ERR
                000000                  R0=%0
                000001                  R1=%1
                000002                  R2=%2
                000003                  R3=%3
                000004                  R4=%4
                000005                  R5=%5
                000006                  SP=%6
                000007                  PC=%7
                000000                  SIGNS=0
                000004                  A1=4
                000006                  B1=6
                000010                  A2=8.
                000012                  B2=10.
                177302                  AC=177302
                177304                  MQ=177304
                177312                  NOR=177312
                177316                  ASH=177316
                000000                  FP=%0
000002' 062716  100000          $$BR1:  ADD     #100000,@SP
                                ;IFDF   FPU
                                SADR1:  ;WORD   170001
                                ;WORD   172420
                                ;WORD   172020
                                ;WORD   174040
                                JMP     @(R4)+
                                ;ENDC
                                ;IFNDF  FPU
000006' 010446                  SADR1:  MOV     R4,-(SP)
000010' 005046                          CLR     -(SP)
000012' 005002                          CLR     R2
000014' 005003                          CLR     R3
000016' 006366  000006                  ASL     B1(SP)
000022' 006166  000004                  ROL     A1(SP)
000026' 156603  000005                  BISB    A1+1(SP),R3
000032' 001574                          BEQ     OUT$2
000034' 106116                          ROLB    @SP
000036' 006366  000012                  ASL     B2(SP)
000042' 006166  000010                  ROL     A2(SP)
000046' 156602  000011                  BISB    A2+1(SP),R2
000052' 001014                          BNE     A2N$2
000054' 106016                          RORB    @SP
000056' 006066  000004                  ROR     A1(SP)
000062' 006066  000006                  ROR     B1(SP)
000066' 016666  000004  000010          MOV     A1(SP),A2(SP)
000074' 016666  000006  000012          MOV     B1(SP),B2(SP)
000102' 000550                          BR      OUT$2
000104' 106166  000001          A2N$2:  ROLB    SIGNS+1(SP)
```

395

```
  835  000110' 112766  000001  000011          MOVB    #1,A2+1(SP)
  836  000116' 112766  000001  000005          MOVB    #1,A1+1(SP)
  837  000124' 160302                          SUB     R3,R2
  838  000126' 003005                          BGT     EXAS2
  839  000130' 016600  000010                  MOV     A2(SP),R0
  840  000134' 016601  000012                  MOV     B2(SP),R1
  841  000140' 000415                          BR      SCKS2
  842  000142' 060203                  EXAS2:  ADD     R2,R3
  843  000144' 016600  000004                  MOV     A1(SP),R0
  844  000150' 016601  000006                  MOV     B1(SP),R1
  845  000154' 016666  000010  000004          MOV     A2(SP),A1(SP)
  846  000162' 016666  000012  000006          MOV     B2(SP),B1(SP)
  847  000170' 000316                          SWAB    @SP
  848  000172' 005402                          NEG     R2
  849  000174' 126616  000001          SCKS2:  CMPB    SIGNS+1(SP),@SP
  850  000200' 001403                          BEQ     ECKS2
  851  000202' 005401                          NEG     R1
  852  000204' 005500                          ADC     R0
  853  000206' 005400                          NEG     R0
  854  000210' 005702                  ECKS2:  TST     R2
  855  000212' 001450                          BEQ     SFDS2
  856  000214' 022702  177747          SFTS2:  CMP     #-25.,R2
  857  000220' 003405                          BLE     SFRS2
  858  000222' 016600  000004                  MOV     A1(SP),R0
  859  000226' 016601  000006                  MOV     B1(SP),R1
  860  000232' 000456                          BR      NODS2
  861                                          .IFDF   EAE
  862                                  SFRS2:  MOV     R1,@#MQ
  863                                          MOV     R0,@#AC
  864                                          MOV     R2,@#ASH
  865                                          MOV     @#MQ,R1
  866                                          MOV     @#AC,R0
  867                                          .ENDC
  868                                          .IFDF   MULDIV
  869                                  SFRS2:  .WORD   073002
  870                                          .ENDC
  871                                          .IFNDF  EAE&MULDIV
  872  000234' 022702  177778          SFRS2:  CMP     #-8.,R2
  873  000240' 003431                          BLE     SF0S2
  874  000242' 005004                          CLR     R4
  875  000244' 005700                          TST     R0
  876  000246' 100001                          BPL     NCPS2
  877  000250' 005104                          COM     R4
  878  000252' 022702  177760          NCPS2:  CMP     #-16.,R2
  879  000256' 002405                          BLT     SRLS2
  880  000260' 010001                          MOV     R0,R1
  881  000262' 010400                          MOV     R4,R0
  882  000264' 062702  000020                  ADD     #16.,R2
  883  000270' 001421                          BEQ     SFDS2
  884  000272' 022702  177770          SRLS2:  CMP     #-8.,R2
  885  000276' 003412                          BLE     SF0S2
  886  000300' 062702  000020                  ADD     #16.,R2
  887  000304' 006301          SFLS2:  ASL     R1
  888  000306' 006100                  ROL     R0
  889  000310' 006104                  ROL     R4
```

396


```
  890  000312' 005302                          DEC     R2
  891  000314' 003373                          BGT     SFLS2
  892  000316' 010001                          MOV     R0,R1
  893  000320' 010400                          MOV     R4,R0
  894  000322' 000404                          BR      SFDS2
  895  000324' 006200                  SF0S2:  ASR     R0
  896  000326' 006001                          ROR     R1
  897  000330' 005202                          INC     R2
  898  000332' 002774                          BLT     SF0S2
  899                                          .ENDC
  900  000334' 066600  000004          SFDS2:  ADD     A1(SP),R0
  901  000340' 066601  000006                  ADD     B1(SP),R1
  902  000344' 005500                          ADC     R0
  903  000346' 126616  000001                  CMPB    SIGNS+1(SP),@SP
  904  000352' 001034                          BNE     SUBS2
  905  000354' 030027  001000                  BIT     R0,#1000
  906  000360' 001403                          BEQ     NODS2
  907  000362' 006200                          ASR     R0
  908  000364' 006001                          ROR     R1
  909  000366' 005203                          INC     R3
  910  000370' 000303                  NODS2:  SWAB    R3
  911  000372' 001020                          BNE     OVRS2
  912  000374' 150003                          BISB    R0,R3
  913  000376' 006016                          ROR     @SP
  914  000400' 006003                          ROR     R3
  915  000402' 006001                          ROR     R1
  916  000404' 005501                          ADC     R1
  917  000406' 005503                          ADC     R3
  918  000410' 102411                          BVS     OVRS2
  919  000412' 103410                          BCS     OVRS2
  920  000414' 010366  000010          STRS2:  MOV     R3,A2(SP)
  921  000420' 010166  000012                  MOV     R1,B2(SP)
  922  000424' 005726                  OUTS2:  TST     (SP)+
  923  000426' 012604                          MOV     (SP)+,R4
  924  000430' 022626                          CMP     (SP)+,(SP)+
  925  000432' 000134                          JMP     @(R4)+
  926  000434' 004567  003000          OVRS2:  JSR     R5,$EMR
  927  000440' 000771                          BR      OUTS2
  928  000442'     003                         .BYTE   3
  929  000443'     002                         .BYTE   2
  930  000444' 005700                  SUBS2:  TST     R0
  931  000446' 003005                          BGT     BT9S2
  932  000450' 001413                          BEQ     ETSS2
  933  000452' 005400                          NEG     R0
  934  000454' 005401                          NEG     R1
  935  000456' 005600                          SBC     R0
  936  000460' 000316                          SWAB    @SP
  937                                  BT9S2:
  938                                          .IFDF   EAE
  939                                          BIT     R0,#700
  940                                          BNE     B9AS2
  941                                          MOV     R1,@#MQ
  942                                          MOV     R0,@#AC
  943                                          CLR     @#NOR
  944                                          SUB     @#NOR,R3
```

397

```
                                        MOV     #-6,@#ASH
                                        ADD     #0,R3
                                        BLE     UNF$2
                                        MOV     @#AC,R0
                                        MOV     @#MQ,R1
                                        BR      NOO$2
                                        ;ENDC
000462' 030027  000400          B9A$2:  BIT     R0,#400
000466' 001014                          BNE     UTS$2
000470' 005303                          DEC     R3
000472' 006301                          ASL     R1
000474' 006100                          ROL     R0
000476' 000771                          BR      B9A$2
000500' 005701                  ZTS$2:  TST     R1
                                        ;IFDF   EAE
                                        BNE     BT9$2
                                        BR      ZERS2
                                        ;ENDC
                                        ;IFNDF  EAE
000502' 001415                          BEQ     ZERS2
000504' 000301                          SWAB    R1
000506' 150100                          BISB    R1,R0
000510' 105001                          CLRB    R1
000512' 162703  000010                  SUB     #8.,R3
000516' 000761                          BR      BT9$2
                                        ;ENDC
000520' 005703                  UT$S2:  TST     R3
000522' 003322                          BGT     NOO$2
000524' 004567  003510          UNF$2:  JSR     R5,$ERR
000530' 000401                          BR      UNO$2
000532'     005                         ;BYTE   5
000533'     002                         ;BYTE   2
000534' 005001                  UNO$2:  CLR     R1
000536' 005003                  ZERS2:  CLR     R3
000540' 000725                          BR      STR$2
                                        ;ENDC
                                        ;ENDC
                                        ;EOT
```

398


```
                                )                               BASIC SOURCE FILE #10
                                        ;TITLE  $ALG03
                                        ;IFDF   CND$3
                                        ;GLOBL  $ERR,ALOG
                                        ;IFNDF  $BAS
                                        ;GLOBL  ALOG10
                                        ;ENDC
                                        ;IFNDF  FPU
                                        ;GLOBL  $POLSH,$ADR,$SBR,$MLR,$DVR,$IR
                                        ;ENDC
            000000                      R0=%0
            000001                      R1=%1
            000002                      R2=%2
            000003                      R3=%3
            000004                      R4=%4
            000005                      R5=%5
            000006                      SP=%6
            000007                      PC=%7
            000000                      F0=%0
            000001                      F1=%1
            000002                      F2=%2
            000003                      F3=%3
                                        ;IFNDF  FPU
                                        ;IFNDF  $BAS
                                ALOG10: MOV     @PC,-(SP)
                                        BR      LOG$3
                                        ;ENDC
000542' 005046                  ALOG:   CLR     -(SP)
000544' 016504  000002          LOG$3:  MOV     2(R5),R4
000550' 012746  071030                  MOV     #071030,-(SP)
000554' 012746  137661                  MOV     #137661,-(SP)
000560' 024646                          CMP     -(SP),-(SP)
000562' 016446  000002                  MOV     2(R4),-(SP)
000566' 011446                          MOV     @R4,-(SP)
000570' 003534                          BLE     ERR$3
000572' 006316                          ASL     @SP
000574' 116666  000001  000014          MOVB    1(SP),12.(SP)
000602' 112766  000200  000001          MOVB    #200,1(SP)
000610' 006016                          ROR     @SP
000612' 012746  002363                  MOV     #002363,-(SP)
000616' 012746  040865                  MOV     #040865,-(SP)
000622' 016646  000006                  MOV     6(SP),-(SP)
000626' 016646  000006                  MOV     6(SP),-(SP)
000632' 012746  002303                  MOV     #002303,-(SP)
000636' 012746  040865                  MOV     #040865,-(SP)
000642' 004467  003230                  JSR     R4,$POLSH
000646' 000002' 000760' 000006'         ;WORD   $$BR,UPS3,$ADR,$DVR
        000654' 001234'
                                
000656' 001006' 001006'                 ;WORD   DUP$3,DUP$3
000662' 002322' 000734' 000746'         ;WORD   $MLR,REG$3,$TK$3,$TK$3,$TK$3
        000670' 000746' 000746'
000674' 002322' 000006' 002322'         ;WORD   $MLR,$ADR,$MLR,$ADR,$MLR,$ADR,$MLR,$ADR
```

399

```
        000702' 000006' 002322' 000006'
        000710' 002322' 000006'
 1037
 1038   000714' 000772' 002236' 001828'          .WORD   SCLS3,SIR,PL2S3,SMLR
        000722' 002322'
 1039   000724' 000006' 001832'                  .WORD   SADR,EXIS3
 1040
 1041   000730' 002322' 001832'                  .WORD   SMLR,EXIS3
 1042   000734' 012600                   REGS3:  MOV     (SP)+,R0
 1043   000736' 012601                           MOV     (SP)+,R1
 1044   000740' 012702 001116'                   MOV     #CONS3+4,R2
 1045   000744' 000402                           BR      STCS3
 1046   000746' 010146                   STKS3:  MOV     R1,-(SP)
 1047   000750' 010046                           MOV     R0,-(SP)
 1048   000752' 014246                   STCS3:  MOV     -(R2),-(SP)
 1049   000754' 014246                           MOV     -(R2),-(SP)
 1050   000756' 000134                           JMP     @(R4)+
 1051   000760' 012666 000012            UPS3:   MOV     (SP)+,10.(SP)
 1052   000764' 012666 000012                    MOV     (SP)+,10.(SP)
 1053   000770' 000134                           JMP     @(R4)+
 1054   000772' 005046                   SCLS3:  CLR     -(SP)
 1055   000774' 156616 000006                    BISB    6(SP),@SP
 1056   001000' 162716 000200                    SUB     #200,@SP
 1057   001004' 000134                           JMP     @(R4)+
 1058   001006' 016646 000002            DUPS3:  MOV     2(SP),-(SP)
 1059   001012' 016646 000002                    MOV     2(SP),-(SP)
 1060   001016' 000134                           JMP     @(R4)+
 1061   001020' 012746 071030            PL2S3:  MOV     #071030,-(SP)
 1062   001024' 012746 040061                    MOV     #040061,-(SP)
 1063   001030' 000134                           JMP     @(R4)+
 1064   001032' 105366 000005            EXIS3:  DECB    5(SP)
 1065   001036' 002405                           BLT     LGTS3
 1066   001040' 012746 055731                    MOV     #055731,-(SP)
 1067   001044' 012746 037736                    MOV     #037736,-(SP)
 1068   001050' 000134                           JMP     @(R4)+
 1069   001052' 012600                   LGTS3:  MOV     (SP)+,R0
 1070   001054' 012601                           MOV     (SP)+,R1
 1071   001056' 005726                           TST     (SP)+
 1072   001060' 000205                           RTS     R5
 1073   001062' 062706 000016            ERRS3:  ADD     #14.,SP
 1074   001066' 004567 003144                    JSR     R5,$ERR
 1075   001072' 000205                           RTS     R5
 1076   001074'    004                           .BYTE   4
 1077   001075'    012                           .BYTE   10.
 1078                                            .ENDC
 1079                                            .IFDF   FPU
 1080
 1081                                            .IFNDF  $BAS
 1082                                    ALOG10: MOV     @PC,R4
 1083                                            BR      LOGS3
 1084                                            .ENDC
 1085
 1086                                    ALOG:   CLR     R4
 1087                                    LOGS3:  SETF
 1088                                            SETI
```

400


```
 1089                                            MOV     #FCOS3,R0
 1090                                            LDF     @2(R5),F2
 1091                                            CFCC
 1092                                            BLE     ERRS3
 1093                                            STEXP   F2,R1
 1094                                            LDCIF   R1,F3
 1095                                            MULF    (R0)+,F3
 1096                                            LDEXP   #0,F2
 1097                                            LDF     F2,F1
 1098                                            SUBF    (R0),F2
 1099                                            ADDF    (R0)+,F1
 1100                                            DIVF    F1,F2
 1101                                            LDF     F2,F1
 1102                                            MULF    F1,F1
 1103                                            MOV     #3,R1
 1104                                            LDF     (R0)+,F0
 1105                                    XPOS3:  MULF    F1,F0
 1106                                            DEC     R1
 1107                                            ADDF    (R0)+,F0
 1108                                            BGT     XPOS3
 1109                                            MULF    F2,F0
 1110                                            ADDF    (R0)+,F0
 1111                                            ADDF    F3,F0
 1112                                            TST     R4
 1113                                            BEQ     LGTS3
 1114                                            MULF    (R0)+,F0
 1115                                    LGTS3:  STF     F0,-(SP)
 1116                                            MOV     (SP)+,R0
 1117                                            MOV     (SP)+,R1
 1118                                            RTS     R5
 1119                                    ERRS3:  JSR     R5,$ERR
 1120                                            RTS     R5
 1121                                            .BYTE   4
 1122                                            .BYTE   10.
 1123                                    FCOS3:  .WORD   040061,071030
 1124                                            .WORD   040063,002363
 1125                                            .ENDC
 1126   001076' 037632 014525                    .WORD   037632,014525
 1127   001102' 037714 120036                    .WORD   037714,120036
 1128   001106' 040052 125332                    .WORD   040052,125332
 1129   001112' 040400 000000            CONS3:  .WORD   040400,000000
 1130                                            .IFDF   FPU
 1131                                            .WORD   137661,071030
 1132                                            .WORD   037730,055731
 1133                                            .ENDC
 1134                                            .ENDC
 1135                                            .TITLE  $ANT03
 1136                                            .IFDF   CND$4
 1137                                            .GLOBL  AINT,$INTR
 1138           000000                           R0=%0
 1139           000001                           R1=%1
 1140           000002                           R2=%2
 1141           000003                           R3=%3
 1142           000004                           R4=%4
 1143           000005                           R5=%5
```

401

```
 1144            000006                          SP=%6
 1145            000007                          PC=%7
 1146            177304                          MQ=177304
 1147            177314                          LSH=177314
 1148            000000                          F0=%0
 1149            000001                          F1=%1
 1150                                            .IFDF   FPU
 1151
 1152                                            .IFNDF  $BAS
 1153                                    AINT1   .WORD   170001
 1154                                            .WORD   172475,2
 1155                                            .WORD   171467,24
 1156                                            .WORD   174140
 1157                                            MOV     (SP)+,R0
 1158                                            MOV     (SP)+,R1
 1159                                            RTS     R5
 1160                                            .ENDC
 1161
 1162                                    SINTR1  .WORD   170001
 1163                                            .WORD   172420
 1164                                            .WORD   171467,4
 1165                                            .WORD   174140
 1166                                            JMP     @(R4)+
 1167                                    ONES41  .WORD   000200,0
 1168                                            .ENDC
 1169                                            .IFNDF  FPU
 1170  001116' 016504 000002                    AINT1   MOV     2(R5),R4
 1171  001122' 011400                                    MOV     @R4,R0
 1172  001124' 016401 000002                            MOV     2(R4),R1
 1173  001130' 010702                                    MOV     PC,R2
 1174  001132' 000403                                    BR      AI1S4
 1175  001134' 005002                    SINTR1  CLR     R2
 1176  001136' 012600                                    MOV     (SP)+,R0
 1177  001140' 012601                                    MOV     (SP)+,R1
 1178  001142' 010003                    AI1S41  MOV     R0,R3
 1179  001144' 006103                            ROL     R3
 1180  001146' 105003                            CLRB    R3
 1181  001150' 000303                            SWAB    R3
 1182  001152' 162703 000230                    SUB     #230,R3
 1183  001156' 002020                            BGE     ONES4
 1184  001160' 022703 177750                    CMP     #-30,R3
 1185  001164' 002403                            BLT     SHFS4
 1186  001166' 005000                            CLR     R0
 1187  001170' 005001                            CLR     R1
 1188  001172' 000412                            BR      ONES4
 1189  001174' 010346                    SHFS41  MOV     R3,-(SP)
 1190                                            .IFNDF  EAE&MULDIV
 1191  001176' 006000                    RORS41  ROR     R0
 1192  001200' 006001                            ROR     R1
 1193  001202' 005203                            INC     R3
 1194  001204' 002774                            BLT     RORS4
 1195  001206' 012603                            MOV     (SP)+,R3
 1196  001210' 006301                    ASLS41  ASL     R1
 1197  001212' 006100                            ROL     R0
 1198  001214' 005203                            INC     R3
```

4/02


```
 1199  001216' 002774                            BLT     ASLS4
 1200                                            .ENDC
 1201                                            .IFDF   EAE
 1202                                            MOV     #MQ,R3
 1203                                            MOV     R1,@R3
 1204                                            MOV     R0,-(R3)
 1205                                            MOV     #SP,@#LSH
 1206                                            NEG     @SP
 1207                                            MOV     (SP)+,@#LSH
 1208                                            MOV     (R3)+,R0
 1209                                            MOV     @R3,R1
 1210                                            .ENDC
 1211                                            .IFDF   MULDIV
 1212                                            .WORD   073010
 1213                                            NEG     @SP
 1214                                            .WORD   073020
 1215                                            .ENDC
 1216  001220' 005702                    ONES41  TST     R2
 1217  001222' 001401                            BEQ     ON1S4
 1218  001224' 000205                            RTS     R5
 1219  001226' 010146                    ON1S41  MOV     R1,-(SP)
 1220  001230' 010046                            MOV     R0,-(SP)
 1221  001232' 000134                            JMP     @(R4)+
 1222                                            .ENDC
 1223                                            .ENDC
 1224                                            .TITLE  SCMD02
 1225                                            .IFDF   CNDS5
 1226                                            .GLOBL  SCMD
 1227                                            R0=%0
 1228                                            R1=%1
 1229                                            R2=%2
 1230                                            R4=%4
 1231                                            SP=%6
 1232                                            PC=%7
 1233                                            F0=%0
 1234                                            .IFDF   FPU
 1235                                    SCMD1   .WORD   170011
 1236                                            .WORD   172420
 1237                                            .WORD   173420
 1238                                            .WORD   170000
 1239                                            JMP     @(R4)+
 1240                                            .ENDC
 1241                                            .IFNDF  FPU
 1242                                    SCMD1   MOV     @PC,R0
 1243                                            MOV     6.(SP),R1
 1244                                            BGE     FPSS5
 1245                                            ASL     R0
 1246                                            MOV     (SP)+,R2
 1247                                            BLT     SMES5
 1248                                            BR      NEGS5
 1249                                    FPSS51  MOV     (SP)+,R2
 1250                                            BLT     PLS55
 1251                                    SMES51  CMP     R1,R2
 1252                                            BNE     OUTS5
 1253                                            CMP     6.(SP),@SP
```

4/03

```
1254                                        BNE     OUT$5
1255                                        CMP     10.(SP),2(SP)
1256                                        BNE     OUT$5
1257                                        CMP     12.(SP),4(SP)
1258                                        BNE     OUT$5
1259                                        CLR     R0
1260                                OUT$5:  ROR     R0
1261                                        BCS     PLS$5
1262                                NEG$5:  NEG     R0
1263                                PLS$5:  ADD     #14.,SP
1264                                        TST     R0
1265                                        JMP     @(R4)+
1266                                        .ENDC
1267                                        .ENDC
1268                                        .TITLE  $CMR02
1269                                        .IFDF   CND$6
1270                                        .GLOBL  $CMR
1271                                        R0=%0
1272                                        R1=%1
1273                                        R2=%2
1274                                        R4=%4
1275                                        SP=%6
1276                                        PC=%7
1277                                        F0=%0
1278                                        .IFDF   FPU
1279                                $CMR:   .WORD   170001
1280                                        .WORD   172420
1281                                        .WORD   173420
1282                                        .WORD   170000
1283                                        JMP     @(R4)+
1284                                        .ENDC
1285                                        .IFNDF  FPU
1286                                $CMR:   MOV     @PC,R0
1287                                        MOV     4(SP),R1
1288                                        BGE     FPS$6
1289                                        ASL     R0
1290                                        MOV     (SP)+,R2
1291                                        BLT     SME$6
1292                                        BR      NE0$6
1293                                FPS$6:  MOV     (SP)+,R2
1294                                        BLT     PLS$6
1295                                SME$6:  CMP     R1,R2
1296                                        BNE     OUT$6
1297                                        CMP     4(SP),@SP
1298                                        BNE     OUT$6
1299                                        CLR     R0
1300                                OUT$6:  ROR     R0
1301                                        BCS     PLS$6
1302                                NEG$6:  NEG     R0
1303                                PLS$6:  ADD     #6,SP
1304                                        TST     R0
1305                                        JMP     @(R4)+
1306                                        .ENDC
1307                                        .ENDC
1308                                        .TITLE  $DBL02
```

404


```
1309                                        .IFDF   CND$7
1310
1311                                        .GLOBL  DBLE
1312                                        R0=%0
1313                                        R1=%1
1314                                        R2=%2
1315                                        R3=%3
1316                                        R5=%5
1317                                DBLE:   MOV     2(R5),R2
1318                                        MOV     (R2)+,R0
1319                                        MOV     @R2,R1
1320                                        CLR     R2
1321                                        CLR     R3
1322                                        RTS     R5
1323                                        .ENDC
1324                                        .TITLE  $DCI01
1325                                        .IFDF   CND$8
1326                                        .GLOBL  $DCI,$RCI
1327                                        R0=%0
1328                                        R1=%1
1329                                        R2=%2
1330                                        R3=%3
1331                                        R4=%4
1332                                        R5=%5
1333                                        SP=%6
1334                                        PC=%7
1335                                NUMEND=0
1336                                POINTL=2
1337                                DIGITS=4
1338                                BEXP=6
1339                                ESIGN=8.
1340                                SIGN=10.
1341                                EEXP=12.
1342                                P=30.
1343                                D=32.
1344                                ERP=26.
1345                                LENGTH=34.
1346                                TEMP=LENGTH
1347                                RESULT=P
1348                                START=36.
1349                                END=START
1350                                $RCI:   CLR     -(SP)
1351                                        INC     @SP
1352                                        BR      CNV$8
1353                                $DCI:   CLR     -(SP)
1354                                CNV$8:  MOV     R0,-(SP)
1355                                        MOV     R1,-(SP)
1356                                        MOV     R2,-(SP)
1357                                        MOV     R3,-(SP)
1358                                        MOV     R4,-(SP)
1359                                        MOV     R5,-(SP)
1360                                        CLR     -(SP)
1361                                        CLR     -(SP)
1362                                        CLR     -(SP)
1363                                        MOV     #69.,-(SP)
```

405

```
1364                                    MOV     #10.,-(SP)
1365                                    CLR     -(SP)
1366                                    CLR     -(SP)
1367                                    MOV     START(SP),R5
1368                                    ADD     LENGTH(SP),END(SP)
1369                                    CLR     R0
1370                                    CLR     R1
1371                                    CLR     R2
1372                                    CLR     R3
1373                            SCN$8:  MOVB    (R5)+,R4
1374                                    BIC     #177600,R4
1375                                    CMPB    R4,#'
1376                                    BNE     SG8$8
1377                                    CMP     R5,START(SP)
1378                                    BLT     SCN$8
1379                                    JMP     ZER$8
1380                            SG8$8:  CMPB    R4,#'-
1381                                    BEQ     FLO$8
1382                                    CMPB    R4,#'.
1383                                    BNE     NCK$8
1384                                    INC     SIGN(SP)
1385                                    BR      FLO$8
1386                            NXT$8:  MOVB    (R5)+,R4
1387                                    BIC     #177600,R4
1388                                    CMPB    R4,#'
1389                                    BNE     NCK$8
1390                                    MOV     #'0,R4
1391                            NCK$8:  CMPB    R4,#'0
1392                                    BLT     PCK$8
1393                                    BNE     NNZ$8
1394                                    TST     R0
1395                                    BNE     NNZ$8
1396                                    TST     R1
1397                                    BNE     NNZ$8
1398                                    TST     R2
1399                                    BNE     NNZ$8
1400                                    TST     R3
1401                                    BEQ     FLO$8
1402                            NNZ$8:  CMPB    R4,#'9
1403                                    BGT     EXC$8
1404                                    DEC     DIGITS(SP)
1405                                    BGE     A2I$8
1406                                    INC     EEXP(SP)
1407                                    BR      FLO$8
1408                            A2I$8:  SUB     #60,R4
1409                                    JSR     PC,MLP$8
1410                                    JSR     PC,LFT$8
1411                                    ADD     R4,R3
1412                                    ADC     R2
1413                                    ADC     R1
1414                                    ADC     R0
1415                            FLO$8:  CMP     R5,END(SP)
1416                                    BLT     NXT$8
1417                                    MOV     R5,@SP
1418                            SCL$8:  TST     R0
```

406


```
1419                                    BNE     SC1$8
1420                                    TST     R1
1421                                    BNE     SC1$8
1422                                    TST     R2
1423                                    BNE     SC1$8
1424                                    TST     R3
1425                                    BEQ     ZER$8
1426                            SC1$8:  CMP     @SP,R5
1427                                    BNE     NOP$8
1428                                    SUB     P(SP),EEXP(SP)
1429                            NOP$8:  TST     POINTL(SP)
1430                                    BNE     PNT$8
1431                                    MOV     D(SP),@SP
1432                            PNT$8:  SUB     POINTL(SP),@SP
1433                                    SUB     @SP,EEXP(SP)
1434                                    BGT     MUL$8
1435                                    BLT     DIV$8
1436                                    JMP     FLT$8
1437                            MUL$8:  CMP     R0,#31462
1438                                    BHI     MOV$8
1439                                    JSR     PC,MLP$8
1440                                    INC     BEXP(SP)
1441                            D10$8:  DEC     EEXP(SP)
1442                                    BGT     MUL$8
1443                                    JMP     FLT$8
1444                            MOV$8:  JSR     PC,MOV$8
1445                                    ADD     #3,BEXP(SP)
1446                                    BR      D10$8
1447                            PCK$8:  CMPB    R4,#',
1448                                    BNE     ERR$8
1449                            PTP$8:  TST     POINTL(SP)
1450                                    BNE     ERR$8
1451                                    MOV     R5,POINTL(SP)
1452                                    BR      FLO$8
1453                            ERR$8:  COMB    ERF+1(SP)
1454                            ZER$8:  CLR     R0
1455                                    CLR     R1
1456                                    CLR     R2
1457                                    CLR     R3
1458                                    JMP     STR$8
1459                            EXC$8:  CMPB    R4,#'E
1460                                    BEQ     EXT$8
1461                                    CMPB    R4,#'Q
1462                                    BNE     ERR$8
1463                            EXT$8:  MOV     R5,@SP
1464                                    DEC     @SP
1465                                    MOV     R3,TEMP(SP)
1466                                    CLR     R3
1467                                    CMP     R5,END(SP)
1468                                    BGE     ERR$8
1469                                    MOVB    (R5)+,R4
1470                                    BIC     #177600,R4
1471                                    CMPB    R4,#'.
1472                                    BEQ     EF1$8
1473                                    CMPB    R4,#'.
```

407

```
1474                                    BNE     ENM$8
1475                                    INC     ESIGN(SP)
1476                            EF1$8:  CMP     R5,END(SP)
1477                                    BGE     ERR$8
1478                            EF2$8:  MOVB    (R5)+,R4
1479                                    BIC     #177600,R4
1480                            ENM$8:  CMPB    R4,#'
1481                                    BNE     EN1$8
1482                                    MOV     #'0,R4
1483                            EN1$8:  CMPB    R4,#'0
1484                                    BLT     ERR$8
1485                                    CMPB    R4,#'9
1486                                    BGT     ERR$8
1487                                    SUB     #60,R4
1488                                    ASL     R3
1489                                    ADD     R3,R4
1490                                    ASL     R3
1491                                    ASL     R3
1492                                    ADD     R4,R3
1493                                    CMP     R5,END(SP)
1494                                    BLT     EF2$8
1495                                    TST     ESIGN(SP)
1496                                    BEQ     EN2$8
1497                                    NEG     R3
1498                            EN2$8:  ADD     R3,EEXP(SP)
1499                                    MOV     TEMP(SP),R3
1500                                    JMP     SGL$8
1501                            DIV$8:  TST     R0
1502                                    BLT     OV1$8
1503                            OV2$8:  DEC     BEXP(SP)
1504                                    JSR     PC,LFT$8
1505                                    BPL     OV2$8
1506                            OV1$8:  MOV     #16.,R4
1507                                    JSR     PC,RIT$8
1508                                    MOV     R3,-(SP)
1509                                    MOV     R2,-(SP)
1510                                    MOV     R1,-(SP)
1511                                    MOV     R0,-(SP)
1512                            OV3$8:  JSR     PC,RIT$8
1513                                    CLC
1514                                    JSR     PC,RIT$8
1515                                    MOV     #2,R5
1516                                    CLC
1517                            OV4$8:  JSR     PC,RIT$8
1518                                    ADD     6(SP),R3
1519                                    ADC     R2
1520                                    ADC     R1
1521                                    ADC     R0
1522                                    ADD     4(SP),R2
1523                                    ADC     R1
1524                                    ADC     R0
1525                                    ADD     2(SP),R1
1526                                    ADC     R0
1527                                    ADD     @SP,R0
1528                                    DEC     R5
```

408


```
1529                                    BGT     OV4$8
1530                                    DEC     R4
1531                                    BGT     OV3$8
1532                                    ADD     #8.,SP
1533                                    SUB     #3,BEXP(SP)
1534                                    INC     EEXP(SP)
1535                                    BLT     DIV$8
1536                            FLT$8:  DEC     BEXP(SP)
1537                                    JSR     PC,LFT$8
1538                                    BCC     FLT$8
1539                                    ADD     #200,BEXP(SP)
1540                                    BLE     UND$8
1541                                    CMP     BEXP(SP),#377
1542                                    BGT     OVR$8
1543                                    CLRB    R3
1544                                    BISB    R2,R3
1545                                    SWAB    R3
1546                                    CLRB    R2
1547                                    BISB    R1,R2
1548                                    SWAB    R2
1549                                    CLRB    R1
1550                                    BISB    R0,R1
1551                                    SWAB    R1
1552                                    CLRB    R0
1553                                    BISB    BEXP(SP),R0
1554                                    SWAB    R0
1555                                    ROR     SIGN(SP)
1556                                    JSR     PC,RIT$8
1557                                    ADC     R3
1558                                    ADC     R2
1559                                    ADC     R1
1560                                    ADC     R0
1561                                    BVS     OVR$8
1562                                    BCS     OVR$8
1563                            STR$8:  TSTB    ERF(SP)
1564                                    BEQ     DPR$8
1565                                    ROL     R2
1566                                    ADC     R1
1567                                    ADC     R0
1568                                    BVS     OVR$8
1569                                    BCS     OVR$8
1570                                    MOV     R0,R2
1571                                    MOV     R1,R3
1572                            DPR$8:  MOV     R0,RESULT(SP)
1573                                    MOV     R1,RESULT+2(SP)
1574                                    MOV     R2,RESULT+4(SP)
1575                                    MOV     R3,RESULT+6(SP)
1576                                    ADD     #14.,SP
1577                                    MOV     (SP)+,R5
1578                                    MOV     (SP)+,R4
1579                                    MOV     (SP)+,R3
1580                                    MOV     (SP)+,R2
1581                                    MOV     (SP)+,R1
1582                                    MOV     (SP)+,R0
1583                                    TSTB    @SP
```

409

```
1584                                    BEQ     RRN$8
1585                                    MOV     (SP)+,2(SP)
1586                                    MOV     (SP)+,2(SP)
1587                            RRN$8:  ROL     (SP)+
1588                                    RTS     PC
1589                            OVR$8:
1590                            UND$8:  JMP     ERR$8
1591                            M54$8:  CMP     R0,#144314
1592                                    BLO     M55$8
1593                                    CLC
1594                                    JSR     PC,RII$8
1595                                    INC     BEXP+0+2(SP)
1596                            M55$8:  MOV     R0,-(SP)
1597                                    MOV     R1,-(SP)
1598                                    MOV     R2,-(SP)
1599                                    MOV     R3,-(SP)
1600                                    CLC
1601                                    JSR     PC,RII$8
1602                                    CLC
1603                                    JSR     PC,RII$8
1604                                    BR      M5A$8
1605                            ML9$8:  MOV     R0,-(SP)
1606                                    MOV     R1,-(SP)
1607                                    MOV     R2,-(SP)
1608                                    MOV     R3,-(SP)
1609                                    JSR     PC,LFT$8
1610                                    JSR     PC,LFT$8
1611                            M5A$8:  ADD     (SP)+,R3
1612                                    ADC     R2
1613                                    ADC     R1
1614                                    ADC     R0
1615                                    ADD     (SP)+,R2
1616                                    ADC     R1
1617                                    ADC     R0
1618                                    ADD     (SP)+,R1
1619                                    ADC     R0
1620                                    ADD     (SP)+,R0
1621                                    RTS     PC
1622                            LFT$8:  ASL     R3
1623                                    ROL     R2
1624                                    ROL     R1
1625                                    ROL     R0
1626                                    RTS     PC
1627                            RII$8:  ROR     R0
1628                                    ROR     R1
1629                                    ROR     R2
1630                                    ROR     R3
1631                                    RTS     PC
1632                                    .ENDC
1633                                    .TITLE  SDC004
1634                                    .IFDF   CND$9
1635
1636                                    .GLOBL  $ECO,$FCO,$GCO,$DCO
1637                                    R0=%0
1638                                    R1=%1
```

410


```
1639                                    R2=%2
1640                                    R3=%3
1641                                    R4=%4
1642                                    R5=%5
1643                                    SP=%6
1644                                    PC=%7
1645                            POINT=2
1646                            BEXP=4
1647                            EEXP=6
1648                            TYPE=12.
1649                            P=16.
1650                            D=18.
1651                            L=20.
1652                            S=22.
1653                            $GCO:   MOV     #42403,R0
1654                                    BR      XCO$9
1655                            $FCO:   CLR     R0
1656                                    BR      XCO$9
1657                            $DCO:   MOV     (SP)+,R0
1658                                    MOV     (SP)+,R1
1659                                    MOV     (SP)+,R2
1660                                    MOV     @SP,R3
1661                                    MOV     #42002,@SP
1662                                    MOV     R4,-(SP)
1663                                    MOV     4(SP),R4
1664                                    MOV     R0,4(SP)
1665                                    BR      XC1$9
1666                            $ECO:   MOV     #42402,R0
1667                            XCO$9:  MOV     (SP)+,R3
1668                                    MOV     (SP)+,R1
1669                                    MOV     (SP)+,R2
1670                                    MOV     R3,-(SP)
1671                                    MOV     R0,-(SP)
1672                                    CLR     R3
1673                                    MOV     R4,-(SP)
1674                                    CLR     R4
1675                            XC1$9:  MOV     R5,-(SP)
1676                                    CLR     -(SP)
1677                                    CLR     -(SP)
1678                                    CMP     -(SP),-(SP)
1679                                    ADD     S(SP),L(SP)
1680                                    MOV     S(SP),R0
1681                            CLE$9:  MOVB    #' ,(R4)+
1682                                    CMP     R0,L(SP)
1683                                    BLO     CLE$9
1684                                    ROL     R3
1685                                    ROL     @SP
1686                                    SWAB    R3
1687                                    MOVB    R3,BEXP(SP)
1688                                    BNE     NNZ$9
1689                                    CLR     R0
1690                                    BR      NOD$9
1691                            NNZ$9:  SEC
1692                                    ROR     R3
1693                                    CLRB    R3
```

411

```
1694                                    SWAB    R2
1695                                    BISB    R2,R1
1696                                    CLRB    R2
1697                                    SWAB    R3
1698                                    BISB    R3,R2
1699                                    CLRB    R3
1700                                    SWAB    R4
1701                                    BISB    R4,R3
1702                                    CLRB    R4
1703                                    SUB     #200,BEXP(SP)
1704                                    BLT     DIV$9
1705                                    BEQ     NOM$9
1706                            MUL$9:  TST     R1
1707                                    BLT     ML1$9
1708                                    ASL     R4
1709                                    ROL     R3
1710                                    ROL     R2
1711                                    ROL     R1
1712                                    DEC     BEXP(SP)
1713                                    BGT     MUL$9
1714                                    BR      NOM$9
1715                            ML1$9:  JSR     PC,M49$9
1716                                    INC     EEXP(SP)
1717                                    SUB     #3,BEXP(SP)
1718                                    BGT     MUL$9
1719                                    BEQ     NOM$9
1720                            DIV$9:  CMP     R1,#146314
1721                                    BHIS    DV1$9
1722                                    CMP     BEXP(SP),#3
1723                                    BGT     DV1$9
1724                                    JSR     PC,M54$9
1725                                    DEC     EEXP(SP)
1726                                    ADD     #2,BEXP(SP)
1727                                    BR      DV2$9
1728                            DV1$9:  JSR     PC,RIT$9
1729                            DV2$9:  INC     BEXP(SP)
1730                                    BNE     DIV$9
1731                            NOM$9:  CLR     R0
1732                            NO1$9:  JSR     PC,M54$9
1733                                    JSR     PC,ML0$9
1734                                    TST     R0
1735                                    BNE     NOD$9
1736                                    DEC     EEXP(SP)
1737                                    BR      NO1$9
1738                            NOD$9:  TSTB    TYPE(SP)
1739                                    BEQ     FF7$9
1740                                    RORB    TYPE(SP)
1741                                    BCC     EF7$9
1742                                    TST     EEXP(SP)
1743                                    BLT     EF7$9
1744                                    CMP     EEXP(SP),D(SP)
1745                                    BGT     EF7$9
1746                                    CLRB    TYPE(SP)
1747                                    SUB     #4,L(SP)
1748                                    SUB     EEXP(SP),D(SP)
```

4/2


```
1749                                    CLR     P(SP)
1750                            FF7$9:  MOV     EEXP(SP),R5
1751                            FFE$9:  ADD     D(SP),R5
1752                                    ADD     P(SP),R5
1753                                    JSR     PC,RUQ$9
1754                                    MOV     L(SP),R5
1755                                    SUB     D(SP),R5
1756                                    TSTB    TYPE(SP)
1757                                    BNE     FF5$9
1758                                    ADD     EEXP(SP),P(SP)
1759                                    BLE     FF5$9
1760                                    SUB     P(SP),R5
1761                                    SUB     #2,R5
1762                                    JSR     PC,ISN$9
1763                                    BR      FF3$9
1764                            FF5$9:  SUB     #3,R5
1765                                    JSR     PC,ISN$9
1766                                    MOVB    #'0,(R5)+
1767                                    MOVB    #'.,(R5)+
1768                            FF4$9:  CMP     R5,L(SP)
1769                                    BHIS    FF3$9
1770                                    MOVB    #'0,(R5)+
1771                                    BR      FF4$9
1772                            FF3$9:  MOV     L(SP),R5
1773                                    SUB     D(SP),R5
1774                                    DEC     R5
1775                                    MOV     R5,POINT(SP)
1776                                    TST     P(SP)
1777                                    BGT     FF6$9
1778                                    INC     R5
1779                            FF6$9:  SUB     P(SP),R5
1780                                    JSR     PC,DGB$9
1781                                    TSTB    TYPE(SP)
1782                                    BEQ     ONE$9
1783                                    BR      EFE$9
1784                            ED7$9:  SUB     #4,L(SP)
1785                                    CLR     R5
1786                                    TST     P(SP)
1787                                    BLE     FFE$9
1788                                    MOV     D(SP),R5
1789                                    ADD     P(SP),R5
1790                                    JSR     PC,RUQ$9
1791                                    MOV     L(SP),R5
1792                                    SUB     D(SP),R5
1793                                    DEC     R5
1794                                    MOV     R5,POINT(SP)
1795                                    SUB     P(SP),R5
1796                                    DEC     R5
1797                                    JSR     PC,ISN$9
1798                                    JSR     PC,DGB$9
1799                            EFE$9:  SUB     P(SP),EEXP(SP)
1800                                    MOV     L(SP),R3
1801                                    MOVB    TYPE+1(SP),(R3)+
1802                                    MOV     EEXP(SP),R4
1803                                    BGE     EXP$9
```

4/3

```
1884                                    NEG     R4
1885                                    MOVB    #'0,(R3)+
1886                                    BR      EX1$9
1887                            EXP$9:  MOVB    #' ,(R3)+
1888                            EX1$9:  MOVB    #'0,@R3
1889                            EX3$9:  SUB     #10.,R4
1890                                    BLT     EX2$9
1891                                    INCB    @R3
1892                                    BR      EX3$9
1893                            EX2$9:  ADD     #72,R4
1894                                    MOVB    R4,1(R3)
1895                            ONE$9:  ADD     #8.,SP
1896                                    MOV     (SP)+,R5
1897                                    MOV     (SP)+,R4
1898                                    MOV     (SP)+,6(SP)
1899                                    MOV     (SP)+,6(SP)
1900                                    CMP     (SP)+,(SP)+
1901                                    ROL     (SP)+
1902                                    RTS     PC
1903                            M54$9:  MOV     R1,-(SP)
1904                                    MOV     R2,-(SP)
1905                                    MOV     R3,-(SP)
1906                                    MOV     R4,-(SP)
1907                                    JSR     PC,RIT$9
1908                                    JSR     PC,RIT$9
1909                                    ADC     R4
1910                                    ADC     R3
1911                                    ADC     R2
1912                                    ADC     R1
1913                                    ADD     (SP)+,R4
1914                                    ADC     R3
1915                                    ADC     R2
1916                                    ADC     R1
1917                                    ADC     R0
1918                                    ADD     (SP)+,R3
1919                                    ADC     R2
1920                                    ADC     R1
1921                                    ADC     R0
1922                                    ADD     (SP)+,R2
1923                                    ADC     R1
1924                                    ADC     R0
1925                                    ADD     (SP)+,R1
1926                                    ADC     R0
1927                                    RTS     PC
1928                            M45$9:  MOV     #14.,R5
1929                                    JSR     PC,RIT$9
1930                                    MOV     R4,-(SP)
1931                                    MOV     R3,-(SP)
1932                                    MOV     R2,-(SP)
1933                                    MOV     R1,-(SP)
1934                            M5I$9:  JSR     PC,RIT$9
1935                                    JSR     PC,RIT$9
1936                                    MOV     #2,R0
1937                            M52$9:  JSR     PC,RIT$9
1938                                    ADD     6(SP),R4
```

4/4


```
1939                                    ADC     R3
1940                                    ADC     R2
1941                                    ADC     R1
1942                                    ADD     4(SP),R3
1943                                    ADC     R2
1944                                    ADC     R1
1945                                    ADD     2(SP),R2
1946                                    ADC     R1
1947                                    ADD     @SP,R1
1948                                    DEC     R0
1949                                    BGT     M52$9
1950                                    DEC     R5
1951                                    BGT     M5I$9
1952                                    ADD     #8.,SP
1953                                    RTS     PC
1954                            ML8$9:  MOV     R5,-(SP)
1955                                    MOV     #3,R5
1956                            M8I$9:  ASL     R4
1957                                    ROL     R3
1958                                    ROL     R2
1959                                    ROL     R1
1960                                    ROL     R0
1961                                    DEC     R5
1962                                    BGT     M8I$9
1963                                    MOV     (SP)+,R5
1964                                    RTS     PC
1965                            ERR$9:  TST     (SP)+
1966                                    MOV     8(SP),R3
1967                                    MOV     6(SP),R4
1968                                    TSTB    TYPE(SP)
1969                                    BEQ     ST$$9
1970                                    ADD     #4,R4
1971                            ST$$9:  MOVB    #'0,(R3)+
1972                                    CMP     R3,R4
1973                                    BLO     ST$$9
1974                                    COM     TYPE(SP)
1975                                    BR      ONE$9
1976                            RU0$9:  CMP     R5,#20.
1977                                    BGT     RU1$9
1978                                    MOV     R5,@EXP+6+2(SP)
1979                                    BEQ     RU3$9
1980                                    BLT     RU1$9
1981                                    MOV     R0,-(SP)
1982                                    MOV     R1,-(SP)
1983                                    MOV     R2,-(SP)
1984                                    MOV     R3,-(SP)
1985                                    MOV     R4,-(SP)
1986                                    MOV     #100000,R1
1987                                    CLR     R2
1988                                    CLR     R3
1989                                    CLR     R4
1990                            ROP$9:  DEC     @EXP+6+2+10.(SP)
1991                                    BEQ     RU0$9
1992                                    JSR     PC,M4$$9
1993                                    JSR     PC,RIT$9
```

4/5

```
1914                                    JSR     PC,RITS9
1915                                    JSR     PC,RITS9
1916                                    BR      RUPS9
1917                            RDDS9:  CLR     R0
1918                                    ADD     (SP)+,R4
1919                                    ADC     R3
1920                                    ADC     R2
1921                                    ADC     R1
1922                                    ADD     (SP)+,R3
1923                                    ADC     R2
1924                                    ADC     R1
1925                                    ADD     (SP)+,R2
1926                                    ADC     R1
1927                                    ADD     (SP)+,R1
1928                                    ADC     R0
1929                                    ADD     (SP)+,R0
1930                            RU2S9:  CMP     #10.,R0
1931                                    BGT     RU1S9
1932                                    INC     EEXP+2(SP)
1933                            RU1S9:  RTS     PC
1934                            RU3S9:  ADD     #5,R0
1935                                    BR      RU2S9
1936                            ISNS9:  CMP     R5,S-0+2(SP)
1937                                    BLO     SPCS9
1938                                    ROR     0+2(SP)
1939                                    BCC     ISRS9
1940                                    MOVB    #'-,@R5
1941                            ISRS9:  INC     R5
1942                                    RTS     PC
1943                            SPCS9:  ROR     0+2(SP)
1944                                    BCS     ERRS9
1945                                    INC     R5
1946                                    CMP     R5,S+2(SP)
1947                                    BLO     ERRS9
1948                                    RTS     PC
1949                            DGSS9:  CMP     #10.,R0
1950                                    BGT     DG1S9
1951                                    MOVB    #'1,(R5)+
1952                                    SUB     #10.,R0
1953                            DG1S9:  CMP     POINT+2(SP),R5
1954                                    BNE     DG2S9
1955                                    MOVB    #'.,(R5)+
1956                            DG2S9:  CMP     L+2(SP),R5
1957                                    BLOS    DIGS9
1958                            DG3S9:  ADD     #60,R0
1959                                    MOVB    R0,(R5)+
1960                                    CLR     R0
1961                                    JSR     PC,M54S9
1962                                    JSR     PC,ML8S9
1963                                    BR      DG1S9
1964                            DIGS9:  RTS     PC
1965                            RITS9:  CLC
1966                                    ROR     R1
1967                                    ROR     R2
1968                                    ROR     R3
```

416


```
1969                                    ROR     R4
1970                                    RTS     PC
1971                                    .ENDC
1972                                    .EOT
1973
```

417

```
                                                                BASIC SOURCE FILE #11
                        .TITLE  $QLG03
                        .IFDF   CNO$1@
                        .GLOBL  DLOG,DLOG1@,$ERR
                        .IFNDF  FPU
                        .GLOBL  $POLSH,$ADD,$SBD,$MLD,$DVD,$ID,$POPR4
                        .ENDC
                        R0=%0
                        R1=%1
                        R2=%2
                        R3=%3
                        R4=%4
                        R5=%5
                        SP=%6
                        PC=%7
                        F0=%0
                        F1=%1
                        F2=%2
                        F3=%3
                        .IFNDF  FPU
DLOG1@: MOV     @PC,-(SP)
                BR      LOG$1@
DLOG:   CLR     -(SP)
LOG$1@: MOV     R5,-(SP)
                MOV     2(R5),R4
                ADD     #8.,R4
                MOV     #147572,-(SP)
                MOV     #173721,-(SP)
                MOV     #071027,-(SP)
                MOV     #137601,-(SP)
                SUB     #8.,SP
                MOV     -(R4),-(SP)
                MOV     -(R4),-(SP)
                MOV     -(R4),-(SP)
                MOV     -(R4),-(SP)
                BLE     ERR$1@
                ASL     @SP
                MOVB    1(SP),26.(SP)
                MOVB    #200,1(SP)
                ROR     @SP
                MOV     #157145,-(SP)
                MOV     #031771,-(SP)
                MOV     #002303,-(SP)
                MOV     #040005,-(SP)
                MOV     14.(SP),-(SP)
                MOV     14.(SP),-(SP)
                MOV     14.(SP),-(SP)
                MOV     14.(SP),-(SP)
                MOV     #157145,-(SP)
                MOV     #031771,-(SP)
                MOV     #002303,-(SP)
                MOV     #040005,-(SP)
                JSR     R4,$POLSH
                .WORD   $SBD,UPS1@,$ADD,$DVD
```

418


```
                .WORD   DUPS1@,DUPS1@
                .WORD   $MLD
                .WORD   $POPR4
                .WORD   REG$1@
XPDS1@: .WORD   $MLD,$ADD,$MLD,$ADD,$MLD,$ADD,$MLD,$ADD
                .WORD   $MLD,$ADD,$MLD,$ADD,$MLD,$ADD

                .WORD   SCL$1@,$ID,PL2S1@,$MLD
                .WORD   $ADD,EXI$1@

                .WORD   $MLD,EXI$1@
ERR$1@: ADD     #24.,SP
                JSR     R5,$ERR
                BR      ERO$1@
                .BYTE   4
                .BYTE   3
REG$1@: MOV     #CON$1@+8.,R4
                MOV     #7,R5
                BR      STC$1@
STK$1@: MOV     R3,-(SP)
                MOV     R2,-(SP)
                MOV     R1,-(SP)
                MOV     R0,-(SP)
STC$1@: MOV     -(R4),-(SP)
                MOV     -(R4),-(SP)
                MOV     -(R4),-(SP)
                MOV     -(R4),-(SP)
                DEC     R5
                BGT     STK$1@
                MOV     #XPD$1@,R4
                JMP     @(R4)+
UPS1@:  MOV     (SP)+,22.(SP)
                MOV     (SP)+,22.(SP)
                MOV     (SP)+,22.(SP)
                MOV     (SP)+,22.(SP)
                JMP     @(R4)+
SCL$1@: CLR     -(SP)
                BISB    12.(SP),@SP
                SUB     #200,@SP
                JMP     @(R4)+
DUPS1@: MOV     6(SP),-(SP)
                MOV     6(SP),-(SP)
                MOV     6(SP),-(SP)
                MOV     6(SP),-(SP)
                JMP     @(R4)+
PL2S1@: MOV     #147572,-(SP)
                MOV     #173721,-(SP)
                MOV     #071027,-(SP)
                MOV     #040001,-(SP)
                JMP     @(R4)+
EXI$1@: DECB    11.(SP)
                BLT     LO$1@
                MOV     #024102,-(SP)
                MOV     #124007,-(SP)
                MOV     #055730,-(SP)
```

419

```
2084                                    MOV     #037736,-(SP)
2085                                    JMP     @(R4)+
2086                            LGTS10: MOV     (SP)+,R0
2087                                    MOV     (SP)+,R1
2088                                    MOV     (SP)+,R2
2089                                    MOV     (SP)+,R3
2090                            ERQS10: MOV     (SP)+,R5
2091                                    TST     (SP)+
2092                                    RTS     R5
2093                                    .ENDC
2094                                    .IFDF   FPU
2095                            DLOG10: MOV     #PC,R4
2096                                    BR      LOGS10
2097                            DLOG:   CLR     R4
2098                            LOGS10: SETD
2099                                    SETI
2100                                    MOV     #FCQS10,R0
2101                                    LDD     @2(R5)!,F2
2102                                    CFCC
2103                                    BLE     ERRS10
2104                                    STEXP   F2,R1
2105                                    LDCID   R1,F3
2106                                    MULD    (R0)+,F3
2107                                    LDEXP   #0,F2
2108                                    LDD     F2,F1
2109                                    SUBD    (R0),F2
2110                                    ADDD    (R0)+,F1
2111                                    DIVD    F1,F2
2112                                    LDD     F2,F1
2113                                    MULD    F1,F1
2114                                    MOV     #6,R1
2115                                    LDD     (R0)+,F0
2116                            XPDS10: MULD    F1,F0
2117                                    DEC     R1
2118                                    ADDD    (R0)+,F0
2119                                    BGT     XPDS10
2120                                    MULD    F2,F0
2121                                    ADDD    (R0)+,F0
2122                                    ADDD    F3,F0
2123                                    TST     R4
2124                                    BEQ     LGTS10
2125                                    MULD    (R0),F0
2126                            LGTS10: STD     F0,-(SP)
2127                                    MOV     (SP)+,R0
2128                                    MOV     (SP)+,R1
2129                                    MOV     (SP)+,R2
2130                                    MOV     (SP)+,R3
2131                                    RTS     R5
2132                            ERRS10: JSR     R5,$ERR
2133                                    RTS     R5
2134                                    .BYTE   4
2135                                    .BYTE   3
2136                            FCQS10: .WORD   040861,071027
2137                                    .WORD   173721,147572
2138                                    .WORD   040069,002363
```

420


```
2139                                    .WORD   031771,157145
2140                                    .ENDC
2141                                    .WORD   037459,100270
2142                                    .WORD   157100,174770
2143                                    .WORD   037471,072731
2144                                    .WORD   137710,117115
2145                                    .WORD   037543,111153
2146                                    .WORD   060101,135465
2147                                    .WORD   037622,044436
2148                                    .WORD   007300,063062
2149                                    .WORD   037714,140314
2150                                    .WORD   153490,165773
2151                                    .WORD   040052,125252
2152                                    .WORD   125247,004643
2153                            CONS10: .WORD   040400,000000
2154                                    .WORD   000000,000057
2155                                    .IFDF   FPU
2156                                    .WORD   137661,071027
2157                                    .WORD   173721,147572
2158                                    .WORD   037730,055730
2159                                    .WORD   124467,024162
2160                                    .ENDC
2161                                    .ENDC
2162                                    .TITLE  SQNT03
2163                                    .IFDF   CND$11
2164                                    .GLOBL  $DINT
2165                                    R0=%0
2166                                    R1=%1
2167                                    R2=%2
2168                                    R3=%3
2169                                    R4=%4
2170                                    R5=%5
2171                                    SP=%6
2172                                    MQ=177304
2173                                    ASH=177312
2174                                    F0=%0
2175                                    F1=%1
2176                                    .IFDF   FPU
2177                            $DINT:  .WORD   170011
2178                                    .WORD   172420
2179                                    .WORD   171467,4
2180                                    .WORD   174140
2181                                    JMP     @(R4)+
2182                                    .WORD   040200,0,0,0
2183                                    .ENDC
2184                                    .IFNDF  FPU
2185                            $DINT:  MOV     (SP)+,R0
2186                                    MOV     (SP)+,R1
2187                                    MOV     (SP)+,R2
2188                                    MOV     (SP)+,R3
2189                                    MOV     R4,-(SP)
2190                                    MOV     R5,-(SP)
2191                                    MOV     R0,R4
2192                                    ROL     R4
2193                                    CLRB    R4
```

421

```
2194                                    SWAB    R4
2195                                    SUB     #270,R4
2196                                    BGE     ONES11
2197                                    CMP     #-70,R4
2198                                    BLT     SHFS11
2199                                    CLR     R0
2200                                    CLR     R1
2201                            C23S11: CLR     R2
2202                                    CLR     R3
2203                                    BR      ONES11
2204                            SHFS11:
2205                                    .IFNDF  EAE&MULDIV
2206                                    MOV     R4,R5
2207                                    CMP     #-32.,R4
2208                                    BLT     RORS11
2209                                    BEQ     C23S11
2210                                    ADD     #32.,R4
2211                                    MOV     R4,R5
2212                            RR1S11: ROR     R0
2213                                    ROR     R1
2214                                    INC     R4
2215                                    BLT     RR1S11
2216                            ASLS11: ASL     R1
2217                                    ROL     R0
2218                                    INC     R5
2219                                    BLT     ASLS11
2220                                    BR      C23S11
2221                            RORS11: ROR     R2
2222                                    ROR     R3
2223                                    INC     R4
2224                                    BLT     RORS11
2225                            ASLS11: ASL     R3
2226                                    ROL     R2
2227                                    INC     R5
2228                                    BLT     ASLS11
2229                                    .ENDC
2230                                    .IFDF   EAE
2231                                    MOV     @#0,R0
2232                                    .ENDC
2233                                    .IFDF   MULDIV!EAE
2234                                    CMP     #-32.,R4
2235                                    BLT     R23S11
2236                                    BEQ     C23S11
2237                            R0S11:  ADD     #32.,R4
2238                                    .IFDF   MULDIV
2239                                    .WORD   073004
2240                                    NEG     R4
2241                                    .WORD   073004
2242                                    .ENDC
2243                                    .IFDF   EAE
2244                                    MOV     R1,@R5
2245                                    MOV     R0,-(R5)
2246                                    MOV     R4,@#ASH
2247                                    NEG     R4
2248                                    MOV     R4,@#ASH
```

422


```
2249                                    MOV     (R5)+,R0
2250                                    MOV     @R5,R1
2251                                    .ENDC
2252                                    BR      C23S11
2253                                    .IFDF   MULDIV
2254                            R23S11: .WORD   073204
2255                                    NEG     R4
2256                                    .WORD   073204
2257                                    .ENDC
2258                                    .IFDF   EAE
2259                            R23S11: MOV     R3,@R5
2260                                    MOV     R2,-(R5)
2261                                    MOV     R4,@#ASH
2262                                    NEG     R4
2263                                    MOV     R4,@#ASH
2264                                    MOV     (R5)+,R2
2265                                    MOV     @R5,R3
2266                                    .ENDC
2267                                    .ENDC
2268                            ONES11: MOV     (SP)+,R5
2269                                    MOV     (SP)+,R4
2270                                    MOV     R3,-(SP)
2271                                    MOV     R2,-(SP)
2272                                    MOV     R1,-(SP)
2273                                    MOV     R0,-(SP)
2274                                    JMP     @(R4)+
2275                                    .ENDC
2276                                    .ENDC
2277                                    .TITLE  $DR02
2278                                    .IFDF   CND$12
2279                                    .GLOBL  $DR,$ERR
2280                                    R4=%4
2281                                    R5=%5
2282                                    SP=%6
2283                                    PC=%7
2284                                    .IFDF   FPU
2285                            $DR:    .WORD   170001
2286                                    .WORD   177420
2287                                    .WORD   170000
2288                                    BVS     OV1$12
2289                                    .WORD   174400
2290                                    JMP     @(R4)+
2291                                    .ENDC
2292                                    .IFNDF  FPU
2293                            $DR:    ROL     4(SP)
2294                                    ADC     2(SP)
2295                                    ADC     @SP
2296                                    BCS     OVR$12
2297                                    BVS     OVR$12
2298                            DR1$12: MOV     (SP)+,2(SP)
2299                                    MOV     (SP)+,2(SP)
2300                                    JMP     @(R4)+
2301                            OVR$12: CMP     (SP)+,(SP)+
2302                                    CMP     (SP)+,(SP)+
2303                                    .ENDC
```

423

```
2304                                    OV[$12:  JSR     R5,$EMR
2305                                             BR      DR2$12
2306                                             .BYTE   3
2307                                             .BYTE   23.
2308                                    DR2$12:  CLR     -(SP)
2309                                             CLR     -(SP)
2310                                             JMP     @(R4)+
2311                                             .ENDC
2312                                             .TITLE  $QSN04
2313                                             .IFDF   CNO$13
2314
2315                                             .GLOBL  DSIN,QCOS
2316                                             .IFNDF  FPU
2317                                             .GLOBL  $ADD,$SBD,$MLD,$DVD,$DINT,$POLSH,$POPR4
2318                                             .ENDC
2319                                             R0=%0
2320                                             R1=%1
2321                                             R2=%2
2322                                             R3=%3
2323                                             R4=%4
2324                                             R5=%5
2325                                             SP=%6
2326                                             PC=%7
2327                                             F0=%0
2328                                             F1=%1
2329                                             F2=%2
2330                                             F3=%3
2331                                             .IFNDF  FPU
2332                                    QCOS:    MOV     R5,-(SP)
2333                                             MOV     2(R5),R4
2334                                             CLR     -(SP)
2335                                             MOV     6(R4),-(SP)
2336                                             MOV     4(R4),-(SP)
2337                                             MOV     2(R4),-(SP)
2338                                             MOV     @R4,-(SP)
2339                                             MOV     #064302,-(SP)
2340                                             MOV     #121041,-(SP)
2341                                             MOV     #007732,-(SP)
2342                                             MOV     #040311,-(SP)
2343                                             JSR     R4,$POLSH
2344                                             .WORD   $ADD,SNC$13
2345                                    DS[N:    MOV     R5,-(SP)
2346                                             MOV     2(R5),R4
2347                                             CLR     -(SP)
2348                                             MOV     6(R4),-(SP)
2349                                             MOV     4(R4),-(SP)
2350                                             MOV     2(R4),-(SP)
2351                                             MOV     @R4,-(SP)
2352                                    SNC$13:  ASL     @SP
2353                                             ROR     0,(SP)
2354                                             ROR     @SP
2355                                             MOV     #064302,-(SP)
2356                                             MOV     #121041,-(SP)
2357                                             MOV     #007732,-(SP)
2358                                             MOV     #040711,-(SP)
```

424


```
2359                                             JSR     R4,$POLSH
2360                                             .WORD   $DVD
2361                                             .WORD   DUP$13
2362                                             .WORD   $DINT
2363                                             .WORD   $SBD
2364                                             .WORD   X4$13
2365                                             .WORD   DUP$13
2366                                             .WORD   $DINT
2367                                             .WORD   QUD$13
2368                                             .WORD   $SBD
2369                                             .WORD   QST$13
2370                                    QSE$13:  .WORD   DUP$13
2371                                             .WORD   DUP$13
2372                                             .WORD   $MLD
2373                                             .WORD   $POPR4
2374                                             .WORD   PLY$13
2375                                    XPD$13:  .WORD   $MLD,$ADD,$MLD,$ADD,$MLD,$ADD,$MLD,$ADD
2376                                             .WORD   $MLD,$ADD,$MLD,$ADD,$MLD,$ADD,$MLD,$ADD
2377                                             .WORD   $MLD
2378                                    PR4$13:  .WORD   $POPR4
2379                                             .WORD   RTN$13
2380                                    RTN$13:  TST     (SP)+
2381                                             BGE     RT1$13
2382                                             ADD     #100000,R0
2383                                    RT[$13:  MOV     (SP)+,R5
2384                                             RTS     R5
2385                                    DUP$13:  MOV     6(SP),-(SP)
2386                                             MOV     6(SP),-(SP)
2387                                             MOV     6(SP),-(SP)
2388                                             MOV     6(SP),-(SP)
2389                                             JMP     @(R4)+
2390                                    X4$13:   TST     @SP
2391                                             BEQ     ZER$13
2392                                             INCB    1(SP)
2393                                             JMP     @(R4)+
2394                                    ZER$13:  MOV     #PR4$13,R4
2395                                             JMP     @(R4)+
2396                                    QUD$13:  BIS     @SP,10,(SP)
2397                                             JMP     @(R4)+
2398                                    QST$13:  TSTB    8,(SP)
2399                                             BEQ     Q13$13
2400                                             ADD     #100000,@SP
2401                                             CLR     -(SP)
2402                                             CLR     -(SP)
2403                                             CLR     -(SP)
2404                                             MOV     #40200,-(SP)
2405                                             JSR     R4,$POLSH
2406                                             .WORD   $ADD,QSR$13
2407                                    QSR$13:  MOV     #QSE$13,R4
2408                                    Q13$13:  ASRB    9,(SP)
2409                                             BCC     QUT$13
2410                                             ADD     #100000,@SP
2411                                    QUT$13:  JMP     @(R4)+
2412                                    PLY$13:  MOV     #CON$13+6,,R4
2413                                             MOV     #7,,R5
```

425

```
                                        BR      PY1$13
                                PY2$13: MOV     R3,-(SP)
                                        MOV     R2,-(SP)
                                        MOV     R1,-(SP)
                                        MOV     R0,-(SP)
                                PY1$13: MOV     -(R4),-(SP)
                                        MOV     -(R4),-(SP)
                                        MOV     -(R4),-(SP)
                                        MOV     -(R4),-(SP)
                                        DEC     R5
                                        BGT     PY2$13
                                        MOV     #XPO$13,R4
                                        JMP     @(R4)+
                                        .ENDC
                                        .IFDF   FPU
                                DCOS:   SETD
                                        LDD     @2(R5),F0
                                        ADDD    PI2$13,F0
                                        BR      SNC$13
                                DSIN:   SETD
                                        LDD     @2(R5),F0
                                SNC$13: SETI
                                        MOV     #FCOS$13,R0
                                        CLR     R4
                                        CFCC
                                        BGE     POS$13
                                        INC     R4
                                        ABSD    F0
                                POS$13: DIVD    (R0)+,F0
                                        MODD    #1.0,F0
                                        CFCC
                                        BEQ     RTN$13
                                        MODD    #4.0,F0
                                        STCDI   F1,R1
                                        ROR     R1
                                        BCC     Q13$13
                                        NEGD    F0
                                        ADDD    #1.0,F0
                                Q13$13: ROR     R1
                                        BCC     Q12$13
                                        NEGD    F0
                                Q12$13: LDD     F0,F2
                                        MULD    F2,F2
                                        MOV     #6.,R1
                                        LDD     (R0)+,F1
                                XPO$13: MULD    F2,F1
                                        DEC     R1
                                        ADDD    (R0)+,F1
                                        BGT     XPO$13
                                        MULD    F1,F0
                                        TST     R4
                                        BEQ     RTN$13
                                        NEGD    F0
                                RTN$13: STD     F0,-(SP)
                                        MOV     (SP)+,R0
```

426


```
                                        MOV     (SP)+,R1
                                        MOV     (SP)+,R2
                                        MOV     (SP)+,R3
                                        RTS     R5
                                PI2$13: .WORD   040311,007732
                                        .WORD   121041,064302
                                FCOS13: .WORD   040711,007732
                                        .WORD   121041,064302
                                        .ENDC
                                        .WORD   026710,100703
                                        .WORD   045277,140362
                                        .WORD   130467,130273
                                        .WORD   103896,123153
                                        .WORD   032166,074657
                                        .WORD   047256,154742
                                        .WORD   133561,101646
                                        .WORD   107210,134016
                                        .WORD   035056,036032
                                        .WORD   041214,103131
                                        .WORD   136231,064546
                                        .WORD   071423,125024
                                        .WORD   037243,032743
                                        .WORD   035655,051557
                                        .WORD   140045,056747
                                        .WORD   036455,171222
                                CONS13: .WORD   040311,007732
                                        .WORD   121041,064302
                                        .ENDC
                                        .TITLE  SQSQ04
                                        .IFDF   CND$14

                                        .GLOBL  DSQRT,$ERR
                                        .IFNDF  FPU
                                        .GLOBL  $ADD,$DVD,$POLSH
                                        .ENDC
                                        R0=%0
                                        R1=%1
                                        R2=%2
                                        R3=%3
                                        R4=%4
                                        R5=%5
                                        F0=%0
                                        F1=%1
                                        F2=%2
                                        SP=%6
                                        .IFNDF  FPU
                                DSQRT:  MOV     R5,-(SP)
                                        MOV     2(R5),R5
                                        MOV     @R5,R1
                                        BMI     ERR$14
                                        BEQ     ZER$14
                                        MOV     2(R5),R2
                                        MOV     #4,-(SP)
                                        ASR     R1
                                        ROR     R2
```

427

```
2524                                    ADD     #20100,R1
2525                                    CLR     -(SP)
2526                                    CLR     -(SP)
2527                                    MOV     R2,-(SP)
2528                                    MOV     R1,-(SP)
2529                                    CLR     -(SP)
2530                                    CLR     -(SP)
2531                                    MOV     2(R5),-(SP)
2532                                    MOV     @R5,-(SP)
2533                                    CLR     -(SP)
2534                                    CLR     -(SP)
2535                                    MOV     R2,-(SP)
2536                                    MOV     R1,-(SP)
2537                            LUPS14: JSR     R4,$POLSH
2538                                    .WORD   $DVD,$ADD,UPLS14
2539                            UPLS14: SUB     #200,@SP
2540                                    DEC     8.(SP)
2541                                    BEQ     OUTS14
2542                                    MOV     6(R5),-(SP)
2543                                    MOV     4(R5),-(SP)
2544                                    MOV     2(R5),-(SP)
2545                                    MOV     @R5,-(SP)
2546                                    MOV     14.(SP),-(SP)
2547                                    MOV     14.(SP),-(SP)
2548                                    MOV     14.(SP),-(SP)
2549                                    MOV     14.(SP),-(SP)
2550                                    BR      LUPS14
2551                            OUTS14: MOV     (SP)+,R0
2552                                    MOV     (SP)+,R1
2553                                    MOV     (SP)+,R2
2554                                    MOV     (SP)+,R3
2555                                    TST     (SP)+
2556                            RTNS14: MOV     (SP)+,R5
2557                                    RTS     R5
2558                            ERRS14: JSR     R5,$ERR
2559                                    BR      RTNS14
2560                                    .BYTE   4
2561                                    .BYTE   4
2562                            ZERS14: CLR     R0
2563                                    CLR     R1
2564                                    CLR     R2
2565                                    CLR     R3
2566                                    BR      RTNS14
2567                                    .ENDC
2568                                    .IFDF   FPU
2569                            DSQRT:  MOV     2(R5),R4
2570                                    MOV     @R4,R1
2571                                    BMI     ERRS14
2572                                    BEQ     ZERS14
2573                                    MOV     2(R4),R2
2574                                    ASR     R1
2575                                    ROR     R2
2576                                    ADD     #20100,R1
2577                                    CLR     -(SP)
2578                                    CLR     -(SP)
```

428


```
2579                                    MOV     R2,-(SP)
2580                                    MOV     R1,-(SP)
2581                                    MOV     #4,R0
2582                                    SETD
2583                                    LDD     (SP)+,F0
2584                                    LDD     @R4,F2
2585                            LUPS14: LDD     F0,F1
2586                                    LDD     F2,F0
2587                                    DIVD    F1,F0
2588                                    ADDD    F1,F0
2589                                    DEC     R0
2590                                    DIVD    #2.0,F0
2591                                    BGT     LUPS14
2592                                    STD     F0,-(SP)
2593                                    MOV     (SP)+,R0
2594                                    MOV     (SP)+,R1
2595                                    MOV     (SP)+,R2
2596                                    MOV     (SP)+,R3
2597                                    RTS     R5
2598                            ERRS14: JSR     R5,$ERR
2599                                    RTS     R5
2600                                    .BYTE   4
2601                                    .BYTE   4
2602                            ZERS14: CLR     R0
2603                                    CLR     R1
2604                                    CLR     R2
2605                                    CLR     R3
2606                                    RTS     R5
2607                                    .ENDC
2608                                    .ENDC
2609                                    .TITLE  SQTN03
2610                                    .IFDF   CNOS15
2611
2612                                    .GLOBL  DATAN,DATAN2
2613                                    .IFNDF  FPU
2614                                    .GLOBL  $ADD,$SBD,$MLD,$DVD,$POLSH,$POPR4
2615                                    .ENDC
2616                                    R0=%0
2617                                    R1=%1
2618                                    R2=%2
2619                                    R3=%3
2620                                    R4=%4
2621                                    R5=%5
2622                                    SP=%6
2623                                    F0=%0
2624                                    F1=%1
2625                                    F2=%2
2626                                    F3=%3
2627                                    F4=%4
2628                                    F5=%5
2629                                    .IFNDF  FPU
2630                            DATAN2: MOV     R5,-(SP)
2631                                    CLR     -(SP)
2632                                    CLR     -(SP)
2633                                    CLR     -(SP)
```

429

```
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        MOV     2(R5),R4
                                        MOV     6(R4),-(SP)
                                        MOV     4(R4),-(SP)
                                        MOV     2(R4),-(SP)
                                        MOV     @R4,-(SP)
                                        MOV     @SP,R0
                                        MOV     4(R5),R4
                                        MOV     6(R4),-(SP)
                                        MOV     4(R4),-(SP)
                                        MOV     2(R4),-(SP)
                                        MOV     @R4,-(SP)
                                        MOV     @SP,R1
                                        BEQ     INF$15
                                        ASL     R0
                                        CLRB    R0
                                        SWAB    R0
                                        ASL     R1
                                        CLRB    R1
                                        SWAB    R1
                                        SUB     R1,R0
                                        CMP     #56.,R0
                                        BLT     INF$15
                                DIV$15: JSR     R4,$POLSH
                                        .WORD   $DVD,UPL$15
                                UPL$15: TST     @4(R5)
                                        BGE     ATE$15
                                        MOV     #040311,16.(SP)
                                        MOV     #007732,18.(SP)
                                        MOV     #121041,20.(SP)
                                        MOV     #064301,22.(SP)
                                        TST     @2(R5)
                                        BGE     ATE$15
                                        ADD     #100000,16.(SP)
                                ATE$15: TST     @SP
                                        BR      AT1$15
                                INF$15: ADD     #56.,SP
                                        MOV     #040311,R0
                                        MOV     #007732,R1
                                        MOV     #121041,R2
                                        MOV     #064301,R3
                                        TST     @2(R5)
                                        BGE     INR$15
                                        ADD     #100000,R0
                                INR$15: RTS     R4
                                DATAN:  MOV     R5,-(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
```

430


```
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        MOV     2(R5),R4
                                        MOV     6(R4),-(SP)
                                        MOV     4(R4),-(SP)
                                        MOV     2(R4),-(SP)
                                        MOV     @R4,-(SP)
                                AT1$15: BGE     PLU$15
                                        ADD     #100000,@SP
                                        INC     2.(SP)
                                PLU$15: CMP     @SP,#40200
                                        BLO     LE1$15
                                        BGT     GT1$15
                                        TST     2(SP)
                                        BNE     GT1$15
                                        TST     4(SP)
                                        BNE     GT1$15
                                        TST     6(SP)
                                        BEQ     LE1$15
                                GT1$15: MOV     #140311,8.(SP)
                                        MOV     #007732,10.(SP)
                                        MOV     #121041,12.(SP)
                                        MOV     #064301,14.(SP)
                                        DEC     2.(SP)
                                        MOV     6(SP),-(SP)
                                        MOV     6(SP),-(SP)
                                        MOV     6(SP),-(SP)
                                        MOV     6(SP),-(SP)
                                        MOV     #40200,8.(SP)
                                        CLR     10.(SP)
                                        CLR     12.(SP)
                                        CLR     14.(SP)
                                        JSR     R4,$POLSH
                                        .WORD   $DVD,LE1$15
                                LE1$15: MOV     6(SP),-(SP)
                                        MOV     6(SP),-(SP)
                                        MOV     6(SP),-(SP)
                                        MOV     6(SP),-(SP)
                                        CLR     8.(SP)
                                        CLR     10.(SP)
                                        CLR     12.(SP)
                                        CLR     14.(SP)
                                        CMP     @SP,#037611
                                        BLO     L1$15
                                        BHI     TN8$15
                                        CMP     2(SP),#030242
                                        BHI     TN8$15
                                        BLO     L1$15
                                        CMP     4(SP),#172366
                                        BHI     TN8$15
                                        BLO     L1$15
                                        CMP     6(SP),#065261
```

431

```
2744                                    BLOS    L19S19
2745                            TN8S15: MOV     #040000,0,(SP)
2746                                    MOV     #005221,10.(SP)
2747                                    MOV     #140533,12.(SP)
2748                                    MOV     #115404,14.(SP)
2749                                    MOV     @SP,R0
2750                                    MOV     2(SP),R1
2751                                    MOV     4(SP),R2
2752                                    MOV     6(SP),R3
2753                                    MOV     #062524,-(SP)
2754                                    MOV     #041302,-(SP)
2755                                    MOV     #131727,-(SP)
2756                                    MOV     #140335,-(SP)
2757                                    MOV     R3,-(SP)
2758                                    MOV     R2,-(SP)
2759                                    MOV     R1,-(SP)
2760                                    MOV     R0,-(SP)
2761                                    CLR     -(SP)
2762                                    CLR     -(SP)
2763                                    CLR     -(SP)
2764                                    MOV     #40200,-(SP)
2765                                    MOV     #062524,-(SP)
2766                                    MOV     #041302,-(SP)
2767                                    MOV     #131727,-(SP)
2768                                    MOV     #040335,-(SP)
2769                                    MOV     R3,-(SP)
2770                                    MOV     R2,-(SP)
2771                                    MOV     R1,-(SP)
2772                                    MOV     R0,-(SP)
2773                                    JSR     R4,$POLSH
2774                                    .WORD   $MLD,$SBD,UPS15,$SBD,$DVD,L19S15
2775                            L19S15: MOV     @SP,R0
2776                                    MOV     2(SP),R1
2777                                    MOV     4(SP),R2
2778                                    MOV     6(SP),R3
2779                                    MOV     R3,-(SP)
2780                                    MOV     R2,-(SP)
2781                                    MOV     R1,-(SP)
2782                                    MOV     R0,-(SP)
2783                                    MOV     R3,-(SP)
2784                                    MOV     R2,-(SP)
2785                                    MOV     R1,-(SP)
2786                                    MOV     R0,-(SP)
2787                                    JSR     R4,$POLSH
2788                                    .WORD   $MLD
2789                                    .WORD   $POPR4,PLY$15
2790                            XP0S15: .WORD   $MLD,$ADD,$MLD,$ADD,$MLD,$ADD
2791                                    .WORD   $MLD,$ADD,$MLD,$ADD,$MLD,$ADD
2792                                    .WORD   $MLD,$ADD,$MLD,$ADD,$MLD,$ADD
2793                                    .WORD   $ADD
2794                                    .WORD   SGN$15
2795                                    .WORD   $ADD
2796                                    .WORD   $POPR4
2797                                    .WORD   EXI$15
2798                            EXI$15: TST     (SP)+
```

432


```
2799                                    MOV     (SP)+,R5
2800                                    RTS     R5
2801                            UP$15:  MOV     (SP)+,22.(SP)
2802                                    MOV     (SP)+,22.(SP)
2803                                    MOV     (SP)+,22.(SP)
2804                                    MOV     (SP)+,22.(SP)
2805                                    JMP     @(R4)+
2806                            PLY$15: MOV     #CON$15+8.,R4
2807                                    MOV     #9.,R5
2808                                    BR      PY1$15
2809                            PY2S15: MOV     R3,-(SP)
2810                                    MOV     R2,-(SP)
2811                                    MOV     R1,-(SP)
2812                                    MOV     R0,-(SP)
2813                            PY1S15: MOV     -(R4),-(SP)
2814                                    MOV     -(R4),-(SP)
2815                                    MOV     -(R4),-(SP)
2816                                    MOV     -(R4),-(SP)
2817                                    DEC     R5
2818                                    BGT     PY2S15
2819                                    MOV     #XP0S15,R4
2820                                    JMP     @(R4)+
2821                            SGN$15: TST     10.(SP)
2822                                    BEQ     SGI$15
2823                                    ADD     #100000,@SP
2824                            SGI$15: JMP     @(R4)+
2825                                    .ENDC
2826                                    .IFDF   FPU
2827                            DATAN2: SETD
2828                                    MOV     2(R5),R3
2829                                    MOV     4(R5),R4
2830                                    MOV     @R3,R0
2831                                    MOV     @R4,R1
2832                                    BEQ     INF$15
2833                                    ASL     R0
2834                                    CLRB    R0
2835                                    SWAB    R0
2836                                    ASL     R1
2837                                    CLRB    R1
2838                                    SWAB    R1
2839                                    SUB     R1,R0
2840                                    CMP     #58.,R0
2841                                    BLT     INF$15
2842                                    LDD     PI$15,F3
2843                                    LDD     @R3,F0
2844                                    CFCC
2845                                    BGE     A1P$15
2846                                    NEGD    F3
2847                            A1P$15: LDD     @R4,F1
2848                                    CFCC
2849                                    BLT     A2M$15
2850                                    CLRD    F3
2851                            A2M$15: DIVD    F1,F0
2852                                    BR      ATI$15
2853                            INF$15: LDD     PI2$15,F1
```

433

```
                                        TST     @R3
                                        BGE     EX1$15
                                        NEGD    F1
                                        BR      EX1$15
                                DAFAN1  SETD
                                        CLRD    F3
                                        LDD     @2(R5),F0
                                AT1$15: CLR     R4
                                        CFCC
                                        STD     F3,F5
                                        CLRD    F3
                                        BGE     PLU$15
                                        ABSD    F0
                                        INC     R4
                                PLU$15: LDD     #1.0,F1
                                        CMPD    F0,F1
                                        CFCC
                                        BLE     LE1$15
                                GT1$15: DEC     R4
                                        DIVD    F0,F1
                                        LDD     F1,F0
                                        LDD     P12$15,F3
                                LE1$15: STD     F3,F4
                                        CLRD    F3
                                        CMPD    T15$15,F0
                                        CFCC
                                        BGE     L15$15
                                        LDD     P16$15,F3
                                        LDD     F0,F1
                                        MULD    RT3$15,F0
                                        SUBD    #1.0,F0
                                        ADDD    RT3$15,F1
                                        DIVD    F1,F0
                                L15$15: LDD     F0,F2
                                        MULD    F0,F0
                                        MOV     #FCOS15,R0
                                        MOV     #8.,R1
                                        LDD     (R0)+,F1
                                XPO$15: MULD    F0,F1
                                        DEC     R1
                                        ADDD    (R0)+,F1
                                        BGT     XPO$15
                                        MULD    F2,F1
                                        ADDD    F3,F1
                                        SUBD    F4,F1
                                        TST     R4
                                        BEQ     SG1$15
                                        NEGD    F1
                                SG1$15: ADDD    F0,F1
                                EX1$15: STD     F1,-(SP)
                                        MOV     (SP)+,R0
                                        MOV     (SP)+,R1
                                        MOV     (SP)+,R2
                                        MOV     (SP)+,R3
                                        RTS     R5
```

434


```
                                PI$15:  .WORD   040511,007732
                                        .WORD   121041,064301
                                PI2$15: .WORD   040311,007732
                                        .WORD   121041,064301
                                T15$15: .WORD   037611,030242
                                        .WORD   172360,065261
                                P16$15: .WORD   040000,005221
                                        .WORD   146553,115454
                                RT3$15: .WORD   040337,131727
                                        .WORD   041302,062524
                                        .ENDC
                                FCOS15: .WORD   037665,150707
                                        .WORD   162300,163630
                                        .WORD   137204,143233
                                        .WORD   004010,000413
                                        .WORD   037235,043002
                                        .WORD   027154,142446
                                        .WORD   137272,025671
                                        .WORD   116412,065630
                                        .WORD   037343,107647
                                        .WORD   023625,025401
                                        .WORD   137422,044444
                                        .WORD   071335,116151
                                        .WORD   037514,140314
                                        .WORD   146224,165650
                                        .WORD   137652,125252
                                        .WORD   125252,113602
                                CONS15: .WORD   040200,000000
                                        .WORD   000000,000000
                                        .ENDC
                                        .EOT
```

435

```
                                ;                                   BASIC SOURCE FILE #12
                                ;TITLE  SDVD03
                                ;IFDF   CND$10
                                ;GLOBL  SDVD,$ERRA
                                R0=%0
                                R1=%1
                                R2=%2
                                R3=%3
                                R4=%4
                                R5=%5
                                SP=%6
                                PC=%7
                                F0=%0
                                F1=%1
                                D=0,
                                N=16,
                                Q=16,
                                ;IFDF   FPU
SDVD1:                          ;WORD   170011
                                ;WORD   172520
                                ;WORD   172420
                                ;WORD   174401
                                ;WORD   174040
                                JMP     @(R4)+
                                ;ENDC
                                ;IFNDF  FPU
SDVD1:                          MOV     R4,-(SP)
                                MOV     R5,-(SP)
                                CLR     R0
                                CLR     R1
                                CLR     R2
                                CLR     R3
                                CLR     -(SP)
                                ASL     N+0+2(SP)
                                ROL     @SP
                                CLR     -(SP)
                                TST     0(SP)
                                BEQ     DCHS10
                                BISB    N+1(SP),@SP
                                BEQ     ZERS10
                                BISB    N(SP),R0
                                SWAB    R0
                                SEC
                                ROR     R0
                                BISB    N+3(SP),R0
                                BISB    N+2(SP),R1
                                SWAB    R1
                                BISB    N+5(SP),R1
                                BISB    N+4(SP),R2
                                SWAB    R2
                                BISB    N+7(SP),R2
                                BISB    N+6(SP),R3
                                SWAB    R3
                                ASL     0(SP)
                                ADC     2(SP)
```

436


```
                                CLR     R4
                                BISB    D+1(SP),R4
                                SUB     R4,@SP
                                SWAB    0(SP)
                                SEC
                                ROR     0(SP)
                                MOVB    D+3(SP),D(SP)
                                MOVB    D+2(SP),D+3(SP)
                                MOVB    D+5(SP),D+2(SP)
                                MOVB    D+4(SP),D+5(SP)
                                MOVB    D+7(SP),D+4(SP)
                                MOVB    D+6(SP),D+7(SP)
                                CLRB    D+6(SP)
                                CLR     Q(SP)
                                CLR     Q+2(SP)
                                CLR     Q+4(SP)
                                CMP     R0,0(SP)
                                BHI     DLWS10
                                BLO     DHIS10
                                CMP     R1,D+2(SP)
                                BHI     DLWS10
                                BLO     DHIS10
                                CMP     R2,D+4(SP)
                                BHI     DLWS10
                                BLO     DHIS10
                                CMP     R3,D+6(SP)
                                BHI     DLWS10
                                BNE     DHIS10
                                INC     @SP
                                CLR     R4
                                BR      FLTS10
DCHS16:                         MOV     #1403,R0
                                BR      ECIS10
UNDS16:                         MOV     #4005,R0
ECLS16:                         TST     -(SP)
ECIS16:                         JSR     R5,$ERRA
ZERS16:                         CMP     (SP)+,(SP)+
                                CLR     Q+0+4(SP)
                                CLR     Q+2+4(SP)
                                CLR     Q+4+4(SP)
                                CLR     Q+6+4(SP)
                                BR      RTNS10
DLWS16:                         ROR     R0
                                ROR     R1
                                ROR     R2
                                ROR     R3
                                INC     @SP
DHIS16:                         MOV     #9.,R5
                                JSR     PC,DVIS16
                                MOVB    R4,0(SP)
                                TST     R5
                                BNE     FLIS10
                                MOV     #16.,R5
                                JSR     PC,DVIS16
```

437

