```
* THIS IS A COPY OF MSW0 CREATED ON 11/20 AT 14:00. DURING THE NEXT
* TWO WEEKS OR SO D.C. WILL BE ADDING COMMENTS TO IT.
*  THE CHANGES OF 11/25, TO INSERT THE BPT2 OPTION
*  AND TO PRINT MDA AND MDA/2**13  ARE INCORPORATED
*  NOV. 15, 14:00. RADP2 CHANGED TO POINT TO BAND 110, PAGE 2.
MSW0    IDENT   H18 2/13/68
* MONITOR MODULE 0
        ENTRY   RCVRA
DST     ZRO
        BRM     DTS
        SKN     NDCL
        BRU     *-1
        SKN     DRMTRY
        BRR     DST
        MIN     DST
        BRR     DST

* THIS ROUTINE INITIALIZES THE DRUM I/O
DRMSET  ZRO     0
        LDA     =-1
        STA     NDCL
        STA     DTKS1
        STA     DSCFLG
        LDA     =DRQ
        STA     EDCL
        STA     IDCL
        CLA
        STA     DSCINT
        BRR     DRMSET

* 'IOPSET'  9/27/65
*
* THIS ROUTINE INITIALIZES THE FILE CONTROL TABLE AND ALL
*  NON-TELETYPE I/O.
*
IOPSET  ZRO
        LDA     =400000038
        LDX     =-NFILE+3
ISET1   ADD     =1
        STA     EFA,2
        BRX     ISET1
        LDA     =3
        STA     FFLST
        LDX     =LADIU
        LDB     =400000008
        STB     EFA1
        LDB     =-1
ISET2   STB     EDIU,2
        BRX     ISET2
        LDX     =NDEV
ISET3   LDA     EBUFS,2
        SKA     =400008
        BRU     *+2
        STB     ADIU,2
```

```
         BRX     ISET3
ISET5    CXA
         SKE     =NTAPE
         BRU     *+2
         BRU     ISET4
         STB     TJNO,2
         CLA
         STA     TSTATE,2
         STB     TAPREL,2
         STA     TDIBA,2
         STA     TDFNO,2
         EAX     1,2
         BRU     ISET5
ISET4    CLA
         STA     BLK31
         BRR     IOPSET
* 'TTYSET'  9/26/65
*
* THIS ROUTINE INITIALIZES THE TELETYPE HARDWARE AND TABLES
* IT IS CALLED WITH -1 OR 0 DEPENDING ON WHETHER TTYASG, TTYTBL AND LINK
* WORDS ARE TO BE RESET OR NOT
*
TTYSET   ZRO     0
         STA     SRFLG
         LDB     =TTYBUF
         LDA     =NTTB-1
         STA     T
         CLX
TSET1    STB     TIS4,2
         STB     TIS5,2
         STB     TOS4,2
         STB     TOS5,2
         SKN     SRFLG
         BRU     TSET3
         LDA     =ITTBL
         STA     TTYTBL,2
         LDA     ADISK
         STA     TTYASG,2
         LDA     =NTTB+400000000B
         STA     LCW,2
TSET3    CLA
         STA     TIS2,2
         STA     TTYTI4,2
         LDA     TOS5,2
         ADD     =NTTYC
         COPY    AB
         LDA     =-1
         STA     TOS2,2
         STA     TOS3,2
         EAX     1,2
         SKR     T
         BRU     TSET1
         LDA     =NTTB+400000000B
         STA     LCW,2
```

```
         LDA     =-1
         STA     ATIS2
         LDA     =ATTBUF
         STA     ATIS4
         STA     ATIS5
         SKN     SRFLG
         BRR     TTYSET
         LDX     =-NJOB+2
         STX     ETTNQ
TSET2    CXA
         ADD     =NJOB
         STA     ETTNQ,2
         BRX     TSET2
         LDA     =1
         STA     FULST
         BRR     TTYSET
SRFLG    ZRO     Ø

*
*        SYSTEM INITIALIZE ROUTINE
*
* RESET SYSTEM
SSET     ZRO
         BRM     DRMSET
         LDA =WR4; MUL =3; LSH 23; SUB =1
         STA WR4X3; SUB =34000B*3; STA XWR4X3
         LDX     =-NPAC*NPPAR
         STX     CLOCK3
         CLB
SET1     LDA     =700000B
         STA     PTEST,2
         CXA
         ADD     =PPTR
         ADD     =NPPAR
         STA     PPTR,2
         EAX     NPPAR-1,6
         BRX     SET1
         LDA     =PPTR2
         STA     FPLST
         STB     PPTRU
         STB     PUCTR
         LDX     NLQ
SET6     LDA     Q1E,2
         STA     Q1,2
         SUB     =PNEXT
         SUB     =3
         STA     Q1N,2
         EAX     2,6
         BRX     SET6
         SUB     NLQ
         LDX     =-3
         STA     Q1N,2
         LDX     =-NPUQ*3
SET3     CXA
```

```
        ADD     =EPUCT3
        STA     EPUCT,2
        EAX     2,6
        BRX     SET3
        CLB
        STB     EPUCM3
        LDA     =PUCT
        STA     FPULST
        LDA     =PUBPTR
        STA     PUBPTR
        STA     PUEPTR
        LDA     =2S
        LDX     =-NSMEM
        STA     ESRMC,2
        BRX     *-1
        LDA     =-1
        CLB
        LDX     =NSMEM-NMEM
        STA     ERMC,2
        STB ERMT,2; STB ERMA,2
        BRX     *-3

        LDX =-NDRAT*NSEC; LDA =-1; STA EDRAT,2; BRX *-1
        LDX     =-NJOB
        CLA
        STA     EPMTP,2
        BRX     *-1

        LDA     BST
        LDX     =-100B+NPOP
        STA     200B-NPOP,2
        BRX     *-1
        STA     177B
        BRM     IOPSET
        LDA     =-1
        BRM     TTYSET
        BRR     SSET

WDSU    ZRO; LDA =FF; SNB =30000B; CAX; LDA WDSU1
        STA 30000B,2; BRX *-1
        LDA =FP; SUB =40000B; COPY AX,BA; LDA WDSU1; STA 40000B,2; BRX *-1
        LDX =-77B; STA 100B,2; BRX *-1
        LDX =WDTB-WDTE
        LDA WDTE,2; BRX *+1; STA* WDTE,2; BRX *-3; BRR WDSU
WDSU1   BRM RCVR


WDTB    EQU     *

        BRU SETSET; 0 24B
        BRM CONT; 0 27B
        BRU LOAD; 0 30B
WDPIT   BRM     TRAPI
        0       40B
```

```
WDIVT   BRM     TRAP1
        0       41B
WDRMT   BRM     TRAPR
        0       43B
WDUMT   BRM     TRAPT
        0       44B
WD31    BRM     INT31
        0       31B
WD33    BRM     INT33
        0       33B
WDPWFI  BRM     PWFI
        0       37B
CLOCK1  BRM     CLINT
        0       74B
CLOCK2  SKR     CLOCK3
        0       75B
