

```

3BRS IDENT 6/30/67
* ENTRY POINTS
ENTRY TFC02
ENTRY TRAPB,CPUP,IOP,NTRPB,TRPB
ENTRY T,BSX,SS01,SS02,SS03,BST,EPBPX,BSE
ENTRY MGET,MPUT,TRAP,MDEL,NTRAP,TRAPS
ENTRY MX01,MS03,MX13
ENTRY TRAP1,TRAPM,TRAPR
ENTRY RFK,TFK,FKSTW,HFK,GFK,SCFK,DFK
* ENTRIES FROM MDBG
ENTRY PB,PX,PPB,NFORK,DBTOP

```

```

*
* 'SBRM','SBRR','EXS' 6/28/66
*

```

```

SBRM POPD 17000000B,1,1,0,1
SMB STX SS03; EAX* 0; XXA; ETR ADMSK
MRG =40040000B; XMA 0; EBR =40000B; STA* 0
CXA; LDX SS03; BRR 0

```

```

*
SBRR POPD 17100000B,1,1,0,1
SRB STX SS03; LDX* 0; EAX 0,6; STX T; LDX SS03; BRR* T (TIME 40 US)

```

```

* EXECUTE INSTRUCTION AND SKIP
* INPUT; EXS Y
* Y IS LOC OF INSTRUCTION TO EXECUTE. Y MAY CONTAIN A BRM.
* Y WILL BE EXECUTED IN SYSTEM MODE.

```

```

EXS POPD 15200000B,1,1,0,1
XSP STX SS03; LDX PACPTR; SKN PGU,2; BRM TRAPB
LDX SS03; LDX* 0; STX XSP1; LDX SS03
XSP1 ZR0; BRR 0; MIN 0; BRR 0

```

```

*
* 'BRS' 9/27/65
*

```

```

* THIS ROUTINE DISPATCHES ON THE ADDRESS OF A 'BRS'
*

```

```

T ZR0;* UNIVERSAL TEMPORARY STORAGE
BSX ZR0;* BRS EXIT
SS01 ZR0;* SAVE (A)

```

```

SS02 ZR0;* SAVE (B)
SS03 ZR0;* SAVE (X)
*
BRS P0PD 17300000B,1,1,0,1
BS STA SS01; STB SS02; STX SS03; EAX* 0
LDA #40000B; ADM 0
CXA; ETR ADMSK; SKG #BSTU=BST; BRU BS1
BST BRM TRAPB (0); BRU M0N0PN; BRU M0NCLS; BRM TRAPB
BRU MPT (4); BRU SSCH; BRU SSIN; BRM TRAPB
BRU I0H (8); BRU FKST; BRU PPAN; BRU CIB
BRU CET (12); BRU SKI; BRU D0B; N0P 0
N0P 1 (16); N0P 2; N0P 3; N0P 4
N0P 5 (20); BRU FNA; BRM TRAPB; BRU LNKS
BRU LNKC (24); BRU MSGS; BRU SKR0UT; BRU ASTT
BRU RSTT (28); BRU C0B; BRU FKRD; BRU FKWT
BRU FKTM (32); EAX GETSTR; EAX 0UTMSG; EAX 0UTSTR
EAX 0UTNUM (36); EAX GSL00K; EAX GETNUM; BRM TRAPB
BRU RDET (40); BRU I0RET; BRU RREAL; BRU RDRL
BRU STRL (44); BRU SQ0; BRU NR0UT; BRU SR0UT
N0P 6 (48); BRU SRIR; BRU FFIX; BRU FFLT
N0P 1002B (52); N0P 1003B; BRU RRSB; BRU MRSB
BRU MBEX (56); BRU C00; BRU SSMF; BRU CBRF
N0P 9 (60); BRM TRAPB; BRM TRAPB; BRM TRAPB
BRM TRAPB (64); BRM TRAPB; BRU DFDL; BRU DFER
BRU EBSM (68); BRU GBSM; BRM TRAPB; BRU SKXEC
BRU EXDMS (72); BRU EPPAN; BRU FSWT; BRU FSFZ
BRU FSMT (76); BRU FSTM; BRU SAIR; BRU SIIR
BRU MBR0 (80); BRU WREAL; BRU SWSF; BRM TRAPB
BRM TRAPB (84); BRU SET8P; BRU CLR8P; BRU DFRX
BRU RTEX (88); BRU FSCF; BRU DFR; N0P 1004B
BRM TRAPB (92); BRM TRAPB; BRM TRAPB; N0P 7
N0P 8 (96); BRM TRAPB; BRM TRAPB; BRM TRAPB
BRM TRAPB (100); BRM TRAPB; BRM TRAPB; BRM TRAPB
BRU RSYB (104); BRU WSYB; BRU FKWA; BRU FKRA
BRU FKTA (108); BRU DMS; BRU RDU; BRU BRSRET
BRU TS0FF (112); BRU DFCD; BRU MTDI; BRM TRAPB
BRU RURL (116); BRU SURL; BRU TGET; BRU TREL
BRU APMTE (120); BRU DPMTE; BRU MPAN; BRM TRAPB

```

```

IF      V1
BRU ARD (BE+1,124); BRU AWD; BRU CARRY; BRU PEBRS
BRU SDBM (BE+5,128); BRU TTYON; BRU BPTST; BRU CRASH
BRU RDSYB (BE+9,132); BRU WDSYB; BRU CRSW; BRU TIMINT
BRU SETSW (BE+13,136); BRM TRAPB; BRU RDPGE; BRU MFSYS
BRU CKBUF (BE+17,140); BRM TRAPB; NBP 10
ENDF

```

```

BSTU   BRM TRAPB
BS1    LDA BST,2; STA BSX; SKA =2000000B; BRU BSE
        LDA SS01; LDX PACPTR; EXU BSX (G8 IF DIRECT)

```

```

*
* CHECK FOR CLASS 2 BRS
BSE    LDX 0; STX SBRST; SKA =1000000B; BRU BSE2

```

```

*
* CLASS 3 BRS SETUP AND RETURN ROUTINE
* EXEC BRS'S

```

```

UBRSET LDA NFORK; SKG =0; BRM TRAPB
        BRM GFK; BRU FKSTW; BRM STFK; MRG X6; LDX FK04 (NEW PACPTR)
        STA PQU,2; LDA BSX; RCY 9; ETR =37B; CAX; LDA BRSTV,2
        RSH 12; STA T; LDA =4001B; LCY 12; STA 0
        LDX PACPTR; LDA RL1,2; STA UBRL1; LDB RL2,2; STB UBRL2
        LDA =NCMEM*1000000B+10000B; MRG T; CLB
        LDX FK04; STA RL1,2; STB RL2,2; LDA SS01; STA PA,2; STA UBA
        LDA BSX; ETR =777B; ADD 0; STA PL,2
        LDA =EXECL; RSH 15; LDA JOB; LSH 15; MRG X5; STA PTAB,2
        LDX STFK2 (NEW XPB)
        LDA SS02; STA PB,2; STA UBB; LDA SS03; STA PX,2; STA UBX
        LDA SBRST; STA 0; STA UPL; LDB =700004B; BRU PBPST

```

```

*
* CLASS 2 MONITOR BRS'S
* SAVE RETURN
BSE2   LDX =UBRSET; STX 0; LDA SS01; LDX SS03; BRU* BSX
*
*
* RETURN FROM CLASS 2 BRS'S

```

EP0PX SKN TIME; BRR SBRST  
SKN TTIME; SKN ACTR; BRU \*\*2; BRR SBRST  
STA SS01; STB SS02; STX SS03; LDA SBRST; STA 0; BRU SQ0

\*  
\*

\* BRS 111 RETURN FROM EXEC BRIS (CLASS 3/4)  
BRSRET LDA PPTR,2; MRG PLMSK; CAX  
LDA PTEST,2; SKE =700004B; BRM TRAPB; SUB =3; STA PTEST,2  
LDA UPL; STA 0; STX TFO1; LDX PACPTR; BRM DFK  
LDX TFO1; LDA PIM,2; LRSH 3; ETR =7; MRG X4  
STA XPB; CLB; STB EXEC1; LDB =-1; LDA PTAB,2; SKA X2; STB EXEC1  
STX PACPTR; BRM CHRL; BRM TRAPB; BRU P0PX

\* BRS 46 SET NON-TERMINABILITY  
NR0UT SKN PQU,2; BRM TRAPB; LDA PIM,2; MRG X1; STA PIM,2; BRU P0PX  
\* BRS 26 SKIP IF TERMINATION PENDING  
SKR0UT LDA PIM,2; SKA X2; MIN 0; BRU P0PX  
\* BRS 47 CLEAR NON-TERMINABILITY  
SR0UT SKN PQU,2; BRM TRAPB; LDA PIM,2; SKA X1; BRU \*\*2; BRU P0PX  
ETR =4777777B; STA PIM,2; BRU PACQE  
\* BRS 90 DECLARE FORK FOR RUB0UT  
DFR CXA; LDX UTTY; STA TTYASG,2; BRU P0PX

\*  
\*  
\*

MEMORY ALLOCATION LOGIC

\* ASSIGN A POSITION IN PMT  
PMGET ZR0; LDX J0B; LDA PMA,2; SKA =27700000B; BRU \*\*2; BRR PMGET  
ETR =77B; STA SMGET; CLA; LDX =NCMEM-1; BRU MGET4  
MGET1 EAX 1,2; SKE\* PMTJ0B; BRU MGET4; MIN PMGET; BRR PMGET  
MGET4 SKR SMGET; BRU MGET1; BRR PMGET  
\* ASSIGN A POSITION IN SMT  
SMGET ZR0; ABC; LDX =NSMT  
SMG1 SKE SMTE,2; BRU SMG2; CXA; ADD =NSMT; MIN SMGET; BRR SMGET  
SMG2 BRX SMG1; BRR SMGET



```

*
*
* GET A BLOCK OF MEMORY
* INPUT; A=CORE ADDRESS IN PAGE, INDIRECT BIT IF MEMORY MUST
* COME FROM AN UPPER FORK.
* B=NUMBER TO ADD TO PMA, X=PACPTR
MGET ZR0; BRM MX01; SKE =0; BRU MX03
LDA -=2; STA MX09; LDX MS03
* SCAN FOR LOCAL OR FIXED MEMORY FORK
MGET11 STX MX08; LDA RL1,2; LDB RL2,2; LDX MX07; MIN MX09
LCY 0,2; SKA =77000000B; BRU MGET12; LDX MX08
SKN PIM,2; BRU +=2; BRU MX03; SKN PTAB,2
BRU MGET13 (CHECK HIGHER FORKS FIRST)
LDA PMTA; SKA =40000B; BRU MX03
MIN MX09; BRM PMGET; BRU MX03
STX MGTS2; BRM PMTA; BRU MX03; LDX MS03
* PROPAGATE NEW BYTE AS NECESSARY
MGET10 STX MX08; LDA RL1,2; LDB RL2,2; LDX MX07
LCY 0,2; RCY 18; MRG MGTS2; LCY 18; RCY 0,2
LDX MX08; STA RL1,2; STB RL2,2
LDA PPTR,2; MRG PLMSK; CAX; SKR MX09; BRU MGET10
* SET UP NEW MAP AND CLEAR BLOCK
LDX MS03; BRM CHRL; BRU MX03; LDX MS01; EAX 4000B,6
COPY XA,B; MRG =23600000B; STA MGET14; LDX -=4000B
MGET14 STB 4000B,6; BRX MGET14; BRU MX04
MGET12 LRSH 18; LDX MS03; BRU MGET10
MGET13 LDA PPTR,2; MRG PLMSK; CAX; BRU MGET11; BRU MX03
*
* ASSIGN BLOCK IN PMT
PMTA ZR0; STX MGTS1; CLEAR; SKN MGTS5; BRU MGET3 (ASSIGN TS BLOCK)
LDA RMAP
SKE =0; BRU MGET9; BRR PMTA (NO ROOM)
MGET9 LDX -=1; N0D 24; CXA; CNA (PAGE NO.); COPY AX
LDA X4; LRSH 0,2; CXB; E0R =-1; ETR RMAP
STA RMAP; COPY BX,B
MGET3 LDA J0B; SUB =1; LSH 4; CNA; RCH 1200B (CNA,CXA)
LSH 5 (RAD ADDR.); ADD =NSSP; LSH 1
MRG X4; LDX MGTS1; STA* PMTJ0B; LDX J0B; LDA PMA,2
SKG X2; ADD MS02; STA PMA,2; MIN PMTA; BRR PMTA