```
                                        MOV     R4,Q+2(SP)
                                        TST     R5
                                        BNE     FL1$10
                                        MOV     #16.,R5
                                        JSR     PC,DV1$16
                                        MOV     R4,Q+4(SP)
                                        TST     R5
                                        BNE     FL1$10
                                        MOV     #16.,R5
                                        JSR     PC,DV1$16
                                        BR      FLT$10
                                FL1$16: CLR     R4
                                FLT$16: MOV     (SP)+,R5
                                        ADD     #200,R5
                                        BLE     UND$10
                                        CMP     #377,R5
                                        BLT     OVR$10
                                        MOVB    R5,Q+1-2(SP)
                                SGN$16: ROR     (SP)+
                                        ROR     Q+6-4(SP)
                                        ROR     Q+2-4(SP)
                                        ROR     Q+4-4(SP)
                                        ROR     R4
                                        ADC     R4
                                        ADC     Q+4-4(SP)
                                        ADC     Q+2-4(SP)
                                        ADC     Q+6-4(SP)
                                        MOV     R4,Q+0-4(SP)
                                        BCS     OV1$10
                                        BVS     OV1$10
                                RTN$16: MOV     (SP)+,R5
                                        MOV     (SP)+,R4
                                        ADD     #8.,SP
                                        JMP     @(R4)+
                                OV1$16: TST     -(SP)
                                OVR$16: MOV     #2883,R0
                                        BR      EQL$10
                                DV1$16: ASL     R4
                                        ASL     R3
                                        ROL     R2
                                        ROL     R1
                                        ROL     R0
                                        BCS     GO$16
                                        CMP     0+6+2(SP),R0
                                        BHI     NGO$10
                                        BLO     GO$16
                                        CMP     0+2+2(SP),R1
                                        BHI     NGO$10
                                        BLO     GO$16
                                        CMP     0+4+2(SP),R2
                                        BHI     NGO$10
                                        BLO     GO$16
                                        CMP     0+6+2(SP),R3
                                        BHI     NGO$10
                                        BEQ     NGO$10
```

438


```
                                GO$16:  SUB     0+6+2(SP),R3
                                        SBC     R2
                                        SBC     R1
                                        SBC     R0
                                        SUB     0+4+2(SP),R2
                                        SBC     R1
                                        SBC     R0
                                        SUB     0+2+2(SP),R1
                                        SBC     R0
                                        SUB     0+0+2(SP),R0
                                        INC     R4
                                NGO$16: DEC     R5
                                        BGT     DV1$10
                                        RTS     PC
                                NGO$16: INC     R4
                                        BR      EQ1$10
                                EQ2$16: ASL     R4
                                EQ1$16: DEC     R5
                                        BGT     EQ2$10
                                        INC     R5
                                RTS$16: RTS     PC
                                        .ENDC
                                        .ENDC
                                        .TITLE  $DV103
                                        .IFDF   CND$17
                                        .GLOBL  $DV1,$ERR
                                        R0=%0
                                        R1=%1
                                        R2=%2
                                        R3=%3
                                        R4=%4
                                        R5=%5
                                        SP=%6
                                        MQ=177304
                                        .IFNDF  EAE&MULDIV
                                $DV1:   CLR     R0
                                        MOV     (SP)+,R1
                                        BGT     P1$17
                                        BEQ     CHK$17
                                        INC     R0
                                        NEG     R1
                                P1$17:  MOV     @SP,R3
                                        BGT     P2$17
                                        BEQ     ERR$17
                                        INC     R0
                                        NEG     R3
                                P2$17:  MOV     R4,-(SP)
                                        MOV     #0.,R4
                                        CLR     R2
                                        SWAB    R3
                                        BEQ     DIV$17
                                        ASL     R4
                                        SWAB    R3
                                DIV$17: ASL     R3
                                        ROL     R2
```

439

```
3161                                            BEQ     LUPS17
3162                                            INC     R3
3163                                            SUB     R1,R2
3164                                            BHIS    LUPS17
3165                                            ADD     R1,R2
3166                                            DEC     R3
3167                                    LUPS17: DEC     R4
3168                                            BGT     DIVS17
3169                                            MOV     (SP)+,R4
3170                                            NEG     R3
3171                                            ASR     R0
3172                                            BCS     P3S17
3173                                            NEG     R3
3174                                            BVS     CHKS17
3175                                    P3S17:  MOV     R3,@SP
3176                                            JMP     @(R4)+
3177                                    ZERS17: CLR     @SP
3178                                            JMP     @(R4)+
3179                                            .ENDC
3180                                            .IFDF   EAE
3181                                    $DVI:   MOV     #MQ,R0
3182                                            MOV     (SP)+,R1
3183                                            BEQ     CHKS17
3184                                            MOV     (SP)+,@R0
3185                                            TST     -(R0)
3186                                            MOV     R1,-(R0)
3187                                            CMP     (R0)+,(R0)+
3188                                            MOV     @R0,-(SP)
3189                                            JMP     @(R4)+
3190                                            .ENDC
3191                                            .IFDF   MULDIV
3192                                    $DVI:   MOV     2(SP),R1
3193                                            .WORD   006700
3194                                            .WORD   071020
3195                                            MOV     R0,@SP
3196                                            BCS     CHKS17
3197                                            JMP     @(R4)+
3198                                            .ENDC
3199                                    CHKS17: JSR     R5,$ERR
3200                                            JMP     @(R4)+
3201                                            .BYTE   3
3202                                            .BYTE   5
3203                                            .ENDC
3204                                            .TITLE  $DVR00
3205                                            .IFDF   CND$10
3206                                            .GLOBL  $DVR,$ERRA
3207            000000                          R0=%0
3208            000001                          R1=%1
3209            000002                          R2=%2
3210            000003                          R3=%3
3211            000004                          R4=%4
3212            000005                          R5=%5
3213            000006                          SP=%6
3214            000007                          PC=%7
3215            177304                          MQ=177304
```

440


```
3216            177312                          NOR=177312
3217            177314                          LSH=177314
3218            177316                          ASH=177316
3219            000000                          F0=%0
3220            000001                          F1=%1
3221            000010                          D=8.
3222            000014                          N=12.
3223            000014                          Q=12.
3224                                            .IFDF   FPU
3225                                    $DVR:   .WORD   170001
3226                                            .WORD   172520
3227                                            .WORD   172420
3228                                            .WORD   174401
3229                                            .WORD   174040
3230                                            JMP     @(R4)+
3231                                            .ENDC
3232                                            .IFNDF  FPU
3233    001234' 010446                  $DVR:   MOV     R4,-(SP)
3234    001236' 010546                          MOV     R5,-(SP)
3235    001240' 005000                          CLR     R0
3236    001242' 005001                          CLR     R1
3237    001244' 005046                          CLR     -(SP)
3238    001246' 006366  000012                  ASL     N+0-2(SP)
3239    001252' 006116                          ROL     @SP
3240    001254' 005046                          CLR     -(SP)
3241    001256' 005766  000010                  TST     D(SP)
3242    001262' 001456                          BEQ     OCHS10
3243    001264' 156616  000015                  BISB    N+1(SP),@SP
3244    001270' 001451                          BEQ     ZERS10
3245    001272' 156600  000014                  BISB    N(SP),R0
3246    001276' 000300                          SWAB    R0
3247    001300' 000261                          SEC
3248    001302' 006000                          ROR     R0
3249    001304' 156600  000017                  BISB    N+3(SP),R0
3250    001310' 156601  000016                  BISB    N+2(SP),R1
3251    001314' 000301                          SWAB    R1
3252    001316' 005002                          CLR     R2
3253    001320' 005003                          CLR     R3
3254    001322' 006366  000010                  ASL     D(SP)
3255    001326' 005566  000002                  ADC     2(SP)
3256    001332' 156602  000011                  BISB    D+1(SP),R2
3257    001336' 160216                          SUB     R2,@SP
3258    001340' 005002                          CLR     R2
3259    001342' 156602  000010                  BISB    D(SP),R2
3260    001346' 000302                          SWAB    R2
3261    001350' 000261                          SEC
3262    001352' 006002                          ROR     R2
3263    001354' 156602  000013                  BISB    D+3(SP),R2
3264    001360' 156603  000012                  BISB    D+2(SP),R3
3265    001364' 000303                          SWAB    R3
3266                                            .IFDF   EAE!MULDIV
3267                                            CLC
3268                                            ROR     R0
3269                                            ROR     R1
3270                                            ROR     R2
```

441

```
3271                                            RQR     R3
3272                                            .ENDC
3273  001366' 020002                           CMP     R0,R2
3274  001370' 103440                           BLO     DH1$10
3275                                            .IFNDF  EAE&MULDIV
3276  001372' 101034                           BHI     DLW$10
3277  001374' 020103                           CMP     R1,R3
3278  001376' 101032                           BHI     DLW$10
3279  001400' 001034                           BNE     DH1$10
3280  001402' 005066  000014                   CLR     Q(SP)
3281  001406' 005216                           INC     @SP
3282  001410' 005003                           CLR     R3
3283  001412' 000445                           BR      FLT$10
3284                                            .ENDC
3285                                            .IFDF   EAE!MULDIV
3286                                            BHIS    DLW$10
3287                                            .ENDC
3288  001414' 022626                  ZERS10!   CMP     (SP)+,(SP)+
3289  001416' 000415                           BR      ECL$10
3290  001420' 005726                  DCW$10!   TST     (SP)+
3291  001422' 012700  004003                   MOV     #4003,R0
3292  001426' 000406                           BR      ECL$10
3293  001430' 005746                  OV1$10!   TST     -(SP)
3294  001432' 012700  003003          OVR$10!   MOV     #3003,R0
3295  001436' 000402                           BR      ECL$10
3296  001440' 012700  001405          UNO$10!   MOV     #1405,R0
3297  001444' 005726                  ECL$10!   TST     (SP)+
3298  001446' 004567  002576                   JSR     R5,$ERRA
3299  001452' 005066  000010          EC1$10!   CLR     Q+8.+4(SP)
3300  001456' 005066  000012                   CLR     Q+2.+4(SP)
3301  001462' 000445                           BR      RTN$10
3302  001464' 006000                  DLW$10!   RQR     R0
3303  001466' 006001                           RQR     R1
3304  001470' 005216                           INC     @SP
3305                                            .IFNDF  EAE&MULDIV
3306  001472' 012704  000011          DH1$10!   MOV     #9.,R4
3307  001476' 004767  000104                   JSR     PC,DV1$10
3308  001502' 110566  000014                   MOVB    R5,Q(SP)
3309
3310  001506' 005704                           TST     R4
3311  001510' 001402                           BEQ     NT0$10
3312  001512' 005005                           CLR     R5
3313  001514' 000404                           BR      FLT$10
3314  001516' 012704  000020          NT0$10!   MOV     #16.,R4
3315  001522' 004767  000060                   JSR     PC,DV1$10
3316                                            .ENDC
3317                                            .IFDF   EAE!MULDIV
3318                                  DH1$10!   CLC
3319                                            RQR     R3
3320                                            RQR     R0
3321                                            RQR     R1
3322                                            .ENDC
3323                                            .IFDF   EAE
3324                                            MOV     #MQ,R3
3325                                            MOV     R1,@R3
```

442


```
3326                                            MOV     R0,-(R5)
3327                                            MOV     R2,-(R5)
3328                                            TST     (R5)+
3329                                            MOV     (R5)+,R1
3330                                            MOV     (R5)+,R4
3331                                            MOV     R3,@R5
3332                                            TST     -(R5)
3333                                            ASR     R1
3334                                            SUB     R1,-(R5)
3335                                            DEC     @#ASH
3336                                            MOV     R2,-(R5)
3337                                            CMP     (R5)+,(R5)+
3338                                            NEG     @R5
3339                                            MOV     #2,@#ASH
3340                                            ADD     R4,-(R5)
3341                                            CLR     @#NOR
3342                                            SUB     @#NOR,@SP
3343                                            MOV     #-4,@#LSH
3344                                            MOV     (R5)+,Q(SP)
3345                                            MOV     @R5,R3
3346                                            .ENDC
3347                                            .IFDF   MULDIV
3348                                            MOV     R0,R4
3349                                            MOV     R1,R5
3350                                            .WORD   071402
3351                                            MOV     R5,R1
3352                                            MOV     R4,R0
3353                                            .WORD   070403
3354                                            ASR     R1
3355                                            SUB     R1,R4
3356                                            .WORD   073427,-1
3357                                            .WORD   071402
3358                                            NEG     R4
3359                                            .WORD   073427,-14.
3360                                            ADD     R0,R4
3361                                  NBT$10!   .WORD   073427,1
3362                                            BMI     NB1$10
3363                                            DEC     @SP
3364                                            BR      NBT$10
3365                                  NB1$10!   .WORD   073427,-7
3366                                            MOV     R4,Q(SP)
3367                                            .ENDC
3368  001526' 012604                  FLT$10!   MOV     (SP)+,R4
3369  001530' 062704  000200                   ADD     #200,R4
3370  001534' 003741                           BLE     UNO$10
3371  001536' 022704  000377                   CMP     #377,R4
3372  001542' 002733                           BLT     OVR$10
3373  001544' 110466  000013                   MOVB    R4,Q+1.+2(SP)
3374  001550' 006020                  SGN$10!   RQR     (SP)+
3375  001552' 006066  000010                   RQR     Q+8.+4(SP)
3376  001556' 006005                           RQR     R5
3377  001560' 005505                           ADC     R5
3378  001562' 005566  000010                   ADC     Q+8.+4(SP)
3379  001566' 010566  000012                   MOV     R5,Q+2.+4(SP)
3380  001572' 103716                           BCS     OV1$10
```

443

```
3381  001574' 102715                  BVS     OV1$18
3382  001576' 012605              RTN$18: MOV     (SP)+,R5
3383  001600' 012604                      MOV     (SP)+,R4
3384  001602' 022626                      CMP     (SP)+,(SP)+
3385  001604' 000134                      JMP     @(R4)+
3386                                      .IFNDF  EAE&MULDIV
3387  001606' 006305              DVI$18: ASL     R5
3388  001610' 006301                      ASL     R1
3389  001612' 006100                      ROL     R0
3390  001614' 103406                      BCS     GO$18
3391  001616' 020200                      CMP     R2,R0
3392  001620' 101010                      BHI     NGO$18
3393  001622' 103403                      BLO     GO$18
3394  001624' 020301                      CMP     R3,R1
3395  001626' 101005                      BHI     NGO$18
3396  001630' 001407                      BEQ     NGO$18
3397  001632' 160301              GO$18:  SUB     R3,R1
3398  001634' 005600                      SBC     R0
3399  001636' 160200                      SUB     R2,R0
3400  001640' 005205                      INC     R5
3401  001642' 005304              NGO$18: DEC     R4
3402  001644' 003360                      BGT     DVI$18
3403  001646' 000207                      RTS     PC
3404  001650' 005205              NGO$18: INC     R5
3405  001652' 000401                      BR      EQ1$19
3406  001654' 006305              EQ2$18: ASL     R5
3407  001656' 005304              EQ1$18: DEC     R4
3408  001660' 003375                      BGT     EQ2$18
3409  001662' 005204                      INC     R4
3410  001664' 000207              RTS$18: RTS     PC
3411                                      .ENDC
3412                                      .ENDC
3413                                      .ENDC
3414                                      .TITLE  $DXP05
3415                                      .IFDF   GND$19
3416                                      .GLOBL  DEXP,$ERRA
3417                                      .IFNDF  FPU
3418                                      .GLOBL  $ADD,$SBD,$MLD,$DVD,$ID,$DI,$POLSH,$POPR4
3419                                      .ENDC
3420                                      R0=%0
3421                                      R1=%1
3422                                      R2=%2
3423                                      R3=%3
3424                                      R4=%4
3425                                      R5=%5
3426                                      SP=%6
3427                                      F0=%0
3428                                      F1=%1
3429                                      F2=%2
3430                                      F3=%3
3431                                      .IFDF   FPU
3432                              DEXP:   MOV     @2(R5),R0
3433                                      .ENDC
3434                                      .IFNDF  FPU
3435                              DEXP:   MOV     R5,-(SP)
```

444


```
3436                                    MOV     2(R5),R4
3437                                    MOV     @R4,R0
3438                                    .ENDC
3439                                    BGT     POS$19
3440                                    CMP     R0,#141662
3441                                    BHI     ZER$19
3442                                    BR      SMT$19
3443                            POS$19: CMP     R0,#41660
3444                                    BHI     OVR$19
3445                            SMT$19: ASL     R0
3446                                    CMP     R0,#43000
3447                                    BLO     ONE$19
3448                                    .IFNDF  FPU
3449                                    SUB     #28.,SP
3450                                    ADD     #8.,R4
3451                                    MOV     -(R4),-(SP)
3452                                    MOV     -(R4),-(SP)
3453                                    MOV     -(R4),-(SP)
3454                                    MOV     -(R4),-(SP)
3455                                    MOV     #013701,-(SP)
3456                                    MOV     #024534,-(SP)
3457                                    MOV     #125073,-(SP)
3458                                    MOV     #40270,-(SP)
3459                                    JSR     R4,$POLSH
3460                                    .WORD   $MLD
3461                                    .WORD   DUP$19
3462                                    .WORD   $DI
3463                                    .WORD   ADJ$19
3464                                    .WORD   $ID
3465                                    .WORD   $SBD
3466                                    .WORD   M16$19
3467                                    .WORD   DUP$19
3468                                    .WORD   $DI
3469                                    .WORD   DSV$19
3470                                    .WORD   $ID
3471                                    .WORD   $SBD,Q16$19
3472                                    .WORD   DUP$19,DUP$19
3473                                    .WORD   $MLD
3474                                    .WORD   $POPR4
3475                                    .WORD   UPL$19
3476                            ONE$19: MOV     #40200,R0
3477                                    BR      Z1$19
3478                            OVR$19: MOV     #1004,R0
3479                                    BR      ECL$19
3480                            ZER$19: MOV     #2005,R0
3481                            ECL$19: JSR     R5,$ERRA
3482                                    CLR     R0
3483                            Z1$19:  CLR     R1
3484                                    CLR     R2
3485                                    CLR     R3
3486                                    BR      OUT$19
3487                            UPL$19: MOV     #033343,-(SP)
3488                                    MOV     #015345,-(SP)
3489                                    MOV     #152405,-(SP)
3490                                    MOV     #040746,-(SP)
```

445

```
3491                                    MOV     R3,-(SP)
3492                                    MOV     R2,-(SP)
3493                                    MOV     R1,-(SP)
3494                                    MOV     R0,-(SP)
3495                                    MOV     #153703,-(SP)
3496                                    MOV     #153011,-(SP)
3497                                    MOV     #113300,-(SP)
3498                                    MOV     #037174,-(SP)
3499                                    MOV     R3,-(SP)
3500                                    MOV     R2,-(SP)
3501                                    MOV     R1,-(SP)
3502                                    MOV     R0,-(SP)
3503                                    MOV     #171002,-(SP)
3504                                    MOV     #074435,-(SP)
3505                                    MOV     #101232,-(SP)
3506                                    MOV     #041240,-(SP)
3507                                    JSR     R4,$POLSH
3508                                    .WORD   $ADD,AUPS19
3509                                    .WORD   $MLD,$ADD,$MLD
3510                                    .WORD   TWCS19
3511                                    .WORD   $ADD,ABPS19
3512                                    .WORD   $SBD,$DVD
3513                                    .WORD   SCL$19
3514                            SCL$19: MOV     #RT2$19+8.,R5
3515                            ASR$19: ASR     0.(SP)
3516                                    BCC     NML$19
3517                                    MOV     -(R5),-(SP)
3518                                    MOV     -(R5),-(SP)
3519                                    MOV     -(R5),-(SP)
3520                                    MOV     -(R5),-(SP)
3521                                    JSR     R4,$POLSH
3522                                    .WORD   $MLD,ASR$19
3523                            NML$19: BEQ     SC1$19
3524                                    SUB     #8.,R5
3525                                    BR      ASR$19
3526                            SC1$19: MOV     (SP)+,R0
3527                                    MOV     (SP)+,R1
3528                                    MOV     (SP)+,R2
3529                                    MOV     (SP)+,R3
3530                                    TST     (SP)+
3531                                    MOV     (SP)+,R4
3532                                    SWAB    R4
3533                                    CLRB    R4
3534                                    ASR     R4
3535                                    ADD     R4,R0
3536                                    BMI     OVR$19
3537                            OUT$19: MOV     (SP)+,R5
3538                                    RTS     R5
3539                            ADJ$19: TST     02(R5)
3540                                    BGE     ARN$19
3541                                    DEC     @SP
3542                            ARN$19: MOV     @SP,20.(SP)
3543                                    JMP     @(R4)+
3544                            M16$19: ADD     #1000.,@SP
3545                                    JMP     @(R4)+
```

446


```
3546                            D16$19: SUB     #1000.,@SP
3547                                    BPL     D6R$19
3548                                    CLR     @SP
3549                            D6R$19: JMP     @(R4)+
3550                            DSV$19: MOV     @SP,20.(SP)
3551                                    JMP     @(R4)+
3552                            AUP$19: MOV     (SP)+,38.(SP)
3553                                    MOV     (SP)+,38.(SP)
3554                                    MOV     (SP)+,38.(SP)
3555                                    MOV     (SP)+,38.(SP)
3556                                    JMP     @(R4)+
3557                            ABP$19: MOV     (SP)+,22.(SP)
3558                                    MOV     (SP)+,22.(SP)
3559                                    MOV     (SP)+,22.(SP)
3560                                    MOV     (SP)+,22.(SP)
3561                                    JMP     @(R4)+
3562                            DUP$19: MOV     6(SP),-(SP)
3563                                    MOV     6(SP),-(SP)
3564                                    MOV     6(SP),-(SP)
3565                                    MOV     6(SP),-(SP)
3566                                    JMP     @(R4)+
3567                            TWC$19: MOV     #8.,R0
3568                            TWI$19: MOV     14.(SP),-(SP)
3569                                    DEC     R0
3570                                    BGT     TWI$19
3571                                    JMP     @(R4)+
3572                                    .WORD   040269,002363,031771,157145
3573                                    .WORD   040230,033760,050615,134251
3574                                    .WORD   040213,112701,161752,105727
3575                            RT2$19: .WORD   040209,125983,063714,044173
3576                                    .ENDC
3577                                    .IFDF   FPU
3578                                    SETD
3579                                    SETI
3580                                    MOV     #FCO$19,R0
3581                                    LDD     @2(R5),F2
3582                                    MODD    (R0)+,F2
3583                                    STCDI   F3,R4
3584                                    TSTD    F2
3585                                    CFCC
3586                                    BGE     M16$19
3587                                    ADDD    #1.0,F2
3588                                    DEC     R4
3589                            M16$19: MODD    #16.0,F2
3590                                    STCDI   F3,R5
3591                                    DIVD    #16.0,F2
3592                                    LDD     F2,F3
3593                                    MULD    F3,F3
3594                                    LDD     F3,F1
3595                                    ADDD    (R0)+,F1
3596                                    MULD    (R0)+,F3
3597                                    ADDD    (R0)+,F3
3598                                    MULD    F2,F3
3599                                    LDD     F1,F0
3600                                    ADDD    F3,F0
```

447

```
                                        SUBD    F3,F1
                                        DIVD    F3,F0
                                SCL$19: ASR     R3
                                        BCC     NML$19
                                        MULD    (R0)+,F0
                                        BR      SCL$19
                                NML$19: BEQ     SCI$19
                                        ADD     #8.,R0
                                        BR      SCL$19
                                SCI$19: STD     F0,-($SP)
                                        MOV     ($SP)+,R0
                                        MOV     ($SP)+,R1
                                        MOV     ($SP)+,R2
                                        MOV     ($SP)+,R3
                                        SWAB    R4
                                        CLRB    R4
                                        ASR     R4
                                        ADD     R4,R0
                                        BMI     OVR$19
                                        RTS     R5
                                ONE$19: MOV     #40200,R0
                                        BR      Z1$19
                                OVR$19: MOV     #1004,R0
                                        BR      ECL$19
                                ZER$19: MOV     #2005,R0
                                ECL$19: JSR     R5,$ERRA
                                        CLR     R0
                                Z1$19:  CLR     R1
                                        CLR     R2
                                        CLR     R3
                                        RTS     R5
                                FCO$19: .WORD   40270,125073,024534,013701
                                        .WORD   041240,101232,074433,171042
                                        .WORD   037156,113360,153001,153703
                                        .WORD   040746,152405,015345,033343
                                        .WORD   040205,125303,063714,044173
                                        .WORD   040214,113701,161772,105727
                                        .WORD   040230,033760,050615,134251
                                        .WORD   040265,002363,031771,137145
                                        .ENDC
                                        .ENDC
                                        .TITLE  $EXP04
                                        .IFDF   CND$20

                                        .GLOBL  EXP,$ERRA
                                        .IFNDF  FPU
                                        .GLOBL  $ADR,$SBR,$MLR,$OVR,$IR,$RI,$POLSH
                                        .ENDC
000000                                  R0=%0
000001                                  R1=%1
000002                                  R2=%2
000003                                  R3=%3
000004                                  R4=%4
000005                                  R5=%5
000006                                  SP=%6
```

448


```
000007                                  PC=%7
000000                                  F0=%0
000001                                  F1=%1
000002                                  F2=%2
000003                                  F3=%3
001666' 016504 000002           EXP:    MOV     2(R5),R4
001672' 011400                          MOV     @R4,R0
001674' 003004                          BGT     POS$20
001676' 020027 141662                   CMP     R0,#141662
001702' 101146                          BHI     ZER$20
001704' 000403                          BR      SMT$20
001706' 020027 041660           POS$20: CMP     R0,#41660
001712' 101137                          BHI     OVR$20
001714' 006300                  SMT$20: ASL     R0
001716' 020027 063000                   CMP     R0,#63000
001722' 103527                          BLO     ONE$20
                                        .IFNDF  FPU
001724' 005746                          TST     -(SP)
001726' 005046                          CLR     -(SP)
001730' 012746 040200                   MOV     #40200,-(SP)
001734' 016446 000002                   MOV     2(R4),-(SP)
001740' 011446                          MOV     @R4,-(SP)
001742' 016446 000002                   MOV     2(R4),-(SP)
001746' 011446                          MOV     @R4,-(SP)
001750' 004467 002122                   JSR     R0,$POLSH
002044'                                 .WORD   PLE$20
002322'                                 .WORD   $MLR
002714'                                 .WORD   $RI
002056'                                 .WORD   ESV$20
002236'                                 .WORD   $IR
002044'                                 .WORD   PLE$20
001234'                                 .WORD   $OVR
000002'                                 .WORD   $SBR
002064'                                 .WORD   CFR$20
002322'                                 .WORD   $MLR
000006'                                 .WORD   $ADR
001234'                                 .WORD   $OVR
000006'                                 .WORD   $ADR
000006'                                 .WORD   $ADR
001234'                                 .WORD   $OVR
002024'                                 .WORD   INC$20
000006'                                 .WORD   $ADR
002032'                                 .WORD   DUP$20
002322'                                 .WORD   $MLR
002160'                                 .WORD   SCL$20
002716 100200           INC$20: ADD     #100200,@SP
000134                          JMP     @(R4)+
016646 000002           DUP$20: MOV     2(SP),-(SP)
016646 000002                   MOV     2(SP),-(SP)
000134                          JMP     @(R4)+
012746 125073           PLE$20: MOV     #125073,-(SP)
012746 040270                   MOV     #40270,-(SP)
000134                          JMP     @(R4)+
011646 000012           ESV$20: MOV     @SP,12.(SP)
000134                          JMP     @(R4)+
```

449

```
3711  002064' 006116                  CFRS20: ROL     @SP
3712  002066' 006100                          ROL     R0
3713  002070' 162716  000400                  SUB     #400,@SP
3714  002074' 101430                          BLOS    ZERS20
3715  002076' 006000                          ROR     R0
3716  002100' 006016                          ROR     @SP
3717  002102' 011600                          MOV     @SP,R0
3718  002104' 016601  000002                  MOV     2(SP),R1
3719  002110' 012746  036602                  MOV     #036602,-(SP)
3720  002114' 012746  141100                  MOV     #141100,-(SP)
3721  002120' 010146                          MOV     R1,-(SP)
3722  002122' 010046                          MOV     R0,-(SP)
3723  002124' 012746  071571                  MOV     #071571,-(SP)
3724  002130' 012746  042420                  MOV     #042420,-(SP)
3725  002134' 012746  056133                  MOV     #056133,-(SP)
3726  002140' 012746  041500                  MOV     #041500,-(SP)
3727  002144' 010146                          MOV     R1,-(SP)
3728  002146' 010046                          MOV     R0,-(SP)
3729  002150' 010146                          MOV     R1,-(SP)
3730  002152' 010046                          MOV     R0,-(SP)
3731  002154' 000134                          JMP     @(R4)+
3732                                          ;ENDC
3733                                          ;IFDF   FPU
3734                                          $EID
3735                                          $ETI
3736                                          MOV     #FCOS20,R0
3737                                          LDCFD   @R4,F2
3738                                          MODD    (R0)+,F2
3739                                          STCDI   F3,R4
3740                                          LDD     #1.0,F0
3741                                          DIVD    (R0)+,F2
3742                                          $ETF
3743                                          LDCDF   F2,F2
3744                                          CFCC
3745                                          BEQ     SC1S20
3746                                          LDF     F2,F3
3747                                          MULF    F3,F3
3748                                          ADDF    (R0)+,F3
3749                                          LDF     (R0)+,F1
3750                                          DIVF    F3,F1
3751                                          ADDF    F2,F1
3752                                          ADDF    (R0)+,F1
3753                                          DIVF    F1,F2
3754                                          MULF    #2.0,F2
3755                                          SUBF    F2,F0
3756                                          MULF    F0,F0
3757                                  SC1S20: STF     F0,-(SP)
3758                                          ;ENDC
3759                                          ;IFNDF  FPU
3760  002156' 022626                  ZERS20: CMP     (SP)+,(SP)+
3761                                          ;ENDC
3762  002160' 012600                  SCLS20: MOV     (SP)+,R0
3763  002162' 012601                          MOV     (SP)+,R1
3764                                          ;IFNDF  FPU
3765  002164' 012604                          MOV     (SP)+,R4
```

450


```
3766                                          ;ENDC
3767  002166' 000304                          SWAB    R4
3768  002170' 105004                          CLRB    R4
3769  002172' 006204                          ASR     R4
3770  002174' 060400                          ADD     R4,R0
3771  002176' 100405                          BMI     OVRS20
3772  002200' 000205                          RTS     R5
3773  002202' 005001                  ONES20: CLR     R1
3774  002204' 012700  040200                  MOV     #40200,R0
3775  002210' 000205                          RTS     R5
3776  002212' 012700  002404          OVRS20: MOV     #2404,R0
3777  002216' 000402                          BR      ECLS20
3778  002220' 012700  002405          ZERS20: MOV     #2405,R0
3779  002224' 004567  002020          ECLS20: JSR     R5,SERRA
3780  002230' 005000                          CLR     R0
3781  002232' 005001                          CLR     R1
3782  002234' 000205                          RTS     R5
3783                                          ;IFDF   FPU
3784                                  FCOS20: ;WORD   040270,125073
3785                                          ;WORD   024534,013761
3786                                          ;WORD   040470,125073
3787                                          ;WORD   024534,013761
3788                                          ;WORD   041500,056133
3789                                          ;WORD   042420,071571
3790                                          ;WORD   141100,036602
3791                                          ;ENDC
3792                                          ;ENDC
3793                                          ;EOT
3794
```

451

```
                                ;                                   BASIC SOURCE FILE #13
                                        .TITLE  $FCL02
                                        .IFDF   CNDS21
                                        .GLOBL  $FCALL
                                        R0=%0
                                        R4=%4
                                        R5=%5
                                        SP=%6
                                $FCALL: MOV     #RET$21,-(SP)
                                        MOV     #137,-(SP)
                                        MOV     R5,-(SP)
                                        MOV     #401,-(SP)
                                        MOV     SP,R5
                                        JSR     R0,@R4
                                RET$21: ADD     #8,,SP
                                        JMP     @(SP)+
                                        .ENDC
                                        .TITLE  $FIX02
                                        .IFDF   CNDS22
                                        .GLOBL  IFIX,$R1,$POLSH
                                        R0=%0
                                        R4=%4
                                        R5=%5
                                        SP=%6
                                IFIX:   MOV     2(R5),R4
                                        MOV     2(R4),-(SP)
                                        MOV     @R4,-(SP)
                                RNDS22: JSR     R4,$POLSH
                                        .WORD   $R1,UPL$22
                                UPL$22: MOV     (SP)+,R0
                                        RTS     R5
                                        .ENDC
                                        .TITLE  $FLT02
                                        .IFDF   CNDS23
                                        .GLOBL  FLOAT,$IR,$POLSH,$POPR3
                                        R0=%0
                                        R1=%1
                                        R4=%4
                                        R5=%5
                                        SP=%6
                                FLOAT:  MOV     @2(R5),-(SP)
                                        JSR     R4,$POLSH
                                        .WORD   $IR
                                        .WORD   $POPR3
                                        .WORD   UPL$23
                                UPL$23: RTS     R5
                                        .ENDC
                                        .TITLE  $ICI02
                                        .IFDF   CNDS24
                                        .GLOBL  $ICI,$OCI
                                        R0=%0
                                        R1=%1
                                        R2=%2
                                        SP=%6
                                        PC=%7
```

452


```
                                $OCI:   MOV     #67,-(SP)
                                        BR      GO$24
                                $ICI:   MOV     #471,-(SP)
                                GO$24:  MOV     R1,-(SP)
                                        MOV     8.(SP),R1
                                        ADD     6(SP),8.(SP)
                                        MOV     4(SP),6(SP)
                                        MOV     R0,4(SP)
                                        MOV     R2,-(SP)
                                        CLR     -(SP)
                                        CLR     R0
                                ST?$24: MOVB    (R1)+,R2
                                        BIC     #177600,R2
                                        CMPB    R2,#'
                                        BNE     SG$$24
                                        CMP     R1,12.(SP)
                                        BLT     ST?$24
                                        BR      SGN$24
                                SG$$24: TSTB    7(SP)
                                        BNE     SN1$24
                                        INC     @SP
                                        BR      NCK$24
                                SN[$24: CMPB    R2,#'+
                                        BEQ     FL0$24  +
                                        CMPB    R2,#'-
                                        BNE     NCK$24
                                        INC     @SP
                                        BR      FL0$24
                                NX?$24: MOVB    (R1)+,R2
                                        BIC     #177600,R2
                                        CMPB    R2,#'
                                        BNE     NCK$24
                                        MOVB    #60,R2
                                NCK$24: CMPB    R2,#'9
                                        BLT     ERR$24
                                        CMPB    R2,6(SP)
                                        BGT     ERR$24
                                        SUB     #60,R2
                                        TSTB    7(SP)
                                        BEQ     DCL$24
                                        ASL     R0
                                        BVS     ERR$24
                                        SUB     R0,R2
                                        ASL     R0
                                        BVS     ERR$24
                                        ASL     R0
                                        BVS     ERR$24
                                        SUB     R2,R0
                                        BVS     ERR$24
                                FL0$24: CMP     R1,12.(SP)
                                        BLT     NX?$24
                                SGN$24: ROR     (SP)+
                                        BCS     ONE$24
                                        NEG     R0
                                        BVS     NOM$24
```

453

```
3905                                    CLC
3906                            ONES24: MOV     (SP)+,R2
3907                                    MOV     (SP)+,R1
3908                                    ROL     (SP)+
3909                                    MOV     R0,4(SP)
3910                                    MOV     (SP)+,R0
3911                                    RTS     PC
3912                            ERRS24: TST     (SP)+
3913                            NGMS24: CLR     R0
3914                                    COM     4(SP)
3915                                    BR      ONES24
3916                            OCLS24: ROL     R0
3917                                    BCS     ERRS24
3918                                    ROL     R0
3919                                    BCS     ERRS24
3920                                    ROL     R0
3921                                    BCS     ERRS24
3922                                    ADD     R2,R0
3923                                    BR      FLDS24
3924                                    .ENDC
3925                                    .TITLE  SIC002
3926                                    .IFDF   GNOS25
3927
3928                                    .GLOBL  SIC0,SOC0
3929                                    R0=%0
3930                                    R1=%1
3931                                    R2=%2
3932                                    R3=%3
3933                                    R4=%4
3934                                    SP=%6
3935                                    PC=%7
3936                            SOC0:   MOV     #OCTS25-RELS25,R0
3937                                    BR      GOS25
3938                            SIC0:   MOV     #DECS25-RELS25,R0
3939                            GOS25:  MOV     R4,-(SP)
3940                                    MOV     8.(SP),R3
3941                                    MOV     6.(SP),R2
3942                                    BGE     LPS525
3943                                    CLR     R2
3944                                    CLR     6(SP)
3945                            LPS525: MOV     4.(SP),R4
3946                                    MOV     #' ,-(SP)
3947                                    CMP     R0,#OCTS25-RELS25
3948                                    BEQ     POS525
3949                                    TST     R4
3950                                    BGE     POS525
3951                                    NEG     R4
3952                                    MOV     #'-,@SP
3953                            POS525: CLR     -(SP)
3954                                    ADD     PC,R0
3955                            RELS25:
3956                            TST525: TST     @R0
3957                                    BEQ     MOVS25
3958                                    CLR     R1
3959                            SUBS25: SUB     @R0,R4
```

454


```
3960                                    BLO     BACS25
3961                                    INC     R1
3962                                    BR      SUBS25
3963                            BACS25: ADD     (R0)+,R4
3964                                    TST     R1
3965                                    BNE     NZES25
3966                                    TST     @SP
3967                                    BEQ     TSTS25
3968                            NZES25: ADD     #60,R1
3969                                    MOV     R1,-(SP)
3970                                    BR      TSTS25
3971                            MOVS25: ADD     R2,R3
3972                                    ADD     #60,R4
3973                                    MOVB    R4,-(R3)
3974                            DCRS25: DEC     R2
3975                                    BLE     FULS25
3976                                    MOVB    (SP)+,-(R3)
3977                                    BNE     DCRS25
3978                                    MOVB    (SP)+,@R3
3979                            FILS25: DEC     R2
3980                                    BEQ     ONES25
3981                                    MOVB    #' ,-(R3)
3982                                    BR      FILS25
3983                            FULS25: TST     (SP)+
3984                                    BNE     ERRS25
3985                                    CMP     #' ,(SP)+
3986                                    BNE     STSS25-4,
3987                            ONES25: MOV     (SP)+,R4
3988                                    MOV     (SP)+,4(SP)
3989                                    TST     (SP)+
3990                                    ROL     (SP)+
3991                                    RTS     PC
3992                            ERRS25: TST     (SP)+
3993                                    BNE     ERRS25
3994                                    TST     (SP)+
3995                                    MOV     8.(SP),R3
3996                            STSS25: MOVB    #'0,(R3)+
3997                                    DEC     6(SP)
3998                                    BGT     STSS25
3999                                    COM     6(SP)
4000                                    BR      ONES25
4001                            DECS25: .WORD   10000.,1000.,100.,10.,0
4002                            OCTS25: .WORD   100000,10000,1000,100,10,0
4003                                    .ENDC
4004                                    .TITLE  SINT02
4005                                    .IFDF   GNOS24
4006                                    .GLOBL  INT,IOINT,$RI,$POLSH
4007                                    R0=%0
4008                                    R4=%4
4009                                    R5=%5
4010                                    SP=%6
4011                            INT:
4012                            IOINT:  MOV     2(R5),R4
4013                                    MOV     2(R4),-(SP)
4014                                    MOV     @R4,-(SP)
```

455

```
                                        JSR     R4,$POLSH
                                        .WORD   $R1,UPLS26
                                UPLS26: MOV     (SP)+,R0
                                        RTS     R5
                                        .ENDC
                                        .TITLE  SIR04
                                        .IFDF   CND$27
                                        .GLOBL  SIR
                                        .IFNDF  $BAS
                                        .GLOBL  SID
                                        .ENDC
                    000000              R0=X0
                    000001              R1=X1
                    000002              R2=X2
                    000003              R3=X3
                    000004              R4=X4
                    000006              SP=X6
                    177304              MQ=177304
                    177312              NOR=177312
                    000000              F0=X0
                                        .IFDF   FPU
                                        .IFNDF  $BAS
                                SID:    SETD
                                        BR      IQI$27
                                                        .ENDC
                                SIR:    SETF
                                IQI$27: SETI
                                        LDCIF   (SP)+,F0
                                        STF     F0,-(SP)
                                        JMP     @(R4)+
                                        .ENDC
                                        .IFNDF  FPU
                                        .IFNDF  $BAS
                                SID:    MOV     @SP,-(SP)
                                        MOV     @SP,-(SP)
                                        CLR     2(SP)
                                        CLR     4(SP)
                                        .ENDC
002236' 005046                  SIR:    CLR     -(SP)
002240' 016601  000002                  MOV     2(SP),R1
002244' 003002                          BGT     POS$27
002246' 001424                          BEQ     ZER$27
002250' 005401                          NEG     R1
002252' 006146                  POS$27: ROL     -(SP)
                                        .IFNDF  EAE
002254' 012702  000220                  MOV     #220,R2
                                        .ENDC
                                        .IFDF   EAE
                                        MOV     #217,R2
```

456


```
                                        .ENDC
002260' 105066  000004                  CLRB    4(SP)
                                NOM$27:
                                        .IFNDF  EAE
002264' 006101                          ROL     R1
002266' 103402                          BCS     NOD$27
002270' 005302                          DEC     R2
002272' 000774                          BR      NOM$27
                                        .ENDC
                                        .IFDF   EAE
                                        MOV     #MQ,R3
                                        CLR     @R3
                                        MOV     R1,-(R3)
                                        MOV     #NOR,R0
                                        CLR     @R0
                                        SUB     (R0)+,R2
                                        MOV     #2,@R0
                                        MOV     @R3,R1
                                        .ENDC
002274' 110166  000005          NOD$27: MOVB    R1,5(SP)
002300' 105001                          CLRB    R1
002302' 150201                          BISB    R2,R1
002304' 000301                          SWAB    R1
002306' 006026                          ROR     (SP)+
002310' 006001                          ROR     R1
002312' 106066  000003                  RORB    3(SP)
002316' 010116                          MOV     R1,@SP
002320' 000134                  ZER$27: JMP     @(R4)+
                                        .ENDC
                                        .ENDC
                                        .TITLE  $MLD05
                                        .IFDF   CND$20
                                        .GLOBL  $MLD,$ERRA
                                        R0=X0
                                        R1=X1
                                        R2=X2
                                        R3=X3
                                        R4=X4
                                        R5=X5
                                        SP=X6
                                        PC=X7
                                        MQ=177304
                                        AS=0,
                                        B=16.,
                                        RESLT=12,
                                        SIGN=2
                                        F0=X0
                                        .IFDF   FPU
                                SML01:  .WORD   170011
                                        .WORD   172420
                                        .WORD   171020
                                        .WORD   174040
                                        JMP     @(R4)+
                                        .ENDC
                                        .IFNDF  FPU
```

457

```
                        SMLD1:  MOV     R4,-(SP)
                                MOV     R5,-(SP)
                                ASL     A+8.+4(SP)
                                ROL     -(SP)
                                CLR     -(SP)
                                MOVB    A+1(SP),@SP
                                BEQ     $ERR328
                                MOVB    A(SP),A+1(SP)
                                BEQ
                                ROR     A(SP)
                                MOVB    A+3(SP),A(SP)
                                SWAB    A+2(SP)
                                MOVB    A+5(SP),A+2(SP)
                                SWAB    A+4(SP)
                                MOVB    A+7(SP),A+4(SP)
                                SWAB    A+6(SP)
                                CLRB    A+6(SP)
                                ASL     B(SP)
                                ADC     $IGN(SP)
                                TSTB    B+1(SP)
                                BNE     NN$328
                        $ERR328:CMP     (SP)+,(SP)+
                        $EC$328:JMP     $EC2328
                        NN$328: CLR     R0
                                CLR     R1
                                .IFNDF  EAE&MULDIV
                                CLR     R2
                                CLR     R3
                                CLR     R5
                                ROR     B(SP)
                                MOV     #16.,-(SP)
                                MOV     B+6+2(SP),R4
                                BEQ     B4E328
                                JSR     PC,MT0328
                                MOV     #16.,@SP
                        B6E328: MOV     B+4+2(SP),R4
                                BNE     B4N328
                                TST     B+4+2(SP)
                                BEQ     B4E328
                        B4N328: JSR     PC,MT2328
                                JSR     PC,MLT328
                                MOV     #16.,@SP
                        B4E328: MOV     B+2+2(SP),R4
                                BNE     B2N328
                                TST     B+4+2(SP)
                                BNE     B2N328
                                TST     B+6+2(SP)
                                BEQ     B2E328
                        B2N328: JSR     PC,MLT328
                        B2E328: MOV     B+0+2(SP),R4
                                MOV     #7,@SP
                                JSR     PC,MLT328
                                JSR     PC,MT1328
                                TST     (SP)+
                                ADD     (SP)+,R4
```

458


```
                                .ENDC
                                .IFDF   EAE!MULDIV
                                CLR     R4
                                BISB    B+1(SP),R4
                                ADD     R4,@SP
                                MOVB    #1,B+1(SP)
                                ROR     B(SP)
                                SWAB    B(SP)
                                MOVB    B+3(SP),B(SP)
                                SWAB    B+2(SP)
                                MOVB    B+5(SP),B+2(SP)
                                SWAB    B+4(SP)
                                MOVB    B+7(SP),B+4(SP)
                                SWAB    B+6(SP)
                                CLRB    B+6(SP)
                                .ENDC
                                .IFDF   EAE
                                MOV     #MQ,R4
                                MOV     A(SP),-(SP)
                                MOV     B+6+2(SP),@R4
                                JSR     R5,EMUS28
                                MOV     (SP)+,R2
                                MOV     (SP)+,R3
                                MOV     A+2(SP),-(SP)
                                MOV     B+4+2(SP),@R4
                                JSR     R5,EMUS28
                                ADD     (SP)+,R2
                                ADC     R1
                                ADD     (SP)+,R3
                                ADC     R2
                                ADC     R1
                                MOV     A+4(SP),-(SP)
                                MOV     B+2+2(SP),@R4
                                JSR     R5,EMUS28
                                ADD     (SP)+,R2
                                ADC     R1
                                ADD     (SP)+,R3
                                ADC     R2
                                ADC     R1
                                MOV     A+6(SP),-(SP)
                                MOV     B+0+2(SP),@R4
                                JSR     R5,EMUS28
                                ADD     (SP)+,R2
                                ADC     R1
                                ADD     (SP)+,R3
                                ADC     R2
                                ADC     R1
                                MOV     R2,R3
                                MOV     R1,R2
                                CLR     R1
                                MOV     A(SP),-(SP)
                                MOV     B+4+2(SP),@R4
                                JSR     R5,EMUS28
                                ADD     (SP)+,R2
                                ADC     R1
```

459

```
                                        ADD     (SP)+,R3
                                        ADC     R2
                                        ADC     R1
                                        MOV     A+2(SP),-(SP)
                                        MOV     B+2+2(SP),@R4
                                        JSR     R5,EMUS28
                                        ADD     (SP)+,R2
                                        ADC     R1
                                        ADD     (SP)+,R3
                                        ADC     R2
                                        ADC     R1
                                        MOV     A+4(SP),-(SP)
                                        MOV     B+0+2(SP),@R4
                                        JSR     R5,EMUS28
                                        ADD     (SP)+,R2
                                        ADC     R1
                                        ADD     (SP)+,R3
                                        ADC     R2
                                        ADC     R1
                                        MOV     A(SP),-(SP)
                                        MOV     B+2+2(SP),@R4
                                        JSR     R5,EMUS28
                                        ADD     (SP)+,R1
                                        ADC     R0
                                        ADD     (SP)+,R2
                                        ADC     R1
                                        ADC     R0
                                        MOV     A+2(SP),-(SP)
                                        MOV     B+0+2(SP),@R4
                                        JSR     R5,EMUS28
                                        ADD     (SP)+,R1
                                        ADC     R0
                                        ADD     (SP)+,R2
                                        ADC     R1
                                        ADC     R0
                                        MOV     A(SP),-(SP)
                                        MOV     B+0+2(SP),@R4
                                        JSR     R5,EMUS28
                                        ADD     (SP)+,R0
                                        ADD     (SP)+,R1
                                        ADC     R0
                                        MOV     (SP)+,R4
                                .ENDC
                                .IFDF   MVLDIV
                                MOV     A(SP),-(SP)
                                MOV     B+0+2(SP),R4
                                JSR     PC,EMUS28
                                MOV     R4,R2
                                MOV     R5,R3
                                MOV     A+2(SP),-(SP)
                                MOV     B+4+2(SP),R4
                                JSR     PC,EMUS28
                                ADD     R4,R2
                                ADC     R1
                                ADD     R5,R3
```

460


```
                                        ADC     R2
                                        ADC     R1
                                        MOV     A+4(SP),-(SP)
                                        MOV     B+2+2(SP),R4
                                        JSR     PC,EMUS28
                                        ADD     R4,R2
                                        ADC     R1
                                        ADD     R5,R3
                                        ADC     R2
                                        ADC     R1
                                        MOV     A+6(SP),-(SP)
                                        MOV     B+0+2(SP),R4
                                        JSR     PC,EMUS28
                                        ADD     R4,R2
                                        ADC     R1
                                        ADD     R5,R3
                                        ADC     R2
                                        ADC     R1
                                        MOV     R2,R3
                                        MOV     R1,R2
                                        CLR     R1
                                        MOV     A(SP),-(SP)
                                        MOV     B+4+2(SP),R4
                                        JSR     PC,EMUS28
                                        ADD     R4,R2
                                        ADC     R1
                                        ADD     R5,R3
                                        ADC     R2
                                        ADC     R1
                                        MOV     A+2(SP),-(SP)
                                        MOV     B+2+2(SP),R4
                                        JSR     PC,EMUS28
                                        ADD     R4,R2
                                        ADC     R1
                                        ADD     R5,R3
                                        ADC     R2
                                        ADC     R1
                                        MOV     A+4(SP),-(SP)
                                        MOV     B+0+2(SP),R4
                                        JSR     PC,EMUS28
                                        ADD     R4,R2
                                        ADC     R1
                                        ADD     R5,R3
                                        ADC     R2
                                        ADC     R1
                                        MOV     A(SP),-(SP)
                                        MOV     B+2+2(SP),R4
                                        JSR     PC,EMUS28
                                        ADD     R4,R1
                                        ADC     R0
                                        ADD     R5,R2
                                        ADC     R1
                                        ADC     R0
                                        MOV     A+2(SP),-(SP)
                                        MOV     B+0+2(SP),R4
```

461

```
4345                                    JSR     PC,EMU$28
4346                                    ADD     R4,R1
4347                                    ADC     R0
4348                                    ADD     R5,R2
4349                                    ADC     R1
4350                                    ADC     R0
4351                                    MOV     A(SP),-(SP)
4352                                    MOV     B+0+2(SP),R4
4353                                    JSR     PC,EMU$28
4354                                    ADD     R4,R0
4355                                    ADD     R5,R1
4356                                    ADC     R0
4357                                    MOV     (SP)+,R4
4358                                    .ENDC
4359                                    ASL     R3
4360                                    ROL     R2
4361                                    ROL     R1
4362                                    ROL     R0
4363                                    BCS     NOM$28
4364                                    ASL     R3
4365                                    ROL     R2
4366                                    ROL     R1
4367                                    ROL     R0
4368                                    DEC     R4
4369                            NOM$28: SUB     #200,R4
4370                                    BLE     UND$28
4371                                    CMP     #377,R4
4372                                    BLT     OVR$28
4373                                    CLRB    R3
4374                                    BISB    R2,R3
4375                                    SWAB    R3
4376                                    CLRB    R2
4377                                    BISB    R1,R2
4378                                    SWAB    R2
4379                                    CLRB    R1
4380                                    BISB    R0,R1
4381                                    SWAB    R1
4382                                    CLRB    R0
4383                                    BISB    R4,R0
4384                                    SWAB    R0
4385                                    ROR     (SP)+
4386                                    ROR     R0
4387                                    ROR     R1
4388                                    ROR     R2
4389                                    ROR     R3
4390                                    ADC     R3
4391                                    ADC     R2
4392                                    ADC     R1
4393                                    ADC     R0
4394                                    BCS     OV1$28
4395                                    BVS     OV1$28
4396                            OUT$28: MOV     R0,RESULT(SP)
4397                                    MOV     R1,RESULT+2(SP)
4398                                    MOV     R2,RESULT+4(SP)
4399                                    MOV     R3,RESULT+6(SP)
```

462


```
4400                                    MOV     (SP)+,R5
4401                                    MOV     (SP)+,R4
4402                                    ADD     #8.,SP
4403                                    JMP     @(R4)+
4404                            OV1$28: TST     -(SP)
4405                            OVR$28: MOV     #77003,R0
4406                                    BR      ECL$28
4407                            UND$28: MOV     #13005,R0
4408                            ECL$28: TST     (SP)+
4409                                    JSR     R5,$ERRA
4410                            ZE2$28: CLR     R0
4411                                    CLR     R1
4412                                    CLR     R2
4413                                    CLR     R3
4414                                    BR      OUT$28
4415                                    .IFNDF  EAE&MULDIV
4416                            MLT$28: ASR     R4
4417                                    BCC     X0$28
4418                            MT1$28: ADD     A+6+4(SP),R3
4419                                    ADC     R2
4420                                    ADC     R1
4421                                    ADC     R0
4422                                    ADC     R5
4423                                    ADD     A+4+4(SP),R2
4424                                    ADC     R1
4425                                    ADC     R0
4426                                    ADC     R5
4427                                    ADD     A+2+4(SP),R1
4428                                    ADC     R0
4429                                    ADC     R5
4430                                    ADD     A+0+4(SP),R0
4431                                    ADC     R5
4432                            X0$28:  ASR     R5
4433                                    ROR     R0
4434                                    ROR     R1
4435                                    ROR     R2
4436                                    ROR     R3
4437                                    DEC     2(SP)
4438                                    BGT     MLT$28
4439                                    RTS     PC
4440                            MT2$28: DEC     2(SP)
4441                            MT0$28: ASR     R4
4442                                    BCC     X00$28
4443                                    ADD     A+2+4(SP),R1
4444                                    ADC     R0
4445                                    ADC     R5
4446                                    ADD     A+0+4(SP),R0
4447                                    ADC     R5
4448                            X00$28: ASR     R5
4449                                    ROR     R0
4450                                    ROR     R1
4451                                    ROR     R2
4452                                    ROR     R3
4453                                    DEC     2(SP)
4454                                    BGT     MT0$28
```

463

```
4455                                    RTS     PC
4456                                    .ENDC
4457                                    .IFDF   EAE
4458                            EMUS28: CLR     @SP
4459                                    TST     @R4
4460                                    BEQ     MZS28
4461                                    BGT     MPLS28
4462                                    TST     2(SP)
4463                                    BEQ     MZS28
4464                                    BGT     MNGS28
4465                                    ADD     (R4)+,@SP
4466                                    ADD     2(SP),@SP
4467                                    BR      EMLS28
4468                            MPLS28: TST     2(SP)
4469                                    BEQ     MZS28
4470                                    BGT     MLQS28
4471                                    ADD     (R4)+,@SP
4472                                    BR      EMLS28
4473                            MNGS28: ADD     2(SP),@SP
4474                            MLQS28: TST     (R4)+
4475                            EMLS28: MOV     2(SP),@R4
4476                                    MOV     -(R4),2(SP)
4477                                    ADD     -(R4),@SP
4478                                    TST     (R4)+
4479                                    JMP     @R5
4480                            MZS28:  CLR     2(SP)
4481                                    JMP     @R5
4482                                    .ENDC
4483                                    .IFDF   MULDIV
4484                            EMUS28: CLR     -(SP)
4485                                    TST     R4
4486                                    BEQ     MZS28
4487                                    BGT     MPLS28
4488                                    TST     4(SP)
4489                                    BEQ     MZS28
4490                                    BGT     MNLS28
4491                                    BR      MNGS28
4492                            MPLS28: TST     4(SP)
4493                                    BEQ     MZS28
4494                                    BGT     MLQS28
4495                                    ADD     R4,@SP
4496                                    BR      MLQS28
4497                            MNGS28: ADD     R4,@SP
4498                            MNLS28: ADD     4(SP),@SP
4499                            MLQS28: .WORD   ^B70466,4
4500                            MQNS28: ADD     (SP)+,R4
4501                                    MOV     (SP)+,@SP
4502                                    RTS     PC
4503                            MZS28:  CLR     R4
4504                                    CLR     R5
4505                                    BR      MQNS20
4506                                    .ENDC
4507                                    .ENDC
4508                                    .ENDC
4509                                    .TITLE  SML105
```

464


```
4510                                    .IFDF   CND529
4511                                    .GLOBL  SML1,$ERR
4512                                    R0=%0
4513                                    R1=%1
4514                                    R2=%2
4515                                    R3=%3
4516                                    R4=%4
4517                                    R5=%5
4518                                    SP=%6
4519                                    $RS29=177711
4520                                    MQ=177304
4521                                    .IFNDF  EAE&MULDIV
4522                            SML1:   CLR     R0
4523                                    MOV     (SP)+,R1
4524                                    BGT     P1S29
4525                                    BEQ     ZERS29
4526                                    INC     R0
4527                                    NEG     R1
4528                            P1S29:  MOV     @SP,R3
4529                                    BGT     P2S29
4530                                    BEQ     ZERS29
4531                                    INC     R0
4532                                    NEG     R3
4533                            P2S29:  MOV     R4,-(SP)
4534                                    MOV     #0,,R4
4535                                    CMP     R1,R3
4536                                    BGE     CLRS29
4537                                    MOV     R1,R2
4538                                    MOV     R3,R1
4539                                    MOV     R2,R3
4540                            CLRS29: CLR     R2
4541                            MULS29: ROR     R2
4542                                    ROR     R3
4543                                    BCC     CYCS29
4544                                    ADD     R1,R2
4545                            CYCS29: DEC     R4
4546                                    BGT     MULS29
4547                                    MOV     (SP)+,R4
4548                                    TSTB    R3
4549                                    BNE     OVRS29
4550                                    BISB    R2,R3
4551                                    SWAB    R3
4552                                    CLRB    R2
4553                                    SWAB    R2
4554                                    ASR     R2
4555                                    BNE     OVRS29
4556                                    ROR     R3
4557                                    NEG     R3
4558                                    BPL     OVRS29
4559                                    ROR     R0
4560                                    BCS     OUTS29
4561                                    NEG     R3
4562                                    BVS     OVRS29
4563                            OUTS29: MOV     R3,@SP
4564                                    JMP     @(R4)+
```

465

```
4565                                    NGMS29: NEG     R3
4566                                            BVC     OVRS29
4567                                            ROR     R0
4568                                            BCS     OUTS29
4569                                            BR      OVRS29
4570                                    ZERS29: CLR     @SP
4571                                            JMP     @(R4)+
4572                                            .ENDC
4573                                            .IFDF   EAE
4574                                    SMLI:   MOV     #MQ,R0
4575                                            MOV     (SP)+,(R0)+
4576                                            MOV     (SP)+,@R0
4577                                            MOV     -(R0),-(SP)
4578                                            BITB    #2,SR$29
4579                                            BEQ     OVRS29
4580                                            JMP     @(R4)+
4581                                            .ENDC
4582                                            .IFDF   MULDIV
4583                                    SMLI:   MOV     (SP)+,R0
4584                                            .WORD   170020
4585                                            MOV     R1,-(SP)
4586                                            BCS     OVRS29
4587                                            JMP     @(R4)+
4588                                            .ENDC
4589                                    OVRS29: CLR     (SP)
4590                                            JSR     R5,$ERR
4591                                            JMP     @(R4)+
4592                                            .BYTE   3
4593                                            .BYTE   19.
4594                                            .ENDC
4595                                            .TITLE  SMLR05
4596                                            .IFDF   CND$30
4597                                            .GLOBL  SMLR,$ERRA
4598                    000000                  R0=X0
4599                    000001                  R1=X1
4600                    000002                  R2=X2
4601                    000003                  R3=X3
4602                    000004                  R4=X4
4603                    000005                  R5=X5
4604                    000006                  SP=X6
4605                    000007                  PC=X7
4606                    177304                  MQ=177304
4607                    177311                  SR=177311
4608                    177314                  LSH=177314
4609                    000000                  FS=X0
4610                    000010                  A=8.
4611                    000014                  B=12.
4612                    000010                  RESLT=8.
4613                    000002                  SIGN=2
4614                                            .IFDF   FPU
4615                                    SMLR:   .WORD   170001
4616                                            .WORD   172420
4617                                            .WORD   171020
4618                                            .WORD   174040
4619                                            JMP     @(R4)+
```

465 A


```
4620                                            .ENDC
4621                                            .IFNDF  FPU
4622  002322' 010446                    SMLR:   MOV     R4,-(SP)
4623  002324' 010546                            MOV     R5,-(SP)
4624                                            .IFNDF  EAE&MULDIV
4625  002326' 016602  000004                    MOV     A+0-4(SP),R2
4626  002332' 006302                            ASL     R2
4627  002334' 006146                            ROL     -(SP)
4628  002336' 005046                            CLR     -(SP)
4629  002340' 000302                            SWAB    R2
4630  002342' 110216                            MOVB    R2,@SP
4631  002344' 001507                            BEQ     ZE1$30
4632  002346' 000261                            SEC
4633  002350' 006002                            ROR     R2
4634  002352' 105002                            CLRB    R2
4635  002354' 156602  000013                    BISB    A+3(SP),R2
4636  002360' 005003                            CLR     R3
4637  002362' 156603  000012                    BISB    A+2(SP),R3
4638  002366' 000303                            SWAB    R3
4639  002370' 006366  000014                    ASL     B(SP)
4640  002374' 005566  000002                    ADC     SIGN(SP)
4641  002400' 105766  000015                    TSTB    B+1(SP)
4642  002404' 001467                            BEQ     ZE1$30
4643  002406' 006066  000014                    ROR     B(SP)
4644  002412' 005000                            CLR     R0
4645  002414' 005001                            CLR     R1
4646  002416' 016604  000016                    MOV     B+2(SP),R4
4647  002422' 001406                            BEQ     B2E$30
4648  002424' 012705  000017            B2NS30: MOV     #17.,R5
4649  002430' 004767  000220                    JSR     PC,MTB$30
4650  002434' 004767  000160                    JSR     PC,MLT$30
4651  002440' 016604  000014            B2E$30: MOV     B(SP),R4
4652  002444' 012705  000007                    MOV     #7,R5
4653  002450' 004767  000144                    JSR     PC,MLT$30
4654  002454' 004767  000144                    JSR     PC,MTI$30
4655  002460' 062604                            ADD     (SP)+,R4
4656                                            .ENDC
4657                                            .IFDF   EAE
4658                                            MOV     #MQ,R4
4659                                            MOV     #100000,R5
4660                                            MOV     B+2-4(SP),@R4
4661                                            MOV     B+0-4(SP),-(R4)
4662                                            BEQ     ZER$30
4663                                            INC     @@LSH
4664                                            RORB    @@SR
4665                                            ROL     -(SP)
4666                                            MOV     (R4)+,-(SP)
4667                                            CLRB    @SP
4668                                            SWAB    @SP
4669                                            MOV     #7,@@LSH
4670                                            MOV     @R4,-(SP)
4671                                            BISB    R5,-(R4)
4672                                            MOV     (R4)+,-(SP)
4673                                            MOV     A+2-4(SP),@R4
4674                                            MOV     A+0-4(SP),-(R4)
```

466

```
                                        BEQ     $E2$30
                                        INC     @#LSH
                                        RORB    @#SR
                                        ADC     6(SP)
                                        MOV     @R4,R3
                                        CLRB    R3
                                        SWAB    R3
                                        ADD     R3,4(SP)
                                        MOV     #7,@#LSH
                                        MOV     (R4)+,R2
                                        BIS     R5,R2
                                        CLR     R0
                                        CLR     R1
                                        MOV     (R4)+,R3
                                        BNE     A2NS30
                                        TST     -(R4)
                                        BR      A2ES30
                                A2NS30: MOV     @SP,@R4
                                        CMP     -(R4),-(R4)
                                        ADD     R3,@R4
                                        TST     R3
                                        BPL     A2PS30
                                        ADD     @SP,@R4
                                A2PS30: MOV     (R4)+,R1
                                A2ES30: MOV     2(SP),(R4)+
                                        BNE     B2NS30
                                        TST     -(R4)
                                        BR      B2ES30
                                B2NS30: MOV     R2,@R4
                                        CMP     -(R4),-(R4)
                                        ADD     2(SP),@R4
                                        TST     2(SP)
                                        BPL     B2PS30
                                        ADD     R2,@R4
                                B2PS30: ADD     (R4)+,R1
                                        ADC     R0
                                B2ES30: MOV     R2,(R4)+
                                        ADD     R2,R0
                                        MOV     @SP,@R4
                                        ADD     (SP)+,R0
                                        ADD     -(R4),R1
                                        ADC     R0
                                        ADD     -(R4),R0
                                        TST     (SP)+
                                        MOV     (SP)+,R4
                                        .ENDC
                                        .IFDF   MUL,DIV
                                        MOV     B^2=4(SP),R5
                                        MOV     B^0=4(SP),R4
                                        BEQ     $ERR30
                                        .WORD   073427,1
                                        ROL     -(SP)
                                        MOV     R4,-(SP)
                                        CLRB    @SP
                                        SWAB    @SP
```

467


```
                                        .WORD   073427,7
                                        MOV     R5,-(SP)
                                        BIS     #100000,R4
                                        MOV     R4,-(SP)
                                        MOV     A^2=4(SP),R3
                                        MOV     A^0=4(SP),R2
                                        BEQ     $E2$30
                                        .WORD   073227,1
                                        ADC     4(SP)
                                        MOV     R2,R0
                                        CLRB    R0
                                        SWAB    R0
                                        ADD     R0,4(SP)
                                        .WORD   073227,7
                                        BIS     #100000,R2
                                        CLR     R0
                                        CLR     R1
                                        TST     R3
                                        BEQ     A2ES30
                                        .WORD   070403
                                        ADD     R3,R4
                                        TST     R3
                                        BPL     A2PS30
                                        ADD     @SP,R4
                                A2PS30: MOV     R4,R1
                                A2ES30: MOV     2(SP),R4
                                        BEQ     B2ES30
                                        .WORD   070402
                                        ADD     2(SP),R4
                                        TST     2(SP)
                                        BPL     B2PS30
                                        ADD     R2,R4
                                B2PS30: ADD     R4,R1
                                        ADC     R0
                                B2ES30: MOV     R2,R4
                                        ADD     R2,R0
                                        .WORD   070410
                                        ADD     (SP)+,R0
                                        ADD     R5,R1
                                        ADC     R0
                                        ADD     R4,R0
                                        TST     (SP)+
                                        MOV     (SP)+,R4
                                        .ENDC
002462' 006101                          ROL     R1
002464' 006100                          ROL     R0
002466' 103403                          BCS     NOMS30
002470' 006101                          ROL     R1
002472' 006100                          ROL     R0
002474' 005304                          DEC     R4
002476' 162704  000200          NOMS30: SUB     #200,R4
002502' 003436                          BLE     UNDS30
002504' 022704  000377                  CMP     #377,R4
002510' 002427                          BLT     OVRS30
002512' 105001                          CLRB    R1
```

468

```
 4785  002514' 150001                  BISB    R0,R1
 4786  002516' 000301                  SWAB    R1
 4787  002520' 105000                  CLRB    R0
 4788  002522' 150400                  BISB    R4,R0
 4789  002524' 000300                  SWAB    R0
 4790  002526' 006026                  ROR     (SP)+
 4791  002530' 006000                  ROR     R0
 4792  002532' 006001                  ROR     R1
 4793  002534' 005501                  ADC     R1
 4794  002536' 005500                  ADC     R0
 4795  002540' 103414                  BCS     OV1$30
 4796  002542' 102413                  BVS     OV1$30
 4797  002544' 016666  000010  OUT$30| MOV     R0,RE$LT(SP)
 4798  002550' 016166  000012          MOV     R1,RE$LT+2(SP)
 4799  002554' 012605                  MOV     (SP)+,R5
 4800  002556' 012604                  MOV     (SP)+,R4
 4801  002560' 022626                  CMP     (SP)+,(SP)+
 4802  002562' 000134                  JMP     @(R4)+
 4803                                  .IFDF   EAE!MUL!DIV
 4804                          EE2$30| CMP     (SP)+,(SP)+
 4805                                  .ENDC
 4806  002564' 022626                  CMP     (SP)+,(SP)+
 4807  002566' 000411                  BR      EER$30
 4808  002570' 005726          OVR$30| TST     (SP)+
 4809  002572' 012700  000003  OVI$30| MOV     #3,R0
 4810  002576' 000403                  BR      ECL$30
 4811  002600' 012700  003485  UNO$30| MOV     #3485,R0
 4812  002604' 005726                  TST     (SP)+
 4813  002606' 004567  001436  ECL$30| JSR     R5,$ERRA
 4814  002612' 005000          EER$30| CLR     R0
 4815  002614' 005001                  CLR     R1
 4816  002616' 000752                  BR      OUT$30
 4817                                  .IFNDF  EAE&MUL!DIV
 4818  002620' 006204          MLT$30| ASR     R4
 4819  002622' 103004                  BCC     X8$30
 4820  002624' 060301          MTI$30| ADD     R3,R1
 4821  002626' 005500                  ADC     R0
 4822  002630' 103406                  BCS     COV$30
 4823  002632' 060200                  ADD     R2,R0
 4824  002634' 006000          X8$30|  ROR     R0
 4825  002636' 006001                  ROR     R1
 4826  002640' 005305                  DEC     R5
 4827  002642' 003366                  BGT     MLT$30
 4828  002644' 000207                  RTS     PC
 4829  002646' 060200          COV$30| ADD     R2,R0
 4830  002650' 000261                  SEC
 4831  002652' 000770                  BR      X8$30
 4832  002654' 006204          MT8$30| ASR     R4
 4833  002656' 103001                  BCC     X88$30
 4834  002660' 060200                  ADD     R2,R0
 4835  002662' 006000          X88$30| ROR     R0
 4836  002664' 006001                  ROR     R1
 4837  002666' 005305                  DEC     R5
 4838  002670' 003371                  BGT     MT8$30
 4839  002672' 000207                  RTS     PC
```

469


```
 4840                                  .ENDC
 4841                                  .ENDC
 4842                                  .ENDC
 4843                                  .EOT
 4844
```

470

```
                                        .       BASIC SOURCE FILE #14
                                .TITLE  $NEGS2
                                .IFDF   CND$31
                                .GLOBL  $NGI,$NGR,$NGO,$ERR
                                R4=%4
                                R5=%5
                                SP=%6
                        $NGI:   NEG     @$P
                                BVS     OVR$31
                                JMP     @(R4)+
                        $NGR:
                        $NGO:   TST     @$P
                                BEQ     ZER$31
                                ADD     #100000,@$P
                        ZER$31: JMP     @(R4)+
                        OVR$31: JSR     R5,$ERR
                                JMP     @(R4)+
                                .BYTE   3
                                .BYTE   11.
                                .ENDC
                                .TITLE  $PHR87
                                .IFDF   CND$32

                        R0      =       %0
                        R1      =       %1
                        R2      =       %2
                        R3      =       %3
                        R4      =       %4
                        R5      =       %5
                        SP      =       %6
                        PC      =       %7
                                .GLOBL  $PSHR5,$PSHR4,$PSHR3,$PSHR2,$PSHR1
                        $PSHR5:
                        $PSHR4: MOV     R3,-(SP)
                                MOV     R2,-(SP)
                        $PSHR3: MOV     R1,-(SP)
                        $PSHR2:
                        $PSHR1: MOV     R0,-(SP)
                                JMP     @(R4)+
                                .ENDC
                                .TITLE  $PPR84
                                .IFDF   CND$33
000000                          R0=%0
000001                          R1=%1
000002                          R2=%2
000003                          R3=%3
000004                          R4=%4
000005                          R5=%5
000006                          SP=%6
000007                          PC=%7
                                .GLOBL  $POPR5,$POPR4,$POPR3
                        $POPR5:
002674' 012600          $POPR4: MOV     (SP)+,R0
002676' 012601                  MOV     (SP)+,R1
002700' 012602                  MOV     (SP)+,R2
```

471


```
002702' 012603                  MOV     (SP)+,R3
002704' 000134                  JMP     @(R4)+
002706' 012600          $POPR3: MOV     (SP)+,R0
002710' 012601                  MOV     (SP)+,R1
002712' 000134                  JMP     @(R4)+
                                .ENDC
                                .TITLE  $R0S2
                                .IFDF   CND$34
                                .GLOBL  $R0
                                R4=%4
                                SP=%6
                                F0=%0
                                F1=%1
                                .IFDF   FPU
                        $R0I:   .WORD   170011
                                .WORD   177420
                                .WORD   174400
                                JMP     @(R4)+
                                .ENDC
                                .IFNDF  FPU
                        $R0I:   MOV     2(SP),-(SP)
                                MOV     2(SP),-(SP)
                                CLR     4(SP)
                                CLR     6(SP)
                                JMP     @(R4)+
                                .ENDC
                                .ENDC
                                .TITLE  $RI84
                                .IFDF   CND$35
                                .GLOBL  $RI,$ERR

                                .IFNDF  $0AS
                                .GLOBL  $0I
                                .ENDC

000000                          R0=%0
000001                          R1=%1
000002                          R2=%2
000003                          R3=%3
000004                          R4=%4
000005                          R5=%5
000006                          SP=%6
177304                          MQ=177304
177314                          LSH=177314
000000                          F0=%0
                                .IFDF   FPU

                                .IFNDF  $0AS
                        $0I:    SETD
                                BR      RI0$35
                                .ENDC

                        $RI:    SETF
                        RI0$35: SETI
                                LDD     (SP)+,F0
```

472

```
                                        STCDI   F0,-(SP)
                                        JMP     @(R4)+
                                        .ENDC
                                        .IFNDF  FPU

                                        .IFNDF  $BA$
                                SDI:    MOV     (SP)+,2(SP)
                                        MOV     (SP)+,2(SP)
                                        .ENDC

002714' 005002                  SRI:    CLR     R2
002716' 005202                          INC     R2
002720' 012601                          MOV     (SP)+,R1
002722' 006116                          ROL     @SP
002724' 006101                          ROL     R1
002726' 006146                          ROL     -(SP)
002730' 110103                          MOVB    R1,R3
002732' 105001                          CLRB    R1
002734' 000301                          SWAB    R1
002736' 162701  000201                  SUB     #201,R1
002742' 002433                          BLT     $ERR$3$
002744' 001413                          BEQ     DNE$3$
002746' 022701  000017                  CMP     #15.,R1
002752' 002422                          BLT     OVR$3$
002754' 000303                          SWAB    R3
002756' 105003                          CLRB    R3
002760' 156003  000003                  BISB    3(SP),R3
                                SF1$3$:
                                        .IFNDF  EAE&MULDIV
006103                                  ROL     R3
006102                                  ROL     R2
005301                          DEC$3$: DEC     R1
003374                                  BGT     SF1$3$
                                        .ENDC
                                        .IFDF   EAE
                                        MOV     #MQ,R0
                                        MOV     R3,@R0
                                        MOV     R2,-(R0)
                                        MOV     R1,@0LSH
                                        MOV     @R0,R2
                                        .ENDC
                                        .IFDF   MULDIV
                                        .WORD   073201
                                        .ENDC
005402                          DNE$3$: NEG     R2
102406                                  BVS     NGM$3$
003007                                  BGT     OVR$3$
006026                          SGN$3$: ROR     (SP)+
103401                                  BCS     OUT$3$
005402                                  NEG     R2
010216                          OUT$3$: MOV     R2,@SP
000134                                  JMP     @(R4)+
006026                          NGM$3$: ROR     (SP)+
103774                                  BCS     OUT$3$
005726                          OVR$3$: TST     -(SP)
```

473


```
003022' 004567  001212                  JSR     R5,$ERR
003026' 000401                          BR      $ER$3$
003030'    003                          .BYTE   3
003031'    026                          .BYTE   22.
003032' 005002                  $ER$3$: CLR     R2
003034' 000762                          BR      SGN$3$
                                        .ENDC
                                        .ENDC
                                        .TITLE  $$GL02
                                        .IFDF   CND$3@
                                        .GLOBL  SNGL,$ERR
                                        R0=%0
                                        R1=%1
                                        R4=%4
                                        R5=%5
                                SNGL:   MOV     2(R5),R4
                                        MOV     (R4)+,R0
                                        MOV     (R4)+,R1
                                        MOV     @R4,R4
                                        ROL     R4
                                        ADC     R1
                                        ADC     R0
                                        BCS     OVR$3@
                                        BVS     OVR$3@
                                        RTS     R5
                                OVR$36: JSR     R5,$ERR
                                        RTS     R5
                                        .BYTE   4
                                        .BYTE   12.
                                        .ENDC
                                        .TITLE  $$IN04
                                        .IFDF   CND$37
                                        .GLOBL  SIN,COS
                                        .IFNDF  FPU
                                        .GLOBL  $ADR,$MLR,$SBR,$DVR,$INTR,$POL$H
                                        .ENDC
000000                                  R0=%0
000001                                  R1=%1
000002                                  R2=%2
000003                                  R3=%3
000004                                  R4=%4
000005                                  R5=%5
000006                                  SP=%6
000007                                  PC=%7
000000                                  F0=%0
000001                                  F1=%1
000002                                  F2=%2
000003                                  F3=%3
                                        .IFNDF  FPU
003036' 016504  000002          COS:    MOV     2(R5),R4
003042' 005046                          CLR     -(SP)
003044' 016446  000002                  MOV     2(R4),-(SP)
003050' 011446                          MOV     @R4,-(SP)
003052' 012746  007733                  MOV     #007733,-(SP)
003056' 012746  040311                  MOV     #040311,-(SP)
```

474

```
$$IN04  MACX11   V021   23-MAR-73   11:34   PAGE 7-4

 5065  003062' 004467  001810              JSR     R4,$POLSH
 5066  003066' 000006' 003106'            ,WORD   $ADR,$NCS37
 5067  003072' 016504  000002      SINI:   MOV     2(R5),R4
 5068  003076' 005046                      CLR     -(SP)
 5069  003100' 016446  000002              MOV     2(R4),-(SP)
 5070  003104' 011446                      MOV     @R4,-(SP)
 5071  003106' 006316              SNCS37: ASL     @SP
 5072  003110' 006066  000004              ROR     4(SP)
 5073  003114' 006016                      ROR     @SP
 5074  003116' 012746  007733              MOV     #007733,-(SP)
 5075  003122' 012746  040711              MOV     #040711,-(SP)
 5076  003126' 004467  000744              JSR     R4,$POLSH
 5077  003132' 001234'                     ,WORD   $OVR
 5078  003134' 003238'                     ,WORD   DUP$37
 5079  003136' 001134'                     ,WORD   $INTR
 5080  003140' 000002'                     ,WORD   $$BR
 5081  003142' 003242'                     ,WORD   X4$37
 5082  003144' 003238'                     ,WORD   DUP$37
 5083  003146' 001134'                     ,WORD   $INTR
 5084  003150' 003254'                     ,WORD   QUD$37
 5085  003152' 000002'                     ,WORD   $$BR
 5086  003154' 003262'                     ,WORD   QS$T37
 5087  003156' 003238'             QSE$37: ,WORD   DUP$37
 5088  003160' 003238'                     ,WORD   DUP$37
 5089  003162' 002322'                     ,WORD   $MLR
 5090  003164' 003332'                     ,WORD   PLY$37
 5091  003166' 002322'                     ,WORD   $MLR
 5092  003170' 000006'                     ,WORD   $ADR
 5093  003172' 002322'                     ,WORD   $MLR
 5094  003174' 000006'                     ,WORD   $ADR
 5095  003176' 002322'                     ,WORD   $MLR
 5096  003200' 000006'                     ,WORD   $ADR
 5097  003202' 002322'                     ,WORD   $MLR
 5098  003204' 000006'                     ,WORD   $ADR
 5099  003206' 002322'                     ,WORD   $MLR
 5100
 5101  003210' 003212'                     ,WORD   RTN$37
 5102  003212' 012600              RTN$37: MOV     (SP)+,R0
 5103  003214' 012601                      MOV     (SP)+,R1
 5104  003216' 005726                      TST     (SP)+
 5105  003220' 002002                      BGE     RT1$37
 5106  003222' 062706  100000              ADD     #100000,R0
 5107  003226' 000205              RT1$37: RTS     R5
 5108  003230' 016646  000002      DUP$37: MOV     2(SP),-(SP)
 5109  003234' 016646  000002              MOV     2(SP),-(SP)
 5110  003240' 000134                      JMP     @(R4)+
 5111  003242' 005716              X4$37:  TST     @SP
 5112  003244' 001762                      BEQ     RTN$37
 5113  003246' 105266  000001              INCB    1(SP)
 5114  003252' 000134                      JMP     @(R4)+
 5115  003254' 051666  000010      QUD$37: BIS     @SP,8.(SP)
 5116  003260' 000134                      JMP     @(R4)+
 5117  003262' 105766  000004      QS?$37: TSTB    4(SP)
 5118  003266' 001413                      BEQ     Q13$37
 5119  003270' 062716  100000              ADD     #100000,@SP
```

475

```
$$IN04  MACX11   V021   23-MAR-73   11:34   PAGE 7-5

 5120  003274' 005046                      CLR     -(SP)
 5121  003276' 012746  040200              MOV     #40200,-(SP)
 5122  003302' 004467  000570              JSR     R4,$POLSH
 5123  003306' 000006' 003312'            ,WORD   $ADR,QSRS37
 5124  003312' 012704  003196'      QSRS37: MOV     #QSES37,R4
 5125  003316' 106266  000005      Q13S37: ASRB    5(SP)
 5126  003322' 103002                      BCC     QUT$37
 5127  003324' 062716  100000              ADD     #100000,@SP
 5128  003330' 000134              QUT$37: JMP     @(R4)+
 5129  003332' 012600              PLY$37: MOV     (SP)+,R0
 5130  003334' 012601                      MOV     (SP)+,R1
 5131  003336' 012702  003412'             MOV     #CON$37+4,R2
 5132  003342' 012703  000005              MOV     #5,R3
 5133  003346' 000402                      BR      PY1$37
 5134  003350' 010146              PY2$37: MOV     R1,-(SP)
 5135  003352' 010046                      MOV     R0,-(SP)
 5136  003354' 014246              PY1$37: MOV     -(R2),-(SP)
 5137  003356' 014246                      MOV     -(R2),-(SP)
 5138  003360' 005303                      DEC     R3
 5139  003362' 003372                      BGT     PY2$37
 5140  003364' 000134                      JMP     @(R4)+
 5141                                      .ENDC
 5142                                      .IFDF   FPU
 5143                              COS:    SETD
 5144                                      LDCFD   @2(R5),F0
 5145                                      ADDD    P12$37,F0
 5146                                      BR      SNCS37
 5147                              SIN:    SETD
 5148                                      LDCFD   @2(R5),F0
 5149                              SNCS37: SETI
 5150                                      MOV     #FCOS37,R0
 5151                                      CLR     R4
 5152                                      CFCC
 5153                                      BGE     POS$37
 5154                                      INC     R4
 5155                                      ABSD    F0
 5156                              POS$37: DIVD    (R0)+,F0
 5157                                      MODD    #0.25,F0
 5158                                      SETF
 5159                                      LDEDF   F0,F0
 5160                                      CFCC
 5161                                      BEQ     RTN$37
 5162                                      MODF    #4.0,F0
 5163                                      STCFI   F1,R1
 5164                                      ROR     R1
 5165                                      BCC     Q13$37
 5166                                      NEGF    F0
 5167                                      ADDF    #1.0,F0
 5168                              Q13S37: ROR     R1
 5169                                      BCC     Q12$37
 5170                                      NEGF    F0
 5171                              Q12S37: LDF     F0,F2
 5172                                      MULF    F2,F2
 5173                                      MOV     #4,R1
 5174                                      LDF     (R0)+,F1
```

476

```
                                XPO$37: MULF    F2,F1
                                        DEC     R1
                                        ADDF    (R0)+,F1
                                        BGT     XPO$37
                                        MULF    F1,F0
                                        TST     R4
                                        BEQ     RTN$37
                                        NEGF    F0
                                RTN$37: STF     F0,-(SP)
                                        MOV     (SP)+,R0
                                        MOV     (SP)+,R1
                                        RTS     R5
                                FCO$37:
                                PI2337: .WORD   040311,007732
                                        .WORD   121041,064302
                                        .ENDC
000366' 035036  153072                  .WORD   035036,153672
000372' 136231  023143                  .WORD   136231,023143
000376' 037243  032130                  .WORD   037243,032130
000402' 140845  056741                  .WORD   140845,056741
000406' 040311  007733          CON$37: .WORD   040311,007733
                                        .ENDC
                                        .TITLE  $TINH02
                                        .IFDF   CND$38
                                        .GLOBL  TANH,EXP,$ADR,$SBR,$MLR,$DVR,$FCALL
                                        .GLOBL  $POL$H,$PSHR3
                                        R0=%0
                                        R1=%1
                                        R4=%4
                                        R5=%5
                                        SP=%6
                                        PC=%7
                                TANH:   MOV     R5,-(SP)
                                        MOV     2(R5),R5
                                        MOV     @R5,R0
                                        BEQ     ZER$38
                                        ASL     R0
                                        CLRB    R0
                                        SWAB    R0
                                        CMP     R0,#205
                                        BLT     STE$38
                                        MOV     #40200,R0
                                        CLR     R1
                                        TST     @R5
                                        BGE     OUT$38
                                        ADD     #100000,R0
                                        BR      OUT$38
                                STE$38: CMP     R0,#177
                                        BGT     TAN$38
                                        CMP     R0,#104
                                        BGE     SML$38
                                        MOV     2(R5),R1
                                        MOV     @R5,R0
                                        BR      OUT$38
                                TAN$38: MOV     2(R5),-(SP)
```



```
                                        MOV     @R5,-(SP)
                                        ADD     #200,@SP
                                        MOV     SP,R5
                                        MOV     #EXP,R4
                                        JSR     PC,$FCALL
                                        MOV     R1,-(SP)
                                        MOV     R0,-(SP)
                                        CLR     -(SP)
                                        MOV     #40200,-(SP)
                                        MOV     R1,-(SP)
                                        MOV     R0,-(SP)
                                        CLR     -(SP)
                                        MOV     #40200,-(SP)
                                        JSR     R4,$POL$H
                                        .WORD   $$BR,UP$38,$ADR,$DVR,UPL$38
                                UPL$38: MOV     (SP)+,R0
                                        MOV     (SP)+,R1
                                OUT$38: MOV     (SP)+,R5
                                        RTS     R5
                                SML$38: MOV     2(R5),R1
                                        MOV     @R5,R0
                                        JSR     R4,$POL$H
                                        .WORD   $PSHR3,$PSHR3,$PSHR3,$MLR,XSQ$38
                                XSQ$38: MOV     2(SP),-(SP)
                                        MOV     2(SP),-(SP)
                                        JSR     R4,$POL$H
                                        .WORD   P35$38,$ADR,ONE$38
                                        .WORD   $PSHR3,P45$38,$PSHR3,$DVR,$ADR,$ADR,$DVR
                                        .WORD   $$BR,$MLR,UPL$38
                                ONE$38: MOV     4(SP),R0
                                        MOV     6(SP),R1
                                        CLR     6(SP)
                                        MOV     #40200,4(SP)
                                        JMP     @(R4)+
                                P45$38: MOV     #136237,-(SP)
                                        MOV     #41466,-(SP)
                                P15$38: MOV     #165707,-(SP)
                                        MOV     #61722,-(SP)
                                        JMP     @(R4)+
                                P35$38: MOV     #110407,-(SP)
                                        MOV     #41416,-(SP)
                                        JMP     @(R4)+
                                ZER$38: CLR     R0
                                        CLR     R1
                                        BR      OUT$38
                                UP$38:  MOV     (SP)+,10,(SP)
                                        MOV     (SP)+,10,(SP)
                                        JMP     @(R4)+
                                        .ENDC
                                        .TITLE  $ATN03
                                        .IFDF   CND$37
                                        .GLOBL  ATAN
                                        .IFNDF  $BA8
```

```
                                        .GLOBL  ATAN2
                                        .ENDC
                                        .IFNDF  FPU
                                        .GLOBL  SADR,$$BR,$MLR,$DVR,$POLSH,$POPRS
                                        .ENDC
                        000000          R0=%0
                        000001          R1=%1
                        000002          R2=%2
                        000003          R3=%3
                        000004          R4=%4
                        000005          R5=%5
                        000006          SP=%6
                        000000          F0=%0
                        000001          F1=%1
                        000002          F2=%2
                        000003          F3=%3
                        000004          F4=%4
                        000005          F5=%5
                                        .IFNDF  FPU
                                        .IFNDF  $$AS
                                ATAN2:  CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        CLR     -(SP)
                                        MOV     2(R5),R4
                                        MOV     2(R4),-(SP)
                                        MOV     R4,-(SP)
                                        MOV     @SP,R0
                                        MOV     4(R5),R4
                                        MOV     2(R4),-(SP)
                                        MOV     R4,-(SP)
                                        MOV     @SP,R1
                                        BEQ     INF$39
                                        ASL     R0
                                        CLRB    R0
                                        SWAB    R0
                                        ASL     R1
                                        CLRB    R1
                                        SWAB    R1
                                        SUB     R1,R0
                                        CMP     #20,,R0
                                        BLT     INF$39
                                DIV$39: JSR     R4,$POLSH
                                        .WORD   $DVR,UPL$39
                                UPL$39: TST     @4(R5)
                                        BGE     ATE$39
                                        MOV     #040311,6.(SP)
                                        MOV     #007733,10.(SP)
                                        TST     @2(R5)
                                        BGE     ATE$39
                                        ADD     #100000,8.(SP)
                                ATE$39: TST     @SP
```

479