WDTI    BRM     TII
        0       200B
WDTO    BRM     TOI
        0       201B
WDTN    BRM     TNI
        0       202B
WDTF    BRM     TFI
        0       203B
WDIRM   BRM     IRER
        0       65B
        BRM     ERM
        0       64B
        IF      ARMF
WDATI   BRM     ATII
        0       204B
        ENDF

WDTE    EQU     *


* SCAN THROUGH USER T.S. BLOCKS
TSCN    ZRO
        LDX     =-NJOB
TSCN1   STX     TSCNS1
        LDA     EPMTP,2
        SKG     =0
        BRU     TSCN2
        CAX
        SKN     0,2
        BRU     TSCN5
        LDA     0,2
        LSH     5
        ETR     =3774000B
        STA     T
TSCN3   LDA     T
        LDB     =34000B
        BRM     GDBC
        BRU     TSCN3
```

```
        LDA     =MBUFX
        STA     FBWRD
        LDX     =340000B
        LDA     =-1
        STA     MAGT,2
        IF -1
        STA     SMIFIL
        STA     SMOFIL
        ENDF
TSCN4   LDA     T
        LDB     =400340000B
        BRM     GDBC
        BRU     TSCN4
TSCN2   LDX     TSCNS1
        BRX     TSCN1
        BRR     TSCN
TSCN5   EQU     TSCN2
TSCNS1  ZRO

* COPY OUT TABLE FOR CRASH RECOVERY
RCT     ZRO
        STB     RCT1
        CAB
        SKR     RCT1
RCT2    XXB
        LDA     0,2
        EAX     1,2
        XXB
        STA     0,2
        EAX     1,2
        SKR     RCT1
        BRU     RCT2
        CBA
        BRR     RCT
RCT1    ZRO

*
* CRASH RECOVERY PROGRAM
*
RCVRA   LDA     =RCVT
        LDB     =NRCVT
        LDX     =340000B
        BRM     RCT
        LDA     =TIND
        LDB     =NJOB
        BRM     RCT
        LDA     =DRAT
        LDB     =NDRAT*NSEC
        BRM     RCT
        LDA =TABLE; LDB =LTABLE; BRM RCT
        LDA =PMT; LDB =NUME4*NJOB; BRM RCT
        LDA =FFLST; LDB =4*NFILE+1; BRM RCT
* WRITE OUT PAGES IN CORE
        LDX     =NME4-1
```

```
RES11   STK     , RCT
        LDA RMT,2; SKG =0; BRU RES12
        LDA* RMT,2; SKA =X4; BRU RES12
        LSH     5
        ETR     =37740000B
        STA     RCT1
RES13   LDA     RCT
        SKG     =7
        ADD     =108-NS4E4
        CLS
        LSH     11
        MRG     DCWBIT
        CAB
        LDA     RCT1
        BRM     GDBC
        BRU     RES13
RES12   LDA     RCT
        SKE     =10B
        BRU     RES14
RES15   LDA     =DMPSND*200000B+NS4E4*40000B
        LDB     =400000B
        LDX     =400000B-NS4E4*40000B
        BRM     GDAC
        BRU     RES15
RES14   LDA     RCT
        SUB     =1
        CAX
        SKE     =NS4E4-1
        BRU     RES11
* READ IN NEW COPY OF SYSTEM
        BPT4
        BRU     *-1
        READ    6*40000B,0,10*200000B
        BRM     SSET
* RESTORE CRITICAL TABLES
        LDA     =340000B
        LDB     =NRCVT
        LDX     =RCVT
        BRM     RCT
        LDX     =TTND
        LDB     =NJOB
        BRM     RCT
        LDX     =DRAT
        LDB     =NDRAT*NSEC
        BRM     RCT
        LDX =TABLE; LDB =LTABLE; BRM RCT
        LDX =PMT; LDB =NJ4E4*NJOB; BRM RCT
        LDX =FFLST; LDB =4*NFILE+1; BRM RCT
* MARK ALL MEMORY 'ON DRUM'
        LDX     =PMT
RES22   LDA     0,2
        ETR =177740B
        SKA     =77700B
        MRG     X4
```

```
        STA     0,2
        EAX     1,2
        CXA
        SKE     =EPAT
        BRU     RES22
* SET UP ALL JOBS WITH PAC SLOT
        CLA
        STA     JOB
RES20   LDX     JOB
        LDA     PMTP,2
        SKG     =0
        BRU     RES21
        LDA     TTNO,2
        STA     FILE
        BRM     GFX
        HLT
        STX     PUPAC
        BRM     SETPAC
        LDA     EXECRS
        STA     PL,2
RES21   MIN     JOB
        LDA     JOB
        SKE     =NJOB
        BRU     RES20
        IF -1
        BRM     TSCN
        ENDF
        BRM     DRMSET
        ARMI    AIRWD
        BRU     TTYGO
EXECRS  ZRO     EXECR,4
$CONTA  READ    6*4000B,0,10*20000B
        BRU 24B


*READ CHARACTER FROM CONSOLE TYPEWRITER.
*LEAVES CHARACTER IN SDS CODE, RIGHT JUSTIFIED IN A.
RDCCT   ZRO; DIR; SKS 14000B; BRU *-1
        EOM 2001B; WIM RDCCT1; DISW
        SKS 14000B; BRU *-1; EIR
        LDA RDCCT1; ETR =77B; BRR RDCCT
RDCCT1  ZRO

*READ UNSIGNED DECIMAL NUMBER FROM CONSOLE TYPEWRITER.
*TERMINATES ON ANY NON-NUMERIC CHARACTER, WITH TERMINATOR IN B.
RDDCT   ZRO; CLA; STA RDDCT1
RDDCT2  BRM RDCCT; SKG =9; BRU *+4; CAB; LDA RDDCT1; BRR RDDCT
        XMA RDDCT1; MWL =5; CBA; ADM RDDCT1; BRU RDDCT2
RDDCT1  ZRO

*PRINT CHARACTER ON CONSOLE TYPEWRITER.
*TAKES CHARACTER RIGHT JUSTIFIED IN A.  DESTROYS REGISTERS
PRCCT   ZRO; BRM CASDS; LCY 18; STA RDCCT1
        DIR; SKS 14000B; BRU *-1
```

```
        EOM 2041B; MIN RDCCT1; TOPW
        SKS 14000B; BRU *-1; EIR; BRR PRCCT

*PRINT MESSAGE ON CONSOLE TYPEWRITER.  TAKES IN A ADDRESS OF
*STRING, 4 CHARACTERS PER WORD.  &=CARRIAGE RETURN, /=TERMINATOR AS FOR
*BRS 34.
PRMCT   ZRO; STA PRMCT1
PRMCT2  LDB* PRMCT1; LDX =-4
PRMCT3  STX PRMCT6; LSH 6; ETR =77B; SKE =17B; BRU PRMCT4
        LDA =8000; STA PRMCT1; SKR PRMCT1; BRU *-1; BRR PRMCT
PRMCT4  SKE =4; BRU *+2; LDA =1; STB PRMCT5; BRM PRCCT; LDB PRMCT5
        LDX PRMCT6; BRX PRMCT3; MIN PRMCT1; BRU PRMCT2
PRMCT1  ZRO
PRMCT5  ZRO
PRMCT6  ZRO

PRNCT   ZRO; LDX =40000B; STB PRNCT1
PRNCT2  LRSH 23; DIV PRNCT1; EAX -1,2
        STB PRNCT4,2; SKE =0; BRU PRNCT2
PRNCT3  LDA PRNCT4,2; ADD =20B; STX PRNCT1
        BRM PRCCT; LDX PRNCT1; BRX PRNCT3; BRR PRNCT
PRNCT1  ZRO
PRNCT4  BES 12

*SET DATE FROM CONSOLE TYPEWRITER.