```

```

*
*
* RELEASE MEMORY
MPUT  ZR0; BRM MPUTA; BRU **2; MIN MPUT; BRR MPUT
*
MGTS1  ZR0
MGTS2  ZR0
*
*
*
MPUTA  ZR0; BRM MX01; SKG =NCMEM-1; BRM TRAPB
      CAX; LDA* PMTJ0B; SKE =0; SKA X1; BRM TRAPB
      SKA X2; BRU **2; BRU MPUT2
      CXB; LDX MS03; SKN PQU,2; BRM TRAPB; CBX
MPUT2  BRM MDEL; BRU MX13
*
* INPUT; X=RELABELING BYTE.
MDEL   ZR0; STX MDEL1; LDX =-NPAC*NPPAR
MDEL4  STX MDEL2; LDA PTAB,2; RCY 15; ETR =77B
      SKE J0B; BRU MDEL7+1; LDA RL1,2; LDB RL2,2
      LDX J0B; LDX RL3,2; BRM UPRL
      LDX =-10; LDA MDEL1; STA MDEL3
MDEL5  SKE SRTE,2; BRU MDEL6; STX MDEL3
      CLA; STA SRTE,2; LDA MDEL1
MDEL6  BRX MDEL5; SKN MDEL3; BRU MDEL7; BRM PTRL; STX MDEL3
      LDX MDEL2; STA RL1,2; STB RL2,2; LDX J0B; LDA MDEL3; STA RL3,2
MDEL7  LDX MDEL2; EAX NPPAR=1,6; BRX MDEL4
      LDX J0B; LDA PMA,2; SKG X2; ADD =100000B-100B; STA PMA,2
      LDA MDEL1; AXC; XMA* PMTJ0B; SKA X4; BRU MDEL8
MDEL9  CAB; ETR =37B; C0PY AX,BA,B; STB RMT,2; STB RMA,2
MPUT3  RSH 6; ETR =17B (PAGE N0.); SKE =0; BRU **2
      BRU MPUT4 (TS BLOCK); CAX; LDA X4; LRSH 0,2
      MRG RMAP; STA RMAP
MPUT4  BRM RSAM
      LDX PACPTR; LDA RL1,2; LDB RL2,2; LDX J0B; LDX RL3,2
      BRM SWAP; BRM TRAPB; STX BRRL3
      BRM LABEL; BRR MDEL
MDEL8  SKA X1; BRU **2; BRU MPUT3; C0PY AX,B
      LDA SMT,2; SKA =17600000B; BRU MDEL10; STB SMT,2; BRU MDEL9

```

MDEL10 SUB =200000B; STA SMT,2; BRR MDEL  
\*MGTS5 DATA =1 TS BLOCK ASSIGNMENT FLAG.

\*  
MDEL1 ZR0  
MDEL2 ZR0  
MDEL3 ZR0

\* COMMON ENTRY FOR MGET,MPUT,CHKR0  
\* GET RELABELING BYTE INTO A  
\* INPUT; A=ADDR. IN PAGE, X=PACPTR  
MX01 ZR0; STA PMTA; ETR =34000B; STA MS01; STB MS02; STX MS03  
LDB =1,2; STB MX00; LRSB 11; STA MX05; MUL =3  
LDB =1,2; STB MX00; LRSB 11; STA MX05; MUL =3  
LDA RL1,2; LDX RL2,2; XXB; STX MX07  
RCY 18; LCY 0,2; ETR =77B; STA MX06; BRR MX01

\*  
MX13 LDX MS03; BRM CHRL; BRU MX03  
MX04 MIN MX00  
MX03 LDA MS01; LDX MS03; BRR MX00

\*  
MS01 ZR0 0 ADDR. OF 1ST WORD IN PAGE  
MS02 ZR0 0 CONSTANT FOR MODIFYING PMA IN PMTA  
MS03 ZR0 0 PACPTR  
MX00 ZR0 0 RETURN FOR MGET OR MPUT  
MX05 ZR0 0 PAGE NUMBER  
MX06 ZR0 0 RELABELING BYTE  
MX07 ZR0 0 RELABELLING BYTE INDEX

\*  
MX08 ZR0  
MX09 ZR0

\*  
\* 'MBEX', 'MBR0', 'APMTE', 'DPMTE', 'MPT' 6/30/66  
\*  
\* BRS'S FOR MODIFYING THE MEMORY TABLES  
\*  
\* BRS 56 MAKE BLOCK EXEC  
\* INPUT; A=PMT NO. IF BIT 0 OF A=1, MAKE BLOCK EXEC.

```

*           IF BIT 0 OF A=0, MAKE BLOCK NOT EXEC.
* OUTPUT: BIT 0 OF A=1 IF BLOCK WAS FORMERLY EXEC.
MBEX  SKN PQU,2; BRM TRAPB; ETR =77B; BRM CRTA; BRM TRAPB; MRG X2
      SKN SS01; EOR X2; XMA 0,2; LCY 1
MBEX1  ETR X4; XMA SS01; ETR =77B; ADM SS01; BRU P0PX
*
* BRS 80  MAKE BLOCK READ ONLY
* INPUT: A=PMT NO. IF A LT 0, MAKE FILE R0. OTHERWISE, READ-WRITE
* OUTPUT: A=FORMER STATE OF PMT ENTRY
MBR0  ETR =77B; STA MBR06; BRM CRTA; BRM TRAPB; STA CRTA
      SKA X2; BRU MBR01; LDA MBR06; SKG =NCMEM-1; BRU MBR01
MBR02  LDA 0,2; SKE =0; BRU **2; BRM TRAPB; SKN SS01; BRU MBR03
      SKA =40B; BRU MBR05; MRG =40B; STA 0,2
      SKA X4; BRU MBR05; BRM 0MW; BRM RTS
      SKN NRCL; BRU *=1; BRU MBR04
MBR03  ETR =(NOT)40B; STA 0,2; SKA X4; BRU MBR05
MBR04  LDX PACPTR; BRM CHRL; BRM TRAPB
MBR05  LDA CRTA; LCY 18; BRU MBEX1
MBR01  XXB; SKN PQU,2; BRM TRAPB; CBX; BRU MBR02
MBR06  ZR0 0
*
* BRS 120  ADD A PAGE FOR SPECIFIED PMT ENTRY.
* INPUT: A=REL. BYTE
APMTE  SKN PQU,2; BRM TRAPB; SKG =NCMEM-1; BRM TRAPB; AXC
      SKE* PMTJOB; BRM TRAPB; LDB =-100000B+100B; STB MS02
      LDA =TRAP=1; STA MX00; BRM PMTA; BRU MTRAP; BRU P0PX
*
* BRS 121  DELETES PMT ENTRY
* INPUT: A=REL. BYTE
DPMTE  ETR =77B; SKG =0; BRU P0PX; SKG =NCMEM-1; BRM TRAPB (SMT ENTRY)
      STA MDEL; BRM CRTA; BRU P0PX
      CBX; SKA X2; BRU DPMTE1 (EXEC PAGE)
DPMTE2  LDX MDEL; BRM MDEL; BRU P0PX
DPMTE1  SKN PQU,2; BRM TRAPB; BRU DPMTE2
*
* BRS 4  DELETES A PMT ENTRY. REMOVES PMT POINTER
* FROM ALL OTHER FORKS.
* INPUT: A=ADDR. IN PAGE TO RELEASE.
MPT  BRM MPUT; BRM TRAPB; BRU P0PX

```

```

*
* COMPUTE RELABELING TABLE ADDRESS
* INPUT: A=PSEUDO=REL, BYTE
* OUTPUT: A=PMT ENTRY, B=INPUT X, X=PMT/SMT ADDRESS
* RETURN SKIPS IF PMT IS NOT EQUAL ZERO.
CRTA  ZR0; COPY XB,AX; SKG =NCMEM=1; BRU CRTA1
      EAX* PMTJ0B; LDA 0,2; SKA X1; BRU CRTA1
CRTA2  SKE =0; MIN CRTA; BRR CRTA
CRTA1  ADD =SMT; CAX; LDA 0,2; BRU CRTA2
*
* 'RDRL','RURL','STRL','SURL' 6/28/66
*
* READ AND SET RELABELING
*
* BRS 43  READ FORK RELABELING
RDRL  LDA RL1,2; LDB RL2,2; LDX SS03; BRU XP0P
*
* BRS 116  READ PROGRAM RELABELING FROM TS BLOCK
RURL  SKN PQ0,2; BRM TRAPB; LDA UPRRL1; LDB UPRRL2; LDX SS03; BRU XP0P
*
* BRS 44  SET FORK RELABELING
STRL  LDA X4; SKN PQ0,2; BRU **2; BRU **4
      LDA X6; SKN EXEC1; LDA X2; CAX
STRL2 LDA SS01; BRM SCRL; BRM TRAPB; LDA SS01; LDB SS02
      STA RL1,2; STB RL2,2; BRM CHRL; BRM TRAPB; BRU P0PX
*
* BRS 117 SET PROGRAM RELABELING IN TS BLOCK
SURL  SKN PQ0,2; BRM TRAPB; AXC; SKE UEXFLG; BRU **3
      LDA X2; BRU **4; LDA X4; SKN UEXFLG; MRG X2; XXA
      BRM SCRL; BRM TRPB; LDA SS01; LDB SS02
      STA UPRRL1; STB UPRRL2; BRU P0PX
*
* UNPACK RELABELING AND CHECK FOR LEGALITY.
* INPUT: A=RL1, B=RL2
* INPUT: X=X2 FOR USER, X=X6 FOR SUBSYSTEM, X=X4 FOR SYSTEM
* OUTPUT: X=PACPTR
* NO SKIP: A PMT ENTRY=0 OR USER TRIED TO RELABEL AN EXEC PAGE.
* SKIP: RELABELING OK.
SCRL  ZR0; STX STRL5; XXA; ETR X2; XXA

```

```

        STX STRL4; CLX; BRM UPRL
        LDX **10
STRL3  LDA SRTE,2; SKG =NCMEM-1; BRU STRL6
STRL7  BRM CRTA; BRR SCRL; CBX
        SKA STRL4; BRR SCRL; BRX STRL3; MIN SCRL; LDX PACPTR; BRR SCRL
STRL6  SKN STRL5; BRU **2; BRU STRL7; SKE =0; BRR SCRL; BRU STRL7
STRL4  ZR0 0 0 FOR EXEC
STRL5  ZR0 0 POSITIVE FOR USER STATUS

```

```

*
* 'EBSM', 'GBSM' 7/15/66
*
* BRS'S FOR MODIFYING SMT
*
* BRS 69 PUT SMT POINTER INTO PMT
* INPUT: A=SMT NUMBER
* OUTPUT: A=PMT NUMBER