```
                                        BR      AT1$39
                                INF$39: ADD     #10.,SP
                                        MOV     #040311,R0
                                        MOV     #007733,R1
                                        TST     @2(R5)
                                        BGE     INR$39
                                        ADD     #100000,R0
                                INR$39: RTS     R5
                                        .ENDC

003412' 005046                  ATAN1:  CLR     -(SP)
003414' 005046                          CLR     -(SP)
003416' 005046                          CLR     -(SP)
003420' 005046                          CLR     -(SP)
003422' 005046                          CLR     -(SP)
003424' 016504 000002                   MOV     2(R5),R4
003430' 016446 000002                   MOV     2(R4),-(SP)
003434' 011446                          MOV     @R4,-(SP)
003436' 002004                  AT1$39: BGE     PLU$39
003440' 062716 100000                   ADD     #100000,@SP
003444' 005266 000014                   INC     12.(SP)
003450' 021627 040200           PLU$39: CMP     @SP,#40200
003454' 103431                          BLO     LE1$39
003456' 003003                          BGT     GT1$39
003460' 005766 000002                   TST     2(SP)
003464' 001425                          BEQ     LE1$39
003466' 012766 140311 000004    GT1$39: MOV     #140311,4(SP)
003474' 012766 007733 000006            MOV     #007733,6(SP)
003502' 005366 000014                   DEC     12.(SP)
003506' 016646 000002                   MOV     2(SP),-(SP)
003512' 016646 000002                   MOV     2(SP),-(SP)
003516' 012766 040200 000004            MOV     #40200,4(SP)
003524' 005066 000006                   CLR     6(SP)
003530' 004467 000342                   JSR     R4,$POLSH
003534' 001234' 003540'                 .WORD   $DVR,LE1$39
003540' 016646 000002           LE1$39: MOV     2(SP),-(SP)
003544' 016646 000002                   MOV     2(SP),-(SP)
003550' 005066 000004                   CLR     4(SP)
003554' 005066 000006                   CLR     6(SP)
003560' 021627 037611                   CMP     @SP,#037611
003564' 103445                          BLO     L1$39
003566' 101004                          BHI     TN$39
003570' 026627 000002 030243            CMP     2(SP),#30243
003576' 101440                          BLO$    L1$39
003600' 012766 040000 000004    TN$39:  MOV     #40000,4(SP)
003606' 012766 005222 000006            MOV     #5222,6(SP)
003614' 011600                          MOV     @SP,R0
003616' 016601 000002                   MOV     2(SP),R1
003622' 012746 131727                   MOV     #131727,-(SP)
003626' 012746 140335                   MOV     #140335,-(SP)
003632' 010146                          MOV     R1,-(SP)
003634' 010046                          MOV     R0,-(SP)
003636' 005046                          CLR     -(SP)
003640' 012746 040200                   MOV     #40200,-(SP)
003644' 012746 131727                   MOV     #131727,-(SP)
```

480

```
 5395  003650' 012746 040335                MOV     #$40335,-(SP)
 5396  003654' 010146                       MOV     R1,-(SP)
 5397  003656' 010046                       MOV     R0,-(SP)
 5398  003660' 004467 000212                JSR     R4,$POLSH
 5399  003664' 002322' 000002' 003770'      .WORD   $MLR,$$BR,UP$39,$$BR,$DVR,L15$39
       003672' 000002' 001234' 003700'
 5400  003700' 011600              L15$39:  MOV     @SP,R0
 5401  003702' 016601 000002                MOV     2(SP),R1
 5402  003706' 010146                       MOV     R1,-(SP)
 5403  003710' 010046                       MOV     R0,-(SP)
 5404  003712' 010146                       MOV     R1,-(SP)
 5405  003714' 010046                       MOV     R0,-(SP)
 5406  003716' 004467 000154                JSR     R4,$POLSH
 5407  003722' 002322'                      .WORD   $MLR
 5408  003724' 004002'                      .WORD   PLY$39
 5409  003726' 002322' 000006' 002322'      .WORD   $MLR,$ADR,$MLR,$ADR,$MLR,$ADR
       003734' 000006' 002322' 000006'
 5410  003742' 002322' 000006' 002322'      .WORD   $MLR,$ADR,$MLR,$ADR
       003750' 000006'
 5411  003752' 000006'                      .WORD   $ADR
 5412  003754' 004036'                      .WORD   $GN$39
 5413  003756' 000006'                      .WORD   $ADR
 5414  003760' 002706' 003764'              .WORD   $POPR3,EX[$39
 5415  003764' 005726              EX[$39:  TST     (SP)+
 5416  003766' 000205                       RTS     R5
 5417  003770' 012666 000012       UP$39:   MOV     (SP)+,12(SP)
 5418  003774' 012666 000012                MOV     (SP)+,12(SP)
 5419  004000' 000134                       JMP     @(R4)+
 5420  004002' 012600              PLY$39:  MOV     (SP)+,R0
 5421  004004' 012601                       MOV     (SP)+,R1
 5422  004006' 012702 004070'               MOV     #CON$39+4,R2
 5423  004012' 012703 000005                MOV     #5,R3
 5424  004016' 000402                       BR      PY1$39
 5425  004020' 010146              PY2$39:  MOV     R1,-(SP)
 5426  004022' 010046                       MOV     R0,-(SP)
 5427  004024' 014246              PY[$39:  MOV     -(R2),-(SP)
 5428  004026' 014246                       MOV     -(R2),-(SP)
 5429  004030' 005303                       DEC     R3
 5430  004032' 003372                       BGT     PY2$39
 5431  004034' 000134                       JMP     @(R4)+
 5432  004036' 005766 000010       SGN$39:  TST     8.(SP)
 5433  004042' 001402                       BEQ     SG1$39
 5434  004044' 062716 100000                ADD     #100000,@SP
 5435  004050' 000134              SG[$39:  JMP     @(R4)+
 5436                                       .ENDC
 5437                                       .IFDF   FPU
 5438                              ATAN2:   SETF
 5439                                       MOV     2(R5),R3
 5440                                       MOV     4(R5),R4
 5441                                       MOV     @R3,R0
 5442                                       MOV     @R4,R1
 5443                                       BEQ     INF$39
 5444                                       ASL     R0
 5445                                       CLRB    R0
 5446                                       SWAB    R0
```

481


```
 5447                                       ASL     R1
 5448                                       CLRB    R1
 5449                                       SWAB    R1
 5450                                       SUB     R1,R0
 5451                                       CMP     #26.,R0
 5452                                       BLT     INF$39
 5453                                       LDF     PI$39,F3
 5454                                       LDF     @R3,F0
 5455                                       CFCC
 5456                                       BGE     A1P$39
 5457                                       NEGF    F3
 5458                              A1P$39:  LDF     @R4,F1
 5459                                       CFCC
 5460                                       BLT     A2M$39
 5461                                       CLRF    F3
 5462                              A2M$39:  DIVF    F1,F0
 5463                                       BR      AT1$39
 5464                              INF$39:  LDF     PI2$39,F1
 5465                                       TST     @R3
 5466                                       BGE     EX[$39
 5467                                       NEGF    F1
 5468                                       BR      EX[$39
 5469                              ATAN1:   SETF
 5470                                       CLRF    F3
 5471                                       LDF     @2(R5),F0
 5472                              AT[$39:  CLR     R4
 5473                                       CFCC
 5474                                       STF     F3,F5
 5475                                       CLRF    F3
 5476                                       BGE     PLU$39
 5477                                       ABSF    F0
 5478                                       INC     R4
 5479                              PLU$39:  LDF     #1.0,F1
 5480                                       CMPF    F0,F1
 5481                                       CFCC
 5482                                       BLE     LE1$39
 5483                              GT[$39:  DEC     R4
 5484                                       DIVF    F0,F1
 5485                                       LDF     F1,F0
 5486                                       LDF     PI2$39,F3
 5487                              LE[$39:  STF     F3,F4
 5488                                       CLRF    F3
 5489                                       CMPF    T15$39,F0
 5490                                       CFCC
 5491                                       BGE     L15$39
 5492                                       LDF     PI6$39,F3
 5493                                       LDF     F0,F1
 5494                                       MULF    RT3$39,F0
 5495                                       SUBF    #1.0,F0
 5496                                       ADDF    RT3$39,F1
 5497                                       DIVF    F1,F0
 5498                              L15$39:  LDF     F0,F2
 5499                                       MULF    F0,F0
 5500                                       MOV     #FCOS$9,R0
 5501                                       MOV     #4,R1
```

482

```
                                        LDF     (R0)+,F1
                                XPDS39: MULF    F0,F1
                                        DEC     R1
                                        ADDF    (R0)+,F1
                                        BGT     XPDS39
                                        MULF    F2,F1
                                        ADDF    F3,F1
                                        SUBF    F4,F1
                                        TST     R0
                                        BEQ     SGLS39
                                        NEGF    F1
                                SGL$39: ADDF    F5,F1
                                EXI$39: STF     F1,-(SP)
                                        MOV     (SP)+,R0
                                        MOV     (SP)+,R1
                                        RTS     R5
                                PI$39:  .WORD   040511,007733
                                PI2$39: .WORD   040311,007733
                                T15$39: .WORD   037611,030243
                                PI6$39: .WORD   040000,005222
                                RTS$39: .WORD   040339,131727
                                        .ENDC
004052' 037305  035302          FCO$39: .WORD   037305,035302
004056' 137421  056514                  .WORD   137421,056514
004062' 037514  143333                  .WORD   037514,143333
004066' 137652  125244                  .WORD   137652,125244
004072' 040200  000000          CON$39: .WORD   040200,000000
                                        .ENDC
                                        .TITLE  $POL07
                                        .GLOBL  $POLSH
        000004                  R4=X4
        000006                  SP=X6

                                        .GLOBL  SV20A
                                SV20A:
004076' 005726                  $POLSH: TST     (SP)+
004100' 000134                          JMP     @(R4)+
                                        .IFDF   EAE
                                        .GLOBL  $EAE
                                $EAE:
                                        .ENDC
                                        .IFDF   EIS!MVLDIV
                                        .GLOBL  $EIS
                                $EIS:
                                        .ENDC
                                        .IFDF   FPU
                                        .GLOBL  $FPU
                                $FPU:
                                        .ENDC
                                        .IFDF   FIS
                                        .GLOBL  $FIS
                                $FIS:
                                        .ENDC
                                        .IFDF   RSX
                                        .GLOBL  $RSX
```

483


```
                                $RSX:
                                        .ENDC
                                        .TITLE  $SQT03
                                        .IFDF   CND$41
                                        .GLOBL  SQRT,$ERR
                                        .IFNDF  FPU
                                        .GLOBL  $ADR,$DVR,$POLSH
                                        .ENDC
        000000                          R0=X0
        000001                          R1=X1
        000004                          R4=X4
        000005                          R5=X5
        000006                          SP=X6
        000000                          F0=X0
        000001                          F1=X1
        000002                          F2=X2
                                        .IFDF   FPU
                                SQRT:   MOV     @2(R5),R1
                                        .ENDC
                                        .IFNDF  FPU
004102' 010546                  SQRT:   MOV     R5,-(SP)
004104' 016505  000002                  MOV     2(R5),R5
004110' 011501                          MOV     @R5,R1
                                        .ENDC
004112' 100443                          BMI     ERR$41
004114' 001446                          BEQ     ERR$41
                                        .IFNDF  FPU
004116' 012746  000003                  MOV     #3,-(SP)
                                        .ENDC
004122' 006201                          ASR     R1
004124' 062701  020100                  ADD     #20100,R1
004130' 005046                          CLR     -(SP)
004132' 010146                          MOV     R1,-(SP)
                                        .IFNDF  FPU
004134' 005046                          CLR     -(SP)
004136' 011546                          MOV     @R5,-(SP)
004140' 005046                          CLR     -(SP)
004142' 010146                          MOV     R1,-(SP)
004144' 004467  177726          LUP$41: JSR     R4,$POLSH
004150' 001234' 000006' 004156'         .WORD   $DVR,$ADR,UPL$41
004156' 162716  000200          UPL$41: SUB     #200,@SP
004162' 005366  000004                  DEC     4(SP)
004166' 001410                          BEQ     OUT$41
004170' 016546  000002                  MOV     2(R5),-(SP)
004174' 011546                          MOV     @R5,-(SP)
004176' 016646  000006                  MOV     6(SP),-(SP)
004202' 016646  000006                  MOV     6(SP),-(SP)
004206' 000756                          BR      LUP$41
004210' 012600                  OUT$41: MOV     (SP)+,R0
004212' 012601                          MOV     (SP)+,R1
004214' 005726                          TST     (SP)+
004216' 012605                  RTN$41: MOV     (SP)+,R5
004220' 000205                          RTS     R5
004222' 004567  000012          ERR$41: JSR     R5,$ERR
```

484

```
5612  004226' 000773                  BR      RTNS41
5613  004230' 000004                  .BYTE   4
5614  004231' 000013                  .BYTE   11.
5615  004232' 005000          ZERS41: CLR     R0
5616  004234' 005001                  CLR     R1
5617  004236' 000767                  BR      RTNS41
5618                                  .ENDC
5619                                  .IFDF   FPU
5620                                  MOV     #3,R0
5621                                  SETF
5622                                  LDF     (SP)+,F0
5623                                  LDF     #2(R5),F2
5624                          LUPS41: LDF     F0,F1
5625                                  LDF     F2,F0
5626                                  DIVF    F1,F0
5627                                  ADDF    F1,F0
5628                                  DEC     R0
5629                                  DIVF    #2.0,F0
5630                                  BGT     LUPS41
5631                                  STF     F0,-(SP)
5632                                  MOV     (SP)+,R0
5633                                  MOV     (SP)+,R1
5634                                  RTS     R5
5635                          ERRS41: JSR     R5,$ERR
5636                                  RTS     R5
5637                                  .BYTE   4
5638                                  .BYTE   11.
5639                          ZERS41: CLR     R0
5640                                  CLR     R1
5641                                  RTS     R5
5642                                  .ENDC
5643                                  .ENDC
5644                                  .TITLE  $ERR01
5645                                  .GLOBL  $ERR,$ERRA,SERVEC
5646          000000                  R0 = %0
5647          000005                  R5 = %5
5648          000006                  SP = %6
5649          000007                  PC = %7
5650  004240' 010046          $ERR:   MOV     R0,-(SP)
5651  004242' 016500  000002          MOV     2(R5),R0
5652  004246' 000401                  BR      ERR$43
5653  004250' 010046          $ERRA:  MOV     R0,-(SP)
5654                          ERR$43: .IFNDF  CLASS5
5655  004252' 120027  000005          CMPB    R0,#5
5656  004256' 001402                  BEQ     IGN$43
5657                                  .ENDC
5658  004260' 004777  000004          JSR     PC,@SERVEC
5659  004264' 012600          IGN$43: MOV     (SP)+,R0
5660  004266' 000205                  RTS     R5
5661  004270' 004272'         SERVEC: .WORD   HLT$43
5662  004272' 000000          HLT$43: HALT
5663  004274' 000776                  BR      HLT$43
5664                                  .TITLE  $LOR01
5665                                  .IFDF   CND$44&CND$42
5666                                  R44%4
```

485


```
5667                                  SP=%6
5668                          $LOR:   MOV     (SP)+,2(SP)
5669                                  MOV     (SP)+,2(SP)
5670                                  JMP     @(R4)+
5671                                  .ENDC
5672                                  .TITLE  $LOD01
5673                                  .IFDF   CND$45&CND$42
5674                                  R0=%0
5675                                  R4=%4
5676                                  SP=%6
5677                          $LOD:   MOV     SP,R0
5678                                  ADD     #8.,R0
5679                                  MOV     (SP)+,(R0)+
5680                                  MOV     (SP)+,(R0)+
5681                                  MOV     (SP)+,(R0)+
5682                                  MOV     (SP)+,(R0)+
5683                                  JMP     @(R4)+
5684                                  .ENDC
5685                                  .TITLE  $STR01
5686                                  .IFDF   CND$46&CND$42
5687                                  R0=%0
5688                                  R1=%1
5689                                  R2=%2
5690                                  R3=%3
5691                                  R4=%4
5692                                  R5=%5
5693                                  SP=%6
5694                                  PC=%7
5695                          $STR:   MOV     #FAC$42,R5
5696                                  TST     30(SP)
5697                                  BEQ     STK$46
5698                                  CLR     30(SP)
5699                                  MOV     SP,R0
5700                                  MOV     SP,R1
5701                                  CMP     (R1)+,(R1)+
5702                                  MOV     #13.,R2
5703                          LP$46:  MOV     (R1)+,(R0)+
5704                                  DEC     R2
5705                                  BNE     LP$46
5706                                  MOV     (R5)+,(R0)+
5707                                  MOV     (R5)+,(R0)+
5708                                  JMP     @(R4)+
5709                          STK$46: MOV     (R5)+,(R0)+
5710                                  MOV     (R5)+,(R0)+
5711                                  CMP     (SP)+,(SP)+
5712                                  JMP     @(R4)+
5713                                  .ENDC
5714                                  .TITLE  $STO01
5715                                  .IFDF   CND$47&CND$42
5716                                  R0=%0
5717                                  R1=%1
5718                                  R2=%2
5719                                  R3=%3
5720                                  R4=%4
5721                                  R5=%5
```

486

```
                                SP=%6
                                PC=%7
                        SSTD1:  MOV     #FACS42,R5
                                TST     34(SP)
                                BEQ     STKS47
                                CLR     34(SP)
                                MOV     SP,R0
                                MOV     SP,R1
                                ADD     #10,R1
                                MOV     #13,R2
                        LPS47:  MOV     (R1)+,(R0)+
                                DEC     R2
                                BNE     LPS47
                                MOV     (R5)+,(R0)+
                                MOV     (R5)+,(R0)+
                                MOV     (R5)+,(R0)+
                                MOV     (R5)+,(R0)+
                                JMP     @(R4)+
                        STKS47: MOV     (R5)+,(R0)+
                                MOV     (R5)+,(R0)+
                                MOV     (R5)+,(R0)+
                                MOV     (R5)+,(R0)+
                                ADD     #10,SP
                                JMP     @(R4)+
                                .ENDC
                                .TITLE  FPMP11 FLOATING POINT & MATH PACKAGE
        000001                  .END
```

487

A	= 000010	AC	= 177302	AINT	001116RG	AIIS4	001142R
ALOG	000542RG	ASH	= 177316	ASLS4	001210R	ATAN	003412RG
AT1S39	003436R	A1	= 000004	A2	= 000010	A2NS2	000104R
B	= 000014	BT9S2	000462R	B1	= 000006	B2	= 000012
B2NS30	002424R	B2ES30	002440R	B9AS2	000402R	CFRS20	002064R
CNDS18	= 000001	CNDS2	= 000001	CNDS20	= 000001	CNDS27	= 000001
CNDS3	= 000001	CNDS30	= 000001	CNDS33	= 000001	CNDS35	= 000001
CNDS37	= 000001	CNDS39	= 000001	CNDS4	= 000001	CNDS41	= 000001
CONS3	001112R	CONS37	003406R	CONS39	004072R	COS	003036RG
COVS30	002646R	D	= 000010	DCHS18	001420R	DECS35	002770R
DHIS18	001472R	DLWS18	001464R	DNES35	002774R	DNES4	001220R
DN1S4	001226R	DUPS20	002032R	DUPS3	001000R	DUPS37	003230R
DV1S18	001606R	ECKS2	000210R	ECLS18	001444R	ECLS20	002224R
ECLS30	002606R	EC1S18	001452R	EQ1S18	001636R	EQ2S18	001654R
ERBS43	004252R	ERRFPU	000002RG	ERRS3	001002R	ERRS41	004222R
ESVS20	002056R	EXAS2	000142R	EXIS3	001032R	EXIS39	003764R
EXP	001666RG	FCOS39	004052R	FLTS18	001526R	F0	=%000000
F1	=%000001	F2	=%000002	F3	=%000003	F4	=%000004
F5	=%000005	GOS18	001632R	GT1S39	003406R	HLTS43	004272R
IFPMP	000000RG	IGNS43	004264R	INCS20	002024R	LE1S39	003740R
LGTS3	001052R	LOGS3	000544R	LSH	= 177314	LUPS41	004144R
L1SS39	003700R	MLTS30	002620R	MQ	= 177304	MT0S30	002654R
MT1S30	002624R	N	= 000014	NCPS2	000272R	NGMS35	003014R
NGOS18	001642R	NODS2	000370R	NODS27	002274R	NOMS27	002264R
NOMS30	002476R	NOR	= 177312	NODS18	001670R	NT0S18	001510R
ONES20	002202R	OUTS2	000424R	OUTS30	002544R	OUTS35	003010R
OUTS41	004210R	OVRS18	001432R	OVRS2	000434R	OVRS20	002212R
OVRS30	002570R	OVRS35	003020R	OV1S18	001430R	OV1S30	002572R
PC	=%000007	PLES20	002044R	PLUS39	003450R	PLYS37	003332R
PLYS39	004002R	PL2S3	001020R	POSS20	001706R	POSS27	002252R
PY1S37	003354R	PY1S39	004024R	PY2S37	003350R	PY2S39	004020R
Q	= 000014	QSES37	003156R	QSRS37	003312R	QSTS37	003262R
QUDS37	003254R	QUTS37	003330R	Q1SS37	003316R	REBS3	000734R
RESLT	= 000010	RORS4	001176R	RTNS18	001576R	RTNS37	003212R
RTNS41	004216R	RTSS18	001664R	RT1S37	003226R	R0	=%000000
R1	=%000001	R2	=%000002	R3	=%000003	R4	=%000004
R5	=%000005	SCKS2	000174R	SCLS20	002160R	SCLS3	000772R
SFDS2	000334R	SFLS2	000304R	SFRS2	000234R	SFTS2	000214R
SFTS35	002764R	SF0S2	000324R	SGNS18	001550R	SGNS35	003002R
SGNS39	004036R	SG1S39	004050R	SHFS4	001174R	SIGN	= 000002
SIGNS	= 000000	SIN	003072RG	SMTS20	001714R	SNCS37	003106R
SP	=%000006	SQRT	004102RG	SR	= 177311	SRLS2	000272R
STCS3	000752R	STKS3	000746R	STRS2	000414R	SUBS2	000444R
TNSS39	003600R	UNDS18	001440R	UNDS2	000534R	UNDS30	002600R
UNFS2	000524R	UPLS41	004156R	UPS3	000760R	UPS39	003770R
UTSS2	000520R	X0S30	002634R	X00S30	002602R	X4S37	003242R
ZERS18	001414R	ZERS2	000536R	ZERS20	002220R	ZERS27	002320R
ZERS30	002612R	ZERS35	003032R	ZERS41	004232R	ZE1S30	002564R
ZFRS20	002156R	ZTSS2	000500R	$ADR	000006RG	$BAS	= 000001
$OVR	001234RG	$ERR	004240RG	$ERRA	004250RG	$ERVEC	004270RG
$INTR	001134RG	$IR	002236RG	$MLR	002322RG	$POLSH	004070RG
$POPR3	002706RG	$POPR4	002674RG	$POPR5	002674RG	$RI	002714RG
$SBR	000002RG	$V20A	004076RG	.	= 004276R		

488

410

ERRORS DETECTED: 0

RUN-TIME: 46 SECONDS
CORE USED: 3K

491

492

507

```
F3        1885#   3668#   5857#   9381#
F4        5382#
F5        9383#
FCOS39    5524#
FIS       5551
FLTS18    3283    3313    3368#
FPU         24      32     188     887     814     991    1880    1879    1138    1198    1169    3224    3232    3646
          3672    3733    3759    3764    3783    4037    4038    4614    4621    4945    4958    5043    5058    5142
          5288    5304    5437    5547    5563    5574    5577    5584    5591    5619
GOS18     3398    3393    3397#
GTIS39    5363    5366#
HLTS43    5661    5662#   5663
IFPMP       20      23#
IGNS43    5656    5659#
INCS28    3696    3781#
LIMS39    5388    5383    5399    5488#
LEIS39    5362    5365    5374    5375#
LOTS3     1865    1869#
LOOS3     1814#
LSH       1147#   3217#   4088#   4943#
LUPS41    5596#   5685
MIN         38
MLTS38    4658    4653    4818#   4827
MQ         802#   1146#   3215#   4034#   4686#   4942#
MTBS38    4649    4832#   4838
MTIS38    4654    4828#
MULDIV     868     871    1178    1211    3266    3275    3289    3305    3317    3347    3386    4624    4721    4883
          4817    4983    4996    5543
N         3222#   3238    3243    3245    3249    3258
NCPS2      876     878#
NGMS35    5888    5887#
NGOS18    3392    3395    3401#
NODS2      868     986     918#    972
NODS27    4075    4087#
NOMS27    4872#   4877
NOMS38    4776    4788#
NOR        883#   3216#   4835#
NQDS18    3396    3404#
NTBS18    3311    3314#
ONES28    3671    3773#
OVTS2      822     833     922#    927
OVTS38    4797#   4816
OVTS35    5883    5885#   5888
OVTS41    5608    5606#
OV1S18    3293#   3388    3381
OV1S38    4795    4796    4889#
OVRS18    3294#   3372
OVRS2      911     918     919     926#
OVRS28    3668    3771    3776#
OVRS38    4783    4888#
OVRS35    4978    5881    5889#
PC          28     795#   1881#   1145#   1173    3214#   3387    3315    3483    3418    3656#   4885#   4649    4658
          4653    4654    4828    4839    4894#   5853#   5649#   5658
```

493

```
PL2S3     1838    1861#
PLES28    3681    3686    3786#
PLUS39    5358    5361#
PLYS37    5898    5129#
PLYS39    5488    5428#
POSS28    3663    3667#
POSS27    4861    4864#
PY1S37    5133    5136#
PY1S39    5424    5427#
PY2S37    5134#   5139
PY2S39    5425#   5438
Q         3223#   3288    3299    3388    3388    3373    3375    3378    3379
Q13S37    5118    5125#
QSES37    5887#   5124
QSRS37    5123    5124#
QSTS37    5886    5117#
QUDS37    5884    5115#
QUTS37    5126    5128#
R0         788#    839     843     852     853     858     875     888     881     888     892     893     895     988
           982     985     987     912     938     933     935     952     956     966     994#   1842   1847   1869
          1138#   1171    1178    1178    1186    1191    1197    1228    3287#   3235    3245    3246    3248    3249
          3273    3291    3294    3296    3382    3389    3391    3398    3399    3649#   3662    3664    3667    3669
          3678    3712    3715    3717    3722    3728    3738    3762    3778    3774    3776    3778    3788    4828#
          4598#   4644    4775    4778    4785    4787    4788    4789    4791    4794    4797    4889    4811    4814
          4821    4823    4824    4829    4834    4835    4887#   4897    4982    4935#   5846#   5182    5186    5129
          5135    5291#   5386    5391    5397    5488    5483    5485    5428    5426    5566#   5886    5815    5646#
          5658    5651    5653    5655    5659
R1         789#    848     844     851     859     888     887     892     896     981     988     915     916     921
           934     955     958     965     966     967     977     995#   1843   1846   1878   1139#   1172   1177
          1187    1192    1196    1219    3288#   3236    3258    3251    3277    3383    3388    3394    3397    3658#
          3718    3721    3727    3729    3763    3773    3781    4829#   4868    4863    4874    4889    4898    4891
          4892    4894    4896    4599#   4645    4774    4777    4784    4785    4786    4792    4793    4798    4815
          4828    4825    4836    4888#   4898    4983    4936#   4967    4969    4971    4972    4973    4974    4977
          4986    5847#   5183    5138    5134    5292#   5387    5398    5396    5481    5482    5484    5421    5425
          5567#   5588    5587    5588    5598    5595    5687    5616
R2         798#    817     826     837     842     848     854     856     872     878     882     884     886     898
           897     996#   1844    1848    1849    1148#   1173    1175    1216    3289#   3252    3256    3257    3258
          3259    3268    3262    3263    3273    3391    3399    3651#   4838#   4866    4876    4891    4688#   4625
          4626    4629    4638    4633    4634    4635    4823    4829    4834    4889#   4899    4937#   4965    4966
          4985    4999    5884    5885    5814    5848#   5131    5136    5137    5293#   5422    5427    5428
R3         791#    818     821     837     842     989     918     912     914     917     928     954     968     971
           978     997#   1141#   1178    1179    1188    1181    1182    1184    1189    1193    1195    1198    3218#
          3253    3264    3265    3277    3394    3397    3652#   4831#   4881#   4636    4637    4638    4828    4898#
          4988    4938#   4971    4979    4988    4981    4984    5849#   5132    5138    5294#   5423    5429
R4         792#    815     874     877     881     889     893     923     925     998#   1814    1818    1819    1831
          1858    1893    1857    1868    1863    1868    1142#   1178    1171    1172    1221    3211#   3233    3386
          3318    3314    3368    3369    3371    3373    3383    3385    3481    3487    3489    3653#   3661    3662
          3676    3677    3678    3679    3688    3782    3789    3788    3718    3731    3765    3767    3768    3769
          3778    4832#   4897    4882#   4622    4646    4651    4655    4779    4788    4782    4788    4888    4882
          4818    4832    4891#   4981    4984    4939#   5888    5858#   5859    5861    5862    5865    5867    5869
          5878    5876    5118    5114    5116    5122    5124    5128    5148    5295#   5355    5356    5357    5373
          5398    5486    5419    5431    5435    5532#   5558    5568#   5596
R5         793#    926     973     999#   1814    1872    1874    1875    1143#   1178    1218    3212#   3234    3282
```

494

504

496

506

```
                        ;
                        ; BASIC/PTS  PART3=BASICH
                        ;
                        ; DEC-11-LPTBA-A-LA3
                        ;
                        ; COPYRIGHT 1973
                        ;
                        ; DIGITAL EQUIPMENT CORPORATION
                        ; MAYNARD, MASSACHUSETTS 01754
                        ;
```



```
                            .TITLE  BASICH V00$A EDIT #020 (REF) 01/31/73
                                                BASIC SOURCE FILE #15
                        ;
                        ;
                        ;
                        ;
                        ;
                        ; USER AREA AND ONCE-ONLY INIT. CODE,
                        ;
                        ; TO BE LINKED LAST AFTER BASICL AND FPMP-11 TO FORM BASIC.
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
```

```
                                ;
                                ; GLOBALS
                                ;
                                        .GLOBL  TPB, TKB
                                        .GLOBL  START, FILLCO
                                        .GLOBL  USRAREA
                                        .GLOBL  LIMIT, PDL, ARRAYS, PDSIZE
                                        .GLOBL  TABLES, TBLSEND
                                        .GLOBL  COLUMN, FAC1, FAC2, .RPAR
                                        .GLOBL  ERRMIX, ERRPDL, ERRSYN, ERRARG
                                        .GLOBL  EVAL, INT, MAKEST, OPRATO, SOPRAT
                                        .GLOBL  .COMMA, T1, T2, T3
                                        .GLOBL  ARGB, SAVCHAR, NUMSGN, STPRO
                                        .GLOBL  NORM, VAL
                                        .GLOBL  RND1, RND2
                                        .GLOBL  SSTKSZ ,IFPMP


                                ;
                                ; ASSEMBLY PARAMETERS
                                ;
                                        .IFNDF  STPBSZ  ;TELEPRINTER BUFFER SIZE
                                STPBSZ  #20             ;CHARACTERS
                                        .ENDC
                                ;
                                        .IFNDF  SKBBSZ  ;KEYBOARD BUFF, SIZE
                                SKBBSZ  #20             ;CHARACTERS
                                        .ENDC
                                ;
                                        .IFNDF  SULNSP  ;USER LINE SPACE
                                SULNSP  #128            ;BYTES
                                        .ENDC
```



```
                000000'                 .CSECT
                000000                  R0      =%0
                000001                  R1      =%1
                000002                  R2      =%2
                000003                  R3      =%3
                000004                  R4      =%4
                000005                  R5      =%5
                000006                  SP      =%6
                000007                  PC      =%7
                000040                  BL      =40

000000' 000000                  USRAREA:.WORD   0
                                BASICH:
                                ;RND FUNCTION -- GENERATE RANDOM NUMBER
                                ;       SCAN PAST ARG IF PRESENT
000002' 004737  000000G         RNDLFN: JSR     PC,@#EVAL
000006' 103452                          BCS     ERRNDA
000010' 122127  000000G                 CMPB    (R1)+,#.RPAR
000014' 001045                          BNE     ERRNDS
                                RNDFN1:
000016' 010500                  RNDFN2: MOV     R5,R0
000020' 062700  000000G                 ADD     #RND2,R0
000024' 011003                          MOV     @R0,R3
000026' 014002                          MOV     -(R0),R2
000030' 006303                          ASL     R3              ;MULT BY 2
000032' 006102                          ROL     R2
000034' 062002                          ADD     (R0)+,R2        ;NOW BY 3
000036' 061003                          ADD     @R0,R3
000040' 005502                          ADC     R2
000042' 061002                          ADD     @R0,R2          ;NOW BY 2**16**3
000044' 100002                          BPL     RPLUS
000046' 062702  100000                  ADD     #100000,R2      ;GET 2**32+G
000052' 010310                  RPLUS:  MOV     R3,@R0          ;STORE NEW GENERATORS
000054' 010240                          MOV     R2,-(R0)
000056' 012700  000201                  MOV     #201,R0         ;INITIAL EXPONENT
000062' 006303                  RNORM:  ASL     R3              ;FLOAT RESULT
000064' 006102                          ROL     R2
000066' 103402                          BCS     REXP            ;JUMP WHEN LEADING BIT FOUND
000070' 005300                          DEC     R0              ;ADJUST EXPONENT FOR SHIFT
000072' 000773                          BR      RNORM
000074' 105003                  REXP:   CLRB    R3
000076' 150203                          BISB    R2,R3
000100' 000303                          SWAB    R3
000102' 105002                          CLRB    R2
000104' 150002                          BISB    R0,R2           ;INSERT EXPONENT INTO RESULT
000106' 000302                          SWAB    R2
000110' 006002                          ROR     R2
000112' 006003                          ROR     R3              ;INSERT + SIGN
000114' 010265  000000G                 MOV     R2,FAC1(R5)
000120' 010365  000000G                 MOV     R3,FAC2(R5)
000124' 000137  000000G                 JMP     @#OPRATOR
000130' 000137  000000G         ERRNDS: JMP     @#ERRSYN
000134' 000137  000000G         ERRNDA: JMP     @#ERRARG
                                RNDFNE:
```

500

```
                                        ;ABS FUNCTION ROUTINE
000140' 004737 0000000G         ABSFN1: JSR     PC,@#EVAL
000144' 103430                          BCS     ERABSA
000146' 122127 0000000G                 CMPB    (R1)+,#',RPAR
000152' 001027                          BNE     ERABS$
000154' 005765 0000000G                 TST     FAC1(R5)
000160' 001405                          BEQ     ABSINT
000162' 100017                          BPL     ABSX
000164' 042765 100000 0000000G          BIC     #100000,FAC1(R5)
000172' 000413                          BR      ABSX
000174' 005765 0000000G         ABSINT: TST     FAC2(R5)
000200' 100010                          BPL     ABSX
000202' 005465 0000000G                 NEG     FAC2(R5)
000206' 102005                          BVC     ABSX
000210' 012765 044000 0000000G          MOV     #44000,FAC1(R5)
000216' 005065 0000000G                 CLR     FAC2(R5)
000222' 000137 0000000G         ABSX:   JMP     @#OPRATOR
000226' 000137 0000000G         ERABSA: JMP     @#ERRARG
000232' 000137 0000000G         ERABS$: JMP     @#ERR$YN
                                ABSFNE:
```



```
                                        ;SGN FUNCTION ROUTINE
000236' 004737 0000000G         SGNFN1: JSR     PC,@#EVAL
000242' 103424                          BCS     ERSGNA
000244' 122127 0000000G                 CMPB    (R1)+,#',RPAR
000250' 001023                          BNE     ERSGN$
000252' 005000                          CLR     R0
000254' 005765 0000000G                 TST     FAC1(R5)
000260' 001003                          BNE     SGNFLT
000262' 005765 0000000G                 TST     FAC2(R5)
000266' 001410                          BEQ     SGNX
000270' 100002                  SGNFLT: BPL     SGNPOS
000272' 005300                          DEC     R0
000274' 000401                          BR      .+4
000276' 005200                  SGNPOS: INC     R0
000300' 010065 0000000G                 MOV     R0,FAC2(R5)
000304' 005065 0000000G                 CLR     FAC1(R5)
000310' 000137 0000000G         SGNX:   JMP     @#OPRATOR
000314' 000137 0000000G         ERSGNA: JMP     @#ERRARG
000320' 000137 0000000G         ERSGN$: JMP     @#ERR$YN
                                SGNFNE:
```

```
                                        ;TAB FUNCTION ROUTINE
000324' 004737 0000000G         TABFN:: JSR     PC,@#ARGB
000330' 122127 0000000          TABB:   CMPB    (R1)+,#,RPAR
000334' 001036                          BNE     ERTABS
000336' 005046                          CLR     -(SP)
000340' 116516 0000000G                 MOVB    FAC2(R5),(SP)
000344' 021627 000110           TABB1:  CMP     (SP),#72.
000350' 103403                          BLO     TABC
000352' 162716 000110                   SUB     #72.,(SP)
000356' 000772                          BR      TABB
000360' 167516 0000000G         TABC:   SUB     @COLUMN(R5),(SP)
000364' 100001                          BPL     .+4
000366' 005016                          CLR     (SP)
000370' 001002                          BNE     TABNON
000372' 005316                          DEC     (SP)
000374' 000414                          BR      TABNULL
000376' 010502                  TABNON: MOV     R5,R2
000400' 004737 0000000G                 JSR     PC,@#MAKESTR
000404' 011600                          MOV     (SP),R0
000406' 005002                          CLR     R2
000410' 151002                          BISB    (R0),R2
000412' 062700 000003                   ADD     #3,R0
000416' 112720 000040                   MOVB    #BL,(R0)+
000422' 005302                          DEC     R2
000424' 003374                          BGT     .+6
000426' 000137 0000000G         TABNULL:JMP     @#OPRATR
000432' 000137 0000000G         ERTABS: JMP     @#ERRSYN
                                TABFNE:
```



```
                                        ; LEN FUNCTION ROUTINE
000436' 004737 0000000G         LENFN:: JSR     PC,@#EVAL
000442' 103016                          BCC     ERLENA
000444' 122127 0000000G                 CMPB    (R1)+,#,RPAR
000450' 001015                          BNE     ERLENS
000452' 005065 0000000G                 CLR     FAC1(R5)
000456' 005065 0000000G                 CLR     FAC2(R5)
000462' 012602                          MOV     (SP)+,R2
000464' 005202                          INC     R2
000466' 001402                          BEQ     .+6
000470' 114265 0000000G                 MOVB    -(R2),FAC2(R5)
000474' 000137 0000000G                 JMP     @#OPRATOR
000500' 000137 0000000G         ERLENA: JMP     @#ERRARG
000504' 000137 0000000G         ERLENS: JMP     @#ERRSYN
                                LENFNE:
```

```
                                        ; ASC FUNCTION ROUTINE
000510' 004737 000000G          ASCFN:  JSR     PC,@#EVAL
000514' 103023                          BCC     ERASCA
000516' 122127 000000G                  CMPB    (R1)+,#,RPAR
000522' 001022                          BNE     ERASCS
000524' 012602                          MOV     (SP)+,R2
000526' 020227 177777                   CMP     R2,#177777
000532' 001414                          BEQ     ERASCA
000534' 121227 000001                   CMPB    (R2),#1
000540' 001011                          BNE     ERASCA
000542' 005065 000000G                  CLR     FAC1(R5)
000546' 005065 000000G                  CLR     FAC2(R5)
000552' 116265 000003  000000G          MOVB    3(R2),FAC2(R5)
000560' 000137 000000G                  JMP     @#OPRATOR
000564' 000137 000000G          ERASCA: JMP     @#ERRARG
000570' 000137 000000G          ERASCS: JMP     @#ERR$YN
                                ASCFNE:
                                        ; CHR$ FUNCTION ROUTINE
000574' 004737 000000G          CHRSFN: JSR     PC,@#ARGB
000600' 122127 000000G                  CMPB    (R1)+,#,RPAR
000604' 001016                          BNE     ERCHRS
000606' 142765 000200  000000G          BICB    #200,FAC2(R5)   ;TAKE CHAR MOD 128
000614' 012746 000001                   MOV     #1,-(SP)
000620' 010502                          MOV     R5,R2
000622' 004737 000000G                  JSR     PC,@#MAKE$TR
000626' 011600                          MOV     (SP),R0
000630' 116500 000000G 000003           MOVB    FAC2(R5),3(R0)
000636' 000137 000000G                  JMP     @#OPRATR
000642' 000137 000000G          ERCHRS: JMP     @#ERR$YN
                                CHRFNE:
```



```
                                        ; POS FUNCTION ROUTINE
000646' 004737 000000G          POSFN:  JSR     PC,@#EVAL
000652' 103133                          BCC     ERPOSA
000654' 122127 000000G                  CMPB    (R1)+,#,COMMA
000660' 001132                          BNE     ERPOS$
000662' 004737 000000G                  JSR     PC,@#EVAL
000666' 103125                          BCC     ERPOSA
000670' 122127 000000G                  CMPB    (R1)+,#,COMMA
000674' 001124                          BNE     ERPOS$
000676' 004737 000000G                  JSR     PC,@#EVAL
000702' 103517                          BCS     ERPOSA
000704' 004737 000000G                  JSR     PC,@#INT
000710' 122127 000000G                  CMPB    (R1)+,#,RPAR
000714' 001114                          BNE     ERPOS$
000716' 005000                          CLR     R0
000720' 005765 000000G                  TST     FAC1(R5)
000724' 001077                          BNE     POSF
000726' 005765 000000G                  TST     FAC2(R5)
000732' 003474                          BLE     POSF
000734' 156500 000000G                  BISB    FAC2(R5),R0     ;R0 IS N
                                ;CHECK NULL XS
000740' 026627 000002  177777           CMP     2(SP),#177777
000746' 001002                          BNE     POSX
000750' 005000                          CLR     R0
000752' 000464                          BR      POSF
                                ;COMPUTE LENGTH OF XS
000754' 005002                  POSX:   CLR     R2
000756' 157602 000002                   BISB    @2(SP),R2
                                ;CHECK NULL YS
000762' 021627 177777                   CMP     (SP),#177777
000766' 001004                          BNE     POSY
                                ;NULL YS, ANS IS MIN(N,LEN(XS))
000770' 020002                          CMP     R0,R2
000772' 003454                          BLE     POSF
000774' 010200                          MOV     R2,R0
000776' 000452                          BR      POSF
                                ;SAVE ADDR OF END OF YS IN T2
001000' 005003                  POSY:   CLR     R3
001002' 157603 000000                   BISB    @(SP),R3
001006' 061603                          ADD     (SP),R3
001010' 062703 000003                   ADD     #3,R3
001014' 010365 000000G                  MOV     R3,T2(R5)
001020' 012603                          MOV     (SP)+,R3
001022' 062703 000003                   ADD     #3,R3
                                ;SAVE ADDR OF END OF XS IN T1
001026' 061602                          ADD     (SP),R2
001030' 062702 000003                   ADD     #3,R2
001034' 010265 000000G                  MOV     R2,T1(R5)
001040' 012602                          MOV     (SP)+,R2
001042' 060002                          ADD     R0,R2
001044' 062702 000002                   ADD     #2,R2
                                ;FIND MATCHING FIRST CHARACTER
001050' 121322                  POSFST: CMPB    (R3),(R2)+
001052' 001406                          BEQ     POSREM
001054' 005200                  POSTRY: INC     R0
```

506

```
  298  001056' 020265  000000G                CMP     R2,T1(R5)
  299  001062' 001372                          BNE     POSFS1
  300  001064' 005000                          CLR     R0
  301  001066' 000417                          BR      POSF2
  302                                  ;CHECK THAT REMAINING CHARS MATCH
  303  001070' 010246                  POSREM: MOV     R2,-(SP)
  304  001072' 010346                          MOV     R3,-(SP)
  305  001074' 005203                          INC     R3
  306  001076' 020365  000000G         POSNXT: CMP     R3,T2(R5)
  307  001102' 001410                          BEQ     POSF
  308  001104' 020265  000000G                 CMP     R2,T1(R5)
  309  001110' 001402                          BEQ     POSNO
  310  001112' 122223                          CMPB    (R2)+,(R3)+
  311  001114' 001770                          BEQ     POSNXT
  312  001116' 012603                  POSNO:  MOV     (SP)+,R3
  313  001120' 012602                          MOV     (SP)+,R2
  314  001122' 000754                          BR      POSTRY
  315                                  ;RETURN FROM FUNCTION
  316  001124' 022626                  POSF:   CMP     (SP)+,(SP)+
  317  001126' 010065  000000G         POSF2:  MOV     R0,FAC2(R5)
  318  001132' 005065  000000G                 CLR     FAC1(R5)
  319  001136' 000137  000000G                 JMP     @#OPRATOR
  320  001142' 000137  000000G         ERPOSA: JMP     @#ERRARG
  321  001146' 000137  000000G         ERPOSS: JMP     @#ERRSYN
  322                                  POSFNE:
  323
  324
```



```
  325                                  ; SEG FUNCTION ROUTINE
  326  001152' 004737  000000G         SEGFN:  JSR     PC,@#EVAL
  327  001156' 103114                          BCC     ERSEGA
  328  001160' 122127  000000G                 CMPB    (R1)+,#,COMMA
  329  001164' 001113                          BNE     ERSEG$
  330  001166' 004737  000000G                 JSR     PC,@#EVAL
  331  001172' 103506                          BCS     ERSEGA
  332  001174' 004737  000000G                 JSR     PC,@#INT
  333  001200' 016546  000000G                 MOV     FAC1(R5),-(SP)
  334  001204' 016546  000000G                 MOV     FAC2(R5),-(SP)
  335  001210' 122127  000000G                 CMPB    (R1)+,#,COMMA
  336  001214' 001077                          BNE     ERSEG$
  337  001216' 004737  000000G                 JSR     PC,@#EVAL
  338  001222' 103472                          BCS     ERSEGA
  339  001224' 004737  000000G                 JSR     PC,@#INT
  340  001230' 122127  000000G                 CMPB    (R1)+,#,RPAR
  341  001234' 001067                          BNE     ERSEG$
  342                                  ;GET X VALUE IN R2
  343  001236' 012602                          MOV     (SP)+,R2
  344  001240' 012600                          MOV     (SP)+,R0
  345  001242' 100403                          BMI     SEGX0
  346  001244' 001055                          BNE     SEGNUL
  347  001246' 005702                          TST     R2
  348  001250' 003002                          BGT     SEGL
  349  001252' 012702  000001          SEGX0:  MOV     #1,R2
  350                                  ;COMPUTE LENGTH OF AS IN R0
  351  001256' 005000                  SEGL:   CLR     R0
  352  001260' 021627  177777                  CMP     (SP),#177777    ;CHECK NULL STRING
  353  001264' 157600  000000                  BISB    @(SP),R0
  354                                  ;GET Y VALUE IN R3
  355  001270' 005765  000000G         SEGTY:  TST     FAC1(R5)
  356  001274' 100441                          BMI     SEGNUL
  357  001276' 001402                          BEQ     SEGTY2
  358  001300' 010003                          MOV     R0,R3
  359  001302' 000403                          BR      SEGRNG
  360  001304' 016503  000000G         SEGTY2: MOV     FAC2(R5),R3
  361  001310' 003433                          BLE     SEGNUL
  362                                  ;CHECK 0<=R2<=R3<=R0
  363  001312' 020003                  SEGRNG: CMP     R0,R3
  364  001314' 101001                          BHI     SEGR2
  365  001316' 010003                          MOV     R0,R3
  366  001320' 020203                  SEGR2:  CMP     R2,R3
  367  001322' 101026                          BHI     SEGNUL
  368                                  ;COMPUTE LENGTH OF OUTPUT STRING
  369  001324' 160203                          SUB     R2,R3
  370  001326' 005203                          INC     R3
  371                                  ;MAKE NEW STRING OF THE CORRECT LENGTH
  372  001330' 010246                          MOV     R2,-(SP)        ;SAVE START CHAR
  373  001332' 010502                          MOV     R5,R2
  374  001334' 010346                          MOV     R3,-(SP)
  375  001336' 004737  000000G                 JSR     PC,@#MAKESTR
  376                                  ;FIX STACK AND MOVE IN CHARS FROM OLD STRING
  377  001342' 012602                          MOV     (SP)+,R2        ;NEW STRING POINTER
  378  001344' 012600                          MOV     (SP)+,R0        ;START CHAR
  379  001346' 061600                          ADD     (SP),R0         ;ADD OLD STR POINTER
```

508

```
001350' 010216                          MOV     R2,(SP)         ;SAVE NEW STRING
001352' 005003                          CLR     R3
001354' 151203                          BISB    (R2),R3         ;R3 IS NEW STRING LENGTH
001356' 062702 000003                   ADD     #3,R2           ;ADDR FIRST CHAR IN NEW STR
001362' 062700 000002                   ADD     #2,R0           ;ADDR START CHAR IN OLD STR
001366' 112022                  SEGLP:  MOVB    (R0)+,(R2)+     ;FILL IN NEW STRING
001370' 005303                          DEC     R3
001372' 001375                          BNE     SEGLP
001374' 000137 000000G                  JMP     @#STPRO
                                ;OUTPUT NULL STRING
001400' 012716 177777           SEGNUL: MOV     #177777,(SP)
001404' 000137 000000G          SEGX:   JMP     @#SOPRATR
001410' 000137 000000G          ERSEGA: JMP     @#ERRARG
001414' 000137 000000G          ERSEGS: JMP     @#ERRSYN
                                SEGFNE:
```



```
                                ; VAL FUNCTION ROUTINE
001420' 004737 000000G          VALFN:  JSR     PC,@#EVAL
001424' 103056                          BCC     ERVALA
001426' 122127 000000G                  CMPB    (R1)+,#,RPAR
001432' 001051                          BNE     ERVALS
                                ;READ STRING
001434' 011600                          MOV     (SP),R0
001436' 020027 177777                   CMP     R0,#177777      ;CHECK NULL STRING
001442' 001006                          BNE     VALR
001444' 005065 000000G                  CLR     FAC1(R5)
001450' 005065 000000G                  CLR     FAC2(R5)
001454' 005726                          TST     (SP)+
001456' 000435                          BR      VALJ
001460' 005002                  VALR:   CLR     R2
001462' 151002                          BISB    (R0),R2
001464' 010216                          MOV     R2,(SP)
001466' 062700 000003                   ADD     #3,R0
001472' 060002                          ADD     R0,R2
001474' 105012                          CLRB    (R2)
001476' 010446                          MOV     R4,-(SP)
001500' 010146                          MOV     R1,-(SP)
001502' 010246                          MOV     R2,-(SP)
                                ;READ ASCII NUMBER AND EXPONENT
001504' 004737 000000G                  JSR     PC,@#VAL
                                ;NORMALIZE TO FORM FLT PT NUMBER
001510' 004737 000000G                  JSR     PC,@#NORM
001514' 010365 000000G                  MOV     R3,FAC1(R5)
001520' 010465 000000G                  MOV     R4,FAC2(R5)
                                ;CHECK END OF STRING
001524' 105710                  VALCE:  TSTB    (R0)
001526' 001403                          BEQ     VALX
001530' 122027 000040                   CMPB    (R0)+,#BL
001534' 001773                          BEQ     VALCE
                                ;RESTORE REGS AND RETURN TO EVAL
001536' 020026                  VALX:   CMP     R0,(SP)+
001540' 001010                          BNE     ERVALA
001542' 012601                          MOV     (SP)+,R1
001544' 012604                          MOV     (SP)+,R4
001546' 012602                          MOV     (SP)+,R2
001550' 110210                          MOVB    R2,(R0)         ;RESTORE STRING LENGTH
001552' 000137 000000G          VALJ:   JMP     @#OPRATOR
001556' 000137 000000G          ERVALS: JMP     @#ERRSYN
001562' 000137 000000G          ERVALA: JMP     @#ERRARG
                                VALFNE:
```

```
                                ; STR FUNCTION ROUTINE
001566' 004737 0000000G         STRFN: JSR     PC,@#EVAL
001572' 103432                         BCS     ERSTRA
001574' 122127 0000000G                CMPB    (R1)+,#,RPAR
001600' 001031                         BNE     ERSTRS
001602' 012746 000020                  MOV     #20,-(SP)
001606' 010502                         MOV     R5,R2
001610' 004737 0000000G                JSR     PC,@#MAKESTR
001614' 011603                         MOV     (SP),R3
001616' 062703 000003                  ADD     #3,R3
001622' 010365 0000000G                MOV     R3,T2(R5)
001626' 005065 0000000G                CLR     T1(R5)
001632' 004737 0000000G                JSR     PC,@#NUMSGN
001636' 0000000G                       .WORD   SAVCHAR
001640' 010602                         MOV     SP,R2
001642' 016546 0000000G                MOV     T1(R5),-(SP)
001646' 004737 0000000G                JSR     PC,@#MAKESTR
001652' 012616                         MOV     (SP)+,(SP)
001654' 000137 0000000G                JMP     @#STPRO         ;PROTECT STRING

001660' 000137 0000000G         ERSTRA: JMP     @#ERRARG
001664' 000137 0000000G         ERSTRS: JMP     @#ERRSYN
                                STRFNE:
                                FNEND:
```



```
                                ;
                                ;
                                ; USER AREA STORAGE CELLS
                                ;
001670' 000000                  USR:    .WORD   0               ;SYMBOLS
001672' 000000                          .WORD   0               ;LIMIT
001674' 000000                          .WORD   0               ;PDL
001676' 000000                          .WORD   0               ;POSIZE
001700' 000000                          .WORD   0               ;ARRAYS
001702' 000000                          .WORD   0               ;HIFREE
001704' 000000                          .WORD   0               ;LOFREE
001706' 002214'                         .WORD   USRCODE         ;CODE
001710' 002074'                         .WORD   USRLINE         ;LINE
001712' 000000                          .WORD   0               ;VARSAV
001714' 000000                          .WORD   0               ;$S1SAV
001716' 000000                          .WORD   0               ;$S2SAV
001720' 000000                          .WORD   0               ;LINENO
001722' 000000                          .WORD   0               ;SSBCTR
001724' 000000                          .WORD   0               ;COLUMN
001726' 000000                          .WORD   0               ;CLMNTTY
001730' 000000                          .WORD   0               ;FAC1
001732' 000000                          .WORD   0               ;FAC2
001734' 000000                          .WORD   0               ;R0SAVE
001736' 000000                          .WORD   0               ;R1SAVE
001740' 000000                          .WORD   0               ;R2SAVE
001742' 000000                          .WORD   0               ;R3SAVE
001744' 000000                          .WORD   0               ;R4SAVE
001746' 000000                          .WORD   0               ;T1
001750' 000000                          .WORD   0               ;T2
001752' 000000                          .WORD   0               ;T3
001754' 000000                          .WORD   0               ;RND1
001756' 000000                          .WORD   0               ;RND2
001760' 000000                  RAND:   .WORD   0               ;RNDCT
001762' 000000                          .WORD   0               ;LOSTR
001764' 000000                          .WORD   0               ;HISTR
001766'     000                         .BYTE   0               ;ODEV
001767'     000                         .BYTE   0               ;IDEV
001770' 002048'                         .       TPBFH           ;TPHD
001772' 002004'                         .       KBBFH           ;KBHD
001774' 000000                          .WORD   0               ;ECHOSP
001776'     000                         .BYTE   0               ;CNCFLG
001777'     000                         .BYTE   0               ;CNOFLG
002000'     000                         .BYTE   0               ;FILLCO
002001'     000                         .BYTE   0               ;FILLNO
002002'     000                         .BYTE   0               ;FILLCH
002003'     000                         .BYTE   0               ; (SPARE BYTE)
```

512

```
                                        ;
                                        ; USER I/O BUFFER AND BUFFER HDR, SPACE (TTY)
                                        ;
002804' 002828'                 KBDFH1: .       KBBUF1          ;START
                                        .       KBBEN1          ;END
002810' 002828'                         .WORD   KBBUF1          ;GET1
002812' 002828'                         .WORD   KBBUF1          ;GET2
002814' 002828'                         .WORD   KBBUF1          ;PUT
002816' 000000                          .       0               ;SPECFC
                002828'         KBBUF1: .;
                002848'                 .;+$KBBSZ               ;DEFAULT SIZE = 20 CHARS.
                002837'         KBBEN1: .;=.
                                        .EVEN
                                        ;
002840' 002854'                 TPDFH1: .       TPBUF1          ;START
002842' 002873'                         .       TPBEN1          ;END
002844' 000000                          .       0               ;GET1 (NOT USED)
002846' 002854'                         .WORD   TPBUF1          ;GET2
002850' 002854'                         .WORD   TPBUF1          ;PUT
002852' 000000                          .       0               ;SPECFC
                002854'         TPBUF1: .;
                002874'                 .;+$TPBSZ               ;DEFAULT = 20 CHARS.
                002873'         TPBEN1: .;=.
                                        .EVEN
                                        USRLINE:
                002214'                 .;+SULNSP               ;DEFAULT = 128
                                        USRCODE:
```



```
                                        ;
                                        ; *-ONCE-ONLY INIT, CODE
                                        ;    FIND OUT HOW MUCH CORE AND EXPAND USRAREA ACCORDINGLY
                                        ;
002214' 162704  017770          DECCORE:SUB     #17770,R4       ;COME HERE ON TRAP FOR BAD MEM,
002220' 022626                          CMP     (SP)+,(SP)+
002222' 000412                          BR      TRYCORE         ;  REF,, AND DEC, CORE SIZE
                                        ;
002224' 000005                  ONCEONLY:RESET
002226' 012706  003364'                 MOV     #TMPSTK,SP      ;TEMP, STACK
002232' 004767  000000G                 JSR     PC,IFPMP        ;INIT FOR FPMP ROUTINES
002236' 012737  002214' 000004          MOV     #DECCORE,@#4    ;SET UP TRAP ADDR,
002244' 012704  160000                  MOV     #160000,R4      ;28K
002250' 005744                  TRYCORE:TST     -(R4)           ;TRAPS TO DECCORE IF BAD ADDR
002252' 012737  000000  000004          MOV     #0,@#4          ;FALLS THRU IF OK = RESTORE TRAP ADDR,
002260' 010467  001100                  MOV     R4,HICORE
                                        ;
                                        ;INITIALIZE TOP OF BASIC
002264' 012701  000002'                 MOV     #0BASIC,R1
                                        ;PRINT HEADING AND READ ANSWER
002270' 004367  000756          INIMO:  JSR     R3,PRMSG
002274'    015     012                  .BYTE   15,12
002276' 040502  044523  020103          .ASCII  'BASIC V00(A)'
002304' 030126  030460     101
002311'    015     012                  .BYTE   15,12
002313'    117  052120  043040          .ASCII  'OPT ENS: A=ALL, N=NONE, I=IND)'
002320' 051516  020072  026501
002326' 046101  026114  047040
002334' 047055  047117  026105
002342' 044440  044455  042116
002350'    015     012                  .BYTE   15,12
002352'    077                          .ASCII  '?'
002353'    000                          .BYTE   0
                                        .EVEN
002354' 004767  000722                  JSR     PC,RDANS
002360' 120027  000116                  CMPB    R0,#'N          ;CHECK FOR N
002364' 001512                          BEQ     INISAV
002366' 005002                          CLR     R2              ;CLEAR ALL FLAG
002370' 120027  000101                  CMPB    R0,#'A          ;CHECK FOR A
002374' 001002                          BNE     INITI
002376' 005202                          INC     R2              ;SET ALL FLAG
002400' 000415                          BR      INIALL
002402' 120027  000111          INITI:  CMPB    R0,#'I          ;CHECK FOR I
002406' 001330                          BNE     INIMO
                                        ;PRINT HEADING FOR INDIV FUNCTIONS
002410' 004367  000036                  JSR     R3,PRMSG
002414'    015     012     012          .BYTE   15,12,12
002417'    040  020531  042531          .ASCII  ' Y=YES  N=NO'
002424' 020123  047040  047055
002432'    117
002433'    000                          .BYTE   0
002434' 012705  003374'         INIALL: MOV     #ENTAB,R5
002440' 012704  000000G                 MOV     #TABLES,R4
                                        ;GET NEXT FUNCTION TABLE ADDRESS
002444' 011503                  INILP:  MOV     (R5),R3
```

514

```
593  002446' 100005                  BPL     INIFN
594  002450' 062705  000002          ADD     #2,R5           ;SKIP A FUNCTION
595  002454' 062704  000002          ADD     #2,R4
596  002460' 000771                  BR      INILP
597                                  ;PRINT QUESTION FOR FUNCTION, SET ANSWER
598  002462' 020427  0000000         INIFN:  CMP     R4,@TBLEND
599  002466' 101051                  BHI     INISAV
600  002470' 020527  003944'         CMP     R5,#FNTABE
601  002474' 103046                  BHIS    INISAV
602  002476' 005702                  TST     R2
603  002500' 001027                  BNE     INIDF
604  002502' 016567  000004  000014  MOV     4(R5),FNNAM1
605  002510' 016567  000006  000016  MOV     6(R5),FNNAM2
606  002516' 004367  000530          JSR     R3,PRMSG
607  002522'    015     012          .BYTE   15,12
608  002524' 000000          FNNAM1: .WORD   0
609  002526' 000000          FNNAM2: .WORD   0
610  002530' 039440     040          .ASCII  ' ) '
611  002533'    000                  .BYTE   0
612                                  .EVEN
613  002534' 004767  000542          JSR     PC,RDANS
614  002540' 120027  000116          CMPB    R0,#'N          ;CHECK FOR N
615  002544' 001002                  BNE     INITY
616  002546' 005724                  TST     (R4)+
617  002550' 000413                  BR      ININXT
618  002552' 120027  000131  INITY: CMPB    R0,#'Y          ;CHECK FOR Y
619  002556' 001341                  BNE     INIFN
620                                  ;DEFINE THE FUNCTION
621  002560' 016100                  INIDF:  MOV     R1,R0
622  002562' 162700  0000000         SUB     #TABLE5,R0
623  002566' 010024                  MOV     R0,(R4)+        ;PLACE ADDRESS IN TABLE
624  002570' 012321          INIMV:  MOV     (R3)+,(R1)+
625  002572' 020365  000002          CMP     R3,2(R5)
626  002576' 103774                  BLO     INIMV
627  002600' 062705  000010  ININXT: ADD     #10,R5
628  002604' 020427  0000000         CMP     R4,@TBLEND
629  002610' 101715                  BLOS    INILP
630
631                                  ;SAVE TOP OF BASIC
632  002612' 010167  175102          INISAV: MOV     R1,USRAREA
633                                  ;MOVE USER AREA TO TOP OF BASIC
634  002616' 012702  001700          MOV     #USR,R2
635  002622' 010203                  MOV     R2,R3           ;COMPUTE RELOCATION CONSTANT
636  002624' 160103                  SUB     R1,R3
637  002626' 020227  002214'         MOVLP:  CMP     R2,#USRCODE
638  002632' 103005                  BHIS    MOVDON
639  002634' 012221                  MOV     (R2)+,(R1)+
640  002636' 001773                  BEQ     MOVLP
641  002640' 160341                  SUB     R3,-(R1)        ;RELOCATE NON-ZERO ENTRIES
642  002642' 005721                  TST     (R1)+
643  002644' 000770                  BR      MOVLP
644
645                                  MOVDON:
```

515


```
646
647                                  ;INITIALIZE USER AREA
648  002646' 016704  000512          MOV     HICORE,R4       ;R4 IS HIGHEST ADDR, IN CORE
649  002652' 162704  000300          SUB     #300,R4
650  002656' 016705  175116          MOV     USRAREA,R5
651  002662' 010465  0000000         MOV     R4,LIMIT(R5)
652  002666' 162704  000134          SUB     #134,R4         ;POINTER SPACE
653  002672' 010465  0000006         MOV     R4,FOR(R5)      ;BASE OF STACK
654  002676' 162704  0000006         SUB     #$$TKSE,R4      ;MAX. STACK SIZE
655  002702' 010465  0000006         MOV     R4,ARRAYS(R5)   ;HIEST ARRAY STOR. LOC.
656  002706' 062704  000040          ADD     #40,R4
657  002712' 010465  0000000         MOV     R4,POSIZE(R5)
658  002716' 012765  032331  0000000 MOV     #32331,RND1(R5)
659  002724' 012765  163251  0000000 MOV     #163251,RND2(R5)
660
661                                  ;DIALOGUE TO SET UP TERMINAL TYPE
662  002732' 005003          ASKSER: CLR     R3
663  002734' 004367  000312          JSR     R3,PRMSG
664  002740'    015     012     012  .BYTE   15,12,12
665  002743'    106  051501  020124  .ASCII  'FAST $ER TERM?'
     002750' 042523  020122  042524
     002756' 046522     077
666  002761'    000                  .BYTE   0
667                                  .EVEN
668  002762' 004767  000314          JSR     PC,RDANS
669  002766' 120027  000116          CMPB    R0,#'N
670  002772' 001457                  BEQ     ASKDON
671  002774' 120027  000131          CMPB    R0,#'Y
672  003000' 001354                  BNE     ASKSER
673  003002' 005203                  INC     R3
674
675  003004' 004367  000242  ASKLA:  JSR     R3,PRMSG
676  003010'    015     012          .BYTE   15,12
677  003012' 040514  030063  024040  .ASCII  'LA36 (300 BAUD)?'
     003020' 030063  020060  040502
     003026' 042125  037451
678  003032'    000                  .BYTE   0
679         003034'                  .EVEN
680  003034' 004767  000242          JSR     PC,RDANS
681  003040' 120027  000131          CMPB    R0,#'Y
682  003044' 001432                  BEQ     ASKDON
683  003046' 120027  000116          CMPB    R0,#'N
684  003052' 001354                  BNE     ASKLA
685  003054' 005203                  INC     R3
686
687  003056' 004367  000170  ASKVT:  JSR     R3,PRMSG
688  003062'    015     012          .BYTE   15,12
689  003064' 052126  032460  024040  .ASCII  'VT05 (>=600 BAUD)?'
     003072' 036476  030060  020060
     003100' 040502  042125  037451
690  003106'    000                  .BYTE   0
691         003110'                  .EVEN
692  003110' 004767  000166          JSR     PC,RDANS
693  003114' 120027  000131          CMPB    R0,#'Y
694  003120' 001404                  BEQ     ASKDON
```

516

```
                                CMPB    R0,#'N
                                BNE     ASKVT
                                BR      ASKSER
                        ASKDON: MOV     R3,R2
                                BEQ     FILLDON
                                ASL     R3
                                ADD     R3,R2
                                ADD     #FILLIB,R2
                                MOV     R5,R3
                                ADD     #FILLCO,R3
                                MOVB    (R2)+,(R3)+
                                MOVB    (R2)+,(R3)+
                                MOVB    (R2)+,(R3)+
                        FILLDON:
                        ;PRINT OUT IF USER FUNCTIONS LOADED
                                TST     @#46
                                BEQ     NOUSR
                                JSR     R3,PRMSG
                                .BYTE   12.,12.,12.
                                .ASCII  /USER FNS LOADED/
                                .BYTE 0
                                .EVEN
                        NOUSR:
                                JSR     R3,PRMSG
                                .BYTE   12.,12.,12.,0
                                .EVEN

                        ;CHECK IF RNDL LOADED
                                MOV     #TABLE5,R2
                                MOV     (R2)+,R3
                                BEQ     NORND
                                ADD     #RNDFN-RNDLFN,R3
                                MOV     R3,(R2)         ;DEFINE RND ALSO
                        NORND:
                                JMP     START

                        ;SUBROUTINE TO PRINT A MESSAGE
                        PRMSG:  MOV     R3,R0
                        PRLP:   TSTB    @#TPS+2
                                BPL     PRLP
                                MOVB    (R0)+,@#TPB
                                BNE     PRLP
                                MOV     R0,R3
                                INC     R3
                                ASR     R3
                                ASL     R3
                                RTS     R3

                        ;SUBROUTINE TO READ ANSWER FROM TELETYPE
                        RDANS:  INC     RAND
                                TSTB    @#TKS
```

517


```
                                BPL     RDANS
                                MOVB    @#TKS+2,R0
                                BIC     #177600,R0
                                MOV     R0,INCH
                                JSR     R3,PRMSG
                        INCH:   .WORD   0
                                MOV     INCH,R0
                                RTS     PC
                                .       .+20
                        TMPSTCK:
                        HICORE: .WORD   0

                        FILLTB=.-3
                                .BYTE   0,11.,15.       ;LA30
                                .BYTE   0,04,12.        ;VT05

                        FNTAB:
                                .WORD   RNDLFN
                                .WORD   RNDFNE
                                .ASCII  /RND !
                                .WORD   -1              ; RND
                                .WORD   -1              ; SIN
                                .WORD   -1              ; COS
                                .WORD   -1              ; SQR
                                .WORD   -1              ; ATN
                                .WORD   -1              ; EXP
                                .WORD   -1              ; LOG
                                .WORD   ABSFN           ; ABS
                                .WORD   ABSFNE
                                .ASCII  /ABS !
                                .WORD   -1              ; INT
                                .WORD   SGNFN           ; SGN
                                .WORD   SGNFNE
                                .ASCII  /SGN !
                                .WORD   TABFN           ; TAB
                                .WORD   TABFNE
                                .ASCII  /TAB !
                                .WORD   LENFN           ; LEN
                                .WORD   LENFNE
                                .ASCII  /LEN !
                                .WORD   ASCFN           ; ASC
                                .WORD   ASCFNE
                                .ASCII  /ASC !
                                .WORD   CHRSFN          ; CHR$
                                .WORD   CHRFNE
                                .ASCII  /CHR$!
                                .WORD   POSFN           ; POS
                                .WORD   POSFNE
                                .ASCII  /POS !
                                .WORD   SEGFN           ; SEG
                                .WORD   SEGFNE
                                .ASCII  /SEG$!
                                .WORD   VALFN           ; VAL
                                .WORD   VALFNE
                                .ASCII  /VAL !
```

518

```
  002  003534' 001566'                    ,WORD   STRFN          ; STR
  003  003536' 001678'                    ,WORD   STRFNE
  004  003540' 052123  022122             ,ASCII  /$TRS/
  005                              FNTABE:
  006
  007          002224'                    ,END    ONCEONL
```



```
ABSFN    000140R     ABSFNE   000236R     ABSINT   000174R     ABSX     000222R
ARGB   = ******  G   ARRAYS = ******  G   ASCFN    000510R     ASCFNE   000974R
ASKDON   003132R     ASKLA    003004R     ASKSER   002732R     ASKVT    003050R
BASICH   000002R     BL     = 000040      CHRFNE   000646R     CHRSFN   000574R
COLUMN = ******  G   DECCOR   002214R     ERABSA   000226R     ERABSS   000232R
ERASCA   000564R     ERASCS   000570R     ERCHRS   000642R     ERLENA   000500R
ERLENS   000504R     ERPOSA   001142R     ERPOSS   001146R     ERRARG = ******  G
ERRMIX = ******  G   ERRNDA   000134R     ERRNOS   000130R     ERRPOL = ******  G
ERRSYN = ******  G   ERSEGA   001410R     ERSEGS   001414R     ERSGNA   000314R
ERSGNS   000320R     ERSTRA   001660R     ERSTRS   001664R     ERTABS   000432R
ERVALA   001562R     ERVALS   001556R     EVAL   = ******  G   FAC1   = ******  G
FAC2   = ******  G   FILLCO = ******  G   FILLDO   003102R     FILLTB = 003306R
FNEND    001670R     FNNAM1   002524R     FNNAM2   002526R     FNTAB    003374R
FNTABE   003544R     HICORE   003364R     IFPMP  = ******  G   INCH     003334R
INIALL   002434R     INIOF    002546R     INIFN    002402R     INIHD    002270R
INILP    002444R     INIMV    002570R     ININXT   002000R     INISAV   002012R
INITI    002402R     INITY    002552R     INT    = ******  G   KBBEN1 = 002037R
KBBFH1   002064R     KBBUF1 = 002020R     LENFN    000436R     LENFNE   000510R
LIMIT  = ******  G   MAKEST = ******  G   MOVDON   002646R     MOVLP    002620R
NORM   = ******  G   NORND    003246R     NOUSR    003220R     NUMSGN = ******  G
ONCEON   002224R     OPRATO = ******  G   PC     =%000007      POL    = ******  G
POSIZE = ******  G   POSF     001124R     POSFN    000646R     POSFNE   001152R
POSFST   001050R     POSF2    001126R     POSNO    001110R     POSNXT   001076R
POSREM   001070R     POSTRY   001054R     POSX     000754R     POSY     001000R
PRLP     003254R     PRMSG    003252R     RAND     001700R     ROANS    003302R
REXP     000074R     RNDFN    000616R     RNDFNE   000140R     RNDFN2   000616R
RNDLFN   000002R     RND1   = ******  G   RND2   = ******  G   RNORM    000662R
RPLUS    000052R     R0     =%000000      R1     =%000001      R2     =%000002
R3     =%000003      R4     =%000004      R5     =%000005      SAVCHA = ******  G
SEGFN    001152R     SEGFNE   001420R     SEGL     001256R     SEGLP    001366R
SEGNUL   001400R     SEGRNG   001312R     SEGR2    001320R     SEGTY    001270R
SEGTY2   001304R     SEGX     001404R     SEGX0    001252R     SGNFLT   000270R
SGNFN    000236R     SGNFNE   000324R     SGNPOS   000276R     SGNX     000310R
SOPRAT = ******  G   SP     =%000006      START  = ******  G   STPRO  = ******  G
STRFN    001566R     STRFNE   001670R     TAB8     000344R     TABC     000360R
TABFN    000324R     TABFNE   000436R     TABLES = ******  G   TABNON   000370R
TABNUL   000426R     TBLSEN = ******  G   TKS    = ******  G   TMPSTC   003364R
TPB    = ******  G   TPBEN1 = 002073R     TPBFH1   002064R     TPBUF1 = 002054R
TRYCOR   002250R     T1     = ******  G   T2     = ******  G   T3     = ******  G
USR      001670R     USRARE   000000RG    USRCOD   002214R     USRLIN   002074R
VAL    = ******  G   VALCE    001524R     VALFN    001420R     VALFNE   001566R
VALJ     001552R     VALR     001460R     VALX     001536R     $KBBSZ = 000020
$STKSZ = ******  G   STPBSZ = 000020      $ULNSP = 000120      ,COMMA = ******  G
,RPAR  = ******  G   .      = 003544R

ERRORS DETECTED: 0
```

520

RUN-TIME: 18 SECONDS
CORE USED: 3K

ABSFN	128#	774											
ABSFNE	138#	775											
ABSINT	125	129#											
ABSX	126	128	138	132	135#								
ARGB	41	164	229										
ARRAY$	35	655											
ASCFN	211#	787											
ASCFNE	226#	788											
ASKDON	678	682	694	699#									
ASKLA	675#	684											
ASKSER	662#	672	697										
ASKVT	687#	696											
BASICH	76#	563											
BL	73#	185	428										
CHRFNE	248#	791											
CHRSFN	229#	798											
COLUMN	37	173											
DECCOR	549#	556											
ERABSA	121	136#											
ERABS$	123	137#											
ERASCA	212	217	219	224#									
ERASC$	214	225#											
ERCHR$	231	239#											
ERLENA	195	205#											
ERLEN$	197	206#											
ERPOSA	245	249	253	328#									
ERPOS$	247	251	256	321#									
ERRARG	38	116	136	158	285	224	328	392	439	463			
ERRMIX	38												
ERRNDA	80	116#											
ERRND$	82	115#											
ERRPDL	38												
ERRSYN	38	115	137	159	189	286	225	239	321	393	438	464	
ERSEGA	327	331	338	392#									
ERSEG$	329	336	341	393#									
ERSGNA	143	158#											
ERSGN$	145	159#											
ERSTRA	445	463#											
ERSTR$	447	464#											
ERTAB$	166	189#											
ERVALA	399	432	439#										
ERVAL$	401	438#											
EVAL	39	79	128	142	194	211	244	248	252	326	338	337	398	444
FAC1	37	112	124	127	133	147	159	198	228	298	318	333	355	496
	423													
FAC2	37	113	129	131	134	149	159	168	199	283	221	222	232	237
	268	262	317	334	368	487	424							
FILLCO	33	705												
FILLDO	708	709#												
FILLTB	703	759#												
FNEND	466#													
FNNAM1	604	608#												
FNNAM2	605	609#												

524

525

```
          270     272     281     282     285     288     291     303     304     312     313     316     333     334
          343     344     352     353     372     374     377     378     379     380     390     403     408     412
          416     417     418     431     433     434     439     448     491     497     458     468     550     554
START      33     738
STPRO      41     388     401
STRFN    444#     802
STRFNE   465#     803
T1         40     298     298     308     454     458
T2         40     284     300     453
T3         40
TAB8     169#     172
TABC     178     173#
TABFN    164#     781
TABFNE   198#     782
TABLE5     36     398     622     724
TABNON   176     179#
TABNUL   178     188#
TBL5EN     36     398     628
TKS        32     746     748
TMP$TC   554     756#
TPB        32     734     736
TPBEN1   532     539#
TPBFH1   507     531#
TPBUF1   531     534     535     537#
TRYCOR   551     558#
USR      474#     634
USRARE     34     758     632     658
USRCOD   481     543#     637
USRLIN   482     541#
VAL        42     428
VALCE    426#     429
VALFN    398#     799
VALFNE   448#     808
VALJ     489     437#
VALR     485     418#
VALX     427     431#
,          153     174     187     282     526     527#    528     537     538#    539     542#    755#    759
,COMMA      40     246     250     328     335
,RPAR       37      81     122     144     165     196     213     238     255     348     488     446
$KBBSZ      56     527
$$TKSZ      44     654
$TPBSZ      52     538
$ULNSP      68     542
```

```
LNKX11   V021   22-MAR-73   10:22

#BASIC,NOSMAP',DRK<BASIC,FPMP,BASICH/B10/E

LOAD MAP

TRANSFER ADDRESS: 030150
LOW LIMIT: 000000
HIGH LIMIT: 031470
**********
MODULE  BASICL
SECTION ENTRY   ADDRESS SIZE
<. ABS.>        000000  000000
        ARGB    011650
        ARRAYS  000010
        COLUMN  000034
        ERRARG  007734
        ERRMIX  011650
        ERRPOL  011120
        ERRSYN  011650
        EVAL    010616
        FAC1    000040
        FAC2    000042
        FILLCO  000110
        GETVAR  013604
        INT     014004
        LIMIT   000002
        MAKEST  011650
        MSG     013474
        NORM    013554
        NUMOUT  013746
        NUMSGN  013760
        OPRATO  011272
        POL     000004
        POSIZE  000006
        RND1    000064
        RND2    000066
        SAVCHA  011650
        SOPRAT  011650
        START   001034
        STOVAR  017244
        STPRO   011650
        TABLE5  010756
        TBL5EN  011002
        TKS     177560
        TPB     177566
        T1      000056
        T2      000060
        T3      000062
        VAL     020702
        $$TKSZ  000200
        .COMMA  000243
        .EOL    000201
        .LPAR   000255
        .RPAR   000237
<       >       000000  021426
**********
MODULE  FPMP11
```

526

527

```
SECTION ENTRY   ADDRESS SIZE
<       >       021426  004270
        AINT    022544
        ALOG    022170
        ATAN    023040
        COS     024404
        ERRFPU  021430
        EXP     023314
        IFPMP   021426
        SIN     024520
        SQRT    025530
        SADR    021434
        SOVR    022662
        SERR    025666
        SERRA   025676
        SERVEC  025716
        SINTR   022502
        SIR     023664
        SMLR    023750
        SPOLSH  025524
        SPOPR3  024334
        SPOPR4  024322
        SPOPR5  024322
        SRI     024342
        SSBR    021430
        SV20A   025524
**********
MODULE  BASICH
SECTION ENTRY   ADDRESS SIZE
<       >       025724  003544
        USRARE  025724

 RUN-TIME:  5 SECONDS
 2K CORE USED
```

```
LNKX11   V021   22-MAR-73   10:21

#BASICS,STRMAP:ORK<BASICL,FPMP,BASICH/B10/E

LOAD MAP

TRANSFER ADDRESS: 032706
LOW LIMIT: 000000
HIGH LIMIT: 034226
**********
MODULE  BASICL
SECTION ENTRY   ADDRESS SIZE
<. ABS.>        000000  000000
        ARGB    010276
        ARRAYS  000010
        COLUMN  000034
        ERRARG  010326
        ERRMIX  013026
        ERRPOL  011776
        ERRSYN  012554
        EVAL    011402
        FAC1    000040
        FAC2    000042
        FILLCO  000104
        GETVAR  015410
        INT     015620
        LIMIT   000002
        MAKEST  017264
        MSG     017530
        NORM    017610
        NUMOUT  020002
        NUMSGN  020014
        OPRATO  012176
        POL     000004
        POSIEE  000006
        RND1    000064
        RND2    000066
        SAVCHA  020646
        SOPRAT  012730
        START   001102
        STOVAR  021332
        STPRO   013176
        TABLE5  011562
        TBL5EN  011626
        TKS     177560
        TPB     177566
        T1      000056
        T2      000060
        T3      000062
        VAL     023440
        $STKSZ  000200
        .COMMA  000243
        .EOL    000201
        .LPAR   000255
        .RPAR   000237
<       >       000000  024164
**********
MODULE  FPMP11
```

528

534

```
SECTION ENTRY   ADDRESS SIZE
<       >       024164  004270
        AINT    025302
        ALOG    024726
        ATAN    027570
        COS     027222
        ERRFPU  024166
        EXP     026052
        IFPMP   024164
        SIN     027256
        SQRT    030266
        $AOR    024172
        $OVR    025420
        $ERR    030424
        $ERRA   030434
        $ERVEC  030434
        $INTR   025320
        $IR     026422
        $MLR    026506
        $POLSH  030262
        $POPR3  027072
        $POPR4  027066
        $POPR5  027060
        $RI     027100
        $$BR    024166
        $V20A   030262
**********
MODULE  BASICH
SECTION ENTRY   ADDRESS SIZE
<       >       030462  003544
        USRARE  030462

 RUN-TIME:  5 SECONDS
 2K CORE USED
```

```
PERPAR -- PERIPHERAL SUPPORT PA RT-11 MACRO VM02-09   16-OCT-74 01:51:42 PAGE 1

1               .TITLE PERPAR -- PERIPHERAL SUPPORT PACKAGE PARAMETER MODULE.
2               ;
3               ; DEC-11-L8PAA-A-LA     BASIC KERNEL V02-01
4               ;
5               ; COPYRIGHT (C) 1974
6               ;
7               ; DIGITAL EQUIPMENT CORPORATION
8               ; MAYNARD, MASSACHUSETTS 01754
9               ;
10              ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
11              ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
12              ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
13              ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
14              ; MAY APPEAR IN THIS DOCUMENT.
15              ;
16              ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
17              ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
18              ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
19              ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
20              ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
21              ;
22              ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
23              ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
24              ; WHICH IS NOT SUPPLIED BY DEC.
25
26              ; THE CONDITIONALS CONTAINED IN THIS MODULE AFFECT THE ASSEMBLY
27              ; OF THE FUNCTION TABLE MODULE "FTBL.MAC".
28              ; TO OBTAIN THE DESIRED CONDITIONAL DEFINITION(S),
29              ; REMOVE (USING AN EDITOR) THE
30              ; SEMI-COLON APPEARING BEFORE THE CONDITIONAL.
31              ;$DISK=0                 ;DEFINE FOR RT-11
32                      .IFNDF $DISK
33      000000  $STRNG=0                ;DO NOT DEFINE FOR PTS BASIC WITHOUT
34                                      ;STRINGS,- DEFINED FOR PTS V01 WITH STRINGS
35                      .ENDC           ;
36
37      000000  $LPS=0                  ;DEFINE FOR LPS
38                      .IFDF $LPS
39              ;$V=0                   ;DEFINE FOR LPS WITH VECTORS STARTING
40              ;                       ;    AT 300,  DEFAULT SETTING IS VECTORS AT
41              ;                       ;    340, SET $V = ANY OTHER DISPLACEMENT IF
42              ;                       ;       VECTORS START AT DISPLACEMENTS
43              ;                       ;       OTHER THAN 0 OR 40 FROM
44              ;                       ;       VECTOR 300
45
46      000000  $ADC=0                  ;INCLUDE A/D ROUTINES.
47      000000  $CLK=0                  ;INCLUDE CLOCK ROUTINES.
48      000000  $DIO=0                  ;INCLUDE DIGITAL IO ROUTINES
49      000000  $DIS=0                  ;INCLUDE DISPLAY ROUTINES.
50                      .ENDC   ;$LPS
51              ;
52              ;
53      000000  $VT11=0                 ;FOR GT40 (GT44)
54              ;
55              ;
56
57
```

```
                        .IFDF   $VT11
000000 $CLOCK=0                         ;FOR SYSTEM CLOCK (KW11L)
                        .ENDC

                        .EOT
                .TITLE PERVEC VECTOR DEFINITION MODULE FOR BASIC SUPPORT PACKAGES.
                ;
                ; DEC-11-LBPVA-A-LA     BASIC KERNEL V02-01
                ;
                ; COPYRIGHT (C) 1974
                ;
                ; DIGITAL EQUIPMENT CORPORATION
                ; MAYNARD, MASSACHUSETTS 01754
                ;
                ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
                ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
                ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
                ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
                ; MAY APPEAR IN THIS DOCUMENT.
                ;
                ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
                ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
                ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
                ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
                ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
                ;
                ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
                ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
                ; WHICH IS NOT SUPPLIED BY DEC.
```



```
                ; THIS MODULE DEFINES THE HARDWARE ADRESSES USED BY
                ; SUCH HARDWARE AS THE "LPS", THE "VT11"(GT40) AND THE "LV11",
                ; IF THE VECTORS FOR THESE DEVICES SHOULD CHANGE
                ; THIS MODULE MUST BE EDITED TO REFLECT THE CHANGE.

                        .IFDF $LPS
                        .IFNDF  $V
                        $V=40
                        .ENDC
                        .GLOBL LPSAD,LPSADB,LPSDR,LPSDMA
                        .GLOBL LPSCKS,LPSPB,LPSDRS,LPSDIB
                        .GLOBL LPSDOR
                        .GLOBL LPDISS,LPDISX,LPDISY
                        .GLOBL CKLIVA,CKLIP,DRSIVA,DRSIP,LPSIVA,LPSIP

                ; DEVICE EQUATES:

170400 LPSAD    =       170400  ;LPS A/D STATUS REG.
170402 LPSADB   =       170402  ;LPS A/D BUFFER LED REG.
170404 LPSCKS   =       170404  ;LPS CLOCK STATUS REG.
170406 LPSPB    =       170406  ;LPS CLOCK BUFFER PRESET REG.
170410 LPSDR    =       170410  ;LPS DIGITAL I/O STATUS REG.
170410 LPSDRS   =       LPSDR
170412 LPSDIB   =       170412  ;LPS DIGITAL INPUT REG.
170414 LPSDOR   =       170414  ;LPS DIGITAL OUTPUT REG.
170416 LPDISS   =       170416  ;LPS DISPLAY STATUS REG.
170420 LPDISX   =       170420  ;LPS DISPLAY REG. X
170422 LPDISY   =       170422  ;LPS DISPLAY REG. Y
170436 LPSDMA   =       170436  ;LPS DMA REGG.