SDAT    ZRO; LDA =SDAT1; BRM PRMCT; BRM RDDCT; SUB =1
        MUL =82320; STB DMIN; BRM RDDCT; SUB =1; MUL =720
        CBA; ADM DMIN; BRM RDDCT; MUL =30; CBA; ADM DMIN
        BRM RDDCT; ADM DMIN
        CBA; SKE =33B; BRU SDAT+1
        LDA =-1; STA CLOCK3; LDA =BIT0; STA REAL
        LDA SDAT3; STA 74B; CKN; BRR SDAT
SDAT3   BRM *+1; ZRO; MIN REAL; BRU *-2
SDAT1   TEXT    '$ENTER [MO-DAY-HOUR-MIN]: /'

*CONVERT FROM TRIMMED ASCII TO SDS CODES
CASDS   ZRO
        ETR =77B; LRSH 2; AXC; LSH 2; MUL =3; LDA ASDST,2
        COPY BX,AB,A; LSH 6,2; ETR =77B; BRR CASDS
ASDST   DATA    12521212B,53121214B,74345420B,73403361B
        DATA    00010203B,04050607B,10111556B,36331612B
        DATA    12212223B,24252627B,30314142B,43444546B
        DATA    47505162B,63646566B,67707135B,76551212B

*ERASE TAPE
ERT     ZRO; STB ERT1; CLB
        LRSH 10; ETR =37B; ARG =214000B
        STA ERT2; STB ERT3; EOM* 10000B
ERT2    EOM 14000B; POT ERT3
ERT1    EOM 36773
        CATW; BRU *-1; BRR ERT
ERT3    ZRO

*ERASE TAPE FORWARD
```

```
ERTF    ZRO; LDB =2036773; BR4 ERT; BRR ERTF

*ERASE TAPE BACKWARDS
ERTR    ZRO; LDB =2076773; BR4 ERT; BRR ERTR

*PREPARE TAPE FOR DISC DUMP
PRET    ZRO
PRET2   LDA =49; BR4 PR4CT
        BR4 TWAIT7; LDA =426; BR4 PR4CT; BR4 RDDCT
PRET1   BR4 TWAIT7
        EOM 14017B; BR4 TWAIT7; SKS 14017B; BRU PRET2
        SKS 17217B; BRU *+2; BRU PRET2
        SKS 12017B; BRU *+2; BRU PRET2
        LDA =600; BR4 ERTF; BRR PRET


*WAIT FOR DISC
DWAIT   ZRO; ORT; BRU *-1
        CATW; BRU *-1; RDC; BRR DWAIT

*EXECUTE DISC COMMAND
DDO     ZRO; STB DDO1; BR4 DWAIT
        ADA; PDT 33777B
        EOM* 10000B; EOM 14204B; PDT =34000B
DDO1    EOM 2626B; BR4 DWAIT
        DET; BRR DDO; CETW; BRR DDO; CZTW; BRR DDO
        LDA 33777B; ADD =100B; ETR =777777B; STA DDO2
        ADA; PDT DDO2; MIN DDO; BRR DDO
DDO2    ZRO

*READ DISC
DREAD   ZRO; LDB =202626B; BR4 DDO; MIN DREAD; BRR DREAD

*WRITE DISC
DRITE   ZRO; LDB =203666B; BR4 DDO; MIN DRITE; BRR DRITE

*GET 100000B WORDS FROM DISC
DGET    ZRO; LDA =3; STA DGET1
DGET2   BR4 DREAD; BRR DGET
DGET3   SKR DGET1; BRU DGET2
        LDA =410; BR4 PR4CT
        BR4 PRDA; MIN DGET; BRR DGET
DGET1   ZRO

*WAIT FOR TAPE UNIT 7
TWAIT7  ZRO; CATW; BRU *-1
        SKS 10417B; BRR TWAIT7; BRU *-2

*WAIT FOR TAPE UNIT 3
TWAIT3  ZRO; CATW; BRU *-1; SKS 10413B; BRR TWAIT3; BRU *-2

*EXECUTE TAPE COMMAND
TDO     ZRO; STB TDO1
        EOM* 10000B; EOM 14004B; PDT =73777B
```

```
TDD1    EDM 3657B; BRR TDD

*WRITE 100013 WORDS ON TAPE 7
TRITE   ZRO; BRM TWAIT7; LDB =2036578
        BRM TDD; BRM TWAIT7; CETW; MIN TRITE
        BRR TRITE

*READ 100013 WORDS FROM TAPE 3
TREAD   ZRO; BRM TWAIT3; LDB =2036138
        BRM TDD; BRM TWAIT3
        BRM TERR3; MIN TREAD; BRR TREAD

*CALCULATE NUMBER OF WORDS WRITTEN ON TAPE
TCT     ZRO; ASCW; PIN TCT1
        LDA TCT1; SUB =33777B; STA TCT1; BRR TCT
TCT1    ZRO

*WRITE DISC BLOCK ON TAPE 7
TPUT    ZRO
TPUT1   BRM TRITE; BRR TPUT
        BRM TCT; BRM ERTR; BRM TRITE; BRR TPUT
        BRM TCT; BRM ERTR; LDA TCT1; BRM ERTF
        LDA =M11; BRM PRMCT
        BRM PRDA; BRU TPUT1

*DUMP DISC ON TAPE 7
DSDMP   ZRO; BRM PRET; LDA DFRD
DSDMP1  STA 33777B; BRM DGET; BRM TPUT
        LDA 33777B; ADD =100B; SKG DTO; BRU DSDMP1
        BRM TEOF; BRR DSDMP
DFRD    ZRO
DTO     ZRO

*LOAD DISC FROM TAPE 3
DSLD    ZRO; BRM PRETL; BRR DSLD
DSLD1   BRM TGET; BRM DPUT; SKN EOF; BRU DSLD1; BRR DSLD

*PREPARE TAPE FOR DISC LOAD
PRETL   ZRO; LDA =M12; BRM PRMCT
        BRM RDCCT; SKE =70B; BRR PRETL
        LDA =37777777B; STA EOF
        CLA; STA CLOC
        DRT; BRU *-1; ADA; POT CLOC; MIN PRETL
PRETL2  LDA =M13; BRM PRMCT
        BRM TWAIT3; LDA =M26; BRM PRMCT; BRM RDDCT
        SKS 17213B; BRU *+2; BRU PRETL2
        EDM 14013B; BRM TWAIT3
        SKS 12013B; BRR PRETL; BRU PRETL2

*GET DISC BLOCK FROM TAPE 3
TGET    ZRO; LDA =5; STA TGET2
TGET3   BRM TREAD; BRR TGET
        SKS 13613B; BRU TGET1
        SKR TGET2; BRU TGET4
```

```
        LDA =414; BRM PRMCT
        BRM PRCL; BRM PRTDA
TGETS   MIN TGET; BRR TGET
TGET1   MIN EOF; EOM 140138; BRU TGETS
TGET4   BRM SCANR3; BRU TGET3
TGET2   ZRO

*WRITE EOF ON TAPE 7 AND REWIND
TEOF    ZRO; LDA =600; BRM ERTF
        EOM* 10000B; EOM 14000B; POT TEOF1
        EOM 2057B; BRM TWAIT7; EOM 14017B; BRR TEOF
TEOF1   ZRO* =17000000B
EOF     DATA 37777777B

*TYPE DRUM ADDRESS
PRDA    ZRO; LDA 33777B; LDB =8; BRM PRNCT; BRR PRDA

*TYPE PRESUMED DRUM ADDRESS
PRCL    ZRO; LDA CLOC; LDB =8; BRM PRNCT; BRR PRCL

PRTDA   ZRO; LDA =415; BRM PRMCT; BRM PRDA; BRR PRTDA

*PUT DISC BLOCK ON DISC
DPUT    ZRO; LDA =3; STA DPUT2
        LDA CLOC; SKE 33777B; BRU DPUT1