```

```

GBSM   SKN EXEC1; BRM TRAPB; LDX PACPTR; LDA SS01
        SKG =NSMT-1; SKG =0; BRM TRAPB; COPY XB,AX; LDA SMT,2
        SKE =0; BRU **2; BRM TRAPB; SKA X2; BRU GBSM1
GBSM2  BRM PMGET; BRU MTRAP; CXA; XMA SS01
        IF      V1
        ADD X5
        ELSF   1
        ADD X6
        ENDF
        LDX SS01; STA* PMTJOB; CAX
        LDA =200000B; ADM SMT,2; BRU PBPX
GBSM1  XXB; SKN PQU,2; BRM TRAPB; XXB; BRU GBSM2

```

```

*
        IF      =V8
* BRS 68 PUT PMT ENTRY INTO SMT
* INPUT: A=PMT REL. BYTE
* OUTPUT: A=SMT REL. BYTE
EBSM   SKN PQU,2; BRM TRAPB; BRM SMGET; BRU MTRAP; XMA SS01
        XXA; ADD EBSM3; XMA* PMTJOB; LDX SS01; STA SMT,2; BRU PBPX
        IF      V1
EBSM3  ZR0 NSMT,1
        ENDF
        ELSF   1

```

EBSM3 ZR0 NSMT,6  
ENDF

```
*
*   TRAP ROUTINES
*
* SIMULATED RUBOUT
* TERMINATES THE SPECIFIED NO. OF FORKS UP TO THE
* FIRST EXEC FORK.
* INPUT: A=NO. OF FORKS TO TERMINATE.
* BRS 73
EPPAN SUB =1; STA TFK
EPPAN1 SKR TFK; BRU **2; BRU PPAN; STX PACPTR
      LDA PPTR,2; MRG PLMSK; COPY XB,AX
      SKN PQU,2; BRU EPPAN1; STB PACPTR
* BRS 10
PPAN  LDA 0; CLB; BRM PTRAP
*
*
* BRS 71  SKIP IF EXFLG SET. SET B=UEXFLG VALUE.
* UEXFLG VALUE:  1    0    -1  -2
* EXEC TYPE:    1  NEITHER  BOTH  2
SKXEC  LDB UEXFLG; STB SS02; SKN UEXFLG; BRU P0PX; MIN 0; BRU P0PX
*
* BRS BE+16  MAKE FORK SYSTEM STATUS
MFSYS  SKE =76543210B; BRM TRAPB
      LDX J0B; LDA =14000000B; SKA CPARW,2; BRU **2; BRM TRAPB
      LDX PACPTR; LDA X4; MRG PQU,2; STA PQU,2; BRU P0PX
*
*
* ILLEGAL INSTRUCTION TRAP
TRAPI  ZR0; STA SS01; STB SS02; STX SS03; LDA TRAPI; LDB =1; BRM PTRAP
*
*GENERAL TRAP LOGIC
PTRAP  ZR0; LDX PACPTR; SKG =-1; BRU **2; LDA 0; ETR =50037777B
      STA PL,2; LDA SS01; STA PA,2; SKN XPB; BRM M0NCR
```

```

LDX XPB; LDA SS02; STA PB,2
LDA SS03; STA PX,2; LDX PACPTR
BRM RFK; BRM TFK; BRU PACG0
*ILLEGAL SYSP0P EXIT
TRAPS ZR0; BRM TRAPB
NTRPB ZR0 0
NTRAP BRM MPPACT; BRM TRAPB
TRPB BSS 0
TRAPB ZR0 0
TRAP LDA 0; LDB =1; BRM PTRAP
*READ-ONLY TRAP
TRAPR ZR0; STA SS01; STB SS02; STX SS03; LDA TRAPR; STA 0; BRU MTRAP
*MEMORY TRAP
TRAPM ZR0; STA TX01; STB TX02; STX TX03
LDA TRAPM; ETR =50037777B; STA TX00; BRM CAE; MRG =40000B
LDX PACPTR; LDB =-100000B+100B; BRM MGET; BRU TX05
R0V; LDA TX00; LCY 1; LSH 1
LDA TX01; LDB TX02; LDX TX03; BRU* TX00
TX05 LDA TX01; STA SS01; LDA TX02; STA SS02; LDA TX03; STA SS03
LDA TX00; SKA X4; STA 0
MTRAP LDX PACPTR; LDA =1000000B; BRM IIR; BRU MTPAN; BRU P0PINT
MTPAN LDA 0; LDB =2; BRM PTRAP
TX00 ZR0
TX01 ZR0
TX02 ZR0
TX03 ZR0
* COMPUTE EFFECTIVE 0UT-0F-00UNDS ADDRESS
* INPUT; A=TRAP L0C.
* OUTPUT; A=ADDRESS 0R X4 IF P0P CAUSED TR0UBLE
* N0 SKIP; P0P 0R P=L0C. BRANCHED T0.
* TIME = 114 + N CY
CAE ZR0; STA CAE1; STX CAE3
CAE11 BRM CEX; LDA* CAE1; BRU CAE4
* ADDRESS F0UND
CAE5 LDA CAE1; BRR CAE
* ADDRESS N0T F0UND
CAE4 LDA CEX3; STA CAE2; SKA X1; BRU CAE6; LDA IABIT; ADM CAE1
LDX CAE3; BRM CEX; EAX* CAE1; BRU CAE7
* INDIRECT ADDRESS CHAIN 0UT-0F-00UNDS

```



```

LDA **40000B; ADM CAE1; BRU CAE9
CAE8 LDA CAE1; ETR X4; MRG CEX3; ETR #60037777B; STA CAE1
CAE9 LDX CAE3; BRM CEX; LDA* CAE1; BRU CAE8; BRU CAE5
* POP CAUSED TROUBLE
CAE6 LDA X4; BRR CAE
* CHECK FOR EXU OTHERWISE X CONTAINS BAD ADDRESS
CAE7 CXA; ETR #40037777B; STA CAE1; LDA CAE2; EBR EXUW
SKA #17700000B; BRU CAE10; LDX CAE3; BRU CAE11
* NOT EXU
CAE10 MIN CAE; BRU CAE5

```

```

CAE1 ZR0
CAE2 ZR0
CAE3 ZR0
EXUW EXU 0
IABIT ZR0* 0

```

```

* EXECUTE NEXT INSTRUCTION AND SKIP IF OUT-OF-BOUNDS
* TIME = 31 + N CY
CEX ZR0; MIN CEX; STA CEX1; LDA CEX2; XMA 41B; STA CEX3; EXU* CEX
CEX4 XMA CEX3; STA 41B; LDA CEX1; BRR CEX
* OUT-OF BOUNDS IF WE COME HERE
CEX2 BRU **1; MIN CEX; BRU CEX4

```

```

CEX1 ZR0 0 SAVE A
CEX3 ZR0 0 CONTENTS OF A AS A RESULT OF EXECUTING NEXT INSTR.

```

```

* BRS 122
* SIMULATE MEMORY TRAP FROM EXEC BRS.
* INPUT: A=CORE ADDR.
MPAN LDA PQU,2; EBR X6; SKA X6; BRM TRAPB
LDA PPTR,2; MRG PLMSK; CAX; LDA SSQ1
LDB **100000B+100B; BRM MGET; BRU **2; BRU POPX
* MEMORY TRAP
LDA SSQ1; STA UPL; BRU MTRAP

```

```

*
*

```

```

IF V1
* PARITY INTERRUPT ROUTINES
CPUP ZR0; STA CPA; STB CPB; STX CPX; SKN CPUP; BRM MCR
MIN CPUPC; LDA CPUP; STA TRAPI; LDA CPA; BRI **1
BRU TRAPI+1
*
I0P ZR0; E0D 12000B; PIN I0PI; STA IPA; STX IPX
MIN I0PC; SKN I0P; BRU **2; BRM MCR (USER MODE)
LDA I0PI; STA* I0PP; MIN I0PP; LDA EI0PL; LDX =I0PL
SKG I0PP; STX I0PP; LDA IPA; LDX IPX; BRI I0P
CPA ZR0
CPB ZR0
CPX ZR0
IPA ZR0
IPX ZR0
I0PP DATA I0PL
I0PL BSS 15
EI0PL DATA *
I0PI ZR0
*
MCR ZR0; CKF; DIR; BRU* *
*
*
*
* PEBRS BRS BE+4
* READS OF SETS A WORD IN THE MONITOR.
* INPUT; X=LOC. OF WORD
* READS IF B IS POSITIVE.
* SETS WORD TO CONTENTS OF A IF B IS NEGATIVE.
* OUTPUT; A=FORMER CONTENTS OF WORD.
PEBRS SKN EXEC1; BRU **2; BRU **3; SKN PQU,2; BRM TRAPB
LDA SS03; SKG =177777B; SKG =-1; BRM TRAPB
RSH 11; ETR =37B; MRG =600B; XMA RRL3; STA PEB2
LRR3; P0T RRL3; CLA; LSH 11
LDB =77B; SKB RRL3; ADD =34000B
SKN PQU,2; BRU PEBRS1; SKN SS02; BRU PEBRS1
CAX; LDA SS01; XMA 0,2
PEBRS2 STA SS01; LDA PEB2; XMA RRL3; LRR3; P0T RRL3; BRU P0PX
PEBRS1 CAX; LDA 0,2; BRU PEBRS2

```

```

PEB2  ZR0 0  TEMP STORAGE FOR RRL3
*
*
* SETSW  BRS BE+13
* SET SWITCHES FOR MONITOR AND EXEC
* INPUT; A=NEW SWITCH VALUE, X=SWITCH NUMBER,
* OUTPUT; A=OLD SWITCH VALUE.
SETSW  SKN PQU,2; BRM TRAPB; LDX SS03; XMA SWEX,2
      STA SS01; BRU P0PX
*
*
* BRS BE+15  READ PAGE FROM RAD
* INPUT; A=SMT POINTER
* OUTPUT; B=RAD ADDRESS OR UNCHANGED
RDPGE  SKN PQU,2; BRM TRAPB; COPY AX,B; SKG =NCMEM-1
      SKG =0; BRM TRAPB; SKN SMT,2; BRU **2; BRU P0PX
      LDA =RTC; STA RSYBT1; LDA SMT,2; ETR =177700B
      LRSH 1; STA SS02; LDA SMT,2; ETR =37B; LSH 11; BRU RSYB2
*
*
      ENDF

* TRAPPED MONITOR FEATURES
      TRP  SSMF,CBRF,SWSF
      TRP  ASTT,RSTT,LNKS,LNKC,MSGG,EBSM
      TRP  FSWT,FSTM,FSFZ,FSMT,FSCF,RRSB,MRSB
* TRAPPED EXEC FEATURES
      TRP  DNSM,ENSM,GNSM,ESSF,GSSF,HELPS,HELPM
      TRP  ECCOPY,ECSAVE,ECPLAC,ECFNDU,ECCSLT
      TRP  RGDEF,RGDEL,SGDEF,SGDEL

*
* FORK LOGIC
*
* FIND HIGHEST FORK IN STRUCTURE
HFK  ZR0
HFK1  LDA PPTR,2; SKA PRMSK; BRU **2; BRR HFK
      MRG PLMSK; CAX; BRU HFK1
* GET FORK ENTRY. PUT PPB INTO PIM.