                ; INTERRUPT VECTOR PAIRS:

000344 CKLIVA   =       304+$V  ;ADR. OF CLOCK INTERRUPT VECTOR
000346 CKLIP    =       306+$V  ;ADR. OF CLOCK INT.     PRIORITY

000350 DRSIVA   =       310+$V  ;ADR. OF DRS INT. VECTOR
000352 DRSIP    =       312+$V  ;ADR. OF DRS INT. PRIORITY.

000340 LPSIVA   =       300+$V  ;ADR. OF THE A/D INT. VECTOR.
000342 LPSIP    =       302+$V  ;ADR. OF THE INT.PRIORITY.

                        .ENDC   ;$LPS
                        .IFDF   $VT11   ;GT40
                        .GLOBL  DPC,DSR,DISX,DISY,GTVECT

172000 DPC      =       172000  ;VT11 DISPLAY PC
172002 DSR      =       DPC+2   ;VT11 DISPLAY STATUS REG
172004 DISX     =       DSR+2   ;VT11 X STATUS REG
172006 DISY     =       DISX+2  ;VT11 Y STATUS REG
000320 GTVECT   =       320     ;ADR. OF VT11 [GT40 (GT44)] INTERRUPT
                                ;VECTOR LIST. REDEFINING GTVECT
```

```
58                                      ;REDEFINES THE ENTIRE SET
59                                      ;OF DISPLAY PROCESSOR INT. VECTORS,
60              ;GTVECT:                ;DISPLAY STOP VECTOR
61              ;GTVECT+4:              ;LIGHT PEN HIT VECTOR
62              ;GTVECT+10:             ;DISPLAY TIME OUT VECTOR
63                      .ENDC   ;SVT11
64
65
66      000001'         .END
```


SYMBOL TABLE

```
CKLIP = 000346 G        CKLIVA= 000344 G        DISX  = 172004 G
DISY  = 172006 G        DPC   = 172000 G        DRSIP = 000352 G
DRSIVA= 000350 G        DSR   = 172002 G        GTVECT= 000320 G
LPDISS= 170416 G        LPDISX= 170420 G        LPDISY= 170422 G
LPSAD = 170400 G        LPSADB= 170402 G        LPSCKS= 170404 G
LPSDIB= 170412 G        LPSDMA= 170436 G        LPSDOR= 170414 G
LPSDR = 170410 G        LPSDRS= 170410 G        LPSIP = 000342 G
LPSIVA= 000340 G        LPSPB = 170406 G        $ADC  = 000000
$CLK  = 000000          $CLOCK= 000000          $DIO  = 000000
$DIS  = 000000          $LPS  = 000000          $STRNG= 000000
$V    = 000040          $VT11 = 000000
. ABS.  000000     000
        000000     001
ERRORS DETECTED: 0
FREE CORE: 18189. WORDS
```

```
PERVEC,GTL,LP:=PERPAR,GTL,PERVEC
```

```
                ;TITLE PERPAR -- PERIPHERAL SUPPORT PACKAGE PARAMETER MODULE.
                ;
                ; DEC-11-L8PAA-A-LA     BASIC KERNEL V02-01
                ;
                ; COPYRIGHT (C) 1974
                ;
                ; DIGITAL EQUIPMENT CORPORATION
                ; MAYNARD, MASSACHUSETTS 01754
                ;
                ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
                ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
                ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
                ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
                ; MAY APPEAR IN THIS DOCUMENT.
                ;
                ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
                ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
                ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
                ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
                ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
                ;
                ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
                ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
                ; WHICH IS NOT SUPPLIED BY DEC.

                ; THE CONDITIONALS CONTAINED IN THIS MODULE AFFECT THE ASSEMBLY
                ; OF THE FUNCTION TABLE MODULE "FTBL.MAC",
                ; TO OBTAIN THE DESIRED CONDITIONAL DEFINITION(S),
                ; REMOVE (USING AN EDITOR) THE
                ; SEMI-COLON APPEARING BEFORE THE CONDITIONAL.
                ;$DISK=0                ;DEFINE FOR RT-11
                        .IFNDF $DISK
000000 $STRNG=0                         ;DO NOT DEFINE FOR PTS BASIC WITHOUT
                                        ;STRINGS,- DEFINED FOR PTS V01 WITH STRINGS
                        .ENDC           ;

000000 $LPS=0                           ;DEFINE FOR LPS
                        .IFDF $LPS
                ;$V=0                   ;DEFINE FOR LPS WITH VECTORS STARTING
                ;                       ;    AT 300,  DEFAULT SETTING IS VECTORS AT
                ;                       ;    340, SET $V = ANY OTHER DISPLACEMENT IF
                ;                       ;       VECTORS START AT DISPLACEMENTS
                ;                       ;       OTHER THAN 0 OR 40 FROM
                ;                       ;       VECTOR 300

000000 $ADC=0                           ;INCLUDE A/D ROUTINES.
000000 $CLK=0                           ;INCLUDE CLOCK ROUTINES.
000000 $DIO=0                           ;INCLUDE DIGITAL IO ROUTINES
000000 $DIS=0                           ;INCLUDE DISPLAY ROUTINES.
                        .ENDC   ;$LPS
                ;
                ;
000000 $VT11=0                          ;FOR GT40 (GT44)
                ;
                ;
```

3


```
                        .IFDF   $VT11
000000 $CLOCK=0                         ;FOR SYSTEM CLOCK (KW11L)
                        .ENDC

                        .EOT
                .TITLE PTSINT -- PTS INTERFACE MODULE FOR SUPPORT PACKAGES.
                ;
                ; DEC-11-L8PIA-A-LA     BASIC KERNEL V02-01
                ;
                ; COPYRIGHT (C) 1974
                ;
                ; DIGITAL EQUIPMENT CORPORATION
                ; MAYNARD, MASSACHUSETTS 01754
                ;
                ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
                ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
                ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
                ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
                ; MAY APPEAR IN THIS DOCUMENT.
                ;
                ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
                ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
                ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
                ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
                ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
                ;
                ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
                ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
                ; WHICH IS NOT SUPPLIED BY DEC.
                ;
                        .GLOBL FTBL,BOMB
000000                  .ASECT
000034          .=34
                        .IFNDF $DISK            ;BOMB VARIES IF ONE INCLUDES
                        .IFDF  $STRNG           ;STRINGS
010320  BOMB=010320                             ;ENTRY WITH STRINGS
                        .ENDC                   ;$STRNG IS DEFINED
                        .IFNDF $STRNG           ;NO STRINGS
                BOMB=007742                     ;ENTRY WITHOUT STRINGS
                        .ENDC                   ;$STRNG NOT DEFINED
                        .ENDC                   ;$DISK
0034 010320             .WORD   BOMB,0          ;SET UP THE TRAP ERROR VECTOR.
0036 000000
000040                  .ASECT
000046          .=46
0046 000000G            .WORD   FTBL            ;THE FUNCTION TABLE STARTS HERE FOR PTS.
                        .IFDF $LPS
                        .IFDF $DIS
                        .GLOBL DISPLY
0050 000167             JMP DISPLY
     000000G
                        .ENDC   ;$DIS
                        .ENDC   ;$LPS
000001'                 .END
```

```
PTSINT -- PTS INTERFACE MODULE  RT-11 MACRO VM02-09   16-OCT-74 01:52:29 PAGE 1+
SYMBOL TABLE

$OMB   = 010320 G        DISPLY= ****** G        FTBL  = ****** G
$ADC   = 000000          $CLK  = 000000          $CLOCK= 000000
$DIO   = 000000          $DIS  = 000000          $LPS  = 000000
$STRNG= 000000           $VT11 = 000000
. ABS.  000054     000
        000000     001
ERRORS DETECTED: 0
FREE CORE: 18268. WORDS

PTSINT,GTL,LP:=PERPAR,GTL,PTSINT
```

```
PERPAR -- PERIPHERAL SUPPORT PA RT-11 MACRO VM02-09   16-OCT-74 01:53:02 PAGE 1


1               .TITLE PERPAR -- PERIPHERAL SUPPORT PACKAGE PARAMETER MODULE.
2               ;
3               ; DEC-11-LBPAA-A-LA     BASIC KERNEL V02-01
4               ;
5               ; COPYRIGHT (C) 1974
6               ;
7               ; DIGITAL EQUIPMENT CORPORATION
8               ; MAYNARD, MASSACHUSETTS 01754
9               ;
10              ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
11              ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
12              ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
13              ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
14              ; MAY APPEAR IN THIS DOCUMENT.
15              ;
16              ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
17              ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
18              ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
19              ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
20              ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
21              ;
22              ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
23              ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
24              ; WHICH IS NOT SUPPLIED BY DEC.
25
26              ; THE CONDITIONALS CONTAINED IN THIS MODULE AFFECT THE ASSEMBLY
27              ; OF THE FUNCTION TABLE MODULE "FTBL.MAC",
28              ; TO OBTAIN THE DESIRED CONDITIONAL DEFINITION(S),
29              ; REMOVE (USING AN EDITOR) THE
30              ; SEMI-COLON APPEARING BEFORE THE CONDITIONAL.
31              ;$DISK=0                 ;DEFINE FOR RT-11
32                      .IFNDF $DISK
33      000000 $STRNG=0                 ;DO NOT DEFINE FOR PTS BASIC WITHOUT
34                                      ;STRINGS,- DEFINED FOR PTS V01 WITH STRINGS
35                      .ENDC           ;
36
37      000000 $LPS=0                   ;DEFINE FOR LPS
38                      .IFDF $LPS
39              ;$V=0                   ;DEFINE FOR LPS WITH VECTORS STARTING
40              ;                       ;    AT 300.  DEFAULT SETTING IS VECTORS AT
41              ;                       ;    340. SET $V = ANY OTHER DISPLACEMENT IF
42              ;                       ;       VECTORS START AT DISPLACEMENTS
43              ;                       ;       OTHER THAN 0 OR 40 FROM
44              ;                       ;       VECTOR 300
45
46      000000 $ADC=0                   ;INCLUDE A/D ROUTINES.
47      000000 $CLK=0                   ;INCLUDE CLOCK ROUTINES.
48      000000 $DIO=0                   ;INCLUDE DIGITAL IO ROUTINES
49      000000 $DIS=0                   ;INCLUDE DISPLAY ROUTINES.
50                      .ENDC   ;$LPS
51              ;
52              ;
53      000000 $VT11=0                  ;FOR GT40 (GT44)
54              ;
55              ;
56
57
```

```
                        .IFDF   $VT11
000000 $CLOCK=0                                 ;FOR SYSTEM CLOCK (KW11L)
                        .ENDC

                        .EOT
                .TITLE  FTBL--BASIC FUNCTION TABLE MODULE

                ;       DEC-11-LBFTA-A-LA       BASIC KERNEL V02-01
                ;
                ; COPYRIGHT (C) 1973,1974
                ;
                ; DIGITAL EQUIPMENT CORPORATION
                ; MAYNARD, MASSACHUSETTS 01754
                ;
                ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
                ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
                ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
                ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
                ; MAY APPEAR IN THIS DOCUMENT.
                ;
                ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
                ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
                ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
                ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
                ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
                ;
                ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
                ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
                ; WHICH IS NOT SUPPLIED BY DEC.


                        .GLOBL FTBL

                ;       MUST INTERFACE PROPERLY WHEN LPS IS ALSO USED
                ;


000000'         .CSECT
00000   FTBL:
                        .IFDF $VT11     ;GT40
                        .GLOBL  AGET,APNT,APUT,CONT,ERAS,ESUB,FPUT
                        .GLOBL  FIGR,FPUT,INIT,LPEN,RDOT
                        .GLOBL  SCAL,STAT,DSTP,SUBP,TEXT,TRAK
                        .GLOBL  VECT,XGRA,YGRA,FIX,FREE
                        .GLOBL  NOSC,ON,OFF,SAVE
                        .IFDF   $DISK
                        .GLOBL  RSTR
                        .ENDC   ;$DISK
                        .IFDF   $CLOCK
                        .GLOBL  TIME,TIMR
                        .ENDC   ;$CLOCK

0000    123             .ASCII  /SCAL/
0001    103
0002    101
0003    114
0004 000000G            .WORD   SCAL
```

5


```
0006    126             .ASCII  /VECT/
0007    105
0010    103
0011    124
0012 000000G            .WORD   VECT
0014    122             .ASCII  /RDOT/
0015    104
0016    117
0017    124
0020 000000G            .WORD   RDOT
0022    101             .ASCII  /APNT/
0023    120
0024    116
0025    124
0026 000000G            .WORD   APNT
                        .IFDF   $CLOCK
0030    124             .ASCII  /TIME/
0031    111
0032    115
0033    105
0034 000000G            .WORD   TIME
0036    124             .ASCII  /TIMR/
0037    111
0040    115
0041    122
0042 000000G            .WORD   TIMR
                        .ENDC   ;$CLOCK
0044    123             .ASCII  /STAT/
0045    124
0046    101
0047    124
0050 000000G            .WORD   STAT
0052    124             .ASCII  /TEXT/
0053    105
0054    130
0055    124
0056 000000G            .WORD   TEXT
0060    123             .ASCII  /SUBP/
0061    125
0062    102
0063    120
0064 000000G            .WORD   SUBP
0066    105             .ASCII  /ESUB/
0067    123
0070    125
0071    102
0072 000000G            .WORD   ESUB
0074    114             .ASCII  /LPEN/
0075    120
0076    105
0077    116
0100 000000G            .WORD   LPEN
0102    116             .ASCII  /NOSC/
0103    117
0104    123
0105    103
0106 000000G            .WORD   NOSC
```

```
137 0110    104          .ASCII  /DON/
    0111    117
    0112    116
138 0113    000          .BYTE   0
139 0114 000000G         .WORD    ON
140 0116    117          .ASCII  /ON/
    0117    116
141 0120 000000          .WORD   0
142 0122 000000G         .WORD   ON       ; SECOND NAME
143 0124    117          .ASCII  /OFF/
    0125    106
    0126    106
144 0127    000          .BYTE   0
145 0130 000000G         .WORD    OFF
146 0132    124          .ASCII  /TRAK/
    0133    122
    0134    101
    0135    113
147 0136 000000G         .WORD    TRAK
148 0140    105          .ASCII  /ERAS/
    0141    122
    0142    101
    0143    123
149 0144 000000G         .WORD    ERAS
150 0146    111          .ASCII  /INIT/
    0147    116
    0150    111
    0151    124
151 0152 000000G         .WORD    INIT
152 0154    104          .ASCII  /DSTP/
    0155    123
    0156    124
    0157    120
153 0160 000000G         .WORD    DSTP
154 0162    123          .ASCII  /STOP/  ;SECOND NAME
    0163    124
    0164    117
    0165    120
155 0166 000000G         .WORD   DSTP
156 0170    104          .ASCII  /DCNT/
    0171    103
    0172    116
    0173    124
157 0174 000000G         .WORD    CONT
158 0176    103          .ASCII  /CONT/  ;SECOND NAME
    0177    117
    0200    116
    0201    124
159 0202 000000G         .WORD   CONT
160 0204    130          .ASCII  /XGRA/
    0205    107
    0206    122
    0207    101
161 0210 000000G         .WORD    XGRA
162 0212    131          .ASCII  /YGRA/
    0213    107
    0214    122
```



```
    0215    101
163 0216 000000G         .WORD    YGRA
164 0220    101          .ASCII  /AGET/
    0221    107
    0222    105
    0223    124
165 0224 000000G         .WORD    AGET
166 0226    104          .ASCII  /DFIX/
    0227    106
    0230    111
    0231    130
167 0232 000000G         .WORD   FIX
168 0234    106          .ASCII  /FIX/   ;SECOND NAME
    0235    111
    0236    130
169 0237    000          .BYTE   0
170 0240 000000G         .WORD    FIX
171 0242    106          .ASCII  /FREE/
    0243    122
    0244    105
    0245    105
172 0246 000000G         .WORD    FREE
173 0250    101          .ASCII  /APUT/
    0251    120
    0252    125
    0253    124
174 0254 000000G         .WORD    APUT
175 0256    106          .ASCII  /FIGR/
    0257    111
    0260    107
    0261    122
176 0262 000000G         .WORD    FIGR
177 0264    106          .ASCII  /FPUT/
    0265    120
    0266    125
    0267    124
178 0270 000000G         .WORD    FPUT
179 0272    104          .ASCII  /DSAV/
    0273    123
    0274    101
    0275    126
180 0276 000000G         .WORD    SAVE
181 0300    123          .ASCII  /SAVE/  ;SECOND NAME
    0301    101
    0302    126
    0303    105
182 0304 000000G         .WORD   SAVE
183                      .IFDF   $DISK
184                      .ASCII  /RSTR/
185                      .WORD    RSTR
186                      .ENDC   ;$DISK
187                      .ENDC   ;$VT11
188
189
190
191
192                      .IFDF $LPS
```

```
193
194                     .GLOBL USE,RDB,ACC
195                     ;
196                             .IFDF $ADC
197                             .GLOBL ADC,RTS,LED
198                             .ENDC   ;$ADC
199                     ;
200                             .IFDF $CLK
201                             .GLOBL SETR,SETC,HIST,WAIT
202                             .ENDC   ;$CLK
203                     ;
204                             .IFDF $DIO
205                             .GLOBL DIR,DOR,DRS
206                             .GLOBL REL
207                             .ENDC ;$DIO
208                     ;
209                             .IFDF $DIS
210                             .GLOBL CLRD,PUTD,DIS,FSH,DXY,DISPLY
211                             .ENDC ;$DIS
212
213
214
215 0306    125                 .ASCII  /USE/
    0307    123
    0310    105
216 0311    000                 .BYTE   0
217 0312 000000G                .WORD   USE
218                     ;
219 0314    122                 .ASCII  /RDB/
    0315    104
    0316    102
220 0317    000                 .BYTE   0
221 0320 000000G                .WORD   RDB
222                     ;
223 0322    101                 .ASCII  /ACC/
    0323    103
    0324    103
224 0325    000                 .BYTE   0
225 0326 000000G                .WORD   ACC
226                     ;
227                     ;
228                     ;
229                             .IFDF   $ADC
230                     ;
231 0330    101                 .ASCII  /ADC/
    0331    104
    0332    103
232 0333    000                 .BYTE   0
233 0334 000000G                .WORD   ADC
234                     ;
235 0336    122                 .ASCII  /RTS/
    0337    124
    0340    123
236 0341    000                 .BYTE   0
237 0342 000000G                .WORD   RTS
238                     ;
239 0344    114                 .ASCII  /LED/
```

FTBL--BASIC FUNCTION TABLE MODU RT-11 MACRO VM02-09 16-OCT-74 01:53:02 PAGE 1+

```
    0345    105
    0346    104
240 0347    000                 .BYTE   0
241 0350 000000G                .WORD   LED
242                     ;
243                             .ENDC   ;$ADC
244                     ;
245                     ;
246                     ;
247                             .IFDF $CLK
248                     ;
249 0352    123                 .ASCII  /SETR/
    0353    105
    0354    124
    0355    122
250 0356 000000G                .WORD   SETR
251                     ;
252 0360    123                 .ASCII  /SETC/
    0361    105
    0362    124
    0363    103
253 0364 000000G                .WORD   SETC
254                     ;
255 0366    110                 .ASCII  /HIST/
    0367    111
    0370    123
    0371    124
256 0372 000000G                .WORD   HIST
257                     ;
258 0374    127                 .ASCII  /WAIT/
    0375    101
    0376    111
    0377    124
259 0400 000000G                .WORD   WAIT
260                     ;
261                             .ENDC ;$CLK
262                     ;
263                     ;
264                     ;
265                             .IFDF   $DIO
266                     ;
267 0402    104                 .ASCII  /DIR/
    0403    111
    0404    122
268 0405    000                 .BYTE   0
269 0406 000000G                .WORD   DIR
270                     ;
271 0410    104                 .ASCII  /DOR/
    0411    117
    0412    122
272 0413    000                 .BYTE   0
273 0414 000000G                .WORD   DOR
274                     ;
275 0416    104                 .ASCII  /DRS/
    0417    122
    0420    123
276 0421    000                 .BYTE   0
```

```
277 0422 000000G        .WORD   DRS
278                     ;
279 0424     122        .ASCII  /REL/
    0425     105
    0426     114
280 0427     000        .BYTE   0
281 0430 000000G        .WORD   REL
282                     ;
283                     .ENDC   ;$DIO
284                     ;
285                     ;
286                     ;
287                     .IFDF   $DIS
288                     ;
289 0432     103        .ASCII  /CLRD/
    0433     114
    0434     122
    0435     104
290 0436 000000G        .WORD   CLRD
291                     ;
292 0440     120        .ASCII  /PUTD/
    0441     125
    0442     124
    0443     104
293 0444 000000G        .WORD   PUTD
294                     ;
295 0446     104        .ASCII  /DIS/
    0447     111
    0450     123
296 0451     000        .BYTE   0
297 0452 000000G        .WORD   DIS
298                     ;
299 0454     106        .ASCII  /FSH/
    0455     123
    0456     110
300 0457     000        .BYTE   0
301 0460 000000G        .WORD   FSH
302                     ;
303 0462     104        .ASCII  /DXY/
    0463     130
    0464     131
304 0465     000        .BYTE   0
305 0466 000000G        .WORD   DXY
306                     ;
307                     .ENDC   ;$DIS
308                     .ENDC   ;$LPS
309 0470 000000         .WORD   0       ;END OF THE BASIC FUNCTION TABLE.
310      000001'        .END
```

FTBL--BASIC FUNCTION TABLE MODU RT-11 MACRO VM02-09 16-OCT-74 01:53:02 PAGE 1+

SYMBOL TABLE

```
ACC    = ****** G        ADC    = ****** G        AGET   = ****** G
APNT   = ****** G        APUT   = ****** G        CLRD   = ****** G
CONT   = ****** G        DIR    = ****** G        DIS    = ****** G
DISPLY= ****** G         DOR    = ****** G        DRS    = ****** G
DSTP   = ****** G        DXY    = ****** G        ERAS   = ****** G
ESUB   = ****** G        FIGR   = ****** G        FIX    = ****** G
FPUT   = ****** G        FREE   = ****** G        FSH    = ****** G
FTBL    000000RG         HIST   = ****** G        INIT   = ****** G
LED    = ****** G        LPEN   = ****** G        NOSC   = ****** G
OFF    = ****** G        ON     = ****** G        PUTD   = ****** G
ROB    = ****** G        RDOT   = ****** G        REL    = ****** G
RTS    = ****** G        SAVE   = ****** G        SCAL   = ****** G
SETC   = ****** G        SETR   = ****** G        STAT   = ****** G
SUBP   = ****** G        TEXT   = ****** G        TIME   = ****** G
TIMR   = ****** G        TRAK   = ****** G        USE    = ****** G
VECT   = ****** G        WAIT   = ****** G        XGRA   = ****** G
YGRA   = ****** G        $ADC   = 000000          $CLK   = 000000
$CLOCK= 000000           $DIO   = 000000          $DIS   = 000000
$LPS   = 000000          $STRNG= 000000           $VT11  = 000000

. ABS.  000000     000
        000472     001
ERRORS DETECTED: 0
FREE CORE: 18081. WORDS

FTBL.GTL,LP:=PERPAR.GTL,FTBL
```

```
1               .TITLE  LPS0  V01-01
2               ;.SBTTL LPS MAIN KERNEL MODULE #0
3               ;
4               ; DEC-11-LBEXA-B-LA1    BASIC KERNEL V02-01
5               ;
6               ; COPYRIGHT (C) 1973,1974
7               ;
8               ; DIGITAL EQUIPMENT CORPORATION
9               ; MAYNARD, MASSACHUSETTS 01754
10              ;
11              ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
12              ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
13              ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
14              ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
15              ; MAY APPEAR IN THIS DOCUMENT.
16              ;
17              ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
18              ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
19              ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
20              ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
21              ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
22              ;
23              ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
24              ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
25              ; WHICH IS NOT SUPPLIED BY DEC.
26              ;
27              ; WRITTEN BY:  RICK HULLY        FEB,, 1973
28              ;
```



```
1               ;
2               ; THE FOLLOWING COMMON SUPPORT ROUTINES EXIST IN THIS MODULE:
3               ;
4               ;       POLENT - ENTER POISH MODE OPERATION
5               ;       CKCMGN - POLISH ROUTINE TO CHECK NEXT TOKEN EQEAL TO
6               ;                ,COMMA AND THEN FETCH THE NEXT FLOATING
7               ;                POINT NUMERIC FROM THE INPUT STRING.
8               ;       GETNUM - POLISH ROUTINE TO FETCH A FLOATING POINT
9               ;                NUMERIC FROM INPUT STRING.
10              ;       SAVREG - POLISH ROUTINE TO SAVE R0, R2 AND R3
11              ;                ON THE STACK.
12              ;       SAVFAC - POLISH ROUTINE TO SAVE THE FAC ON THE
13              ;                STACK.
14              ;       SAVINT - POLISH ROUTINE TO SAVE ONLY THE INTEGER
15              ;                PORTION OF THE FAC ON THE STACK.
16              ;       RESREG - POLISH ROUTINE TO RESTORE R3, R2 AND
17              ;                R0 FROM THE STACK.
18              ;       INTST  - POLISH ROUTINE TO INTEGERIZE FAC AND CHECK
19              ;                ITS RANGE AGAINST A GIVEN VALUE.
20              ;       FLOAT  - ROUTINE TO FLOAT A 16 BIT UNSIGNED INTEGER
21              ;       ENTERP - ROUTINE TO ENTER POLISH MODE FOR ONE ROUTINE
22              ;                ONLY AND THEN EXIT BACK TO CALLER.
23              ;       BUFRES - POLISH ROUTINE TO RESET BUFFER POINTERS TO A
24              ;                CIRCULAR BUFFER.
25              ;       GETV   - POLISH ROUTINE TO CALCULATE ADDRESS OF
26              ;                VARIABLE SPECIFIED IN THE INPUT LINE.
27              ;       GETBUF - POLISH ROUTINE TO FETCH NEXT VARIABLE FROM
28              ;                INPUT LINE, TEST FOR AN ARRAY NAME AND
29              ;                CALCULATE THE POSITION IN THE ARRAY FROM
30              ;                SUBSCRIPTS WHICH MAY BE GIVEN.
31              ;       CKBUF  - POLISH ROUTINE TO CHECK IF ARRAY ADDRESS
32              ;                SPECIFIED IN VARSAV(R5) HAS BEEN DEFINED BY
33              ;                A "USE" COMMAND.
34              ;       GETDAT - ROUTINE TO FETCH DATA INTO USER'S BUFFER
35              ;                AND TEST FOR NON-EXISTANT DATA.
36              ;       STODAT - ROUTINE TO STORE DATA INTO USER'S BUFFER
37              ;                AND TEST FOR DATA OVERRUN.
38              ;       GETADD - ROUTINE TO CALCULATE ARRAY ADDRESS FROM
39              ;                ARRAY NAME GIVEN BY CALLER.
40              ;       CKLPAR - POLISH ROUTINE TO CHECK NEXT TOKEN
41              ;                EQUAL TO A LEFT PARENS.  A TEST
42              ;                IS THEN MADE TO INSURE THAT WE STILL HAVE
43              ;                ROOM LEFT TO WORK WITH.
44              ;       CKCOMA - POLISH ROUTINE TO CHECK NEXT TOKEN
45              ;                EQUAL TO A ,COMMA.
46              ;       STORXT - COMMON EXIT ROUTINE TO STORE FAC IN VARIABLE
47              ;                SET UP IN VARSAV(R5), CHECK LAST TOKEN EQUAL
48              ;                TO A RIGHT PARENS (ALTERNATE ENTRY POINT -
49              ;                NAMED CKRPAR), AND RETURN TO THE BASIC
50              ;                INTERPRETER.
51              ;       CKRPAR - ROUTINE TO CHECK LAST TOKEN EQUAL TO A
52              ;                RIGHT PARENS AND RETURN TO THE BASIC
53              ;                INTERPRETER.
54              ;       CONBCD - ROUTINE TO CONVERT BCD TO BINARY
55              ;
```

```
1                       ;       .SBTTL  PROGRAM GLOBALS
2                       ;
3                       ; GLOBALS REQUIRED FOR COMMUNICATION WITH THE BASIC
4                       ; INTREPRETER.
5                       ;
6                               .GLOBL  ERRPDL,ERRSYN,ERRARG
7                               .GLOBL  EVAL,GETVAR,STOVAR
8                               .GLOBL  INT,,COMMA,,RPAR
9                               .GLOBL  NUMSGN,,LPAR
10                      ;
11                      ; GLOBALS REQUIRED FOR COMMUNICATIONS WITH THE VARIOUS
12                      ; LPS MODULES.
13                      ;
14                              .GLOBL  TABLE,NARRAY,FLOAT
15                              .GLOBL  POLENT,GETADD,GETNUM
16                              .GLOBL  CKCMGN,INTST,GETV
17                              .GLOBL  GETBUF,REGSAV,ERNOR
18                              .GLOBL  STORXT,CKRPAR,ERBUF
19                              .GLOBL  CKCOMA,ENTERP,STODAT
20                              .GLOBL  GETDAT,RTSON,DRSON
21                              .GLOBL  SAVER2,SAVFAC,RESTR2
22                              .GLOBL  RESTOR,HISTON,BCDON
23                              .GLOBL  CONBCD,BUFRES,DRSNPT
24                              .GLOBL  SAVINT,DRSBUF
25                      ;
26                      ; GLOBALS REQUIRED FOR COMMAND PROCESSORS
27                      ;
28                              .GLOBL  USE,RDB,ACC
29                      ;
30                      ; GLOBALS FOR DEVICE ADDRESSES
31                      ;
32                              .GLOBL  LPSAD,LPSDR
33                      ;
```



```
1                       ;       .SBTTL  REGISTER ASSIGNMENTS AND EQUATES
2                       ;
3               000000 R0       =       %0
4               000001 R1       =       %1
5               000002 R2       =       %2
6               000003 R3       =       %3
7               000004 R4       =       %4
8               000005 R5       =       %5
9               000006 SP       =       %6
10              000007 PC       =       %7
11                      ;
12                      ; GENERAL EQUATES
13                      ;
14              177776 PS       =       -2              ;PS = PROCESSOR STATUS WORD
15              000207 RETURN   =       207             ;207 = RTS PC
16              000134 POLRET   =       134             ;134 = JMP @(R4)+
17                      ;
18                      ; BASIC USER'S EQUATES
19                      ;
20              000022 VARSAV   =       22              ;VAR SAVE FOR ASSIGNMENT
21              000024 SS1SAV   =       24              ;$$1 SAVE FOR ASSIGNMENT
22              000026 SS2SAV   =       26              ;$$2 SAVE FOR ASSIGNMENT
23              000040 FAC1     =       40              ;HIGH ORDER FLOATING VALUE
24              000042 FAC2     =       42              ;LOW  ORDER FLOATING VALUE
25              000062 T3       =       62              ;SHORT TERM TEMPORARY
26                      ;
27                      ; BUFFER DESCRIPTOR EQUATES
28                      ;
29              000000 BEG      =       0               ;OFFSET TO BEGINNING OF BUFFER OINTER
30              000002 END      =       2               ;OFFSET TO END OF BUFFER POINTER
31              000004 PUT      =       4               ;OFFSET TO PUT DATA POINTER
32              000006 GET      =       6               ;OFFSET TO GET DATA POINTER
33              000010 BAD      =       10              ;OFFSET TO BAD DATA FLAG
34              000012 DEL      =       12              ;OFFSET TO DISPLAY DELTA X
35                      ;
36                      ; BUFFER DESCRIPTOR TABLE
37                      ;
38              000005 NARRAY=5                         ;CURRENT NUMBER OF ARRAY ENTRIES
39                                                      ;       SET AT 5.
40 00000 000000 TABLE:  .WORD   0,0,0,0,0,0             ;6 WORDS PER ENTRY
   00002 000000
   00004 000000
   00006 000000
   00010 000000
   00012 000000
41 00014 000000         .WORD   0,0,0,0,0,0
   00016 000000
   00020 000000
   00022 000000
   00024 000000
   00026 000000
42 00030 000000         .WORD   0,0,0,0,0,0
   00032 000000
   00034 000000
   00036 000000
   00040 000000
   00042 000000
```

```
43 00044 000000          .WORD   0,0,0,0,0,0
   00046 000000
   00050 000000
   00052 000000
   00054 000000
   00056 000000
44 00060 000000          .WORD   0,0,0,0,0,0
   00062 000000
   00064 000000
   00066 000000
   00070 000000
   00072 000000
45                       ;
46                       ; NON-REENTRANT VARIABLES
47                       ;
48 00074    000 RTSON:   .BYTE   0               ;RTS OPERATION IN PROGRESS
49 00075    000 DRSON:   .BYTE   0               ;DRS OPERATION IN PROGRESS
50 00076    000 HISTON:  .BYTE   0               ;HIST OPERATION IN PROGRESS
51 00077    000 BCDON:   .BYTE   0               ;BCD/BINARY SWITCH
52 00100 000000 DRSBUF:  .WORD   0               ;DRS COMMAND (BUFFER DESCRIPTOR
53                                               ;    ADDRESS)
54 00102 000000 DRSNPT:  .WORD   0               ;DRS COMMAND (NBR OF POINTS)
55                       ;
```



```
1                        ;       .SBTTL  "USE" COMMAND PROCESSOR
2                        ;
3                        ;*****************************************************************
4                        ;
5                        ; "USE" COMMAND PROCESSOR
6                        ;
7                        ; BASIC FORM:   CALL "USE"(BUF[,BUF2[,....[,BUFN]]])
8                        ;
9                        ; PURPOSE:  DEFINE BUFFER AREAS FOR USE WITH THE RTS, HIST,
10                       ;               DRS, RDB, CLRD, PUTD, DIS, FSH, AND DXY
11                       ;               COMMANDS.  THIS COMMAND DEFINES THE SPECIFIED
12                       ;               ARRAYS, BUF1, BUF2, ETC.,... AS CIRCULAR
13                       ;               DATA ARRAYS.  ANY NUMBER OF BUFFERS MAY BE
14                       ;               SPECIFIED, BUT ALL MUST BE GIVEN IN A SINGLE
15                       ;               "USE" STATEMENT.  CURRENTLY THE SYSTEM WILL
16                       ;               SUPPORT UP TO 5 OF THESE AREAS, HOWEVER, THIS
17                       ;               MAY BE CHANGED AT ANY TIME BY REASSEMBLY.  EACH
18                       ;               BUF DEFINED MUST HAVE PREVIOUSLY BEEN DEFINED
19                       ;               IN A "DIM" STATEMENT.  THE FOLLOWING FORM, HOWEVER,
20                       ;               IS LEGAL:
21                       ;                       10 DIM A(50),B(100),C(200)
22                       ;                       20 CALL "USE"(A,A(25),B,C)
23                       ;
24 00104                 USE:                    ;ENTRY POINT TO PROCESSOR
25 00104 004767          USE:    JSR     PC,POLENT       ;ENTRY POLISH MODE AND CHECK
         001214
26                                               ;    FOR ENOUGH AREA TO WORK
27                                               ;    WITH AND THAT FIRST TOKEN
28                                               ;    IS INDEED A LEFT PARENS.
29 00110 000112'         ,+2                     ;EXIT POLISH MODE
30 00112 012746          MOV     #NARRAY,-(SP)   ;#NARRAY'S ALLOWED.
         000005
31 00116 012746          MOV     #TABLE,-(SP)    ;WHERE DEFINITIONS WILL END UP
         000000'
32 00122 004767 USE3:    JSR     PC,GETADD       ;GET ARRAY FROM INPUT LINE AND
         001070
33                                               ;    CALCULATE ITS ARRAY ADDRESS,
34                                               ;    THEN CHECK TO MAKE SURE ITS
35                                               ;    WITHIN RANGE.
36 00126 011602 USE2:    MOV     (SP),R2         ;GET TABLE ADDRESS
37 00130 016522          MOV     VARSAV(R5),(R2)+ ;SAVE BEGINNING ADDRESS OF
         000022
38                                               ;    ARRAY IN TABLE
39 00134 010322          MOV     R3,(R2)+        ;SAVE ENDING ADDRESS OF ARRAY
40                                               ;    IN TABLE.
41 00136 016522          MOV     VARSAV(R5),(R2)+ ;RESET GET AND PUT POINTERS
         000022
42 00142 016522          MOV     VARSAV(R5),(R2)+
         000022
43 00146 005022          CLR     (R2)+           ;CLEAR BAD DATA FLAG
44 00150 005022          CLR     (R2)+           ;CLEAR DELTA X INCREMENT
45 00152 010216          MOV     R2,(SP)         ;SAVE CURRENT TABLE POSITION
46 00154 005762          TST     -12(R2)         ;WERE WE ZEROING THE REST OF
         177766
47                                               ;    THE ARRAY?
48 00160 001420          BEQ     USE4            ;YES:  CONTINUE ZEROING THEN
49 00162 022702          CMP     #TABLE+14,R2    ;CHECK IF PREVIOUSLY DEFINED ARRAY
```

```
          000014'
50 00166 001415           BEQ     USE4            ;    WAS FROM SAME DIMENSIONED
51                                                ;    ARRAY,  1ST ENTRY IS NOT
52                                                ;    CHECKED.
53 00170 026262           CMP     -26(R2),-12(R2)
         177752
         177766
54 00176 001011           BNE     USE4            ;ARRAYS NOT SAME,  CHECK IF THE
55 00200 026262           CMP     -30(R2),-14(R2) ;    LARGER ONE IS THE PREVIOUS
         177750
         177764
56                                                ;    ONE.
57 00206 101005           BHI     USE4            ;YES:  FORGET IT THEN,  USER MUST
58                                                ;    WANT TO OVERAY HIS DATA AREAS.
59 00210 016203           MOV     -14(R2),R3      ;DEFINE PREVIOUS ARRAY'S END ADDRESS
         177764
60 00214 005743           TST     -(R3)           ;    TO BE 2 LESS THAN THE START
61 00216 010362           MOV     R3,-26(R2)      ;    OF THE NEW ARRAY.
         177752
62 00222 005366 USE4:     DEC     2(SP)           ;ANY MORE SPACE AVAILABLE?
         000002
63 00226 003003           BGT     USE1            ;YES:  CHECK NEXT TOKEN FOR A
64                                                ;    ,COMMA OR RIGHT PARENS.
65 00230 022626           CMP     (SP)+,(SP)+     ;CLEAN UP STACK
66 00232 000167           JMP     CKRPAR          ;ALL DONE, CHECK FOR RIGHT
         001130
67                                                ;    PARENS AND RETURN TO
68                                                ;    THE BASIC INTREPRETER.
69 00236 122127 USE1:     CMPB    (R1)+,#.COMMA   ;NEXT TOKEN A ,COMMA?
         000000G
70 00242 001727           BEQ     USE3            ;YES:  GET NEXT ARRAY NAME
71 00244 005301           DEC     R1              ;BACK UP TO UNCLASSIFIED TOKEN
72 00246 005003           CLR     R3              ;CLEAR REST OF TABLE
73 00250 005065           CLR     VARSAV(R5)
         000022
74 00254 000724           BR      USE2
75                ;
```



```
1                 ;       .SBTTL  "RDB" COMMAND PROCESSOR
2                 ;
3                 ;*****************************************************************
4                 ;
5                 ; "RDB" COMMAND PROCESSOR
6                 ;
7                 ; BASIC FORM:   CALL "RDB"(BUF,VAR)
8                 ;
9                 ; PURPOSE:  RETURN THE NEXT DATA POINT FROM THE SPECIFIED
10                ;               BUFFER.  RETURNS VALUES OF 65535 >= VAR >= 0
11                ;               FOR GOOD DATA.  BAD DATA (DEFINED AS OVERRUN)
12                ;               IS RETURNED AS OUTSIDE THE SPECIFIED RANGE
13                ;               (VIZ., NEG. FLOATING POINT MINUS ONE).  IF NO
14                ;               DATA EXISTS YET, I.E., IT'S WAITING FOR A NEW
15                ;               SAMPLE, A FLOATING POINT MINUS TWO WILL BE
16                ;               RETURNED.
17                ;
18                ;               WHEN THE REFERENCED BUFFER REFERS TO
19                ;               ANALOG SAMPLING (RTS FUNCTION), THE VALUES
20                ;               RETURNED ARE IN THE RANGE 4095 >= VAR >= 0.
21                ;
22                ;               WHEN THE REFERENCED BUFFER REFERS TO A CLOCKED
23                ;               HISTOGRAM SAMPLING (HIST FUNCTION), THE VALUES
24                ;               RETURNED ARE IN THE RANGE 65535 >= VAR >= 0.
25                ;               THESE VALUES ARE EITHER THE NUMBER OF TICKS
26                ;               ACCUMULATED OR THE NUMBER REMAINING DEPENDING
27                ;               ON THE CURRENT CLOCK MODE.
28                ;
29                ;               WHEN THE REFERENCED BUFFER REFERS TO A DIGITAL
30                ;               I/O OPERATION (DRS FUNCTION), A VALUE BETWEEN
31                ;               65535 >= VAR >= 0 IS RETURNED FROM THE NEXT
32                ;               POSITION IN THE SPECIFIED BUFFER.
33                ;
34                ;               IN ALL CASES, A FLOATING POINT MINUS ONE
35                ;               IS RETURNED WHEN THE BUFFER IS OVERRUN.
36                ;
37 00256          RDB:                            ;ENTRY POINT TO PROCESSOR
38 00256 004767           JSR     PC,POLENT       ;ENTER POLISH MODE AND CHECK
         001042
39                                                ;    FOR ENOUGH AREA TO WORK
40                                                ;    WITH AND THAT FIRST TOKEN
41                                                ;    IS INDEED A LEFT PARENS.
42 00262 000702'          +GETBUF                 ;GET BUFFER ADDRESS FROM COMMAND STRING
43                                                ;    AND CHECK IT AGAINST THE
44                                                ;    DEFINITION TABLE.
45 00264 000366'          +SAVER2                 ;SAVE DESCRIPTOR ADDRESS ON STACK
46 00266 001352'          +CKCOMA                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
47 00270 000656'          +GETV                   ;GET VARIABLE ADDRESS.
48 00272 000274'          .+2                     ;EXIT POLISH MODE
49 00274 012602           MOV     (SP)+,R2        ;RESTORE DESCRIPTOR ADDRESS
50 00276 004767           JSR     PC,GETDAT       ;GET DATA FROM ARRAY SPECIFIED BY R2
         000454
51 00302 103402           BCS     RDB1            ;CS MEANS R3 IS RETURNING AN
52                                                ;    ERROR CONDITION.
53 00304 004767           JSR     PC,FLOAT        ;FLOAT THE 16 BIT UNSIGNED INTEGER
         000234
54 00310 000167 RDB1:     JMP     STORXT          ;SAVE FAC IN PREVIOUSLY OPENED
```

```
        001046
55                                              ;    VARIABLE, CHECK FOR ")", AND
56                                              ;    RETURN TO THE BASIC INTERPRETER.
57              ;
```



```
1               ;       .SBTTL  "ACC" COMMAND PROCESSOR
2               ;
3               ;**************************************************************
4               ;
5               ; "ACC" COMMAND PROCESSOR
6               ;
7               ; BASIC FORM:   CALL "ACC"(BUF)
8               ;
9               ; PURPOSE:  ACCESS ENTIRE BUFFER "BUF".  THIS COMMAND RESETS
10              ;               ALL BUFFER POINTERS TO ALLOW FULL ACCESS TO
11              ;               THE ARRAY VIA THE "RDB" COMMAND.  THE "PUTD"
12              ;               POINTER IS PLACED AT THE END OF THE ARRAY AND
13              ;               THE "RDB" POINTER IS PLACED AT THE BEGINNING.
14              ;
15 00314        ACC:                            ;ENTRY POINT TO PROCESSOR
16 00314 004767         JSR     PC,POLENT       ;ENTER POLISH MODE AND CHECK
         001004
17                                              ;    FOR ENOUGH AREA TO WORK
18                                              ;    WITH AND THAT FIRST TOKEN
19                                              ;    IS INDEED A LEFT PARENS.
20 00320 000702'         +GETBUF                ;GET BUFFER ADDRESS FROM COMMAND
21                                              ;    AND CHECK IT AGAINST THE
22                                              ;    DEFINITION TABLE.
23 00322 000640'         +BUFRES                ;RESET BUFFER POINTERS
24 00324 000326'         .+2                    ;LEAVE POLISH MODE
25 00326 012762         MOV     #400,BAD(R2)    ;SET BAD ID INDICATING NEXT PUT
         000400
         000010
26                                              ;    BEFORE ANOTHER GET WILL CAUSE
27                                              ;    OVERLAY OF GOOD DATA.
28 00334 000137         JMP     @#CKRPAR        ;MAKE SURE LAST TOKEN IS A ")" AND
         001366'
29                                              ;    RETURN TO THE BASIC
30                                              ;    INTERPRETER.
31              ;
```

```
                        ;       .SBTTL  CKCMGN AND GETNUM
                        ;
                        ;*****************************************************************
                        ;
                        ; POLISH ROUTINE TO CHECK NEXT TOKEN EQUAL TO A ,COMMA
                        ; AND THEN FETCH THE NEXT FLOATING POINT NUMERIC FROM INPUT
                        ; STRING.
                        ;
                        ; ROUTINE IS CALLED IN POLISH MODE
                        ;
                        ; REGISTERS USED:       EVAL USES R0, R2, AND R3
                        ;                       R1 IS BUMPED PAST VARIABLE
                        ;                       RESULT IS RETURNED IN FAC
                        ;
00340                   CKCMGN:
00340 122127            CMPB    (R1)+,#,COMMA   ;CHECK NEXT TOKEN EQUAL TO
      000000G
                                                ;    A ,COMMA.
00344 001402            BEQ     GETNUM          ;YES:  GET NUMERIC NOW
00346 000137            JMP     @#ERRSYN                ;SYNTAX ERROR
      000000G
                        ;
                        ;*****************************************************************
                        ;
                        ; POLISH ROUTINE TO FETCH A FLOATING POINT NUMERIC FROM
                        ; INPUT STRING.
                        ;
                        ; ROUTINE IS CALLED IN POLISH MODE
                        ;
                        ; REGISTERS USED:       EVAL USES R0, R2, AND R3
                        ;                       R1 IS BUMPED PAST VARIABLE
                        ;                       RESULT IS RETURNED IN FAC
                        ;
00352                   GETNUM:
00352 004737            JSR     PC,@#EVAL       ;GET NUMERIC
      000000G
00356 103401            BCS     ERARG           ;ILLEGAL ARG TYPE
00360 000134            POLRET                  ;RETURN IN POLISH MODE
                        ;
00362 000137 ERARG:     JMP     @#ERRARG        ;ILLEGAL ARG TYPE
      000000G
                        ;
```



```
                        ;       .SBTTL  SAVE/RESTORE R2 AND FAC (SAVER2,SAVFAC, RESTR2)
                        ;
                        ;*****************************************************************
                        ;
                        ; POLISH ROUTINES TO SAVE/RESTORE REGISTER R2 OR
                        ; THE FAC ON THE STACK.
                        ;
                        ; ALL ROUTINES ARE CALLED IN POLISH MODE
                        ;
                        ; REGISTERS USED:  ONLY THE STACK
                        ;
00366                   SAVER2:
00366 010246            MOV     R2,-(SP)        ;SAVE R2 ON THE STACK.
00370 000134            POLRET                  ;RETURN IN POLISH MODE
                        ;
00372                   SAVFAC:
00372 016546            MOV     FAC1(R5),-(SP)  ;SAVE THE FAC ON THE STACK
      000040
00376                   SAVINT:
00376 016546            MOV     FAC2(R5),-(SP)
      000042
00402 000134            POLRET                  ;RETURN IN POLISH MODE
                        ;
00404                   RESTR2:
00404 012602            MOV     (SP)+,R2        ;RESTORE R2 FROM THE STACK
00406 000134            POLRET                  ;RETURN IN POLISH MODE
                        ;
```

```
                        ;       .SBTTL  INTERRUPT REGISTER SAVE/RESTORE ROUTINES
                        ;
                        ;*****************************************************************
                        ;
                        ; ROUTINES TO SAVE/RESTORE R0, R2 AND R3 ON STACK AND
                        ; RETURN TO INTERRUPTED PROGRAM.
                        ;
000410                  REGSAV:
000410 010246                   MOV     R2,-(SP)        ;PUSH R2 ON STACK
000412 016602                   MOV     2(SP),R2        ;GET PC FROM CALL
       000002
000416 010066                   MOV     R0,2(SP)        ;NOW PUT R0 THERE
       000002
000422 010346                   MOV     R3,-(SP)        ;SAVE R3
000424 010207                   MOV     R2,PC           ;RETURN TO CALLER
                        ;
000426                  RESTOR:
000426 012603                   MOV     (SP)+,R3        ;RESTORE R3
000430 012602                   MOV     (SP)+,R2        ;RESTORE R2
000432 012600                   MOV     (SP)+,R0        ;RESTORE R0
000434 000002                   RTI                     ;RETURN TO INTERRUPTED PROGRAM
                        ;
```



```
                        ;       .SBTTL  INST
                        ;
                        ;*****************************************************************
                        ;
                        ; POLISH ROUTINE TO INTEGERIZE FAC AND CHECK ITS
                        ; RANGE AGAINST A GIVEN VALUE.
                        ;
                        ; ROUTINE IS CALLED IN POLISH MODE
                        ;
                        ; REGISTERS USED:        R0 ONLY
                        ;                        T3(R5) MUST POINT TO RANGE ELEMENT
                        ;                        (T3(R5) IS BUMPED BY 2 AFTER CALL)
                        ;
                        ;
00436                   INTST:
00436 004737                    JSR     PC,@#INT        ;INTEGERIZE RESULT
       000000G
00442 005765                    TST     FAC1(R5)
       000040
00446 100433                    BMI     ERNOR           ;NEGATIVE NUMBER IS ILLEGAL
00450 001416                    BEQ     IN1             ;ALREADY IN AN INTEGER FORM
00452 026527                    CMP     FAC1(R5),#44200 ;TEST NUMBER OUT OF RANGE
       000040
       044200
                                                        ;    (I.E. >65535).
00460 103022                    BHIS    IN2             ;YES:  SET EQUAL TO MAX (65535)
00462 156565                    BISB    FAC1(R5),FAC2(R5); NUMBER MUST BE A 16 BIT INTEGER
       000040
       000042
00470 000365                    SWAB    FAC2(R5)        ;    TO GET THIS FAR,  CONVERT IT
       000042
00474 052765                    BIS     #100000,FAC2(R5) ;    TO ONE WORD THEN.
       100000
       000042
00502 005065            IN3:    CLR     FAC1(R5)
       000040
00506 026575            IN1:    CMP     FAC2(R5),@T3(R5) ;NUMBER IS IN 16 BIT FORM
       000042
       000062
                                                        ;    NOW, MAKE SURE ITS IN RANGE.
00514 101010                    BHI     ERNOR           ;NUMBER TO LARGE, "ARG ERROR"
00516 062765                    ADD     #2,T3(R5)       ;BUMP TO NEXT RANGE FOR NEXT
       000002
       000062
                                                        ;    CALL.
00524 000134                    POLRET                  ;RETURN IN POLISH MODE
00526 012765            IN2:    MOV     #-1,FAC2(R5)    ;SET OVERFLOW EQUAL TO MAX (65535).
       177777
       000042
00534 000762                    BR      IN3
                        ;
00536                   ERNOR:
00536 104400                    TRAP    0
00540    116                    .ASCII  /NOR/
00541    117
00542    122
00543    000                    .BYTE   0
                        ;
```

```
                        ;       .SBTTL  FLOAT
                        ;
                        ;*****************************************************************
                        ;
                        ; ROUTINE TO FLOAT A 16 BIT UNSIGNED INTEGER
                        ;
                        ; REGISTERS USED:        R3 ON CALL MUST HAVE 16 BIT NUMBER
                        ;                                TO FLOAT.
                        ;                        FAC HAS FLOATED RESULTED
                        ;
00544                   FLOAT:
00544 005703                    TST     R3              ;16 BIT NUMBER?
00546 100017                    BPL     FLOAT1          ;NO:  NUMBER OK AS IS THEN
00550 010346                    MOV     R3,-(SP)        ;PUT NUMBER ON STACK
00552 042716                    BIC     #100000,(SP)    ;IGNORE MOST SIGNIF. BIT
      100000
00556 105016                    CLRB    (SP)            ;CLEAR OUT LEAST SIGNIF.
00560 000316                    SWAB    (SP)            ;NOW MOVE MOST SIGNIF. TO RIGHT
                                                        ;    BYTE
00562 062716                    ADD     #44000,(SP)     ;ADD IN EXPONENT
      044000
00566 012665                    MOV     (SP)+,FAC1(R5)  ;SAVE MOST SIGNIF. AND EXPONENT
      000040
00572 010346                    MOV     R3,-(SP)        ;NOW WORK ON LEAST SIGNIF.
00574 000316                    SWAB    (SP)
00576 105016                    CLRB    (SP)            ;CLEAR OUT THE REMAINING LEAST
                                                        ;    SIGNIF. 8 BITS.
00600 012665                    MOV     (SP)+,FAC2(R5)  ;SAVE IN FAC NOW
      000042
00604 000207                    RETURN                  ;RETURN TO CALLER
                        ;
00606 005065            FLOAT1: CLR     FAC1(R5)        ;LEAVE AS INTEGER
      000040
00612 010365                    MOV     R3,FAC2(R5)
      000042
00616 000207                    RETURN                  ;RETURN TO CALLER.
                        ;
```



```
                        ;       .SBTTL  ENTERP & BUFRES
                        ;
                        ;*****************************************************************
                        ;
                        ; ROUTINE TO ENTER POLISH MODE FOR ONE ROUTINE AND THEN
                        ; EXIT BACK TO CALLER.
                        ;
                        ; CALL:
                        ;       JSR     PC,ENTERP       ;ENTER POLISH MODE
                        ;       .WORD   ROUTINE         ;ROUTINE TO ENTER
                        ;       NEXT INSTRUCTION        ;RETURN MADE HERE
                        ;
00620                   ENTERP:
00620 012704                    MOV     #RETURP,R4      ;SETUP POLISH RETURN PC
      000630'
00624 017607                    MOV     @(SP),PC        ;CALL ROUTINE IN POLISH MODE
      000000
                        ;
00630                   RETURP:
00630 000632'                   .+2                     ;EXIT POLISH MODE
00632 062716                    ADD     #2,(SP)         ;BUMP PAST ARG
      000002
00636 000207                    RETURN                  ;RETURN TO CALLER
                        ;
                        ;*****************************************************************
                        ;
                        ; POLISH ROUTINE TO RESET THE POINTERS (BOTH GET AND PUT) OF
                        ; A CIRCULAR BUFFER.  EFFECTIVELY EMPTIES THE BUFFER BEFORE
                        ; A SAMPLING OPERATION OCCURS.
                        ;
                        ; ROUTINE IS CALLED IN POLISH MODE
                        ;
                        ;                               R2 ON CALL MUST CONTAIN ADDRESS
                        ;                               OF BUFFER IN DESCRIPTOR TABLE.
                        ;                               NO REGISTERS (GENERAL PURPOSE)
                        ;                               ARE ALTERED BY THE ROUTINE.
                        ;
00640                   BUFRES:
00640 011262                    MOV     (R2),PUT(R2)    ;RESET THE GET AND PUT POINTERS
      000004
00644 011262                    MOV     (R2),GET(R2)
      000006
00650 005062                    CLR     BAD(R2)         ;CLEAR OUT BAD DATA FLAG
      000010
00654 000134                    POLRET                  ;RETURN IN POLISH MODE
                        ;
                        ;
```

```
1                       ;       .SBTTL  GETV AND GETBUF
2                       ;
3                       ;*****************************************************************
4                       ;
5                       ; POLISH ROUTINE TO CALCULATE ADDRESS OF VARIABLE
6                       ; SPECIFIED IN THE INPUT LINE.
7                       ;
8                       ; ROUTINE IS CALLED IN POLISH MODE
9                       ;
10                      ; REGISTERS USED:       EVAL USES R0, R2, AND R3
11                      ;                       R1 IS BUMPED PAST VARIABLE
12                      ;                       RESULT IS STORED IN VARSAV(R5)
13                      ;
14 00656                GETV:
15 00656 112102                 MOVB    (R1)+,R2        ;BUILD WORD OFFSET
16 00660 100406                 BMI     SYNER           ;TOKEN ILLEGAL HERE
17 00662 000302                 SWAB    R2
18 00664 152102                 BISB    (R1)+,R2
19 00666 061502                 ADD     (R5),R2         ;R2 NOW POINTS TO SYMBOL TABLE ENTRY
20 00670 004737                 JSR     PC,@#GETVAR     ;GET NAME AND SUBSCRIPTS SET UP
         000000G
21 00674 000134                 POLRET                  ;RETURN IN POLISH MODE
22                      ;
23 00676 000137 SYNER:  JMP     @#ERRSYN        ;SYNTAX ERROR
         000000G
24                      ;
25                      ;*****************************************************************
26                      ;
27                      ; POLISH ROUTINE TO FETCH NEXT VARIABLE FROM INPUT LINE,
28                      ; TEST FOR AN ARRAY NAME AND CALCULATE THE POSITION
29                      ; IN THE ARRAY FROM SUBSCRIPTS WHICH MAY BE GIVEN.
30                      ;
31                      ; ROUTINE IS CALLED IN POLISH MODE
32                      ;
33                      ; ON EXIT:              R0 POINTS TO ARRAY BASE ADDRESS
34                      ;                       R1 IS BUMPED PAST ARRAY NAME
35                      ;                       R2 POINTS TO SYMBOL TABLE ADDRESS
36                      ;                       R3 POINTS TO END OF ARRAY
37                      ;                       VARSAV(R5) POINTS TO ACTUAL ARRAY ADDRESS
38                      ;
39                      ; ROUTINE FALLS THROUGH TO CKBUF.
40                      ;
41 00702                GETBUF:
42 00702 004767                 JSR     PC,GETADD       ;CALCULATE ARRAY ADDRESS FROM
         000310
43                                                      ;       ARRAY NAME POINTED TO BY R1.
44 00706 026503                 CMP     VARSAV(R5),R3   ;DETERMINE IF GIVEN SUBSCRIPTED
         000022
45                                                      ;       ARRAY IS WITHIN BOUNDS OF
46                                                      ;       DIMENSIONED ARRAY.
47 00712 103016                 BHIS    ERBUF           ;BUFFER ERROR
48                      ;
```



```
1                       ;       .SBTTL  CKBUF
2                       ;
3                       ;*****************************************************************
4                       ;
5                       ; POLISH ROUTINE TO CHECK IF ARRAY ADDRESS SPECIFIED
6                       ; IN VARSAV(R5) HAS BEEN DEFINED BY A "USE" COMMAND.  IF SO,
7                       ; THE ENTRY ADDRESS IN THE TABLE IS RETURNED IN R2 AND THE
8                       ; ARRAY POINTERS INITIALIZED.
9                       ;
10                      ; ROUTINE IS CALLED IN POLISH MODE
11                      ;
12                      ; CALL IS MADE WITH STARTING ADDRESS OF ARRAY IN VARSAV(R5)
13                      ;
14                      ; REGISTERS USED: ON EXIT, R2 HAS TABLEZDN]WUWWUVUSS
15                      ;                       FOR THIS BUFFER.
16                      ;
17 00714                CKBUF:
18 00714 012702                 MOV     #NARRAY,R2      ;SET UP TO SEARCH "USE" DEFINITION
         000005
19 00720 012746                 MOV     #TABLE,-(SP)    ;       TABLE.
         000000'
20 00724 027665 CK2:    CMP     @(SP),VARSAV(R5) ;MATCH?
         000000
         000022
21 00732 001002                 BNE     CK1             ;NO:  LOOK AT NEXT ENTRY
22 00734 012602                 MOV     (SP)+,R2        ;RETURN ENTRY ADDRESS IN R2
23 00736 000134                 POLRET                  ;RETURN IN POLISH MODE
24 00740 062716 CK1:    ADD     #14,(SP)        ;LOOK AT NEXT ENTRY
         000014
25 00744 005302                 DEC     R2              ;ALL DONE?
26 00746 003366                 BGT     CK2             ;NO:  CONTINUE
27 00750                ERBUF:                          ;NOT FOUND:  BUFFER ERROR
28 00750 104400                 TRAP    0
29 00752    102         .ASCII  /BUF/
   00753    125
   00754    106
30 00755    000         .BYTE   0
31                      ;
```

```
1                       ;       .SBTTL  GETDAT
2                       ;
3                       ;*****************************************************************
4                       ;
5                       ; ROUTINE TO TEST USER'S BUFFER FOR DATA, AND IF IT
6                       ; EXISTS, REMOVE IT AND RETURN IN R3.
7                       ;
8                       ; CALL:
9                       ;       JSR     PC,GETDAT
10                      ;
11                      ; REGISTERS USED:       R2 MUST POINT TO BUFFER DESCRIPTOR
12                      ;                               BLOCK ADDRESS BEFORE CALL.
13                      ;                       R3 WILL HAVE DATA ON RETURN.
14                      ;                       CARRY WILL BE SET IF AN ERROR OCCURED.
15                      ;                               IN THIS CASE, R3 AND THE FAC
16                      ;                               WILL HAVE THE ERROR CODE.
17                      ;
18 00756                GETDAT:
19 00756 000241         GETDAT: CLC                     ;CLEAR CARRY BIT IN PS
20 00760 016746         GETDAT: MOV     PS,-(SP)        ;SAVE CURRENT PS ON STACK
         177776'
21 00764 052767         GETDAT: BIS     #340,PS         ;RAISE TO LEVEL 7 FOR PROTECTION
         000340
         177776'
22 00772 005065         GETDAT: CLR     FAC1(R5)        ;ALWAYS AN INTEGER RETURNED
         000040
23 00776 105762         GETDAT: TSTB    BAD(R2)         ;CHECK TO SEE IF DATA OVERRUN
         000010
24                                                      ;       OCCURRED.
25 01002 001034         GETDAT: BNE     GET1            ;YES:  RETURN -1 AS VALUE OF DATA
26 01004 026262         GETDAT: CMP     GET(R2),PUT(R2) ;DOES DATA EXIST?
         000006
         000004
27 01012 001422         GETDAT: BEQ     GET2            ;PROBABLY NOT, BUT CHECK FOR FULL
28                                                      ;       BUFFER ANYWAY.
29 01014 005062 GET0:   GETDAT: CLR     BAD(R2)         ;CLEAR BAD DATA FLAG IF SET
         000010
30 01020 017203         GETDAT: MOV     @GET(R2),R3     ;GET DATA
         000006
31 01024 062762         GETDAT: ADD     #2,GET(R2)      ;UPDATE POINTER
         000002
         000006
32 01032 026262         GETDAT: CMP     GET(R2),END(R2) ;DID IT CAUSE A WRAP AROUND?
         000006
         000002
33 01040 101402         GETDAT: BLOS    GET3            ;NO
34 01042 011262         GETDAT: MOV     (R2),GET(R2)    ;WRAP POINTER AROUND
         000006
35 01046 010365 GET3:   GETDAT: MOV     R3,FAC2(R5)     ;SAVE RESULT IN FAC AS AN INTEGER
         000042
36 01052 012667         GETDAT: MOV     (SP)+,PS        ;RESTORE PS
         177776'
37 01056 000207         GETDAT: RETURN                  ;RETURN TO CALLER
38                      ;
39 01060 105762 GET2:   GETDAT: TSTB    BAD+1(R2)       ;IS THERE REALY DATA HERE?
         000011
40 01064 001353         GETDAT: BNE     GET0            ;YES:  GO GET IT THEN
```



```
41 01066 012703         GETDAT: MOV     #-2,R3          ;INDICATE NO DATA EXISTS YET
         177776
42 01072 000404         GETDAT: BR      GET4
43 01074 012703 GET1:   GETDAT: MOV     #-1,R3          ;INDICATE DATA OVERRUN OCCURRED.
         177777
44 01100 105062         GETDAT: CLRB    BAD(R2)         ;CLEAR OUT BAD DATA FLAG
         000010
45                                                      ;       SO NEXT FETCH WILL RESULT
46                                                      ;       IN GOOD DATA.
47 01104 005216 GET4:   GETDAT: INC     (SP)            ;SET CARRY SHOWING FAILURE
48 01106 000757         GETDAT: BR      GET3            ;RETURN
49                      ;
```

```
                ;       .SBTTL  STODAT
                ;
                ;*****************************************************************
                ;
                ; ROUTINE TO STORE DATA INTO USER'S BUFFER AND TEST FOR
                ; DATA OVERFLOW.
                ;
                ; CALL:
                ;       JSR     PC,STODAT
                ;
                ; REGISTERS USED:       R2 MUST HAVE BUFFER DESCRIPTOR BLOCK
                ;                               ADDRESS BEFORE CALL.
                ;                       R3 MUST HAVE DATA TO INSERT BEFORE
                ;                               CALL.
                ;                       R3 IS DESTROYED ON RETURN
                ;
01110           STODAT:
01110 016746    MOV     PS,-(SP)        ;SAVE CURRENT PS ON STACK
      177776'
01114 052767    BIS     #340,PS         ;RAISE TO LEVEL 7 FOR PROTECTION
      000340
      177776'
01122 105762    TSTB    BAD+1(R2)       ;OVERRUN BIT SET?
      000011
01126 001025    BNE     NOROOM          ;YES:  SET OVERFLOW FLAG
01130 016246    MOV     PUT(R2),-(SP)   ;UPDATE PUT POINTER
      000004
01134 062716    ADD     #2,(SP)
      000002
01140 021662    CMP     (SP),END(R2)    ;DID IT CAUSE A WRAP AROUND?
      000002
01144 101401    BLOS    NOWRAP          ;NO
01146 011216    MOV     (R2),(SP)       ;WRAP POINTER AROUND
01150 021662    NOWRAP: CMP     (SP),GET(R2)    ;WILL WE OVERRUN THE GET POINTER
      000006
                                        ;       NEXT TIME?
01154 001003    BNE     STOD1           ;NO
01156 152762    BISB    #1,BAD+1(R2)    ;YES:  REMEMBER THIS THEN
      000001
      000011
01164 010372    STOD1:  MOV     R3,@PUT(R2)     ;SAVE DATA
      000004
01170 012662    MOV     (SP)+,PUT(R2)   ;SAVE NEW PUT POINTER ALSO
      000004
01174 012667    MOV     (SP)+,PS        ;RESTORE PS
      177776'
01200 000207    RETURN                  ;RETURN TO CALLER
                ;
01202 052762    NOROOM: BIS     #1,BAD(R2)      ;REMEMBER THAT BAD DATA EXISTS
      000001
      000010
01210 016246    MOV     PUT(R2),-(SP)   ;DON'T UPDATE THE PUT POINTER
      000004
01214 000763    BR      STOD1           ;RESTORE PS AND RETURN TO CALLER
                ;
```



```
                ;       .SBTTL  GETADD
                ;
                ;*****************************************************************
                ;
                ; ROUTINE TO CALCULATE ARRAY ADDRESS FROM ARRAY NAME
                ; POINTED TO BY R1.
                ;
                ; REGISTERS ON RETURN:  R0 POINTS TO ARRAY BASE ADDRESS
                ;                       R1 IS UPDATED PAST ARRAY NAME
                ;                       R2 POINTS TO SYMBOL TABLE ENTRY
                ;                       R3 HAS ADDRESS OF END OF ARRAY
                ;                       VARSAV(R5) POINTS TO ENTRY TO ARRAY
                ;
01216           GETADD:
01216 112102    MOVB    (R1)+,R2        ;BUILD WORD OFFSET
01220 100450    BMI     SYNERR          ;TOKEN ILLEGAL HERE
01222 000302    SWAB    R2
01224 152102    BISB    (R1)+,R2
01226 061502    ADD     (R5),R2         ;R2 NOW POINTS TO SYMBOL TABLE
01230 022712    CMP     #177776,(R2)    ;VARIABLE MUST BE AN ARRAY
      177776
01234 001042    BNE     SYNERR          ;SYNTAX ERROR IF NOT
01236 010246    MOV     R2,-(SP)        ;SAVE R2 TEMP
01240 004737    JSR     PC,@#GETVAR     ;GET NAME AND SUBSCRIPTS SET UP
      000000G
01244 012602    MOV     (SP)+,R2        ;RESTORE R2
01246 022765    CMP     #-1,SS2SAV(R5)  ;SINGLY SUBSCRIPTED ARRAYS ONLY,
      177777
      000026
01254 001032    BNE     SYNERR
01256 016203    MOV     2(R2),R3        ;GET ADDRESS OF ARRAY
      000002
01262 010300    MOV     R3,R0           ;CALCULATE EXACT ARRAY ADDRESS
01264 010065    MOV     R0,VARSAV(R5)   ;       FROM GIVEN SUBSCRIPTS.
      000022
01270 005765    TST     SS1SAV(R5)
      000024
01274 100405    BMI     G1              ;NO SUBSCRIPTS GIVEN.
01276 006365    ASL     SS1SAV(R5)      ;CONVERT SUBSCRIPT TO A WORD OFFSET
      000024
01302 066565    ADD     SS1SAV(R5),VARSAV(R5) ;SUBSCRIPT PLUS ARRAY ADDRESS
      000024
      000022
01310 016246    G1:     MOV     4(R2),-(SP)     ;(+ 4 * BYTE SUBSCRIPTS)
      000004
01314 006316            ASL     (SP)
01316 006316            ASL     (SP)
01320 062603            ADD     (SP)+,R3
01322 000207            RETURN                  ;RETURN TO CALLER
                ;
```

```
                ;       .SBTTL  POLENT, CKLPAR AND CKCOMA
                ;
                ;*****************************************************************
                ;
                ; ENTER POLISH MODE OPERATION
                ;
                ; REGISTERS USED:       R4 BECOMES THE POLISH PC.
                ;
001324          POLENT:
01324 012604            MOV     (SP)+,R4        ;GET PC FROM CALLER AND SETUP
                                                ;    THE POLISH PC.
                ;
                ;*****************************************************************
                ;
                ; POLISH ROUTINE TO CHECK CHARACTER POINTED TO BY R1 EQUAL
                ; TO A LEFT PARENS.  A TEST IS THEN MADE TO INSURE
                ; THAT WE STILL HAVE ROOM LEFT TO WORK WITH.
                ;
                ; ROUTINE IS CALLED IN POLISH MODE
                ;
                ; REGISTERS USED:       R1 GETS BUMPED BY 1
                ;
01326           CKLPAR:
01326 122127            CMPB    (R1)+,#,LPAR    ;FIRST CHAR MUST BE A LEFT
        000000G
                                                ;    PARENS.
01332 001003            BNE     SYNERR          ;NO:  SYNTAX ERROR
01334 020604            CMP     SP,R4           ;CHECK ROOM NOW
01336 103403            BLO     ERPDL           ;OUT OF ROOM, TELL USER
01340 000134            POLRET                  ;RETURN IN POLISH MODE
01342           SYNERR:
01342 000137            JMP     @#ERRSYN        ;SYNTAX ERROR
        000000G
01346           ERPDL:
01346 000137            JMP     @#ERRPDL        ;OUT OF ROOM
        000000G
                ;
                ;*****************************************************************
                ;
                ; POLISH ROUTINE TO CHECK CHARACTER POINTED TO BY R1
                ; EQUAL TO A ,COMMA.
                ;
                ; ROUTINE IS CALLED IN POLISH MODE
                ;
                ; REGISTERS USED:  R1 GETS BUMPED BY ONE.
                ;
01352           CKCOMA:
01352 122127            CMPB    (R1)+,#,COMMA   ;FIRST CHAR MUST BE A COMMA
        000000G
01356 001371            BNE     SYNERR          ;NO: SYNTAX ERROR
01360 000134            POLRET                  ;RETURN IN POLISH MODE
                ;
```



```
                ;       .SBTTL  STORXT AND CKRPAR
                ;
                ;*****************************************************************
                ;
                ; COMMON EXIT ROUTINE TO STORE FAC IN VARIABLE SET UP
                ; IN VARSAV(R5), CHECK LAST TOKEN EQUAL TO A RIGHT
                ; PARENS (ALTERNATE ENTRY POINT TO ROUTINE), AND
                ; RETURN TO THE BASIC INTREPRETER.
                ;
01362           STORXT:
01362 004737            JSR     PC,@#STOVAR     ;SAVE FAC
        000000G
01366           CKRPAR:
01366 122127            CMPB    (R1)+,#,RPAR    ;MAKE SURE LAST TOKEN IS A
        000000G
                                                ;    RIGHT PARENS.
01372 001363            BNE     SYNERR          ;SYNTAX ERROR IF NOT
01374 000207            RETURN                  ;RETURN TO THE BASIC INTREPRETER
                ;
```

```
1                       ;       .SBTTL  CONVERT BCD TO BINARY
2                       ;
3                       ;***************************************************************
4                       ;
5                       ; ROUTINE TO CONVERT BCD TO BINARY
6                       ;
7                       ; CALL:
8                       ;       JSR     PC,CONBCD
9                       ;
10                      ; REGISTERS USED:       R3 MUST CONTAIN BCD NUMBER BEFORE CALL
11                      ;                       R3 WILL HAVE BINARY RESULT ON RETURN
12                      ;
13 01376                CONBCD:
14 01376 010046                 MOV     R0,-(SP)        ;SAVE REGISTERS R0, R2, AND R4
15 01400 010246                 MOV     R2,-(SP)        ;    ON THE STACK
16 01402 010446                 MOV     R4,-(SP)
17 01404 012700                 MOV     #4,R0           ;4 BCD DIGITS TO DECODE
         000004
18 01410 005046                 CLR     -(SP)           ;ACCUMULATE AREA
19 01412 012702 CON1:   MOV     #4,R2           ;4 SHIFTS / BCD CHAR
         000004
20 01416 005004                 CLR     R4              ;CLEAR OUT WORK AREA
21 01420 006103 CON2:   ROL     R3              ;SHIFT CHAR OUT
22 01422 006104                 ROL     R4
23 01424 005302                 DEC     R2              ;4 SHIFTS YET?
24 01426 003374                 BGT     CON2            ;NO:  CONTINUE
25 01430 011646                 MOV     (SP),-(SP)      ;MULTIPLY ACCUMULATED DIGITS BY 10
26 01432 006316                 ASL     (SP)
27 01434 006316                 ASL     (SP)
28 01436 062616                 ADD     (SP)+,(SP)
29 01440 006316                 ASL     (SP)
30 01442 060416                 ADD     R4,(SP)         ;ADD IN NEW DIGIT
31 01444 005300                 DEC     R0              ;IS THIS THE LAST BCD CHARACTER?
32 01446 003361                 BGT     CON1            ;NO:  CONTINUE
33                      ;
34 01450 012603                 MOV     (SP)+,R3        ;RESULT RETURNED IN R3
35 01452 012604                 MOV     (SP)+,R4        ;RESTORE ALL REGISTERS
36 01454 012602                 MOV     (SP)+,R2
37 01456 012600                 MOV     (SP)+,R0
38 01460 000207                 RETURN                  ;RETURN TO CALLER
39                      ;
40       000001'        .END                    ;END OF MODULE #0
```


SYMBOL TABLE

```
ACC     000314RG        BAD   = 000010          BCDON   000077RG
BEG   = 000000          BUFRES  000640RG        CKBUF   000714R
CKCMGN  000340RG        CKCOMA  001352RG        CKLPAR  001326R
CKRPAR  001366RG        CK1     000740R         CK2     000724R
CONBCD  001376RG        CON1    001412R         CON2    001420R
DEL   = 000012          DRSBUF  000100RG        DRSNPT  000102RG
DRSON   000075RG        END   = 000002          ENTERP  000620RG
ERARG   000362R         ERBUF   000750RG        ERNOR   000536RG
ERPDL   001346R         ERRARG= ****** G        ERRPDL= ****** G
ERRSYN= ****** G        EVAL  = ****** G        FAC1  = 000040
FAC2  = 000042          FLOAT   000544RG        FLOAT1  000606R
GET   = 000006          GETADD  001216RG        GETBUF  000702RG
GETDAT  000756RG        GETNUM  000352RG        GETV    000656RG
GETVAR= ****** G        GET0    001014R         GET1    001074R
GET2    001060R         GET3    001046R         GET4    001104R
G1      001310R         HISTON  000076RG        INT   = ****** G
INTST   000436RG        IN1     000506R         IN2     000526R
IN3     000502R         LPSAD = ****** G        LPSDR = ****** G
NARRAY= 000005 G        NOROOM  001202R         NOWRAP  001150R
NUMSGN= ****** G        PC    =%000007          POLENT  001324RG
POLRET= 000134          PS    = 177776          PUT   = 000004
RDB     000256RG        RDB1    000310R         REGSAV  000410RG
RESTOR  000426RG        RESTR2  000404RG        RETURN= 000207
RETURP  000630R         RTSON   000074RG        R0    =%000000
R1    =%000001          R2    =%000002          R3    =%000003
R4    =%000004          R5    =%000005          SAVER2  000366RG
SAVFAC  000372RG        SAVINT  000376RG        SP    =%000006
SS1SAV= 000024          SS2SAV= 000026          STODAT  001110RG
STOD1   001164R         STORXT  001362RG        STOVAR= ****** G
SYNER   000676R         SYNERR  001342R         TABLE   000000RG
T3    = 000062          USE     000104RG        USE1    000236R
USE2    000126R         USE3    000122R         USE4    000222R
VARSAV= 000022          .COMMA= ****** G        .LPAR = ****** G
.RPAR = ****** G
. ABS.  000000      000
        001462      001
ERRORS DETECTED: 0
FREE CORE: 18142. WORDS

.LP:=LPS0
```

```
1               .TITLE  LPS1  V01-01
2               ;.SBTTL LPS MODULE #1
3               ;
4               ; DEC-11-LBEXA-B-LA2    BASIC KERNEL V02-01
5               ;
6               ; COPYRIGHT (C) 1973,1974
7               ;
8               ; DIGITAL EQUIPMENT CORPORATION
9               ; MAYNARD, MASSACHUSETTS 01754
10              ;
11              ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
12              ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
13              ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION,
14              ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
15              ; MAY APPEAR IN THIS DOCUMENT.
16              ;
17              ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
18              ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
19              ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
20              ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
21              ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
22              ;
23              ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
24              ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
25              ; WHICH IS NOT SUPPLIED BY DEC.
26              ;
27              ; WRITTEN BY:  RICK HULLY       FEB,, 1973
28              ;
```

22


```
1               ;       .SBTTL  CONTENTS OF MODULE #1
2               ;
3               ; THE FOLLOWING COMMAND PROCESSORS EXIST IN THIS MODULE:
4               ;
5               ;       ADC
6               ;       RTS
7               ;       LED
8               ;
```

```
1                       ;       .SBTTL  PROGRAM GLOBALS
2                       ;
3                       ; GLOBALS REQUIRED FOR COMMUNICATION WITH THE BASIC
4                       ; INTREPRETER.
5                       ;
6                               .GLOBL  ERRPDL,ERRSYN,ERRARG
7                               .GLOBL  EVAL,GETVAR,STOVAR
8                               .GLOBL  INT,,COMMA,,RPAR
9                               .GLOBL  NUMSGN,,LPAR
10                      ;
11                      ; GLOBALS REQUIRED FOR COMMUNICATIONS WITH THE VARIOUS
12                      ; LPS MODULES.
13                      ;
14                              .GLOBL  TABLE,NARRAY,FLOAT
15                              .GLOBL  POLENT,GETADD,GETNUM
16                              .GLOBL  CKCMGN,INTST,GETV
17                              .GLOBL  GETBUF,REGSAV,ERNOR
18                              .GLOBL  STORXT,CKRPAR,ERBUF
19                              .GLOBL  CKCOMA,ENTERP,STODAT
20                              .GLOBL  GETDAT,RTSON,DRSON
21                              .GLOBL  SAVER2,SAVFAC,RESTR2
22                              .GLOBL  RESTOR,HISTON,BCDON
23                              .GLOBL  CONBCD,BUFRES,DRSNPT
24                              .GLOBL  SAVINT
25                      ;
26                      ; GLOBALS REQUIRED FOR COMMAND PROCESSORS
27                      ;
28                              .GLOBL  ADC,RTS,LED
29                      ;
30                      ; GLOBALS FOR DEVICE ADDRESSES
31                      ;
32                              .GLOBL  LPSAD,LPSADB,LPSDR,LPSDMA
33                      ;
34                      ; GLOBALS FOR INTERRUPT PRIORITY AND VECTOR DEFINITION:
35                              .GLOBL  LPSIVA,LPSIP
```

23


```
1                       ;       .SBTTL  REGISTER ASSIGNMENTS AND EQUATES
2                       ;
3       000000 R0       =       %0
4       000001 R1       =       %1
5       000002 R2       =       %2
6       000003 R3       =       %3
7       000004 R4       =       %4
8       000005 R5       =       %5
9       000006 SP       =       %6
10      000007 PC       =       %7
11                      ;
12                      ; GENERAL EQUATES
13                      ;
14      177776 PS       =       -2              ;PS = PROCESSOR STATUS WORD
15      000207 RETURN   =       207             ;207 = RTS PC
16      000134 POLRET   =       134             ;134 = JMP @(R4)+
17                      ;
18                      ; BASIC USER'S EQUATES
19                      ;
20      000022 VARSAV   =       22              ;VAR SAVE FOR ASSIGNMENT
21      000024 SS1SAV   =       24              ;SS1 SAVE FOR ASSIGNMENT
22      000026 SS2SAV   =       26              ;SS2 SAVE FOR ASSIGNMENT
23      000040 FAC1     =       40              ;HIGH ORDER FLOATING VALUE
24      000042 FAC2     =       42              ;LOW  ORDER FLOATING VALUE
25      000062 T3       =       62              ;SHORT TERM TEMPORARY
26                      ;
27                      ; BUFFER DESCRIPTOR EQUATES
28                      ;
29      000000 BEG      =       0               ;OFFSET TO BEGINNING OF BUFFER POINTER
30      000002 END      =       2               ;OFFSET TO END OF BUFFER POINTER
31      000004 PUT      =       4               ;OFFSET TO PUT DATA POINTER
32      000006 GET      =       6               ;OFFSET TO GET DATA POINTER
33      000010 BAD      =       10              ;OFFSET TO BAD DATA FLAG
34      000012 DEL      =       12              ;OFFSET TO DISPLAY DELTA X
35                      ;
36                      ; LIMIT TEST
37                      ;
38 00000 000020 MAXA:   .WORD   20              ;16
39 00002 000020         .WORD   20              ;16
40 00004 177777         .WORD   177777          ;65535
41 00006 000017         .WORD   17              ;17
42                      ;
43                      ; INTERRUPT VECTOR SET UP
44                      ;
45                      ;
46                      ; NON-REENTRANT VARIABLES
47                      ;
48 00010 000000 RTSBUF: .WORD   0               ;RTS COMMAND (BUFFER DESCRIPTOR
49                                              ;     ADDRESS)
50 00012 000000 RTSNPT: .WORD   0               ;RTS COMMAND (NBR OF POINTS)
51 00014 000000 RTSSC:  .WORD   0               ;RTS COMMAND (STARTING CHANNEL)
52 00016 000000 RTSNSC: .WORD   0               ;RTS COMMAND (NBR OF SUCCESSIVE
53                                              ;     CHANNELS)
54 00020 000000 RTSMOD: .WORD   0               ;RTS COMMAND MODE
55 00022 000000 RTSEND: .WORD   0               ;RTS COMMAND END ADDRESS FOR
56                                              ;     DMA OPERATIONS
57 00024 000000 RTSCSR: .WORD   0               ;RTS COMMAND CSR SETTING
```

```
58 00026 000000 CNT:    .WORD   0               ;COUNTER USED BY LED COMMAND
59 00030 000000 LEDRES: .WORD   0,0,0,0         ;STORAGE AREA FOR LED CHARACTERS
   00032 000000
   00034 000000
   00036 000000
60 00040     017 BLANK:  .BYTE   17,17,17,17,17,17,17 ;7 LED BLANK CHARACTERS FOR PADDING
   00041     017
   00042     017
   00043     017
   00044     017
   00045     017
   00046     017
61 00047     015 MINUS:  .BYTE   15              ;LED MINUS CHARACTER
62                 ;
```

24


```
1                  ;       .SBTTL  "ADC" COMMAND PROCESSOR
2                  ;
3                  ;*****************************************************************
4                  ;
5                  ; "ADC" COMMAND PROCESSOR
6                  ;
7                  ; BASIC FORM:   CALL "ADC"(CHAN,VAR)
8                  ;
9                  ; PURPOSE:  INITIATE AN A/D CONVERSION FROM THE SPECIFIED
10                 ;               CHANNEL (0 <= CHAN <= 15), WAIT FOR IT TO
11                 ;               COMPLETE, AND RETURN A FLOATING POINT RESULT
12                 ;               IN "VAR" (0 <= VAR <= 4095) AS THE VALUE OF THE
13                 ;               FUNCTION.  THE A/D CANNOT BE CURRENTLY IN-
14                 ;               VOLVED IN A REAL TIME SAMPLING (RTS)
15                 ;               OPERATION.
16                 ;
17 00050           ADC:                            ;ENTRY POINT TO PROCESSOR
18 00050 105737            TSTB    @#RTSON         ;RTS OPERATION CURRENTLY UNDERWAY?
           000000G
19 00054 001403            BEQ     ADC2            ;NO:  CONTINUE
20 00056 104400            TRAP    0
21 00060     101           .ASCII  /ADC/           ;    IN PROGRESS.
   00061     104
   00062     103
22 00063     000           .BYTE   0
23 00064 012765 ADC2:      MOV     #MAXA,T3(R5)    ;FOR LIMIT TESTING OF "CHAN"
           000000'
           000062
24 00072 004737            JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
           000000G
25                                                 ;    FOR ENOUGH AREA TO WORK
26                                                 ;    WITH AND THAT FIRST TOKEN
27                                                 ;    IS INDEED A LEFT PARENS.
28 00076 000000G           +GETNUM                 ;GET CHANNEL FROM COMMAND STRING
29 00100 000000G           +INTST                  ;TEST <= 15.
30 00102 000000G           +CKCOMA                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
31 00104 000106'           .+2                     ;EXIT POLISH MODE
32 00106 000365            SWAB    FAC2(R5)        ;SETUP A/D CHANNEL #
           000042
33 00112 005265            INC     FAC2(R5)        ;INSERT A/D START BIT
           000042
34 00116 016537            MOV     FAC2(R5),@#LPSAD ;START CONVERSION ON SPECIFIED CHANNEL
           000042
           000000G
35 00124 004737            JSR     PC,@#ENTERP     ;ENTER IMMEDIATE POLISH MODE TO GET
           000000G
36 00130 000000G           +GETV                   ;    VARIABLE ADDRESS.
37 00132 105737 ADC1:      TSTB    @#LPSAD         ;CHECK DONE FLAG ON THIS CHANNEL
           000000G
38 00136 100375            BPL     ADC1            ;NOT READY YET
39 00140 013765            MOV     @#LPSADB,FAC2(R5) ;GET RESULT IN FAC
           000000G
           000042
40 00146 005065            CLR     FAC1(R5)        ;MOST SIGNIF. IS ALWAYS ZERO
           000040
41 00152 000137            JMP     @#STORXT        ;STORE RESULT IN PREVIOUSLY OPENED
           000000G
```

```
42                                      ;    VARIABLE, CHECK FOR ")" AND
43                                      ;    RETURN TO THE BASIC INTERPRETER.
44              ;
```

25


```
1               ;       .SBTTL  "RTS" COMMAND PROCESSOR
2               ;
3               ;*****************************************************************
4               ;
5               ; "RTS" COMMAND PROCESSOR
6               ;
7               ; BASIC FORM:   CALL "RTS"(BUF,SC,NSC,NPTS,MODE)
8               ;
9               ; PURPOSE:  PERFORM REAL TIME BUFFERED/CLOCKED SAMPLING OF THE
10              ;               A/D.  THE A/D CAN BE ENABLED WITH A VARIETY
11              ;               OF OPERATIONS DEPENDING ON THE "MODE" SPECIFIED.
12              ;               THE NORMAL MODE OF OPERATION (MODE=0) CAUSES
13              ;               THE A/D TO SAMPLE WHENEVER ST #1 FIRES.
14              ;               TO ENABLE OTHER OPTIONS, MERELY ADD THEIR CODE
15              ;               NUMBER TO THE MODE.  THE FOLLOWING LIST
16              ;               DESCRIBES OPTIONS AVAILABLE (ALL OPTIONS ARE
17              ;               NORMALLY DISABLED):
18              ;
19              ;               CODE    OPTION
20              ;                +1  ENABLE BURST MODE (USED ONLY WITH DMA)
21              ;                +2  ENABLE CLOCK AND DISABLE ST #1
22              ;                +4  ENABLE DUAL SAMPLE AND HOLD
23              ;               +10  ENABLE DMA (10 OCTAL)
24              ;
25              ;               THE A/D WILL BE STARTED BY A CLOCK OVERFLOW
26              ;               OR SCHMITT TRIGGER #1 (ST).  POINTERS WILL BE
27              ;               BE USED TO DETERMINE IF GOOD DATA EXISTS IN THE
28              ;               BUFFER ARRAYS OR IF DATA WRAP AROUND OCCURS.
29              ;               SINCE DATA IS STORED IN CIRCULAR BUFFERS (EX-
30              ;               CLUDING DMA OPERATIONS), POINTERS WILL BE USED
31              ;               TO INSURE THAT INCOMING DATA RATE DOES NOT
32              ;               EXCEED THE REMOVAL RATE.  A BAD DATA INDICATOR
33              ;               WILL BE SET WHEN DATA OVER-RUN OCCURS.  THE
34              ;               BUFFER POINTERS ARE INITIALLY RESET BEFORE THE
35              ;               SAMPLING OPERATION BEGINS.
36              ;
37              ;               A/D CHANNELS ARE SAMPLED ON EVERY CLOCK OVERFLOW
38              ;               OR FIRING OF ST #1 WITH THE RESULT STORED IN
39              ;               CONSECUTIVE DATA CELLS.  DATA IS STORED IN
40              ;               IDENTICAL FORMAT AS THAT READ FROM THE A/D.  WHEN
41              ;               A CLOCK OVERFLOW OR ST OCCURS, THE A/D SAMPLES
42              ;               THE FIRST CHANNEL SPECIFIED BY "SC" AND THEN
43              ;               SAMPLES THE NEXT "NSC" -1 CONSECUTIVE CHANNELS.
44              ;               SAMPLING THEN CONTINUES UNTIL "NPTS" CLOCK OVERFLOWS
45              ;               OR ST'S HAVE BEEN RECEIVED.
46              ;
47              ;               IN DUAL SAMPLE AND HOLD MODE, THE "NSC" PARAMETER
48              ;               IS THE NUMBER OF PAIRS OF CHANNELS TO REAL PER
49              ;               EVENT.
50              ;
51              ;               DMA OPERATIONS MAY/MAY NOT USE DUAL SAMPLE AND HOLD.
52              ;               DMA ALLOWS DIRECT HARDWARE STORAGE OF A/D
53              ;               RESULTS IN A SPECIFIED BUFFER ARRAY.  A MAXIMUM
54              ;               OF 4096 SAMPLES MAY BE TAKEN AT ANY
55              ;               ONE TIME WITH REMOVAL OF DATA ALLOWED ONLY
56              ;               WHEN THE BUFFER IS COMPLETELY FILLED.  THE
57              ;               "NSC" PARAMETER IS IGNORED AND IS CONSIDERED
```