DPUT3   BRM DRITE; BRU DPUT4
        SKR DPUT2; BRU DPUT3
        LDA =416; BRM PRMCT; BRM PRDA
DPUT4   LDA =100B; ADM CLOC; BRR DPUT
DPUT1   LDA =417; BRM PRMCT; BRM PRCL; BRM PRTDA
        LDA 33777B; STA CLOC; BRU DPUT3
DPUT2   ZRO;CLOC ZRO

*CHECK TAPE 3 FOR READ ERRORS
TERR3   ZRO; CETW; BRR TERR3
        CZTW; BRR TERR3; SKS 13613B; BRR TERR3
        MIN TERR3; BRR TERR3

*SCAN TAPE
SCAN    ZRO; STB SCAN1
        EOM* 10000B; EOM 14000B; POT =40000B
SCAN1   EOM 7633B; BRR SCAN

*BACKSPACE RECORD ON TAPE 3
SCANR3  ZRO; LDB =2376333B; BRM SCAN; BRR SCANR3

NC1     EQU 3
CT1     DATA 243,433,623
ECT1    EQU *
BR1     DATA D,L,S
EBR1    EQU *
NC2     EQU 3
```

```
CT2     DATA 23B,24B,51B
ECT2    EQU *
BR2     DATA DCR,DDT,DRT
EBR2    EQU *
BR3     DATA LCR,LDT,LRT
EBR3    EQU *

$SETSA  LDA =425; BRM PRMCT; LDA =2; STA SETSA1
        BRM RDCCT; SKR SETSA1; BRU *-2
        BRM RDCCT; SKE =33B; BRU *-2
        BRM SDAT; LDA =-1; STA OKCTR
        LDA =ARGS; STA PARG; STA RARG; STA CARG
CL3     SKN OKCTR; BRU OK2
        LDA =OKSTK; STA OKPTR
CL1     LDA =421
        BRM PRMCT; BRM RDCCT; LDX =-NC1
        SKE ECT1,2; BRU *+2; BRU* EBR1,2; BRX *-3
TRAG    LDA =413; BRM PRMCT; LDA RARG; STA PARG; BRU CL1

D       LDA =419; BRM PRMCT; BRM RDCCT; LDX =-NC2
        SKE ECT2,2; BRU *+2; BRU* EBR2,2; BRX *-3
        BRU TRAG

L       LDA =420; BRM PRMCT; BRM RDCCT; LDX =-NC2
        SKE ECT2,2; BRU *+2; BRU* EBR3,2; BRX *-3
        BRU TRAG

S       LDA =424; BRM PRMCT; BRM OKTOGO; BRM SSET
        BPT2  SKIP BIT MAP IF BPT2 SET; BRU *+2  CH 11/25/68
        BRM BMAP; BRU TTYGO

DCR     EQU TRAG
DRT     EQU TRAG
LRT     EQU TRAG
LCR     EQU TRAG
DDT     LDA =422; BRM PRMCT; BRM RDDCT; BRM STARG
        LDA =427; BRM PRMCT; BRM RDDCT; BRM STARG
        LDA =423; BRM PRMCT; BRM OKTOGO
        BRM FARG; CLB; LSH 13; STA DFRD
        BRM FARG; CLB; ADD =1; LSH 13; SUB =1; STA DTO
        BRM DSDMP; BRU CL3
LDT     LDA =423; BRM PRMCT; BRM OKTOGO; BRM DSLD; BRU CL3

SETSA1  ZRO
NOKST   EQU 5
OKCTR   DATA -1
OKPTR   DATA OKSTK
OKSTK   BSS 5

ARGS    BSS 10
PARG    DATA ARGS
RARG    DATA ARGS
CARG    DATA ARGS
FARG    ZRO; LDA* CARG; MIN CARG; BRR FARG
```

```
STARG   ZRO; XMA PARG; SKE =PARG; BRU *+2; BRU TRAG
        XMA PARG; STA* PARG; MIN PARG; BRR STARG

OKPSH   ZRO; MIN OKCTR; LDA PARG; STA RARG; LDA OKTOGO
        STA* OKPTR; MIN OKPTR; BRR OKPSH

OKTOGO  ZRO; BRM RDCCT; SKE =73B; BRU OK1
        BRM OKPSH; LDA OKCTR; SKG =NOKST-2; BRU OK1; BRU OK3
OK1     SKE =33B; BRU TRAG; BRM OKPSH
OK3     LDA =OKST%; STA OKPTR
OK2     LDA* OKPTR; STA OKTOGO
        MIN OKPTR; SKR OKCTR; NOP; BRR OKTOGO

        FORGT NC1,NC2

M9      TEXT '$READY TAPE ON UNIT 7, RING IN, 800BPI/'
M10     TEXT '$DISC READ ERROR /'
M11     TEXT '$TAPE WRITE ERROR /'
M12     TEXT '$DO YOU REALLY WONT TO LOAD THE DISC /'
M13     TEXT '$MOUNT DUMP TAPE ON UNIT 3, 800BPI/'
M14     TEXT '$TAPE READ ERROR /'
M15     TEXT ' TAPE SAYS /'
M16     TEXT '$DISC READ ERROR /'
M17     TEXT '$SEQUENCE CHECK /'
M18     TEXT '$TRY AGAIN/'
M19     TEXT 'UMP /'
M20     TEXT 'OAD /'
M21     TEXT '$>/'
M22     TEXT 'ISC /'
M23     TEXT 'ISC FROM TAPE/'
M24     TEXT 'TART SYSTEM/'
M25     TEXT '$$$$$$940 SYSTEM STARTUP$$OPERATOR: /'
M26     TEXT 'STAPE NUMBER: /'
M27     TEXT 'TO /'
M28     TEXT 'TO TAPE/ '
*****
LITORG BSS 340B
BMAP ZRO
*       READ PAGE 2 OF MSW0 FROM THE RAD
 DISW; LDA RADP2  RAD ADD. BD. IN BITS 4-10. PG. IN 11-12
 LDB =34000B; LDX =4000B; BRM GDAC; ZRO
*       INITIALIZE THE COUNTER OF USER DISC BLOCKS ASSIGNED
*        NAVDB STANDS FOR "NUMBER OF AVAILABLE DISC BLOCKS"
 LDA =NAVDB; STA UDBA
*       RELABEL USER CORE FOR THE BEGINNING OF BMAP
 EQM 21000B; PDT =14151617B
*       CLEAR ALL COUNTERS
 LDX =-LZRO; CLA; STA ENDZRO,2; BRX *-1
 BRM EM1; BRM EM2; BRM EM3; BRM EM4; BRM SAVER
 LDA EM1PTR; BRM PRMCT
 LDA HDA; LDB =3; BRM PRNCT
 LDA EM2PTR; BRM PRMCT
 LDA HDA; RSH 13; LDB =10; BRM PRNCT
 BRR BMAP
```

```
EM1 TEXT '$$THE HIGHEST DISC ADDRESS IS  /'
EM1PTR DATA EM1
EM2 TEXT '$IT IS ON DISC  /'
EM2PTR DATA EM2
*
*     THE CODE FROM HERE TO "EN1" DEFINES STORAGE AND SOME CONSTANTS
RADP2 DATA 270000B  BAND 11(DECIMAL) PAGE 2
BLKSIZ EQU 400B  THE SIZE OF A BLOCK ON THE DISC
*
*       THE FILE DIRECTORY, INDEX BLOCK AND DATA BLOCK BUFFERS
*        ARE KEPT IN USER CORE
FDBUF EQU 4000B;IBBUF EQU FDBUF+BLKSIZ;DBBUF EQU IBBUF+NDXW
*       THIER ABSOLUTE ADDRESSES MUST BE DEFINED FOR THE I/O ROUTINES
ABFD DATA 40000B+FDBUF;ABIB DATA 40000B+IBBUF;ABDB DATA 40000B+DBBUF
EIBLK EQU IBBUF+NDXW  END OF INDEX BLOCK
TABL EQU 6000B "TABL" IS THE BIT MAP OF THE UPPER DISC.