```

```

GFK   ZR0; LDA FPLST; SKG =0; BRR GFK
      SUB =PPTR; COPY AX,A
      XMA PPTR,2; STA FPLST; MIN GFK; BRR GFK
* PUT NEW FØRK ØN QIØ
* SET PØWN(ØLD)=NEW, PØWN(NEW)=0
*   PFØRK(NEW)=ØLD, PPAR(NEW)=PØWN(ØLD)
STFK  ZR0; STX FK04; SKR NFØRK; BRU **2; BRM MØNCR
      LDX PPB; LDB PB,2; STB PPB; CXB; STB STFK2
      LSH 3; LDX FK04; STB PIM,2; CXA
      LDX =QIØ; BRM QPUT; LDX PACPTR; LSH 12
      ETR PLMSK; XMA PPTR,2; CAB; ETR PRMSK; ADM PPTR,2
      LDA FK04; XXA; ETR PRMSK; STA PPTR,2; LDA PACDMB; STA PTEST,2
      LDX PACPTR; CLA; RSH 3; LDA QUTAB; LSH 15; BRR STFK
STFK2 ZR0 0   XPB FØR NEW FØRK
* DELETE PAC ENTRY WHOSE PACT PTR IS GIVEN IN X
DFK   ZR0; STX DF01; LDA =700000B; BRM QSCH
* REMØVED FRØM QUEUE IF DISMISSED
      LDA PPTR,2; SKA PRMSK; BRU **2; BRR DFK (EXEC NOT DELETED)
      MIN NFØRK
* REMOVE PB PØINTER
      LDA PIM,2; RSH 3; ETR =7; LDX PPB; XXA
      STA PB,2; STX PPB; LDX DF01; LDA PPTR,2
      MRG PLMSK; CAB; LDA =1; SKA PIM,2; BRU DF06 (INT.)
DF08  CXA; LDX UTTY
* PFØRK IN B, PACT PTR IN A
      SKE TTYASG,2; BRU **2; STB TTYASG,2 (PROPAGATE RUBØUT 'UP')
* PUT PFØRK IN X, PØWN(PFØRK) IN A
      CBX; LDA PPTR,2; RSH 12; SKE DF01; BRU DF02
      COPY XB,AX; LDA PQU,2; SKA PRMSK; BRU DF03
* LAST FØRK ØN PØWN TØ BE CLEARED= PUT PFØRK IN X, PACT PTR IN B
      XXB; LDA PPTR,2; ETR PRMSK; STA PPTR,2; COPY BX,BA
* PUT PACT ENTRY ØN FPLST
DF04  ADD =PPTR; XMA FPLST; STA PPTR,2; BRR DFK
* MOVE PØWN ØVER
DF03  COPY BX,B (NEW PØWN IN A,PFØRK IN X); LCY 12
      XMA PPTR,2; ETR PRMSK; ADM PPTR,2
DF05  LDX DF01; CXA; BRU DF04
* SEARCH PPAR THEN REMOVE
DF02  CAX; LDA PQU,2; MRG PLMSK; SKE DF01; BRU DF02

```

```

COPY XB,AX; LDA PQU,2; ETR PRMSK; CBX
XMA PQU,2; ETR PLMSK; ADM PQU,2; BRU DF05
IF      V5
* REMOVE INTERRUPT FROM PU.
DF06   EOR PIM,2; STA PIM,2; LDX PUBPTR
DF09   CXA; SKE *PUBPTR; BRU DF07; LDX DF01; LDA PPTR,2
      MRG PLMSK; BRU DF08
DF07   LDX 0,2; LDA 3,2; ETR *77B; SKE UTTY; BRU DF09
      LDA 1,2; SKE DF010; BRU DF09
      LDA 3,2; RSH 12; MRG PLMSK; SKE DF01; BRU DF09
      LDA PUDEAD; STA 1,2; BRU DF09
DF010  7 PUTIM
      ENDF
DF01   ZR0
* SEARCH SUBSIDIARY FORK STRUCTURE FOR SPECIFIED FORK
*   SKIP IF PRESENT, NO SKIP IF TERMINATED
SHFK   ZR0; LDA PPTR,2; SKA PLMSK; BRU *+2; BRR SHFK; RSH 12
SHF01  CAX; LDA PTAB,2; EOR SS01; SKA ADMSK; BRU *+3
      MIN SHFK; BRR SHFK
      LDA PQU,2; SKA PRMSK; BRU *+2; BRR SHFK; MRG PLMSK; BRU SHF01
* SCAN FORK STRUCTURE AND OPERATE
*   A= OPERATION
*   X= PACPTR
*   THE OPERATION SPEC. BY A IS CALLED WITH CORRECT PACPTR IN X
*   PERFORMS OP FOR ALL LOWER AND PARALLEL FORKS.
SCFK   ZR0; STA SCF01; STB SCF02; STX SCF03
* SCAN TO LOCAL 'BOTTOM'
SCF06  SKN PPTR,2; BRU SCF07
      LDA PPTR,2; RSH 12; CAX; BRU SCF06
SCF07  CXA; SKE SCF03; BRU SCF05
* PERFORM OP AND EXIT
      BRM* SCF01; LDB SCF02; LDX SCF03; BRR SCFK
SCF05  LDA PQU,2; SKA PRMSK; BRU SCF08
* PERFORM OP AND GO 'UP'
      LDA PPTR,2; MRG PLMSK; STA SCF04; BRM* SCF01; LDX SCF04; BRU SCF07
* PERFORM OP AND GO 'PARALLEL'
SCF08  MRG PLMSK; STA SCF04; BRM* SCF01; LDX SCF04; BRU SCF06
SCF01  ZR0 0  OPERATION
SCF02  ZR0 0  STATUS WORD

```

```

SCF03  ZR0 0  PACPTR
SCF04  ZR0
*
* TERMINATE FORK STRUCTURE
* INPUT: X=PACPTR OF LOWER FORK
TFK  ZR0; LDA PPTR,2; STA TF01; LDA PTAB,2; STA TF06
      LDA #DFK; BRM SCFK; LDA TF01; STX TF01; SKA PRMSK; BRU TF02
* EXEC TOP-LEVEL PANIC
      LDA PACDMB; STA PTEST,2; SKN XPB; BRM M0NCR
      LDX XPB; LDA EXECL6,4
      STA PX,2 (SAVE STATUS); LDX TF01
      LDA TFC01; XMA PL,2; STA TFC02
      LDA PQU,2; ETR #47777777B; STA PQU,2; BRU TF09
* ORDINARY PANIC
TF02  MRG PLMSK; COPY AX,AB; LDA PIM,2; RSH 3; ETR #7; STA TF07
      CXB; LDA PTEST,2; E0R #700000B
      SKA #7700000B; BRU TF03
* DISMISSED 0N ACT. COND. 7
      CAX; BRU *,2
      BRU TF03 (ACTIVE); BRU TF04 (BRS31); BRU TF05 (BRS 106)
      BRU TF08 (PERFORMING EXEC BRS); BRU TF04 (BRS 109)
      BRU TF04 (BRS 9)
* CAUSE INTERRUPT IF ARMED
TF03  LDA #400000B; CBX; BRM IIR; NOP; BRR TFK
* DISMISSED FOR BRS 106
TF05  CBX; MIN PL,2
* PERFORMING EXEC BRS
TF08  LDA PACDMB; CBX; STA PTEST,2; MIN PL,2
      LDA TF06; ETR ADMSK; STA PA,2; LDA SCF02
      SKN XPB; BRM M0NCR
      LDX TF07; STA PX,2; CBX
TF09  CXA; STX TF06; LDX #QI0; BRM QPUT; LDX TF06; BRR TFK
* DISMISSED 0N BRS 31 OR BRS 109
TF04  CBX; LDA TF06; E0R PA,2; SKA ADMSK; BRU TF03; BRU TF08
TF01  ZR0
TF06  ZR0 0  PANIC TABLE ADDRESS
TF07  ZR0 0  FORK NUMBER FOR HIGHER FORK
TFC01  ZR0 EXECP,4
TFC02  ZR0 0  PL BEFORE IT BECAME EXECP,4

```

```

*
* READ FORK STATUS
* INPUT: B=STATUS, X=PACPTR OF FORK TO BE READ
RFK   ZR0; LDA PQU,2; EOR X6; SKA X6; BRU RF12
      LDA PPTR,2; MRG PLMSK; CAX EXEC BRS
RF12  LDA =RF06; BRM SCFK; LDX SCF03; BRM CHRL; BRU *-2
      LDB SCF02; LDX SCF03; BRR RFK
*
* INPUT: X=PACPTR OF LOWER OR PARALLEL FORK.
RF06  ZR0
      STX RF08; LDX J0B; LDX RL3,2; CLAB
      BRM SWAP; BRU *-1 (HANG); BRM LABEL; LDX RF08
      LDA PL,2; SKA X4; BRU **2; LDA SBRST; STA RF14
      CXA; EOR SCF03; SKA ADMSK; BRU RF10
      LDB SCF02; SKB X4; BRU **2; BRU RF07
RF10  LDA PTEST,2; LDB =-1; SKE PACDMB; LDB =-2 (fork was running)
RF07  LDA PQU,2; EOR X6; SKA X6; BRU **3; CXA; BRU RF11 (EXEC BRS)
      LDA PPTR,2; SKA PRMSK; BRU **2; CXA (TOP-LEVEL EXEC)
RF11  MRG PLMSK; STA RF13; STB FK04
      LDA PIM,2; RSH 3; ETR =7; STA RF15
      LDA PTAB,2; ETR ADMSK; MRG X4; STA RF08 (part of table address 4)
      CXA; ADD RFC01; STA RF09; STA RF16
      LDA RF08; ETR =34000B; LRSH 11; MUL =3 (not sub table nor for use)
      LDX RF13; LDA RL1,2; LDX RL2,2; XXB; RCY 18; LCY 0,2
      ETR =77B; CLB; RCY 0,2; LCY 18
      LDX J0B; LDX RL3,2; BRM SWAP; BRU *-1 (PANIC TABLE PAGE AND TS)
      BRM LABEL; LDA RF14; STA* RF08; MIN RF08; LDX =-3
RF09  LDA PL4,2; STA* RF08; MIN RF08; EAX 1,2; CXB
      LDX RF15; LDA PB,2; STA* RF08; MIN RF08
      LDA PX,2; STA* RF08; MIN RF08; CBX
RF16  LDA PL4,2; STA* RF08; MIN RF08; BRX **3
      LDA FK04; STA* RF08; BRR RF06
RF08  ZR0
RF13  ZR0
RF14  ZR0
RF15  ZR0 0 PB POINTER OF FORK TO READ
RFC01 LDA PL4,2
*
* BRS 9 START SUBSIDIARY FORK