```
58                      ;               ONE.
59                      ;
60                      ;               IF NPTS IS GIVEN AS ZERO, ANY RTS SAMPLING
61                      ;               CURRENTLY IN PROGRESS IS DISABLED.
62                      ;
63 00156                RTS:                            ;ENTRY POINT TO PROCESSOR
64 00156 012737         MOV     #ADCINT,@#LPSIVA        ;SET INT. SERVICE ROUTINE ENTRY POINT.
         001040'
         000000G
65 00164 012737         MOV     #300,@#LPSIP            ;SET INT. PRIORITY.
         000300
         000000G
66 00172 012765         MOV     #MAXA,T3(R5)    ;FOR LIMIT CHECKS ON ALL INTEGERS
         000000'
         000062
67 00200 005037         CLR     @#LPSAD         ;STOP ANY PREVIOUS RTS OPERATION
         000000G
68 00204 105037         CLRB    @#RTSON
         000000G
69 00210 004737         JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
         000000G
70                                              ;       FOR ENOUGH AREA TO WORK
71                                              ;       WITH AND THAT FIRST TOKEN
72                                              ;       IS INDEED A LEFT PARENS.
73 00214 000000G        +GETBUF                 ;GET BUFFER ADDRESS FROM COMMAND STRING
74                                              ;       AND CHECK IT AGAINST THE
75                                              ;       DEFINITION TABLE.
76 00216 000000G        +BUFRES                 ;RESET BUFFER POINTERS
77 00220 000000G        +SAVER2                 ;SAVE R2 ON THE STACK SINCE IT HAS
78                                              ;       THE DESCRIPTOR ADDRESS.
79 00222 000000G        +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
80 00224 000000G        +INTST                  ;       AND GET "SC".  INTEGERIZE
81                                              ;       RESULT AND TEST <=16.
82 00226 000000G        +SAVINT                 ;SAVE INTEGER ON STACK
83 00230 000000G        +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
84 00232 000000G        +INTST                  ;       AND GET "NSC".  INTEGERIZE
85                                              ;       RESULT AND TEST <=15.
86 00234 000000G        +SAVINT                 ;SAVE INTEGER ON STACK
87 00236 000000G        +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
88 00240 000000G        +INTST                  ;       AND GET "NPTS".  INTEGERIZE
89                                              ;       RESULT AND TEST <=65535.
90 00242 000000G        +SAVINT                 ;SAVE INTEGER ON STACK
91 00244 000000G        +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
92 00246 000000G        +INTST                  ;       AND GET "MODE".  INTEGERIZE
93                                              ;       RESULT AND TEST <=17.
94 00250 000252'        .+2                     ;LEAVE POLISH MODE
95 00252 016500         MOV     FAC2(R5),R0     ;GET MODE
         000042
96 00256 010067         MOV     R0,RTSMOD       ;SAVE IT FOR LATER
         177536
97 00262 012667         MOV     (SP)+,RTSNPT    ;GET NPTS FROM STACK
         177524
98 00266 001424         BEQ     RTS00           ;ZERO POINTS MEANS DISABLE SAMPLING
99 00270 012603         MOV     (SP)+,R3        ;GET NSC FROM STACK
100 0272 010367         MOV     R3,RTSNSC       ;SAVE FOR LATER
         177520
101 0276 012667         MOV     (SP)+,RTSSC     ;GET SC FROM STACK
```



```
         177512
102 0302 012602         MOV     (SP)+,R2        ;GET R2 (BUFFER DESCRIPTOR ADDRESS)
103 0304 010267         MOV     R2,RTSBUF
         177500
104 0310 032700         BIT     #4,R0           ;CHECK SC-1+(NSC*DUAL BIT) <= 15
         000004
105 0314 001401         BEQ     RTS01           ;NOT DUAL
106 0316 006303         ASL     R3              ;DUAL BIT SET, NSC = NSC*2
107 0320 005303 RTS01:  DEC     R3              ;NSC = NSC - 1
108 0322 066703         ADD     RTSSC,R3        ;SC = 1 + (NSC*DUAL BIT)
         177466
109 0326 020327         CMP     R3,#15.         ;MUST BE <=15
         000017
110 0332 003405         BLE     RTS02           ;OK
111 0334 000137         JMP     @#ERNOR         ;NUMBER OUT OF RANGE
         000000G
112 0340 062706 RTS00:  ADD     #6,SP           ;CLEAN UP STACK AND EXIT
         000006
113 0344 000524         BR      RTS10
114 0346 012703 RTS02:  MOV     #100,R3         ;R3 WILL END UP HAVING A/D STATUS
         000100
115                                             ;       REGISTER SETTING.
116 0352 016746         MOV     RTSSC,-(SP)     ;STARTING CHANNEL #
         177436
117 0356 000316         SWAB    (SP)
118 0360 052603         BIS     (SP)+,R3        ;INSERT SC INTO A/D STATUS
119 0362 032700         BIT     #4,R0           ;DUAL SAMPLE AND HOLD?
         000004
120 0366 001402         BEQ     RTS03           ;NO
121 0370 052703         BIS     #40000,R3       ;YES:  INSERT INTO STATUS REG.
         040000
122 0374 032700 RTS03:  BIT     #2,R0           ;CLOCK ENABLE OR ST?
         000002
123 0400 001003         BNE     RTS04           ;CLOCK
124 0402 052703         BIS     #20,R3          ;ST
         000020
125 0406 000402         BR      RTS05
126 0410 052703 RTS04:  BIS     #40,R3          ;CLOCK
         000040
127 0414 006000 RTS05:  ROR     R0              ;NOW GET BURST MODE
128 0416 103002         BCC     RTS06
129 0420 052703         BIS     #10,R3          ;SET BURST MODE BIT IN STATUS REG
         000010
130 0424 032700 RTS06:  BIT     #4,R0           ;DMA REQUESTED?
         000004
131 0430 001444         BEQ     RTS09           ;NO:  STATUS REGISTER OK AS IS THEN
132 0432 005203         INC     R3              ;ADD IN START BIT TO GET THINGS GOING
133                                             ;       FOR DMA OPERATIONS.
134 0434 011200         MOV     (R2),R0         ;MAKE SURE A/D WON'T OVERRUN
135 0436 066700         ADD     RTSNPT,R0       ;       BUFFER SIZE ALLOCATED.
         177350
136 0442 066700         ADD     RTSNPT,R0
         177344
137 0446 032703         BIT     #40000,R3       ;IF DUAL SAMPLE AND HOLD, DOUBLE
         040000
138                                             ;       NUMBER OF POINTS.
139 0452 001404         BEQ     RTS07           ;NO
```

```
140 0454 066700          ADD     RTSNPT,R0
         177332
141 0460 066700          ADD     RTSNPT,R0
         177326
142 0464 010067 RTS07:   MOV     R0,RTSEND       ;SAVE END ADDRESS
         177332
143 0470 026200          CMP     END(R2),R0      ;EXCEEDED ARRAY SIZE?
         000002
144 0474 003402          BLE     RTS08           ;NO:  OK THEN
145 0476 000137          JMP     @#ERBUF         ;DMA EXCEEDS BUFFER SIZE
         000000G
146 0502 012700 RTS08:   MOV     #LPSAD,R0       ;SETUP STATUS REGISTER AND DMA
         000000G
147 0506 012704          MOV     #LPSDMA,R4      ;    REGISTER.
         000000G
148 0512 012710          MOV     #6,(R0)         ;NOW SET UP THE DMA
         000006
149 0516 011214          MOV     (R2),(R4)       ;DMA CURRENT ADDRESS
150 0520 012710          MOV     #4,(R0)
         000004
151 0524 016714          MOV     RTSNPT,(R4)     ;DMA WORD COUNT
         177262
152 0530 005414          NEG     (R4)            ;TWO'S COMPLEMENT
153 0532 012710          MOV     #2,(R0)         ;DMA STATUS REGISTER
         000002
154 0536 012714          MOV     #10000,(R4)     ;DMA ENABLE BIT
         010000
155 0542 110337 RTS09:   MOVB    R3,@#RTSON      ;INDICATE AN RTS IS IN PROGRESS
         000000G
156 0546 010367          MOV     R3,RTSCSR       ;SAVE CSR SETTING FOR INTERRUPT HANDLER
         177252
157 0552 010337          MOV     R3,@#LPSAD      ;START UP THE A/D OPERATON
         000000G
158 0556 000137 RTS10:   JMP     @#CKRPAR        ;CHECK LAST TOKEN EQUAL TO A RIGHT
         000000G
159                                              ;    PARENS AND RETURN TO THE BASIC
160                                              ;    INTERPRETER.
161              ;
```

27


```
1                ;       .SBTTL  "LED" COMMAND PROCESSOR
2                ;
3                ;*****************************************************************
4                ;
5                ; "LED" COMMAND PROCESSOR
6                ;
7                ; BASIC FORM:   CALL "LED"(VAR)
8                ;
9                ; PURPOSE:  LED WILL DISPLAY THE FLOATING POINT VALUE OF THE
10               ;               VARIABLE "VAR".  UP TO 6 POSITIVE OR 5 NEGATIVE
11               ;               DIGITS CAN BE DISPLAYED IN THE LEDS.  AN OPTIONAL
12               ;               DECIMAL POINT MAY ALSO BE INCLUDED.  NUMBERS WHICH
13               ;               ARE UNABLE TO BE DISPLAYED (VIZ,, "E" NUMBERS AND
14               ;               NEGATIVE 6 DIGITS ONES) ARE SHOWN BY ALL MINUS
15               ;               SIGNS IN THE LEDS.
16               ;
17 00562         LED:                             ;ENTRY POINT TO PROCESSOR
18 00562 004737          JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
         000000G
19                                               ;    FOR ENOUGH AREA TO WORK
20                                               ;    WITH AND THAT FIRST TOKEN
21                                               ;    IS INDEED A LEFT PARENS.
22 00566 000000G         +GETNUM                 ;GET NUMERIC FROM COMMAND ("VAR")
23 00570 000572'         ,+2                     ;LEAVE POLISH MODE
24 00572 005067          CLR     CNT             ;CHARACTER COUNTER
         177230
25 00576 012765          MOV     #LEDRES,T3(R5)  ;WHERE RESULTS WILL END UP
         000030'
         000062
26 00604 004737          JSR     PC,@#NUMSGN     ;GENERATE CHARACTER STRING
         000000G
27 00610 000722'         ,WORD   LEDOUT          ;ROUTINE TO PROCESS CHARACTERS
28 00612 016500          MOV     T3(R5),R0       ;GET LAST DIGIT POSITION +1
         000062
29 00616 012702          MOV     #LEDRES+6,R2    ;WHERE RESULT WILL END UP +1
         000036'
30 00622 026727 LED1:    CMP     CNT,#6          ;GOOD # GENERATED?
         177200
         000006
31 00630 003010          BGT     LED3            ;NO:  INSERT A "=" SIGN THEN
32 00632 020227 LED2:    CMP     R2,#LEDRES      ;INSERT NEXT CHARACTER IF SPACE IS
         000030'
33                                               ;    AVAILABLE.
34 00636 001413          BEQ     LED5            ;ALL DONE, NOW PUT THEM UP ON THE
35                                               ;    LED'S.
36 00640 020027          CMP     R0,#LEDRES      ;GET NEXT CHARACTER IF ONE EXISTS
         000030'
37 00644 001405          BEQ     LED4            ;NO:  FILL WITH BLANKS THEN
38 00646 114042          MOVB    -(R0),-(R2)     ;COPY ONE CHARACTER OVER
39 00650 000764          BR      LED1            ;WORK ON NEXT CHARACTER
40               ;
41 00652 012700 LED3:    MOV     #MINUS+1,R0     ;INSERT "=" SIGN FOR BAD NUMBER
         000050'
42 00656 000765          BR      LED2
43               ;
44 00660 012700 LED4:    MOV     #BLANK+7,R0     ;INSERT LEADING BLANKS
         000047'
```

```
45 00664 000762          BR      LED2
46                       ;
47 00666 012700 LED5:    MOV     #LEDRES+6,R0    ;WHERE TO START PULLING
         000036'
48 00672 005046          CLR     -(SP)           ;CLEAR LED REGISTER TO START
49 00674 114016 LED6:    MOVB    -(R0),(SP)      ;GET CHARACTER
50 00676 011637          MOV     (SP),@#LPSADB   ;DISPLAY CHAR
         000000G
51 00702 062716          ADD     #400,(SP)       ;POINTS TO NEXT DIGIT POSITION
         000400
52 00706 022716          CMP     #3000,(SP)      ;ALL DONE? (3000=400*6)
         003000
53 00712 002370          BGE     LED6            ;NO:  DO NEXT CHAR
54 00714 005726          TST     (SP)+           ;CLEAN UP STACK
55 00716 000137          JMP     @#CKRPAR        ;CHECK NEXT TOKEN EQUAL TO A RIGHT
         000000G
56                                               ;    PARENS AND RETURN TO THE BASIC
57                                               ;    INTERPRETER.
58                       ;
```

28


```
1                        ;
2                        ; ROUTINE TO SUPPORT LED COMMAND
3                        ;
4 000722                 LEDOUT:
5 000722 122700          CMPB    #'+,R0          ;"+" ?
         000053
6 000726 001421          BEQ     LEDXT           ;YES:  IGNORE
7 000730 122700          CMPB    #' ,R0          ;IGNORE BLANKS ALSO
         000040
8 000734 001416          BEQ     LEDXT
9 000736 122700          CMPB    #'E,R0          ;LETTER E?
         000105
10 00742 001432          BEQ     LEDBAD          ;YES:  CAN'T DISPLAY E NUMBERS
11 00744 122700          CMPB    #'.,R0          ;DECIMAL POINT?
         000056
12 00750 001411          BEQ     LEDOT1          ;YES:  CHECK TO SEE IF ITS THE
13                                               ;    FIRST CHARACTER.
14 00752 042700          BIC     #177760,R0      ;RID GARBAGE
         177760
15 00756 110075 LEDSTO:  MOVB    R0,@T3(R5)      ;SAVE CHARACTER
         000062
16 00762 005265          INC     T3(R5)          ;BUMP TO NEXT POSITION
         000062
17 00766 005267          INC     CNT             ;COUNT IT
         177034
18 00772 000207 LEDXT:   RETURN                  ;RETURN TO CALLER
19                       ;
20 00774 112700 LEDOT1:  MOVB    #20,R0          ;CHANGE CHAR TO LED "."
         000020
21 01000 022765          CMP     #LEDRES,T3(R5)  ;WAS THIS THE FIRST CHARACTER
         000030'
         000062
22                                               ;    RECEIVED?
23 01006 001763          BEQ     LEDSTO          ;YES:  STORE AS SINGLE CHAR
24 01010 005365          DEC     T3(R5)          ;BACK UP 1 CHAR POSITION IN ORDER
         000062
25 01014 152775          BISB    #20,@T3(R5)     ;    TO INSERT THE DECIMAL POINT.
         000020
         000062
26 01022 005265          INC     T3(R5)
         000062
27 01026 000207          RETURN                  ;RETURN TO CALLER
28                       ;
29 01030 012767 LEDBAD:  MOV     #7,CNT          ;INDICATE THAT NUMBER IS TOO LARGE
         000007
         176770
30 01036 000207          RETURN                  ;    TO HANDLE AND RETURN.
31                       ;
```

```
1                       ;       .SBTTL  A/D INTERRUPT PROCESSING ROUTINE
2                       ;
3                       ;**************************************************************
4                       ;
5                       ; A/D INTERRUPT PROCESSING ROUTINE
6                       ;
7 001040                ADCINT:
8 001040 004737         JSR     PC,@#REGSAV     ;SAVE R0, R2 AND R3 ON STACK
         000000G
9 001044 013703         MOV     @#LPSADB,R3     ;READ IN SAMPLE IN CASE WE
         000000G
10                                              ;NEED IT LATER.
11 01050 042737         BIC     #40,@#LPSAD     ;DISABLE "CLOCK OVERFLOW ENABLE"
         000040
         000000G
12 01056 105737         TSTB    @#RTSON         ;RTS OPERATION IN PROGRESS?
         000000G
13 01062 001006         BNE     ADC01           ;YES
14 01064 005037 ADC02:  CLR     @#LPSAD         ;DISABLE A/D AND EXIT
         000000G
15 01070 105037         CLRB    @#RTSON         ;(USED PRIMARILY BY DMA OPERATIONS)
         000000G
16 01074 000137 ADC09:  JMP     @#RESTOR        ;RESTORE REGISTERS AND EXIT
         000000G
17 01100 032767 ADC01:  BIT     #10,RTSMOD      ;DMA REQUEST FINISHED (IF ONE WAS
         000010
         176712
18                                              ;    PREVIOUSLY STARTED)?
19 01106 001410         BEQ     ADC03           ;NO
20 01110 016702         MOV     RTSBUF,R2       ;DMA IS DONE, ALLOW ARRAY TO BE
         176674
21 01114 016762         MOV     RTSEND,PUT(R2)  ;    ACCESSIBLE.
         176702
         000004
22 01122 011262         MOV     (R2),GET(R2)
         000006
23 01126 000756         BR      ADC02           ;DISABLE A/D AND EXIT
24                      ;
25 01130 016746 ADC03:  MOV     RTSNSC,-(SP)    ;NBR OF POINTS TO PROCESS
         176662
26 01134 016702         MOV     RTSBUF,R2       ;WHERE TO STORE RESULTS
         176650
27 01140 004737 ADC04:  JSR     PC,@#STODAT     ;STORE 1ST SAMPLE
         000000G
28 01144 032767         BIT     #4,RTSMOD       ;DUAL SAMPLE AND HOLD?
         000004
         176646
29 01152 001411         BEQ     ADC05           ;NO
30 01154 005237         INC     @#LPSAD         ;YES:  READ IN 2ND SAMPLE AND STORE IT
         000000G
31 01160 105737 ADC06:  TSTB    @#LPSAD         ;WAIT FOR IT
         000000G
32 01164 100375         BPL     ADC06
33 01166 013703         MOV     @#LPSADB,R3     ;GET A/D RESULT
         000000G
34 01172 004737         JSR     PC,@#STODAT     ;STORE IT
         000000G
```



```
35 01176 005316 ADC05:  DEC     (SP)            ;ALL SAMPLES TAKEN?
36 01200 003411         BLE     ADC07           ;YES:  TERMINATE THIS SET
37 01202 062737         ADD     #401,@#LPSAD    ;GO TO NEXT CHANNEL AND START CONVERSION
         000401
         000000G
38 01210 105737 ADC08:  TSTB    @#LPSAD         ;WAIT FOR IT
         000000G
39 01214 100375         BPL     ADC08
40 01216 013703         MOV     @#LPSADB,R3     ;GET RESULT
         000000G
41 01222 000746         BR      ADC04           ;STORE IT
42                      ;
43 01224 005726 ADC07:  TST     (SP)+           ;CLEAN UP STACK
44 01226 005367         DEC     RTSNPT          ;ALL POINTS TAKEN?
         176560
45 01232 003714         BLE     ADC02           ;YES:  STOP EVERYTHING THEN
46 01234 016737         MOV     RTSCSR,@#LPSAD  ;RESTORE A/D STATUS REGISTER
         176564
         000000G
47 01242 000714         BR      ADC09           ;RESTORE G,P, REGISTERS AND EXIT
48                      ;
49                      ;
50       000001'        .END                    ;END OF MODULE #1
```

SYMBOL TABLE

```
ADC     000050RG        ADCINT  001040R         ADC01   001100R
ADC02   001064R         ADC03   001130R         ADC04   001140R
ADC05   001176R         ADC06   001160R         ADC07   001224R
ADC08   001214R         ADC09   001074R         ADC1    000132R
ADC2    000064R         BAD     = 000010        BCDON = ****** G
BEG     = 000000        BLANK   000040R         BUFRES= ****** G
CKCMGN= ****** G        CKCOMA= ****** G        CKRPAR= ****** G
CNT     000026R         CONBCD= ****** G        DEL     = 000012
DRSNPT= ****** G        DRSON = ****** G        END     = 000002
ENTERP= ****** G        ERBUF = ****** G        ERNOR = ****** G
ERRARG= ****** G        ERRPDL= ****** G        ERRSYN= ****** G
EVAL  = ****** G        FAC1  = 000040          FAC2  = 000042
FLOAT = ****** G        GET   = 000006          GETADD= ****** G
GETBUF= ****** G        GETDAT= ****** G        GETNUM= ****** G
GETV  = ****** G        GETVAR= ****** G        HISTON= ****** G
INT   = ****** G        INTST = ****** G        LED     000562RG
LEDBAD  001134R         LEDOUT  000774R         LEDOUT  000722R
LEDRES  000630R         LEDSTO  000756R         LEDXT   000772R
LED1    000622R         LED2    000632R         LED3    000652R
LED4    000660R         LED5    000666R         LED6    000674R
LPSAD = ****** G        LPSADR= ****** G        LPSDMA= ****** G
LPSDR = ****** G        LPSIP = ****** G        LPSIVA= ****** G
MAXA    000000R         MINUS   000047R         NAMRAY= ****** G
NUMSGN= ****** G        PC    =%000007          POLENT= ****** G
POLRET= 000134          PS    = 177776          PUT   = 000004
REGSAV= ****** G        RESTOR= ****** G        RESTR2= ****** G
RETURN= 001207          RTS     000156RG        RTSBUF  002010R
RTSCSR  000024R         RTSEND  000022R         RTSMOD  000020R
RTSNPT  000012R         RTSNSC  000016R         RTSON = ****** G
RTSSC   000014R         RTS00   000340R         RTS01   000320R
RTS02   000346R         RTS03   000374R         RTS04   000410R
RTS05   000414R         RTS06   000424R         RTS07   000460R
RTS08   000502R         RTS09   000542R         RTS10   000556R
R0    =%000000          R1    =%000001          R2    =%000002
R3    =%000003          R4    =%000004          R5    =%000005
SAVER2= ****** G        SAVFAC= ****** G        SAVINT= ****** G
SP    =%000006          SS1SAV= 000024          SS2SAV= 000026
STODAT= ****** G        STOMXT= ****** G        STOVAR= ****** G
TABLE = ****** G        T3    = 000062          .LPAR = ****** G
.COMMA= ****** G        .RPAR = ****** G

. ABS.  000000          000
        001244          001
ERRORS DETECTED: 0
FREE CORE: 18459. WORDS

,LP:=LPS1
```

30

```
1               .TITLE  LPS2  V01-01
2               ;.SBTTL LPS MODULE #2
3               ;
4               ; DEC-11-LBEXA-B-LA3    BASIC KERNEL V02-01
5               ;
6               ; COPYRIGHT (C) 1973,1974
7               ;
8               ; DIGITAL EQUIPMENT CORPORATION
9               ; MAYNARD, MASSACHUSETTS 01754
10              ;
11              ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
12              ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
13              ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
14              ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
15              ; MAY APPEAR IN THIS DOCUMENT.
16              ;
17              ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
18              ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
19              ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
20              ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
21              ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
22              ;
23              ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
24              ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
25              ; WHICH IS NOT SUPPLIED BY DEC.
26              ;
27              ; WRITTEN BY:  RICK HULLY        FEB., 1973
28              ;
```



```
1               ;       .SBTTL  CONTENTS OF MODULE #2
2               ;
3               ; THE FOLLOWING COMMAND PROCESSORS EXIST IN THIS MODULE:
4               ;
5               ;       SETR
6               ;       SETC
7               ;       HIST
8               ;       WAIT
9               ;
```

```
1                       ;       .SBTTL  PROGRAM GLOBALS
2                       ;
3                       ; GLOBALS REQUIRED FOR COMMUNICATION WITH THE BASIC
4                       ; INTREPRETER.
5                       ;
6                               .GLOBL  ERRPDL,ERRSYN,ERRARG
7                               .GLOBL  EVAL,GETVAR,STOVAR
8                               .GLOBL  INT,.COMMA,.RPAR
9                               .GLOBL  NUMSGN,.LPAR,$IR,$MLR
10                              .GLOBL  $POLSH
11                      ;
12                      ; GLOBALS REQUIRED FOR COMMUNICATIONS WITH THE VARIOUS
13                      ; LPS MODULES.
14                      ;
15                              .GLOBL  TABLE,NARRAY,FLOAT
16                              .GLOBL  POLENT,GETADD,GETNUM
17                              .GLOBL  CKCMGN,INTST,GETV
18                              .GLOBL  GETBUF,REGSAV,ERNOR
19                              .GLOBL  STORXT,CKRPAR,ERBUF
20                              .GLOBL  CKCOMA,ENTERP,STODAT
21                              .GLOBL  GETDAT,RTSON,DRSON
22                              .GLOBL  SAVER2,SAVFAC,RESTR2
23                              .GLOBL  RESTOR,HISTON,BCDON
24                              .GLOBL  CONBCD,BUFRES,DRSNPT
25                              .GLOBL  SAVINT,DRSBUF
26                      ;
27                      ; GLOBALS REQUIRED FOR COMMAND PROCESSORS
28                      ;
29                              .GLOBL  SETR,SETC,HIST,WAIT
30                      ;
31                      ; GLOBALS FOR DEVICE ADDRESSES
32                      ;
33                              .GLOBL  LPSCKS,LPSPB,LPSDRS,LPSDIB
34                      ;
35                      ; GLOBALS FOR INTERRUPT PRIORITY AND VECTOR DEFINITION.
36                              .GLOBL CKLIVA,CKLIP
```



```
1                       ;       .SBTTL  REGISTER ASSIGNMENTS AND EQUATES
2                       ;
3       000000 R0       =       %0
4       000001 R1       =       %1
5       000002 R2       =       %2
6       000003 R3       =       %3
7       000004 R4       =       %4
8       000005 R5       =       %5
9       000006 SP       =       %6
10      000007 PC       =       %7
11                      ;
12                      ; GENERAL EQUATES
13                      ;
14      177776 PS       =       -2              ;PS = PROCESSOR STATUS WORD
15      000207 RETURN   =       207             ;207 = RTS PC
16      000134 POLRET   =       134             ;134 = JMP @(R4)+
17                      ;
18                      ; BASIC USER'S EQUATES
19                      ;
20      000022 VARSAV   =       22              ;VAR SAVE FOR ASSIGNMENT
21      000024 SS1SAV   =       24              ;SS1 SAVE FOR ASSIGNMENT
22      000026 SS2SAV   =       26              ;SS2 SAVE FOR ASSIGNMENT
23      000040 FAC1     =       40              ;HIGH ORDER FLOATING VALUE
24      000042 FAC2     =       42              ;LOW  ORDER FLOATING VALUE
25      000046 R1SAVE   =       46              ;R1 SAVE LOCATION
26      000062 T3       =       62              ;SHORT TERM TEMPORARY
27                      ;
28                      ; BUFFER DESCRIPTOR EQUATES
29                      ;
30      000000 BEG      =       0               ;OFFSET TO BEGINNING OF BUFFER POINTER
31      000002 END      =       2               ;OFFSET TO END OF BUFFER POINTER
32      000004 PUT      =       4               ;OFFSET TO PUT DATA POINTER
33      000006 GET      =       6               ;OFFSET TO GET DATA POINTER
34      000010 BAD      =       10              ;OFFSET TO BAD DATA FLAG
35      000012 DEL      =       12              ;OFFSET TO DISPLAY DELTA X
36                      ;
37                      ; LIMIT TEST
38                      ;
39 00000 000007 MAXC:   .WORD   7               ;7
40 00002 000007         .WORD   7               ;7
41 00004 177777         .WORD   177777          ;65535
42                      ;
43                      ;
44                      ; NON-REENTRANT VARIABLES
45                      ;
46 00006 000000 HSTNPT: .WORD   0               ;HIST COMMAND (NBR OF POINTS)
47 00010 000000 HSTBUF: .WORD   0               ;HIST COMMAND (BUFFER DESCRIPTOR
48                                              ;    ADDRESS)
49 00012    000 CKSTFG: .BYTE   0,0             ;CLOCK/ST#1 FLAGS FOR WAIT COMMAND
   00013    000
50                      ;
```

```
1                       ;       .SBTTL  "SETR" COMMAND PROCESSOR
2                       ;
3                       ;**************************************************************
4                       ;
5                       ; "SETR" COMMAND PROCESSOR
6                       ;
7                       ; BASIC FORM:   CALL "SETR"(RATE,MODE,PRESET)
8                       ;
9                       ; PURPOSE:  SET CLOCK RATE WILL SET THE CLOCK RUNNING AT THE
10                      ;               DESIGNATED "RATE" AND IN THE SPECIFIED
11                      ;               "MODE".  THE "PRESET" VALUE IS THE CLOCK COUNTER
12                      ;               VALUE.  THE INTERRUPT ENABLE WILL ALWAYS BE
13                      ;               SET TO A ONE.
14                      ;
15                      ;               VALUES OF "RATE"
16                      ;                 0  NO RATE SELECTED
17                      ;                 1  1 MHZ
18                      ;                 2  100 KHZ
19                      ;                 3  10 KHZ
20                      ;                 4  1 KHZ
21                      ;                 5  100 HZ
22                      ;                 6  SCHMITT TRIGGER #1 (MEANINGFUL ONLY
23                      ;                      FOR HIST OPERATIONS),
24                      ;                 7  LINE FREQUENCY (50 HZ OR 60 HZ)
25                      ;
26                      ;               VALUES OF "MODE"
27                      ;                 0  SINGLE INTERVAL MODE,  COUNTER COUNTS FROM
28                      ;                      PRESET VALUE TO OVERFLOW AND STOPS.
29                      ;                 1  REPEATED INTERVAL MODE,  COUNTER COUNTS FROM
30                      ;                      PRESET VALUE TO OVERFLOW, TRANSFERS
31                      ;                      BUFFER/PRESET TO COUNTER, AND BEGINS
32                      ;                      AGAIN.
33                      ;                 2  EXTERNAL EVENT TIMING MODE,  THE COUNTER IS
34                      ;                      FREE RUNNING, AND A PULSE FROM ST #2 WILL
35                      ;                      TRANSFER CONTENTS OF COUNTER TO BUFFER/
36                      ;                      PRESET AND THEN CONTINUE COUNTING.
37                      ;                 3  EVENT TIMING FROM ZERO BASE MODE IS THE
38                      ;                      SAME EXCEPT THAT WHEN THE TRANSFER OF THE
39                      ;                      COUNTER TO THE BUFFER/PRESET IS DONE,
40                      ;                      THE COUNTER IS CLEARED AND THE COUNT BEGINS
41                      ;                      FROM ZERO.
42                      ;               MODE+4  START CLOCK ONLY WHEN ST#1 FIRES.
43                      ;
44 00014                SETR:                           ;ENTRY POINT TO PROCESSOR
45 00014 012737         MOV     #CKLINT,@#CKLIVA        ;INTERRUPT ENTRY POINT.
         000540'
         000000G
46 00022 012737         MOV     #240,@#CKLIP            ;PROCESSOR PRIORITY.
         000240
         000000G
47 00030 012765         MOV     #MAXC,T3(R5)    ;FOR LIMIT TESTS ON RATE, MODE, AND
         000000'
         000062
48                                              ;       AND PRESET.
49 00036 004737         JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
         000000G
50                                              ;       FOR ENOUGH AREA TO WORK
```



```
51                                              ;       WITH AND THAT FIRST TOKEN
52                                              ;       IS INDEED A LEFT PARENS.
53 00042 000000G        +GETNUM                 ;GET "RATE" IN COMMAND
54 00044 000000G        +INTST                  ;INTEGERIZE IT AND TEST <=7
55 00046 000000G        +SAVINT                 ;SAVE IN ON THE STACK
56 00050 000000G        +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
57                                              ;       AND GET NEXT NUMERIC ("MODE")
58 00052 000000G        +INTST                  ;INTEGERIZE IT AND TEST <=7
59 00054 000000G        +SAVINT                 ;SAVE RESULT ON THE STACK.
60 00056 000000G        +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
61 00060 000000G        +INTST                  ;       GET NEXT NUMERIC, AND TEST <=65535.
62 00062 000064'        .+2                     ;LEAVE POLISH MODE
63 00064 005037         CLR     @#LPSCKS        ;TURN OFF CLOCK MOMENTARILY
         000000G
64 00070 005465         NEG     FAC2(R5)        ;TAKE TWO'S COMPLEMENT OF THE PRESET
         000042
65 00074 016537         MOV     FAC2(R5),@#LPSPB ;      VALUE AND ENTER INTO THE BUFFER
         000042
         000000G
66                                              ;       PRESET REGISTER.
67 00102 012600         MOV     (SP)+,R0        ;GET MODE
68 00104 012602         MOV     (SP)+,R2        ;GET RATE
69 00106 006302         ASL     R2              ;NORMALIZE TO BITS 01-03 OF WORD
70 00110 000300         SWAB    R0              ;PUT MODE IN LEFT BYTE
71 00112 052700         BIS     #101,R0         ;ASSUME NO ST#1 CLOCK STARTS
         000101
72 00116 032700         BIT     #2000,R0        ;ST #1 REQUESTED?
         002000
73 00122 001404         BEQ     SETR1           ;NO
74 00124 042700         BIC     #2001,R0        ;CLEAR OUT ST BIT AND START BIT
         002001
75 00130 052700         BIS     #20000,R0       ;ADD IN REAL ST BIT IN STATUS REGISTER
         020000
76 00134                SETR1:
77 00134 105067         CLRB    CKSTFG          ;CLEAR CLOCK FLAG BEFORE STARTING
         177652
78 00140 050002         BIS     R0,R2           ;COMBINE MODE + RATE + ST#1 + START
79 00142 010237         MOV     R2,@#LPSCKS     ;NOW START UP CLOCK
         000000G
80 00146 000137         JMP     @#CKRPAR        ;CHECK LAST TOKEN EQUAL TO A RIGHT
         000000G
81                                              ;       PARENS AND RETURN TO THE BASIC
82                                              ;       INTERPRETER.
83                      ;
```

```
                        ;       .SBTTL  "SETC" COMMAND PROCESSOR
                        ;
                        ;*****************************************************************
                        ;
                        ; "SETC" COMMAND PROCESSOR
                        ;
                        ; BASIC FORM:   CALL "SETC"(RATE,TIME)
                        ;
                        ; PURPOSE:  SET CLOCK TO THAT SPECIFIED BY "RATE" AND "TIME",
                        ;               THE CLOCK STATUS REGISTER IS SET TO "RATE" AND
                        ;               WILL RUN FOR "TIME" SECONDS.  A CLOCK INTERRUPT
                        ;               WILL THEN OCCUR WHICH MAY BE USED TO INITIATE
                        ;               ANY OF THE CLOCK CONTROLLED FUNCTIONS.  THE
                        ;               TIME ARGUMENT IS EVALUATED AS TICKS = TIME IN SECS
                        ;               TIMES CLOCK RATE SPECIFIED IN "RATE", E.G. IF
                        ;               CLOCK RATE WAS 10KHZ, THEN TICKS = TIME IN SECONDS
                        ;               TIMES 10KHZ.  THE "TICKS" ARE ENTERED INTO
                        ;               THE CLOCK PRESET/BUFFER REGISTER.  THE CLOCK
                        ;               ALWAYS RUNS IN MODE ZERO.
                        ;
                        ;               LEGAL VALUES OF RATE ARE:  4, 5, AND 7 (SEE SETR
                        ;               FOR EXPLANATION OF MODES).
                        ;
000152                  SETC:                           ;ENTRY POINT TO PROCESSOR
000152 012737           MOV     #CKLINT,@#CKLIVA
000540'
000060G
000160 012737           MOV     #240,@#CKLIP
000240
000064G
000166 012765           MOV     #MAXC+2,T3(R5)  ;LIMIT TESTING OF "RATE" AND "TIME"
000002'
000062
000174 004737           JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
000000G
                                                ;       FOR ENOUGH AREA TO WORK
                                                ;       WITH AND THAT FIRST TOKEN
                                                ;       IS INDEED A LEFT PARENS.
000200 000000G          +GETNUM                 ;GET "RATE" FROM COMMAND
000202 000000G          +INTST                  ;INTEGERIZE AND TEST <= 7
000204 000000G          +SAVINT                 ;SAVE RATE ON STACK
000206 000000G          +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A
                                                ;       COMMA AND GET NEXT NUMERIC
                                                ;       FROM COMMAND STRING.  ("TIME").
000212'                 .+2                     ;EXIT POLISH MODE
000212 010165           MOV     R1,R1SAVE(R5)   ;PERSERVE R1
000046
000216 005037           CLR     @#LPSCKS        ;TURN OFF CLOCK MOMENTARILY
000060G
000222 021627           CMP     (SP),#4         ;RATE MUST BE MODES 4,5, OR 7
000004
000226 002451           BLT     ERARG           ;ILLEGAL RATE SPECIFIED
000230 021627           CMP     (SP),#6         ;RATE OF 6 IS ALSO ILLEGAL
000006
000234 001446           BEQ     ERARG
000236 006316           ASL     (SP)            ;RATE = RATE * 2  (TO POSITION IN PLACE
                                                ;       FOR THE CLOCK STATUS REGISTER).
```



```
000240 011602           MOV     (SP),R2         ;CONVERT CODE TO A FLOATING POINT NUMBER
000242 005046           CLR     -(SP)           ;2ND HALF OF FP# IS ALWAYS ZERO
000244 016246           MOV     FPNBR-10(R2),-(SP)
000346'
000250 016546           MOV     FAC2(R5),-(SP)  ;PUT "TIME" ON STACK
000042
000254 016546           MOV     FAC1(R5),-(SP)
000040
000260 001005           BNE     SETC1           ;TIME IS ALREADY FLOATED
000262 005726           TST     (SP)+           ;FLOAT "TIME"
000264 004437           JSR     R4,@#$POLSH
000000G
000270 000000G          +$IR
000272 000274'          .+2                     ;EXIT POLISH MODE
000274 004437 SETC1:    JSR     R4,@#$POLSH
000000G
000300 000000G          +$MLR                   ;MULTIPLY "TIME" * "RATE"
000302 000304'          .+2                     ;EXIT POLISH MODE
000304 012665           MOV     (SP)+,FAC1(R5)  ;PUT RESULT IN FAC
000040
000310 012665           MOV     (SP)+,FAC2(R5)
000042
000314 004737           JSR     PC,@#ENTERP
000000G
000320 000000G          +INTST                  ;INTERGIZE AND NEGATE NUMBER
000322 005465           NEG     FAC2(R5)
000042
000326 016537           MOV     FAC2(R5),@#LPSPB ;PUT INTO BLOCK BUFFER/PRESET
000042
000060G
                                                ;       REGISTER.
000334 012602           MOV     (SP)+,R2        ;GET "RATE"
000336 012700           MOV     #101,R0         ;CLOCK + INTERRUPT ENABLE BITS
000101
000342 016501           MOV     R1SAVE(R5),R1   ;RESTORE R1
000046
000346 000167           JMP     SETR1           ;START UP CLOCK AND CHECK LAST
177562
                                                ;       TOKEN EQUAL TO A RIGHT PARENS,
                                                ;       THEN RETURN TO THE BASIC
                                                ;       INTERPRETER.
                        ;
000352 000137 ERARG:    JMP     @#ERRARG        ;ILLEGAL ARGUMENT
000000G
                        ;
000356 042572 FPNBR:    .WORD   042572          ;FLOATING POINT 1000
000360 041710           .WORD   041710          ;FLOATING POINT 100
000362 000000           .WORD   0               ;FLOATING POINT 0
                        .IFNDF  CYC50
000364 041510           .WORD   041510          ;FLOATING POINT 60
                        .ENDC
                        .IFDF   CYC50
                        .WORD   041500          ;FLOATING POINT 50
                        .ENDC
                        ;
```

```
                        ;       .SBTTL  "HIST" COMMAND PROCESSOR
                        ;
                        ;*****************************************************************
                        ;
                        ; "HIST" COMMAND PROCESSOR"
                        ;
                        ; BASIC FORM:   CALL "HIST"(BUF,NPTS)
                        ;
                        ; PURPOSE:  HISTOGRAM - TIMED SCHMITT TRIGGER.  THE HIST FUNCTION
                        ;               INPUTS "NPTS" VALUES FROM THE CLOCK PRESET/BUFFER
                        ;               INTO THE SPECIFIED BUFFER, BUF, WHEN ST #2 FIRES.
                        ;               THE CLOCK MODE MUST BE PUT IN 2 OR 3 TO BE MEANINGFUL.
                        ;               THE RDB FUNCTION IS USED TO RETRIEVE THE DATA.
                        ;               THE BUFFER IS ALWAYS CIRCULAR AND THE DATA POINTERS
                        ;               INITIALLY RESET BEFORE THE SAMPLING BEGINS.
                        ;
17 00366                HIST:                           ;ENTRY POINT TO PROCESSOR
18 00366 012765         HIST:   MOV     #MAXC+4,T3(R5)  ;FOR LIMIT CHECKS ON NPTS
         000004'
         000062
19 00374 105067         CLRB    HISTON          ;TURN OFF ANY PREVIOUSLY INITIATED
         000000G
20                                              ;       HISTOGRAM SAMPLING.
21 00400 004737         JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
         000000G
22                                              ;       FOR ENOUGH AREA TO WORK
23                                              ;       WITH AND THAT FIRST TOKEN
24                                              ;       IS INDEED A LEFT PARENS.
25 00404 000000G        +GETBUF                 ;GET BUFFER ADDRESS FROM COMMAND
26                                              ;       AND CHECK IT AGAINST THE
27                                              ;       DEFINITION TABLE.
28 00406 000000G        +BUFRES                 ;RESET BUFFER POINTERS
29 00410 000000G        +SAVER2                 ;SAVE BUFFER ADDRESS
30 00412 000000G        +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
31                                              ;       AND THE NEXT NUMERIC FROM
32                                              ;       FROM THE COMMAND, ("NPTS")
33 00414 000000G        +INTST                  ;INTERGIZE RESULT AND TEST <=65535
34 00416 000420'        ,+2                     ;LEAVE POLISH MODE
35 00420 016567         MOV     FAC2(R5),HSTNPT ;SAVE "NPTS"
         000042
         177360
36 00426 012667         MOV     (SP)+,HSTBUF    ;GET BUFFER DESCRIPTOR ADDRESS
         177356
37 00432 105267         INCB    HISTON          ;ARM TIMED SAMPLING
         000000G
38 00436 000137         JMP     @#CKRPAR        ;CHECK NEXT TOKEN EQUAL TO RIGHT
         000000G
39                                              ;       PARENS AND RETURN TO THE
40                                              ;       BASIC INTERPRETER.
41                      ;
```



```
                        ;       .SBTTL  "WAIT" COMMAND PROCESSOR
                        ;
                        ;*****************************************************************
                        ;
                        ; "WAIT" COMMAND PROCESSOR
                        ;
                        ; BASIC FORM:   CALL "WAIT"(N)
                        ;
                        ; PURPOSE:  DISABLE FURTHER PROGRAM EXECUTION AND WAIT UNTIL
                        ;               THE SPECIFIED EVENT "N" OCCURS.  "N" IS DEFINED
                        ;               AS FOLLOWS:
                        ;
                        ;               N=0     WAIT FOR CLOCK OVERFLOW
                        ;               N=1     WAIT FOR ST #1 TO FIRE
                        ;               N=2     WAIT FOR N=0 OR N=1
                        ;               N=0,1,2 RETURNS IMMEDIATELY
                        ;
18 00442                WAIT:                           ;ENTRY POINT TO PROCESSOR
19 00442 012765         WAIT:   MOV     #MAXC+4,T3(R5)  ;FOR LIMIT TESTS ON N
         000004'
         000062
20 00450 004737         JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
         000000G
21                                              ;       FOR ENOUGH AREA TO WORK
22                                              ;       WITH AND THAT FIRST TOKEN
23                                              ;       IS INDEED A LEFT PARENS.
24 00454 000000G        +GETNUM                 ;GET "N"
25 00456 000000G        +INTST                  ;INTEGERIZE IT
26 00460 000462'        ,+2                     ;LEAVE POLISH MODE
27 00462 026527         CMP     FAC2(R5),#2     ;N>2 ALWAYS RETURNS IMMEDIATELY
         000042
         000002
28 00470 101015         BHI     WAIT4
29 00472 012702         MOV     #CKSTFG,R2      ;ADDRESS OF CLOCK/ST FLAG
         000012'
30 00476 016500         MOV     FAC2(R5),R0     ;GET "N"
         000042
31 00502 001412         BEQ     WAIT1           ;N=1 MEANS CLOCK OVERFLOW
32 00504 005202         INC     R2              ;ASSUME ST FOR NOW
33 00506 006000         ROR     R0              ;N=1 MEANS ST OVERFLOW
34 00510 103407         BCS     WAIT1
35 00512 005767 WAIT3:  TST     CKSTFG          ;N=2 MEANS ST OR CLOCK OVERFLOW
         177274
36 00516 001775         BEQ     WAIT3
37 00520 005067         CLR     CKSTFG          ;CLEAR ALL FLAGS NOW
         177266
38 00524 000137 WAIT4:  JMP     @#CKRPAR        ;CHECK NEXT TOKEN EQUAL TO A
         000000G
39                                              ;       RIGHT PARENS AND RETURN TO
40                                              ;       THE BASIC INTERPRETER.
41 00530 105712 WAIT1:  TSTB    (R2)            ;WAIT FOR SPECIFIED FLAG ONLY
42 00532 001776         BEQ     WAIT1
43 00534 105012         CLRB    (R2)            ;CLEAR FLAG FOR NEXT TIME
44 00536 000772         BR      WAIT4
45                      ;
```

```
                ;       .SBTTL  LPS CLOCK INTERRUPT ROUTINE
                ;
                ;*****************************************************************
                ;
                ; LPS CLOCK INTERRUPT ROUTINE
                ;
000540          CKLINT:
000540 004737           JSR     PC,@#REGSAV     ;SAVE R0, R2, AND R3 ON STACK
       000000G
000544 012700           MOV     #CKSTFG,R0      ;SET ST/CLOCK FLAG
       000012'
000550 105737           TSTB    @#LPSCKS        ;WAS THIS A CLOCK OVERFLOW?
       000000G
000554 100401           BMI     CLK0            ;YES:  SET CLOCK FLAG
000556 005200           INC     R0              ;SET ST FLAG
000560 152710 CLK0:     BISB    #1,(R0)         ;SET APPROPRIATE FLAG
       000001
000564 032737           BIT     #1400,@#LPSCKS  ;MODE ZERO OPERATION?
       001400
       000000G
000572 001003           BNE     CLK4            ;NO
000574 042737           BIC     #1717,@#LPSCKS  ;YES:  DISABLE CLOCK THEN
       001717
       000000G
000602 105767 CLK4:     TSTB    HISTON          ;HIST OPERATION IN PROGRESS?
       000000G
000606 001413           BEQ     CLK1            ;NO:  TEST DRS THEN
000610 013703           MOV     @#LPSPB,R3      ;READ PRESET/BUFFER REGISTER
       000000G
000614 016702           MOV     HSTBUF,R2       ;GET HIST BUFFER ADDRESS
       177170
000620 004737           JSR     PC,@#STODAT     ;STORE DATA IN USER'S BUFFER
       000000G
000624 005367           DEC     HSTNPT          ;ALL POINTS TAKEN?
       177156
000630 001002           BNE     CLK1            ;NO
000632 105067           CLRB    HISTON          ;CLEAR HIST DONE FLAG
       000000G
000636 105737 CLK1:     TSTB    @#DRSON         ;DRS OPERATON IN PROGRESS?
       000000G
000642 100023           BPL     CLK2            ;NO:  EXIT
000644 052737           BIS     #2,@#LPSDRS     ;READ DIGITAL INPUT REGISTER
       000002
       000000G
000652 013703           MOV     @#LPSDIB,R3
       000000G
000656 105737           TSTB    @#BCDON         ;CONVERT TO BCD?
       000000G
000662 001002           BNE     CLK3            ;NO:  TREAT AS A PURE BINARY NUMBER
000664 004737           JSR     PC,@#CONBCD     ;CONVERT TO BCD
       000000G
000670 013702 CLK3:     MOV     @#DRSBUF,R2     ;GET DRS BUFFER ADDRESS
       000000G
000674 004737           JSR     PC,@#STODAT     ;STORE DATA IN USER'S BUFFER
       000000G
000700 005337           DEC     @#DRSNPT        ;ALL POINTS TAKEN
       000000G
```



```
000704 001002           BNE     CLK2            ;NO
000706 105037           CLRB    @#DRSON         ;CLEAR DRS DONE FLAG
       000000G
000712 000137 CLK2:     JMP     @#RESTOR        ;RESTORE REGISTERS AND EXIT
       000000G
                ;
000001'         .END                            ;END OF MODULE #2
```

```
LPS2  V01-01    RT-11 MACRO VM02-09   16-OCT-74 02:00:38 PAGE 9+
SYMBOL TABLE

BAD    = 000010          BCDON = ****** G        BEG    = 000000
BUFRES= ****** G         CKCMGN= ****** G        CKCOMA= ****** G
CKLINT  000540R          CKLIP = ****** G        CKLIVA= ****** G
CKRPAR= ****** G         CKSTFG  000012R         CLK0    000560R
CLK1    000636R          CLK2    000712R         CLK3    000670R
CLK4    000602R          CONBCD= ****** G        DEL    = 000012
DRSBUF= ****** G         DRSNPT= ****** G        DRSON = ****** G
END    = 000002          ENTERP= ****** G        ERARG   000352R
ERBUF = ****** G         ERNOR = ****** G        ERRARG= ****** G
ERRPDL= ****** G         ERRSYN= ****** G        EVAL   = ****** G
FAC1   = 000040          FAC2   = 000042         FLOAT = ****** G
FPNBR   000356R          GET    = 000006         GETADD= ****** G
GETBUF= ****** G         GETDAT= ****** G        GETNUM= ****** G
GETV   = ****** G        GETVAR= ****** G        HIST    000366RG
HISTON= ****** G         HSTBUF  000010R         HSTNPT  000006R
INT    = ****** G        INTST = ****** G        LPSCKS= ****** G
LPSDIB= ****** G         LPSDRS= ****** G        LPSPB = ****** G
MAXC    000000R          NARRAY= ****** G        NUMSGN= ****** G
PC     =%000007          POLENT= ****** G        POLRET= 000134
PS     = 177776          PUT    = 000004         REGSAV= ****** G
RESTOR= ****** G         RESTR2= ****** G        RETURN= 000207
RTSON = ****** G         R0     =%000000         R1     =%000001
R1SAVE= 000046           R2     =%000002         R3     =%000003
R4     =%000004          R5     =%000005         SAVER2= ****** G
SAVFAC= ****** G         SAVINT= ****** G        SETC    000152RG
SETC1   000274R          SETR    000014RG        SETR1   000134R
SP     =%000006          SS1SAV= 000024          SS2SAV= 000026
STODAT= ****** G         STORXT= ****** G        STOVAR= ****** G
TABLE = ****** G         T3     = 000062         VARSAV= 000022
WAIT    000442RG         WAIT1   000530R         WAIT3   000512R
WAIT4   000524R          $IR    = ****** G       $MLR   = ****** G
$POLSH= ****** G         .COMMA= ****** G        .LPAR = ****** G
.RPAR = ****** G
. ABS.  000000     000
        000716     001
ERRORS DETECTED: 0
FREE CORE: 18167. WORDS

.LP:=LPS2
```

```
LPS3  V01-01    RT-11 MACRO VM02-09   16-OCT-74 02:02:11 PAGE 1


1               .TITLE  LPS3  V01-01
2               ;.SBTTL LPS MODULE #3
3               ;
4               ; DEC-11-LBEXA-B-LA4    BASIC KERNEL V02-01
5               ;
6               ; COPYRIGHT (C) 1973,1974
7               ;
8               ; DIGITAL EQUIPMENT CORPORATION
9               ; MAYNARD, MASSACHUSETTS 01754
10              ;
11              ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
12              ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
13              ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
14              ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
15              ; MAY APPEAR IN THIS DOCUMENT.
16              ;
17              ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
18              ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
19              ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
20              ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
21              ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
22              ;
23              ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
24              ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
25              ; WHICH IS NOT SUPPLIED BY DEC.
26              ;
27              ; WRITTEN BY:  RICK HULLY       FEB., 1973
28              ;
```

```
1               ;       .SBTTL  CONTENTS OF MODULE #3
2               ;
3               ; THE FOLLOWING COMMAND PROCESSORS EXIST IN THIS MODULE:
4               ;
5               ;       DIR
6               ;       DOR
7               ;       DRS
8               ;       REL
9               ;
```



```
1               ;       .SBTTL  PROGRAM GLOBALS
2               ;
3               ; GLOBALS REQUIRED FOR COMMUNICATION WITH THE BASIC
4               ; INTREPRETER.
5               ;
6                       .GLOBL  ERRPDL,ERRSYN,ERRARG
7                       .GLOBL  EVAL,GETVAR,STOVAR
8                       .GLOBL  INT,,COMMA,,RPAR
9                       .GLOBL  NUMSGN,,LPAR
10              ;
11              ; GLOBALS REQUIRED FOR COMMUNICATIONS WITH THE VARIOUS
12              ; LPS MODULES.
13              ;
14                      .GLOBL  TABLE,NARRAY,FLOAT
15                      .GLOBL  POLENT,GETADD,GETNUM
16                      .GLOBL  CKCMGN,INTST,GETV
17                      .GLOBL  GETBUF,REGSAV,ERNOR
18                      .GLOBL  STORXT,CKRPAR,ERBUF
19                      .GLOBL  CKCOMA,ENTERP,STODAT
20                      .GLOBL  GETDAT,RTSON,DRSON
21                      .GLOBL  SAVER2,SAVFAC,RESTR2
22                      .GLOBL  RESTOR,HISTON,BCDON
23                      .GLOBL  CONBCD,BUFRES,DRSNPT
24                      .GLOBL  SAVINT,DRSBUF
25              ;
26              ; GLOBALS REQUIRED FOR COMMAND PROCESSORS
27              ;
28                      .GLOBL  DIR,DOR,DRS,REL
29              ;
30              ; GLOBALS FOR DEVICE ADDRESSES
31              ;
32                      .GLOBL  LPSDRS,LPSDIB,LPSDOR
33              ;
34              ; GLOBALS FOR INTERRUPT PRIORITY AND VECTOR ADRESS.
35                      .GLOBL DRSIVA,DRSIP
```

```
                ;       .SBTTL  REGISTER ASSIGNMENTS AND EQUATES
                ;
        000000 R0       =       %0
        000001 R1       =       %1
        000002 R2       =       %2
        000003 R3       =       %3
        000004 R4       =       %4
        000005 R5       =       %5
        000006 SP       =       %6
        000007 PC       =       %7
                ;
                ; GENERAL EQUATES
                ;
        177776 PS       =       =2              ;PS = PROCESSOR STATUS WORD
        000207 RETURN   =       207             ;207 = RTS PC
        000134 POLRET   =       134             ;134 = JMP @(R4)+
                ;
                ; BASIC USER'S EQUATES
                ;
        000022 VARSAV   =       22              ;VAR SAVE FOR ASSIGNMENT
        000024 SS1SAV   =       24              ;SS1 SAVE FOR ASSIGNMENT
        000026 SS2SAV   =       26              ;SS2 SAVE FOR ASSIGNMENT
        000040 FAC1     =       40              ;HIGH ORDER FLOATING VALUE
        000042 FAC2     =       42              ;LOW  ORDER FLOATING VALUE
        000056 T1       =       56              ;SHORT TERM TEMPORARY
        000060 T2       =       60              ;SHORT TERM TEMPORARY
        000062 T3       =       62              ;SHORT TERM TEMPORARY
                ;
                ; BUFFER DESCRIPTOR EQUATES
                ;
        000000 BEG      =       0               ;OFFSET TO BEGINNING OF BUFFER POINTER
        000002 END      =       2               ;OFFSET TO END OF BUFFER POINTER
        000004 PUT      =       4               ;OFFSET TO PUT DATA POINTER
        000006 GET      =       6               ;OFFSET TO GET DATA POINTER
        000010 BAD      =       10              ;OFFSET TO BAD DATA FLAG
        000012 DEL      =       12              ;OFFSET TO DISPLAY DELTA X
                ;
                ; LIMIT TEST
                ;
00000 177777 MAXD:      .WORD   177777          ;65535
00002 000002 MAXR:      .WORD   2               ;2
                ;
```



```
                ;       .SBTTL  "DIR" COMMAND PROCESSOR
                ;
                ;*****************************************************************
                ;
                ; "DIR" COMMAND PROCESSOR
                ;
                ; BASIC FORM:   CALL "DIR"(N,VAR,NEWCSR)
                ;
                ;PURPOSE:  READ DIGITAL INPUT REGISTER.  IF N=0, INPUT IS
                ;               FOUR BCD DIGITS CONVERTED TO FLOATING POINT.
                ;               IF N#0, THEN THE BINARY RESULT READ FROM THE
                ;               REGISTER IS DIRECTLY CONVERTED TO A FLOATING
                ;               POINT NUMBER.  THE REGISTER READ IS ACCOM-
                ;               PLISHED VIA AN "INTERNAL LOAD" REQUEST AND
                ;               DOES NOT RESPOND TO INTERRUPTS.  THE RESULT IS
                ;               PLACED IN "VAR", WHERE 0 <= VAR <= 65535,
                ;
                ;               THE NEW CSR REGISTER SETTING IS RETURNED IN NEWCSR.
                ;
00004           DIR:                            ;ENTRY POINT TO PROCESSOR
00004 004737            JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
      000000G
                                                ;    FOR ENOUGH AREA TO WORK
                                                ;    WITH AND THAT FIRST TOKEN
                                                ;    IS INDEED A LEFT PARENS.
00010 000000G           +GETNUM                 ;GET "N" IN COMMAND
00012 000000G           +SAVFAC                 ;SAVE FAC ON STACK
00014 000000G           +CKCOMA                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
00016 000000G           +GETV                   ;CALCULATE ADDRESS OF "VAR"
00020 000000G           +CKCOMA                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
00022 000024'           .+2                     ;LEAVE POLISH MODE
00024 052737            BIS     #2,@#LPSDRS     ;ISSUE AN INPUT LOAD COMMAND
      000002
      000000G
00032 013703            MOV     @#LPSDIB,R3     ;READ DIGITAL INPUT REGISTER
      000000G
00036 052626            BIS     (SP)+,(SP)+     ;WAS N=0? (I.E. READ BCD?)
00040 001002            BNE     DIR1            ;NO:  READ STRAIGHT BINARY
00042 004767            JSR     PC,CONBCD       ;CONVERT BCD TO BINARY
      000000G
00046 004737 DIR1:      JSR     PC,@#FLOAT      ;FLOAT RESULT
      000000G
00052 004737            JSR     PC,@#STOVAR     ;STORE RESULT IN "VAR"
      000000G
00056 004737            JSR     PC,@#ENTERP     ;GET ADDRESS WHERE STATUS REGISTER
      000000G
00062 000000G           +GETV                   ;    WILL END UP.
00064           DIR2:
00064 013703            MOV     @#LPSDRS,R3     ;GET STATUS REGISTER
      000000G
00070           DIR3:
00070 004737            JSR     PC,@#FLOAT      ;FLOAT IT
      000000G
00074 000137            JMP     @#STORXT        ;STORE RESULT, CHECK LAST TOKEN EQUAL
      000000G
                                                ;    TO A RIGHT PARENS AND RETURN
                                                ;    TO THE BASIC INTERPRETER.
```

```
47                ;
```



```
1                 ;       .SBTTL  "DOR" COMMAND PROCESSOR
2                 ;
3                 ;**********************************************************************
4                 ;
5                 ; "DOR" COMMAND PROCESSOR
6                 ;
7                 ; BASIC FORM:   CALL "DOR"(M,N,NEWDOR)
8                 ;
9                 ; PURPOSE:  WRITE DIGITAL OUTPUT REGISTER.  SELECTED BITS IN
10                ;               THE REGISTER MAY BE SET OR CLEARED.  IF M=0, SET
11                ;               BITS IN REGISTER.  IF M#0, CLEAR BITS IN
12                ;               REGISTER.  "N" IS THE BIT PATTERN TO SET OR CLEAR.
13                ;               THE NEW RESULT IN THE REGISTER IS RETURNED AS A
14                ;               FLOATING POINT NUMBER IN NEWDOR.
15                ;
16 00100          DOR1:                           ;ENTRY POINT TO PROCESSOR
17 00100 012765   DOR:    MOV     #MAXD,T3(R5)    ;FOR LIMIT TEST OF "N"
          000000'
          000062
18 00106 004737           JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
          000000G
19                                                ;    FOR ENOUGH AREA TO WORK
20                                                ;    WITH AND THAT FIRST TOKEN
21                                                ;    IS INDEED A LEFT PARENS.
22 00112 000000G          +GETNUM                 ;GET "M" IN COMMAND
23 00114 000000G          +SAVFAC                 ;SAVE FAC ON STACK
24 00116 000000G          +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A
25 00120 000000G          +INTST                  ;    COMMA AND GET "N" IN COMMAND,
26                                                ;    INTEGERIZE RESULT AND TEST
27                                                ;    <65535.
28 00122 000000G          +CKCOMA                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
29 00124 000126'          .+2                     ;LEAVE IN POLISH MODE
30 00126 016503           MOV     FAC2(R5),R3     ;GET "N"
          000042
31 00132 052626           BIS     (SP)+,(SP)+     ;M=0? (I.E. SET BITS IN REGISTER?)
32 00134 001003           BNE     DOR1            ;BNE MEANS CLEAR BITS
33 00136 050337           BIS     R3,@#LPSDOR     ;BEQ MEANS SET BITS
          000000G
34 00142 000402           BR      DOR2
35 00144 040337 DOR1:     BIC     R3,@#LPSDOR     ;CLEAR BITS
          000000G
36 00150 004737 DOR2:     JSR     PC,@#ENTERP     ;GET ADDRESS OF NEWDOR
          000000G
37 00154 000000G          +GETV
38 00156 013703           MOV     @#LPSDOR,R3     ;RETURN NEW OUTPUT REGISTER SETTING
          000000G
39 00162 000167           JMP     DIR3            ;    AND CLEANUP.
          177702
40                ;
```

40

```
                ;       .SBTTL  "DRS" COMMAND PROCESSOR
                ;
                ;*****************************************************************
                ;
                ; "DRS" COMMAND PROCESSOR
                ;
                ; BASIC FORM:   CALL "DRS"(BUF,MODE,NPTS,M,NEWCSR)
                ;
                ; PURPOSE:  DIGITAL READOUT SAMPLING.  SAMPLES THE DIGITAL
                ;               INPUT REGISTER IN A SIMILAR FASHION AS THE
                ;               RTS FUNCTION,  WHEN M=0, EACH TIME THE CLOCK
                ;               FIRES (OR ST, WHICH CAN TRIGGER THE CLOCK),
                ;               THE DIGITAL INPUT REGISTER IS READ, POSSIBLY
                ;               CONVERTED FROM BCD TO BINARY (DEPENDING ON THE
                ;               "MODE"), AND STORED IN THE CIRCULAR BUFFER BUF.
                ;               THE CIRCULAR BUFFER POINTERS ARE INITIALLY RESET
                ;               BEFORE THE SAMPLING OPERATION BEGINS.
                ;
                ;               IF THE DRS IS NOT CLOCK DRIVEN, M<>0, IT MAY BE
                ;               DRIVEN BY DIGITAL INPUTS, I.E. WHENEVER A NEW
                ;               VALUE IS RECEIVED BY THE INPUT REGISTER, THE VALUE
                ;               IS READ AND STORED IN THE BUFFER BUF.
                ;
                ;               IF THE "MODE"=0, THEN READ BCD OTHERWISE READ BINARY
                ;               DIRECTLY.  IF "NPTS" IS GIVEN AS ZERO, DISABLE
                ;               THE SAMPLING.
                ;
                ;               THE NEW SETTING OF THE DIGITAL CONTROL STATUS
                ;               REGISTER IS RETURNED IN NEWCSR.
                ;
31 00166        DRS:                            ;ENTRY POINT TO PROCESSOR
32 00166 012737 MOV     #DRSINT,@#DRSIVA        ;INT. ENTRY POINT.
         000446'
         000000G
33 00174 012737 MOV     #200,@#DRSIP            ;AND PRIORITY.
         000200
         000000G
34 00202 012765 MOV     #MAXD,T3(R5)    ;FOR LIMIT CHECKS ON "NPTS"
         000000'
         000062
35 00210 105037 CLRB    @#DRSON         ;STOP ANY PREVIOUS DRS OPERATION
         000000G
36 00214 042737 BIC     #100,@#LPSDRS
         000100
         000000G
37 00222 004737 JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
         000000G
38                                      ;    FOR ENOUGH AREA TO WORK
39                                      ;    WITH AND THAT FIRST TOKEN
40                                      ;    IS INDEED A LEFT PARENS
41 00226 000000G        +GETBUF         ;GET BUFFER ADDRESS FROM COMMAND STRING
42                                      ;    AND CHECK IT AGAINST THE
43                                      ;    DEFINITION TABLE.
44 00230 000000G        +BUFRES         ;RESET BUFFER POINTERS
45 00232 000000G        +SAVER2         ;SAVE R2 ON STACK SINCE IT HAS THE
46                                      ;    BUFFER DESCRIPTOR ADDRESS.
47 00234 000000G        +CKCMGN         ;GET "MODE" AND SAVE ON STACK
```



```
48 00236 000000G        +SAVFAC
49 00240 000000G        +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
50                                              ;    AND GET NPTS AND INTEGERIZE
51 00242 000000G        +INTST                  ;    MAX ALLOWABLE NBR POINTS IS
52                                              ;    65535.
53 00244 000000G        +SAVINT                 ;SAVE NPTS AS AN INTEGER ON THE STACK
54 00246 000000G        +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA AND
55                                              ;    GET "M".
56 00250 000000G        +SAVFAC                 ;SAVE IT ON THE STACK
57 00252 000000G        +CKCOMA                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
58 00254 000000G        +GETV                   ;GET VARIABLE ADDRESS OF NEWCSR
59 00256 000260'        .+2                     ;LEAVE POLISH MODE
60 00260 052626         BIS     (SP)+,(SP)+     ;M=0?
61 00262 001403         BEQ     DRS01           ;YES:  DRS IS CLOCK DRIVEN
62 00264 112700         MOVB    #1,R0           ;SET DRSON ON EXIT EQUAL TO PLUS NON-
         000001
63 00270 000402         BR      DRS02           ;    ZERO VALUE,
64 00272 012700 DRS01:  MOV     #-1,R0          ;M#0.  SET DRS ON EXIT EQUAL TO NEGATIVE
         177777
65                                              ;    NON-ZERO VALUE INDICATING DIGITAL
66                                              ;    INPUT LOADING.
67 00276 012667 DRS02:  MOV     (SP)+,DRSNPT    ;NBR OF POINTS TO TAKE
         000000G
68 00302 001420         BEQ     DRS05           ;IF ZERO, DISABLE SAMPLING
69 00304 105037         CLRB    @#BCDON         ;ASSUME BCD MODE
         000000G
70 00310 052626         BIS     (SP)+,(SP)+     ;GET MODE (BCD/BINARY)
71 00312 001402         BEQ     DRS03           ;BCD
72 00314 105237         INCB    @#BCDON         ;READ BINARY
         000000G
73 00320 012667 DRS03:  MOV     (SP)+,DRSBUF    ;SAVE BUFFER DESCRIPTOR ADDRESS
         000000G
74 00324 110037         MOVB    R0,@#DRSON      ;SET DRS OPERATION GOING
         000000G
75 00330 100403         BMI     DRS04           ;CLOCK DRIVEN?
76 00332 052737         BIS     #100,@#LPSDRS   ;YES:  ENABLE INPUT INTERRUPT ENABLE
         000100
         000000G
77 00340 000167 DRS04:  JMP     DIR2            ;RETURN STATUS REGISTER TO CALLER
         177520
78 00344 062706 DRS05:  ADD     #6,SP           ;CLEAN UP STACK
         000006
79 00350 000773         BR      DRS04           ;RETURN STATUS REGISTER TO CALLER
80              ;
```

```
1                       ;       .SBTTL  "REL" COMMAND PROCESSOR
2                       ;
3                       ;*****************************************************************
4                       ;
5                       ; "REL" COMMAND PROCESSOR
6                       ;
7                       ; BASIC FORM:   CALL "REL"(S,DIR)
8                       ;
9                       ; PURPOSE:  CLOSE OR OPEN RELAY "S".  THE COMMAND OPENS RELAY
10                      ;               "S" (S = 1 OR 2) IF "DIR" IS EQUAL TO ZERO,
11                      ;               OTHERWISE IT CLOSES IT.
12                      ;
13 00352                REL:                            ;ENTRY POINT TO PROCESSOR
14 00352 012765         REL:    MOV     #MAXR,T3(R5)    ;FOR LIMIT CHECK ON "S"
         000002'
         000062
15 00360 004767                 JSR     PC,POLENT       ;ENTER POLISH MODE AND CHECK
         000000G
16                                                      ;       FOR ENOUGH AREA TO WORK
17                                                      ;       WITH AND THAT FIRST TOKEN
18                                                      ;       IS INDEED A LEFT PARENS.
19 00364 000000G                +GETNUM                 ;GET NUMERIC FROM COMMAND ("S")
20 00366 000000G                +INTST                  ;INTEGERIZE AND TEST <=2
21 00370 000000G                +SAVINT                 ;SAVE "S" ON THE STACK
22 00372 000000G                +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
23                                                      ;       AND GET NEXT NUMERIC FROM
24                                                      ;       COMMAND STRING.
25 00374 000376'                .+2                     ;LEAVE POLISH MODE
26 00376 012703                 MOV     #LPSDRS,R3      ;ADDRESS OF LPS DIGITAL I/O
         000000G
27                                                      ;       STATUS REGISTER.
28 00402 012602                 MOV     (SP)+,R2        ;GET RELAY NUMBER ("S")
29 00404 003416                 BLE     ERARG           ;ARG ERROR:  RELAY NUMBER NOT 1 OR 2
30 00406 006002                 ROR     R2
31 00410 103401                 BCS     REL1            ;RELAY #1 SELECTED
32 00412 005203                 INC     R3              ;RELAY #2:  POINT TO LEFT BYTE IN
33                                                      ;       STATUS REGISTER
34 00414 056565 REL1:   BIS     FAC2(R5),FAC1(R5) ;SET OR CLEAR FLAG?
         000042
         000040
35 00422 001403                 BEQ     REL3            ;CLEAR
36 00424 152713                 BISB    #1,(R3)         ;SET RELAY "S"
         000001
37 00430 000402                 BR      REL2
38 00432 142713 REL3:   BICB    #1,(R3)         ;RESET RELAY "S"
         000001
39 00436 000137 REL2:   JMP     @#CKRPAR        ;MAKE SURE LAST TOKEN IS A ")"
         000000G
40                                                      ;       AND RETURN TO THE BASIC
41                                                      ;       INTERPRETER.
42 00442 000137 ERARG:  JMP     @#ERRARG        ;ARG ERROR
         000000G
43                      ;
```



```
1                       ;       .SBTTL  DRS INTERRUPT ROUTINE
2                       ;
3                       ;*****************************************************************
4                       ;
5                       ; DRS INTERRUPT ROUTINE
6                       ;
7 000446                DRSINT:
8 000446 004737                 JSR     PC,@#REGSAV     ;SAVE R0, R2, AND R3 ON STACK
         000000G
9 000452 013703                 MOV     @#LPSDIB,R3     ;READ DIGITAL INPUT REGISTER
         000000G
10 00456 105737                 TSTB    @#DRSON         ;DRS OPERATION IN PROGRESS?
         000000G
11 00462 003414                 BLE     DRS4            ;NO:  DISABLE INTERRUPTS AND EXIT
12 00464 105737                 TSTB    @#BCDON         ;CONVERT TO BCD?
         000000G
13 00470 001002                 BNE     DRS1            ;NO:  TREAT AS A PURE BINARY NUMBER
14 00472 004767                 JSR     PC,CONBCD       ;CONVERT BCD TO BINARY
         000000G
15 00476 016702 DRS1:   MOV     DRSBUF,R2       ;STORE DATA IN USER'S BUFFER
         000000G
16 00502 004737                 JSR     PC,@#STODAT
         000000G
17 00506 005367                 DEC     DRSNPT          ;ALL POINTS TAKEN?
         000000G
18 00512 001007                 BNE     DRS2            ;NO
19 00514 105037 DRS4:   CLRB    @#DRSON         ;CLEAR DRSON DONE FLAG
         000000G
20 00520 042737                 BIC     #100,@#LPSDRS   ;DISABLE DIR INTERRUPT ENABLE
         000100
         000000G
21 00526 000137 DRS3:   JMP     @#RESTOR        ;RESTORE REGISTERS AND EXIT
         000000G
22 00532 052737 DRS2:   BIS     #100,@#LPSDRS   ;RE-ENABLE INTERRUPT ENABLE BIT
         000100
         000000G
23 00540 000772                 BR      DRS3            ;RESTORE REGISTERS AND EXIT
24                      ;
25       000001'                .END                    ;END OF MODULE #3
```

```
LPS3  V01-01    RT-11 MACRO VM02-09   16-OCT-74 02:02:11 PAGE 9+
SYMBOL TABLE

BAD    = 000010          BCDON = ****** G        BEG    = 000000
BUFRES= ****** G         CKCMGN= ****** G        CKCOMA= ****** G
CKRPAR= ****** G         CONBCD= ****** G        DEL    = 000012
DIR     000004RG         DIR1    000046R         DIR2    000064R
DIR3    000070R          DOR     000100RG        DOR1    000144R
DOR2    000150R          DRS     000166RG        DRSBUF= ****** G
DRSINT  000446R          DRSIP = ****** G        DRSIVA= ****** G
DRSNPT= ****** G         DRSON = ****** G        DRS01   000272R
DRS02   000276R          DRS03   000320R         DRS04   000340R
DRS05   000344R          DRS1    000476R         DRS2    000532R
DRS3    000526R          DRS4    000514R         END    = 000002
ENTERP= ****** G         ERARG   000442R         ERBUF  = ****** G
ERNOR  = ****** G        ERRARG= ****** G        ERRPDL= ****** G
ERRSYN= ****** G         EVAL   = ****** G       FAC1   = 000040
FAC2   = 000042          FLOAT  = ****** G       GET    = 000006
GETADD= ****** G         GETBUF= ****** G        GETDAT= ****** G
GETNUM= ****** G         GETV   = ****** G       GETVAR= ****** G
HISTON= ****** G         INT    = ****** G       INTST  = ****** G
LPSDIB= ****** G         LPSDOR= ****** G        LPSDRS= ****** G
MAXD    000000R          MAXR    000002R         NARRAY= ****** G
NUMSGN= ****** G         PC     =%000007         POLENT= ****** G
POLRET= 000134           PS     = 177776         PUT    = 000004
REGSAV= ****** G         REL     000352RG        REL1    000414R
REL2    000436R          REL3    000432R         RESTOR= ****** G
RESTR2= ****** G         RETURN= 000207          RTSON  = ****** G
R0     =%000000          R1     =%000001         R2     =%000002
R3     =%000003          R4     =%000004         R5     =%000005
SAVER2= ****** G         SAVFAC= ****** G        SAVINT= ****** G
SP     =%000006          SS1SAV= 000024          SS2SAV= 000026
STODAT= ****** G         STORXT= ****** G        STOVAR= ****** G
TABLE  = ****** G        T1     = 000056         T2     = 000060
T3     = 000062          VARSAV= 000022          .COMMA= ****** G
.LPAR  = ****** G        .RPAR  = ****** G
. ABS.    000000     000
          000542     001
ERRORS DETECTED: 0
FREE CORE: 17906. WORDS

LPS3,LP:=LPS3
```

```
LPS4  V01-01    RT-11 MACRO VM02-09   16-OCT-74 02:03:31 PAGE 1


1                .TITLE  LPS4  V01-01
2                ;.SBTTL LPS MODULE #4
3                ;
4                ; DEC-11-LBEXA-B-LA5    BASIC KERNEL  V02-01
5                ;
6                ; COPYRIGHT (C) 1973,1974
7                ;
8                ; DIGITAL EQUIPMENT CORPORATION
9                ; MAYNARD, MASSACHUSETTS 01754
10               ;
11               ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
12               ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
13               ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
14               ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
15               ; MAY APPEAR IN THIS DOCUMENT.
16               ;
17               ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
18               ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
19               ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
20               ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
21               ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
22               ;
23               ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
24               ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
25               ; WHICH IS NOT SUPPLIED BY DEC.
26               ;
27               ; WRITTEN BY:  RICK MULLY       FEB., 1973
28               ;
```

```
1               ;       .SBTTL  CONTENTS OF MODULE #4
2               ;
3               ; THE FOLLOWING COMMAND PROCESSORS EXIST IN THIS MODULE:
4               ;
5               ;       CLRD
6               ;       PUTD
7               ;       DIS
8               ;       FSH
9               ;       DXY
10              ;
```



```
1               ;       .SBTTL  PROGRAM GLOBALS
2               ;
3               ; GLOBALS REQUIRED FOR COMMUNICATION WITH THE BASIC
4               ; INTREPRETER.
5               ;
6                       .GLOBL  ERRPDL,ERRSYN,ERRARG
7                       .GLOBL  EVAL,GETVAR,STOVAR
8                       .GLOBL  INT,.COMMA,.RPAR
9                       .GLOBL  NUMSGN,.LPAR,$IR
10                      .GLOBL  $MLR,$DVR,$POLSH
11                      .GLOBL  DISPLY
12              ;
13              ; GLOBALS REQUIRED FOR COMMUNICATIONS WITH THE VARIOUS
14              ; LPS MODULES.
15              ;
16                      .GLOBL  TABLE,NARRAY,FLOAT
17                      .GLOBL  POLENT,GETADD,GETNUM
18                      .GLOBL  CKCMGN,INTST,GETV
19                      .GLOBL  GETBUF,REGSAV,ERNOR
20                      .GLOBL  STORXT,CKRPAR,ERBUF
21                      .GLOBL  CKCOMA,ENTERP,STODAT
22                      .GLOBL  GETDAT,RTSON,DRSON
23                      .GLOBL  SAVER2,SAVFAC,RESTR2
24                      .GLOBL  RESTOR,HISTON,BCDON
25                      .GLOBL  CONBCD,BUFRES,DRSNPT
26                      .GLOBL  SAVINT
27              ;
28              ; GLOBALS REQUIRED FOR COMMAND PROCESSORS
29              ;
30                      .GLOBL  CLRD,PUTD,DIS,FSH,DXY
31              ;
32              ; GLOBALS FOR DEVICE ADDRESSES
33              ;
34                      .GLOBL  LPDISS,LPDISX,LPDISY
35              ;
36              ;GLOBALS FOR INTERRUPT VECTOR AND PRIORITY.
37                      .GLOBL LPSIVA,LPSIP
```

```
                        ;       ,SBTTL  REGISTER ASSIGNMENTS AND EQUATES
                        ;
000000 R0       =       %0
000001 R1       =       %1
000002 R2       =       %2
000003 R3       =       %3
000004 R4       =       %4
000005 R5       =       %5
000006 SP       =       %6
000007 PC       =       %7
                        ;
                        ; GENERAL EQUATES
                        ;
177776 PS       =       -2              ;PS = PROCESSOR STATUS WORD
000207 RETURN   =       207             ;207 = RTS PC
000134 POLRET   =       134             ;134 = JMP @(R4)+
                        ;
                        ; BASIC USER'S EQUATES
                        ;
000022 VARSAV   =       22              ;VAR SAVE FOR ASSIGNMENT
000024 SS1SAV   =       24              ;SS1 SAVE FOR ASSIGNMENT
000026 SS2SAV   =       26              ;SS2 SAVE FOR ASSIGNMENT
000040 FAC1     =       40              ;HIGH ORDER FLOATING VALUE
000042 FAC2     =       42              ;LOW  ORDER FLOATING VALUE
000044 R0SAVE   =       44              ;R0 SAVE LOCATION
000046 R1SAVE   =       46              ;R1 SAVE LOCATION
000050 R2SAVE   =       50              ;R2 SAVE LOCATION
000052 R3SAVE   =       52              ;R3 SAVE LOCATION
000056 T1       =       56              ;SHORT TERM TEMPORARY
000060 T2       =       60              ;SHORT TERM TEMPORARY
000062 T3       =       62              ;SHORT TERM TEMPORARY
                        ;
                        ; BUFFER DESCRIPTOR EQUATES
                        ;
000000 BEG      =       0               ;OFFSET TO BEGINNING OF BUFFER POINTER
000002 END      =       2               ;OFFSET TO END OF BUFFER POINTER
000004 PUT      =       4               ;OFFSET TO PUT DATA POINTER
000006 GET      =       6               ;OFFSET TO GET DATA POINTER
000010 BAD      =       10              ;OFFSET TO BAD DATA FLAG
000012 DEL      =       12              ;OFFSET TO DISPLAY DELTA X
                        ;
```



```
                        ;
                        ; DISPLAY TABLE EQUATES
                        ;
000000 DSTY     =       0               ;BUFFER START ADDRESS FOR Y
000002 DSTX     =       2               ;BUFFER START ADDRESS FOR X OR
                                        ;    X POSITION ON FACE OF CRT.
000004 DPTY     =       4               ;BUFFER POINTER FOR Y
000006 DPTX     =       6               ;BUFFER POINTER FOR X OR X INCREMENT
000010 DTNPTS   =       10              ;NBR OF POINTS (TOTAL)
000012 DLNPTS   =       12              ;NBR OF POINTS (RUNNING)
000014 DSW      =       14              ;DISPLAY ON/OFF SWITCH
000015 DSX      =       15              ;SWITCH FOR X INCREMENT OR X BUFFER
000016 DNPTI    =       16              ;INITIAL "N" (NBR OF POINTS PER CALL
000020 DNPTR    =       20              ;RUNNING VALUE FOR "N"
000022 DISINC   =       22              ;Y/(X) BUFFER INCREMENT ON DATA
000024 DENDY    =       24              ;END OF ARRAY Y
000026 DENDX    =       26              ;END OF ARRAY X
                        ;
                        ;
                        ; DISPLAY TABLE FOR "DIS" AND "DXY" COMMANDS
                        ;
00000 000000 RESDIS:    .WORD   0,0,0,0,0,0
00002 000000
00004 000000
00006 000000
00010 000000
00012 000000
00014 000000            .WORD   0,0,0,0,0,0
00016 000000
00020 000000
00022 000000
00024 000000
00026 000000
                        ;
                        ; DISPLAY TABLE FOR "FSH" COMMAND
                        ;
00030 000000 NONRES:    .WORD   0,0,0,0,0,0
00032 000000
00034 000000
00036 000000
00040 000000
00042 000000
00044 000000            .WORD   0,0,0,0,0,0
00046 000000
00050 000000
00052 000000
00054 000000
00056 000000
                        ;
                        ; LIMIT TESTS
                        ;
00060 007777 MAXD:      .WORD   7777            ;4095
00062 007777 MAXC:      .WORD   7777            ;4095
00064 177777 MAXP:      .WORD   177777          ;65535
                        ;
```

```
                ;       .SBTTL  "CLRD" COMMAND PROCESSOR
                ;
                ;*****************************************************************
                ;
                ; "CLRD" COMMAND PROCESSOR
                ;
                ; BASIC FORM:   CALL "CLRD"(BUF,SIZE,SCALE)
                ;
                ; PURPOSE:  DEFINE DISPLAY BUFFER HAVING FIXED DELTA X VALUES.
                ;               "BUF" IS THE NAME OF THE BUFFER TO BE DISPLAYED,
                ;               AND CONTAINS SINGLE WORD VALUES.  VALUES WHICH
                ;               FALL IN THE RANGE 4095 >= VALUE >= 0 ARE DIS-
                ;               PLAYED WHILE VALUES OUTSIDE THIS ARE NOT.
                ;               THE "SIZE" OF THE BUFFER IS THE NUMBER OF Y
                ;               POINTS TO DISPLAY AND MUST BE <= TO THE NUMBER
                ;               DEFINED IN THE USE AND DIM COMMANDS.  THE DELTA X
                ;               IS CALCULATED AS 4096/"SIZE" AND MAY BE FRACTIONAL.
                ;
                ;               IF "SCALE" = 0, CLRD WILL SET ALL BUFFER VALUES
                ;               TO -1 (NON-DISPLAYABLE VALUES).  IF "SCALE" DOES NOT
                ;               =0, CLRD DOES NOT CLEAR THE ARRAY AND THE ORIG-
                ;               INAL DATA IS MULTIPLIED BY "SCALE".  IN EITHER CASE, THE
                ;               PUTD POINTERS ARE RESET TO POINT TO THE
                ;               BEGINNING OF THE ARRAY.  DATA IS ENTERED INTO THE
                ;               ARRAY THROUGH THE PUTD FUNCTION, HOWEVER, A
                ;               CLRD MUST BE ISSUED BEFORE DATA IS INITIALLY
                ;               TRANSFERRED TO THE ARRAY.
                ;
                ;               A CLRD MUST BE ISSUED AT LEAST ONCE BEFORE
                ;               ISSUING THE PUTD, DIS, OR FSH FUNCTIONS.
                ;
00066           CLRD:                           ;ENTRY POINT TO PROCESSOR
00066 012765    CLRD:   MOV     #MAXC,T3(R5)    ;FOR LIMIT TESTING OF "SIZE"
      000062'
      000062
00074 004737            JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
      000000G
                                                ;       FOR ENOUGH AREA TO WORK
                                                ;       WITH AND THAT FIRST TOKEN
                                                ;       IS INDEED A LEFT PARENS.
00100 000000G           +GETBUF                 ;GET BUFFER ADDRESS FROM COMMAND STRING
                                                ;       AND CHECK IT AGAINST THE
                                                ;       DEFINITION TABLE.
00102 000000G           +BUFRES                 ;RESET BUFFER POINTERS
00104 000000G           +SAVER2                 ;SAVE R2 ON THE STACK
00106 000000G           +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
                                                ;       AND GET NUMERIC FROM COMMAND ("SIZE")
00110 000000G           +INTST                  ;INTERGIZE RESULT AND TEST <= 4095.
00112 000000G           +SAVINT                 ;SAVE "SIZE"
00114 000000G           +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
                                                ;       AND GET NUMERIC FROM COMMAND ("MODE")
00116 000000G           SAVFAC                  ;SAVE "SCALE" ON STACK
00120 000122'           .+2                     ;LEAVE POLISH MODE
00122 012665            MOV     (SP)+,T2(R5)    ;GET "SCALE" AND SAVE IT
      000060
00126 012665            MOV     (SP)+,T1(R5)
      000056
```



```
00132 012600            MOV     (SP)+,R0        ;GET "SIZE"
00134 003420            BLE     CLRD0           ;SIZE <= 0 IS ILLEGAL
00136 012602            MOV     (SP)+,R2        ;R2 POINTS TO BUFFER DESCRIPTOR ADDRESS
00140 011203            MOV     (R2),R3         ;GET START ADDRESS OF BUFFER
00142 010065            MOV     R0,R0SAVE(R5)   ;SAVE R0 THRU R3 FOR LATER USE
      000044
00146 010165            MOV     R1,R1SAVE(R5)
      000046
00152 010265            MOV     R2,R2SAVE(R5)
      000050
00156 010365            MOV     R3,R3SAVE(R5)
      000052
00162 016246            MOV     END(R2),-(SP)   ;CALCULATE SIZE OF BUFFER
      000002
00166 160316            SUB     R3,(SP)         ;(END ADDRESS - BEGINNING ADDRESS)/2
00170 006216      *     ASR     (SP)
00172 020026            CMP     R0,(SP)+        ;"SIZE" <= BUFFER SIZE?
00174 003402            BLE     CLRD1           ;YES:  CONTINUE
00176 000137 CLRD0:     JMP     @#ERNOR         ;"SIZE" TOO LARGE
      000000G
00202 020027 CLRD1:     CMP     R0,#4096.       ;IS "SIZE" >= 4096?
      010000
00206 103020            BHIS    CLRD3           ;YES:  SET DELTA X =1 AND FRACTION
                                                ;       =0.
00210 012746            MOV     #4096.,-(SP)    ;4096 POINTS ON X SCALE OF SCOPE
      010000
00214 004437            JSR     R4,@#$POLSH     ;CALCULATE (4096./"SIZE")=0.
      000000G
00220 000000G           +$IR                    ;FLOAT 4096.
00222 000466'           +PUSHR0                 ;NOW PUT "SIZE" BACK ON STACK AND
                                                ;       FLOAT IT.
00224 000000G           +$IR
00226 000000G           +$DVR                   ;(BUF SIZE/"SIZE")
00230 000474'           +PUSH0                  ;PUSH A FLOATING POINT 0. ON STACK
00232 000000G           +$MLR                   ;=0.
00234 000454'           +TRAFAC                 ;TRANSFER RESULT TO THE FAC
00236 000000G           +INTST                  ;INTEGERIZE RESULT IN FAC
00240 000242'           .+2                     ;LEAVE POLISH MODE
00242 016500            MOV     FAC2(R5),R0     ;IF INCREMENT =0, SET =1
      000042
00246 001002            BNE     CLRD2
00250 012700 CLRD3:     MOV     #0.,R0          ;SET INCREMENT =1 (1=0. WITH A FRACTION
      000010
                                                ;       OF ZERO)
00254 016502 CLRD2:     MOV     R2SAVE(R5),R2   ;SAVE DELTA X FOR THIS BUFFER
      000050
00260 010062            MOV     R0,DEL(R2)
      000012
00264 016546            MOV     T2(R5),-(SP)    ;PUT "SCALE" FACTOR ON STACK
      000060
00270 016546            MOV     T1(R5),-(SP)
      000056
00274 001005            BNE     CLRD4           ;IF AN INTEGER, FLOAT RESULT
00276 005726            TST     (SP)+
00300 004437            JSR     R4,@#$POLSH     ;FLOAT "SCALE" FACTOR
      000000G
00304 000000G           +$IR
```