UDBT EQU 0;EUDBT EQU UDBT+MXUSRS  EACH CELL CONTAINS A COUNT
*         OF THE DISC BLOCKS ASSIGNED THE CORRESPONDING USER
*
*       THE STACKS ARE FOR THE SORT AT EN2.  IBT (INDEX BLOCK TABLE)
*        IS KEPT IN USER CORE. ITS MAXIMUN SIZE IS 20000B
BSTACK EQU 4000B;ASTACK EQU 14000B;IBT EQU 20000B     ;IBTSIZ EQU 20000B
*
*       ENTLEN IS THE LENGHT OF A FD ENTRY. THE OTHER SYMBOLS
*        IN THE NEXT THREE LINES IDENTIFY CELLS IN THE FD
ENTLEN EQU 7;MAXCNT EQU 356B
FDWC EQU FDBUF+367B;RGSF EQU FDBUF+370B;RESFA EQU FDBUF+371B
MXUDB EQU FDBUF+372B;FDUDB EQU FDBUF+373B
*
UNDMSK DATA 3777B;NDSMSK DATA 77000003B
*
*     THE FOLLOWING FLAGS ARE USED TO CONTROL THE PROCESSING OF
*      ENTRIES IN THE IBT
UNDFLG DATA 40B6;RESFLG DATA 20B6;IBRFFL DATA 10B6;IBDFLG DATA 4B6
MVFLG DATA 2B6;DMFLG DATA 1B6  ;KMOV DATA 15B;KDM DATA 16B;JTXDEL DATA 4
*
*       FD DATA MISSING BIT, ZERO BITS IN INDEX BLOCK ENTRIES
*        AND INDEX BLOCK END FLAG
FDDMBT DATA 4B6;ZERBTS DATA 67000003B;IBEFLG DATA 10B6
*
*       REFERENCE ADDRESS. FREE SPACE ON THE UPPER DISC IS
*        SOUGHT UPWARDS OF REFADD.
REFADD DATA TOP+4
*
```

```
*       COUNTERS AND TEMP. STORAGE
STAZRO EQU *
ERRCTR RPT 13; ZRO; ENDR;ERRTOT ZRO
UNO ZRO
NZERFD ZRO;NUSDFD ZRO;NTPFLS ZRO;NFILES ZRO;NRESFLS ZRO;NZERADS ZRO
IBA ZRO;DBA ZRO;CDA ZRO;NIBA ZRO
HDA ZRO    11/25/68  HIGHEST DISC ADDRESS
IBTX ZRO;FDX ZRO;IBX ZRO;JTX ZRO;ASTX ZRO
IBPTR ZRO
TA ZRO;TB ZRO;TX ZRO;TB4 ZRO;TTB ZRO
FLFL ZRO;WIBFLG ZRO
ASP ZRO;BSP ZRO;ALP ZRO;BLP ZRO
MB ZRO;UNMB ZRO;MVIB ZRO;MVDB ZRO
TUDB ZRO;FIBT ZRO
ENDZRO EQU *;LZRO EQU ENDZRO-STAZRO
*
*
*
*      EN1 READS THE USER FILE DRIECTORIES AND BUILDS A TABLE OF
*       USER NUMBERS AND INDEX BLOCK ADDRESSES.
*
*      RELABEL THE FILE DIRECTORY BUFFER (FDBUF) INTO USER CORE
EN1 ZRO; LDA =4011404ØB; BRM RELA
S1 MIN UNO; BRM RFD; BRU FDRF1; LDA FDWC,4; SKE =Ø; BRU INUSE
 MIN NZERFD; BRU EP1
*      THE FILE DIRECTORY(FD) FOR THE USER NUMBER IS IN USE.
*       IS IT WELL FORMED?
INUSE MIN NUSDFD; SKG =MAXCNT; SKG =-1; BRU FDNG
 RSH 23; DIV =ENTLEN; XAB; SKE =Ø; BRU FDNG; XAB; ADM NFILES
*      YES. STORE THE USER NUMBER IN THE INDEX BLOCK TABLE (IBT)
 LDX IBTX; LDA UNO; MRG UNOFLG; STA IBT,6; MIN IBTX
 LDA =-ENTLEN; STA TA
*      NOW STORE AN INDEX BLOCK ADDRESS IN THE IBT
FDOK LDA TA; ADD =ENTLEN; STA TA
 SKE FDWC,4; BRU *+2; BRU RESFQ
 CAX; SKN FDBUF+2,6; BRU TAPFIL
 LDA FDBUF+1,6; ETR DSCMSK; SKG =Ø; MIN NZERADS
EP15 LDX IBTX; STA IBT,6; MIN IBTX
 CXA; SKG =IBTSIZ-3; BRU FDOK; BRU OVERFL    IBT OVERFLOW
*      IS THERE A RESTART FILE IN THE DIRECTORY?