```

```

* INPUT; SS01=A=BITS + PANIC TABLE ADDRESS
*     BITS; 0=SYSTEM, 1=PANIC TABLE REL., 2=PR0P. RUB0UT
*     3=FIXED MEMORY, 4=LOCAL MEMORY, 5=SUBSYSTEM STATUS
FKST  ETR =3777B; ADD =6; SKA =4000B; BRM TRAPB (PANIC TABLE 0VERLAP)
      LDA SS01; BRM SHFK; BRU **2; BRM TRAPB (TRIED TO RESTART SAME FORK)
      LDA NF0RK; SKG =0; BRM TRAPB
      BRM GFK; BRU FKSTW (DISMISS UNTIL PACT SLOT FREE); BRM STFK
      LDB SS01; SKB X4; SKN PQU,2; BRU **2; MRG X4; LDX FK04; STA PQU,2
      ETR X4; STA FK09A
      LDA PIM,2; SKB =4000000B; MRG X4; STA PIM,2
      CBA; ETR ADMSK; LRSH 15; LDA J0B; LSH 15
      SKB =2000000B; MRG X4; SKB =1000000B; SKN EXEC1
      BRU **2; MRG X2; MRG X1; STA PTAB,2
      SKN FK09A; BRU **2; BRU FK07B; SKA X2; BRU **3
      LDA X2; BRU **2; LDA X6; STA FK09A
* PR0PAGATE TTYASG D0WN IF CALLED F0R
FK07B LDA UTTY; SKG =-1; BRU FK07A
      CAX; LDA FK04; SKB X1; STA TTYASG,2
FK07A LDX PACPTR; LDA SS01; SKA X2; BRU FK08
      LDA RL1,2; LDB RL2,2
      LDX FK09A; BRM SCRL; BRU FK08B; LDX PACPTR
      LDA RL1,2; LDB RL2,2
* ST0RE RELABELLING, INITIAL A,B,X,L
FK09  LDX FK04; STA RL1,2; STB RL2,2; LDX SS01
      LDA 2,6; LDB 3,6; LDX STFK2; STA PB,2; STB PX,2
      LDX SS01; LDA =-1; STA 6,6 (SET STATUS WD TO RUNNING)
      LDB 1,6; LDA 0,6; ETR =50037777B; SKN 0; BRU **2; MRG X4
      LDX FK04; STA PL,2; STB PA,2; LDX PACPTR
      SKN EXEC1; BRU **2; BRU P0PX; LDB =700006B; BRU P0PST
* PICK UP RELABELLING FROM PANIC TABLE
FK08  CAX; LDA 4,6; LDB 5,6; LDX FK09A
      BRM SCRL; BRU FK08B; LDX SS01; LDA 4,6; LDB 5,6; BRU FK09
FK08B LDX FK04; BRM DFK; BRM TRAPB
* DISMISS UNTIL PACT SLOT IS RELEASED
FKSTW LDB =FPLST; BRU I0QDMS
FK09A ZR0 0  ST0RAGE F0R RELABELLING STATUS
*
* BRS 31
* WAIT F0R SPECIFIED F0RK TO TERMINATE

```



```

* INPUT: A=PANIC TABLE ADDR.
* OUTPUT: X=STATUS
FKWT  BRM SHFK; BRU FK01; LDB =700002B; BRU P0PST
* READ STATUS OF SPECIFIED FORK
FKRD  BRM SHFK; BRU FK01; LDB =-1; BRM RFK
FK01  LDX SS01; LDA 6,6; STA SS03; BRU P0PX
* BRS 32  TERMINATE SPECIFIED FORK
FKTM  BRM SHFK; BRU FK01; LDB =-1; BRM RFK; BRM TFK; BRU FK01
*
* BRS 106
* WAIT FOR ANY FORK TO TERMINATE
FKWA  SKN PPTR,2; BRU P0PX; LDB =700003B; BRU P0PST
* BRS 108  TERMINATE ALL SUBSIDIARY FORKS
FKTA  SKN PPTR,2; BRU P0PX; LDB =FK02
FK03  STB FK04; LDA PPTR,2; RSH 12
FK05  MRG PLMSK; CAX; LDB =-1; BRM* FK04
      LDA PQU,2; SKA PRMSK; BRU FK05; BRU P0PX
FK02  ZR0; BRM RFK; BRM TFK; BRR FK02
* BRS 107  READ STATUS OF ALL SUBSIDIARY FORKS
FKRA  SKN PPTR,2; BRU P0PX; LDB =RFK; BRU FK03
FK04  ZR0

* PROGRAMMED INTERRUPT LOGIC
*
* BRS 49  READ INTERRUPTS
SRIR  LDA PIM,2; ETR =37777000B; LDX SS03; BRU XP0P
*
* BRS 79  CAUSE INTERRUPT
* INPUT: A=INT. NO.
      IF      V5
SIIR  SKG =10; SKG =4; BRM TRAPB; CAX
      ELSF   1
SIIR  SKG =20; SKG =4; BRM TRAPB; CAX
      ENDF
      LDA =4000000B; RSH 0,2; LDX PACPTR; BRM SIR; BRU P0PX; BRU P0PX
*
* BRS 78  ARM INTERRUPTS
      IF      V5
SAIR  ETR =3777000B; SKN PQU,2; ETR =3776000B

```

```
      XMA PIM,2; ETR =74000777B; ADM PIM,2; BRU P8PX
      ELSF 1
SAIR  ETR =3777777B; STA PIM,2; BRU P8PX
      ENDF
```

```
*
      IF V5
* BRS BE+12 INTERRUPT AFTER A SPECIFIED TIME.
* INPUT; A=MASK, B=TIME IN MS, X=INTERRUPT NO.
* PU ENTRY; 2,2=REAL, 3,2; 0-11=PACPTR, 12-17=INT. NO.
TIMINT ETR =3777000B; SKN PGU,2; ETR =3776000B; MRG =1
      XMA PIM,2; STA TIMIN4; ETR =74000777B; ADM PIM,2
      LDA =4; SKA PIM,2; BRU TIMIN3
      CBA; MUL =1727024B (CONVERT MSEC. TO CLOCK TICKS)
      SKB X4; ADD =1; ADD REAL; COPY AX,B; LDA SS03
      ETR =37B; SKG =10; SKG =4; BRM TRAPB; LRSH 6
      LDA PACPTR; LSH 12; MRG UTY; COPY AX,XB; LDA TIMIN2
      DIR; BRM EPU; BRU P8PX
TIMIN3 LDA TIMIN4; STA PIM,2; BRM TRAPB
TIMIN4 ZR0
TIMIN2 7 PUTIM
ENDBRS BSS 0
      ENDF
```

END

3DSC IDENT 7/02/67

\* ENTRY POINTS

ENTRY SDBMA  
ENTRY DTS, DTC, DTW, DST  
ENTRY DIW, DOW, LSM, SSM  
ENTRY SSMFA, CBRFA, SWSFA, DRMSI, DRMSO  
ENTRY DFDLA, DFERA, DFRXA, DFCDA  
ENTRY DRMOPN, DCLS4, DTD, DTX

\* ENTRIES (FROM MDBG)

ENTRY SMIFIL, SMBA, SMDRN, FBWRD, SMOFIL  
ENTRY BX0, BBP, BFP, BIN, BIC, BDN, BDC, BIP, BIA

\*  
IDENT1 BRM IDM

IDENT2 BRM IDM2

IF C181

DMSK DATA 37777777B

ENDF

DADMSK DATA NDISCS\*32\*64=1; \*DISC ADDRESS MASK, RIGHT SHIFTED 2.

\*  
\*  
\*  
\* 'DRMOPN', 'DRMCLS' 11/13/65  
\*  
\*

\* THESE BELONG TO 'MOPN' AND 'MNCLS', AND OPEN AND  
\* CLOSE DRUM FILES  
\*

DRMOPN ZR0; LDB X1; LDA SS01; ETR DADMSK; SKE =0; BRU \*\*2  
BRU DOPN7+1; STA DOPN5; LDX =NFILE+3  
DOPN6 SKB EFA,2; BRU DOPN7; LDA EFA,2; ETR DADMSK; SKE DOPN5  
BRU DOPN7; LDA OUTBIT; SKA DEV; BRU DOPN3C  
SKA FD,2; BRU DOPN3C  
DOPN7 BRX DOPN6; LDX BUFF  
LDA DEV; SKN SS01; BRU DOPN2; MRG DR0BIT; STA DEV  
DOPN2 EQU \*  
IF V2; SKA OUTBIT; SKN SDBM8; BRU \*\*2; BRU DOPN3D; ENDF  
LDA =-1; STA BIC,2; STA BDC,2; STA BDN,2  
COPY XA,B; STB BIN,2; ADD =BX0; STA BIP,2  
LDA SS01;

```

IF V2; ETR DADMSK; ELSF 1; ETR PRMSK; ENDF
SKG =0; BRU D0PN4; BRM BSET
LDA SS01; IF V2; LSH 2; ELSF 1; LSH 8; ENDF; STA BIA,2; CAB
LDA T; ADD =BX0; LDX =NDXWR; BRM DTC
LDA FFLST; STA FILE; BRM DTF; LDA X2; ADM DEV; LDX BUFF
D0PN1 LDA SS01; STA SS03
IF V2; ETR DADMSK; ELSF 1; ETR PRMSK; ENDF
STA T
LDA DEV; SKA =2100000B; BRR DRM0PN; SKR BIP,2; BRR DRM0PN
IF V2
D0PN4 BRM DTA; STA BIA,2; RSH 2; STA SS01
ELSF 1
D0PN4 BRM DTR; BRM DTA; STA BIA,2; RSH 8; STA SS01
ENDF
BRM DTZ; STB BFP,2; STB BBP,2; BRU D0PN1
D0PN3B MIN DRM0PN; BRR DRM0PN
D0PN3C LDA =-1; BRU D0PN3B
D0PN3D LDA =-3; BRU D0PN3B
D0PN5 ZR0 0
*
DCLS4 LDA FILE; SKE SMIFIL; BRU DCLS2
LDA =-1; STA SMIFIL; STA SM0FIL
DCLS2 LDX BUFF; BRM BSET; BRM DTU; BRU DCLS3
*
* 'LAS', 'SAS' 10/18/65
*
* THIS ROUTINE IMPLEMENTS THE LOAD AND STORE TO SECONDARY MEMORY.
*
* P0P 146
LSM LDA* 0; SKN SMIFIL; BRU **2; BRM NTRPB
STB SS02; STX SS03; SKA X2; ADD SS03
ETR =1777777B; RSH 23; DIV =NDDW; SKE* SMDRN; BRU DTGS
CBX; LDA* SMBA; LDB SS02; LDX SS03
SKN TIME; BRR 0; BRU NXP0P
*
* P0P 147
SSM STA SS01; LDA* 0; SKN SM0FIL; BRU **2; BRM NTRPB
STB SS02; STX SS03; SKA X2; ADD SS03
ETR =1777777B; RSH 23; DIV =NDDW; SKE* SMDRN; BRU DTGS

```

CBX; LDA SS01; STA\* SMBA; LDX SMBA; MIN BDC=2,2  
LDB SS02; LDX SS03; SKN TIME; BRR 0; BRU NXP0P

\*  
\*  
\*  
\*  
\*  
\*

\* 'DPU' 3/20/66

\* DRUM I/O EXIT TO PHANTOM USER

DPU ZR0; LDA FILE; RSH 12; LDA PACPTR; RSH 12  
LDA DPU; ETR ADMSK; LDX UTTY; DIR; BRM EPU  
LDX FILE; LDA =DBB; ADM FD,2  
LDX DTXS2; LDA =100000B; ADM 2,2  
BRM DTF; BRU NI0DMS

\*  
\*  
\*  
\*  
\*

\* 'DRMSI', 'DRMS0' 12/7/65

\* SEQUENTIAL I/O DRIVER, CALLED FROM 'GPW'

DRMSI ZR0; LDX BIP,2; LDA 1,2; SKG =0; BRU DSI1  
LDX BUFF; MIN BIP,2; LDA\* BIP,2  
IF V2; ETR =7777777B; CAB  
IF C181  
LSH 2; LDA T; ADD =2; LDX =NDDW+1  
ELSF 1  
LDA T; ADD =1; LDX =NDDW+1  
ENDF  
BRM DTC; LDX BUFF; CXA; ADD =2; STA 0,2  
ELSF 1; ETR =7777400B; CAB  
LDA T; ADD =2; LDX =NDDW; BRM DTC  
LDX BUFF; CXA; ADD =2; STA 0,2; STA 1,2  
LDA\* BIP,2; CAB; ETR =377B  
SKB X1; MRG E0RBIT; ADM 1,2; ENDF; BRM DTF; BRR DRMSI  
DSI1 LDX BUFF; EAX 2,2; STX =2,2; STX =1,2  
LDA E0RBIT; ADM =1,2; LDX FILE; LDA XN2; ADM FD,2; BRR DRMSI