```
94 00306 000310'          .+2                       ;LEAVE POLISH MODE
95 00310 016646 CLRD4:    MOV     2(SP),-(SP)       ;SAVE A 2ND COPY OF "SCALE" ON
         000002
96 00314 016646           MOV     2(SP),-(SP)       ;    THE STACK.
         000002
97 00320 001450           BEQ     CLRD5             ;"SCALE" FACTOR WAS EQUAL TO ZERO,
98                                                  ;    CLEAR OUT ARRAY.
99 00322 017503           MOV     @R3SAVE(R5),R3    ;MULTIPLY DATA BY "SCALE" FACTOR
         000052
100 0326 004737           JSR     PC,@#FLOAT        ;FLOAT IF DATA IS 16 BITS.
         000000G
101 0332 016546           MOV     FAC2(R5),-(SP)    ;PUT DATA ON STACK IN STANDARD
         000042
102 0336 016546           MOV     FAC1(R5),-(SP)    ;    FLOATING POINT FORM.
         000040
103 0342 001005           BNE     CLRD7             ;DATA ALREADY FLOATED
104 0344 005726           TST     (SP)+             ;FLOAT NUMBER
105 0346 004437           JSR     R4,@#SPOLSH
         000000G
106 0352 000000G          +SIR
107 0354 000356'          .+2
108 0356 012765 CLRD7:    MOV     #MAXP,T3(R5)      ;RESET INTEGER TEST
         000064'
         000062
109 0364 004437           JSR     R4,@#SPOLSH       ;ENTER POLISH MODE
         000000G
110 0370 000000G          +SMLR                     ;MULTIPLY DATA BY "SCALE" FACTOR
111 0372 000454'          +TRAFAC                   ;TRANSFER RESULT TO THE FAC
112 0374 000000G          +INTST                    ;INTEGERIZE IT
113 0376 000400'          .+2                       ;LEAVE POLISH MODE
114 0400 016575 CLRD6:    MOV     FAC2(R5),@R3SAVE(R5); SAVE NEW DATA
         000042
         000052
115 0406 062765           ADD     #2,R3SAVE(R5)     ;POINTS TO NEXT DATA POINT
         000002
         000052
116 0414 016502           MOV     R2SAVE(R5),R2     ;ARRAY DESCRIPTOR ADDRESS
         000050
117 0420 026562           CMP     R3SAVE(R5),END(R2) ;END OF ARRAY?
         000052
         000002
118 0426 101730           BLOS    CLRD4             ;NO:  CONTINUE
119 0430 022626           CMP     (SP)+,(SP)+       ;CLEAN UP STACK
120 0432 016501           MOV     R1SAVE(R5),R1     ;RESTORE R1
         000046
121 0436 000137           JMP     @#CKRPAR          ;MAKE SURE LAST TOKEN IS A ")" AND
         000000G
122                                                 ;    RETURN TO THE BASIC INTREPRETER.
123                       ;
124 0442 022626 CLRD5:    CMP     (SP)+,(SP)+       ;IGNORE "SCALE" ON STACK
125 0444 012765           MOV     #-1,FAC2(R5)      ;SET DATA ELEMENT =-1 (NON DISPLAYABLE)
         177777
         000042
126 0452 000752           BR      CLRD6
127                       ;
128 0454 012665 TRAFAC:   MOV     (SP)+,FAC1(R5)    ;POLISH ROUTINE TO TRANSFER STACK
         000040
```



```
129                                                 ;    DATA TO FAC.
130 0460 012665           MOV     (SP)+,FAC2(R5)
         000042
131 0464 000134           POLRET                    ;RETURN IN POLISH MODE
132                       ;
133 0466 016546 PUSHR0:   MOV     R0SAVE(R5),-(SP) ;POLISH ROUTINE TO PUSH R0SAVE (SIZE)
         000044
134                                                 ;    ON THE STACK.
135 0472 000134           POLRET                    ;RETURN IN POLISH MODE
136                       ;
137 0474 016746 PUSH8:    MOV     FP8+2,-(SP)       ;PUSH FLOATING POINT 8. ON STACK
         000010
138 0500 016746           MOV     FP8,-(SP)
         000002
139 0504 000134           POLRET                    ;RETURN IN POLISH MODE
140                       ;
141 0506 041000 FP8:      .WORD   041000,0          ;FLOATING POINT 8.
    0510 000000
```

```
                ;       .SBTTL  "PUTD" COMMAND PROCESSOR
                ;
                ;*****************************************************************
                ;
                ; "PUTD" COMMAND PROCESSOR
                ;
                ; BASIC FORM:   CALL "PUTD"(BUF,Y)
                ;
                ; PURPOSE:  PUT DATA POINT "Y" INTO BUF IN SEQUENTIAL ORDER,
                ;               WHERE 65535 >= Y >= 0,  THIS FUNCTION DOES NOT
                ;               INITIATE A DISPLAY, BUT RATHER JUST ENTERS A
                ;               DATA POINT INTO THE SPECIFIED ARRAY.
                ;
                PUTD:                                   ;ENTRY POINT TO PROCESSOR
00512 012765    MOV     #MAXP,T3(R5)    ;FOR LIMIT TESTING OF "Y"
      000064'
      000062
00520 004737    JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
      000000G
                                        ;       FOR ENOUGH AREA TO WORK
                                        ;       WITH AND THAT FIRST TOKEN
                                        ;       IS INDEED A LEFT PARENS,
00524 000000G   +GETBUF                 ;GET BUFFER ADDRESS FROM COMMAND STRING
                                        ;       AND CHECK IT AGAINST THE
                                        ;       DEFINITION TABLE,
00526 000000G   +SAVER2                 ;SAVE R2 ON THE STACK
00530 000000G   +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
                                        ;       AND GET NUMERIC FROM COMMAND ("Y")
00532 000000G   +INTST                  ;INTEGERIZE RESULT AND TEST <= 65535,
00534 000000G   +RESTR2                 ;RESTORE R2
00536 000540'   ,+2                     ;EXIT POLISH MODE
00540 016503    MOV     FAC2(R5),R3     ;STORE RESULT NOW
      000042
00544 004737    JSR     PC,@#STODAT
      000000G
00550 000137    JMP     @#CKRPAR        ;MAKE SURE LAST TOKEN IS A ")" AND
      000000G
                                        ;       RETURN TO THE BASIC INTREPRETER,
                ;
```



```
                ;       .SBTTL  "DIS" COMMAND PROCESSOR
                ;
                ;*****************************************************************
                ;
                ; "DIS" COMMAND PROCESSOR
                ;
                ; BASIC FORM:   CALL "DIS"(BUF,N,I)
                ;
                ; PURPOSE:  DISPLAY DATA FROM BUF WHENEVER BASIC IS IDLE,  THE
                ;               POINTS DISPLAYED START WITH THE NTH POINT IN
                ;               THE BUFFER AND PROCEED IN INCREMENTS OF I,
                ;               IF I=1, CONSECUTIVE POINTS STARTING WITH THE NTH
                ;               ONE IS DISPLAYED.  IF I=2, EVERY OTHER POINT
                ;               IS DISPLAYED, ETC..
                ;
                DIS:                                    ;ENTRY POINT TO PROCESSOR
00554 012765    MOV     #DIS1,T1(R5)    ;RETURN PC FOR ARGUMENT PARSER
      000572'
      000056
00562 005065    CLR     T2(R5)          ;INDICATE "DIS" ACTIVE
      000060
00566 000167    JMP     DISPAR          ;NOW PARSE LINE
      000216
00572 012703 DIS1:      MOV     #RESDIS,R3      ;SETUP DISPLAY TABLE
      000000'
00576 004767    JSR     PC,DSETUP
      000270
00602 005723    TST     (R3)+           ;R3=R3+2
00604 012713    MOV     #4,(R3)         ;ASSUME 4 POINTS PER CALL TO
      000004
                                        ;       DISPLAY ROUTINE
00610 012323    MOV     (R3)+,(R3)+
00612 016513    MOV     FAC2(R5),(R3)   ;GET Y INCREMENT BETWEEN CHANNELS
      000042
00616 000313    ASL     (R3)            ;CONVERT TO A WORD OFFSET
00620       DIS3:
00620 062706    ADD     #4,SP           ;CLEAN UP STACK
      000004
00624 000137    JMP     @#CKRPAR        ;MAKE SURE LAST TOKEN IS A ")"
      000000G
                                        ;       AND RETURN TO THE BASIC
                                        ;       INTERPRETER,
                ;
```

```
1                       ;       .SBTTL  "FSH" COMMAND PROCESSOR
2                       ;
3                       ;*****************************************************************
4                       ;
5                       ; BASIC FORM:   CALL "FSH"(BUF,N,I)
6                       ;
7                       ; PURPOSE:  IDENTICAL TO DIS EXCEPT THE DATA POINTS IN BUF ARE
8                       ;               DISPLAYED ONLY ONCE.  THE NEXT BASIC STATE-
9                       ;               MENT IS THEN EXECUTED.
10                      ;
11 00630                FSH:                            ;ENTRY POINT TO PROCESSOR
12 00630 012765         FSH1:   MOV     #FSH1,T1(R5)    ;RETURN PC FOR ARGUMENT PARSER
         000646'
         000056
13 00636 005065                 CLR     T2(R5)          ;INDICATE "FSH" ACTIVE
         000060
14 00642 000167                 JMP     DISPAR          ;NOW PARSE LINE
         000142
15 00646 012703 FSH1:   MOV     #NONRES,R3      ;SETUP DISPLAY TABLE
         000030'
16 00652 004767                 JSR     PC,DSETUP
         000214
17 00656 005723                 TST     (R3)+           ;R3=R3+2
18 00660 016313                 MOV     -6(R3),(R3)     ;DISPLAY ENTIRE SWEEP ON EVERY CALL
         177772
19                                                      ;       TO THE DISPLAY ROUTINE.
20 00664 012323                 MOV     (R3)+,(R3)+
21 00666 016513                 MOV     FAC2(R5),(R3)   ;GET Y INCREMENT BETWEEN CHANNELS
         000042
22 00672 006313                 ASL     (R3)            ;CONVERT TO A WORD OFFSET
23 00674 012702                 MOV     #NONRES,R2      ;NOW FLASH UP A COMPLETE SWEEP
         000030'
24 00700 004767                 JSR     PC,DSPLY
         000266
25 00704 000745                 BR      DIS3            ;GET Y INCREMENT BETWEEN CHANNELS,
26                                                      ;       CLEAN UP STACK, DISPLAY DATA,
27                                                      ;       AND RETURN BACK TO THE BASIC
28                                                      ;       INTERPRETER.
29                      ;
```



```
1                       ;       .SBTTL  "DXY" COMMAND PROCESSOR
2                       ;
3                       ;*****************************************************************
4                       ;
5                       ; BASIC FORM:   "DXY"(BUF1,BUF2,N,I)
6                       ;
7                       ; PURPOSE:  DISPLAY DATA FROM BUF1 AND BUF2.  BUF1 AND BUF2
8                       ;               HAVE THE X AND Y VALUES RESPECTIVELY.  NO DELTA
9                       ;               X IS USED FROM THE CLRD,  OTHERWISE
10                      ;               DXY IS IDENTICAL TO DIS AND FSH.
11                      ;
12 00706                DXY:                            ;ENTRY POINT TO PROCESSOR
13 00706 012765         DXY1:   MOV     #DXY1,T1(R5)    ;RETURN PC FOR ARGUMENT PARSER
         000724'
         000056
14 00714 010565                 MOV     R5,T2(R5)       ;INDICATE "DXY" ACTIVE
         000060
15 00720 000167                 JMP     DISPAR          ;NOW PARSE LINE
         000064
16 00724 012703 DXY1:   MOV     #RESDIS,R3      ;SETUP DISPLAY TABLE
         000000'
17 00730 004767                 JSR     PC,DSETUP
         000136
18 00734 052723                 BIS     #400,(R3)+      ;SET DSX ACTIVE
         000400
19 00740 012713                 MOV     #4,(R3)         ;ASSUME 4 POINTS PER CALL TO
         000004
20 00744 012323                 MOV     (R3)+,(R3)+     ;       DISPLAY ROUTINE
21 00746 016513                 MOV     FAC2(R5),(R3)   ;GET Y INCREMENT BETWEEN CHANNELS
         000042
22 00752 006313                 ASL     (R3)            ;CONVERT TO A WORD OFFSET
23 00754 016602                 MOV     4(SP),R2        ;GET X BUFFER DESCRIPTOR ADDRESS NOW
         000004
24 00760 016263                 MOV     END(R2),4(R3)   ;END ADDRESS OF ARRAY X
         000002
         000004
25 00766 162703                 SUB     #20,R3          ;RESET TABLE ADDRESS
         000020
26 00772 011213                 MOV     (R2),(R3)       ;INITIAL START ADDRESS OF X ARRAY
27 00774 061613                 ADD     (SP),(R3)       ;ADD IN STARTING CHANNEL -1
28 00776 005313                 DEC     (R3)
29 01000 011363                 MOV     (R3),4(R3)
         000004
30 01004 005726                 TST     (SP)+           ;POP OFF TOP STACK ELEMENT,
31 01006 000704                 BR      DIS3            ;       REST IS CLEARED OFF IN DIS
32                                                      ;       ROUTINE.
33                      ;
```

```
1                       ;       .SBTTL  DISPLAY COMMAND PARSER FOR DIS, FSH, AND DXY
2                       ;
3                       ; ROUTINE TO PARSE COMMAND LINE FOR DIS, FSH AND THE DXY COMMANDS.
4                       ;
5                       ; CALL:
6                       ;       MOV     #RETURN,T1(R5)
7                       ;       CLR     T2(R5)          ;FOR DIS AND FSH;  MOV R5,T2(R5)
8                       ;                               ;    FOR DXY
9                       ;       JMP     DISPAR
10                      ;RETURN: NEXT INSTRUCTION
11                      ;
12 01010                DISPAR:
13 01010 012765         MOV     #MAXD,T3(R5)    ;FOR LIMIT TESTS ON N AND I
         000060'
         000062
14 01016 004737         JSR     PC,@#POLENT     ;ENTER POLISH MODE AND CHECK
         000000G
15                                              ;    FOR ENOUGH AREA TO WORK
16                                              ;    WITH AND THAT FIRST TOKEN
17                                              ;    IS INDEED A LEFT PARENS.
18 01022 000000G        +GETBUF                 ;GET BUFFER ADDRESS FROM COMMAND
19                                              ;    AND CHECK IT AGAINST THE
20                                              ;    DEFINITION TABLE.
21 01024 000000G        +SAVER2                 ;SAVE R2 ON STACK
22 01026 001056'        +TSTDXY                 ;TEST FOR DXY COMMAND
23 01030 000000G        +CKCOMA                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
24 01032 000000G        +GETBUF                 ;GET NEXT BUFFER ADDRESS AND CHECK IT
25 01034 000000G        +SAVER2                 ;SAVE IT ALSO
26 01036 000000G        +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
27                                              ;    AND GET NUMERIC FROM COMMAND.
28 01040 000000G        +INTST                  ;INTEGERIZE RESULT AND TEST <=7777.
29 01042 000000G        +SAVINT                 ;SAVE "N"
30 01044 000000G        +CKCMGN                 ;CHECK NEXT TOKEN EQUAL TO A COMMA
31                                              ;    AND GET NUMERIC FROM COMMAND.
32 01046 000000G        +INTST                  ;INTEGERIZE RESULT AND TEST <=7777.
33 01050 001052'        .+2                     ;LEAVE POLISH MODE
34 01052 000175         JMP     @T1(R5)         ;RETURN TO CALLER
         000056
35                      ;
36 01056 005765 TSTDXY: TST     T2(R5)          ;DXY COMMAND?
         000060
37 01062 001002         BNE     TST1            ;YES:  RETURN IMMEDIATELY
38 01064 062704         ADD     #6,R4           ;BYPASS ANOTHER BUFFER PROCESSING
         000006
39 01070 000134 TST1:   POLRET                  ;RETURN IN POLISH MODE
40                      ;
```



```
1                       ;       .SBTTL  SETUP DISPLAY TABLE
2                       ;
3                       ; ROUTINE TO SETUP A DISPLAY TABLE
4                       ;
5                       ; CALL:
6                       ;       MOV     #ADDRESS OF TABLE,R3
7                       ;       JSR     PC,DSETUP
8                       ;       R3 ON RETURN POINTS TO DSW IN TABLE
9                       ;
10 01072                DSETUP:
11 01072 016602         MOV     4(SP),R2        ;GET BUFFER DESCRIPTOR ADDRESS
         000004
12 01076 011213         MOV     (R2),(R3)       ;INITIAL START ADDRESS OF ARRAY
13 01100 016263         MOV     END(R2),DENDY(R3) ;END OF ARRAY
         000002
         000024
14 01106 066613         ADD     2(SP),(R3)      ;ADD IN STARTING CHANNEL =1
         000002
15 01112 005323         DEC     (R3)+
16 01114 005023         CLR     (R3)+           ;ASSUME DIS OR FSH COMMAND
17 01116 016323         MOV     -4(R3),(R3)+    ;RESET BUFFER PONTERS
         177774
18 01122 016223         MOV     DEL(R2),(R3)+   ;DELTA X DEFINED IN "CLRD"
         000012
19 01126 001002         BNE     DSET1
20 01130 000137         JMP     @#ERRARG        ;ZERO VALUE IS ILLEGAL HERE
         000000G
21 01134 016213 DSET1:  MOV     END(R2),(R3)    ;CALCULATE NBR OF POINTS TO DISPLAY
         000002
22 01140 161213         SUB     (R2),(R3)
23 01142 006213         ASR     (R3)
24 01144 012323         MOV     (R3)+,(R3)+
25 01146 012713         MOV     #1,(R3)         ;SET DISPLAY ACTIVE
         000001
26 01152 000207         RETURN                  ;RETURN TO CALLER
27                      ;
28                      ; ENTRY POINT FROM BASIC DURING IDLE LOOP IN ORDER TO REFRESH
29                      ; THE DISPLAY.
30                      ;
31 01154                DISPLY:
32 01154 010246         MOV     R2,-(SP)        ;SAVE R2 ON THE STACK
33 01156 012702         MOV     #RESDIS,R2      ;RESIDENT DISPLAY TABLE
         000000'
34 01162 004767         JSR     PC,DSPLY        ;DISPLAY SOME POINTS
         000004
35 01166 012602         MOV     (SP)+,R2        ;RESTORE R2
36 01170 000207         RETURN                  ;RETURN FROM BASIC'S IDLE LOOP
37                      ;
```

```
1                       ;       .SBTTL  ACTUAL DISPLAY ROUTINE
2                       ;
3                       ; ROUTINE TO DISPLAY A SERIES OF POINTS ON THE CRT.
4                       ;
5                       ; CALL:
6                       ;       R2 POINTS TO DISPLAY TABLE
7                       ;       JSR     PC,DSPLY
8                       ;
9  001172               DSPLY:
10 01172 105762 000014          TSTB    DSW(R2)         ;DISPLAY ON?
11 01176 001506                 BEQ     D1              ;NO:  EXIT IMMEDIATELY
12 01200 016262 000016 000020   MOV     DNPTI(R2),DNPTR(R2) ;RESET NBR OF POINTS PER CALL
13 01206 026262 000004 000024 D5:       CMP     DPTY(R2),DENDY(R2) ;HAS Y BUFFER BEEN EXHAUSTED?
14 01214 101102                 BHI     D7              ;YES:  IGNORE POINT
15 01216 017246 000004          MOV     @DPTY(R2),-(SP) ;GET NEXT Y VALUE
16 01222 011637 000000G         MOV     (SP),@#LPDISY
17 01226 066262 000022 000004   ADD     DISINC(R2),DPTY(R2) ;BUMP TO NEXT Y
18 01234 105762 000015          TSTB    DSX(R2)         ;X INCREMENT OR X BUFFER DATA?
19 01240 001416                 BEQ     D2              ;X INCREMENT
20 01242 026262 000006 000026   CMP     DPTX(R2),DENDX(R2) ;HAS X BUFFER BEEN EXHAUSTED?
21 01250 101010                 BHI     D9              ;YES:  IGNORE POINT
22 01252 017246 000006          MOV     @DPTX(R2),-(SP) ;GET NEXT POINT
23 01256 011637 000000G         MOV     (SP),@#LPDISX   ;X BUFFER DATA
24 01262 066262 000022 000006   ADD     DISINC(R2),DPTX(R2) ;BUMP TO NEXT X
25 01270 000427                 BR      D3
26 01272 005726 D9:     TST     (SP)+           ;IGNORE POINT AND CLEAN UP STACK
27 01274 000452                 BR      D7
28 01276 016246 000002 D2:      MOV     DSTX(R2),-(SP)  ;GET X INCREM AND DIVIDE BY 8.
29 01302 016246 000006          MOV     DPTX(R2),-(SP)
30 01306 016246 000022          MOV     DISINC(R2),-(SP) ;GET INCREM FROM Y AND USE TO
31 01312 162716 000002 D12:     SUB     #2,(SP)         ;     MODIFY X.
32 01316 003404                 BLE     D11
33 01320 066266 000006 000002   ADD     DPTX(R2),2(SP)
34 01326 000771                 BR      D12
35 01330 005726 D11:    TST     (SP)+
```



```
36 01332 062662 000002          ADD     (SP)+,DSTX(R2)  ;UPDATE X AXIS
37 01336 006216                 ASR     (SP)
38 01340 006216                 ASR     (SP)
39 01342 006216                 ASR     (SP)
40 01344 011637 000000G         MOV     (SP),@#LPDISX   ;NOW TO THE X REGISTER
41 01350 022627 007777 D3:      CMP     (SP)+,#7777     ;X VALUE IN RANGE?
42 01354 101020                 BHI     D10             ;NO:  IGNORE POINT THEN
43 01356 022627 007777          CMP     (SP)+,#7777     ;HOW ABOUT THE Y POINT?
44 01362 101006                 BHI     D6              ;NO:  INGORE POINT
45 01364 052737 000001 000000G  BIS     #1,@#LPDISS     ;INTENSIFY DATA POINT
46 01372 105737 000000G D4:     TSTB    @#LPDISS        ;WAIT FOR POINT TO GET UP THERE
47 01376 100375                 BPL     D4
48 01400 005362 000012 D6:      DEC     DLNPTS(R2)      ;ALL POINTS UP FOR A COMPLETE SWEEP?
49 01404 003406                 BLE     D7              ;YES:  RESET COUNT AND RETURN
50 01406 005362 000020          DEC     DNPTR(R2)       ;"N" POINTS UP BEFORE RETURNING
51                                                      ;     TO CALLER?
52 01412 003275                 BGT     D5              ;NO:  DO ANOTHER
53 01414 000207         D1:     RETURN                  ;RETURN TO CALLER
54 01416 005726 D10:    TST     (SP)+           ;IGNORE POINT AND CLEANUP STACK
55 01420 000767                 BR      D6
56 01422 016262 000000 000004 D7:       MOV     DSTY(R2),DPTY(R2) ;RESET DISPLAY PARAMETERS AFTER EACH
57                                                      ;     SWEEP.
58 01430 016262 000010 000012   MOV     DTNPTS(R2),DLNPTS(R2)
59 01436 105762 000015          TSTB    DSX(R2)         ;IF X INCREMENT ACTIVE, SET X AXIS=0.
60 01442 001003                 BNE     D8              ;NO
61 01444 005062 000002          CLR     DSTX(R2)
62 01450 000207                 RETURN                  ;RETURN TO CALLER NOW
63 01452 016262 000002 000006 D8:       MOV     DSTX(R2),DPTX(R2) ;RESET X POSITION
64 01460 000207                 RETURN                  ;RETURN TO CALLER NOW
65                      ;
66      000001'         .END                    ;END OF MODULE 4
```

```
LPS4  V01-01    RT-11 MACRO VM02-09   16-OCT-74 02:03:31 PAGE 13+
SYMBOL TABLE

BAD    = 000010          BCDON  = ****** G        BEG    = 000000
BUFRES= ****** G         CKCMGN= ****** G        CKCOMA= ****** G
CKRPAR= ****** G         CLRD    000066RG        CLRD0   000176R
CLRD1   000202R          CLRD2   000254R         CLRD3   000250R
CLRD4   000310R          CLRD5   000442R         CLRD6   000400R
CLRD7   000356R          CONBCD= ****** G        DEL    = 000012
DENDX  = 000026          DENDY  = 000024         DIS     000554RG
DISINC= 000022           DISPAR  001010R         DISPLY  001154RG
DIS1    000572R          DIS3    000620R         DLNPTS= 000012
DNPTI  = 000016          DNPTR  = 000020         DPTX   = 000006
DPTY   = 000004          DRSNPT= ****** G        DRSON  = ****** G
DSETUP  001072R          DSET1   001134R         DSPLY   001172R
DSTX   = 000002          DSTY   = 000000         DSW    = 000014
DSX    = 000015          DTNPTS= 000010          DXY     000706RG
DXY1    000724R          D1      001414R         D10     001416R
D11     001330R          D12     001312R         D2      001276R
D3      001350R          D4      001372R         D5      001206R
D6      001400R          D7      001422R         D8      001452R
D9      001272R          END    = 000002         ENTERP= ****** G
ERBUF  = ****** G        ERNOR  = ****** G       ERRARG= ****** G
ERRPDL= ****** G         ERRSYN= ****** G        EVAL   = ****** G
FAC1   = 000040          FAC2   = 000042         FLOAT  = ****** G
FP8     000506R          FSH     000630RG        FSM1    000646R
GET    = 000006          GETADD= ****** G        GETBUF= ****** G
GETDAT= ****** G         GETNUM= ****** G        GETV   = ****** G
GETVAR= ****** G         HISTON= ****** G        INT    = ****** G
INTST  = ****** G        LPDISS= ****** G        LPDISX= ****** G
LPDISY= ****** G         LPSIP  = ****** G       LPSIVA= ****** G
MAXC    000062R          MAXD    000060R         MAXP    000064R
NARRAY= ****** G         NONRES  000030R         NUMSGN= ****** G
PC     =%000007          POLENT= ****** G        POLRET= 000134
PS     = 177776          PUSHR0  000466R         PUSH8   000474R
PUT    = 000004          PUTD    000512RG        REGSAV= ****** G
RESDIS  000000R          RESTOR= ****** G        RESTR2= ****** G
RETURN= 000207           RTSON  = ****** G       R0     =%000000
R0SAVE= 000044           R1     =%000001         R1SAVE= 000046
R2     =%000002          R2SAVE= 000050          R3     =%000003
R3SAVE= 000052           R4     =%000004         R5     =%000005
SAVER2= ****** G         SAVFAC= ****** G        SAVINT= ****** G
SP     =%000006          SS1SAV= 000024          SS2SAV= 000026
STODAT= ****** G         STORXT= ****** G        STOVAR= ****** G
TABLE  = ****** G        TRAFAC  000454R         TSTDXY  001056R
TST1    001070R          T1     = 000056         T2     = 000060
T3     = 000062          VARSAV= 000022          $DVR   = ****** G
$IR    = ****** G        $MLR   = ****** G       $POL8H= ****** G
.COMMA= ****** G         .LPAR  = ****** G       .RPAR  = ****** G

. ABS.  000000     000
        001462     001
ERRORS DETECTED: 0
FREE CORE: 17734. WORDS

LPS4,LP1=LPS4
```

6-12

```
PERPAR -- PERIPHERAL SUPPORT PA RT-11 MACRO VM02-09   16-OCT-74 02:06:07 PAGE 1

1               .TITLE PERPAR -- PERIPHERAL SUPPORT PACKAGE PARAMETER MODULE.
2               ;
3               ; DEC-11-LBPAA-A-LA     BASIC KERNEL V02-01
4               ;
5               ; COPYRIGHT (C) 1974
6               ;
7               ; DIGITAL EQUIPMENT CORPORATION
8               ; MAYNARD, MASSACHUSETTS 01754
9               ;
10              ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
11              ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
12              ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
13              ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
14              ; MAY APPEAR IN THIS DOCUMENT.
15              ;
16              ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
17              ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
18              ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
19              ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
20              ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
21              ;
22              ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
23              ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
24              ; WHICH IS NOT SUPPLIED BY DEC.
25
26              ; THE CONDITIONALS CONTAINED IN THIS MODULE AFFECT THE ASSEMBLY
27              ; OF THE FUNCTION TABLE MODULE "FTBL.MAC".
28              ; TO OBTAIN THE DESIRED CONDITIONAL DEFINITION(S),
29              ; REMOVE (USING AN EDITOR) THE
30              ; SEMI-COLON APPEARING BEFORE THE CONDITIONAL.
31              ;$DISK=0                ;DEFINE FOR RT-11
32                      .IFNDF $DISK
33      000000 $STRNG=0                 ;DO NOT DEFINE FOR PTS BASIC WITHOUT
34                                      ;STRINGS,- DEFINED FOR PTS V01 WITH STRINGS
35                      .ENDC           ;
36
37      000000 $LPS=0                   ;DEFINE FOR LPS
38                      .IFDF $LPS
39              ;$V=0                   ;DEFINE FOR LPS WITH VECTORS STARTING
40              ;                       ;    AT 300.  DEFAULT SETTING IS VECTORS AT
41              ;                       ;    340, SET $V = ANY OTHER DISPLACEMENT IF
42              ;                       ;       VECTORS START AT DISPLACEMENTS
43              ;                       ;       OTHER THAN 0 OR 40 FROM
44              ;                       ;       VECTOR 300
45
46      000000 $ADC=0                   ;INCLUDE A/D ROUTINES.
47      000000 $CLK=0                   ;INCLUDE CLOCK ROUTINES.
48      000000 $DIO=0                   ;INCLUDE DIGITAL IO ROUTINES
49      000000 $DIS=0                   ;INCLUDE DISPLAY ROUTINES.
50                      .ENDC   ;$LPS
51              ;
52              ;
53      000000 $VT11=0                  ;FOR GT40 (GT44)
54              ;
55              ;
56
57
```

52

```
                        .IFDF   $VT11
000000 $CLOCK=0                         ;FOR SYSTEM CLOCK (KW11L)
                        .ENDC

                        .EOT
                .TITLE  GTB  V01-01 BASIC - GT ROOT MODULE
                ;
                ;       DEC - 11 - LBPGB - A - LA    BASIC KERNEL V02-01
                ;
                ;       GTB.MAC TAPE 1 OF 4
                ;
                ; THE INFORMATION IN THIS LISTING IS SUBJECT TO CHANGE WITHOUT NOTICE
                ; AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
                ; CORPORATION, DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY
                ; FOR ANY ERRORS THAT MAY APPEAR IN THIS LISTING.
                ;
                ; THIS SOFTWARE IS FURNISHED TO THE PURCHASER UNDER A LICENSE
                ; FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH
                ; INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN
                ; SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY
                ; DIGITAL.
                ;
                ; DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE
                ; USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT
                ; SUPPLIED BY DIGITAL.
                ;
                ; COPYRIGHT (C) 1973,1974, BY DIGITAL EQUIPMENT CORPORATION.
                ;
                ;       AUTHOR: DR. STEPHEN R. ALPERT   AUGUST 1973
                        .IFDF   $DISK
                        .SBTTL  GLOBALS, EQUATES
                        .ENDC

000000                  $LPS=0          ; TAKEN CARE OF BY GTNLPS
                        .IFDF   $LPS
                        .GLOBL  NARRAY,TABLE,FLOAT
                        .ENDC
                        .IFNDF  $DEPTH
                        $DEPTH=10.              ; 10. CALLS DEFAULT
                        .ENDC

                        ; GLOBAL FROM BASICH OR BASICL FOR INTERRUPT
                        .GLOBL  USRAREA

                        ; GLOBALS FROM BASICL

                        .GLOBL  DSTOP,DISEND,ACTEND,BEG
                        .GLOBL  DCRASH,WD1,WD2,OLDMOD
                        .GLOBL  ,COMMA,.LPAR,.RPAR,.EOL
                        .GLOBL  EVAL,ERRSYN,ERRARG,INT
                        .GLOBL  GETVAR,STOVAR
                        .GLOBL  NUMOUT,MSG
                        .IFDF   $DISK
                        .GLOBL  IGNORE
                        .ENDC

                        ; GLOBALS FROM FPMP


GTB  V01-01 BASIC - GT ROOT MOD RT-11 MACRO VM02-09  16-OCT-74 02:06:07 PAGE 1+

                        .GLOBL $IR,$MLR,$DVR,$ADR,$SBR,$ERVEC

                        ; GLOBALS FOR INTER-MODULE COMMUNICATION
                        .GLOBL  NSTSK,INSRTF,TAG1,ACTST,DISEND
                        .GLOBL  SCAL,VECT,RDOT,APNT,STAT,TEXT
                        .GLOBL  ON,OFF,NOSC,TAGEND,ADSTK,BUFTST
                        .GLOBL  SUBP,ESUB,LPEN,TRAK,ERAS,CONTA,CKEOL
                        .GLOBL  INIT,INITA,DSTP,STOPA,CONT,XGRA,YGRA,AGET,APUT
                        .GLOBL  FIGR,FPUT,FIX0,FIXSP,FIX
                        .GLOBL  DPC,DSR,DISX,DISY,GTVECT
                        .IFDF   $CLOCK
                        .GLOBL  TIME,TIMR
                        .ENDC
                        .IFDF   $DISK
                        .GLOBL  SAV24,SAVERT,CLOSALL,FREE0,FREE
                        .ENDC
                        .GLOBL  SC,USC,TAGSRH,SCL.X,SCL.Y,ABS,F

                        ; EQUATES FROM BASIC USER AREA

000022                  VARSAV=22
000024                  SS1SAV=24
000026                  SS2SAV=26
000040                  FAC1=40
000042                  FAC2=42
000010                  ARRAYS=10
000012                  HIFREE=12
000014                  LOFREE=14
000044                  R0SAVE=44       ; FOR FPPSAV, FPPRES
                ;       THE FOLLOWING DEFINITION PAIR IS LEGITIMATE (IE, NOT
                ;IN THE VECTOR DEFINITION MODULE ["PERVEC"]) SINCE THEY
                ;ARE FIXED, THEY CANNOT FLOAT.
                        .IFDF   $CLOCK
000100 LKV=100                  ;CLOCK INTERRUPT ADRESS.
177546 LKS=177546               ;CLOCK STATUS REGISTER
                        .ENDC   ;$CLOCK


                ;       THE FOLLOWING IS TO INSURE THAT THE ORIGINAL
                ;       STACK POINTER DOESN'T CLOBBER ANYTHING IMPORTANT
                        .IFDF   $DISK
                        .ASECT
                        .=42
                        .WORD   DCRASH+400      ; REPLACE 24000 IN BASIC
                        .CSECT
                        .ENDC

                        .IFDF   $DISK
```

```
1                       .SBTTL  ASSIGNMENTS
2                       .ENDC
3               ;       ****************************************
4                       
5                       ; REGISTER ASSIGNMENTS
6       000000          R0=%0
7       000001          R1=%1
8       000002          R2=%2
9       000003          R3=%3
10      000004          R4=%4
11      000005          R5=%5
12      000006          SP=%6
13      000007          PC=%7
14      
15                      ; MISCELLANEOUS ASSIGNMENTS
16      177776          PSW=%2          ; PROCESSOR STATUS WORD
17      000207          RETURN=207      ; = RTS PC
18      000134          POLRET=134      ; = JMP @(R4)+
19      000200          PR4=200         ; PRIORITY 4
20      000340          PR7=340         ; PRIORITY 7
21      
22      160000          DJMP=160000     ; DISPLAY JMP
23      173400          $DINT=173400    ; STOP DISPLAY AND INTERRUPT 11
24      
25                      .IFDF   $DISK
```



```
1                       .SBTTL  MISCELLANEOUS ROUTINES
2                       .ENDC
3               ;       ****************************************
4                       ; MISCELLANEOUS ROUTINES
5       
6  000000       SAVINT:
7  000000 016546        MOV     FAC2(R5),-(SP)  ; SAVE 1=WORD INTEGER
          000042
8  000004 000134        POLRET                  ; = JMP @(R4)+
9       
10              ;       ****************************************
11      
12                      ; ROUTINE TO INTEGERIZE THE FAC TO A SINGLE WORD
13                      ; INTEGER, WILL CRASH IF ARGUMENT TOO LARGE
14                      ; RESULT ALSO IN FAC, DESTROYS R0
15      
16 00006        INTEGR:
17 00006 004767         JSR     PC,INT          ; GO TO "INT" OF BASICL
          000000G
18 00012 005765         TST     FAC1(R5)        ; DID WE SUCCEED?
          000040
19 00016 001001         BNE     INTEG0          ; BR IF > 32767
20 00020 000134         POLRET
21 00022 000167 INTEG0: JMP     ERRARG          ; ARGUMENT ERROR, TOO LARGE
          000000G
22      
23              ;       ****************************************
24      
25 00026        PLOOP:
26 00026 012467         MOV     (R4)+,LOOPCT    ; GET COUNT, !NOT REENTRANT!
          000022
27 00032 010467         MOV     R4,LPADR        ; SAVE LOOP ADDRESS
          000020
28 00036 000134         POLRET
29 00040        ELOOP:
30 00040 005367         DEC     LOOPCT          ; DECREMENT LOOP COUNT
          000010
31 00044 003402         BLE     ELOOP0          ; BR IF DONE
32 00046 016704         MOV     LPADR,R4        ; GET LOOP ADDRESS
          000004
33 00052 000134 ELOOP0: POLRET
34      
35 00054 000000 LOOPCT: .WORD 0                 ; NUMBER OF TIMES TO LOOP
36 00056 000000 LPADR:  .WORD 0                 ; ADR OF START OF LOOP
37
```

```
1                       ;       ****************************************
2
3 000060                SETNEC:
4 000060 005067         CLR     OPARGF          ; CLEAR OPTIONAL ARG, FLAG
         000010
5 000064 000134         POLRET
6 000066                SETOPT:
7 000066 010667         MOV     SP,OPARGF       ; SET OPTIONAL ARG, FLAG
         000002
8 000072 000134         POLRET
9
10 00074 000000 OPARGF: .WORD 0                 ; OPTIONAL ARGUMENT FLAG (=1 IF OPT)
11
12                      ;       ****************************************
13
14                      ; ROUTINE TO CK "," AND GET NEXT ARGUMENT INTO THE FAC
15                      ; IF OPTIONAL ARGUMENT FLAG IS SET AND ARGUMENT IS NOT
16                      ; PRESENT, 0 WILL BE RETURNED.
17                      ; BECAUSE OF "EVAL" FROM BASICL, USES ALL REGISTERS
18                      ; ON ENTRY: R1 MUST POINT TO WHERE COMMA SHOULD BE.
19                      ; *** NOTE THAT R4 HERE IS POLISH PC AND NOT THE
20                      ;       STACK SIZE THAT EVAL EXPECTS!!
21
22 00076                GETNUM:
23 00076 112102         MOVB    (R1)+,R2        ; GET NEXT TOKEN?
24 00100 120227         CMPB    R2,#',COMMA     ; IS IT A COMMA?
         000000G
25 00104 001414         BEQ     GETONE          ; YES- GET A NUMBER
26 00106 005767         TST     OPARGF          ; NO- WAS ARG. OPTIONAL?
         177762
27 00112 001415         BEQ     GETSYN          ; NO- ERROR
28 00114 120227         CMPB    R2,#',RPAR      ; YES- IS IT A ")" ?
         000000G
29 00120 001012         BNE     GETSYN          ; NO- ERROR
30 00122 005301         DEC     R1              ; YES- POINT AT IT
31 00124 005065 ZERNUM: CLR     FAC1(R5)
         000040
32 00130 005065         CLR     FAC2(R5)        ; PUT 0 INTO FAC
         000042
33 00134 000134         POLRET
34 00136                GETONE:
35 00136 004767         JSR     PC,EVAL         ; EVALUATE EXPRESSION
         000000G
36 00142 103403         BCS     GETARG          ; ERROR IF STRING
37 00144 000134         POLRET
38 00146 000167 GETSYN: JMP     ERRSYN          ; SYNTAX ERROR
         000000G
39 00152 000167 GETARG: JMP     ERRARG          ; ARGUMENT ERROR
         000000G
40
41                      ;       ****************************************
42
43                      ; IMMEDIATE POLISH FOR ONE ROUTINE
44
45 00156                ENTERP:
46 00156 012704         MOV     #RETURP,R4
         000166'
```



```
47 00162 017607         MOV     @(SP),PC        ; GO TO ROUTINE
         000000
48 00166 000170'RETURP: .+2                     ; RETURN HERE FROM ROUTINE
49 00170 062716         ADD     #2,(SP)         ; ADJUST FOR POLRET
         000002
50 00174 000207         RETURN
51
52                      ;       ****************************************
53
54 00176                NEGSTK:
55 00176 062716         ADD     #100000,(SP)    ; NEGATE STACK
         100000
56 00202 000134         POLRET
57
58                      ;       ****************************************
59
60                      ; ROUTINE TO TEST IF BUFFER ALLOCATED
61
62 00204 012746 BUFPOL: MOV     #POLENT,-(SP)   ; FAKE JSR PC
         000270'
63 00210                BUFTST:
64 00210 005767         TST     DCRASH          ; BUFFER ALLOCATED?
         010376
65 00214 001404         BEQ     BUFT0           ; NO- DO IT
66 00216 026727         CMP     DISEND,#-1      ; INIT BEEN DONE?
         000414
         177777
67 00224 001002         BNE     .+6             ; YES- BRANCH
68 00226 004767 BUFT0:  JSR     PC,INITA        ; INIT AND ALLOCATE BUFFER
         002252
69 00232 000207         RETURN
70
71 00234                PUSH:
72 00234 016546         MOV     FAC2(R5),-(SP)
         000042
73 00240 016546         MOV     FAC1(R5),-(SP)  ; PUSH FAC TO STACK
         000040
74 00244 000134         POLRET
75 00246                PUSH1:
76 00246 005046         CLR     -(SP)
77 00250 012746         MOV     #40200,-(SP)    ; PUT 1.0 ON STACK
         040200
78 00254 000134         POLRET
79 00256                POP:
80 00256 012665         MOV     (SP)+,FAC1(R5)
         000040
81 00262 012665         MOV     (SP)+,FAC2(R5)  ; POP STACK TO FAC
         000042
82 00266 000134         POLRET
```

```
                        ;       ****************************************

                                ; ROUTINE TO START POLISH AND CHECK FOR LEFT PAREN.

000270                  POLENT:
000270 012604                   MOV     (SP)+,R4        ; SET UP POLISH PC
000272 005067                   CLR     OPARGF          ; SET ARG, NECESSARY FLAG
        177576
000276 122127 CKLPAR:           CMPB    (R1)+,#,LPAR    ; FIRST TOKEN A "(" ?
        000000G
000302 001001                   BNE     CKLSYN          ; NO- ERROR
000304 000134                   POLRET
000306 000167 CKLSYN:           JMP     ERRSYN          ; SYNTAX ERROR
        000000G

00312                   QIKPOL:
00312 012604                    MOV     (SP)+,R4        ; SET UP POLISH PC
00314 000134                    POLRET                  ; AND ENTER POLISH MODE

                        ;       ****************************************

                                ; GENERAL EXIT ROUTINE WHICH CHECKS FOR RIGHT PAREN
                                ; CALLED BY :  JMP CKRPAR

00316                   CKRPAR:
00316 122127                    CMPB    (R1)+,#,RPAR    ; A RIGHT PAREN?
        000000G
00322 001004                    BNE     CKRPA0          ; BR IF NOT
00324                   CKEOL:
00324 121127                    CMPB    (R1),#,EOL      ; AND END-OF LINE?
        000000G
00330 001001                    BNE     CKRPA0          ; NO- ERROR
00332                   GETOUT:
                                .IFDF   $DISK
                                TST     (SP)+           ; CLEAN STACK
                                MOV     (SP)+,R5
                                MOV     (SP)+,R4
                                MOV     (SP)+,R1        ; RESTORE BASIC REGISTERS
                                JMP     IGNORE          ; EXIT THRU BASICR
                                .ENDC
                                .IFNDF  $DISK
00332 000207                    RETURN
                                .ENDC
00334 000167 CKRPA0:            JMP     ERRSYN          ; SYNTAX ERROR
        000000G

                        ;       ****************************************

                                ; ROUTINE TO SKIP AN ARGUMENT IN CALLING SEQUENCE
                                ; R1 POINTS TO FIRST TOKEN OF ARGUMENT AND ROUTINE
                                ; SEARCHES FOR A ",COMMA", EXIT POINTING TO ","

00340                   SKPARG:
00340 010046                    MOV     R0,-(SP)        ; SAVE R0 ON STK
00342 005000                    CLR     R0              ; ASSUME NO PARENTHESES
00344 005700 SKPA3:             TST     R0              ; ANY PARENTHESES YET?
00346 001003                    BNE     SKPA4           ; YES- BRANCH
```



```
00350 121127                    CMPB    (R1),#,COMMA    ; DID WE FIND A COMMA?
        000000G
00354 001416                    BEQ     SKPA0           ; YES- EXIT
00356 121127 SKPA4:             CMPB    (R1),#,LPAR     ; A "("?
        000000G
00362 001001                    BNE     SKPA1           ; NO- KEEP LOOKING
00364 005200                    INC     R0              ; YES- INCREMENT NEST DEPTH
00366 121127 SKPA1:             CMPB    (R1),#,RPAR     ; A ")"?
        000000G
00372 001002                    BNE     SKPA2           ; NO- BRANCH
00374 005300                    DEC     R0              ; DECREMENT NEST COUNT
00376 100405                    BMI     SKPA0           ; BRANCH IF EXTRA ")"
00400 122127 SKPA2:             CMPB    (R1)+,#,EOL     ; END OF LINE ?
        000000G
00404 001357                    BNE     SKPA3           ; NO- LOOK SOME MORE
00406 000167                    JMP     ERRSYN          ; YES- ERROR
        000000G
00412 012600 SKPA0:             MOV     (SP)+,R0        ; RESTORE R0
00414 000134                    POLRET                  ; RETURN IN POLISH


                        ;       POLISH REGISTER SAVE AND RESTORE
                                ; FPPSAV CALLED BY JSR R4
                                ; FPPRES CALLED IN POLISH MODE

00416                   FPPSAV:
00416 062705                    ADD     #R0SAVE,R5      ; POINT TO SAVE AREA
        000044
00422 010025                    MOV     R0,(R5)+
00424 010125                    MOV     R1,(R5)+
00426 010225                    MOV     R2,(R5)+
00430 010325                    MOV     R3,(R5)+
00432 012625                    MOV     (SP)+,(R5)+
00434 162705                    SUB     #R0SAVE+12,R5
        000056
00440 000134                    POLRET                  ; ENTER POLISH MODE

00442                   FPPRES:
00442 062705                    ADD     #R0SAVE,R5
        000044
00446 012500                    MOV     (R5)+,R0
00450 012501                    MOV     (R5)+,R1
00452 012502                    MOV     (R5)+,R2
00454 012503                    MOV     (R5)+,R3
00456 012546                    MOV     (R5)+,-(SP)
00460 162705                    SUB     #R0SAVE+12,R5
        000056
00464 000204                    RTS     R4              ; RETURN IN EXEC MODE

                                .IFDF   $DISK
```

```
1                       .SBTTL  STOP DISPLAY INTERRUPT ROUTINE
2                       .ENDC
3                ;      ************************************
4
5                       ; ROUTINE TO PROCESS "SDINT" FROM DPU
6                       ;  CASE 1: CALL TO SUBPICTURE, THEN CONTENTS OF X
7                       ;          (IN DPC) ARE GREATER THAN X+2
8                       ;       X IS THE ADDRESS OF THE WORD AFTER "SDINT"
9                       ;  CASE 2: EXIT FROM SUBP, THEN CONTENTS OF X = 0
10                      ;  CASE 3: ADD ON AT END OF FILE, THEN
11                      ;          CONTENTS OF X < X+2
12
13 00466        DISINT:
14 00466 022767         CMP     #DSTOP,DPC      ; DID WE STOP IN TIGHT LOOP?
         010610'
         000000G
15 00474 001004         BNE     DIS0            ; NO- CONTINUE
16 00476 012767         MOV     #DJMP,DSTOP+2   ; RESTORE FILE
         160000
         010102
17 00504 000002         RTI                     ; YES- DON'T RESTART IT, EXIT
18 00506 010046 DIS0:   MOV     R0,-(SP)        ; SAVE R0
19 00510 010246         MOV     R2,-(SP)        ; SAVE R2
20 00512 016700         MOV     DPC,R0          ; WHERE DID DPU STOP?
         000000G
21 00516 012002         MOV     (R0)+,R2        ; R2 CONTAINS CONTENTS OF X
22                                              ; R0 CONTAINS X+2
23 00520 001414         BEQ     SUB0            ; BR IF CASE 2: END OF SUBPICTURE
24 00522 020002         CMP     R0,R2           ; COMPARE X+2 WITH (X)
25 00524 101022         BHI     SUB2            ; BR IF CASE 3: ADD TO DPY FILE
26 00526 062767         ADD     #2,ADSTK        ; INCREMENT DPY ADR STK PTR
         000002
         000626
27 00534 010077         MOV     R0,@ADSTK       ; STORE X+2 IN STACK
         000622
28 00540 011067         MOV     (R0),DPC        ; START DPC GOING IN SUBPICTURE
         000000G
29 00544 012602 SUB1:   MOV     (SP)+,R2        ; RESTORE R2
30 00546 012600         MOV     (SP)+,R0        ; RESTORE R0
31 00550 000002         RTI                     ; AND RETURN FROM INTERRUPT
32 00552 017700 SUB0:   MOV     @ADSTK,R0       ; PUT X+2 INTO R0
         000604
33 00556 162767         SUB     #2,ADSTK        ; DECREMENT DPY ADR STK PTR
         000002
         000576
34 00564 014067         MOV     -(R0),DPC       ; PUT (X) INTO DPC
         000000G
35 00570 000765         BR      SUB1            ; AND EXIT
36 00572 016740 SUB2:   MOV     WD2,-(R0)       ; INSERT WORD 2
         000144
37 00576 016740         MOV     WD1,-(R0)       ; INSERT WORD 1
         000136
38 00602 016700         MOV     DISEND,R0       ; GET NEW END OF DPY FILE +2
         000030
39 00606 005740         TST     -(R0)           ; POINT AT ACTUAL END
40 00610 010067         MOV     R0,ACTEND       ; STORE IT
         000024
```



```
41 00614 005740         TST     -(R0)           ; POINT TO NEW DISEND
42 00616 010067         MOV     R0,DISEND       ; STORE IT
         000014
43 00622 005067         CLR     INSRTF          ; CLEAR INSERT PENDING FLAG
         000006
44 00626 004767         JSR     PC,INIT3        ; RESUME THE DPU
         002036
45 00632 000744         BR      SUB1            ; AND EXIT
46
47 00634 000000 INSRTF: .WORD 0                 ; INSERT PENDING FLAG (WHEN SET)
48 00636 177777 DISEND: .WORD -1                ; POINTER TO NEW SLOT IN DPY FILE
49 00640 000000 ACTEND: .WORD 0                 ; POINTS TO END OF RUNNING DPY FILE
50 00642 000000 ACTST:  .WORD   0               ; LO=END OF DPY FILE
51                                              ; PRECEDED BY 1 OR 2 WORDS
52                                              ; (2 IF $DISK DEFINED)
53                      .IFDF   $DISK
54
```

```
                        .SBTTL  INSERTION INTO DPY FILE ROUTINES
                        .ENDC
                ;       ************************************
                ;
                ; ROUTINE TO INSERT WORDS INTO DPY FILE. THE FIRST TWO WORDS
                ; ARE SAVED IN WD1, AND WD2, TO OVERLAY THE LAST TWO ENTRIES
                ; IN THE RUNNING DISPLAY FILE.
                ; CALLED BY: JSR  PC,PUTWD
                ;   WORD TO BE INSERTED MUST BE IN R0
                
                
00644           NEWPUT:
00644 005067    NEWPUT: CLR     OLDMOD          ; SO NEXT SGM WILL BE INSERTED
      001114
00650           PUTWD:
00650 010246    PUTWD:  MOV     R2,-(SP)        ; SAVE R2 ON STK
00652 012702    PUT5:   MOV     #DISEND,R2      ; GET CURRENT END OF DPY FILE
      000636'
00656 026712            CMP     DCRASH,(R2)     ; END OF DPY FILE
      007730
00662 103003            BHIS    PUT7            ; NO= CONTINUE
00664           DFOERR:
00664 104400    DFOERR: TRAP    0
                        .IFNDF  $LONGER
00666   104             .ASCII  /DFO/
00667   106
00670   117
                        .ENDC
                        .IFDF   $LONGER
                        .ASCII  /DISPLAY FILE OVERFLOW/
                        .ENDC
00671   000             .BYTE   0
                        .EVEN
00672 005767    PUT7:   TST     INSRTF          ; IS THERE AN INSERT PENDING?
      177736
00676 001375            BNE     PUT7            ; YES= WAIT UNTIL IT IS FREE
00700 026712    PUT6:   CMP     ACTEND,(R2)     ; HAVE WE INSERTED ANY WORDS YET?
      177734
00704 101007            BHI     PUT1            ; NO= STORE IN WD1
00706 001411            BEQ     PUT2            ; YES= ONE= STORE IN WD2
00710 010072            MOV     R0,0(R2)        ; MORE THAN TWO, PUT IT IN LINE,
      000000
00714 062712    PUT3:   ADD     #2,(R2)         ; INCREMENT DPY END PTR
      000002
00720 012602            MOV     (SP)+,R2        ; RESTORE R2
00722 000207            RETURN                  ; EXIT
00724 010067    PUT1:   MOV     R0,WD1          ; STORE IN WORD 1
      000010
00730 000771            BR      PUT3            ; AND EXIT
00732 010067    PUT2:   MOV     R0,WD2          ; STORE IN WORD 2
      000004
00736 000766            BR      PUT3            ; AND EXIT
                
00740 000000    WD1:    .WORD 0                 ; FIRST TEMPORARY WORD
00742 000000    WD2:    .WORD 0                 ; SECOND TEMPORARY WORD FOR DPY FILE
                
                ;       ************************************
```



```
                
                ; ROUTINE TO SET UP THE INSERT INTO THE DPY FILE.
                ; THIS ROUTINE WILL BE CALLED BY EACH ROUTINE AFTER IT
                ; HAS DONE ALL OF ITS "PUTWD", IT IS USUALLY FOLLOWED BY
                ; A JUMP TO CKRPAR.  CALLED BY:  JSR PC, INSERT
                
00744           INSERT:
00744 005777    INSERT: TST     @NSTSK          ; HOW ARE WE DOING ON SUBP?
      000442
00750 001015            BNE     INSXIT          ; NOT 0= STILL IN A SUBP
00752 012700            MOV     #DJMP,R0        ; PUT DJMP INTO R0
      160000
00756 004767            JSR     PC,PUTWD        ; STORE AT END OF DPY FILE
      177666
00762 016702            MOV     ACTEND,R2       ; GET LOOP ADDRESS
      177652
00766 011200            MOV     (R2),R0         ; KEEP THE POINTER TOO
00770 004767            JSR     PC,PUTWD        ; STORE IT AT NEW END OF DPY FILE
      177654
00774 010667            MOV     SP,INSRTF       ; SET INSERT PENDING FLAG
      177634
01000 012742            MOV     #SDINT,-(R2)    ; PUT SDINT AT OLD END
      173400
01004 000207    INSXIT: RETURN                  ; = RTS PC
                        .IFDF   $DISK
```

```
1                       .SBTTL  "SUBP" ROUTINE
2                       .ENDC
3                ;      ****************************************
4
5                       ; ROUTINE TO HANDLE:
6                       ;       CALL "SUBP" (T [,T1])
7
8 001006                SUBP:
9 001006 004767         SUBP:   JSR     PC,BUFPOL       ; ENTER POLISH AND CHECK "("
         177172
10 01012 000136'                +GETONE         ; GET FIRST PARAMETER
11 01014 000006'                +INTEGR         ; INTEGERIZE TO 32767 OR LESS
12 01016 000000'                +SAVINT         ; SAVE INTEGER ON STACK
13 01020 000066'                +SETOPT         ; T1 IS OPTIONAL ARGUMENT
14 01022 000076'                +GETNUM         ; CHK "," AND GET T1 (ELSE 0)
15 01024 000006'                +INTEGR         ; INTEGERIZE TO 32767 OR LESS
16 01026 000000'                +SAVINT         ; SAVE T1 ON THE STACK
17 01030 001032'                .+2             ; AND EXIT FROM POLISH MODE
18 01032 002002         SUBP0:  BGE     SUBP0           ; BR IF T1 >= 0
19 01034 000167 SUBP1:  JMP     ERRARG          ; ARGUMENT ERROR
         000000G
20 01040 016602 SUBP0:  MOV     2(SP),R2        ; MOVE T TO R2
         000002
21 01044 003773         BLE     SUBP1           ; BR IF T <= 0
22 01046 004767         JSR     PC,TAGSRH       ; LOOK FOR T IN FILE, BETTER NOT FIND IT
         000342
23 01052 005700         TST     R0
24 01054 001367         BNE     SUBP1           ; ERROR= TAG ALREADY IN USE
25 01056 012700         MOV     #SDINT,R0
         173400
26 01062 004767         JSR     PC,PUTWD        ; PUT IN SDINT FOR SUBP CALL
         177562
27 01066 016700         MOV     DISEND,R0
         177544
28 01072 062700         ADD     #10,R0          ; ADR OF THE SUBP
         000010
29 01076 012602         MOV     (SP)+,R2        ; PUT T1 INTO R2
30 01100 011646         MOV     (SP),-(SP)      ; MOVE T DOWN ONE PLACE
31 01102 010266         MOV     R2,2(SP)        ; PUT T1 BACK ON STK
         000002
32 01106 001022         BNE     SUBP2           ; BR IF 2 PARAMETER CASE
33 01110 005020         CLR     (R0)+           ; ZERO WORD BEFORE SUBP
34 01112 062767         ADD     #2,NSTSK        ; ADD 2 TO CALL SUBP PTR
         000002
         000272
35 01120 022767         CMP     #NSTSK,NSTSK    ; ARE WE POINTING TO OURSELVES?
         001412'
         000264
36 01126 101407         BLOS    NSTERR          ; YES= OVERFLOW STACK DEPTH
37 01130 016777         MOV     DISEND,@NSTSK   ; NO= SAVE PTR FOR REST-OF-FILE PTR
         177502
         000254
38 01136 062767         ADD     #2,DISEND       ; BUMP PTR PAST THAT WORD
         000002
         177472
39 01144 000413         BR      SUBP3           ; CONTINUE
40
```



```
41 01146                NSTERR:
42 01146 104400         NSTERR: TRAP    0
43                      .IFNDF  $LONGER
44 01150     111        .ASCII  /INS/
   01151     116
   01152     123
45                      .ENDC
46                      .IFDF   $LONGER
47                      .ASCII  /IMPROPERLY NESTED SUBPICTURES/
48                      .ENDC
49 01153     000        .BYTE   0
50                      .EVEN
51
52 01154 004767 SUBP2:  JSR     PC,PUTWD        ; PTR TO REST OF FILE
         177470
53 01160 004767         JSR     PC,TAGSRH       ; FIND T1, HOPEFULLY
         000230
54 01164 005700         TST     R0
55 01166 001722         BEQ     SUBP1           ; FAILED TO FIND IT= ARG, ERROR
56 01170 016000         MOV     -4(R0),R0       ; GET START ADR OF SUBP
         177774
57 01174 004767 SUBP3:  JSR     PC,PUTWD        ; PUT ADR INTO DPY FILE
         177450
58 01200 016777         MOV     DISEND,@TAGEND  ; UPDATE TAG LIST
         177432
         000230
59 01206 012600         MOV     (SP)+,R0        ; GET TAG T
60 01210 004767         JSR     PC,PUTWD        ; INSERT IT INTO DPY FILE
         177434
61 01214 016767         MOV     DISEND,TAGEND   ; FIX PTR TO END OF LIST
         177416
         000214
62 01222 005000         CLR     R0
63 01224 004767         JSR     PC,NEWPUT       ; NEXT TAG PTR=0/ NEW SGM
         177414
64 01230 004767         JSR     PC,INSERT       ; INSERT IF 2-PARAM CALL
         177510
65 01234 012600         MOV     (SP)+,R0        ; GET T1 AGAIN
66 01236 001002         BNE     .+6             ; 2 PARAM, CASE= BRANCH
67 01240 004767         JSR     PC,PUTWD        ; ELSE INSERT A ZERO
         177404
68 01244 000167         JMP     CKRPAR          ; AND EXIT
         177046
69                      .IFDF   $DISK
```

```
1                       .SBTTL  "ESUB" ROUTINE
2                       .ENDC
3               ;       ****************************************
4
5                       ; ROUTINE TO HANDLE:
6                       ;       CALL "ESUB"
7
8 001250        ESUB:
9 001250 004767         JSR     PC,BUFTST       ; ANY BUFFER YET?
         176734
10 01254 012700         MOV     #SDINT,R0       ; PUT SDINT INTO R0
         173400
11 01260 004767         JSR     PC,PUTWD        ; INSERT AT END OF DPY FILE
         177364
12 01264 005000         CLR     R0
13 01266 004767         JSR     PC,PUTWD        ; FOLLOWED BY A 0
         177356
14 01272 016700         MOV     DISEND,R0       ; SAVE DISEND AND USE IT TOO
         177340
15 01276 017767         MOV     @NSTSK,DISEND   ; WHERE TO PUT IT
         000110
         177332
16 01304 001720         BEQ     NSTERR          ; TOO MANY ESUBS IF = 0
17 01306 004767         JSR     PC,NEWPUT       ; INSERT INTO DPY FILE/ NEW SGM
         177332
18 01312 010067         MOV     R0,DISEND       ; STORE DISEND
         177320
19 01316 162767         SUB     #2,NSTSK        ; DROP CALL PTR BY 2
         000002
         000066
20 01324 004767         JSR     PC,INSERT       ; CAN INSERT NOW
         177414
21 01330 000167 ESXIT:  JMP     CKEOL           ; EXIT TO USER
         176770
22
23              ; THE STACK DEPTH OF SUBP IS AN ASSEMBLY
24              ; PARAMETER INSTEAD OF DEFAULTING TO A VALUE OF 10
25              ; *** IT IS THE PARAMETER, $DEPTH
26              ; $DEPTH INCLUDES CALLS TO 2 PARAMETER SUBP
27
28              ;       ****************************************
29
30                      ; STORAGE FOR POINTERS IN SUBP AND ESUB
31
32 01334 000000 ADBAS:  .WORD 0                 ; WORD TO DETECT BOTTOM OF DPY STK
33      001362'         .=.+$DEPTH+$DEPTH
34 01362 000000 ADSTK:  .WORD 0                 ; POINTER TO DPY STACK
35 01364 000000 NSBAS:  .WORD 0                 ; WORD TO DETECT BOTTOM OF NEST
36      001412'         .=.+$DEPTH+$DEPTH
37 01412 000000 NSTSK:  .WORD 0                 ; POINTER TO NEST STACK
38                      .IFDF   $DISK
```



```
1                       .SBTTL  MISCELLANEOUS ROUTINES
2                       .ENDC
3               ;       ****************************************
4
5                       ; ROUTINE TO FIND A GIVEN TAG (PASSED IN R2)
6                       ; ON EXIT R0 = 0 IF TAG NOT FOUND
7                       ; ELSE
8                       ; R0 POINTS TO THE PTR FOLLOWING THE TAG
9                       ; R3 POINT TO THE PTR POINTING TO THE TAG
10                      ;  CALLED BY JSR  PC,TAGSRH
11                      ;  R0 AND R3 ARE CHANGED, R2 NOT CHANGED
12
13 01414        TAGSRH:
14 01414 016700         MOV     TAGHED,R0       ; GET START ADR OF TAG LIST
         000014
15 01420 022002 TGLOOP: CMP     (R0)+,R2        ; DO TAGS MATCH?
16 01422 001403         BEQ     TAGXIT          ; YES- EXIT
17 01424 010003         MOV     R0,R3           ; R1 POINTS TO PTR POINTING TO TAG
18 01426 011000         MOV     (R0),R0         ; GET ADR OF NEXT PTR
19 01430 001373         BNE     TGLOOP          ; LOOP IF NOT AT END OF LIST
20 01432 000207 TAGXIT: RETURN
21
22 01434 010514'TAGHED: .WORD TAG1              ; POINTER TO FIRST TAG
23 01436 000000 TAGEND: .WORD 0                 ; POINTER TO PTR AFTER LAST TAG
24
25              ;       ****************************************
26
27                      ; POLISH ROUTINE TO TEST IF R0 < 64
28                      ; XYBIG IS INCREMENTED IF IT IS >= 64
29                      ; USED IN ARGGET AND OTHER PLACES
30                      ; SAVE IT ON THE STACK
31
32 01440        TST64:
33 01440 020027         CMP     R0,#100         ; COMPARE WITH 64
         000100
34 01444 002402         BLT     SAVER0          ; BR IF < 64
35 01446 005267         INC     XYBIG           ; INCREMENT COUNTER OF BIG VALUES
         000356
36 01452        SAVER0:
37 01452 010046         MOV     R0,-(SP)        ; SAVE R0 ON THE STACK
38 01454 000134         POLRET                  ; RETURN IN POLISH
39                      .IFDF   $DISK
40                      .EOT
```

```
2                       ;
3                       ;       DEC - 11 - LBPGB - A - LA     BASIC KERNEL V02-01
4                       ;
5                       ;       GTB,MAC TAPE 2 OF 4
6                       ;
7                       ; THE INFORMATION IN THIS LISTING IS SUBJECT TO CHANGE WITHOUT NOTICE
8                       ; AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
9                       ; CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY
10                      ; FOR ANY ERRORS THAT MAY APPEAR IN THIS LISTING.
11                      ;
12                      ; THIS SOFTWARE IS FURNISHED TO THE PURCHASER UNDER A LICENSE
13                      ; FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH
14                      ; INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN
15                      ; SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY
16                      ; DIGITAL.
17                      ;
18                      ; DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE
19                      ; USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT
20                      ; SUPPLIED BY DIGITAL.
21                      ;
22                      ; COPYRIGHT (C) 1973,1974, BY DIGITAL EQUIPMENT CORPORATION.
23                      ;
24                      ;       AUTHOR: DR. STEPHEN R. ALPERT   AUGUST 1973
25                              .SBTTL  "ERAS" ROUTINE
26                              .ENDC
27                      ;       ****************************************
28
29                              ; ROUTINE TO HANDLE
30                              ;       CALL "ERAS" [( T)]
31                              ;  IF T IS NOT SPECIFIED, THEN ERASE TRAK
32
33 01456               ERAS:
34 01456 004767                 JSR     PC,BUFTST       ; ANY BUFFER YET?
         176526
35 01462 121127                 CMPB    (R1),#,EOL      ; END OF THE LINE?
         000000G
36 01466 001005                 BNE     ERAS0           ; NO - GET T
37 01470 012767                 MOV     #DJMP,BEG       ; DJMP OVER TRAK
         160000
         007010
38 01476 000167                 JMP     CKEOL           ; EXIT TO USER
         176622
39 01502 004767 ERAS0:  JSR     PC,POLENT       ; ENTER POLISH MODE AND CHECK "("
         176562
40 01506 000136'                .GETONE         ; GET T
41 01510 000006'                .INTEGR         ; INTEGERIZE TO 32767 OR LESS
42 01512 001514'                .+2             ; EXIT POLISH
43 01514 016502                 MOV     FAC2(R5),R2     ; GET T AS AN INTEGER
         000042
44 01520 003002                 BGT     ERAS2           ; BR IF T LEGAL
45 01522 000167 ERAS1:  JMP     ERRARG          ; ARGUMENT ERROR IF T<=0
         000000G
46 01526 004767 ERAS2:  JSR     PC,TAGSRH       ; FIND THE TAG IN DPY FILE
         177662
47 01532 005700                 TST     R0              ; DID WE SUCCEED?
48 01534 001772                 BEQ     ERAS1           ; ARG, ERROR IF CAN'T FIND IT
49 01536 012760                 MOV     #DJMP,-10(R0)   ; REPLACE SDINT BY DJMP
```



```
         160000
         177770
50 01544 005060                 CLR     -2(R0)          ; MAKE THE TAG ZERO - FOR "SAVE"
         177776
51 01550 016002                 MOV     -6(R0),R2       ; GET ADR OF REST OF FILE
         177772
52 01554 011013 ERAS5:  MOV     (R0),(R3)       ; MOVE LINK
53 01556 001405                 BEQ     ERAS3           ; BRANCH IF END
54 01560 021002                 CMP     (R0),R2         ; NEXT SUBP IN REST OF FILE?
55 01562 101005                 BHI     ERAS4           ; YES- BRANCH
56 01564 011000                 MOV     (R0),R0         ; GET NEXT LINK
57 01566 005020                 CLR     (R0)+           ; ZERO TAG FOR "SAVE"
58 01570 000771                 BR      ERAS5           ; LOOP
59 01572 010367 ERAS3:  MOV     R3,TAGEND       ; ADJUST END OF LINK
         177640
60 01576 000167 ERAS4:  JMP     CKRPAR          ; CK ")" AND EXIT
         176514
61                              .IFDF   $DISK
```

```
                        ,SBTTL  "LPEN" ROUTINE (NOT INTERRUPT)
                        ,ENDC
                ;       ****************************************

                        ; ROUTINE TO HANDLE
                        ;       CALL "LPEN" (H, TAG [,X, Y])
                        ; DOESN'T CHECK REST OF ARGS IF H=0

001602          LPEN:
001602 004767           JSR     PC,BUFPOL       ; ENTER POLISH MODE AND CK "("
       176376
001606 001610'                  ,+2             ; EXIT POLISH FOR NOW
001610 004767           JSR     PC,GETADR       ; GET ADR OF H INTO R2
       000222
001614 005022           CLR     (R2)+           ; CLEAR HI-ORDER WORD
001616 016712           MOV     LPEN,F,(R2)     ; STORE HIT VARIABLE
       000056
001622 004767 LPEN0:    JSR     PC,ENTERP       ; IMMED, POLISH
       176330
001626 001702'                  +CKOPCM         ; JUMP BY THE COMMA
001630 004767           JSR     PC,GETADR       ; GET ADR OF TAG INTO R2
       000202
001634 005022           CLR     (R2)+           ; CLEAR HI-ORDER WORD
001636 016712           MOV     LPENT,(R2)      ; STORE THE TAG
       000024
001642 012700           MOV     #LPENX,R0       ; PUT ADDRESS OF SOURCE
       001670'
001646 004767           JSR     PC,OPTSTR       ; STORE OPTIONAL ARGUMENT X
       004444
001652 004767           JSR     PC,OPTSTR       ; STORE OPTIONAL ARGUMENT Y
       004440
001656 005067           CLR     LPEN,F          ; CLEAR LPEN HIT FLAG
       000016
001662 000167           JMP     CKRPAR          ; CHECK ")" AND RETURN
       176430

001666 000000 LPENT:    ,WORD 0                 ; LPEN TAG STORAGE
001670 000000 LPENX:    ,WORD 0,0               ; X COORD, OF LP HIT
001672 000000
001674 000000 LPENY:    ,WORD 0,0               ; Y COORD, OF LP HIT
001676 000000
001700 000000 LPEN,F:   ,WORD 0                 ; LPEN HIT FLAG ( HIT = 1)
                        ,IFDF   $DISK
```



```
                        ,SBTTL  MISCELLANEOUS ROUTINES
                        ,ENDC
                ;       ****************************************

                        ; ROUTINE TO CHECK (POSSIBLY) OPTIONAL COMMA
                                        ;  USES R2

001702          CKOPCM:
001702 112146           MOVB    (R1)+,-(SP)     ; GET THE TOKEN IN QUESTION
001704 000241           CLC                     ; ASSUME FOUND
001706 121627           CMPB    (SP),#,COMMA    ; COMPARE WITH COMMA
       000000G
001712 001410           BEQ     CKOPC0          ; BR IF A MATCH
001714 005767           TST     OPARGF          ; NOT A MATCH, OPTIONAL?
       176154
001720 001407           BEQ     CKOSYN          ; NOT OPTIONAL, ERROR
001722 121627           CMPB    (SP),#,RPAR     ; IS IT A ")" ?
       000000G
001726 001004           BNE     CKOSYN          ; NO- ERROR
001730 005301           DEC     R1              ; POINT AT ")"
001732 000261           SEC                     ; FLAG COMMA NOT FOUND
001734 005726 CKOPC0:   TST     (SP)+           ; CLEAN STACK
001736 000134           POLRET                  ; AND EXIT
001740 000167 CKOSYN:   JMP     ERRSYN          ; SYNTAX ERROR
       000000G

                        ; ROUTINE TO SET-GRAPHIC-MODE FROM R0
                        ; CHECKS TO SEE IF SAME AS PREVIOUS MODE

001744          PUTSGM:
001744 020067           CMP     R0,OLDMOD       ; DIFFERENT MODE?
       000014
001750 001404           BEQ     PUTS0           ; NO- BRANCH
001752 004767           JSR     PC,PUTWD        ; YES- INSERT IT
       176672
001756 010067           MOV     R0,OLDMOD       ; SET "NEW" OLDMOD
       000002
001762 000207 PUTS0:    RETURN                  ; EXIT

001764 000000 OLDMOD:   ,WORD 0                 ; VALUE OF PREVIOUS OLD SET-G-M
                        ,IFDF   $DISK
```

```
1                       .SBTTL  ARGUMENT GET FOR APNT,RDOT AND VECT
2                       .ENDC
3               ;       ************************************
4
5                       ; ROUTINE TO SUPPLY ALL ARGUMENTS FOR "VECT",
6                       ; "APNT" AND "RDOT"
7                       ; ON EXIT:
8                       ;       R0: BITS 10:0 OF SET-GRAPHIC-MODE
9                       ;       (SP): VALUE OF Y
10                      ;       2(SP): VALUE OF X
11                      ;       XYBIG: NO. OF ARGS. (X,Y) EXCEEDING 63
12                      ;       XSIGN: BIT 13 ON IF X NEGATIVE
13                      ;       YSIGN: BIT 6 ON IF Y NEGATIVE
14                      ;  CALLED VIA:  JSR  PC,GETARG
15
16 01766        ARGGET:
17 01766 012667         MOV     (SP)+,GETSAV            ; SAVE RETURN ADR OFF STK
         000034
18 01772 005067         CLR     XYBIG                   ; ASSUME X,Y < 64
         000032
19 01776 004767         JSR     PC,POLENT               ; ENTER POLISH AND CK "("
         176266
20 02002 000136'                +GETONE                 ; GET X OR DX
21 02004 004770'                +SCALEX                 ; SCALE IT
22 02006 001440'                +TST64                  ; CHECK IF < 64 AND ONTO STK
23 02010 000076'                +GETNUM                 ; CK "," AND GET Y
24 02012 005042'                +SCALEY                 ; SCALE IT
25 02014 001440'                +TST64                  ; TST IF < 64 AND ONTO STK
26 02016 003050'                +LIFT                   ; GET L, I, F, T
27 02020 002022'                .+2             ; EXIT FROM POLISH
28 02022 000177         JMP     @GETSAV                 ; RETURN TO CALLER
         000000
29
30 02026 000000 GETSAV: .WORD 0                 ; RETURN ADDRESS
31 02030 000000 XYBIG:  .WORD 0                 ; VALUES > 63
32 02032 000000 XSIGN:  .WORD 0                 ; BIT 13 ON IF X NEG
33 02034 000000 YSIGN:  .WORD 0                 ; BIT 6 ON IF Y NEG
34                      .IFDF   $DISK
```



```
1                       .SBTTL  ADDRESS CALCULATION ROUTINE
2                       .ENDC
3               ;       ************************************
4
5
6                       ; ROUTINE TO FETCH ADDRESS FOR STORING VARIABLES
7                       ; IN ARRAYS, IF SS2<=1 (I.E. XGRA,YGRA, OR FIGR )
8                       ; THEN ONLY ONE WORD ENTRIES ARE ASSUMED,
9                       ;  USES R1, R2, R3,  SAVES R0
10                      ;  CRASHES IF CALLED WHEN R1 POINTS TO CONSTANT
11                      ; DESTROYS THE FAC IF SUBSCRIPTS INVOLVED
12
13 02036        GETADR:
14 02036 010046         MOV     R0,-(SP)        ; SAVE R0
15 02040 112102         MOVB    (R1)+,R2        ; BUILD WORD OFFSET
16 02042 100535         BMI     GETAD3          ; CK IF OPTIONAL ARGUMENT
17 02044 000302         SWAB    R2
18 02046 152102         BISB    (R1)+,R2        ; GET SECOND HALF
19 02050 061502         ADD     (R5),R2         ; ADD PTR TO USER AREA
20 02052 021227         CMP     (R2),#-2        ; IS IT A SCALAR OR ARRAY?
         177776
21 02056 003070         BGT     ADRARG          ; ARGUMENT ERROR IF STRING
22 02060 010246         MOV     R2,-(SP)        ; SAVE R2
23 02062 004767         JSR     PC,GETVAR       ; GET ADR AND SUBSCRIPTS
         000000G
24 02066 012602         MOV     (SP)+,R2        ; RESTORE R2
25 02070 022227         CMP     (R2)+,#-3       ; WAS IT A SCALAR?
         177775
26 02074 001457         BEQ     GETAD0          ; YES- BRANCH
27 02076 026227         CMP     4(R2),#-2       ; IS SS2 FLAGGED FOR XGRA ETC.?
         000004
         177776
28 02104 003461         BLE     GETAD1          ; YES- BRANCH
29 02106 005742         TST     -(R2)           ; FIX UP R2
30 02110 016500         MOV     SS1SAV(R5),R0   ; SET UP SS1
         000024
31 02114 016503         MOV     SS2SAV(R5),R3   ; SET UP SS2
         000026
32 02120 020062         CMP     R0,4(R2)        ; SS1 > SS1MAX?
         000004
33 02124 003040         BGT     ERSS3           ; YES- ERROR
34 02126 005703         TST     R3              ; 2 DIMENSIONAL?
35 02130 100427         BMI     LOCNO2          ; NO- BRANCH
36 02132 005762         TST     6(R2)           ; YES- IS IT DIMENSIONED OKAY?
         000006
37 02136 100433         BMI     ERSS3           ; NO- ERROR
38 02140 020362         CMP     R3,6(R2)        ; SS2 > SS2MAX?
         000006
39 02144 101030         BHI     ERSS3           ; YES- ERROR
40 02146 016246         MOV     4(R2),-(SP)     ; PUT SS2MAX ON STK
         000004
41 02152 005216         INC     (SP)
42 02154 010346         MOV     R3,-(SP)
43 02156 012746         MOV     #20,-(SP)
         000020
44 02162 006303 LCLOOP: ASL     R3
45 02164 006366         ASL     2(SP)
```

```
                000002
46 02170 103002         BCC     .+6             ; MULTIPLY SS2 BY SS1MAX
47 02172 066603         ADD     4(SP),R3
                000004
48 02176 005316         DEC     (SP)            ; AND ADD SS1
49 02200 003370         BGT     LCLOOP
50 02202 062706         ADD     #6,SP           ; CLEAN STK
                000006
51 02206 060300         ADD     R3,R0
52 02210 005200 LOCNO2: INC     R0
53 02212 006300         ASL     R0              ; 2 BYTES/WORD
54 02214 006300         ASL     R0              ; 2 WORDS/ENTRY
55 02216 016202         MOV     2(R2),R2
                000002
56 02222 060002         ADD     R0,R2
57 02224 000403         BR      GETAD0          ; AND GO TO EXIT
58 02226        ERSS3:
59 02226 104400         TRAP    0
60                      .IFNDF  $LONGER
61 02230     123        .ASCII  /SOB/
   02231     117
   02232     102
62                      .ENDC
63                      .IFDF   $LONGER
64                      .ASCII  /SUBSCRIPT OUT OF BOUNDS/
65                      .ENDC
66 02233     000        .BYTE   0
67                      .EVEN
68 02234 012600 GETAD0: MOV     (SP)+,R0        ; RESTORE R0
69 02236 000207         RETURN                  ; EXIT
70 02240 000167 ADRARG: JMP     ERRARG          ; ARGUMENT ERROR
                000000G
71 02244 000167 ADRSYN: JMP     ERRSYN          ; SYNTAX ERROR
                000000G
72 02250 012203 GETAD1: MOV     (R2)+,R3        ; PUT BASE ADR INTO R3
73 02252 022323         CMP     (R3)+,(R3)+     ; POINT R3 AT A(0)
74 02254 005712         TST     (R2)            ; SS1MAX=0?
75 02256 001014         BNE     GETAD6          ; NO- BRANCH AND USE IT
76 02260 010302         MOV     R3,R2           ; POINT AT ARRAY
77 02262 022227         CMP     (R2)+,#DJMP     ; A DJMP AT END?
                160000
78 02266 001375         BNE     .-4             ; NO- LOOP 'TIL FOUND
79 02270 011202         MOV     (R2),R2         ; POINT AT ADR IN DPY FILE
80 02272 005742         TST     -(R2)           ; POINT AT SS1 IN DPY FILE
81                      .IFDF   $LPS
82 02274 002005         BGE     GETAD7
83 02276 011267         MOV     (R2),SCLSAV
                004012
84 02302 012702         MOV     #SCLSAV,R2
                006314'
85 02306 005412         NEG     (R2)
86 02310        GETAD7: .ENDC
87 02310 026512 GETAD6: CMP     SS1SAV(R5),(R2) ; SS1 > SS1MAX?
                000024
88 02314 003344         BGT     ERSS3           ; YES- ERROR
89 02316 016502         MOV     SS1SAV(R5),R2   ; GET SS1
                000024
```

GTB V01-01 BASIC - GT ROOT MOD RT-11 MACRO VM02-09 16-OCT-74 02:06:07 PAGE 15+

```
90 02322 100403         BMI     GETAD2          ; BR IF NO SUBSCRIPTS
91 02324 006302         ASL     R2              ; ADDOUNT FOR WORDS NOT BYTES
92 02326 060302         ADD     R3,R2           ; GET ADDRESS
93 02330 000741         BR      GETAD0          ; EXIT
94 02332 010302 GETAD2: MOV     R3,R2           ; IF NO SUBSCRIPTS GET JUST A
95 02334 000737         BR      GETAD0          ; EXIT
96 02336 005767 GETAD3: TST     OPARGF          ; SEE IF ARG IS OPTIONAL
                175532
97 02342 001740         BEQ     ADRSYN          ; NO- SYNTAX ERROR
98 02344 120227 GETAD5: CMPB    R2,#,RPAR       ; IS IT A ")" ?
                000000G
99 02350 001335         BNE     ADRSYN          ; NO- ERROR
100 2352 005301         DEC     R1              ; YES- POINT AT IT
101 2354 005002         CLR     R2              ; MAKE ADR = 0
102 2356 000726         BR      GETAD0          ; EXIT
103                     .IFDF   $DISK
```

```
1                       .SBTTL  "DSTP", "CONT", AND "INIT" ROUTINES
2                       .ENDC
3               ;       ***************************************
4
5                       ; ROUTINE TO HANDLE "DSTP", "CONT", AND "INIT"
6
7 002360        DSTP:
8 002360 012746         MOV     #CKEOL,-(SP)    ; NORMAL PROGRAM EXIT
         000324'
9 002364        STOPA:
10 02364 010667         MOV     SP,STOP.F       ; SET STOP FLAG
         000054
11 02370 005737         TST     @#GTVECT        ; VECTORS BEEN SET?
         000000G
12 02374 001420         BEQ     STOPB           ; NO- EXIT NOW
13 02376 005767         TST     DPC             ; DPC CLEARED?
         000000G
14 02402 001415         BEQ     STOPB           ; YES
15 02404 005767         TST     RUN.F           ; CURRENTLY RUNNING?
         000036
16 02410 001414         BEQ     STOPB+4         ; NO
17 02412 012767         MOV     #SDINT,DSTOP+2  ; LOAD STOP
         173400
         006166
18 02420 005767 STOPC:  TST     DSR             ; STOPPED YET?
         000000G
19 02424 002375         BGE     .-4             ; NO- LOOP SOME MORE
20 02426 022767         CMP     #DJMP,DSTOP+2   ; BACK TO REGULAR?
         160000
         006152
21 02434 001371         BNE     STOPC           ; NO- WAIT SOME MORE
22 02436 005067 STOPB:  CLR     RUN.F           ; SIGNAL STOPPED
         000004
23 02442 000207         RETURN
24
25 02444 000000 STOP.F: .WORD 0                 ; STOP FLAG (=1 FOR STOP)
26 02446 000000 RUN.F:  .WORD   0               ; FLAG TO INDICATE RUNNING
27
28 02450        CONT:
29 02450 012746         MOV     #CKEOL,-(SP)    ; EXIT FOR CALLER
         000324'
30 02454        CONTA:
31 02454 005767         TST     STOP.F          ; WERE WE STOPPED?
         177764
32 02460 001403         BEQ     CONT0           ; NO- THEN DON'T START IT
33 02462 005067         CLR     STOP.F          ; CLEAR STOP FLAG
         177756
34 02466 000500         BR      INIT3           ; CLEAN SUBP STK AND START UP
35 02470 000207 CONT0:  RETURN
36
37 02472 004767 RESTR:  JSR     PC,INIT3        ; FIX STACK AND START DPU
         000172
38 02476 000002         RTI
39
40 02500        INIT:
41 02500 012746         MOV     #CKEOL,-(SP)    ; NORMAL ENTRY
         000324'
```



```
42 02504        INITA:                          ; ENTRY FROM BUFTST
43 02504 004767         JSR     PC,STOPA        ; MAKE SURE DISPLAY HAS STOPPED
         177654
44              ; ***  SET UP INTERRUPT VECTORS HERE
45 02510 012700         MOV     #GTVECT,R0      ; SET UP DESTINATION
         000000G
46 02514 062700         ADD     #12,R0          ; OFFSET TO TOP
         000012
47 02520 012702         MOV     #PR4,R2         ; AND PRIORITY
         000200
48 02524 010210         MOV     R2,(R0)         ;  PR4 FOR RESTR
49 02526 012740         MOV     #RESTR,-(R0)    ; RESTART
         002472'
50 02532 010240         MOV     R2,-(R0)        ;  PR4 FOR LP INTERRUPT
51 02534 012740         MOV     #LPNINT,-(R0)   ; LP INTERRUPT
         003240'
52 02540 010240         MOV     R2,-(R0)        ;  PR4 FOR STOP DISPLAY
53 02542 012740         MOV     #DISINT,-(R0)   ; STOP DISPLAY INTERRUPT
         000466'
54 02546 005067         CLR     OLDMOD          ; MAKE SURE MODE CHANGES
         177212
55 02552 005067         CLR     INSRTF          ; CLEAR INSERT PENDING FLAG
         176056
56 02556 005067         CLR     SCAL.F          ; CLEAR SCALE FLAG
         000130
57 02562 005067         CLR     LPEN.F          ; CLEAR LP HIT FLAG
         177112
58 02566 005067         CLR     STOP.F          ; CLEAR THE STOP FLAG
         177652
59              ; *** MAKE SURE TRAK IS OFF
60 02572 012767         MOV     #DJMP,BEG       ; MAKE SURE TRAK IS OFF
         160000
         005706
61              ; *** MAKE SURE END OF DPY FILE IS AFTER PERMANENT STUFF
62 02600 005767         TST     DCRASH          ; "FIX" BEEN CALLED?
         006006
63 02604 001004         BNE     INIT1           ; BR IF "FIX" ALREADY CALLED
64 02606        INIT2:
65 02606 010446         MOV     R4,-(SP)
66                      .IFDF   $DISK
67                      JSR     PC,CLOSALL      ; CLOSE ALL FILE I/O
68                      .ENDC
69 02610 004767         JSR     PC,FIX0         ; GET SPACE
         000000G
70 02614 012604         MOV     (SP)+,R4
71 02616 016700 INIT1:  MOV     ACTST,R0
         176020
72 02622 010067         MOV     R0,DSTOP        ; SET LO=END FOR DJMP
         005762
73 02626 010067         MOV     R0,DISEND       ; IT IS A NEW DISEND
         176004
74 02632 012720         MOV     #DJMP,(R0)+     ; PUT A DJMP THERE
         160000
75 02636 012710         MOV     #BEG,(R0)       ; FOLLOWED BY LOOP ADR
         010506'
76 02642 010067         MOV     R0,ACTEND       ; ACTEND POINTS TO WORD AFTER DISEND
         175772
```

```
77                      ; ***  FIX UP TAG POINTER
78 02646 016700          MOV     TAGHED,R0       ; GET FIRST TAG ADDRESS
         176562
79 02652 005720          TST     (R0)+           ; BUMP IT BY 2
80 02654 010067          MOV     R0,TAGEND       ; MAKE THAT TAGEND
         176556
81 02660 005010          CLR     (R0)            ; AND CLEAR LAST POINTER
82                      ; ***  PATCH UP ADSTK AND NSTSK HERE
83 02662 012767          MOV     #NSBAS,NSTSK    ; ADJUST BUILDING STACK
         001364'
         176522
84 02670 012767 INIT3:   MOV     #ADBAS,ADSTK    ; ADJUST RUNNING STACK
         001334'
         176464
85 02676 010667          MOV     SP,RUN,F        ; SIGNAL RUNNING
         177544
86 02702 012767          MOV     #BEG,DPC        ; START UP DPU
         010506'
         000000G
87 02710 000207          RETURN                  ; RETURN TO USER
88
89 02712 000000 SCAL.F:  .WORD 0                 ; SCALE HAS BEEN DONE FLAG
90
91                      ;       ROUTINE TO HANDLE LINKING FOR SAVE AND RESTORE
92                              .IFDF   $DISK
93                      SAVERT:
94                              JSR     PC,EVAL         ; GET STRING (BASICX)
95                              BCS     .+6             ; FOUND A STRING?
96                              JMP     ERRSYN          ; NO- ERROR
97                              CMP     (SP),#-1        ; A NULL STRING?
98                              BEQ     .-10            ; YES- ERROR
99                              CMPB    (R1)+,#.RPAR    ; FOLLOWED BY ")" ?
100                             BNE     .-16            ; NO- ERROR
101                             JMP     SAV24           ; RETURN TO OVERLAY
102
103                             .ENDC
104                             .IFDF   $DISK
```



```
1                               .SBTTL  "NOSC","ON", "OFF" ROUTINES
2                               .ENDC
3                       ;       ****************************************
4                               ; TURN OFF SCALE
5
6 002714                NOSC:
7 002714 005067                 CLR     SCAL.F          ; NO MORE SCALE
         177772
8 002720 000167                 JMP     CKEOL           ; AND EXIT
         175400
9
10                      ;       ****************************************
11                              ; ROUTINE TO TURN SUBP ON AND OFF
12
13 02724                ON:
14 02724 012746                 MOV     #SDINT,-(SP)    ; A SDINT
         173400
15 02730 004767 ON1:    JSR     PC,POLENT       ; ENTER POLISH MODE, CK "("
         175334
16 02734 000136'                +GETONE         ; GET SUBP TAG
17 02736 000006'                +INTEGR         ; INTEGERIZE
18 02740 002742'                .+2             ; EXIT POLISH MODE
19 02742 016502                 MOV     FAC2(R5),R2     ; GET TAG
         000042
20 02746 003002                 BGT     .+6             ; LEGITIMATE- BRANCH
21 02750 000167 ON0:    JMP     ERRARG          ; NO- ERROR
         000000G
22 02754 004767                 JSR     PC,TAGSRH       ; FIND TAG
         176434
23 02760 005700                 TST     R0              ; DID WE SUCCEED?
24 02762 001772                 BEQ     ON0             ; NO- BRANCH
25 02764 162700                 SUB     #10,R0          ; POINT AT SDINT/DJMP
         000010
26 02770 005767                 TST     INSRTF          ; AN INSERT PENDING?
         175640
27 02774 001375                 BNE     .-4             ; YES- LOOP TIL FREE
28 02776 016703                 MOV     ACTEND,R3       ; GET END OF DPY FILE
         175636
29 03002 005743                 TST     -(R3)           ; POINT TO PREVIOUS WORD
30 03004 020003                 CMP     R0,R3           ; WHERE DOES R0 POINT?
31 03006 001403                 BEQ     ON2             ; TO WD1
32 03010 012610                 MOV     (SP)+,@R0       ; STORE WORD
33 03012 000167 ON3:    JMP     CKRPAR          ; AND RETURN TO USER
         175300
34 03016 012667 ON2:    MOV     (SP)+,WD1       ; STORE WORD
         175716
35 03022 000773                 BR      ON3             ; AND EXIT
36
37 03024                OFF:
38 03024 012746                 MOV     #DJMP+1,-(SP)   ; A DJMP+1
         160001
39 03030 000737                 BR      ON1             ; AND CONTINUE
40                              .IFDF   $DISK
```

```
                        .SBTTL  "FIX" ROUTINE CALLER
                        .ENDC
                ;       ****************************************

                        ; ROUTINE TO HANDLE:
                        ;       CALL "FIX"(N)

003032          FIX:
003032 004767           JSR     PC,POLENT       ; ENTER POLISH MODE, CK "("
       175232
03036 000136'           +GETONE                 ; GET N
03040 000006'           +INTEGR                 ; INTEGERIZE IT
03042 003044'           ,+2                     ; EXIT POLISH MODE
                        .IFDF   $DISK
                        JSR     PC,CLOSALL      ; CLOSE ALL FILE I/O
                        .ENDC
03044 000167            JMP     FIXSP           ; JUMP TO OVERLAY
       000000G
                        .IFDF   $DISK
                FREE:
                        JSR     PC,CLOSALL      ; CLOSE ALL FILE I/O
                        JMP     FREE@           ; GO TO OVERLAY
```



```
                        .SBTTL  ROUTINE TO FETCH L, I, F, T
                        .ENDC
                ;       ****************************************

                        ; ROUTINE TO HANDLE L, I, F, T
                        ; UPON ENTRY R1 POINTS TO COMMA PRECEDING L
                        ; UPON EXIT R0 CONTAINS BITS 0:10 OF SGM WORD

                        ; L = LP INTERRUPT ENABLE
                        ; I = INTENSITY OF LINE
                        ; F = FLASH (BLINK)
                        ; T = TYPE OF LINE (SOLID, ETC.)