RESFQ LDA RESFA,4; ETR DSCMSK; SKG =Ø; BRU EP1     NO
*      YES. FLAG ITS ADDRESS AND STORE IT IN THE IBT
 MIN NRESFLS; MRG RESFLG; LDX IBTX; STA IBT,6; MIN IBTX
EP1 LDA UNO; SKE =MXUSRS; BRU S1    GET NEXT FD
*      -1 FLAGS THE END OF THE IBT
 LDA =-1; LDX IBTX; STA IBT,6; BRR EN1
*
TAPFIL MIN NTPFLS; CLA; BRU EP15     TAPE FILE. THERE SHOULDN'T BE ANY
*
*      FILE DIRECTORY READ FAIL
FDRF1 CLX; BRM GENERR; BRU EP1
*      FILE DIRECTORY NO GOOD
FDNG LDX =1; BRU *-3
*
```

```
*
*       POINTERS TO THE INDEX BLOCK ADDRESSES IN IBT ARE CREATED
*        IN ASTACK. A RADIX SORT SHUNTS THEM BACK AND FORTH
*        BETWEEN ASTACK AND BSTACK. BIT 21 IS SORTED LAST BECAUSE THE
*        SORTED TABLE IS USED TO MINIMZE DISC ROTAION TIME WHEN THE INDEX
*        BLOCKS ARE READ. (IT IS QUICKER TO READ ADDRESSES 0, 10, 20
*        ... THEN 4, 14, 24..., THAN 0,4,10,14...). ASTACK IS
*        RELABELED INTO PAGE 1, BSTACK INTO PAGE 3.
*       THE FOREGOING DESCRIBES THE CODE BETWEEN HERE AND EN3.
*
EN2 ZRO; LDA =40104020B; BRM RELA
S2 CLX; LDA =1; SKG IBT,6; BRM ASTO; BRX *+1; CXA; SKE IBTX; BRU S2+1
 LDX =-1; BRM ASTO; LDB =10B; STB TB
ZP CLA; STA ASP; STA BSP; STA ALP; STA BLP
 LDA =40104020B; BRM RELA
ZL SKN FLFL; BRM ALOAD; BRM BLOAD; CXA; SKE =-1; BRU *+2; BRU OP
 LDB TB; SKB IBT,6; BRU ZL; SKN FLFL; BRM BSTO; BRM ASTO; BRU ZL
OP CLA; STA ALP; STA BLP; LDA R1; SKN FLFL; BRU *+4
 ETR =77007777B; ADD N10B4; BRU *+3; ETR =77777700B; ADD =20B; BRM RELA
OL SKN FLFL; BRM ALOAD; BRM BLOAD; CXA; SKE =-1; BRU *+2; BRU CYEND
 LDB TB; SKB IBT,6; BRU *+2; BRU OL; SKN FLFL; BRM BSTO; BRM ASTO; BRU OL
CYEND LDX =-1; SKN FLFL; BRM BSTO; BRM ASTO; LDB TB; SKB =4; BRR EN2
 SKB =400000B; LDB =2; CLA; LCY 1; STB TB
 CLA; SKN FLFL; SKR FLFL; STA FLFL; BRU ZP
*
ASTO ZRO; CXA; LDX ASP; STA ASTACK,6; MIN ASP; XXA; SKE =3777B; BRR ASTO
 LDA R1; ADD =1; BRM RELA; CLA; STA ASP; BRR ASTO
BSTO ZRO; MIN BSTO; CXA; LDX BSP; STA BSTACK,6; MIN BSP; XXA; SKE =3777B
 BRR BSTO; LDA R1; ADD =10000B; BRM RELA; CLA; STA BSP; BRR BSTO
ALOAD ZRO; MIN ALOAD; LDX ALP; LDA ASTACK,6; MIN ALP; XXA; SKE =3777B
 BRR ALOAD; LDA R1; ADD =1; BRM RELA; CLA; STA ALP; BRR ALOAD
BLOAD ZRO; LDX BLP; LDA BSTACK,6; MIN BLP; XXA; SKE =3777B; BRR BLOAD
 LDA R1; ADD =10000B; BRM RELA; CLA; STA BLP; BRR BLOAD
*
*
EN3 ZRO
*       RELABEL ASTACK AND I-O BUFFERS INTO USER CORE. CREATE TABL, A
*       BIT MAP WITH A ONE-BIT FOR EVERY BLOCK ON THE UPPER DISC.
*       CLEAR UDBT (USER DISC BLOCK TABLE), WHICH IS TO CONTAIN COUNTERS
*       OF THE BLOCKS ASSIGNED EACH USER.
S3 LDA =10111220B; BRM RELA
 LDB =-1; LDX =-NFWDS; STB TABL+NFWDS,6; BRX *-1; STB ASTX
 LSH SHIFTER; STB TABL+LTAB-1,4
 CLA; LDX =-MXUSRS; STA UDBT+MXUSRS,6; BRX *-1
*       READ AN IB AND MAP IT AND ITS DATA BLOCKS.
*       IF WIBFLG IS INCREMENTED, THE IB WILL BE WRITTEN.
LOOP3 LDA =-1; STA WIBFLG; LDX ASTX; BRX *+1; CXA; SKE =4000B; BRU *+5
*       RELABEL IN ANOTHER PAGE OF ASTACK.     -1 FLAGS END OF ASTACK.
 MIN R1; LDA R1; BRM RELA; CLX; STX ASTX; SKN ASTACK,6; BRU *+2; BRR EN3
*       IBTX IS FOR INDEX BLOCK TABLE INDEX. IBA FOR INDEX BLOCK
*        ADDRESS. CDA FOR CURRENT DISC ADDRESS.
 LDX ASTACK,6; STX IBTX; LDA IBT,6; ETR DSCMSK; STA IBA; STA CDA
*       SEARCH BACKWARD FOR THE USER NUMBER.
 CXA; SUB =1; CAX; SKN IBT,6; BRU *-3; LDA IBT,6; ETR UNOMSK; STA UNO
```

```
 BRM RIB; BRU IBRF3; BRM IBVER; BRU IBNG; BRM TOPQ; BRM MAP; BRU IBMPF
TOPQR BRM DBSMAP; SKN WIBFLG; BRU *+2; BRU LOOP3
*       IF A DB HAS BEEN EXCISED, WRITE THE IB AND FLAG ITS ENTRY
*        IN THE IBT.
 BRM WIB; BRU IBWF3; LDB DMFLG; BRM FLGIBT; BRU LOOP3
*
*       IS THIS IB WELL-FORMED?
IBVER ZRO; LDX =-NDXW; LDB ZERBTS
BOCHK SKB EIBLK,6; BRR IBVER; BRX BOCHK; LDX =-NDXW
ZERCEL LDA EIBLK,6; SKG \=0; BRU IBEND; BRX ZERCEL; BRR IBVER
IBEND LDA =-1; SKA EIBLK,6; BRR IBVER; BRX *-2; MIN IBVER; BRR IBVER OK
*
*       IB READ FAIL. PRINT MESSAGE, RESERVE BLOCK ON DISC, AND FLAG
*       IBT ENTRY.
IBRF3 LDX =4; BRM GENERR; LDA IBA; SKG =TOP; BRU *+3; BRM DTEB; NOP
 LDB IBRFFL; BRM FLGIBT; BRU LOOP3
*       IB NOT WELL-FORMED. PREPARE TO DELETE IT BY FLAGGING ITS
*       ENTRY IN THE IBT.
IBNG LDX =5; BRM GENERR; LDB IBDFLG; BRU *-5
*       COULDN'T MAP INDEX BLOCK. PREPARE TO DELETE IT.