\*

DRMS0 ZR0; BRM DTD  
LDA BIP,2; BRM DTT; MIN BIP,2  
CBA; BRM DTA  
IF C181

LRSH 2  
ENDF

MRG X1; STA\* BIP,2; BRM DTF; BRR DRMS0

\*  
\* 'DTP', 'DTH', 'DTF', 'DTC', 'DTW', 'DTX', 'DTS', 'DTM', 'DTD', 'DTT'  
\* 'DTA', 'DT0', 'DTE', 'DTN', 'DTU', 'DTZ', 'DTR', 'DTL'  
\* 1/28/66

\*  
\* GENERALLY USEFUL DRUM ROUTINES

\*  
DTLS1 ZR0; \* TEMPORARY BIP

\*  
\* ENTER COMMAND IN LIST AND LOCK MEMORY BLOCK  
\* A= ABS CORE ADDR  
\* B= DRUM ADDR  
\* X= WORD COUNT

IF V2  
DTC ZR0; MIN DTXS1; STB\* EDCL (STORE DISC ADDRESS)  
RCH 204B (CAB,CXA); LDX EDCL  
SKB X4; MRG X4; STA 2,2 (STORE WORD COUNT)  
CBA; ETR =177777B; STA 1,2 (STORE REAL CORE ADDRESS)  
STX DTXS2; LRSH 11; ETR =37B  
XXA; MIN RMC,2; CAX; LDA 0,2 (INCREMENT PAGE LOCK)  
\* (COMPUTE ARM MOTION AND SET NEW ARM POSITION)  
LRSH 13; RCH 401B (CAX,CLA); LSH 6; XMA DPT,2  
SUB DPT,2; SKG =+1; CNA; ADM ARM0T  
\* (GUESS AT TIME TO FINISH I/O)  
CAB; LDA =2; SKB =+1; ADD =10  
LDX DTXS2; CLB; LSH 18; ADM 2,2  
CXA; ADD =3; SKE =DRQU; BRU +=2  
LDA =DRQ; STA EDCL; LDA =NDRQ=3; SUB DTXS1  
SKG NDCL; BRU \*-1; BRR DTC

DPT BSS NDISCS

ELSF 1  
DTC ZR0; MIN DTXS1; STX\* EDCL; LDX EDCL; STA 1,2; STB 2,2  
LDA =DCDI; STA 3,2; STX DTXS2; LDA =DCTX; ADM 0,2  
LDA 1,2; RSH 11; ETR =37B; COPY XB,AX; MIN RMC,2  
CBA; ADD =4; SKE =DRQU; BRU +=2; LDA =DRQ (WRAP AROUND DRUM QUEUE)  
STA EDCL; LDA =NDRQ=2; SUB DTXS1; SKG NDCL; BRU \*-1; BRR DTC

```

        ENDF
DTW     ZR0; MRG DCWBIT; BRM DTC; BRR DTW
* DRUM START FOR FILE OPERATIONS
        IF V2
DTF     ZR0; LDB FILE; LSH 40; LDX DTXS2
        ADM 1,2; LDA =100000B; MRG 2,2; STA 2,2; BRM DTS; BRR DTF
        ELSF 1
DTF     ZR0; LDB FILE; LSH 30; MRG =1; LDX DTXS2; ADM 3,2
        LDA DTXS1; ADD =1; STA FTIME; BRM DTS; BRR DTF
        ENDF
DTFF    ZR0; BRM DTF; LDX FILE; LDA X2; ADM FD,2; BRR DTFF
* DRUM START AND EXIT
DTX     SKN DTXS1; BRU DTX1; SKR 0; NOP; BRU NP0PX
DTX1    BRM DTFF; BRU NFI0DS
*
* START DISC AND WAIT
DST     ZR0; LDA DRUERR; STA DSTE; BRM DTS; SKN IDCL1; BRU *-1
        LDA DRUERR; SKE DSTE; MIN DST; BRR DST
DSTE    ZR0 0
*
* DRUM START
DTS     ZR0; LDA DTXS1; SKA X4; BRU DTS1; ADD =1; DIR; XMA NDCL; ADM NDCL
        EIR; SKE =-1; BRU DTS1
        SKN IDCL1; BRU DTS1; LDB =1; SKA BLK31
        BRU *-1; STB BLK31; LDA IDENT1; STA 31B
        LDA IDENT2; STA 33B; MIN IDMRET; BRM IDM
DTS1    BRM DTP; BRR DTS
* ELIMINATE BLOCK IN SEQ. OUTPUT FILE
DTM     ZR0; STX DTC; LDA BUFF; ADD =23500000B+BX0+NDXW-2
        STA DTM2; ADD =6200000B-3500000B; STA DTM1
        COPY BA,B; SUB DTC; ADM DTM2; CAX
DTM3    STX DTW; CXA; ADD DTC; AXC; XMA 0,2
        IF      C181
        LSH 2
        ENDF
        BRM DTE; LDX DTW; BRX DTM3
        LDA DTC; SUB BUFF; SUB =BX0+NDXW-2; AXC
DTM1    XMA BX0+NDXW-2,2
DTM2    STA BX0+NDXW-2,2; CLA; BRX DTM1

```

```

LDX BUFF; MIN BIC,2; BRR DTM
* DUMP BUFFER FOR SEQ. FILE (LIKE DT0)
DTD ZR0; SKN BDC,2; BRU DTD3
DTD8 CXA; ADD =2; STA 0,2; BRR DTD
DTD3 LDA* BIP,2; SKG X1; BRU DTD1
LDA BX0+NDXWC=3,2; SKE =0; BRU DTD10 (INDEX BLOCK FULL)
DTD2 EQU *
IF V2; LDA* BIP,2; ETR =7777777B; STA* BIP,2
LDA 0,2; SUB BUFF; SUB =2
IF C181
STA NDDW+2,2
ELSF 1
STA 1,2
ENDF
ELSF 1
LDA* BIP,2; ETR =7777400B; ADD 0,2
SUB BUFF; SUB =2; STA* BIP,2; ENDF
BRM DT0; MIN BIC,2; BRU DTD8
DTD1 SKE =0; BRU DTD4; IF =V2; LDA BIP,2; SUB =BX0-1; SKG BUFF; BRU DTD9
LDX BIP,2; LDA =1,2; LDX BUFF; ENDF
BRM DTA
IF C181
LRSH 2
ENDF
STA* BIP,2; BRU DTD2 (WRITING ON END OF FILE)
DTD4 LDA X1; LDX BIP,2; CXB
DTD7 SKA 0,2; BRU DTD6; EAX 1,2; BRU DTD7
DTD6 BRM DTM; BRU DTD2 (DELETE OLD RECORD)
IF C181
DTD10 CLB; LSH 2; ETR DMSK; BRM DTE
ELSF 1
DTD10 BRM DTE
ENDF
MIN BIC,2; CLA; STA BX0+NDXWC=3,2
LDA BX0+NDXWC=4,2; MRG X1; STA BX0+NDXWC=4,2; BRM NTRPB
* INSERT POINTER IN INDEX BLOCK
DTT ZR0; XXA; ADD =26200000B+BX0+NDXW=2; STA DTT1
CX A; SUB BUFF; SUB =BX0+NDXW=3; LDB 0,2; COPY AX,BA
DTT1 XMA BX0+NDXW=2,2; BRX DTT1; LDX BUFF; MIN BIC,2; BRR DTT

```



\* ACQUIRE (OPTIMAL) NEW DRUM BLOCK  
IF V2

GBX2 EQU 20000000B

GETBLK ZR0

LDA JOB

SUB #1

ETR #NDISCS#1

MUL #NPOS\*32

DIV #24

SUB #TABLEN

STA GBS1

LDA #TABLE+GBX2

STA GBASE

LDA #1

RCH 24B

LRSR 0,2

LDX GBS1

SKA\* GBASE

BRU GB2

GB1 CBA

BRX #3

LDA #TABLEN#2

SUB GBS1

CAX

LDA GBASE

SKE #TABLE+GBX2

BRR GETBLK

ADD GBS1

ADD #1

STA GBASE

BRU GB1

GB2 XMA\* GBASE

STX GBS1

ETR\* GBASE

RCH 5

LDX #48

N0D 48

E0R #GBX2

LCY 0,2

```

STX GBS2
LDX GBS1
STB* GBASE
LDA GBASE
ADD GBS1
SUB #TABLE*TABLEN+GBX2
MUL #12
CBA
ADD #25
SUB GBS2
LRSH 23
DIV #40B
STB GBS1
LRSH 23
DIV #NP0S
RCH 412B          CBA,CAX,CLB
LSH 5
ADD #MINP/4
ADM GBS1
CXA
LSH 11
ADD GBS1
LSH 2
MIN GETBLK
BRR GETBLK
GBS1 ZR0
GBS2 ZR0
GBASE ZR0
DTA   ZR0; BRM GETBLK; BRM NTRPB; LDX BUFF; BRR DTA
*           INSERT BIT INTO BIT MAP TO FREE 1 DISC BLOCK
DTE   ZR0; STX DTE3; BRM L0CBIT; BRM M0NCR; BRU DTE1
      SKA TABLE,2; BRM M0NCR; ADM TABLE,2; BRU DTE2
DTE1  MIN 0LCNT
DTE2  LDX DTE3; BRR DTE
DTE3  ZR0

```

```

*
*LOCATE BIT CORRESPONDING TO DISC ADDRESS IN A.
*NO SKIP IF BAD ADDRESS (NO REGS CHANGED). ONE
*SKIP IF OUT OF CYLINDER (A=GARBAGE). TWO SKIPS

```

```

*IF ADDRESS 0K (A=BIT POSITIONED, X=NEG INDEX
*ON TABLE, B=CLEARED).
*(ABOUT 90 CYCLES)
* INPUT: A=DISC ADDRESS
*
L0CBIT ZR0 0
      STA LCBIT
      IF      C181
      ETR DMSK
      ELSF    1
      ETR N0TE0R
      ENDF
      SKG =NDISCS*128*64=1; SKG =-1; BRR L0CBIT
      STA LBS1; ETR =7760000B; MUL =NP0S*400000B; XMA LBS1
      ETR =17777B; SKG =MAXP+177B; SKG =MINP=1; BRU LB1; SUB =MINP
      ADD LBS1; LRSH 25; DIV =24; SUB =TABLEN; C0PY BX,B; STA LBS1
      LDA =40000000B; LRSH 0,2; LDX LBS1; MIN L0CBIT
LB1   MIN L0CBIT; BRR L0CBIT;LBS1 ZR0 0
*
* BIT MAP
LBT   EQU *
      RPT TABLEN=1; DATA =1; ENDR
LWT   EQU      NP0S/2*2*NDISCS*32
LWT   EQU      LWT=LWT/24*24
      IF LWT
LWT1  EQU      1
      RPT 24=LWT;LWT1 EQU LWT1*2; ENDR
      DATA =LWT1
      ELSF 1; DATA =1; ENDF
TABLE EQU *
*
0LCNT ZR0;* C0UNT 0F DATA BLKS RETURNED FR0M 0UTSIDE CYLINDER.
LCBIT ZR0 0 INPUT T0 L0CBIT
*
      ELSF 1
DTA   ZR0; ADD =1000B; LDB =1; BRM DTAS; BRR DTA
DTAR  ZR0; SUB =1000B; LDB =-1; BRM DTAS; BRR DTAR
DTAS  ZR0; STB DTAS3; ETR =37400B; RSH 8
DTA3  C0PY AX,A,B; STX DTAS1; SKE FDBT0,2; BRU DTA0