03050           LIFT:
03050 012767            MOV     #40000,INVIS    ; ASSUME VISIBLE
       040000
       000160
03056 010446            MOV     R4,-(SP)        ; SAVE R4 FOR RETURN
03060 004767            JSR     PC,QIKPOL       ; ENTER POLISH MODE
       175226
03064 000066'           +SETOPT                 ; SET OPTIONAL MODE
03066 000026'           +PLOOP                  ; POLISH LOOP
03070 000004            +4                      ; 4 TIMES
03072 000076'           +GETNUM                 ; GET ",L,I,F,T"
03074 000006'           +INTEGR                 ; INTEGERIZE EACH IN TURN
03076 000000'           +SAVINT                 ; SAVE ON THE STACK
03100 000040'           +ELOOP                  ; END OF POLISH LOOP
03102 003104'           ,+2                     ; EXIT FROM POLISH MODE
03104 005000            CLR     R0              ; CLEAR DESTINATION WORD
03106 012602            MOV     (SP)+,R2        ; GET TYPE OF LINE
03110 003406            BLE     LIFT0           ; BRANCH IF NO CHANGE
03112 020227            CMP     R2,#4           ; COMPARE R2 TO 4
       000004
03116 003003            BGT     LIFT0           ; IGNORE IF T>4
03120 062702            ADD     #3,R2           ; ADD TO GET SET-BIT
       000003
03124 050200            BIS     R2,R0           ; SET T INTO R0
03126 012602 LIFT0:     MOV     (SP)+,R2        ; GET F
03130 004767            JSR     PC,BITGET       ; F=0: 0
       000060
03134 050300            BIS     R3,R0           ; F>0: 30, F<0: 20
03136 012602            MOV     (SP)+,R2        ; GET I
03140 001415            BEQ     LIFT1           ; BRANCH IF NO CHANGE
03142 100003            BPL     LIFT2           ; BRANCH IF VISIBLE
03144 005067            CLR     INVIS           ; MAKE INVISIBLE
       000066
03150 005402            NEG     R2              ; MAKE R2 POSITIVE
03152 005302 LIFT2:     DEC     R2
03154 020227            CMP     R2,#10          ; COMPARE R2 WITH 8
       000010
03160 002005            BGE     LIFT1           ; IGNORE IF TOO BIG
03162 052702            BIS     #10,R2          ; ADD IN 8
       000010
03166 000302            SWAB    R2
03170 006202            ASR     R2              ; PUT BITS IN CORRECT POSITION
03172 050200            BIS     R2,R0           ; I IS SET INTO R0
03174 012602 LIFT1:     MOV     (SP)+,R2        ; GET L
```

```
49 03176 004767          JSR     PC,BITGET       ; L=0: 0
         000012
50 03202 006303          ASL     R3              ; L>0: 140
51 03204 006303          ASL     R3              ; L<0: 100
52 03206 050300          BIS     R3,R0           ; L IS SET IN R0
53 03210 012604          MOV     (SP)+,R4        ; RESORE R4
54 03212 000134          POLRET                  ; AND EXIT
55
56 03214         BITGET:
57 03214 005003          CLR     R3              ; CLEAR DESTINATION
58 03216 005702          TST     R2              ; CHECK SIGN OF R2
59 03220 001405          BEQ     BITZER          ; BRANCH IF R2 = 0
60 03222 002402          BLT     BITNEG          ; BRANCH IF R2 < 0
61 03224 052703          BIS     #10,R3          ; POS: 30
         000010
62 03230 052703 BITNEG:  BIS     #20,R3          ; NEG: 20
         000020
63 03234 000207 BITZER:  RETURN                  ; ZERO: 0
64
65 03236 000000 INVIS:   .WORD 0                 ; VISIBILITY FLAG (BIT 14)
66                       .IFDF  $DISK
```



```
1                        .SBTTL  LP INTERRUPT HANDLER
2                        .ENDC
3                ;       ****************************************
4
5                        ; ROUTINE TO HANDLE LIGHT PEN INTERRUPT
6
7 003240         LPNINT:
8 003240 010046          MOV     R0,-(SP)
9 003242 010246          MOV     R2,-(SP)        ; SAVE REGISTERS
10 03244 010546          MOV     R5,-(SP)        ; SAVE R5
11 03246 016705          MOV     USRAREA,R5      ; SET UP R5 TO USER AREA
         000000G
12 03252 016746          MOV     $ERVEC,-(SP)    ; SAVE ERROR VECTOR IN FPMP
         000000G
13 03256 017700          MOV     @ADSTK,R0       ; GET PTR TO TAG (IF ANY)
         176100
14 03262 001403          BEQ     LPINT0          ; NO TAG- BRANCH
15 03264 005720          TST     (R0)+           ; POINT R0 AT TAG
16 03266 011000          MOV     (R0),R0         ; PUT TAG INTO R0
17 03270 002407          BLT     LPINT1          ; HIT MUST BE IN TRAK
18 03272 005767 LPINT0:  TST     LPEN,F          ; ANOTHER HIT PENDING?
         176402
19 03276 001150          BNE     LPEX1           ; YES- EXIT
20 03300 005367          DEC     LPEN,F          ; NO- MAKE FLAG = -1
         176374
21 03304 010067          MOV     R0,LPENT        ; STORE TAG
         176356
22 03310 016546 LPINT1:  MOV     FAC2(R5),-(SP)  ; SAVE FAC
         000042
23 03314 016546          MOV     FAC1(R5),-(SP)
         000040
24 03320 016746          MOV     ABS,F,-(SP)     ; AND FLAG TOO
         000314
25 03324 005065          CLR     FAC1(R5)
         000040
26 03330 016700          MOV     DISX,R0         ; GET X POSITION
         000000G
27 03334 042700          BIC     #176000,R0      ; ONLY FIRST 10 BITS
         176000
28 03340 010065          MOV     R0,FAC2(R5)     ; PUT IN FAC
         000042
29 03344 005767          TST     LPEN,F          ; IN TRAK?
         176330
30 03350 002420          BLT     LPINT2          ; NO- BRANCH
31 03352 166700          SUB     XT,R0
         005146
32 03356 006200          ASR     R0              ; DIV BY 2
33 03360 006200          ASR     R0              ; DIV BY 2
34 03362 060067          ADD     R0,XT           ; OLD + (NEW-OLD)/4
         005136
35 03366 016765          MOV     XT,FAC2(R5)     ; PUT IN SMOOTHED VALUE
         005132
         000042
36 03374 032767          BIT     #40,DSR         ; ON THE EDGE?
         000040
         000000G
37 03402 001403          BEQ     LPINT2          ; NO
```

```
38 03404 012767          MOV     #1000,XT        ; STORE CENTER IN XT
         001000
         005112
39 03412 010667 LPINT2:  MOV     SP,ABS,F        ; SCALE ABSOLUTE
         000222
40 03416 012700          MOV     #USCL,X,R0      ; UNSCALE MODE
         006362'
41 03422 004767          JSR     PC,SC,USC       ; DO IT NOW
         000672
42 03426 016546          MOV     FAC2(R5),-(SP)
         000042
43 03432 016546          MOV     FAC1(R5),-(SP)  ; SAVE UNSCALED X ON STACK
         000040
44 03436 005065          CLR     FAC1(R5)
         000040
45 03442 016700          MOV     DISY,R0         ; GET Y POSITION
         000000G
46 03446 042700          BIC     #176000,R0      ; ONLY FIRST 10 BITS
         176000
47 03452 010065          MOV     R0,FAC2(R5)     ; PUT INTO FAC
         000042
48 03456 005767          TST     LPEN,F          ; IN TRAK?
         176216
49 03462 002420          BLT     LPINT3          ; NO= BRANCH
50 03464 166700          SUB     YT,R0
         005036
51 03470 006200          ASR     R0              ; DIV BY 2
52 03472 006200          ASR     R0              ; DIV BY 2
53 03474 060067          ADD     R0,YT           ; OLD + (NEW-OLD)/4
         005026
54 03500 016765          MOV     YT,FAC2(R5)     ; PUT IN SMOOTHED VALUE
         005022
         000042
55 03506 032767          BIT     #40,DSR         ; ON THE EDGE?
         000040
         000000G
56 03514 001403          BEQ     LPINT3          ; NO
57 03516 012767          MOV     #600,YT         ; STORE CENTER IN YT
         000600
         005002
58 03524 012700 LPINT3:  MOV     #USCL,Y,R0      ; UNSCALE Y MODE
         006352'
59 03530 004767          JSR     PC,SC,USC       ; DO IT NOW
         000564
60 03534 005767          TST     LPEN,F          ; IN TRAK?
         176140
61 03540 002405          BLT     LPINT4          ; NO= BRANCH
62 03542 016700          MOV     TRAKX,R0        ; FROM STK
         001216
63 03546 016702          MOV     TRAKY,R2        ; FROM FAC
         001214
64 03552 000406          BR      LPINT5          ; STORE IT
65 03554 005467 LPINT4:  NEG     LPEN,F          ; MAKE FLAG = 1
         176120
66 03560 012700          MOV     #LPENX,R0       ; FROM STK
         001670'
67 03564 012702          MOV     #LPENY,R2       ; FROM FAC
```



```
         001674'
68 03570 012620 LPINT5:  MOV     (SP)+,(R0)+
69 03572 012610          MOV     (SP)+,(R0)      ; STORE X
70 03574 016522          MOV     FAC1(R5),(R2)+
         000040
71 03600 016512          MOV     FAC2(R5),(R2)   ; STORE Y
         000042
72 03604 012667          MOV     (SP)+,ABS,F     ; RESTORE FLAG
         000030
73 03610 012665          MOV     (SP)+,FAC1(R5)
         000040
74 03614 012665          MOV     (SP)+,FAC2(R5)  ; RESTORE FAC
         000042
75 03620 012667 LPEX1:   MOV     (SP)+,$ERVEC    ; RESTORE $ERVEC
         000000G
76 03624 012605          MOV     (SP)+,R5        ; RESTORE R5
77 03626 012602          MOV     (SP)+,R2
78 03630 012600          MOV     (SP)+,R0        ; RESTORE REGISTERS
79 03632 005267          INC     DPC             ; RESTART DPU
         000000G
80 03636 000002          RTI                     ; AND EXIT
81
82 03640 000000 ABS,F:   .WORD 0                 ; ABSOLUTE SCALE FLAG
83                       .IFDF  $DISK
84                       .EOT
```

```
                ;
                ;       DEC - 11 - LBPGB - A - LA    BASIC KERNEL V02-01
                ;
                ;       GTB.MAC TAPE 3 OF 4
                ;
                ; THE INFORMATION IN THIS LISTING IS SUBJECT TO CHANGE WITHOUT NOTICE
                ; AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
                ; CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY
                ; FOR ANY ERRORS THAT MAY APPEAR IN THIS LISTING.
                ;
                ; THIS SOFTWARE IS FURNISHED TO THE PURCHASER UNDER A LICENSE
                ; FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH
                ; INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN
                ; SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY
                ; DIGITAL.
                ;
                ; DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE
                ; USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT
                ; SUPPLIED BY DIGITAL.
                ;
                ; COPYRIGHT (C) 1973,1974, BY DIGITAL EQUIPMENT CORPORATION.
                ;
                ;       AUTHOR: DR. STEPHEN R. ALPERT   AUGUST 1973
                        .SBTTL  "STAT" ROUTINE
                        .ENDC
                ;       ***************************************

                        ; ROUTINE TO HANDLE
                        ;       CALL "STAT"( FONT [, INT] )

                        ; FONT >0: NORMAL, =0: NO CHANGE, <0: ITALICS
                        ; INT >0: LP INTENSIFIED, =0: NO CHANGE, <0: HIT NOT INTEN.

03642           STAT:
03642 004767    174336  JSR     PC,BUFPOL       ; ENTER POLISH MODE AND CK ")"
03646 000136'           +GETONE                 ; GET FONT
03650 000006'           +INTEGR                 ; MAKE SURE AN INTEGER
03652 000000'           +SAVINT                 ; SAVE ON STACK
03654 000066'           +SETOPT                 ; INT IS OPTIONAL
03656 000076'           +GETNUM                 ; CK "," AND GET INT
03660 000006'           +INTEGR                 ; MAKE SURE AN INTEGER
03662 003664'           .+2                     ; EXIT FROM POLISH
03664 016502    000042  MOV     FAC2(R5),R2     ; GET INT
03670 005402            NEG     R2              ; NEGATE SO BITGET RETURNS VALUES
03672 004767    177316  JSR     PC,BITGET       ; POS: 20,NEG: 30, ZERO: 0
03676 006303            ASL     R3
03700 006303            ASL     R3
03702 006303            ASL     R3              ; POS: 200, NEG: 300, ZERO: 0
03704 010300            MOV     R3,R0           ; SAVE IN R0
03706 012602            MOV     (SP)+,R2        ; GET FONT
03710 005402            NEG     R2              ; NEGATE SO BITGET RETURNS VALUES
03712 004767    177276  JSR     PC,BITGET
03716 006303            ASL     R3              ; POS: 40,NEG: 60, ZERO: 0
```



```
03720 050300            BIS     R3,R0           ; SET FONT INTO R0
03722 052700    170000  BIS     #170000,R0      ; SET UP LOAD STAT, REG, A
03726 000167    000246  JMP     $DXIT           ; INSERT TO DPY/ CK ")"/ RETURN
                        .IFDF   $DISK
```

```
                        .SBTTL  "RDOT" ROUTINE
                        .ENDC
                ;       ****************************************

                        ; ROUTINE TO HANDLE
                        ;       CALL "RDOT"( DX, DY [,L,I,F,T])

003732          RDOT:
003732 004767           JSR     PC,BUFTST       ; ANY BUFFER?
       174252
03736  005067           CLR     ABS,F           ; RELATIVE SCALE
       177676
03742  004767           JSR     PC,ARGGET       ; GET ARGUMENTS
       176020
03746  005767           TST     XYBIG           ; RDOT OR VECT?
       176056
03752  001424           BEQ     RDOT1           ; RDOT= BRANCH
03754  052700           BIS     #110000,R0      ; LONG VECTOR
       110000
03760  004767           JSR     PC,PUTSGM       ; SET=GRAPHIC=MODE
       175760
03764  016600           MOV     2(SP),R0        ; GET X
       000002
03770  056700           BIS     XSIGN,R0        ; SET SIGN (INVISIBLE)
       176036
03774  004767           JSR     PC,PUTWD        ; INSERT IT
       174650
04000  016700           MOV     YSIGN,R0        ; GET SIGN OF Y
       176030
04004  000300           SWAB    R0
04006  006200           ASR     R0              ; PUT IN BIT 13
04010  062600           ADD     (SP)+,R0        ; ADD IN Y
04012  004767           JSR     PC,PUTWD        ; INSERT IT
       174632
04016  005016           CLR     (SP)            ; MAKE X = 0
04020  005046           CLR     -(SP)           ; MAKE Y = 0
04022  005000           CLR     R0              ; CLEAN BITS 10 - 0
04024  052700 RDOT1:    BIS     #130000,R0      ; RDOT MODE
       130000
04030  004767           JSR     PC,PUTSGM       ; SET=GRAPHIC=MODE
       175710
04034  005716           TST     (SP)            ; Y = 0?
04036  001013           BNE     RDOT0           ; NO= BRANCH
04040  005766           TST     2(SP)           ; YES= X = 0?
       000002
04044  001010           BNE     RDOT0           ; NO= BRANCH
04046  005067           CLR     YSIGN           ; MAKE Y POSITIVE
       175762
04052  012716           MOV     #1,(SP)         ; MAKE Y = +1
       000001
04056  012700           MOV     #101,R0         ; MAKE (0,-1) RDOT
       000101
04062  004767           JSR     PC,PUTWD        ; INSERT IT
       174562
04066  016600 RDOT0:    MOV     2(SP),R0        ; GET X
       000002
04072  000300           SWAB    R0              ; SHIFT TO LEFT
```



```
04074  006200           ASR     R0              ;   MOVE TO BITS 7-12
04076  056700           BIS     INVIS,R0        ; SET VISIBILITY
       177134
04102  056700           BIS     XSIGN,R0        ; SET SIGN OF X
       175724
04106  062600           ADD     (SP)+,R0        ; ADD IN Y
04110  005726           TST     (SP)+           ; CLEAN STACK
04112  056700           BIS     YSIGN,R0        ; SET SIGN OF Y
       175716
04116  000430           BR      $DXIT           ; TAKE "STANDARD" EXIT
                        .IFDF   $DISK
```

```
1                       .SBTTL  "APNT" ROUTINE
2                       .ENDC
3                ;      ****************************************
4
5                       ; ROUTINE TO HANDLE
6                       ;       CALL "APNT" ( X, Y [,L,I,F,T] )
7
8 004120        APNT:
9 004120 004767         JSR     PC,BUFTST       ; ANY BUFFER?
         174064
10 004124 010667        MOV     SP,ABS,F        ; FLAG ABSOLUTE MODE
         177510
11 004130 004767        JSR     PC,ARGGET       ; GET ARGUMENTS
         175632
12 004134 052700        BIS     #114000,R0      ; SET UP MODE WORD
         114000
13 004140 004767        JSR     PC,PUTSGM       ; SET-GRAPHIC-MODE
         175600
14 004144 016600        MOV     2(SP),R0        ; PUT X IN R0
         000002
15 004150 004767        JSR     PC,ENTERP       ; IMMED POLISH
         174002
16 004154 004514'               +POSX           ; MAKE X >=0
17 004156 056700        BIS     INVIS,R0        ; SET VISIBILITY
         177054
18 004162 004767        JSR     PC,PUTWD        ; INSERT X
         174462
19 004166 012600        MOV     (SP)+,R0        ; PUT Y INTO R0
20 004170 005726        TST     (SP)+           ; CLEAN STACK
21 004172 004767        JSR     PC,ENTERP       ; IMMED POLISH
         173760
22 004176 004522'               +POSY           ; MAKE Y >=0
23 004200 004767 $DXIT: JSR     PC,PUTWD        ; INSERT IT
         174444
24 004204 004767 $DXIT0: JSR    PC,INSERT       ; FLAG INSERT READY
         174534
25 004210 000167        JMP     CKRPAR          ; CK ")" AND EXIT TO USER
         174102
26                      .IFDF   $DISK
```



```
1                       .SBTTL  "VECT" ROUTINE
2                       .ENDC
3                ;      ****************************************
4
5                       ; ROUTINE TO HANDLE
6                       ;       CALL "VECT" (DX, DY [,L,I,F,T] )
7
8 004214        VECT:
9 004214 004767         JSR     PC,BUFTST       ; ANY BUFFER?
         173770
10 004220 005067        CLR     ABS,F           ; RELATIVE SCALE MODE
         177414
11 004224 004767        JSR     PC,ARGGET       ; GET ARGUMENTS
         175536
12 004230 005767        TST     XYBIG           ; SHORT OR LONG MODE?
         175574
13 004234 001006        BNE     VECT0           ; LONG MODE= BRANCH
14 004236 052700        BIS     #104000,R0      ; SET SHORT VECTOR MODE
         104000
15 004242 004767        JSR     PC,PUTSGM       ; SET-GRAPHIC-MODE
         175476
16 004246 000167        JMP     RDOT0           ; FINISH UP IN RDOT ROUTINE
         177614
17 004252 052700 VECT0: BIS     #110000,R0      ; SET LONG VECTOR MODE
         110000
18 004256 004767        JSR     PC,PUTSGM       ; SET-GRAPHIC-MODE
         175462
19 004262 016600        MOV     2(SP),R0        ; GET X
         000002
20 004266 056700        BIS     INVIS,R0        ; FLAG VISIBILITY
         176744
21 004272 056700        BIS     XSIGN,R0        ; FLAG SIGN
         175534
22 004276 004767        JSR     PC,PUTWD        ; INSERT X
         174346
23 004302 016700        MOV     YSIGN,R0        ; GET SIGN OF Y
         175526
24 004306 000300        SWAB    R0
25 004310 006200        ASR     R0              ; MOVE BIT 6 TO BIT 13
26 004312 062600        ADD     (SP)+,R0        ; GET VALUE OF Y
27 004314 005726        TST     (SP)+           ; FIX STACK
28 004316 000730        BR      $DXIT           ; EXIT THROUGH APNT ROUTINE
29                      .IFDF   $DISK
```

```
1                       .SBTTL  GENERAL SCALE-UNSCALE ROUTINE
2                       .ENDC
3               ;       ****************************************
4
5                       ; ROUTINE TO DO SCALING AND UNSCALING
6                       ; VALUE TO BE SCALED/UNSCALED AND RESULT ARE IN FAC
7                       ; BE CAREFUL ON REGISTERS SAVED ETC.
8               ; THIS ROUTINE IS RAISED TO LEVEL 4 SO A LPEN INTERRUPT
9               ; WON'T CAUSE A STACK OVERFLOW
10
11
12 04320        SC.USC: 
13 04320 005767         TST     SCAL.F          ; IS THERE ANY SCALING?
         176366
14 04324 001442         BEQ     SCUSC1          ; NO- EXIT
15 04326 016746         MOV     PSW,-(SP)       ; SAVE PSW
         177776'
16 04332 052767         BIS     #200,PSW        ; RAISE TO PR4
         000200
         177776'
17 04340 010146         MOV     R1,-(SP)        ; SAVE R1
18 04342 010246         MOV     R2,-(SP)        ; SAVE R2
19 04344 010346         MOV     R3,-(SP)        ; SAVE R3
20 04346 010446         MOV     R4,-(SP)        ; SAVE R4
21 04350 010546         MOV     R5,-(SP)        ; SAVE R5
22 04352 012702         MOV     #4,R2           ; SET UP COUNT FOR WORD TRANSFER
         000004
23 04356 005720         TST     (R0)+           ; ADJUST POINTER
24 04360 014046 SCUSC0: MOV     -(R0),-(SP)     ; TRANSFER: HI-ORDER=>LO-ADDRESS
25 04362 005302         DEC     R2              ; DECREMENT LOOP COUNT
26 04364 001375         BNE     SCUSC0          ; BRANCH IF NOT DONE
27 04366 012767         MOV     #ERSTAR,SERVEC  ; ERROR VECTOR FOR MULT.
         004434'
         000000G
28 04374 004767         JSR     PC,QIKPOL       ; ENTER POLISH MODE
         173712
29 04400 004574'                +PUSHF          ; FLOAT FAC TO STACK
30 04402 000000G                +$MLR           ; MULTIPLY
31 04404 004456'                +RELABS         ; RELATIVE OR ABSOLUTE SCALE?
32 04406 000000G                +$ADR           ; DO ADDITION IF ABSOLUTE
33 04410 000256'                +POP            ; PUT RESULT IN FAC
34 04412 004414'                .+2             ; EXIT POLISH
35 04414 012605         MOV     (SP)+,R5        ; GET BACK R5
36 04416 012604         MOV     (SP)+,R4
37 04420 012603         MOV     (SP)+,R3
38 04422 012602         MOV     (SP)+,R2        ; RESTORE REGISTERS
39 04424 012601         MOV     (SP)+,R1
40 04426 012667         MOV     (SP)+,PSW       ; RESTORE PSW
         177776'
41 04432 000207 SCUSC1: RETURN
42
43              ;       *** GOES QUITE DEEP IN STACK  --12 WORDS?
44
45 04434        ERSTAR: 
46 04434 104400         TRAP    0
47 04436    117         .ASCII  /OVF/
   04437    126
```



```
   04440    106
48 04441    000         .BYTE   0
49                      .EVEN
50 04442 020027 ERRDIV: CMP     R0,#4000
         004000
51 04446 103772         BLO     ERSTAR
52 04450 104400         TRAP    0
53 04452    104         .ASCII  /DV0/
   04453    126
   04454    060
54 04455    000         .BYTE   0
55                      .EVEN
56
57                      .IFDF   $DISK
```

```
1                       .SBTTL  MISCELLANEOUS ROUTINES
2                       .ENDC
3               ;       ****************************************
4
5                       ; SOME AUXILIARY ROUTINES NEEDED
6
7  004456 005767 177156 RELABS: TST     ABS,F           ; RELATIVE OR ABSOLUTE SCALE?
8  004462 001401        RELAB0: BEQ     RELAB0          ; BRANCH IF RELATIVE
9  004464 000134                POLRET                  ; DO ADDITION STEP IF ABSOLUTE
10 004466 012665 000040 RELAB0: MOV     (SP)+,FAC1(R5)
11 004472 012665 000042         MOV     (SP)+,FAC2(R5)  ; STORE RESULT IN FAC
12 004476 022626                CMP     (SP)+,(SP)+     ; CLEAR STACK OF ADD CONSTANT
13 004500 000164 000006         JMP     6(R4)           ; EXIT- PAST ADD AND POP
14
15 004504 012767 004442' 000000G DIVVEC: MOV    #ERRDIV,SERVEC  ; ERROR VECTOR FOR DIVISION
16 004512 000134                POLRET
17
18              ;       ****************************************
19
20                      ; SOME POLISH ROUTINES (AGAIN)
21
22 004514               POSX:
23 004514 005767 175312         TST     XSIGN           ; CK SIGN OF X
24 004520 000402                BR      POS0            ; BR WITH CONDX CODES SET
25 004522               POSY:
26 004522 005767 175306         TST     YSIGN           ; CK SIGN OF Y
27 004526 001401 POS0:  BEQ     POS1            ; BR IF R0 >=0
28                      .IFDF   $CRASH
29                      JMP     ERRAOR          ; ARGUMENT OUT OF RANGE
30                      .ENDC
31                      .IFNDF  $CRASH
32 004530 005000        CLR     R0              ; MAKE R0 = 0
33                      .ENDC
34 004532 000134 POS1:  POLRET
35
36 004534               ERRAOR:
37 004534 104400        TRAP    0
38                      .IFNDF  $LONGER
39 004536     101       .ASCII  /AOR/
   004537     117
   004540     122
40                      .ENDC
41                      .IFDF   $LONGER
42                      .ASCII  /ARGUMENT OUT OF RANGE/
43                      .ENDC
44 004541     000       .BYTE   0
45                      .EVEN
46
47 004542 010146 SAVER1: MOV     R1,-(SP)        ; SAVE ONLY R1 ON STACK
```



```
48 004544 000134                POLRET
49
50 004546 012602 REVRSE: MOV     (SP)+,R2
51 004550 012603                MOV     (SP)+,R3
52 004552 016646 000002         MOV     2(SP),-(SP)
53 004556 016646 000002         MOV     2(SP),-(SP)
54 004562 010366 000006         MOV     R3,6(SP)
55 004566 010266 000004         MOV     R2,4(SP)
56 004572 000134                POLRET
57
58 004574 016546 PUSHF: MOV     FAC2(R5),-(SP)
   000042
59 004600 016546 000040         MOV     FAC1(R5),-(SP)
60 004604 001015                BNE     PSHXIT          ; EXIT IF REAL ALREADY
61 004606 005726                TST     (SP)+
62 004610 010167 000026         MOV     R1,M1
63 004614 010467 000024         MOV     R4,M2
64 004620 004767 173466         JSR     PC,QIKPOL
65 004624 000000G               +$IR
66 004626 004630'               .+2
67 004630 016701 000026         MOV     M1,R1
68 004634 016704 000004         MOV     M2,R4
69 004640 000134 PSHXIT: POLRET
70
71 004642 000000 M1:    .WORD 0
72 004644 000000 M2:    .WORD 0
73                      .IFDF   $DISK
```

```
1                       ,SBTTL  "TRAK" ROUTINE
2                       ,ENDC
3                ;      ****************************************
4
5                       ; ROUTINE TO HANDLE
6                       ;       CALL "TRAK" (X,Y)
7                       ;  BECAUSE OF "GETADR", X AND Y MUST BE
8                       ;       VARIABLES, NOT CONSTANTS!
9
10 04646        TRAK:
11 04646 004767         JSR     PC,BUFTST       ; ANY BUFFER?
         173336
12 04652 010667         MOV     SP,ABS.F        ; FLAG ABSOLUTE MODE
         176762
13 04656 004767         JSR     PC,POLENT       ; ENTER POLISH MODE, CK "("
         173406
14 04662 004542'                +SAVER1         ; SAVE PTR TO X
15 04664 000136'                +GETONE         ; GET X
16 04666 004770'                +SCALEX         ; SCALE IT
17 04670 004514'                +POSX           ; MAKE SURE >=0
18 04672 001452'                +SAVER0         ; SAVE X ON STACK
19 04674 000076'                +GETNUM         ; GET Y
20 04676 005042'                +SCALEY         ; SCALE IT
21 04700 004522'                +POSY           ; MAKE SURE >=0
22 04702 004704'                .+2             ; EXIT POLISH WITH Y IN R0
23 04704 012767         MOV     #DJMP,BEG       ; TURN OFF TRAK IF IT WAS ON
         160000
         003574
24 04712 010067         MOV     R0,YT           ; STORE Y IN TRAK
         003610
25 04716 012667         MOV     (SP)+,XT        ; STORE X IN TRAK
         003602
26 04722 012601         MOV     (SP)+,R1        ; RESET TOKEN PTR TO X
27 04724 004767         JSR     PC,GETADR       ; GET ITS ADDRESS IN R2
         175106
28 04730 010267         MOV     R2,TRAKX        ; STORE THE ADDRESS
         000030
29 04734 004767         JSR     PC,ENTERP       ; IMMED POLISH
         173216
30 04740 001702'                +CKOPCM         ; CK COMMA
31 04742 004767         JSR     PC,GETADR       ; GET ADDRESS OF Y
         175070
32 04746 010267         MOV     R2,TRAKY        ; STORE ADDRESS OF Y
         000014
33                      ; *** START TRAK AND EXIT
34 04752 012767         MOV     #SDINT,BEG      ; START TRAK
         173400
         003526
35 04760 000167         JMP     CKRPAR          ; CK ")" AND EXIT TO USER
         173332
36
37 04764 000000 TRAKX:  .WORD 0                 ; ADDRESS OF CURRENT X VARIABLE
38 04766 000000 TRAKY:  .WORD 0                 ; ADDRESS OF CURRENT Y VARIABLE
39                      .IFDF   $DISK
```

67


```
1                       ,SBTTL  POLISH SCALEX AND SCALEY
2                       ,ENDC
3                ;      ****************************************
4
5                       ; ROUTINES (POLISH) TO SCALE X OR Y
6                       ; SCALES FAC, SETS SIGN BIT, MAKE VALUE POS., CK IF > 1023
7
8 004770        SCALEX:
9 004770 005067         CLR     XSIGN           ; ASSUME X >= 0
         175036
10 04774 012700         MOV     #SCL.X,R0       ; SET UP FOR SCALE X
         006402'
11 05000 004767         JSR     PC,SC.USC       ; SCALE AS REQUIRED
         177314
12 05004 004767         JSR     PC,INT          ; INTEGERIZE
         000000G
13 05010 005765         TST     FAC1(R5)        ; DID WE SUCCEED?
         000040
14 05014 001402         BEQ     XSCL0           ; YES- BRANCH
15 05016 000167 XSCL2:  JMP     ERRAOR          ; NO- CRASH
         177512
16 05022 016500 XSCL0:  MOV     FAC2(R5),R0     ; MOVE RESULT TO R0
         000042
17 05026 002030         BGE     YSCL1           ; BRANCH IF >=0
18 05030 012767         MOV     #20000,XSIGN    ; MAKE BIT 13 ON IF NEGATIVE
         020000
         174774
19 05036 005400         NEG     R0              ; MAKE R0 > 0
20 05040 000423 XSCL1:  BR      YSCL1           ; CHECK R0 < 1024
21 05042        SCALEY:
22 05042 005067         CLR     YSIGN           ; ASSUME Y >= 0
         174766
23 05046 012700         MOV     #SCL.Y,R0       ; SET UP FOR SCALE Y
         006372'
24 05052 004767         JSR     PC,SC.USC       ; SCALE IT
         177242
25 05056 004767         JSR     PC,INT          ; INTEGERIZE IT < 32767?
         000000G
26 05062 005765         TST     FAC1(R5)        ;
         000040
27 05066 001401         BEQ     YSCL0           ; YES- BRANCH
28 05070 000752         BR      XSCL2           ; NO- CRASH
29 05072 016500 YSCL0:  MOV     FAC2(R5),R0     ; MOVE RESULT TO R0
         000042
30 05076 002004         BGE     YSCL1           ; BRANCH IF >=0
31 05100 012767         MOV     #100,YSIGN      ; SET BIT 6 IF NEGATIVE
         000100
         174726
32 05106 005400         NEG     R0              ; MAKE R0 > 0
33 05110 020027 YSCL1:  CMP     R0,#2000        ; COMPARE WITH 1024
         002000
34 05114 002402         BLT     YSCL2           ; BRANCH IF LESS THAN
35                      .IFDF   $CRASH
36                      BR      XSCL2           ; CRASH
37                      .ENDC
38                      .IFNDF  $CRASH
39 05116 012700         MOV     #1777,R0        ; MAKE = 1023
```

75

```
        001777
40                      .ENDC
41 05122 000134 YSCL2:  POLRET
42                      .IFDF   $DISK
```



```
1                       .SBTTL  "XGRA" AND "YGRA" ROUTINES
2                       .ENDC
3               ;       ****************************************
4
5                       ; ROUTINE TO HANDLE
6                       ;       CALL "XGRA" (DY, A [,L,I,F,T] )
7                       ;       CALL "YGRA" (DX, A [,L,I,F,T] )
8
9                       ; ARRAY A MUST HAVE AT LEAST 2 ENTRIES
10                      ; IF AN ARRAY IS USED WITH XGRA OR YGRA, THEN
11                      ; THE SS2 ENTRY IN SYMTAB BECOMES -2, -3 RESP,
12                      ; **NOTE THAT THIS MEANS THAT
13                      ; 1) ONLY DIMENSIONED ARRAYS MAY BE USED WITH
14                      ;    THESE ROUTINES, AND
15                      ; 2) THEY MUST HAVE ONLY ONE SUBSCRIPT
16                      ; ** AT PRESENT, SINCE THE PLOTTED DATA IS ABSOLUTE,
17                      ;    AN ARRAY MAY BE USED IN ONLY ONE CALL
18                      ;       (THIS CAN BE CHANGED WITHOUT TOO MUCH TROUBLE)
19
20 05124        XGRA:
21 05124 012767         MOV     #120000,MODE    ; MODE FOR GRAPH-X
         120000
         000542
22 05132 012767         MOV     #-2,SS2TMP      ; NEW VALUE OF SS2
         177776
         000536
23 05140 000406         BR      GRA,0           ; BRANCH AND CONTINUE
24 05142        YGRA:
25 05142 012767         MOV     #124000,MODE    ; MODE FOR GRAPH-Y
         124000
         000524
26 05150 012767         MOV     #-3,SS2TMP      ; NEW SS2 FOR THIS MODE
         177775
         000520
27 05156 004767 GRA,0:  JSR     PC,BUFTST       ; ANY BUFFER?
         173026
28 05162 005067         CLR     ABS,F           ; RELATIVE SCALE
         176452
29 05166 005067         CLR     XYBIG           ; ASSUME <64
         174636
30 05172 004767         JSR     PC,POLENT       ; ENTER POLISH, CK ","
         173072
31 05176 000136'                +GETONE         ; GET DY OR DX RESP,
32 05200 005202'                ,+2             ; EXIT POLISH
33 05202 026727         CMP     SS2TMP,#-2      ; XGRA OR YGRA?
         000470
         177776
34 05210 001410         BEQ     GRA,1           ; BRANCH IF XGRA
35 05212 004767         JSR     PC,QIKPOL       ; ENTER POLISH
         173074
36 05216 004770'                +SCALEX         ; SCALE DX IN YGRA
37 05220 004514'                +POSX           ; MAKE SURE >=0
38 05222 005224'                ,+2             ; EXIT POLISH
39 05224 012704         MOV     #GRA,3,R4       ; NEW POLISH PC
         005242'
40 05230 000134         POLRET                  ; AND CONTINUE THERE
41 05232 004767 GRA,1:  JSR     PC,QIKPOL       ; ENTER POLISH
```

```
        173054
42 05236 005042'                +SCALEY         ; SCALE DY FOR XGRA
43 05240 004522'                +POSY           ; MAKE >=0
44 05242 001440'GRA,3:          +TST64          ; CHECK IF < 64 AND ONTO STK
45 05244 001702'                +CKOPCM         ; CHECK OPTIONAL COMMA
46 05246 004542'                +SAVER1         ; SAVE PTR TO ARRAY
47 05250 000340'                +SKPARG         ; TO GET BY ARRAY
48 05252 003050'                +LIFT           ; GET L,I,F,T
49 05254 005256'                ,+2             ; EXIT POLISH
50 05256 056700         BIS     MODE,R0         ; ADD MODE TO SGM
        000412
51 05262 010167         MOV     R1,MODE         ; TEMP, SAVE PTR TO ")"
        000406
52 05266 004767         JSR     PC,NEWPUT       ; INSERT IT/ FORCE NEW SGM
        173352
53 05272 016600         MOV     2(SP),R0        ; GET SCALED INCREMENT
        000002
54 05276 001001         BNE     GRA1            ; BR IF R0>0
55 05300        GRA0:   .IFDF   $CRASH
56                      JMP     ERRAOR          ; ARGUMENT OUT OF RANGE
57                      .ENDC
58                      .IFNDF  $CRASH
59 05300 005200         INC     R0              ; MAKE R0 = 1
60                      .ENDC
61 05302 005767 GRA1:   TST     XYBIG           ; EXCEED 63?
        174522
62 05306 001402         BEQ     GRA2            ; NO= BRANCH
63                      .IFDF   $CRASH
64                      BR      GRA0            ; CRASH
65                      .ENDC
66                      .IFNDF  $CRASH
67 05310 012700         MOV     #77,R0          ; MAKE = 63
        000077
68                      .ENDC
69 05314 052700 GRA2:   BIS     #174100,R0      ; LOAD STAT, REG, B
        174100
70 05320 004767         JSR     PC,PUTWD        ; INSERT IT
        173324
71 05324 012700         MOV     #DJMP,R0        ; FOLLOWED BY DJMP
        160000
72 05330 004767         JSR     PC,PUTWD        ; INSERT IT
        173314
73 05334 012601         MOV     (SP)+,R1        ; POINT AT ARRAY
74 05336 005726         TST     (SP)+           ; CLEAN STK
75 05340 112102         MOVB    (R1)+,R2
76 05342 100411         BMI     GRASYN          ; SHOULD NOT FIND A TOKEN
77 05344 000302         SWAB    R2
78 05346 152102         BISB    (R1)+,R2
79 05350 061502         ADD     (R5),R2         ; ADR IN SYMTAB OF ARRAY
80 05352 121127         CMPB    (R1),#,COMMA    ; BETTER BE FOLLOWED BY A ","
        000000G
81 05356 001407         BEQ     GRA3            ; YES= BRANCH
82 05360 121127         CMPB    (R1),#,RPAR     ; OR MAYBE A ")"
        000000G
83 05364 001404         BEQ     GRA3            ; YES= BRANCH
84 05366 000167 GRASYN: JMP     ERRSYN          ; SYNTAX ERROR
        000000G
```

GTB V01-01 BASIC - GT ROOT MOD RT-11 MACRO VM02-09 16-OCT-74 02:06:07 PAGE 29+

```
85 05372 000167 GRAARG: JMP     ERRARG          ; ARGUMENT ERROR
        000000G
86 05376 022227 GRA3:   CMP     (R2)+,#-2       ; BETTER BE AN ARRAY
        177776
87 05402 001371         BNE     GRASYN          ; NO= ERROR
88 05404 012200         MOV     (R2)+,R0        ; GET ADR OF (SCALAR BEFORE) ARRAY
89 05406 022020         CMP     (R0)+,(R0)+     ; POINT TO A(0)
90 05410 004767         JSR     PC,PUTWD        ; INSERT AFTER DJMP
        173234
91 05414 011201         MOV     (R2),R1         ; IS SS1 >0?
92 05416 003765         BLE     GRAARG          ; NO= ERROR
93 05420 010046         MOV     R0,-(SP)        ; SAVE ADR
94 05422 010200         MOV     R2,R0
95 05424 004767         JSR     PC,PUTWD        ; SAVE ADR OF SS1
        173220
96 05430 012600         MOV     (SP)+,R0        ; RESTORE R0
97 05432 005022         CLR     (R2)+           ; MAKE SS1=0 IN SYMTAB
98 05434 021227         CMP     (R2),#-1        ; CHECK SS2 = -1?
        177777
99 05440 001354         BNE     GRAARG          ; NO= ERROR
100 5442 016712         MOV     SS2TMP,(R2)     ; STORE NEW SS2
        000230
101 5446 010002         MOV     R0,R2           ; STORE START OF SCALED ARRAY
102 5450 010667         MOV     SP,ABS,F        ; ABSOLUTE MODE
        176164
103                     .IFDF   $LPS
104 5454 005067         CLR     LPS,F           ; ASSUME NOT AN LPS ARRAY
        000114
105 5460 012704         MOV     #NARRAY,R4      ; NO, OF ENTRIES IN BUFFER TABLE
        000000G
106 5464 001413         BEQ     GRA7            ; NOTHING THERE
107 5466 012703         MOV     #TABLE,R3       ; ADR OF FIRST 6-WORD BLOCK
        000000G
108 5472 021300 GRA6:   CMP     (R3),R0         ; ADR, MATCH?
109 5474 001003         BNE     GRA10           ; NO= BRANCH
110 5476 010667         MOV     SP,LPS,F        ; SET LPS FLAG
        000072
111 5502 000404         BR      GRA7            ; EXIT
112 5504 062703 GRA10:  ADD     #12.,R3         ; GO TO NEXT BLOCK
        000014
113 5510 005304         DEC     R4              ; ANY ENTRIES LEFT?
114 5512 003367         BGT     GRA6            ; YES= BRANCH
115                     .ENDC
116 5514 010003 GRA7:   MOV     R0,R3           ; GET START OF UNSCALED DATA
117                     .IFDF   $LPS
118 5516 005767         TST     LPS,F           ; AN LPS ARRAY?
        000052
119 5522 001403         BEQ     GRA7A           ; NO= BRANCH
120 5524 006301         ASL     R1              ; YES= DOUBLE ARRAY ENTRIES
121 5526 005401         NEG     R1
122 5530 005201         INC     R1              ; MAKE -DIM+1
123 5532        GRA7A:  .ENDC
124 5532 010100         MOV     R1,R0
125 5534 004767         JSR     PC,PUTWD        ; STORE SS1 IN DPY FILE
        173110
126                     .IFDF   $LPS
127 5540 005767         TST     LPS,F           ; AN LPS ARRAY?
```

```
                000030
128 5544 001414         BEQ     GRALP           ; NO- DON'T NEGATE
129 5546 005401         NEG     R1              ; USED >0 HERE
130 5550 005767 GRA9:   TST     LPS,F           ; AN LPS ARRAY?
         000020
131 5554 001410         BEQ     GRALP           ; NO- BRANCH
132 5556 012300         MOV     (R3)+,R0        ; GET DATA
133 5560 010346         MOV     R3,-(SP)        ; SAVE R3
134 5562 010003         MOV     R0,R3           ; SET UP FOR LPS "FLOAT"
135 5564 004767         JSR     PC,FLOAT        ; FLOAT INTO FAC
         000000G
136 5570 012603         MOV     (SP)+,R3        ; RESTORE R3
137 5572 000405         BR      GRALP1          ; AND CONTINUE
138 5574 000000 LPS,F:  ,WORD   0               ; NON-ZERO IF LPS ARRAY
139                     ,ENDC
140 5576 012365 GRALP:  MOV     (R3)+,FAC1(R5)  ; GET DATA
         000040
141 5602 012365         MOV     (R3)+,FAC2(R5)  ;    AND SECOND WORD
         000042
142                     .IFDF   $LPS
143 5606        GRALP1: .ENDC
144 5606 026727         CMP     SS2TMP,#-2      ; XGRA OR YGRA?
         000064
         177776
145 5614 001405         BEQ     GRA4            ; BR IF XGRA
146 5616 004767         JSR     PC,QIKPOL       ; ENTER POLISH
         172470
147 5622 005042'                +SCALEY         ; SCALE IT TO R0
148 5624 004522'                +POSY           ; MAKE SURE >= 0
149 5626 005642'                +GRA5           ; CONTINUE THERE
150 5630 004767 GRA4:   JSR     PC,QIKPOL       ; ENTER POLISH
         172456
151 5634 004770'                +SCALEX         ; SCALE X TO R0
152 5636 004514'                +POSX           ; MAKE SURE >= 0
153 5640 005642'                .+2             ; EXIT POLISH MODE
154 5642 052700 GRA5:   BIS     #40000,R0       ; MAKE VISIBLE
         040000
155 5646 010022         MOV     R0,(R2)+        ; STORE SCALED VALUE
156 5650 005301         DEC     R1              ; DECREMENT COUNT
157                     .IFDF   $LPS
158 5652 002336         BGE     GRA9            ; LOOP
159                     .ENDC
160                     .IFNDF  $LPS
161                     BGE     GRALP           ; LOOP
162                     .ENDC
163 5654 016701         MOV     MODE,R1         ; RESTORE R1
         000014
164 5660 012722 SDXIT1: MOV     #DJMP,(R2)+     ; INSERT DJMP
         160000
165 5664 016712         MOV     DISEND,(R2)     ; AND ADDRESS
         172746
166 5670 000167         JMP     SDXIT0          ; CK ")" AND EXIT TO USER
         176310
167
168 5674 000000 MODE:   ,WORD   0               ; GRAPH-X OR GRAPH-Y MODE
169 5676 000000 SS2TMP: ,WORD   0               ; NEW VALUE OF SS2MAX
170                     .IFDF   $DISK
```



```
1                       .SBTTL  "SCAL" ROUTINE
2                       .ENDC
3                ;      ****************************************
4
5                       ; ROUTINE TO HANDLE
6                       ;       CALL "SCAL"(LLX,LLY,URX,URY [,INX,INY])
7
8 005700        SCAL:
9 005700 004767         JSR     PC,BUFPOL       ; ENTER POLISH AND CK "("
         172300
10 05704 000136'                +GETONE         ; GET LLX
11 05706 004574'                +PUSHF          ; FLOAT TO STK
12 05710 000026'                +PLOOP          ; POLISH LOOP
13 05712 000002                 +2              ; TWICE
14 05714 000076'                +GETNUM         ; GET ",LLY,URX"
15 05716 004574'                +PUSHF          ; FLOAT TO STK
16 05720 000040'                +ELOOP          ; END POLISH LOOP
17 05722 004546'                +REVRSE         ; REVERSE LLU AND URX
18 05724 000076'                +GETNUM         ; GET URY
19 05726 004574'                +PUSHF          ; FLOAT TO STK
20 05730 005732'                .+2             ; EXIT POLISH, CK INX, INY LATER
21 05732 016667         MOV     4(SP),LLY       ;
         000004
         000410
22 05740 016667         MOV     6(SP),LLY+2     ; SAVE LLY
         000006
         000404
23 05746 016667         MOV     12.(SP),LLX     ;
         000014
         000404
24 05754 016667         MOV     14.(SP),LLX+2   ; SAVE LLX
         000016
         000400
25 05762 004467         JSR     R4,FPPSAV       ; ENTER POLISH AND SAVE REGISTER
         172430
26 05766 006234'                +MULVEC         ; LOAD ERROR VECTOR FOR SUBTRACT
27 05770 000000G                +$SBR           ; LLY-URY
28 05772 000176'                +NEGSTK         ; URY=LLY
29 05774 000256'                +POP            ; SAVE IN FAC
30 05776 000000G                +$SBR           ; LLX-URX
31 06000 000176'                +NEGSTK         ; URX=LLX
32 06002 000442'                +FPPRES         ; RESTORE REGS, AND EXIT POLISH
33 06004 012667         MOV     (SP)+,FX
         000364
34 06010 012667         MOV     (SP)+,FX+2      ; SAVE URX-LLX TEMPORARILY
         000362
35 06014 012767         MOV     #USCL,Y,SCPTR   ; POINTER = LLY+2
         006352'
         000166
36 06022 004767         JSR     PC,SCLGEN       ; GENERATE SCALE IN Y
         000216
37 06026 012667         MOV     (SP)+,FYL
         000336
38 06032 012667         MOV     (SP)+,SCL,Y     ; STORE FYL = -FY*LLY
         000334
39 06036 012667         MOV     (SP)+,IFY
         000302
```

78

```
40 06042 012667          MOV     (SP)+,IFY+2     ; STORE IFY = 1/FY
         000300
41 06046 016567          MOV     FAC1(R5),FY
         000040
         000310
42 06054 016567          MOV     FAC2(R5),FY+2   ; STORE FY
         000042
         000304
43 06062 016765          MOV     FX,FAC1(R5)
         000306
         000040
44 06070 016765          MOV     FX+2,FAC2(R5)   ; PUT URX-LLX IN FAC
         000302
         000042
45 06076 012767          MOV     #USCL,X,SCPTR   ; POINTER = LLX+2
         006362'
         000104
46 06104 004767          JSR     PC,SCLGEN       ; GENERATE SCALE IN X
         000134
47 06110 012667          MOV     (SP)+,FXL
         000264
48 06114 012667          MOV     (SP)+,SCL,X     ; STORE FXL = -FX*LLX
         000262
49 06120 012667          MOV     (SP)+,IFX
         000230
50 06124 012667          MOV     (SP)+,IFX+2     ; STORE IFX = 1/FX
         000226
51 06130 016567          MOV     FAC1(R5),FX     ; STORE FX
         000040
         000236
52 06136 016567          MOV     FAC2(R5),FX+2
         000042
         000232
53 06144 012700          MOV     #FX,R0          ; SET UP SOURCE VARIABLE ADDRESS
         006374'
54 06150 004767          JSR     PC,OPTSTR       ; STORE INX = FX IF USER WANTS IT
         000142
55 06154 016765          MOV     FY,FAC1(R5)
         000204
         000040
56 06162 016765          MOV     FY+2,FAC2(R5)   ; MOVE FY TO FAC
         000200
         000042
57 06170 012700          MOV     #FY,R0          ; SET UP SOURCE VARIABLE ADDRESS
         006364'
58 06174 004767          JSR     PC,OPTSTR       ; STORE INY = FY IF USER WANTS IT
         000116
59 06200 010667 SCAL0:   MOV     SP,SCAL,F       ; SCALING HAS BEEN DONE
         174506
60 06204 000167          JMP     CKRPAR          ; CK ")" AND EXIT TO USER
         172106
61
62 06210 000000 SCPTR:   .WORD 0                 ; PTR FOR A POLISH ROUTINE
63                       .IFDF   $DISK
64                       .EOT
```



```
2                ;
3                ;       DEC - 11 - LBPGB - A - LA    BASIC KERNEL V02-01
4                ;
5                ;       GTB,MAC TAPE 4 OF 4
6                ;
7                ; THE INFORMATION IN THIS LISTING IS SUBJECT TO CHANGE WITHOUT NOTICE
8                ; AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
9                ; CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY
10               ; FOR ANY ERRORS THAT MAY APPEAR IN THIS LISTING.
11               ;
12               ; THIS SOFTWARE IS FURNISHED TO THE PURCHASER UNDER A LICENSE
13               ; FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH
14               ; INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN
15               ; SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY
16               ; DIGITAL.
17               ;
18               ; DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE
19               ; USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT
20               ; SUPPLIED BY DIGITAL.
21               ;
22               ; COPYRIGHT (C) 1973,1974, BY DIGITAL EQUIPMENT CORPORATION.
23               ;
24               ;       AUTHOR: DR. STEPHEN R. ALPERT   AUGUST 1973
25                       .SBTTL  ROUTINES AUXILIARY TO "SCAL"
26                       .ENDC
27               ;       ****************************************
28
29                       ; SOME NECESSARY ROUTINES TO ACCOMPLISH SCALING
30
31 06212        PSHPTR:
32 06212 016700          MOV     SCPTR,R0        ; R0 HAS ADDRESS OF LO-ORDER WORD
         177772
33 06216 011046          MOV     (R0),-(SP)      ; MOVE TO STK
34 06220 014046          MOV     -(R0),-(SP)
35 06222 000134          POLRET
36
37 06224        TEN24:
38 06224 005046          CLR     -(SP)
39 06226 012746          MOV     #42600,-(SP)    ; PUT 1024. ON STK
         042600
40 06232 000134          POLRET
41
42 06234        MULVEC:
43 06234 012767          MOV     #ERSTAR,$ERVEC  ; MULT. ERROR VECTOR
         004434'
         000000G
44 06242 000134          POLRET
45
46               ;       ****************************************
47
48                       ; SCALE GENERATOR ROUTINE
49                       ; ON EXIT:
50                       ;       FX IN FAC
51                       ;       FXL AT (SP), 2(SP)
52                       ;       IFX AT 4(SP), 6(SP)
53
54 06244        SCLGEN:
```

79

```
55 06244 012667          MOV     (SP)+,SCLSAV    ; SAVE RETURN ADDRESS OFF STK
         000044
56 06250 004467          JSR     R4,FPPSAV       ; SAVE REGS AND ENTER POLISH
         172142
57 06254 000246'                 +PUSH1          ; PUT 1.0 ON STK
58 06256 006224'                 +TEN24          ; PUT 1024. ON STK
59 06260 004504'                 +DIVVEC         ; DIVIDE ERROR VECTOR
60 06262 000234'                 +PUSH           ; PUT FAC ON STK
61 06264 000000G                 +$DVR           ; 1024./DIFF, = FX
62 06266 000256'                 +POP            ; PUT FX IN FAC
63 06270 000234'                 +PUSH           ; PUT BACK ON STK
64 06272 000000G                 +$DVR           ; 1./FX = IFX
65 06274 000234'                 +PUSH           ; FX ONTO STK AGAIN
66 06276 000176'                 +NEGSTK         ; NOW -FX
67 06300 006212'                 +PSHPTR         ; PUT LLX ON STK
68 06302 006234'                 +MULVEC         ; MULT, ERROR VECTOR
69 06304 000000G                 +$MLR           ; FXL = -FX*LLX
70 06306 000442'                 +FPPRES         ; EXIT POLISH AND RESTORE REGS
71 06310 000177          JMP     @SCLSAV         ; RETURN TO CALLER
         000000
72
73 06314 000000 SCLSAV:  .WORD 0                 ; RETURN ADDRESS
74                       .IFDF   $DISK
```



```
1                        .SBTTL  OPTIONAL STORE ROUTINE AND DATA LAYOUT FOR SCAL
2                        .ENDC
3                ;       ****************************************
4
5                        ; ROUTINE TO OPTIONALLY STORE (R0),2(R0) IN NEXT PARAM
6                        ;       IF THERE IS ONE,
7                        ; R1 POINTS TO COMMA BEFORE PARAMETER
8                        ; R0 POINTS TO ENTRY TO BE MOVED
9
10 06316         OPTSTR:
11 06316 004767          JSR     PC,QIKPOL       ; ENTER POLISH
         171770
12 06322 000066'                 +SETOPT         ; SET OPTIONAL FLAG
13 06324 001702'                 +CKOPCM         ; CK OPTIONAL ","
14 06326 006330'                 .+2             ; EXIT POLISH
15 06330 103404          BCS     OPTST0          ; NO- A RIGHT PAREN
16 06332 004767          JSR     PC,GETADR       ; GET ADR IN R2
         173500
17 06336 012022          MOV     (R0)+,(R2)+
18 06340 012022          MOV     (R0)+,(R2)+     ; STORE RESULT
19 06342 000207 OPTST0:  RETURN
20
21               ;       ****************************************
22
23                       ; DATA STORAGE FOR SCALING
24
25 06344 000000 IFY:     .WORD 0,0               ; IFY = 1/FY
   06346 000000
26 06350 000000 LLY:     .WORD 0                 ; LLY
27 06352 000000 USCL.Y:  .WORD 0
28
29 06354 000000 IFX:     .WORD 0,0               ; IFX = 1/IFY
   06356 000000
30 06360 000000 LLX:     .WORD 0                 ; LLX
31 06362 000000 USCL.X:  .WORD 0
32
33 06364 000000 FY:      .WORD 0,0               ; FY
   06366 000000
34 06370 000000 FYL:     .WORD 0                 ; FYL = -FY*LLY
35 06372 000000 SCL.Y:   .WORD 0
36
37 06374 000000 FX:      .WORD 0,0               ; FX
   06376 000000
38 06400 000000 FXL:     .WORD 0                 ; FXL = IFX*LLX
39 06402 000000 SCL.X:   .WORD 0
40                       .IFDF   $DISK
```

```
1                       .SBTTL  POLISH ROUTINE TO SORT VARIABLE TYPES
2                       .ENDC
3               ;       ****************************************
4
5                       ; ROUTINE (POLISH) TO RETURN IN R0
6                       ;   -1 IF ARGUMENT IS SCALAR OR REGULAR ARRAY
7                       ;    0 IF XGRA ARRAY
8                       ;    2 IF YGRA ARRAY
9                       ;    4 IF FIGR ARRAY
10                      ; ** CRASH IF A STRING ARGUMENT
11                      ; R1 POINTS TO PARAMETER IN TOKEN FORM
12
13 06404        ADRCHK:
14 06404 004767         JSR     PC,GETADR       ; GET THE ADDRESS
         173426
15 06410 016500         MOV     VARSAV(R5),R0   ; GET POINTER TO SYMTAB ENTRY
         000022
16 06414 021027         CMP     (R0),#-2        ; ARRAY OR SCALAR?
         177776
17 06420 001404         BEQ     ADRCH0          ; BRANCH IF ARRAY
18 06422 003015         BGT     ADRCH3          ; BRANCH IF SCALAR
19 06424 012700 ADRCH1: MOV     #-1,R0          ; FLAG AS -1
         177777
20 06430 000134 ADRCH2: POLRET                  ; AND EXIT
21 06432 016000 ADRCH0: MOV     6(R0),R0        ; GET SS2MAX
         000006
22 06436 005200         INC     R0
23 06440 002371         BGE     ADRCH1          ; BRANCH IF SS2MAX >=-1
24 06442 005200         INC     R0
25 06444 005400         NEG     R0
26 06446 006300         ASL     R0              ; R0 <-- -2 * (R0 + 1)
27 06450 020027         CMP     R0,#4           ; BETTER NOT BE >4
         000004
28 06454 003765         BLE     ADRCH2          ; IT ISN'T, EXIT
29 06456 000167 ADRCH3: JMP     ERRARG          ; IT IS, FAIL
         000000G
30                      .IFDF   $DISK
```



```
1                       .SBTTL  "APUT" ROUTINE
2                       .ENDC
3               ;       ****************************************
4
5                       ; ROUTINE TO HANDLE
6                       ;       CALL "APUT" ( A(I), Z)
7                       ; "A" MUST BE XGRA, YGRA OR FIGR ARRAY
8                       ; "Z" CANNOT BE ANY OF THE ABOVE
9
10 06462        APUT:
11 06462 004767         JSR     PC,BUFPOL       ; ENTER POLISH AND CK "("
         171516
12 06466 004542'                +SAVER1         ; R1 TO STK (PTR TO A(I) )
13 06470 000340'                +SKPARG         ; SKIP OVER A(I)
14 06472 000076'                +GETNUM         ; CK "," AND GET VALUE OF Z
15 06474 004542'                +SAVER1         ; SAVE PTR TO ")" ON STK
16 06476 000234'                +PUSH           ; SAVE Z ON STK
17 06500 006502'                .+2             ; END POLISH MODE
18 06502 016601         MOV     6(SP),R1        ; POINT AT A(I) AGAIN
         000006
19 06506 004767         JSR     PC,QIKPOL       ; IMMED, POLISH
         171600
20 06512 006404'                +ADRCHK         ; CHECK ITS TYPE
21 06514 000256'                +POP            ; PUT Z BACK IN FAC
22 06516 006520'                .+2             ; EXIT POLISH
23 06520 005700         TST     R0              ; XGRA, YGRA, OR FIGR?
24 06522 002002         BGE     APUT1           ; YES- BRANCH
25 06524 000167         JMP     ERRARG          ; NO- CRASH
         000000G
26 06530 010266 APUT1:  MOV     R2,2(SP)        ; SAVE ADR OF A(I) OVER PTR TO A(I)
         000002
27 06534 010667         MOV     SP,ABS.F        ; ABSOLUTE MODE (XGRA,YGRA)
         175100
28 06540 016007         MOV     T1(R0),PC       ; DISPATCH
         006544'
29 06544 006552'T1:             +APUTX          ; XGRA
30 06546 006564'                +APUTY          ; YGRA
31 06550 006576'                +APUTF          ; FIGR
32 06552 004767 APUTX:  JSR     PC,QIKPOL       ; ENTER POLISH
         171534
33 06556 004770'                +SCALEX         ; SCALE IT
34 06560 004514'                +POSX           ; MAKE SURE >=0
35 06562 006646'                +APUT2          ; CONTINUE AT APUT2
36 06564 004767 APUTY:  JSR     PC,QIKPOL       ; ENTER POLISH
         171522
37 06570 005042'                +SCALEY         ; SCALE IT
38 06572 004522'                +POSY           ; MAKE SURE >=0
39 06574 006646'                +APUT2          ; CONTINUE AT APUT2
40 06576 005067 APUTF:  CLR     ABS.F           ; FLAG RELATIVE MODE
         175036
41 06602 006065         ROR     SS1SAV(R5)      ; X OR Y ENTRY?
         000024
42 06606 103012         BCC     APUTF0          ; X- BRANCH
43 06610 004767         JSR     PC,ENTERP       ; IMMED, POLISH
         171342
44 06614 005042'                +SCALEY         ; SCALE Y
45 06616 000367         SWAB    YSIGN
```

```
          173212
46 06622 006267          ASR     YSIGN           ; MOVE SIGN TO BIT 13
          173206
47 06626 056700          BIS     YSIGN,R0        ; SET SIGN BIT IN ANSWER
          173202
48 06632 000405          BR      APUT2           ; CONTINUE AT APUT2
49 06634 004767 APUTF0:  JSR     PC,ENTERP       ; IMMED. POLISH
          171316
50 06640 004770'                 +SCALEX         ; SCALE X
51 06642 056700          BIS     XSIGN,R0        ; SET SIGN BIT
          173164
52 06646           APUT2:
53 06646 012601          MOV     (SP)+,R1        ; POINT TO ")"
54 06650 017646          MOV     @(SP),-(SP)     ; GET VALUE
          000000
55 06654 042716          BIC     #137777,(SP)    ; CLEA ALL BUT VISIBILITY
          137777
56 06660 050016          BIS     R0,(SP)         ; SET IN VALUE
57 06662 012636          MOV     (SP)+,@(SP)+    ; STORE A(I)
58 06664 000464          BR      AGET4           ; EXIT THRU CKRPAR
59                       .IFDF   $DISK
```



```
1                        .SBTTL  "AGET" ROUTINE
2                        .ENDC
3                ;       ****************************************
4
5                        ; ROUTINE TO HANDLE:
6                        ;       CALL "AGET" ( A(I), Z)
7                        ; "A" MUST BE XGRA, YGRA, OR FIGR ARRAY
8                        ; "Z" MUST NOT BE ANY OF THE ABOVE
9
10 06666           AGET:
11 06666 004767          JSR     PC,BUFPOL       ; ENTER POLISH AND CK "("
          171312
12 06672 006404'                 +ADRCHK         ; CK ADR OF A(I)
13 06674 001702'                 +CKOPCH         ; CK ","
14 06676 006700'                 .+2             ; EXIT POLISH
15 06700 005700          TST     R0              ; XGRA, YGRA, OR FIGR?
16 06702 002002          BGE     AGET0           ; YES- BRANCH AND CONTINUE
17 06704 000167 AGET3:   JMP     ERRARG          ; NO- CRASH
          000000G
18 06710 010667 AGET0:   MOV     SP,ABS,F        ; ABSOLUTE MODE
          174724
19 06714 011265          MOV     (R2),FAC2(R5)   ; GET VALUE FROM ARRAY
          000042
20 06720 005065          CLR     FAC1(R5)        ;       INTO FAC
          000040
21 06724 042765          BIC     #40000,FAC2(R5) ; CLEAR INTENSITY BIT
          040000
          000042
22 06732 016007          MOV     T0(R0),PC       ; DISPATCH
          006736'
23 06736 006744'T0:              +AGETX          ; XGRA
24 06740 007004'                 +AGETY          ; YGRA
25 06742 006752'                 +AGETF          ; FIGR
26 06744 012700 AGETX:   MOV     #USCL.X,R0      ; UNSCALE XGRA
          006362'
27 06750 000417          BR      AGET1           ; AND CONTINUE
28 06752 032712 AGETF:   BIT     #20000,(R2)     ; LONG VECTOR <0?
          020000
29 06756 001405          BEQ     AGET2           ; NO- BRANCH
30 06760 005465          NEG     FAC2(R5)        ; YES- NEGATE IT
          000042
31 06764 052765          BIS     #176000,FAC2(R5) ; MAKE IT TRUE NEGATIVE
          176000
          000042
32 06772 005067 AGET2:   CLR     ABS,F           ; FIGR IS RELATIVE
          174642
33 06776 006065          ROR     $S1SAV(R5)      ; X OR Y ENTRY?
          000024
34 07002 103360          BCC     AGETX           ; X- BRANCH
35 07004 012700 AGETY:   MOV     #USCL.Y,R0      ; UNSCALE YGRA
          006352'
36 07010 004767 AGET1:   JSR     PC,SC.USC       ; DO IT NOW
          175304
37 07014 004767          JSR     PC,QIKPOL       ; ENTER POLISH
          171272
38 07020 000234'                 +PUSH           ; PUSH FAC ON STK
39 07022 006404'                 +ADRCHK         ; CK ADR OF Z
```

82

```
40 07024 007026'                .+2             ; EXIT POLISH
41 07026 005700         TST     R0              ; XGRA, YGRA, OR FIGR?
42 07030 002325         BGE     AGET3           ; YES- ERROR => CRASH
43 07032 012622         MOV     (SP)+,(R2)+     ; STORE UNSCALED RESULT
44 07034 012622         MOV     (SP)+,(R2)+     ;       AND REST OF IT
45 07036 000167 AGET4:  JMP     CKRPAR          ; CK ")" AND EXIT TO USER
         171254
46                      .IFDF   $DISK
```



```
1                       .SBTTL  "FIGR" ROUTINE
2                       .ENDC
3               ;       ****************************************
4
5                       ; ROUTINE TO HANDLE:
6                       ;       CALL "FIGR" (A [, L, I, F, T ] )
7
8 007042        FIGR:
9 007042 004767         JSR     PC,BUFPOL       ; ENTER POLISH AND CK "("
         171136
10 07046 004542'                +SAVER1         ; SAVE PTR TO A ON STK
11 07050 000340'                +SKPARG         ; DON'T TRY TO EVALUATE IT
12 07052 003050'                +LIFT           ; GET L, I, F, T
13 07054 004542'                +SAVER1         ; SAVE PTR TO ")"
14 07056 007060'                .+2             ; EXIT POLISH
15 07060 052700         BIS     #110000,R0      ; LONG VECTOR MODE
         110000
16 07064 004767         JSR     PC,NEWPUT       ; INSERT SGM WORD
         171554
17 07070 016601         MOV     2(SP),R1        ; POINT AT A
         000002
18 07074 012616         MOV     (SP)+,(SP)      ; CLEAN STK A BIT
19 07076 112102         MOVB    (R1)+,R2
20 07100 100512         BMI     FIGSYN          ; ERROR IF TOKEN HERE
21 07102 000302         SWAB    R2
22 07104 152102         BISB    (R1)+,R2
23 07106 061502         ADD     (R5),R2         ; OFFSET TO SYMTAB
24 07110 022227         CMP     (R2)+,#-2       ; BETTER BE AN ARRAY
         177776
25 07114 001104         BNE     FIGSYN          ; NO- ERROR
26 07116 012700         MOV     #DJMP,R0
         160000
27 07122 004767         JSR     PC,PUTWD        ; INSERT DJMP
         171522
28 07126 012200         MOV     (R2)+,R0        ; GET ADDRESS OF ARRAY
29 07130 022020         CMP     (R0)+,(R0)+     ; POINT AT A(0)
30 07132 010003         MOV     R0,R3           ; PUT IN R3 ALSO
31 07134 004767         JSR     PC,PUTWD        ; INSERT IN DPY FILE
         171510
32 07140 011201         MOV     (R2),R1         ; IS SS1 >0?
33 07142 003467         BLE     FIGARG          ; NO- ERROR (NOT ENUF ROOM)
34 07144 005301         DEC     R1              ; TO ROUND CORRECTLY
35 07146 052701         BIS     #1,R1           ; MAKE ODD
         000001
36 07152 010200         MOV     R2,R0
37 07154 004767         JSR     PC,PUTWD        ; ADR OF SS1
         171470
38 07160 010100         MOV     R1,R0
39 07162 004767         JSR     PC,PUTWD        ; SAVE SS1 IN DPY FILE
         171462
40 07166 005022         CLR     (R2)+           ; ZERO SS1 IN SYMTAB
41 07170 005201         INC     R1              ; ONE MORE SINCE ZERO TOO
42 07172 021227         CMP     (R2),#-1        ; CHECK SS2 = -1?
         177777
43 07176 001051         BNE     FIGARG          ; NO- ERROR
44 07200 012712         MOV     #-4,(R2)        ; STORE -4 IN SS2MAX
         177774
```

```
45 07204 010302          MOV     R3,R2           ; ALSO IN R2
46 07206 005067          CLR     ABS,F           ; RELATIVE MODE
         174426
47 07212 005067          CLR     SS2TMP          ; SWITCH FOR TOGGLING
         176460
48 07216 012365 FIGRLP: MOV      (R3)+,FAC1(R5)  ; GET ARRAY ELEMENT
         000040
49 07222 012365          MOV     (R3)+,FAC2(R5)  ; GET SECOND HALF
         000042
50 07226 005767          TST     SS2TMP          ; X OR Y ENTRY?
         176444
51 07232 001010          BNE     FIGR1           ; Y= BRANCH
52 07234 005267          INC     SS2TMP          ; MAKE Y NEXT TIME
         176436
53 07240 004767          JSR     PC,ENTERP       ; IMMED. POLISH
         170712
54 07244 004770'                 +SCALEX         ; SCALE X RELATIVE
55 07246 056700          BIS     XSIGN,R0        ; PUT IN SIGN
         172560
56 07252 000413          BR      FIGR2           ; INSERT IT AND CONTINUE
57 07254 005067 FIGR1:  CLR     SS2TMP          ; MAKE X NEXT TIME
         176416
58 07260 004767          JSR     PC,ENTERP       ; IMMED. POLISH
         170672
59 07264 005042'                 +SCALEY         ; SCALE Y RELATIVE
60 07266 000367          SWAB    YSIGN           ; SET SIGN BIT IN BIT 13
         172542
61 07272 006267          ASR     YSIGN           ; ALL SET
         172536
62 07276 056700          BIS     YSIGN,R0        ; SET IN R0
         172532
63 07302 056700 FIGR2:  BIS     INVIS,R0        ; FLAG VISIBLE OR NOT
         173730
64 07306 010022          MOV     R0,(R2)+        ; INSERT SCALED WORD
65 07310 005301          DEC     R1
66 07312 003341          BGT     FIGRLP          ; LOOP IF NOT DONE
67 07314 012601          MOV     (SP)+,R1        ; POINT R1 AT ")"
68 07316 000167          JMP     $DXIT1          ; INSERT/ CK ")"/ EXIT TO USER
         176336
69 07322 000167 FIGARG: JMP     ERRARG          ; ARGUMENT ERROR
         000000G
70 07326 000167 FIGSYN: JMP     ERRSYN          ; SYNTAX ERROR
         000000G
71                       .IFDF   $DISK
```



```
1                        .SBTTL  "FPUT" ROUTINE
2                        .ENDC
3                ;       ****************************************
4
5                        ; ROUTINE TO HANDLE:
6                        ;       CALL "FPUT"(A(I),Z)
7                        ; "A" MUST BE A FIGR ARRAY
8                        ; "Z" MUST NOT BE ANY OF FIGR,XGRA,YGRA
9
10 07332         FPUT:
11 07332 004767          JSR     PC,BUFPOL       ; ENTER POLISH AND CK "("
         170646
12 07336 004542'                 +SAVER1         ; SAVE PTR TO A(I)
13 07340 000340'                 +SKPARG         ; SKIP OVER A(I)
14 07342 000076'                 +GETNUM         ; GET Z INTO FAC
15 07344 004542'                 +SAVER1         ; SAVE PTR TO ")"
16 07346 000234'                 +PUSH           ; SAVE Z ON STK
17 07350 007352'                 .+2             ; EXIT POLISH
18 07352 016601          MOV     6(SP),R1        ; POINT R1 AT A(I)
         000006
19 07356 004767          JSR     PC,QIKPOL       ; IMMED. POLISH
         170730
20 07362 006404'                 +ADRCHK         ; CK ADR OF A(I)
21 07364 000256'                 +POP            ; PUT Z INTO FAC
22 07366 007370'                 .+2             ; EXIT POLISH
23 07370 020027          CMP     R0,#4           ; R0 SHOULD BE 4
         000004
24 07374 001402          BEQ     FPUT0           ; YES- BRANCH
25 07376 000167          JMP     ERRARG          ; NO- ERROR
         000000G
26 07402 010266 FPUT0:  MOV     R2,2(SP)        ; REPLACE PTR TO A(I) BY ITS ADR
         000002
27 07406 005067          CLR     ABS,F           ; RELATIVE MODE
         174226
28 07412 006065          ROR     SS1SAV(R5)      ; X OR Y ENTRY?
         000024
29 07416 103013          BCC     FPUT1           ; X= BRANCH
30 07420 004767          JSR     PC,ENTERP       ; IMMED. POLISH
         170532
31 07424 005042'                 +SCALEY         ; SCALE Y TO R0
32 07426 000367          SWAB    YSIGN           ; MOVE SIGN BIT
         172402
33 07432 006267          ASR     YSIGN           ;   TO 13
         172376
34 07436 016767          MOV     YSIGN,XYBIG     ; SAVE SIGN
         172372
         172364
35 07444 000406          BR      FPUT2           ; AND CONTINUE
36 07446 004767 FPUT1:  JSR     PC,ENTERP       ; IMMED. POLISH
         170504
37 07452 004770'                 +SCALEX         ; SCALE X TO R0
38 07454 016767          MOV     XSIGN,XYBIG     ; SAVE SIGN
         172352
         172346
39 07462 016603 FPUT2:  MOV     2(SP),R3        ; ADR OF A(I)
         000002
40 07466 012616          MOV     (SP)+,(SP)      ; CLEAN STACK
```

84

```
41 07470 004767          JSR     PC,UNDO         ; GET ARG A(I) TO R4
         000106
42 07474 010002          MOV     R0,R2           ; SAVE Z
43 07476 056700          BIS     XYBIG,R0        ; SET SIGN
         172326
44 07502 011346          MOV     (R3),-(SP)      ; GET VALUE
45 07504 042716          BIC     #137777,(SP)    ; CLEAR ALL BUT VISIBILITY
         137777
46 07510 050016          BIS     R0,(SP)         ; SET IN VALUE
47 07512 012613          MOV     (SP)+,(R3)      ; STORE IN A(I)
48 07514 005767          TST     XYBIG           ; Z POS OR NEG?
         172310
49 07520 001401          BEQ     FPUT3           ; POS= BRANCH
50 07522 005402          NEG     R2              ; MAKE CORRECT REPRESENTATION
51 07524 160402 FPUT3:   SUB     R4,R2           ; Z-A(I) IN R2
52 07526 005723          TST     (R3)+           ; LOOK AT NEXT WORD
53 07530 022327          CMP     (R3)+,#DJMP     ; END OF ARRAY?
         160000
54 07534 001417          BEQ     FPUT4           ; YES
55 07536 021327          CMP     (R3),#DJMP      ; END OF ARRAY HERE?
         160000
56 07542 001414          BEQ     FPUT4           ; YES
57 07544 004767          JSR     PC,UNDO         ; GET A(I+2) INTO R4
         000032
58 07550 160204          SUB     R2,R4           ; R4<==A(I+2)-(Z-A(I))
59 07552 002003          BGE     FPUT5           ; BRANCH IF POSITIVE
60 07554 005404          NEG     R4              ; ABS, VAL OF R4
61 07556 052704          BIS     #20000,R4       ; SET SIGN NEG,
         020000
62 07562         FPUT5:
63 07562 011346          MOV     (R3),-(SP)      ; GET VALUE
64 07564 042716          BIC     #137777,(SP)    ; CLEAR ALL BUT VISIBILITY
         137777
65 07570 050416          BIS     R4,(SP)         ; SET NEW VALUE
66 07572 012613          MOV     (SP)+,(R3)      ; STORE NEW A(I+2)
67 07574 012601 FPUT4:   MOV     (SP)+,R1        ; POINT AT ")"
68 07576 000167          JMP     CKRPAR          ; CK ")" AND EXIT TO USER
         170514
69
70               ;       ****************************************
71
72                       ; ROUTINE  UNDO
73                       ; TAKE LONG VECTOR WORD POINTED TO BY R3
74                       ; AND PUT IT INTO R4
75                       ; CALLED VIA: JSR PC,UNDO
76
77 07602 011304 UNDO:    MOV     (R3),R4         ; GET ARGUMENT
78 07604 042704          BIC     #40000,R4       ; CLEAR VISIBILITY
         040000
79 07610 032704          BIT     #20000,R4       ; POS, OR NEG, ?
         020000
80 07614 001403          BEQ     UNDO0           ; POS,= BRANCH
81 07616 042704          BIC     #20000,R4       ; CLEAR SIGN BIT
         020000
82 07622 005404          NEG     R4              ; MAKE CORRECT
83 07624 000207 UNDO0:   RETURN                  ; AND EXIT
84                       .IFDF   $DISK
```



```
1                        .SBTTL  "TEXT" ROUTINE
2                        .ENDC
3                ;       ****************************************
4
5                        ; ROUTINE TO HANDLE:
6                        ;       CALL "TEXT"( <ARG, LIST>)
7                        ; WHERE:
8                        ;       +N: <CR> AND N <LF>
9                        ;        0:  <CR> AND NO <LF>
10                       ;       -N: SHIFT-OUT STRING FOLLOWS
11                       ;       R$ OR "STRING":  STRING DATA
12                       ;  IF SHIFT-OUT, ONLY RIGHTMOST 5 BITS
13                       ;  IF SHIFT-IN, IGNORE ALL NON-PRINTABLES
14
15 07626         TEXT:
16 07626 005067          CLR     SHIFT           ; DEFAULT TO SHIFT IN
         000326
17 07632 004767          JSR     PC,BUFPOL       ; ENTER POLISH, CK"("
         170346
18 07636 007640'                 ,+2             ; EXIT POLISH
19 07640 012700          MOV     #100000,R0      ; SET CHARACTER MODE
         100000
20 07644 004767          JSR     PC,PUTSGM       ; INSERT IT
         172074
21 07650 012702          MOV     #CBUFF,R2       ; R2 POINTS TO R0
         010162'
22 07654 000410          BR      TEXT1           ; GET FIRST ARG,
23 07656 122127 TEXT0:   CMPB    (R1)+,#,COMMA   ; DO WE HAVE A COMMA?
         000000G
24 07662 001405          BEQ     TEXT1           ; YES- GET ARG,
25 07664 124127          CMPB    -(R1),#,RPAR    ; NO- NO MORE ARGS?
         000000G
26 07670 001521          BEQ     TEXT11          ; YES- EXIT
27 07672 000167          JMP     ERRSYN          ; NO- ERROR
         000000G
28 07676 010046 TEXT1:   MOV     R0,-(SP)        ; SAVE CHAR, BUFFER
29 07700 010246          MOV     R2,-(SP)        ; AND PTR
30 07702 004767          JSR     PC,EVAL         ; GET ARG,
         000000G
31 07706 103427          BCS     TEXT2           ; BRANCH IF STRING
32 07710 004767          JSR     PC,ENTERP       ; IMMED, POLISH
         170242
33 07714 000006'                 +INTEGR         ; INTEGERIZE IT
34 07716 012602          MOV     (SP)+,R2        ; RESTORE PTR
35 07720 012600          MOV     (SP)+,R0        ; RESTORE BUFFER
36 07722 016503          MOV     FAC2(R5),R3     ; COUNT INTO R3
         000042
37 07726 002003          BGE     TEXT3           ; BRANCH IF CARRIAGE
38 07730 010667          MOV     SP,SHIFT        ; FLAG FOR SHIFT-OUT
         000224
39 07734 000750          BR      TEXT0           ; GET NEXT ARG,
40 07736 012704 TEXT3:   MOV     #15,R4          ; INSERT <CR>
         000015
41 07742 004767          JSR     PC,LDCHAR       ; INSERT IT
         000216
42 07746 012704          MOV     #12,R4          ; INSERT <LF>
         000012
```

85

```
43 07752 005703          TST     R3              ; ANY TO DO?
44 07754 001740 TEXT5:   BEQ     TEXT8           ; NO- EXIT
45 07756 004767          JSR     PC,LDCHAR       ; YES- INSERT IT
         000202
46 07762 005303          DEC     R3              ; DECREMENT COUNTER
47 07764 000773          BR      TEXT5           ; CONTINUE IN LOOP
48 07766 012603 TEXT2:   MOV     (SP)+,R3        ; GET PTR TO STRING
49 07770 012602          MOV     (SP)+,R2
50 07772 012600          MOV     (SP)+,R0        ; RESTORE BUFFER AND PTR
51 07774 020327          CMP     R3,#-1          ; NULL STRING?
         177777
52 10000 001437          BEQ     TEXT8           ; YES- CLEAR SHIFT IF SET
53 10002 010146          MOV     R1,-(SP)        ; SAVE R1
54 10004 111301          MOVB    (R3),R1         ; GET LENGTH OF STRING
55 10006 062703          ADD     #3,R3           ; POINT AT STRING
         000003
56 10012 005767          TST     SHIFT           ; SHIFT OUT?
         000142
57 10016 001434          BEQ     TEXT6           ; NO- BRANCH
58 10020 012704          MOV     #16,R4          ; SHIFT-OUT
         000016
59 10024 042704 TEXT7:   BIC     #177740,R4      ; CLEAR ALL BUT 5 LO-ORDER BITS
         177740
60 10030 020427          CMP     R4,#17          ; BETTER NOT BE SHIFT-IN
         000017
61 10034 001010          BNE     TEXT13          ; NO- CONTINUE
62 10036 004767          JSR     PC,LDCHAR       ; PUT IN SHIFT IN
         000122
63 10042 012704          MOV     #40,R4          ; AND SPACE
         000040
64 10046 004767          JSR     PC,LDCHAR       ; PUT IT IN
         000112
65 10052 012704          MOV     #16,R4          ; AND SHIFT OUT
         000016
66 10056 004767 TEXT13:  JSR     PC,LDCHAR       ; INSERT IT
         000102
67 10062 112304          MOVB    (R3)+,R4        ; LOAD UP NEXT BYTE
68 10064 005301          DEC     R1              ; DECREMENT LENGTH+1
69 10066 002356          BGE     TEXT7           ; STORE BYTE IF NOT AT END
70 10070 012704          MOV     #17,R4          ; INSERT SHIFT-IN
         000017
71 10074 004767          JSR     PC,LDCHAR       ; INSERT IT
         000064
72 10100 005067 TEXT8:   CLR     SHIFT           ; CLEAR SHIFT-OUT FLAG
         000054
73 10104 012601 TEXT10:  MOV     (SP)+,R1        ; RESTORE R1
74 10106 000663          BR      TEXT0           ; GET NEXT ARG,
75 10110 112304 TEXT6:   MOVB    (R3)+,R4        ; GET CHAR,
76 10112 001403          BEQ     TEXT9A          ; LET NULLS SLIP THROUGH
77 10114 120427          CMPB    R4,#40          ; PRINTABLE
         000040
78 10120 002402          BLT     TEXT9           ; NO- DON'T PRINT IT
79 10122 004767 TEXT9A:  JSR     PC,LDCHAR       ; YES- INSERT IT
         000036
80 10126 005301 TEXT9:   DEC     R1              ; DECREMENT BYTE COUNT
81 10130 001367          BNE     TEXT6           ; CONTINUE IF MORE LEFT
82 10132 000764          BR      TEXT10          ; RESTORE R1 AND GET NEXT ARG
```