IBMPF LDX =6; BRU IBNG+1
*       WRITE FAIL. FREE (I.E. UNMAP) THE SPACE OCCUPIED BY THE DATA
*        BLOCKS AND PREPARE TO DELETE THE INDEX BLOCKS.
IBWF3 BRM DBFREE; LDX =7; BRU IBNG+1
*
*       CYCLE THROUGH THE TABLE OF INDEX BLOCKS. MOVE OR DELETE
*        FILES WHERE THE APPROPRIATE FLAGS HAVE BEEN SET.
EN4 ZRO; LDA =10111240B; BRM RELA  RELABEL IBT INTO USER CORE
S4 CLA; STA IBTX
*       USER LOOP.     -1 FLAGS END OF TABLE.
UL LDX IBTX; STX FIBT; LDA IBT,6; SKE =-1; BRU *+2; BRR EN4
 ETR UNOMSK; STA UNO; BRM RFD; BRU FDRF4; LDA =-ENTLEN; STA FDX
*       INDEX BLOCK LOOP
IBL MIN IBTX; LDX IBTX; LDA =ENTLEN; ADM FDX
 SKN IBT,6; BRU *+2; BRU NUSER  NEW USER (OR END OF TABLE).
*       USE FLAGS OF IBT ENTRY TO PREPARE A JUMP TABLE INDEX
EPDEL LDB IBT,6; LSH 1; STB TB4; LSH 1; CLA; LSH 4; STA JTX
 LDA IBT,6; ETR DSCMSK; STA IBA; STA CDA
GEP LDX JTX; BRU *+1,2; BRU IBL
 BRU DATMIS; BRU MOV; BRU MOV; BRU DEL; BRU DEL; BRU DEL

*       WE MIGHT WRITE CODE TO RETRY AN INDEX BLOCK READ FAIL. BUT
*        WE HAVEN'T, AND THE ROUTINE CALLED "RETRY" JUST DELETES
*        THE BAD BLOCK
 RPT 8; BRU RETRY; ENDR
*      NEW USER. UPDATE FILE DIRECTORY AND WRITE IT.
NUSER CLA; XMA TUDB; LDX UNO; ADD UDBT-1,6; STA UDBT-1,6; STA FDUDB,4
 SKG MXUDB,4; BRU *+2; STA MXUDB,4; CNA; ADM UDBA
 BRM WFD; BRU FDWF; BRU UL
*
*       DATA MISSING.  SET FLAG IN FILE DIRECTORY
DATMIS BRM DM; LDA KDM; BRU EP4
*       IF TB4 IS NEGATIVE, THIS IS A RESTART FILE. THEN DELETE IT.
DM ZRO; SKN TB4; BRU *+3; LDA JTXDEL; BRU EP4+1
```

```
 LDX FDX; LDA FDBUF+2,6; MRG FDDMBT; STA FDBUF+2,6; BRR DM
*
MOV BRM RIB; BRU IBRF4; CLX; STX IBX
 LDA IBA; SKG =TOP; BRM FDA; STA NIBA
MOVDBS STA REFADD; LDX IBX; MIN IBX; LDA IBBUF,6; ETR DSCMSK; STA DBA
 SKG =0; BRU IBQ; SKG =TOP; BRM DBAC; BRU MOVDBS
IBQ LDA IBBUF-1,6; MRG IBEFLG; STA IBBUF-1,6; LDA NIBA; SKE IBA; BRM IBAC
 BRM WIB; BRU IBWF4; LDA KMOV
EP4 ETR JTX; STA JTX; BRU GEP
DBAC ZRO; STA CDA; BRM RDB; BRU DBRF; BRM FDA; STA CDA; BRM WDB; BRU DBWF
 MIN TUDB; MIN MVDB
 LDX IBX; LDA REFADD; STA IBBUF-1,6; BRR DBAC
IBAC ZRO; STA IBA
 MIN TUDB; MIN MVIB
 SKN TB4; BRU *+4; LDA IBA; STA RESFA,4; BRR IBAC
   LDX FDX; LDA FDBUF+1,6; ETR NDSMSK; ADD IBA; STA FDBUF+1,6; BRR IBAC
*
DEL CLB; SKN TB4; BRU *+4; STB RESF,4; STB RESFA,4; BRU IBL
 LDA IBTX; SUB FIBT; MUL =ENTLEN; LSH 23; SKE FDWC,4; LDB =-1; STB FLFL
 LDB =ENTLEN-1; STB TTB
DELOOP SKR FDWC,4; NOP; CLA; LDX FDWC,4; XMA FDBUF,6; CAB
 LDA FDX; ADD TTB; SKN FLFL; BRU *+3; CAX; STB FDBUF,6; SKR TTB
 BRU DELOOP
DELEX SKN FLFL; BRU IBL  FLFL IS POS. IF THIS FILE IS THE LAST IN THIS FD
 LDA FDWC,4; RSH 23; DIV =ENTLEN; ADD FIBT; COPY AX,A; XMA IBT+1,6
 LDX IBTX; STA IBT,6; BRU EPDEL
*
RETRY BRU DM+3
*
FDRF4 LDX IBTX; BRX *+1; SKN IBT,6; BRU *-2; STX IBTX
 CLA; LDX UNO; XMA UDBT-1,6; ADM UDBA
 LDX =2; BRM GENERR; BRU UL
FDWF LDX =3; BRU *-3
IBRF4 LDX =8; BRM GENERR; LDA IBA; BRM DTEB; NOP; BRU DM+3
IBWF4 LDX =9; BRM GENERR; BRM DBFREE; BRU DM+3
DBRF LDA IBX; ADD =IBBUF-1; STA IBPTR; BRM DBDEL; BRU *+6
 LDA NIBA; STA CDA; SKG IBA; BRM UNMAP; BRU DM+3      DELETE IB
 BRM DM; SKR IBX; BRU MOVDBS+1; ZRO
DBWF LDA =-1; ADM TUDB; BRU DBRF
*
*
*
RELA ZRO; STA R1; EOM 20400B; POT R1; BRR RELA;R1 ZRO
*
 BSS 100  TO PROTECT LOCATION 33777
*
RFD ZRO; BRM GFD; LDA ABFD; LDX =BLKSIZ; BRM DTC; BRM DST
 MIN RFD S; BRR RFD
WFD ZRO; BRM GFD; LDA ABFD; LDX =BLKSIZ; BRM DTW; BRM DST
 MIN WFD S; BRR WFD
RIB ZRO; LDB IBA; LDA ABIB; LDX =NDXW; BRM DTC; BRM DST
 MIN RIB S; BRR RIB
WIB ZRO; LDB IBA; LDA ABIB; LDX =NDXW; BRM DTW; BRM DST
 MIN WIB S; BRR WIB
```

```
RDB ZRO; LDB DBA; LDA ABDB; LDX =BLKSIZ; BRM DTC; BRM DST
 MIN RDB S; BRR RDB
WDB ZRO; LDB REFADD; LDA ABDB; LDX =BLKSIZ; BRM DTW; BRM DST
 MIN WDB S; BRR WDB
GFD ZRO; LDA UNO; CLB; LSH 2; ADD =NACCTS; STA CDA; CAB; BRR GFD