```

```

SKE FDBT1,2; BRU DTA1; SKE FDBT2,2; BRU DTA2
CXA; ADD DTAS3; ETR =77B; BRU DTA3
DTA0  LDB FDBT0,2; LDX =46; N0D 48; E0R X2; LCY 2,2
      STX DTAS2; LDX DTAS1; STB FDBT0,2
DTA4  CXB; LSH 18; ADD DTAS2; LSH 14; ADD NSBA
      SKR* DBAJ0B; N0P; LDX BUFF; BRR DTAS
DTA1  LDB FDBT1,2; LDX =46; N0D 48; E0R X2; LCY 2,2
      STX DTAS2; LDX DTAS1; STB FDBT1,2; EAX 24*100B,2; BRU DTA4
DTA2  LDB FDBT2,2; LDX =46; N0D 48; E0R X2; LCY 2,2
      STX DTAS2; LDX DTAS1; STB FDBT2,2; EAX 48*100B,2; BRU DTA4
* ERASE (RELEASE) BLOCK
DTE   ZR0; RSH 8; ETR =17777B; SUB NSBB; CAX; ETR =77B; XXA
      RSH 6; SKG =23; BRU DTE1; SUB =24; EAX 100B,2
      SKG =23; BRU DTE1; SUB =24; EAX 100B,2
DTE1  LDB FDBT0,2; XXA; RCY 1,2; SKA X4; BRU DTE2; MIN* DBAJ0B
      MRG X4; LCY 1,2; CAX; STB FDBT0,2
DTE2  LDX BUFF; BRR DTE
      ENDF
* WRITE OLD DATA BLOCK IF CHANGED
DT0   ZR0; SKN BDC,2; BRU *+2; BRR DT0
      LDA =-1; STA BDC,2; LDA* BIP,2
      IF V2
      IF          C181
      ETR =7777777B; CAB; LSH 2; LDA T; ADD =2; LDX =NDDW+1
      ELSF          1
      ETR =7777777B; CAB; LDA T; ADD =1; LDX =NDDW+1
      ENDF
      ELSF 1
      ETR =7777400B; CAB; LDA T; ADD =2; LDX =NDDW
      ENDF
      BRM DTW; LDX BUFF; BRR DT0
* WRITE OLD INDEX BLOCK IF CHANGED, GET NEW
DTN   ZR0; XMA BIA,2; STA DTN4; SKN BIC,2; BRU DTN1
DTN2  LDA BIA,2; SKE DTN4; SKG =0; BRU DTN3; CAB
      LDA T; ADD =BX0; LDX =NDXWR; BRM DTC
DTN3  LDA BUFF; ADD =BX0; LDX BUFF; STA BIP,2; BRR DTN
DTN1  CAB; LDA T; ADD =BX0; LDX =NDXWR
      BRM DTW; LDX BUFF; LDA =-1; STA BIC,2; BRU DTN2
DTN4  ZR0 0

```

```

* WRITE, RELEASE BUFFER AND EXIT
DTU   ZR0; SKN BDC,2; BRU DTU1; SKN BIC,2; BRU DTU2; BRR DTU
DTU1  BRM BSET; BRM DT0; BRU DTU4
DTU2  BRM BSET
DTU4  LDA BIA,2; BRM DTN; BRU DTX
* ZERO INDEX BLOCK
DTZ   ZR0; LDA BUFF; ADD =23600000B+BX0+NDXW-2; STA DTZ1
      CLB; LDX =-NDXW+2
DTZ1  STB BX0+NDXW-2,2; BRX DTZ1; LDX BUFF; MIN BIC,2; BRR DTZ
      IF -V2
* GET PSUEDO-RANDOM BLOCK NUMBER
DTR   ZR0; LDA DTRS1; ADD =7; ADD REAL
      ETR =77B; STA DTRS1; LCY 8; BRR DTR
      ENDF
* ASSIGN NEW DATA BLOCK
DTL   ZR0; STX DTLS1; IF V2; BRM DTA
      ELSF 1; CXA; SUB BUFF; SKE =BX0; BRU DTL1
      LDA 1,2; SKG =0; BRM DTR; BRM DTAR; BRU DTL2
DTL1  LDA -1,2; SKG =0; BRM DTR; BRM DTA
DTL2  MRG =377B; ENDF; STA* DTLS1; LDX BUFF; MIN BIC,2; BRR DTL
*
* 'DRXI0I','DWI','DW0' 10/30/65
*
* THESE IMPLEMENT SINGLE-WORD TRANSFERS TO RANDOM FILES
*
DRXI0I ZR0; STA SS01; STB SS02; STX SS03
      LDA* 0; ETR ADMSK; SKG =NFILE-1; BRU **2; BRM NTRPB
      IF FCB
      STA FILE; AXC; LDB FC,2; LCY 9; ETR =77B; SKE J0B; BRM NTRPB
      ELSF 1
      STA FILE; AXC; LDB FA,2; LCY 9; SKE J0B; BRM NTRPB
      ENDF
      LDA FD,2; SKA X6; BRU I0IE (FILE BUSY)
      ETR =06000000B; SKE =06000000B; BRM NTRPB
      LDA SS02; ETR =17777777B; RSH 23; DIV =NDDW
      LDX FC,2; SKE BDN,2; BRU DTG; STB T; CXA; ADD T; ADD =2
      BRR DRXI0I
*
* POP 144 DISC WORD INPUT FROM RANDOM FILE

```

```

DIW   BRM DRXI0I; CAX; LDA 0,2; LDB SS02; LDX SS03; BRU NXP0P
*
* P0P 145   DISC WORD OUTPUT TO RANDOM FILE
D0W   BRM DRXI0I; LDX FILE; LDB FD,2; SKB DR0BIT; BRM NTRPB
      LDX FC,2; MIN BDC,2; CAX; LDA SS01; STA 0,2
      LDB SS02; LDX SS03; BRU NXP0P
*
      IF =V2
* 'DBI0','DBI','DB0' 5/20/66
*
* THIS ROUTINE IMPLEMENTS THE FAST DBI/DB0 SYSP0PS
*
DBTP3 ZR0;* COMMAND BUILDER (DTC OR DTW)
DBXS1 ZR0;* RECORD NUMBER
DBXS2 ZR0;* WORD NUMBER
DBXS3 ZR0;* TEMPORARY BIP
*
DBI0  ZR0; STA DBTP1; STB DBTP2; STX DBTP3
      LDX SS03; LDA* 0; BRM I0I; COPY AX,BA
      LDB =06000000B; SKM =06000000B; BRU TRAP
      SKN DBTP1; BRU *+3; SKA DR0BIT; BRU TRAP
DBX1  LDA SS01; SKG =0; BRU DBX2
      LDA SS02; LRSB 23; DIV =NDDW; STA DBXS1; STB DBXS2
      SKB =-1; BRU DBX3A; LDA SS01; SKG =NDDW-1; BRU DBX3
      LDA DBXS1; LRSB 23; DIV =NDXW-2; SKE BIN,2; BRU DBX5
      CBA; ADD BUFF; ADD =BX0; STA DBXS3
* SET UP NEXT COMMAND
DBI0T LDX SS03; LDA 0,6; LDX BUFF
      LDA DBXS1; SKE BDN,2; BRU *+2; BRU DBX4
      LDA* DBXS3; SKE =0; BRU DBT1; LDX DBXS3; BRM DTL
DBT1  LDA SS03; BRM DTH; ADD =NDDW-1; ETR =3777B
      SKG =NDDW-2; BRU DBT2; LDB T
DBT3  LDA* DBXS3; ETR =7777400B; XAB; LDX =NDDW; BRM* DBTP3
      SKN DBTP2; BRU DBX6; LDX DTXS2
      LDB FILE; LSH 30; MRG =4; ADM 3,2
* UPDATE CENTRAL REGISTERS, AND LOOP
DBX6  LDA =NDDW; ADM SS02; ADM SS03; CNA; ADM SS01; MIN DBXS1
      LDX BUFF; MIN DBXS3; LDA SS01; SKG =NDDW-1; BRU DBX1
      LDA DBXS3; SUB BUFF; SKE =BX0+NDXW-2; BRU DBI0T; BRU DBX1

```

```

DBX2 SKN DTXS1; BRU DTX; BRR DBI0
DBX5 STA DBXS3; BRM BSET; LDA DBXS3
      SKN BDC,2; BRU DTG4; BRU DTG5
* GO ACROSS PAGE BOUNDARY
DBT2 LDX SS03; LDA 4000B,6
      IF RELCHN
      CXA; RSH 11; ADD =1; ETR =7; MUL =3; CBX
      LDA RRL1; LDB RRL2; LCY 0,2; RCY 1
      ETR =17400000B; MRG DRLBIT; MRG T; CAB; BRU DBT3
      ELSF 1
      LDX BUFF
      ENDF
* GO THROUGH BUFFER
DBX3 LDA DBXS1
DBX3A SKE BDN,2; BRU DTG
DBX4 LDA =NDDW; SUB DBXS2; SKG SS01; BRU +=2; LDA SS01
      STA T; ADM DBTP1; ADM DBTP2; CNA; CAX
      LDA BUFF; ADD DBXS2; LDB SS03; SKN DBTP2; XAB
      ADM DBTP1; CBA; ADM DBTP2
DBTP1 LDA 0,2
DBTP2 STA 0,2; BRX DBTP1; LDX BUFF; SKN DBTP2; MIN BDC,2
      LDA T; ADM SS02; ADM SS03; CNA; ADM SS01; BRU DBX1
*
DBI P0PD 14200000B,1,1,0,1
DIM STA SS01; STB SS02; STX SS03
      LDA DIMC1; LDB DIMC2; LDX =DTC; BRM DBI0; BRU P0PX
DIMC1 LDA 2,2
DIMC2 STA 0,6
*
DB0 P0PD 14300000B,1,1,0,1
DBM STA SS01; STB SS02; STX SS03
      LDA DBMC1; LDB DBMC2; LDX =DTW; BRM DBI0; BRU P0PX
DBMC1 LDA 0,6
DBMC2 STA 2,2
*
      ENDF
*
*
*

```

```

*
*
* 'DTG' 10/30/65
*
* THIS RELEASES THE CURRENT CONTENTS OF A DRUM BUFFER AND LOADS
* IT WITH A SPECIFIED DATA BLOCK
*
DTGS LDX SMIFIL; LDX FC,2
DTG STA DTC; CXA; ETR ADMSK; STA BUFF; BRM BSET
SKN BDC,2; BRU DTG4
LDA DTC; RSH 23; DIV =NDXW*2; SKE BIN,2; BRU DTG5
* DON'T NEED NEW INDEX BLOCK
LDA DTC; STA BDN,2; CBA; ADD BUFF; ADD =BX0; ETR ADMSK; STA BIP,2
CAX; LDA 0,2; SKG =0; BRU DTG6
* DATA BLOCK ALREADY EXISTS
IF V2
ETR =7777777B; CAB; LDX =NDDW+1; LDA T; ADD =1
ELSF 1
ETR =7777400B; CAB; LDX =NDDW; LDA T; ADD =2
ENDIF
BRM DTC; BRU DTX
* NO SUCH DATA BLOCK, CREATE A GOOD ONE
DTG6 BRM DTL; MIN BDC,2
COPY XA,B; ADD =23600002B+NDDW; STA DTG6C; LDX =-NDDW
DTG6C STB NDDW+2,2; BRX DTG6C; BRU DTX
* WRITE OLD DATA BLOCK
DTG4 BRM DT0; BRU DTX
* FETCH INDEX BLOCK
DTG5 LDB =-1; STB BDN,2; SKG BIN,2; BRU DTG5A
MIN BIN,2; LDA BFP,2; SKG =0; BRU DTG7
DTG8 BRM DTN; BRU DTX
* GO BACKWARD, NOT FORWARD
DTG5A LDA BBP,2; SKR BIN,2; SKE =0; BRU DTG8
* CREATE NEW INDEX BLOCK
IF V2
BRM DTA
ELSF 1
LDA BIA,2; SUB =400B; BRM DTAR
ENDIF