```
83 10134 032702 TEXT11:  BIT     #1,R2           ; FILLED EVEN NO. OF BYTES?
         000001
84 10140 001403          BEQ     TEXT12          ; YES- BRANCH
85 10142 005004          CLR     R4              ; MAKE LAST BYTE = 0
86 10144 004767          JSR     PC,LDCHAR       ; LOAD UP DPY FILE WITH LAST WORD
         000014
87 10150 004767 TEXT12:  JSR     PC,INSERT       ; INSERT INTO FILE
         170570
88 10154 000167          JMP     CKRPAR          ; AND EXIT
         170136
89
90 10160 000000 SHIFT:   .WORD   0               ; SHIFT-OUT FLAG
91 10162 000000 CBUFF:   .WORD   0               ; CHARACTER BUFFER
92
93               ;       ****************************************
94               ; LOAD CHARACTER ROUTINE
95
96 10164 110422 LDCHAR:  MOVB    R4,(R2)+        ; PUT BYTE INTO R0
97 10166 032702          BIT     #1,R2           ; ODD OR EVEN BYTE NEXT?
         000001
98 10172 001005          BNE     LDCH0           ; ODD- DON'T INSERT YET
99 10174 016700          MOV     CBUFF,R0        ; LOAD UP DPY WORD
         177762
100 0200 004767          JSR     PC,PUTWD        ; EVEN- CAN INSERT NOW
         170044
101 0204 005742          TST     -(R2)           ; POINT BACK AT CBUFF
102 0206 000207 LDCH0:   RETURN                  ; RETURN TO CALLER
103                      .IFDF   $CLOCK
104                      .IFDF   $DISK
```

```
                        .SBTTL  "TIME" AND "TIMR" ROUTINES
                        .ENDC
                ;       ****************************************

                        ; ROUTINE TO HANDLE:
                        ;       CALL "TIME"(A)
                        ; WHERE A IS THE NUMBER OF TICKS WANTED

 010210         TIME:
 10210 004767           JSR     PC,POLENT       ; ENTER POLISH, CK "("
       170054
 10214 000136'                  +GETONE         ; GET A
 10216 000006'                  +INTEGR         ; INTEGERIZE IT
 10220 010222'                  .+2             ; EXIT FROM POLISH
 10222 005767           TST     DSPTCH          ; GOT A CLOCK?
       000152
 10226 001055           BNE     TIME2           ; NO- EXIT
 10230 013746           MOV     @#4,-(SP)       ; SAVE LOC 4
       000004
 10234 012737           MOV     #HERE,@#4       ; MAY ADR
       010366'
       000004
 10242 005737           TST     @#177546        ; GOT ONE
       177546
 10246 012637 TIMM:     MOV     (SP)+,@#4       ; RESTORE LOC 4
       000004
 10252 005767           TST     DSPTCH          ; WELL
       000122
 10256 001041           BNE     TIME2           ; NO CLOCK: EXIT
 10260 016500           MOV     FAC2(R5),R0     ; GET COUNT IN TICKS
       000042
 10264 003002           BGT     .+6             ; CONTINUE IF POSITIVE
 10266 000167           JMP     ERRARG          ; NO- ERROR IF <=0
       000000G
 10272 010067           MOV     R0,TIMEC        ; SAVE TIME COUNT
       000156
 10276 005067           CLR     TMSW            ; ASSUME NO ONE ELSE USING CLOCK
       000156
 10302 012700           MOV     #100,R0         ; USE FOR LKV AND BIT 6
       000100
 10306 030067           BIT     R0,LKS          ; CLOCK INTERRUPT ENABLED?
       177546'
 10312 001414           BEQ     TIME0           ; NO- BRANCH
 10314 021027           CMP     (R0),#TMR       ; MY OLD REQUEST?
       010402'
 10320 001004           BNE     TIME1           ; NO- BRANCH
 10322 005767           TST     TMRADR          ; ANY ADR FROM LAST TIME?
       000130
 10326 001406           BEQ     TIME0           ; NO- BRANCH
 10330 000402           BR      .+6             ; YES- BRANCH
 10332 011067 TIME1:    MOV     (R0),TMRADR     ; YES- GET LKV
       000120
 10336 012767           MOV     #6,TMSW         ; SET UP PROPER EXIT
       000006
       000114
 10344 012760 TIME0:    MOV     #300,2(R0)      ; PUT IN BR6
       000300
```



```
       000002
 10352 012710           MOV     #TMR,(R0)       ; PUT ADR AT LKV
       010402'
 10356 010067           MOV     R0,LKS          ; ENABLE CLOCK
       177546'
 10362 000167 TIME2:    JMP     CKRPAR          ; CK ")" AND EXIT TO USER
       167730
 10366 012716 HERE:     MOV     #TIMM,(SP)      ; RET ADR
       010246'
 10372 010667           MOV     SP,DSPTCH       ; SIGNAL NO CLOCK
       000002
 10376 000002           RTI
 10400 000000 DSPTCH:   .WORD   0               ; NON-ZERO IF NO CLOCK

 10402 005767 TMR:      TST     TIMEC           ; TIME ELAPSED YET?
       000046
 10406 003410           BLE     TMR0            ; END OF LOOP?  BRANCH
 10410 005367           DEC     TIMEC           ; DECREMENT THE COUNT
       000040
 10414 005767           TST     TMSW            ; ANYBODY ELSE?
       000040
 10420 001001           BNE     .+4             ; YES
 10422 000002           RTI                     ; NO- CONTINUE
 10424 000177           JMP     @TMRADR         ; GO TO HIM
       000026
 10430 066767 TMR0:     ADD     TMSW,PC         ; AND DISPATCH TO EXIT
       000024
 10434 005067           CLR     LKS             ; CLEAR CLOCK STATUS REGISTER
       177546'
 10440 000002           RTI
 10442 016767           MOV     TMRADR,LKV      ; GIVE BACK VECTOR
       000010
       000100'
 10450 000177           JMP     @LKV            ; AND CONTINUE THERE
       000100'

 10454 000000 TIMEC:    .WORD   0               ; NUMBER OF TICKS
 10456 000000 TMRADR:   .WORD   0               ; OLD CONTENTS OF LKV
 10460 000000 TMSW:     .WORD   0               ; ENDING TYPE ( 0 OR 6)


                ;       ****************************************

                        ; ROUTINE TO HANDLE:
                        ;       CALL "TIMR"(E)
                        ; RETURNS THE CURRENT VALUE OF TIMER IN E

 10462 004767 TIMR:     JSR     PC,POLENT       ; ENTER POLISH, CK"("
       167602
 10466 010470'                  .+2             ; EXIT POLISH
 10470 004767           JSR     PC,GETADR       ; GET ADR OF E IN R2
       171342
 10474 005022           CLR     (R2)+           ; ZERO HI-ORDER WORD
 10476 016722           MOV     TIMEC,(R2)+     ; SAVE CURRENT # OF TICKS
       177752
 10502 000167           JMP     CKRPAR          ; CK ")" AND EXIT TO USER
       167610
```

```
76                      .ENDC
77                      .IFDF   $DISK
```



```
1                 .SBTTL  DISPLAY FILE BEGINING
2                 .ENDC
3                 ;       ***************************************
4
5                         ; THIS PORTION MUST BE LINKED BOTH TO THE RT-11
6                         ; SCROLL BUFFER (IF THERE IS ONE) AND THE REST OF THE
7                         ; USER'S DISPLAY FILE.
8
9 010506          BEG:
10 10506 160000           DJMP                    ; THE START
11 10510 010576'                  +RSTFIL         ; REST OF FILE IF TRAK OFF
12 10512 010520'                  +TAG2           ; ADR FOR TRAK SUBP
13 10514 177777 TAG1:     -1                      ; TAG FOR TRAK ROUTINE
14 10516 000000           .WORD 0                 ; NEXT TAG POINTER
15 10520 170200 TAG2:.    .WORD   170200          ; LD, STATUS REG, A
16 10522 117124           .WORD   117124          ; APNT, I=4, T=0
17 10524 000000 XT:       .WORD   0               ; X POSITION
18 10526 000000 YT:       .WORD   0               ; Y POSITION OF TRAK
19 10530 104140           .WORD   104140  ; SHORT VECTOR LP ENABLE
20 10532 057600           .WORD    57600
21 10534 077677           .WORD    77677
22 10536 040177           .WORD    40177
23 10540 040177           .WORD    40177
24 10542 077677           .WORD    77677
25 10544 057600           .WORD    57600
26 10546 040077           .WORD    40077
27 10550 077777           .WORD    77777
28 10552 057777           .WORD    57777
29 10554 057677           .WORD    57677
30 10556 074000           .WORD    74000
31 10560 040017           .WORD    40017
32 10562 067400           .WORD    67400
33 10564 040136           .WORD    40136
34 10566 047400           .WORD    47400
35 10570 040017           .WORD    40017
36 10572 173400           .WORD   SDINT    ; END OF TRAK SUBPICTURE
37 10574 000000           .WORD        0
38 10576 117124 RSTFIL:   .WORD   117124          ; APNT L=OFF,I=4, F=OFF, T=0
39 10600 000000           .WORD   0,0             ;       AT (0,0)
   10602 000000
40 10604 170240           .WORD   170240          ; LD STAT REG A, NORMAL FONT, LP INT
41 10606 160000           DJMP                    ; DJMP TO START OF FILE
42 10610 010506'DSTOP:    .WORD   BEG             ; INITIALLY TO TRAK OBJ
43
44 10612 000000 DCRASH:   .WORD   0               ; LAST WORD IN DPY FILE (EXCLUDING TEXT)
45       000001'          .END
```

SYMBOL TABLE

```
ABS,F   003640RG        ACTEND  000640RG        ACTST   000642RG
ADBAS   001334R         ADRARG  002240R         ADRCHK  006404R
ADRCH0  006432R         ADRCH1  006424R         ADRCH2  006430R
ADRCH3  006456R         ADRSYN  002244R         ADSTK   001362RG
AGET    006666RG        AGETF   006752R         AGETX   006744R
AGETY   007004R         AGET0   006710R         AGET1   007010R
AGET2   006772R         AGET3   006704R         AGET4   007036R
APNT    004120RG        APUT    006462RG        APUTF   006576R
APUTF0  006634R         APUTX   006552R         APUTY   006564R
APUT1   006530R         APUT2   006646R         ARGGET  001766R
ARRAYS= 000010          BEG     010506RG        BITGET  003214R
BITNEG  003230R         BITZER  003234R         BUFPOL  000204R
BUFTST  000210RG        BUFT0   000226R         CBUFF   010162R
CKEOL   000324RG        CKLPAR  000276R         CKLSYN  000306R
CKOPCM  001702R         CKOPC0  001734R         CKOSYN  001740R
CKRPAR  000316R         CKRPA0  000334R         CONT    002450RG
CONTA   002454RG        CONT0   002470R         DCRASH  010612RG
DFOERR  000664R         DISEND  000636RG        DISINT  000466R
DISX    = ****** G      DISY    = ****** G      DIS0    000506R
DIVVEC  004504R         DJMP    = 160000        DPC     = ****** G
DSPTCH  010400R         DSR     = ****** G      DSTOP   010610RG
DSTP    002360RG        ELOOP   000040R         ELOOP0  000052R
ENTERP  000156R         ERA$    001456RG        ERA$0   001502R
ERA$1   001522R         ERA$2   001526R         ERA$3   001572R
ERA$4   001576R         ERA$5   001554R         ERRAOR  004534R
ERRARG= ****** G        ERRDIV  004442R         ERRSYN= ****** G
ERSS3   002226R         ERSTAR  004434R         ESUB    001250RG
ESXIT   001330R         EVAL    = ****** G      FAC1    = 000040
FAC2    = 000042        FIGARG  007322R         FIGR    007042RG
FIGRLP  007216R         FIGR1   007254R         FIGR2   007302R
FIGSYN  007326R         FIX     003032RG        FIXSP   = ****** G
FIX0    = ****** G      FLOAT   = ****** G      FPPRES  000442R
FPPSAV  000416R         FPUT    007332RG        FPUT0   007402R
FPUT1   007446R         FPUT2   007462R         FPUT3   007524R
FPUT4   007574R         FPUT5   007562R         FX      006374R
FXL     006400R         FY      006364R         FYL     006370R
GETADR  002036R         GETAD0  002234R         GETAD1  002250R
GETAD2  002332R         GETAD3  002336R         GETAD5  002344R
GETAD6  002310R         GETAD7  002310R         GETARG  000152R
GETNUM  000076R         GETONE  000136R         GETOUT  000332R
GETSAV  002026R         GETSYN  000146R         GETVAR= ****** G
GRAARG  005372R         GRALP   005576R         GRALP1  005606R
GRASYN  005366R         GRA,0   005156R         GRA,1   005232R
GRA,3   005242R         GRA0    005300R         GRA1    005302R
GRA10   005504R         GRA2    005314R         GRA3    005376R
GRA4    005630R         GRA5    005642R         GRA6    005472R
GRA7    005514R         GRA7A   005532R         GRA9    005550R
GTVECT= ****** G        HERE    010366R         HIFREE= 000012
IFX     006354R         IFY     006344R         INIT    002500RG
INITA   002504RG        INIT1   002616R         INIT2   002606R
INIT3   002670R         INSERT  000744R         INSRTF  000634RG
INSXIT  001004R         INT     = ****** G      INTEGR  000006R
INTEG0  000022R         INVIS   003236R         LCLOOP  002162R
LOCHAR  010164R         LOCH0   010206R         LIFT    003050R
LIFT0   003126R         LIFT1   003174R         LIFT2   003152R
LKS     = 177546        LKV     = 000100        LLX     006360R
LLY     006350R         LOCNO2  002210R         LOFREE= 000014
```


SYMBOL TABLE

```
LOOPCT  000054R         LPADR   000056R         LPEN    001602RG
LPENT   001666R         LPENX   001670R         LPENY   001674R
LPEN,F  001700R         LPEN0   001622R         LPEX1   003620R
LPINT0  003272R         LPINT1  003310R         LPINT2  003412R
LPINT3  003524R         LPINT4  003554R         LPINT5  003570R
LPNINT  003240R         LPS,F   005574R         MODE    005674R
MSG     = ****** G      MULVEC  006234R         M1      004642R
M2      004644R         NARRAY= ****** G        NEGSTK  000176R
NEWPUT  000644R         NOSC    002714RG        NSBAS   001364R
NSTERR  001146R         NSTSK   001412RG        NUMOUT= ****** G
OFF     003024RG        OLDMOD  001764RG        ON      002724RG
ON0     002750R         ON1     002730R         ON2     003016R
ON3     003012R         OPARGF  000074R         OPTSTR  006316R
OPTST0  006342R         PC      =%000007        PLOOP   000026R
POLENT  000270R         POLRET= 000134          POP     000256R
POSX    004514R         POSY    004522R         POS0    004526R
POS1    004532R         PR4     = 000200        PR7     = 000340
PSHPTR  006212R         PSHXIT  004640R         PSW     = 177776
PUSH    000234R         PUSHF   004574R         PUSH1   000246R
PUTSGM  001744R         PUTS0   001762R         PUTWD   000650R
PUT1    000724R         PUT2    000732R         PUT3    000714R
PUT5    000652R         PUT6    000700R         PUT7    000672R
QIKPOL  000312R         RDOT    003732RG        RDOT0   004066R
RDOT1   004024R         RELABS  004456R         RELAB0  004466R
RESTR   002472R         RETURN= 000207          RETURP  000166R
REVRSE  004546R         RSTFIL  010576R         RUN,F   002446R
R0      =%000000        R0SAVE= 000044          R1      =%000001
R2      =%000002        R3      =%000003        R4      =%000004
R5      =%000005        SAVER0  001452R         SAVER1  004542R
SAVINT  000000R         SCAL    005700RG        SCALEX  004770R
SCALEY  005042R         SCAL,F  002712R         SCAL0   006200R
SCLGEN  006244R         SCLSAV  006314R         SCL,X   006402RG
SCL,Y   006372RG        SCPTR   006210R         SCUSC0  004360R
SCUSC1  004432R         SC,USC  004320RG        SDINT   = 173400
SDXIT   004200R         SDXIT0  004204R         SDXIT1  005660R
SETNEC  000060R         SETOPT  000066R         SHIFT   010160R
SKPARG  000340R         SKPA0   000412R         SKPA1   000366R
SKPA2   000400R         SKPA3   000344R         SKPA4   000356R
SP      =%000006        SS1SAV= 000024          SS2SAV= 000026
SS2TMP  005676R         STAT    003642RG        STOPA   002364RG
STOP8   002436R         STOPC   002420R         STOP,F  002444R
STOVAR= ****** G        SUBP    001006RG        SUBP0   001040R
SUBP1   001034R         SUBP2   001154R         SUBP3   001174R
SUB0    000552R         SUB1    000544R         SUB2    000572R
TABLE   = ****** G      TAGEND  001436RG        TAGMED  001434R
TAGSRH  001414RG        TAGXIT  001432R         TAG1    010514RG
TAG2    010520R         TEN24   006224R         TEXT    007626RG
TEXT0   007656R         TEXT1   007676R         TEXT10  010104R
TEXT11  010134R         TEXT12  010150R         TEXT13  010056R
TEXT2   007766R         TEXT3   007736R         TEXT5   007754R
TEXT6   010110R         TEXT7   010024R         TEXT8   010100R
TEXT9   010126R         TEXT9A  010122R         TGLOOP  001420R
TIME    010210RG        TIMEC   010454R         TIME0   010344R
TIME1   010332R         TIME2   010362R         TIMM    010246R
TIMR    010462RG        TMR     010402R         TMRADR  010456R
TMR0    010430R         TMSW    010460R         TRAK    004646RG
TRAKX   004764R         TRAKY   004766R         TST64   001440R
```

```
GTB  V01-01 BASIC - GT ROOT MOD RT-11 MACRO VM02-09   16-OCT-74 02:06:07 PAGE 48+
SYMBOL TABLE

T0      006736R         T1      006544R         UNDO    007602R
UNDO0   007624R         USCL,X  006362R         USCL,Y  006352R
USRARE= ****** G        VARSAV= 000022          VECT    004214RG
VECT0   004252R         WD1     000740RG        WD2     000742RG
XGRA    005124RG        XSCL0   005022R         XSCL1   005040R
XSCL2   005016R         XSIGN   002032R         XT      010524R
XYBIG   002030R         YGRA    005142RG        YSCL0   005072R
YSCL1   005110R         YSCL2   005122R         YSIGN   002034R
YT      010526R         ZERNUM  000124R         $ADC  = 000000
$ADR  = ****** G        $CLK  = 000000          $CLOCK= 000000
$DEPTH= 000012          $DIO  = 000000          $DIS  = 000000
$DVR  = ****** G        $ERVEC= ****** G        $IR   = ****** G
$LPS  = 000000          $MLR  = ****** G        $SBR  = ****** G
$STRNG= 000000          $VT11 = 000000          .COMMA= ****** G
.EOL  = ****** G        .LPAR = ****** G        .RPAR = ****** G

. ABS.  000000     000
        010614     001
ERRORS DETECTED: 0
FREE CORE: 17006. WORDS

.LP:=PERPAR,GTL,GTB1,GTB2,GTB3,GTB4
```

```
PERPAR -- PERIPHERAL SUPPORT PA RT-11 MACRO VM02-09   16-OCT-74 02:14:36 PAGE 1

1                ;TITLE PERPAR -- PERIPHERAL SUPPORT PACKAGE PARAMETER MODULE.
2                ;
3                ; DEC-11-LBPAA-A-LA     BASIC KERNEL V02-01
4                ;
5                ; COPYRIGHT (C) 1974
6                ;
7                ; DIGITAL EQUIPMENT CORPORATION
8                ; MAYNARD, MASSACHUSETTS 01754
9                ;
10               ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
11               ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
12               ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
13               ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
14               ; MAY APPEAR IN THIS DOCUMENT.
15               ;
16               ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
17               ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
18               ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
19               ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
20               ; OTHERWISE BE PROVIDED IN WRITING BY DEC.
21               ;
22               ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
23               ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
24               ; WHICH IS NOT SUPPLIED BY DEC.
25
26               ; THE CONDITIONALS CONTAINED IN THIS MODULE AFFECT THE ASSEMBLY
27               ; OF THE FUNCTION TABLE MODULE "FTBL.MAC".
28               ; TO OBTAIN THE DESIRED CONDITIONAL DEFINITION(S),
29               ; REMOVE (USING AN EDITOR) THE
30               ; SEMI-COLON APPEARING BEFORE THE CONDITIONAL.
31               ;$DISK=0                ;DEFINE FOR RT-11
32                       .IFNDF $DISK
33       000000 $STRNG=0                 ;DO NOT DEFINE FOR PTS BASIC WITHOUT
34                                       ;STRINGS,= DEFINED FOR PTS V01 WITH STRINGS
35                       .ENDC           ;
36
37       000000 $LPS=0                   ;DEFINE FOR LPS
38                       .IFDF $LPS
39               ;$V=0                   ;DEFINE FOR LPS WITH VECTORS STARTING
40               ;                       ;    AT 300.  DEFAULT SETTING IS VECTORS AT
41               ;                       ;    340. SET $V = ANY OTHER DISPLACEMENT IF
42               ;                       ;       VECTORS START AT DISPLACEMENTS
43               ;                       ;       OTHER THAN 0 OR 40 FROM
44               ;                       ;       VECTOR 300
45
46       000000 $ADC=0                   ;INCLUDE A/D ROUTINES.
47       000000 $CLK=0                   ;INCLUDE CLOCK ROUTINES.
48       000000 $DIO=0                   ;INCLUDE DIGITAL IO ROUTINES
49       000000 $DIS=0                   ;INCLUDE DISPLAY ROUTINES.
50                       .ENDC   ;$LPS
51               ;
52               ;
53       000000 $VT11=0                  ;FOR GT40 (GT44)
54               ;
55               ;
56
57
```

90

```
                        .IFDF   $VT11
000000 $CLOCK=0                         ;FOR SYSTEM CLOCK (KW11L)
                        .ENDC

                        .EOT
                .TITLE  GTC  V01-01  BASIC - GT  DISK OVERLAY MODULE

                ;
                ;       DEC-11-L8PGC-A-LA       BASIC KERNEL V02-01
                ;
                ; THE INFORMATION IN THIS LISTING IS SUBJECT TO CHANGE WITHOUT NOTICE
                ; AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
                ; CORPORATION, DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY
                ; FOR ANY ERRORS THAT MAY APPEAR IN THIS LISTING.
                ;
                ; THIS SOFTWARE IS FURNISHED TO THE PURCHASER UNDER A LICENSE
                ; FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH
                ; INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN
                ; SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY
                ; DIGITAL.
                ;
                ; DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE
                ; USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT
                ; SUPPLIED BY DIGITAL.
                ;
                ; COPYRIGHT (C) 1973,1974, BY DIGITAL EQUIPMENT CORPORATION.
                ;
                ;       AUTHOR: DR. STEPHEN R. ALPERT   AUGUST 1973
                        .IFDF   $DISK
                        .SBTTL  GLOBALS,EQUATES
                        .ENDC


                ;       REGISTER ASSIGNMENTS
000000          R0=%0
000001          R1=%1
000002          R2=%2
000003          R3=%3
000004          R4=%4
000005          R5=%5
000006          SP=%6
000007          PC=%7

                ;       GLOBALS TO COMMUNICATE WITH ROOT SECTION
                        .GLOBL  SAVE,FIXSP,FIX0,FREE,BUFTST
                        .IFDF   $DISK
                        .MCALL  .DSTATUS,.LOCK,.UNLOCK,.CLOSE,.LOOKUP,.SRESET
                        .MCALL  .ENTER,.READW,.WRITW,.FETCH,.CSISPC
                        .GLOBL  RSTR,FREEGET,FREE0
                        .GLOBL  SAV24,SAVERT
                        .ENDC
                        .IFNDF  $VTONLY
                        .GLOBL  RTSON,NARRAY,TABLE
                        .ENDC
                        .IFDF   $DISK
                        .GLOBL  IGNORE          ; FROM BASICR
                        .ENDC
```



```
                        .GLOBL  MSG,,LPAR,,RPAR,,EOL,ERRSYN
                        .GLOBL  NUMOUT,ERRARG,CKEOL
                        .GLOBL  NSTSK,ESUB,INSRTF,STOPA,TAG1,ACTST
                        .GLOBL  ACTEND,BEG,WD1,WD2,DSTOP,OLDMOD
                        .GLOBL  DISEND,CONTA,TAGEND,ADSTK,DCRASH,WD1,WD2
                        .GLOBL  CONT

                ;       EQUATES
000207          RETURN  = 207           ; RTS PC
177776          PSW     =-2             ; PROCESSOR STATUS WORD
160000          DJMP    = 160000        ; A DISPLAY JUMP
173400          SDINT   = 173400        ; A STOP DISPLAY AND INTERRUPT

                ;       EQUATES FROM BASIC
000040          FAC1=   40
000042          FAC2=   42
000010          ARRAYS= 10
000012          HIFREE= 12
000014          LOFREE= 14
000074          HISTR=  74
000072          LOSTR=  72
000004          POL=    4
000076          BUFCHN= 76
                        .IFDF   $DISK
```

```
1                       .SBTTL  "SAVE" ROUTINE
2                       .ENDC
3               ;       ******************************************
4
5                       ; ROUTINE TO HANDLE:
6                       ;       CALL "SAVE" [( <FILE DESCRIPTOR> )]
7
8                       ; MUST HAVE EXTRA WORD BEFORE START OF EACH SUBPICTURE
9                       ; REGISTER USAGE:
10                      ;       R0 = "GET" WORD POINTER
11                      ;       R1 = ADR STACK POINTER (INITIALLY #ADSTK)
12                      ;       R2,R3 = FAST STORAGE
13                      ;       R4 = "PUT" WORD POINTER
14                      ;       R5 = BACKWARD POILTER FOR TAG LINKED LIST
15
16 00000        SAVE:
17 00000 010146         MOV     R1,-(SP)        ; SAVE R1
18 00002 010546         MOV     R5,-(SP)        ; SAVE R5
19 00004 005777 SAVE0:  TST     @NSTSK          ; CURRENTLY BUILDING A SUBPICTURE?
         000000G
20 00010 001403         BEQ     SAV0            ; NO= BRANCH
21                      .IFDF   $CRASH
22                      TRAP    0
23                      .ASCII  /STILL IN SUBPICTURE/
24                      .BYTE   0
25                      .EVEN
26                      .ENDC                   ; $CRASH
27                      .IFNDF  $CRASH
28 00012 004767         JSR     PC,ESUB         ; PUT IN AN EXIT
         000000G
29 00016 000772         BR      SAVE0           ; TEST IF ENOUGH
30                      .ENDC
31 00020 005767 SAV0:   TST     INSRTF          ; INSERT PENDING?
         000000G
32 00024 001375         BNE     SAV0            ; YES= LOOP UNTIL FREE
33                ;  INITIALIZE
34 00026 004767         JSR     PC,STOPA        ; STOP THE DISPLAY
         000000G
35 00032 012700         MOV     #TAG1+2,R0      ; GET PTR TO 1=ST SUBP
         000002G
36 00036 010005         MOV     R0,R5           ; SAVE FOR LATER USE
37 00040 011000 SAV1:   MOV     (R0),R0         ; GET POINTER
38 00042 001405         BEQ     SAV2            ; BRANCH IF ALL SUBP DONE
39 00044 014002         MOV     -(R0),R2        ; GET ADR OF SUBP
40 00046 010203         MOV     R2,R3           ; PUT ADR IN R3 ALSO
41 00050 010243         MOV     R2,-(R3)        ; PUT ADR OF SUBP BEFORE SUBP
42 00052 022020         CMP     (R0)+,(R0)+     ; MOVE R0 TO NEXT TAG
43 00054 000771         BR      SAV1            ; AND LOOP
44                ;  SET UP FOR COMPRESSION OF DPY FILE
45 00056 005015 SAV2:   CLR     (R5)            ; ZERO WORD FOR TAG SEARCH
46 00060 016700         MOV     ACTST,R0        ; "GET" PTR
         000000G
47 00064 010004         MOV     R0,R4           ; "PUT" PTR
48 00066 010546         MOV     R5,-(SP)        ; WRITE ENABLE
49 00070 012701         MOV     #ADSTK,R1       ; REST OF FILE ADR STACK
         000000G
50 00074 020067 SAV3:   CMP     R0,DISEND       ; DONE YET?
```



```
         000000G
51 00100 103154         BHIS    SAV5            ; YES= BRANCH
52 00102 012003         MOV     (R0)+,R3        ; GET WORD
53 00104 100407         BMI     SAV4            ; DPU INST.= BRANCH
54 00106 012746 SAVCON: MOV     #SAV3,-(SP)     ; FAKE A JSR PC
         000074'
55 00112 005766 SAVSET: TST     2(SP)           ; WRITE ENABLED?
         000002
56 00116 001401         BEQ     .+4             ; NO= BRANCH
57 00120 010324         MOV     R3,(R4)+        ; YES= PUT WORD
58 00122 000207         RETURN                  ; EXIT
59                ;  PROCESS DPU INSTRUCTION (SOME OF THEM)
60 00124 020327 SAV4:   CMP     R3,#DJMP        ; A DISPLAY JUMP?
         160000
61 00130 001406         BEQ     SAV6            ; YES= BRANCH
62 00132 020327         CMP     R3,#DJMP+1      ; A JUMP FROM SUBP (OFF)?
         160001
63 00136 001403         BEQ     SAV6            ; YES= CONTINUE
64 00140 020327         CMP     R3,#SDINT       ; A STOP DISPLAY?
         173400
65 00144 001077         BNE     SAV7            ; NO= BRANCH
66                ;  HERE IF SDINT OR DJMP
67 00146 005720 SAV6:   TST     (R0)+           ; NEXT WORD ZERO?
68 00150 001467         BEQ     SAV8            ; YES= (SDINT/0 CASE)= BRANCH
69 00152 021060         CMP     (R0),6(R0)      ; ACTIVE SUBP FOLLOWS?
         000006
70 00156 001023         BNE     SAV9            ; NO= BRANCH
71 00160 005716         TST     (SP)            ; WRITE ENABLED?
72 00162 001002         BNE     SAV10           ; YES= BRANCH
73 00164 012703         MOV     #DJMP,R3        ; CHANGE SDINT TO DJMP
         160000
74 00170 010546 SAV10:  MOV     R5,-(SP)        ; WRITE ENABLE
75 00172 010324         MOV     R3,(R4)+        ; INSERT SDINT OR DJMP
76 00174 010441         MOV     R4,-(R1)        ; SAVE PTR TO "REST OF FILE ADR"
77 00176 022024         CMP     (R0)+,(R4)+     ; R0 TO TAG, R4 TO SUBP ADR
78                ;  INSERT ADR OF SUBP IF FOLLOWING
79 00200 010414         MOV     R4,(R4)         ; OFFSET FOR NEW ADDRESS
80 00202 062724         ADD     #10,(R4)+       ; ADD TO POINT TO CORRECT WORD
         000010
81 00206 011014 SAV18:  MOV     (R0),(R4)       ; MOVE TAG
82 00210 022020         CMP     (R0)+,(R0)+     ; POINT "GET" AT SUBP-2
83 00212 010415 SAV11:  MOV     R4,(R5)         ; SET NEW LINK IN TAG CHAIN
84 00214 005724         TST     (R4)+           ; POINT TO "NEXT TAG PTR"
85 00216 010405         MOV     R4,R5           ; SAVE NEW BACK PTR
86 00220 005024         CLR     (R4)+           ; ZERO PTR TO STOP SEARCH
87 00222 012024         MOV     (R0)+,(R4)+     ; MOVE ADDRESS BEFORE SUBP
88 00224 000723         BR      SAV3            ; AND CONTINUE
89                ;  HERE IF AN ACTIVE SUBPICTURE DOESN'T FOLLOW SDINT OR DJMP
90 00226 005716 SAV9:   TST     (SP)            ; WRITE ENABLE?
91 00230 001003         BNE     SAV12           ; YES= BRANCH
92 00232 062700 SAV14:  ADD     #6,R0           ; MOVE R0
         000006
93 00236 000716         BR      SAV3            ; AND CONTINUE
94 00240 020327 SAV12:  CMP     R3,#DJMP        ; A DISPLAY JUMP?
         160000
95 00244 001004         BNE     SAV13           ; NO= BRANCH
96 00246 021000         CMP     (R0),R0         ; SUBP FOLLOWS?
```

```
97 00250 103770          BLO     SAV14           ; NO= BRANCH
98 00252 005046          CLR     -(SP)           ; YES= DISABLE WRITE
99 00254 000766          BR      SAV14           ; AND CONTINUE
100                ; HERE IF SDINT/ SUBP BACK
101 0256 010324 SAV13:   MOV     R3,(R4)+        ; SET SDINT
102 0260 010414          MOV     R4,(R4)         ; PUT IN CURRENT POSITION
103 0262 062724          ADD     #10,(R4)+       ; OFFSET FOR ADDRESS
         000010
104 0266 012014          MOV     (R0)+,(R4)      ; MOVE OLD SUBP ADR
105 0270 012702          MOV     #TAG1,R2        ; GET ADR OF 1=ST POINTER
         000000G
106 0274 016202 SAV15:   MOV     2(R2),R2        ; GET ADR
         000002
107 0300 001406          BEQ     SAV16           ; BRANCH IF NOT ACTIVE
108 0302 014203          MOV     -(R2),R3        ; GET NEW SUBP ADR
109 0304 026314          CMP     -2(R3),(R4)     ; A MATCH?
         177776
110 0310 001405          BEQ     SAV17           ; YES= FOUND SUBP= BRANCH
111 0312 005722          TST     (R2)+           ; POINT R2 AT NEXT PTR
112 0314 000767          BR      SAV15           ; NO= LOOP
113                ; HERE IF SDINT/ WENABL/ SUBP NOT ACTIVE
114 0316 024444 SAV16:   CMP     -(R4),-(R4)     ; POINT TO SDINT
115 0320 005726          TST     (SP)+           ; CLEAN STK AND ADR=STK
116 0322 000743          BR      SAV14           ; AND CONTINUE
117                ; !!! I DON'T THINK THAT SAV16 WILL BE REACHED
118                ; HERE IF FOUND SUBP
119 0324 010324 SAV17:   MOV     R3,(R4)+        ; SET NEW ADR
120 0326 000727          BR      SAV18           ; SET TAG AND CONTINUE
121                ; HERE IF SDINT/0 CASE
122 0330 005726 SAV8:    TST     (SP)+           ; WRITE ENABLED?
123 0332 001660          BEQ     SAV3            ; NO= CONTINUE
124 0334 010324          MOV     R3,(R4)+        ; INSERT SDINT
125 0336 005024          CLR     (R4)+           ; FOLLOWED BY ZERO
126 0340 010431          MOV     R4,@(R1)+       ; FIX ADR OF REST OF FILE
127 0342 000654          BR      SAV3            ; AND CONTINUE
128                ; HERE IF DPU INST, NOT SDINT OR DJMP
129 0344 010302 SAV7:    MOV     R3,R2           ; GET WORD
130 0346 042702          BIC     #3777,R2        ; LEAVE BITS 15-11
         003777
131 0352 020227          CMP     R2,#110000      ; LONG VECTOR?
         110000
132 0356 001421          BEQ     SAV19           ; YES= BRANCH
133 0360 020227          CMP     R2,#120000      ; XGRA?
         120000
134 0364 103650          BLO     SAVCON          ; NO= BRANCH
135 0366 020227          CMP     R2,#124000      ; YGRA?
         124000
136 0372 101245          BHI     SAVCON          ; NO= BRANCH
137                ; HERE IF XGRA OR YGRA
138 0374 005720          TST     (R0)+           ; ADD TWO TO R0
139 0376 022020 SAV23A:  CMP     (R0)+,(R0)+     ; ADD 4 TO R0
140 0400 012003 SAV23:   MOV     (R0)+,R3        ; GET ADR OF SS1
141 0402 012023          MOV     (R0)+,(R3)+     ; STORE SS1 IN SYMTAB
142                      .IFNDF  $VTONLY
143 0404 002003          BGE     .+10
144 0406 005343          DEC     -(R3)
145 0410 005413          NEG     (R3)
```



```
146 0412 006223          ASR     (R3)+
147                      .ENDC
148 0414 012713          MOV     #-1,(R3)        ; RESTORE SS2
         177777
149 0420 000625          BR      SAV3            ; AND CONTINUE
150                ; HERE IF LONG VECTOR (MAYBE FIGR)
151 0422 021027 SAV19:   CMP     (R0),#DJMP      ; NEXT WORD DJMP FOR FIGR
         160000
152 0426 001227          BNE     SAVCON          ; NO= SAVE WORD AND CONTINUE
153 0430 000762          BR      SAV23A          ; AND CONTINUE
154                ; HERE IF ALL DONE!
155 0432 005726 SAV5:    TST     (SP)+           ; CLEAN OFF OLD WRITE FLAG
156 0434 010467          MOV     R4,DISEND       ; SET NEW DISEND
         000000G
157 0440 012024          MOV     (R0)+,(R4)+     ; PUT IN OLD DJMP
158 0442 010467          MOV     R4,ACTEND       ; SET NEW ACTEND
         000000G
159 0446 012024          MOV     (R0)+,(R4)+     ; PUT IN RETURN ADDRESS
160 0450 012605          MOV     (SP)+,R5        ; RESTORE R5
161 0452 012601          MOV     (SP)+,R1        ; RESTORE R1
162 0454 016702          MOV     ACTST,R2        ; GET START OF DPY FILE
         000000G
163 0460 010204          MOV     R2,R4           ; SAVE IN R4 ALSO
164 0462 016703          MOV     TAG1+2,R3       ; GET ADR OF FIRST SUBP TAG
         000002G
165 0466 010342          MOV     R3,-(R2)        ; PUT BEFORE DPY FILE
166 0470 001425          BEQ     SAV21           ; NO SUBP= BRANCH
167 0472 004767          JSR     PC,ADRSET       ; OFFSET ADR
         000036
168 0476 010302 SAV20:   MOV     R3,R2           ; GET NEXT POINTER
169 0500 001421          BEQ     SAV21           ; NO MORE= BRANCH
170 0502 024242          CMP     -(R2),-(R2)     ; POINT AT REST OF FILE ADR
171 0504 004767          JSR     PC,ADRSET       ; OFFSET ADR
         000024
172 0510 011200          MOV     (R2),R0         ; GET ADR OF SUBP
173 0512 005060          CLR     -2(R0)          ; CLEAR WORD BEFORE SUBP
         177776
174 0516 160400          SUB     R4,R0           ; MAKE OFFSET ADR
175 0520 010022          MOV     R0,(R2)+        ; STORE ADR
176 0522 005722          TST     (R2)+           ; BUMP BY TAG
177 0524 011203          MOV     (R2),R3         ; GET ADR OF NEXT TAG
178 0526 001406          BEQ     SAV21           ; NO MORE= BRANCH
179 0530 012746          MOV     #SAV20,-(SP)    ; FAKE JSR PC
         000476'
180 0534 011200 ADRSET:  MOV     (R2),R0         ; GET ADR
181 0536 160400          SUB     R4,R0           ; SUBTRACT ACTST
182 0540 010022          MOV     R0,(R2)+        ; SAVE OFFSET ADR
183 0542 000207          RETURN
184 0544 010400 SAV21:   MOV     R4,R0           ; ACTST IN R0
185 0546 005740          TST     -(R0)           ; MOVE ONE WORD BACK
186 0550 016740          MOV     ACTEND,-(R0)    ; PUT END IN PRECEDING WORD
         000000G
187 0554 160410          SUB     R4,(R0)         ; MINUS LO END
188 0556 006210          ASR     (R0)            ; CONVERT TO WORDS
189 0560 005210          INC     (R0)            ; GET CORRECT NO OF WORDS
190                ; ************************
191                      .IFDF   $DISK
```

```
192                     CMPB    (R1)+,#,LPAR    ; LEFT PAREN?
193                     BEQ     SAV22           ; YES- DO FILE I/O
194                     DEC     R1              ; POINT AT TOKEN
195                     MOV     R1,R1SVE        ; SAVE R1
196                     JMP     SAV29           ; YES- FINISH UP
197             SAV22:  MOV     R0,WD1          ; SAVE START ADR OF BUFFER
198                     MOV     (R0),R0         ; GET WORD COUNT MINUS 2
199                     ADD     #2,R0           ; MAKE = N
200                     MOV     R0,WD2          ; SAVE WORD COUNT
201                     CLR     -(SP)           ; SIGNIFY WRITE OPERATION
202                     JMP     SAVERT          ; IN ROOT SECTION
203             ;!!!!!!!SAVERT IS IN ROOT SO OVERLAY WORKS
204             SAV24:  MOV     (SP)+,R0        ; POINTER TO STRING
205                     MOV     R1,R1SVE        ; SAVE R1 FOR EXIT
206                     CLR     R1
207                     BISB    (R0)+,R1        ; GET LENGTH
208                     ADD     #2,R0           ; START OF ASCII STRING
209                     ADD     R1,R0           ; POINT AT END +1
210                     CLRB    (R0)            ; FOLLOWED BY 0 BYTE
211                     MOV     R0,-(SP)        ; SAVE PTR TO END
212                     SUB     R1,R0           ; POINT AT BEGINNING
213                     .CSISPC #FILE,#DEFEXT,R0
214                     BCC     .+6             ; NO- ERROR
215                     JMP     ERRARG          ; ELSE ERROR
216                     TST     (SP)+           ; BETTER NOT HAVE SWTICHES
217                     BNE     .-6             ; YES- ERROR
218                     MOVB    R1,@(SP)+       ; RESTORE BYTE COUNT
219                     MOV     #FILE+30.,R0    ; ADR OF RAD 50 NAME
220                     MOV     R0,R2           ; SAVE IN R2
221                     .DSTATUS        #DSTAT  ; R0 ==> DEV:
222                     BCS     DERR0           ; INVALID DEVICE
223                     MOV     #DSTAT+2,R0     ; POINT AT DEVBLK+1 WORD
224                     MOV     (R0)+,R3        ; GET HANDLER LENGTH
225                     TST     (R0)            ; HANDLER LOADED?
226                     BNE     SAV24A          ; YES- BRANCH
227                     MOV     R3,R0           ; GET ITS LENGTH
228                     JSR     PC,FREEGET      ; ASK FOR ROOM
229                     BCS     SAV41           ; NOT ENOUGH AVAILABLE
230                     MOV     R0,-(SP)        ; SAVE R0 ADR OF HANDLER
231                     MOV     R2,R0           ; R0 ==> DEV:
232                     EMT     343             ; FETCH HANDLER
233                     BCS     DERR0           ; BAD FETCH
234             SAV24A: TST     (SP)+           ; READ OR WRITE?
235                     BEQ     SAV40           ; WRITE- BRANCH
236                     JMP     RSTR1           ; READ- BRANCH
237             SAV41:  JSR     R1,MSG
238                     .BYTE   15,12
239                     .ASCII  /?NER-C/
240                     .BYTE   15,12,0
241                     .EVEN
242                     JMP     SAV28           ; FINISH UP
243             SAV40:  .LOCK                   ; LOCK USR IN CORE
244                     TST     DSTAT           ; FILE DEVICE?
245                     BPL     DERR0           ; NO- BRANCH
246                     .CLOSE  7               ; YES- CLOSE CHANNEL 7
247                     .LOOKUP 7,R2            ; LOOKUP THE FILE
248                     BCS     SAV25           ; NOT FOUND- BRANCH
```



```
249                     .UNLOCK                 ; RELEASE THE USR
250                     JSR     R1,MSG          ; ASK IF REPLACE
251                     .ASCII  /FILE ALREADY EXISTS/
252                     .BYTE   0
253                     .EVEN
254                     .CLOSE  7               ; NO- CLOSE CHANNEL 7
255                     .UNLOCK                 ; UNLOCK USR
256             SAV31:  BR      SAV28           ; RESTORE FILE AFTER CKING ")"
257             SAV25:  BIT     #40000,DSTAT    ; READ ONLY DEVICE?
258                     BEQ     SAV26           ; NO- BRANCH
259             DERR0:  JSR     PC,DERR         ; TELL NEWS
260                     BR      SAV31           ; AND FINISH
261             DERR:   .CLOSE  7               ; CLOSE CHANNEL 7
262                     .UNLOCK                 ; UNLOCK USR
263                     JSR     R1,MSG          ; TELL USER NEWS
264                     .BYTE   15,12           ; CR-LF
265                     .ASCII  /?DEV ERR - C/
266                     .BYTE   15,12,0
267                     .EVEN
268                     RETURN                  ; RETURN TO CALLER
269             SAV26:  .ENTER  7,R2            ; ENTER THE FILE
270                     CLR     R0              ; BLOCK ZERO
271                     .WRITW  7,WD1,WD2       ; WRITE OUT DATA
272                     BCS     DERR0           ; ERROR- EXIT THRU MESSAGE
273             SAV27:  .CLOSE  7               ; RELEASE CHANNEL 7
274                     .UNLOCK                 ; RELEASE USR
275                     JSR     R1,MSG
276                     .ASCII  /**SAVE COMPLETED**/
277                     .BYTE 15,12,0
278                     .EVEN
279                     BR      SAV31           ; FIX STACK
280             SAV28:  .ENDC
281                     ;  ************************
282 0562 016704 SAV29:  MOV     ACTST,R4        ; GET START ADR
         000000G
283 0566 012700         MOV     #TAG1+2,R0      ; ADR OF PTR TO FIRST TAG
         000002G
284 0572 011002 SAV30:  MOV     (R0),R2         ; GET ADR OF TAG
285 0574 001415         BEQ     SAV32           ; NO MORE SUBP- BRANCH
286 0576 024242         CMP     -(R2),-(R2)     ; POINT AT ADR OF REST OF FILE
287 0600 060422         ADD     R4,(R2)+        ; UPDATE ADR
288 0602 060422         ADD     R4,(R2)+        ; UPDATE ADR OF SUBP
289 0604 012246         MOV     (R2)+,-(SP)     ; PUT TAG ON STK- ZERO?
290 0606 005712         TST     (R2)            ; ADR OF NEXT TAG ZERO?
291 0610 001401         BEQ     .+4             ; YES- BRANCH
292 0612 060412         ADD     R4,(R2)         ; UPDATE TAG PTR
293 0614 005726         TST     (SP)+           ; WAS TAG ZERO?
294 0616 001002         BNE     .+6             ; NO- BRANCH
295 0620 011210         MOV     (R2),(R0)       ; YES- ELIMINATE LINK
296 0622 001763         BEQ     SAV30           ; LOOP
297 0624 010200         MOV     R2,R0           ; UPDATE PTR R0
298 0626 000761         BR      SAV30           ; LOOP
299 0630 010067 SAV32:  MOV     R0,TAGEND       ; STORE ADR TO LAST TAG+2
         000000G
300                     .IFDF   $DISK
301                     MOV     R1SVE,R1        ; RESTORE R1
302                     .ENDC
```

```
303 0634 000167 001152          JMP     $DXIT           ; EXIT
304                             .IFDF   $DISK
305                     R1SVE:  .WORD   0               ; SAVE R1 HERE
306                     DSTAT:  .=.+8.                  ; DEVBLK
307                     DEFEXT: .RAD50  'DPY'
308                             .WORD   0,0,0
309                     FILE:   .=.+78.                 ; 39 WORD BLOCK
310                             .IFDF   $DISK
```



```
1                               .SBTTL  "RESTORE" ROUTINE
2                               .ENDC
3                       ;       ****************************************
4
5                               ; ROUTINE TO HANDLE:
6                               ;       CALL "RSTR"( <FILE DESCRIPTOR> )
7
8                       RSTR:
9                               JSR     PC,BUFTST       ; ANY BUFFER?
10                              JSR     PC,STOPA        ; STOP THE DISPLAY
11                              CMPB    (R1)+,#,LPAR    ; A "(" ?
12                              BEQ     RSTR0           ; YES- BRANCH
13                              JMP     ERRSYN          ; NO- ERROR
14                      RSTR0:  MOV     R1,-(SP)        ; SIGNAL READ OPERATION
15                              TST     INSRTF          ; INSERT PENDING?
16                              BNE     .-4             ; YES- LOOP TIL FREE
17                              JMP     SAVERT          ; GET NAME IN ROOT
18                      RSTR1:  .LOOKUP 7,R2            ; LOOKUP THE FILE ON CHANNEL 7
19                              BCC     RSTR2           ; BRANCH IF FOUND
20                              .CLOSE  7               ; CLOSE CHANNEL 7
21                              .UNLOCK                 ; UNLOCK THE USR
22                              JSR     R1,MSG
23                              .BYTE   15,12
24                              .ASCII  /SAVE FILE NOT FOUND/
25                              .BYTE   15,12,0
26                              .EVEN
27                      RSTR4:  MOV     R1SVE,R1        ; RESTORE R1
28                              JMP     $DXIT
29                      RSTR2:  BIT     #20000,DSTAT    ; WRITE ONLY DEVICE?
30                              BEQ     RSTR3           ; NO- CONTINUE
31                      RDERR:  JSR     PC,DERR         ; DEVICE ERROR
32                              BR      RSTR4           ; AND EXIT
33                      RSTR3:  .READW  7,#WDD1,#2,#0   ; READ ON 7- 2 WORDS
34                              MOV     @ACTEND,-(SP)   ; SAVE END ADR
35                              MOV     DISEND,R4       ; GET CURRENT LO
36                              MOV     DCRASH,R0       ; HI END
37                              SUB     R4,R0           ; GET SIZE
38                              ASR     R0              ; CONVERT TO WORDS
39                              INC     R0              ; SIZE IN WORDS AVAILABLE
40                              CMP     WDD1,R0         ; ENOUGH ROOM?
41                              BLE     RSTR5           ; YES- BRANCH
42                              .CLOSE  7               ; CLOSE CHANNEL 7
43                              .UNLOCK                 ; RELEASE THE USR
44                              JSR     R1,MSG
45                              .BYTE   15,12
46                              .ASCII  /NOT ENOUGH DISPLAY BUFFER FOR RESTORE/
47                              .BYTE   15,12,0
48                              .EVEN
49                              TST     (SP)+           ; CLEAN STK
50                              BR      RSTR4           ; AND EXIT
51                      ;  GET REST OF DATA
52                      RSTR5:  MOV     (SP)+,R5        ; SAVE END ADDRESS
53                              CLR     R0              ; BLOCK 0
54                              CLR     -(SP)           ; CLEAR STK FOR READW
55                              MOV     WDD1,-(SP)      ; REST OF DATA
56                              ADD     #2,(SP)         ; ACTUAL END
57                              MOV     -(R4),R3        ; SAVE IN R3
```

```
58                      MOV     =(R4),R2        ; SAVE IN R2
59                      MOV     R4,=(SP)        ; ADR TO PUT DATA
60                      EMT     200+7           ; READ IT IN
61                      BCC     .+4             ; ALL OKAY= BRANCH
62                      BR      RDERR           ; NOT OKAY= BRANCH
63                      MOV     R2,(R4)+        ; RESTORE IT
64                      MOV     R3,(R4)+        ; RESTORE IT
65                      .CLOSE  7               ; CLOSE CHANNEL 7
66                      .UNLOCK                 ; UNLOCK USR
67              ;  STRAIGHTEN OUT FILE
68                      MOV     WDD1,R3         ; GET WORD COUNT
69                      ASL     R3              ; CONVERT TO BYTES
70                      ADD     R4,R3           ; POINT AT END
71                      MOV     R5,=(R3)        ; SAVE ADR OF RETURN
72                      TST     #NSTSK          ; IN A SUBP NOW?
73                      BNE     .+6             ; YES= DON'T UPDATE ACTEND
74                      MOV     R3,ACTEND
75                      MOV     #DJMP,=(R3)     ; SHOULDN'T BE NECESSARY
76                      MOV     R3,DISEND       ; NEW DISEND
77                      TST     WDD2            ; ANY SUBP IN NEW FILE?
78                      BEQ     RSTR4           ; NO= EXIT
79                      MOV     WDD2,R0         ; GET OFFSET OF FIRST TAG
80                      ADD     R4,R0           ; MAKE AN ADDRESS
81                      MOV     R0,@TAGEND      ; LINK UP TAG CHAIN
82                      MOV     TAGEND,R0       ; FIX R0 PTR
83                      JMP     SAV30           ; AND FINISH UP
84                      .ENDC
85                      .IFDF   $DISK
```



```
1                       .SBTTL  "FIX" AND "FREE" ROUTINES
2                       .ENDC
3               ;       ****************************************
4
5                       ; ROUTINE TO HANDLE:
6                       ;       CALL "FIX"(N), OR
7                       ;       IMPLICITLY ASK FOR HALF FREE SPACE
8
9  000640       FIX0:                           ; IMPLICIT ENTRY POINT
10 00640 010667         MOV     SP,IMP,F        ; SIGNAL IMPLICIT ENTRY
         001064
11 00644 005000         CLR     R0              ; SIGNAL WANT HALF
12 00646 000422         BR      FIX1            ; BRANCH
13 00650        FIXSP:                          ; ENTRY FROM FIX IN ROOT SECTION
14 00650 005067         CLR     IMP,F           ; CLEAR IMPLICIT ENTRY FLAG
         001054
15 00654 122127         CMPB    (R1)+,#.RPAR    ; A ")" ?
         000000G
16 00660 001402         BEQ     .+6             ; YES= BRANCH
17 00662 000167         JMP     ERRSYN          ; NO= SYNTAX ERROR
         000000G
18 00666 016500         MOV     FAC2(R5),R0     ; GET N
         000042
19 00672 003002         BGT     .+6             ; POSITIVE BRANCH
20 00674 000167 FIXARG: JMP     ERRARG          ; ERROR N <= 0
         000000G
21 00700 005767         TST     DCRASH          ; SPACE ALREADY ALLOCATED?
         000000G
22 00704 001001         BNE     FIX2            ; YES= BRANCH
23 00706 005200         INC     R0              ; NO ASK FOR
24                      .IFDF   $DISK           ; 1 OR 2 MORE WORDS
25                      INC     R0
26                      .ENDC
27 00710 006300 FIX2:   ASL     R0              ; CONVERT TO BYTES
28 00712 103770         BCS     FIXARG          ; TOO MANY= ERROR
29              ; FIND AVAILABLE SPACE
30 00714 026565 FIX1:   CMP     LOSTR(R5),HISTR(R5)
         000072
         000074
31 00722 001476         BEQ     FIX3            ; STRINGS ARE LOW
32 00724 010146         MOV     R1,=(SP)
33 00726 010246         MOV     R2,=(SP)        ; DO DNPACK HERE
34 00730 010346         MOV     R3,=(SP)
35 00732 005046         CLR     =(SP)           ; MAKE ROOM
36 00734 016502         MOV     LOSTR(R5),R2
         000072
37 00740 016501         MOV     LOFREE(R5),R1
         000014
38 00744 010165         MOV     R1,LOSTR(R5)
         000072
39 00750 005016 DNPLOOP:        CLR     (SP)
40 00752 152216         BISB    (R2)+,(SP)
41 00754 001012         BNE     DNPZRO
42 00756 020265 DNPBAD: CMP     R2,HISTR(R5)
         000074
43 00762 103772         BLO     DNPLOOP
44 00764 010165         MOV     R1,HISTR(R5)    ; SAVE HI STRING ADDRESS
```

```
                000074
45 00770 005726         TST     (SP)+
46 00772 012603         MOV     (SP)+,R3
47 00774 012602         MOV     (SP)+,R2
48 00776 012601         MOV     (SP)+,R1
49 01000 000447         BR      FIX3
50 01002 005003 DNPZRO: CLR     R3
51 01004 152203         BISB    (R2)+,R3
52 01006 000303         SWAB    R3
53 01010 152203         BISB    (R2)+,R3
54 01012 061602         ADD     (SP),R2
55 01014 005202         INC     R2
56 01016 030327         BIT     R3,#1
                000001
57 01022 001402         BEQ     .+6
58 01024 005303         DEC     R3
59 01026 061503         ADD     (R5),R3
60 01030 020365         CMP     R3,PDL(R5)      ; 4 = PDL
                000004
61 01034 103350         BHIS    DNPBAD
62 01036 020306         CMP     R3,SP
63 01040 103013         BHIS    DNPGOOD
64 01042 020365         CMP     R3,ARRAY$(R5)
                000010
65 01046 101343         BHI     DNPBAD
66 01050 020365         CMP     R3,HIFREE(R5)
                000012
67 01054 101005         BHI     DNPGOOD
68 01056 020365         CMP     R3,LOFREE(R5)
                000014
69 01062 103335         BHIS    DNPBAD
70 01064 020315         CMP     R3,(R5)
71 01066 103733         BLO     DNPBAD
72 01070 062716 DNPGOOD: ADD    #4,(SP)
                000004
73 01074 161602         SUB     (SP),R2
74 01076 020213         CMP     R2,(R3)
75 01100 001005         BNE     DNPIGNO
76 01102 010113         MOV     R1,(R3)
77 01104 112221         MOVB    (R2)+,(R1)+
78 01106 005316         DEC     (SP)
79 01110 003375         BGT     .-4
80 01112 000721         BR      DNPBAD
81 01114 061602 DNPIGNO:        ADD     (SP),R2
82 01116 000717         BR      DNPBAD
83 01120 016504 FIX3:   MOV     HIFREE(R5),R4   ; GET HIFREE PTR
                000012
84 01124 010446         MOV     R4,-(SP)        ; SAVE IT
85 01126 166516         SUB     HISTR(R5),(SP)  ; FREE BYTES
                000074
86 01132 062716         ADD     #2,(SP)         ; ACTUAL
                000002
87 01136 016767         MOV     DCRASH,WDD2     ; SAVE DCRASH
                000000G
                000570
88 01144 001405         BEQ     FIX3A           ; NO BUFFER= BRANCH
89 01146 166567         SUB     ARRAY$(R5),WDD2 ; SUBTRACT BOTTOM END
```



```
                000010
                000560
90 01154 066716         ADD     WDD2,(SP)       ; GET TOTAL FREE
                000554
91 01160 005700 FIX3A:  TST     R0              ; HAVE WD COUNT?
92 01162 001034         BNE     FIX5            ; YES= BRANCH
93 01164 011600         MOV     (SP),R0         ; GET MAX SPACE FREE
94 01166 006200         ASR     R0              ; DIVIDE BY 2
95 01170 006200         ASR     R0              ; CONVERT TO WORDS
96 01172 010046         MOV     R0,-(SP)        ; SAVE R0
97 01174 005300         DEC     R0
98                      .IFDF   $DISK
99                      DEC     R0              ; TWO WORDS FOR DISK VERSION
100                     .ENDC
101 1176 010065         MOV     R0,FAC2(R5)     ; PUT INTO OUTPUT BUFFER
                000042
102 1202 005065         CLR     FAC1(R5)
                000040
103 1206 004767         JSR     PC,NUMOUT       ; TELL USER
                000000G
104 1212 004167         JSR     R1,MSG
                000000G
105 1216    040         .ASCII  / WORDS FOR DISPLAY FILE/
    1217    127
    1220    117
    1221    122
    1222    104
    1223    123
    1224    040
    1225    106
    1226    117
    1227    122
    1230    040
    1231    104
    1232    111
    1233    123
    1234    120
    1235    114
    1236    101
    1237    131
    1240    040
    1241    106
    1242    111
    1243    114
    1244    105
106 1245    015         .BYTE   15,12,0
    1246    012
    1247    000
107                     .EVEN
108 1250 012600         MOV     (SP)+,R0        ; RESTORE R0
109 1252 006300         ASL     R0              ; BACK TO BYTES
110             ; SHIFT ARRAYS
111 1254 004767 FIX5:   JSR     PC,STOPA        ; STOP DISPLAY
                000000G
112 1260 016703         MOV     DCRASH,R3       ; GET HI END
                000000G
113 1264 010367         MOV     R3,WDD1         ; SAVE OLD DCRASH
```

97

```
                000442
114 1270 001004         BNE     .+12            ; BRANCH IF CALLED BEFORE
115 1272 016567         MOV     ARRAYS(R5),DCRASH       ; NEW DCRASH
                000010
                000000G
116 1300 000412         BR      FIX6            ; CONTINUE
117 1302 016702         MOV     ACTST,R2        ; GET LO END
                000000G
118 1306 160203         SUB     R2,R3           ; GET LENGTH OF CURRENT BUFFER
119 1310 062703         ADD     #2,R3           ; IN BYTES
                000002
120 1314 160300         SUB     R3,R0           ; SHIFT FACTOR
121                                             ; >0: NEED MORE// <0: FREE UP
122 1316 001003         BNE     FIX6            ; MUST CHANGE= BRANCH
123 1320 005726         TST     (SP)+           ; NO CHANGE= CLEAN STK
124 1322 000167         JMP     FIXXIT          ; AND EXIT
                000364
125 1326 016503 FIX6:   MOV     ARRAYS(R5),R3   ; GET TOP END
                000010
126 1332 005700         TST     R0
127 1334 002002         BGE     .+6             ; CONTINUE IF >= 0
128 1336 005726         TST     (SP)+           ; CLEAN STK
129 1340 000442         BR      FIX7            ; BRANCH
130 1342 166716         SUB     WDD2,(SP)       ; BACK TO FREE BELOW HIFREE
                000366
131 1346 020026         CMP     R0,(SP)+        ; ENOUGH ROOM?
132 1350 101421         BLOS    FIX25           ; YES= BRANCH
133 1352 016767 FIXERR: MOV     WDD1,DCRASH     ; RESTORE OLD DCRASH
                000354
                000000G
134 1360 104400         TRAP    0
135 1362    116         .ASCII  /NER - FOR DISPLAY BUFFER/
    1363    105
    1364    122
    1365    040
    1366    055
    1367    040
    1370    106
    1371    117
    1372    122
    1373    040
    1374    104
    1375    111
    1376    123
    1377    120
    1400    114
    1401    101
    1402    131
    1403    040
    1404    102
    1405    125
    1406    106
    1407    106
    1410    105
    1411    122
136 1412    000         .BYTE   0
137                     .EVEN
```



```
138 1414 004767 FIX25:  JSR     PC,PRIOR        ; FIX LEVEL AND LPS JUNK
                000402
139 1420 010402         MOV     R4,R2           ; GET ARRAY PTR
140 1422 160002         SUB     R0,R2           ; NEW ARRAY PTR
141 1424 010246         MOV     R2,-(SP)        ; SAVE ON STK
142 1426 022224         CMP     (R2)+,(R4)+     ; POINT AT CORRECT WORDS
143 1430 020403         CMP     R4,R3           ; DONE?
144 1432 101002         BHI     .+6             ; YES= BRANCH
145 1434 012422         MOV     (R4)+,(R2)+     ; MOVE WORD
146 1436 000774         BR      .-6             ; LOOP
147 1440 005742         TST     -(R2)           ; NEW ARRAYS
148 1442 010246         MOV     R2,-(SP)        ; SAVE ON STK
149 1444 000414         BR      FIX8            ; BRANCH
150             ; SHIFT UP
151 1446 004767 FIX7:   JSR     PC,PRIOR        ; FIX LEVEL AND LPS JUNK
                000350
152 1452 010302         MOV     R3,R2           ; GET ARRAYS
153 1454 160002         SUB     R0,R2           ; NEW ARRAYS
154 1456 005046         CLR     -(SP)
155 1460 010246         MOV     R2,-(SP)        ; SAVE ON STK
156 1462 022223         CMP     (R2)+,(R3)+     ; FOR AUTO= DECREMENT
157 1464 014342         MOV     -(R3),-(R2)     ; MOVE WORD
158 1466 020304         CMP     R3,R4           ; DONE?
159 1470 101375         BHI     .-4             ; NO= LOOP
160 1472 010266         MOV     R2,2(SP)        ; SAVE NEW ARRAY PTR
                000002
161             ; FIX SYMTAB
162 1476 011503 FIX8:   MOV     (R5),R3         ; START OF TABLE
163 1500 020365 FIXSRH: CMP     R3,LOFREE(R5)   ; END?
                000014
164 1504 103070         BHIS    FIX9            ; YES= BRANCH
165 1506 021327         CMP     (R3),#-3        ; ENTRY OR LINE NO, ?
                177775
166 1512 103011         BHIS    FIX12           ; ENTRY BRANCH
167 1514 022323 FIX17:  CMP     (R3)+,(R3)+     ; GO PAST LIN NO,
168 1516 000770         BR      FIXSRH          ; LOOP
169 1520 022327 FIX10:  CMP     (R3)+,#-2       ; AN ARRAY?
                177776
170 1524 001001         BNE     FIX11           ; NO= BRANCH
171 1526 160013         SUB     R0,(R3)         ; NEW ADR OF ARRAY
172 1530 062703 FIX11:  ADD     #10,R3          ; NEXT ENTRY
                000010
173 1534 000761         BR      FIXSRH          ; LOOP
174 1536 021327 FIX12:  CMP     (R3),#-1        ; A STRING?
                177777
175 1542 001366         BNE     FIX10           ; NO= KEEP GOING
176 1544 005723         TST     (R3)+           ; YES= BUMP R3
177 1546 010046         MOV     R0,-(SP)        ; SAVE OFFSET
178 1550 012300         MOV     (R3)+,R0        ; ADR OF START OF STRING
179 1552 012302         MOV     (R3)+,R2        ; GET SS1
180 1554 100442         BMI     FIX13           ; NO= SS= BRANCH
181 1556 011304         MOV     (R3),R4         ; GET SS2
182 1560 100420         BMI     FIX14           ; 1 DIMENSIONAL= BRANCH
183 1562 005204         INC     R4              ; MAKE SS2+1
184 1564 010246         MOV     R2,-(SP)
185 1566 010446         MOV     R4,-(SP)        ; SAVE HERE TOO
186 1570 012746         MOV     #20,-(SP)       ; 16 BITS
```

```
          000020
187 1574 006304 FIXLP:  ASL     R4              ; SS2*2
188 1576 006366         ASL     2(SP)
          000002
189 1602 103002         BCC     .+6
190 1604 066604         ADD     4(SP),R4
          000004
191 1610 005316         DEC     (SP)
192 1612 003370         BGT     FIXLP           ; FORM SS2+SS2
193 1614 062706         ADD     #6,SP           ; TIDY  STK
          000006
194 1620 060402         ADD     R4,R2           ; ADD SS1
195 1622 005202 FIX14:  INC     R2              ; SS1+1
196 1624 006302         ASL     R2              ; WORDS TO BYTES
197 1626 060002         ADD     R0,R2           ; HI ADR OF STRING LIST
198             ; FIX STRINGS
199 1630 020002 FIX15:  CMP     R0,R2           ; DONE?
200 1632 101013         BHI     FIX13           ; DONE LOOP
201 1634 022027         CMP     (R0)+,#-1       ; A NULL STRING ?
          177777
202 1640 001773         BEQ     FIX15           ; YES- LOOP
203 1642 014004         MOV     -(R0),R4        ; ADR OF STR
204 1644 005204         INC     R4              ; POINT AT FIRST BYTE OF PTR
205 1646 000300         SWAB    R0
206 1650 110024         MOVB    R0,(R4)+
207 1652 000300         SWAB    R0              ; RESTORE R0
208 1654 110024         MOVB    R0,(R4)+        ; SET IN SECOND BYTE
209 1656 005720 FIX16:  TST     (R0)+           ; ADD 2 TO ARRAY ADR
210 1660 000763         BR      FIX15
211 1662 012600 FIX13:  MOV     (SP)+,R0        ; RESTORE OFFSET
212 1664 000713         BR      FIX17
213             ; FIX BUFFER CHAIN
214 1666        FIX9:
215                     .IFDF   $DISK
216                     .SRESET                 ; GIVE BACK ALL DEVICE HANDLERS
217                     MOV     R5,R2           ; GET PTR TO BASE OF USER AREA
218                     ADD     #BUFCHN,R2      ; OFFSET TO BUFFER CHAIN POINTER
219             FIX9B:  TST     (R2)            ; ANY BUFFERS?
220                     BEQ     FIX9A           ; NO- BRANCH
221                     SUB     R0,(R2)         ; YES- CORRECT POINTER
222                     MOV     (R2),R2         ; POINT R2 AT NEXT BUFFER
223                     BR      FIX9B           ; CHECK IT
224             FIX9A:
225                     .ENDC
226 1666 011665         MOV     (SP),ARRAYS(R5) ; NEW PTR
          000010
227 1672 012602         MOV     (SP)+,R2        ; KEEP TO DETERMINE ACTST
228 1674 012665         MOV     (SP)+,HIFREE(R5) ; NEW PTR
          000012
229 1700 022222         CMP     (R2)+,(R2)+     ; NEW START
230                     .IFDF   $DISK
231                     TST     (R2)+
232                     .ENDC
233 1702 010267         MOV     R2,ACTST        ; START OF FILE
          000000G
234 1706 012667         MOV     (SP)+,PSW       ; RESTORE PSW
          177776'
```



```
235 1712 012767 FIXXIT: MOV     #-1,DISEND      ; FOR AUTO INIT
          177777
          000000G
236 1720 005767         TST     IMP,F           ; CALLED BY INIT?
          000004
237 1724 001434         BEQ     $DXIT1          ; NO- EXIT
238 1726 000207         RETURN                  ; GO BACK TO INIT
239
240 1730 000000 IMP,F:  .WORD   0
241 1732 000000 WDD1:   .WORD   0
242 1734 000000 WDD2:   .WORD   0
```

```
1                       ;       ************************************
2
3                               ;  ROUTINE TO HANDLE:
4                               ;       CALL "FREE"
5
6                               .IFNDF  $DISK
7  001736               FREE:
8                               .ENDC
9                               .IFDF   $DISK
10                      FREE0:
11                              .ENDC
12 01736 004767                 JSR     PC,STOPA        ; STOP DISPLAY
         000000G
13 01742 012767                 MOV     #-1,DISEND      ; FOR AUTO INIT
         177777
         000000G
14 01750 012767                 MOV     #BEG,DSTOP      ; LOOP DISPLAY
         000000G
         000000G
15 01756 016700                 MOV     DCRASH,R0       ; GET HI END
         000000G
16 01762 001415                 BEQ     $DXIT1          ; NOTHING THERE- EXIT
17 01764 005067                 CLR     DCRASH          ; CLEAR DCRASH
         000000G
18 01770 016503                 MOV     ARRAYS(R5),R3   ;
         000010
19 01774 160300                 SUB     R3,R0           ; TO GIVE IT ALL BACK
20 01776 005400                 NEG     R0
21 02000 016504                 MOV     HIFREE(R5),R4   ; SET UP REGISTERS
         000012
22 02004 005067                 CLR     IMP.F           ; FOR EXIT
         177720
23 02010 000616                 BR      FIX7            ; SHIFT IT
24 02012 000167 $DXIT:          JMP     CONT            ; START DISPLAY AND EXIT
         000000G
25 02016 000167 $DXIT1: JMP     CKEOL           ; CHECK EOL AND EXIT
         000000G
26
27                              .IFDF   $DISK
```



```
1                               .SBTTL  PRIORITY CHANGE AND LPS CLEAN-UP
2                               .ENDC
3                       ;       ****************************************
4
5  002022               PRIOR:
6                               .IFNDF  $VTONLY
7  002022 105767                TSTB    RTSON
         000000G
8  002026 001375                BNE     .-4             ; LOOP 'TIL NO DMA IN PROGRESS
9                               .ENDC
10 02030 011646                 MOV     (SP),-(SP)      ; MOVE DOWN RETURN ADR
11 02032 016766                 MOV     PSW,2(SP)       ; SAVE CURRENT PSW
         177776'
         000002
12 02040 052767                 BIS     #340,PSW        ; SET SELF AT LEVEL 7
         000340
         177776'
13                              .IFNDF  $VTONLY
14 02046 010146                 MOV     R1,-(SP)        ; SAVE R1
15 02050 012746                 MOV     #NARRAY,-(SP)
         000000G
16 02054 001415                 BEQ     PRIOR0          ; THERE IS NO LPS!!
17 02056 012701                 MOV     #TABLE,R1       ; GET START OF LPS ARRAY TABLE
         000000G
18 02062 005771 PRIOR1: TST     @(R1)           ; ANY ENTRIES?
         000000
19 02066 001410                 BEQ     PRIOR0          ; NONE- EXIT
20 02070 060031                 ADD     R0,@(R1)+       ; ADJUST BEG PTR
21 02072 060031                 ADD     R0,@(R1)+       ; ADJUST END PTR
22 02074 060031                 ADD     R0,@(R1)+       ; ADJUST PUT PTR
23 02076 060031                 ADD     R0,@(R1)+       ; ADJUST GET PTR
24 02100 062701                 ADD     #4,R1           ; POINT AT NEXT TABLE ENTRY
         000004
25 02104 005316                 DEC     (SP)            ; MORE LPS ARRAYS?
26 02106 003365                 BGT     PRIOR1          ; YES- LOOP
27 02110 005726 PRIOR0: TST     (SP)+           ; NO- CLEAN STK
28 02112 012601                 MOV     (SP)+,R1
29                              .ENDC
30 02114 000207                 RETURN
31
32       000001'                .END
```

```
GTC  V01-01  BASIC - GT  DISK O RT-11 MACRO VM02-09   16-OCT-74 02:14:36 PAGE 6+
SYMBOL TABLE

ACTEND= ****** G        ACTST = ****** G        ADRSET  000534R
ADSTK = ****** G        ARRAYS= 000010          BEG   = ****** G
BUFCHN= 000076          BUFTST= ****** G        CKEOL = ****** G
CONT  = ****** G        CONTA = ****** G        DCRASH= ****** G
DISEND= ****** G        DJMP  = 160000          DNPBAD  000756R
DNPGOO  001070R         DNPIGN  001114R         DNPLOO  000750R
DNPZRO  001002R         DSTOP = ****** G        ERRARG= ****** G
ERRSYN= ****** G        ESUB  = ****** G        FAC1  = 000040
FAC2  = 000042          FIXARG  000674R         FIXERR  001352R
FIXLP   001574R         FIXSP   000650RG        FIXSRH  001500R
FIXXIT  001712R         FIX0    000640RG        FIX1    000714R
FIX10   001520R         FIX11   001530R         FIX12   001536R
FIX13   001662R         FIX14   001622R         FIX15   001630R
FIX16   001656R         FIX17   001514R         FIX2    000710R
FIX25   001414R         FIX3    001120R         FIX3A   001160R
FIX5    001254R         FIX6    001326R         FIX7    001446R
FIX8    001476R         FIX9    001666R         FREE    001736RG
HIFREE= 000012          HISTR = 000074          IMP,F   001730R
INSRTF= ****** G        LOFREE= 000014          LOSTR = 000072
MSG   = ****** G        NARRAY= ****** G        NSTSK = ****** G
NUMOUT= ****** G        OLDMOD= ****** G        PC    =%000007
PDL   = 000004          PRIOR   002022R         PRIOR0  002110R
PRIOR1  002062R         PSW   = 177776          RETURN= 000207
RTSON = ****** G        R0    =%000000          R1    =%000001
R2    =%000002          R3    =%000003          R4    =%000004
R5    =%000005          SAVCON  000106R         SAVE    000000RG
SAVE0   000004R         SAVSET  000112R         SAV0    000020R
SAV1    000040R         SAV10   000170R         SAV11   000212R
SAV12   000240R         SAV13   000256R         SAV14   000232R
SAV15   000274R         SAV16   000316R         SAV17   000324R
SAV18   000206R         SAV19   000422R         SAV2    000056R
SAV20   000476R         SAV21   000544R         SAV23   000400R
SAV23A  000376R         SAV29   000562R         SAV3    000074R
SAV30   000572R         SAV32   000630R         SAV4    000124R
SAV5    000432R         SAV6    000146R         SAV7    000344R
SAV8    000330R         SAV9    000226R         SDINT = 173400
SDXIT   002012R         SDXIT1  002016R         SP    =%000006
STOPA = ****** G        TABLE = ****** G        TAGEND= ****** G
TAG1  = ****** G        WDD1    001732R         WDD2    001734R
WD1   = ****** G        WD2   = ****** G        $ADC  = 000000
$CLK  = 000000          $CLOCK= 000000          $DIO  = 000000
$DIS  = 000000          $LPS  = 000000          $STRNG= 000000
$VT11 = 000000          .EOL  = ****** G        .LPAR = ****** G
.RPAR = ****** G
. ABS.  000000     000
        002116     001
ERRORS DETECTED: 0
FREE CORE: 17950. WORDS

.LP:=PERPAR,GTL,GTC
```

```
GTNLPS V01-01  ;DUMMY LPS ENTRY RT-11 MACRO VM02-09   16-OCT-74 02:17:24 PAGE 1


1                ;TITLE GTNLPS V01-01  ;DUMMY LPS ENTRY POINTS FOR THE GT PKG,
2                ;
3                ; COPYRIGHT (C) 1974
4                ;
5                ; DIGITAL EQUIPMENT CORPORATION
6                ; MAYNARD, MASSACHUSETTS 01754
7                ;
8                ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
9                ; CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
10               ; AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION,
11               ; DEC ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT
12               ; MAY APPEAR IN THIS DOCUMENT,
13               ;
14               ; THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A
15               ; LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND
16               ; CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT
17               ; NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
18               ; OTHERWISE BE PROVIDED IN WRITING BY DEC,
19               ;
20               ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE
21               ; OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
22               ; WHICH IS NOT SUPPLIED BY DEC,
23               ;
24                       .GLOBL  TABLE,FLOAT,NARRAY,RTSON
25
26       000000          NARRAY=0
27 00000         TABLE:
28 00000         FLOAT:
29 00000         RTSON:
30 00000 000000          .WORD   0
31       000001'         .END
```

101

```
GTNLPS V01-01  ;DUMMY LPS ENTRY RT-11 MACRO VM02-09   16-OCT-74 02:17:24 PAGE 1+
SYMBOL TABLE

FLOAT   000000RG        NARRAY= 000000 G        RTSON   000000RG
TABLE   000000RG
. ABS.  000000     000
        000002     001
ERRORS DETECTED: 0
FREE CORE: 18561. WORDS

,LP:=GTNLPS
```

```
RT-11 LINK    V03-01    LOAD MAP
BGTLPS,SAV              16-OCT-74

SECTION ADDR    SIZE    ENTRY   ADDR    ENTRY   ADDR    ENTRY   ADDR

. ABS.  000000  000400  LIMIT   000002  PDL     000004  NARRAY  000005
                        PDSIZE  000006  ARRAYS  000010  COLUMN  000034
                        FAC1    000040  FAC2    000042  T1      000056
                        T2      000060  T3      000062  RND1    000064
                        RND2    000066  FILLCO  000110  $STKSZ  000200
                        ,EOL    000201  ,RPAR   000237  ,COMMA  000243
                        ,LPAR   000255  GTVECT  000320  LPSIVA  000340
                        LPSIP   000342  CKLIVA  000344  CKLIP   000346
                        DRSIVA  000350  DRSIP   000352  $TART   001102
                        ARGB    010262  ERRARG  010312  BOMB    010320
                        EVAL    011402  TABLES  011562  TBLSEN  011626
                        ERRPDL  011772  OPRATO  012166  ERRSYN  012544
                        SOPRAT  012720  ERRMIX  013012  STPRO   013162
                        GETVAR  015364  INT     015574  MAKEST  017252
                        MSG     017452  NORM    017532  NUMOUT  017724
                        NUMSGN  017736  SAVCHA  020570  STOVAR  021254
                        VAL     023354  LPSAD   170400  LPSADB  170402
                        LPSCKS  170404  LPSPB   170406  LPSDR   170410
                        LPSDRS  170410  LPSDIB  170412  LPSDOR  170414
                        LPDISS  170416  LPDISX  170420  LPDISY  170422
                        LPSDMA  170436  DPC     172000  DSR     172002
                        DISX    172004  DISY    172006  TKS     177560
                        TPB     177566
        000400  024100
        024500  004276  IFPMP   024500  ERRFPU  024502  $SBR    024502
                        $ADR    024506  ALOG    025242  AINT    025616
                        $INTR   025634  $DVR    025734  EXP     026366
                        $IR     026736  $MLR    027022  $POPR5  027374
                        $POPR4  027374  $POPR3  027406  $RI     027414
                        COS     027536  $IN     027572  ATAN    030112
                        $POLSH  030576  $V20A   030576  SQRT    030602
                        $ERR    030740  $ERRA   030750  $ERVEC  030770
        030776  000472  FTBL    030776
        031470  001462  TABLE   031470  RTSON   031564  DRSON   031565
                        HISTON  031566  BCDON   031567  DRSBUF  031570
                        DRSNPT  031572  USE     031574  RDB     031746
                        ACC     032004  CKCMGN  032030  GETNUM  032042
                        SAVER2  032056  SAVFAC  032062  SAVINT  032066
                        RESTR2  032074  REGSAV  032100  RESTOR  032116
                        INTST   032126  ERNOR   032226  FLOAT   032234
                        ENTERP  032310  BUFRES  032330  GETV    032346
                        GETBUF  032372  ERBUF   032440  GETDAT  032446
                        STODAT  032600  GETADD  032706  POLENT  033014
                        CKCOMA  033042  STORXT  033052  CKRPAR  033056
                        CONBCD  033066
        033152  001244  ADC     033222  RTS     033330  LED     033734
        034416  000716  SETR    034432  SETC    034570  HIST    035004
                        WAIT    035060
        035334  000542  DIR     035340  DOR     035434  DRS     035522
                        REL     035706
        036076  001462  CLRD    036164  PUTD    036610  DIS     036652
                        FSM     036726  DXY     037004  DISPLY  037252
        037560  002116  SAVE    037560  FIX0    040420  FIXSP   040430
                        FREE    041516
        041676  010614  BUFTST  042106  CKEOL   042222  INSRTF  042532
                        DISEND  042534  ACTEND  042536  ACTST   042540
                        #D1     042636  #D2     042640  SUBP    042704
                        ESUB    043146  ADSTK   043260  NSTSK   043310
                        TAGSRH  043312  TAGEND  043334  ERAS    043354
```

107

```
                    LPEN    043580  OLDMOD  043662  DSTP    044256
                    STOPA   044262  CONT    044346  CONTA   044352
                    INIT    044376  INITA   044402  NOSC    044612
                    ON      044622  OFF     044722  FIX     044730
                    ABS.,F  045536  STAT    045540  RDOT    045630
                    APNT    046016  VECT    046112  SC.USC  046216
                    TRAK    046544  XGRA    047022  YGRA    047040
                    SCAL    047576  SCL.Y   050270  SCL.X   050300
                    APUT    050360  AGET    050564  FIGR    050740
                    FPUT    051230  TEXT    051524  TIME    052106
                    TIMR    052360  BEG     052404  TAG1    052412
                    DSTOP   052506  DCRASH  052510
        052512  003544  USRARE  052512

TRANSFER ADDRESS = 054736
HIGH LIMIT = 056256
```

```
RT-11 LINK     V03-01     LOAD MAP
BASGT .SAV                16-OCT-74

SECTION ADDR    SIZE    ENTRY   ADDR    ENTRY   ADDR    ENTRY   ADDR

. ABS.  000000  000400  NARRAY  000000  LIMIT   000002  PDL     000004
                        PDSIZE  000006  ARRAYS  000010  COLUMN  000034
                        FAC1    000040  FAC2    000042  T1      000056
                        T2      000060  T3      000062  RND1    000064
                        RND2    000066  FILLCO  000110  $STKSZ  000200
                        .EOL    000201  .RPAR   000237  .COMMA  000243
                        .LPAR   000255  GTVECT  000320  START   001102
                        ARGB    010262  ERRARG  010312  BOMB    010320
                        EVAL    011402  TABLE5  011562  TBL5EN  011626
                        ERRPDL  011772  OPRATO  012166  ERRSYN  012544
                        $OPRAT  012720  ERRMIX  013012  STPRO   013162
                        GETVAR  015364  INT     015574  MAKEST  017252
                        MSG     017452  NORM    017532  NUMOUT  017724
                        NUMSGN  017736  SAVCHA  020570  STOVAR  021254
                        VAL     023354  DPC     172000  DSR     172002
                        DISX    172004  DISY    172006  TKS     177560
                        TPB     177566
        000400  024100
        024500  004276  IFPMP   024500  ERRFPU  024502  $SBR    024502
                        $ADR    024506  ALOG    025242  AINT    025616
                        $INTR   025634  $DVR    025734  EXP     026366
                        $IR     026736  $MLR    027022  $POPR5  027374
                        $POPR4  027374  $POPR3  027406  $RI     027414
                        COS     027536  SIN     027572  ATAN    030112
                        $POLSH  030576  $V20A    030576  SQRT    030602
                        $ERR    030740  $ERRA   030750  $ERVEC  030770
        030776  000310  FTBL    030776
        031306  000002  FLOAT   031306  RTSON   031306  TABLE   031306
        031310  002116  SAVE    031310  FIX0    032150  FIXSP   032160
                        FREE    033246
        033426  010614  BUFTST  033636  CKEOL   033752  INSRTF  034262
                        DISEND  034264  ACTEND  034266  ACTST   034270
                        WD1     034366  WD2     034370  SUBP    034434
                        ESUB    034676  ADSTK   035010  NSTSK   035040
                        TAGSRH  035042  TAGEND  035064  ERAS    035104
                        LPEN    035230  OLDMOD  035412  DSTP    036006
                        STOPA   036012  CONT    036076  CONTA   036102
                        INIT    036126  INITA   036132  NOSC    036342
                        ON      036352  OFF     036452  FIX     036460
                        ABS.,F  037266  STAT    037270  RDOT    037360
                        APNT    037546  VECT    037642  SC.USC  037746
                        TRAK    040274  XGRA    040552  YGRA    040570
                        SCAL    041326  SCL.Y   042020  SCL.X   042030
                        APUT    042110  AGET    042314  FIGR    042470
                        FPUT    042760  TEXT    043254  TIME    043636
                        TIMR    044110  BEG     044134  TAG1    044142
                        DSTOP   044236  DCRASH  044240
        044242  003544  USRARE  044242

TRANSFER ADDRESS = 046466
HIGH LIMIT = 050006
```

```
RT-11 LINK     V03-01     LOAD MAP
BASLPS,SAV                16-OCT-74

SECTION ADDR    SIZE    ENTRY   ADDR    ENTRY   ADDR    ENTRY   ADDR

. ABS.  000000  000400  LIMIT   000002  PDL     000004  NARRAY  000005
                        PDSIZE  000006  ARRAYS  000010  COLUMN  000034
                        FAC1    000040  FAC2    000042  T1      000056
                        T2      000060  T3      000062  RND1    000064
                        RND2    000066  FILLCO  000110  $STKSZ  000200
                        .EOL    000201  .RPAR   000237  .COMMA  000243
                        .LPAR   000255  LPSIVA  000340  LPSIP   000342
                        CKLIVA  000344  CKLIP   000346  DRSIVA  000350
                        DRSIP   000352  START   001102  ARGB    010262
                        ERRARG  010312  BOMB    010320  EVAL    011402
                        TABLE5  011562  TBL5EN  011626  ERRPDL  011772
                        OPRATO  012166  ERRSYN  012544  $OPRAT  012720
                        ERRMIX  013012  STPRO   013162  GETVAR  015364
                        INT     015574  MAKEST  017252  MSG     017452
                        NORM    017532  NUMOUT  017724  NUMSGN  017736
                        SAVCHA  020570  STOVAR  021254  VAL     023354
                        LPSAD   170400  LPSADB  170402  LPSCKS  170404
                        LPSPB   170406  LPSDR   170410  LPSDRB  170410
                        LPSDIB  170412  LPSDOR  170414  LPDISS  170416
                        LPDISX  170420  LPDISY  170422  LPSDMA  170436
                        TKS     177560  TPB     177566
        000400  024100
        024500  004276  IFPMP   024500  ERRFPU  024502  $SBR    024502
                        $ADR    024506  ALOG    025242  AINT    025616
                        $INTR   025634  $DVR    025734  EXP     026366
                        $IR     026736  $MLR    027022  $POPR5  027374
                        $POPR4  027374  $POPR3  027406  $RI     027414
                        COS     027536  SIN     027572  ATAN    030112
                        $POLSH  030576  $V20A   030576  SQRT    030602
                        $ERR    030740  $ERRA   030750  $ERVEC  030770
        030776  000164  FTBL    030776
        031162  001462  TABLE   031162  RTSON   031256  DRSON   031257
                        HISTON  031260  BCDON   031261  DRSBUF  031262
                        DRSNPT  031264  USE     031266  RDB     031440
                        ACC     031476  CKCMGN  031522  GETNUM  031534
                        SAVER2  031550  SAVFAC  031554  SAVINT  031560
                        RESTR2  031566  REGSAV  031572  RESTOR  031610
                        INTST   031620  ERNOR   031720  FLOAT   031726
                        ENTERP  032002  BUFRES  032022  GETV    032040
                        GETBUF  032064  ERBUF   032132  GETDAT  032140
                        STODAT  032272  GETADD  032400  POLENT  032506
                        CKCOMA  032534  STORXT  032544  CKRPAR  032550
                        CONBCD  032560
        032644  001244  ADC     032714  RTS     033022  LED     033426
        034110  000716  $ETR    034124  $ETC    034262  HIST    034476
                        WAIT    034552
        035026  000542  DIR     035032  DOR     035126  DRS     035214
                        REL     035400
        035570  001462  CLRD    035656  PUTD    036302  DIS     036344
                        FSH     036420  DXY     036476  DISPLY  036744
        037252  003544  USRARE  037252

TRANSFER ADDRESS = 041476
HIGH LIMIT = 043016
```