*
TOPQ ZRO; LDA CDA; SKG =TOP; BRU *+2; BRR TOPQ
 LDB MVFLG; BRM FLGIBT; LDA =2; ADM TOPQ; BRR TOPQ
FLGIBT ZRO; STB TB; LDX IBTX; LDA IBT,6; MRG TB; STA IBT,6; BRR FLGIBT
*
MAP ZRO; LDA CDA; BRM DTEB; BRR MAP F
 MIN MB; LDX UNO; MIN UDBT-1,6; MIN MAP; BRR MAP
UNMAP ZRO; STX TX; LDA CDA; BRM DTE
 MIN UNMB; LDX UNO; LDA =-1; ADM UDBT-1,6; LDX TX; BRR UNMAP
*
DBSMAP ZRO; LDA =IBBUF-1; MRG =BIT0; STA IBPTR
DBMLOOP MIN IBPTR; LDA* IBPTR; ETR DSCMSK; STA CDA; STA DBA
 SKG =0; BRR DBSMAP; BRM TOPQ; BRM MAP; BRU DBMPF; BRU DBMLOOP
DBMPF BRM DBDEL; BRU DBMLOOP+1 HOLD IB
 LDA IBA; STA CDA; SKG =TOP; BRU LOOP3; BRM UNMAP; BRU LOOP3
DBDEL ZRO; LDX IBPTR; LDA 1,6; STA 0,6; BRX *+1; SKE =0; BRU *-4
 LDA -2,6; MRG IBEFLG; STA -2,6
 LDA IBBUF,4; ETR DSCMSK; SKE =0; BRU HOLDIB
DELIB LDX N10; BRM GENERR; LDB IBDFLG; BRM FLGIBT; MIN DBDEL; BRR DBDEL
HOLDIB LDX N11; BRM GENERR; MIN WIBFLG; BRR DBDEL
DBFREE ZRO; LDX =IBBUF; LDA 0,6; ETR DSCMSK; STA CDA; SKG =0
 BRR DBFREE; SKG =TOP; BRU *+2; BRM UNMAP; BRX *+1; BRU DBFREE+2
*
FDA ZRO; LDA REFADD; BRM DTA; BRU DTAERR; STA REFADD; BRR FDA
*
DTEB ZRO; BRM DTEC; BRR DTEB; LCY 0,2; ETR =(NOT)BIT0; RCY 0,2; XXB
 SKE TABL,6; MIN DTEB; STA TABL,6; BRR DTEB
DTEC ZRO; ETR DSCMSK; SKG =FAVDS-4; BRR DTEC; MIN DTEC; BRM CDBA
 CAX; LDA TABL,6; XXB; BRR DTEC
CDBA ZRO; SUB =FAVDS; ETR DSCMSK
 SKG HDA; BRU *+2; STA HDA    11/25/68
 LRSH 3; COPY BX,B; LRSH 4; XXB; LSH 1; CXB; LRSH 19; DIV N24; BRR CDBA
DTE ZRO; BRM DTEC; BRR DTE; LCY 0,2; SKA =BIT0; MIN DTECTR; MRG =BIT0
 RCY 0,2; XXB; STA TABL,6; BRR DTE;DTECTR ZRO
DTA ZRO; BRM CDBA; STA DTA3
DTA2 AXC; SKE TABL,6; BRU DTA1; BRX *+1; CXA; SKE =LTAB; BRU *+2
 LDA =WINLEN+1; SKE DTA3; BRU DTA2; BRR DTA F
DTA1 STX DTA3; CLA; LDB TABL,6; LDX N23; NOD 46; EOR X2; LCY 25,2; CXA
 CNA; LDX DTA3; STB TABL,6; COPY XA,AB; BRM CDDA
 MIN DTA S; BRR DTA      ;DTA3 ZRO
CDDA ZRO; STB CDDA1; MUL N24; LSH 23; ADD CDDA1; LRSH 4; COPY BX,B
 LRSH 1; XXB; LSH 4; CXB; LSH 3; ADD =FAVDS; BRR CDDA      ;CDDA1 ZRO
*
*
DTAERR LDA PMERDTA; BRU *+2
OVERFL LDA PMERROV; BRM PRMCT
CHAOS LDA PMERRF; BRM PRMCT; BRM SAVER; BRR BMAP
MERDTA TEXT '$UPPER DISC APPARENTLY FILLED /'
MERROV TEXT '$MORE FILES THAN BMAP PROGRAM CAN PROCESS /'
```

```
MERRF TEXT '$FATAL ERROR. CALL A SYSTEMS PROGRAMMER /'
*
GENERR ZRO; STX TX; LDA PTRS,2; BRM PRMCT
  LDA CDA; LDB =8; BRM PRNCT; BRM PRUNO; MIN ERRTOT
 LDX TX; MIN ERRCTR,2; LDA ERRCTR,2; SKG =9; BRR GENERR; BRU CHAOS
MERRØ TEXT '$$FD READ FAIL IN PART 1. LOCK OUT USER$  /'
MERR1 TEXT '$$ERROR IN FD FORMAT. LOCK OUT USER$  /'
MERR2 TEXT '$$FD READ FAIL IN PART 4. LOCK OUT USER$ /'
MERR3 TEXT '$$FD WRITE FAIL. LOCK OUT USER$  /'
MERR4 TEXT '$$IB READ FAIL IN PART 3. FILE EXCISED$  /'
MERR5 TEXT '$$ERROR IN IB FORMAT. FILE EXCISED$  /'
MERR6 TEXT '$$IB MAPPING ERROR IN PART 3. FILE EXCISED$  /'
MERR7 TEXT '$$IB WRITE FAIL IN PART 3. FILE EXCISED$  /'
MERR8 TEXT '$$IB READ FAIL IN PART4. FILE EXCISED$  /'
MERR9 TEXT '$$IB WRITE FAIL IN PART 4. FILE EXCISED$  /'
MERR1Ø TEXT '$$DB I-O FAIL OR MAPPING ERRROR. FILE EXCISED$  /'
MERR11 TEXT '$$DB I-O FAIL OR MAPPING ERRROR. FILE RETAINED$  /'
PTRS DATA MERRØ,MERR1,MERR2,MERR3,MERR4,MERR5,MERR6,MERR7,MERR8
 DATA MERR9,MERR1Ø,MERR11
PMERDTA DATA MERDTA;PMERROV DATA MERROV;PMERRF DATA MERRF
*
PRUNO ZRO; LDA =UMESS; BRM PRMCT; LDA UNO; LDB =1Ø; BRM PRNCT; BRR PRUNO
UMESS TEXT '$USER  /'
*
N1ØB4 DATA 1ØB4;N24 DATA 24;N23 DATA 23;N1Ø DATA 1Ø;N11 DATA 11
***
SAVER HLT; RSR; BRU *-1; ALR
 POT BAND32; EOD* 1ØØØØB ALERT CHANNEL; EOD 16231ØB; POT =Ø PAGES Ø-13
 WRF; RSR; BRU *-1; RSE; BRU *-1; CETE; BRU *-1; ALR
 POT BAND35; EOD* 1ØØØØB; EOD 16261ØB; POT =2B4 PAGES 14-23
 WRF; RSR; BRU *-1; RSE; BRU *-1; CETE; BRU *-1
 BRR SAVER
BAND32 DATA 1ØØØØB;BAND35 DATA 1Ø6ØØB
*
 BSS LITORG-*
 END
```