```



```

DTG7   STA BBP,2; BRU DTG7A
      EQU *
      BRM DTA; STA BFP,2
DTG7A  STA DTA; MIN BIC,2; LDA BIA,2; BRM DTN
      LDA DTA; XMA BIA,2; STA BIC,2; BRM DPU
      CLAB; XMA BBP,2; SKE BIA,2; BRU DTG7B
      LDA BIC,2; STA BFP,2; BRU DTG3
DTG7B  STB BFP,2; LDA BIC,2; STA BBP,2
DTG3   BRM DTZ; BRU NPUG0
*
* 'SSMF' 10/31/65
*
* BRS 58
* THIS CAUSES A RANDOM FILE TO BE DECLARED AS SECONDARY MEMORY
*
SSMFA  BRM I0I; CBA; LDB =06000000B; SKM =06000000B; BRM NTRPB
      STX SMIFIL; SKB DR0BIT; BRU **2; STX SM0FIL
      LDA BUFF; ADD =20000002B; STA SMBA
      LDA BUFF; ADD =BDN; ETR ADMASK; STA SMDRN; BRU NP0PX
*
* 'SWSF' 10/31/65
* BRS 82
*
* THIS CHANGES THE MODE OF A SEQUENTIAL FILE TO INPUT
* OR OUTPUT
*
SWSFA  BRM I0I; SKB DRMBIT; SKB DRXBIT; BRM NTRPB
      LDA SSQ2; SKE =0; BRU SWSF1
      CBA; ETR =(N0T)100000B; STA FD,2; BRU NP0PX
SWSF1  SKB DR0BIT; BRM NTRPB
* MAKE SURE THE FILE IS OPENED ONLY ONCE
      STB D0PN5; LDA FA,2; STA SWSF3; LDX =-NFILE+3; STX T; LDB DADMSK
SWSF2  LDA EFA,2; SKA X1; BRU SWSF9; SKM SWSF3; BRU SWSF9; MIN T
SWSF9  BRX SWSF2; LDA T; SKE =-NFILE+3+1; BRM NTRPB
      LDX BUFF; LDA D0PN5; MRG 0UTBIT; STA FD,2; BRU NP0PX
SWSF3  ZR0 0 X=BLOCK ADDRESS.
*
* 'DFER','DFRX' 11/11/65
*

```

\* THESE DELETE AND READ INDIVIDUAL INDEX BLOCKS

\* BRS 67 DELETES A BLOCK ON THE DISC

\*  
DFERA SKN SDBM8; BRU \*\*2; BRM NTRPB; CLB; ETR DADMSK  
LCY 2  
BRM DTE; BRU NP0PX

\*  
\* BRS 87 READ AN INDEX BLOCK

DFRXA LDA SS03; ETR ADMSK; STA T; ADD =NDXWR; SUB =1  
ETR NADMSK; SKE =0; BRM NTRPB; LDA T  
BRM DTH; LDB SS01; ETR DADMSK  
LSH 2  
LDA T; LDX =NDXWR; BRM DTC; MIN 0; BRU DTX

\*  
\* 'DFDL' 12/7/65

\*  
\* BRS 66

\* THIS DELETES THE CONTENTS OF A DISC FILE

\*  
DFDLA SKN SDBM8; BRU \*\*2; BRM NTRPB; BRM I0I  
SKB DRMBIT; BRU \*\*2; BRM NTRPB; CAX; BRM BSET  
SKN BDC,2; BRM DTU (EXITS); SKN BIC,2; BRM DTU (EXITS)  
LDA =1; STA BDN,2; LDA BFP,2; SKE =0; BRU DFDL1  
CXA; ADD =BX0+NDXW-3; BRU DFDL3

DFDL2 CLAB; XMA\* BIP,2  
IF C181; LSH 2; ETR DMSK; ENDF  
SKE =0; BRM DTE; LDA BIP,2; SUB =1

DFDL3 STA BIP,2; SUB =BX0-1; SKE BUFF; BRU DFDL2  
LDA BBP,2; SKE =0; BRU DFDL4; BRU NP0PX

DFDL1 MIN BIN,2; BRM DTN; BRU DTX

DFDL4 SKR BIN,2; N0P; LDA BIA,2; BRM DTE  
LDA BBP,2; BRM DTN  
BRM DPU; CLA; STA BFP,2; BRU NPUG0

\*  
\* 'DFCD' 3/20/66

\*  
\* ADD THE NUMBER OF DATA WORDS IN FILE (A) TO X

\*  
\* BRS 113

DFCDA BRM I01; SKB DRMBIT; BRU \*\*2; BRM NTRPB; BRM DSS  
CXA; ADD =27600000B+BX0+NDXW\*2; STA DFCD2; LDX =-NDXW+2  
DFCD2 LDA BX0+NDXW\*2,2; IF V2; SKE =0; LDA =255  
ELSF 1; ETR =377B; ENDF; ADM SS03; BRX DFCD2  
LDX BUFF; BRM BSET; LDA BFP,2; SKG =0; BRU DFCD3  
BRM DTN; MIN BIN,2; BRU DTX  
DFCD3 CLA; STA BIN,2; LDA BBP,2; SKG =0; BRU NP0PX  
LDX FILE; LDB FA,2; LSH 24; ETR DADMSK; LSH 2  
BRM DTN; MIN 0; BRU DTX

\*

\* 'CBRF' 12/7/65

\*

\* THIS DELETES A BLOCK OF INFORMATION FROM A RANDOM FILE

\*

\* BRS 59

CBRFA LDA SS03; BRM I01; CBA; LDB =07000000B; SKM =06000000B; BRM NTRPB  
CBRF4 LDA SS01; SKG =0; BRU NP0PX; LDA SS02; RSH 23; DIV =NDDW  
LDX BUFF; XAB; SKG =0; BRU CBRF1  
CBRF2 XAB; SKE BDN,2; BRU DTG; MIN BDC,2  
CBA; ADD SS01; SKG =NDDW; BRU \*\*2; LDA =NDDW  
CAX; ADD BUFF; ADD =23600002B; STA CBRF3  
CBA; STX T; SUB T; COPY AX,B  
CBRF3 STB NDDW\*2,2; BRX CBRF3  
CBRF0 ADM SS01; CNA; ADM SS02; BRU CBRF4  
CBRF2A CLA; BRU CBRF2  
CBRF1 LDA SS01; SKG =NDDW-1; BRU CBRF2A; BAC  
SKE BDN,2; BRU CBRF5  
LDA\* BIP,2; STB\* BIP,2; BRM DTE  
LDA =-1; STA BDC,2; STA BDN,2  
CBRF6 MIN BIC,2; LDA =-NDDW; BRU CBRF0  
CBRF5 RSH 23; DIV =NDXW\*2; SKE BIN,2; BRU DTG5  
BAC; ADD BUFF; ADD =BX0; AXC  
XMA 0,2; BRM DTE; BRU CBRF6

\*

IF =V2

\* 'DISR', 'DDLRL' 12/7/65

\*

\* THESE INSERT AND DELETE LOGICAL RECORDS IN A SEQUENTIAL FILE

\*

```

DISR   BRM DSS; BRM DSZ; MIN 0; BRU DS01
*
DDLRL  BRM DSS; BRM DSZ; LDX BIP,2; CXB; LDA X1
DDLRL1 EAX 1,2; SKA 0,2; BRU DDLR2; BRU DDLR1
DDLRL2 BRM DTM; BRU DSX
*
* 'DSF','DSB','DSS','DSZ' 12/13/65
*
* SPACE FORWARD OR BACKWARD 1 RECORD
*
DSF     ZR0; CXA; SUB BIP,2; SKE =-BX0-NDXW+3; BRU DSF1
        BRM BSET; LDA BFP,2; SKG =0; BRU DSF2; BRM DTN; BRU DTX
DSF1    MIN BIP,2; LDA* BIP,2; SKA X1; BRR DSF
        SKE =0; BRU DSF+1; SKR BIP,2
DSF2    LDA E0FBIT; MRG X4; MRG FILE; STA SS01; BRU NP0PX
*
DSB     ZR0; CXA; SUB BIP,2; SKE =NDXW+1; BRU DSB1
        BRM BSET; LDA BBP,2; BRM DTN; BRU DTX
DSB1    SKR BIP,2; N0P; LDA* BIP,2; SKG X1; BRU DSB+1; BRR DSB
*
        ENDF
DSS     ZR0; LDX BUFF; LDB DEV; CXA; ADD =2; STA 0,2
        SKB 0UTBIT; ADD =NDDW; STA 1,2; BRR DSS
        IF =V2
*
DSZ     ZR0; LDB DEV; SKB 0UTBIT
DSZ1    SKR BIP,2; LDA BIP,2; SUB =BX0; SKG BUFF; BRU DSZ2
        LDA* BIP,2; SKG X1; BRU DSZ1; BRR DSZ
DSZ2    LDA BBP,2; SKG =0; BRR DSZ
        LDA =NDXW-2; ADM BIP,2; LDA BBP,2; BRM DTN; BRU DTX
*
* 'DIE0R','D0E0R' 12/7/65
*
* SPACE TO END OF RECORD (INPUT) OR WRITE END OF RECORD (OUTPUT)
*   ON SEQUENTIAL FILE
*
DIE0R   LDA =1; STA SS02; BRU DFSR
*
D0E0R   BRU TRAP (NOT IMPLEMENTED)

```

```

*
* 'D0E0F' 3/21/66
*
* DELETE REMAINDER OF FILE
*
D0E0F BRM DSS; LDX BIP,2; CLA
D0E0F1 EAX 1,2; SKE 0,2; BRU D0E0F1
      LDB BIP,2; BRM DTM; BRU NP0PX
*
* 'DFSR', 'DBSR' 12/7/65
*
* SPACE FORWARD OR BACKWARD (B) LOGICAL RECORDS
*
DFSR  LDA SS02; SKG =0; BRU DBSR; LDX BUFF
      LDA 1,2; SKA E0RBIT; SKR SS02; BRM DSS
      LDB DEV; SKB 0UTBIT; SKR BIP,2
DFSR1 LDA SS02; SKG =0; BRU DSX; BRM DSF; SKR SS02; BRU DFSR1
DSX   LDB DEV; SKB 0UTBIT; MIN BIP,2; BRU NP0PX
*
DBSR  BRM DSS; BRM DSZ
DBSR1 LDA SS02; SKG =0; BRU DSX; BRM DSB; SKR SS02; BRU DBSR1
*
* 'DREW', 'DWND' 12/7/65
*
* REWIND OR WIND, THEN SPACE (B) RECORDS IN OPPOSITE DIRECTION
*
DREW  BRM DSS; LDA BBP,2; SKG =0; BRU DREW1
      LDA =-1; STA BDN,2; BRM BSET; BRU DTG5A
DREW1 CXA; ADD =BX0-1; SKB 0UTBIT; ADD =1; STA BIP,2; BRU DFSR
*
DWND  BRM DSS; LDA BFP,2; SKG =0; BRU DWND1
      LDA =-1; STA BDN,2; BRM BSET; BRU DTG5B
DWND1 CXA; ADD =BX0+NDXW-3; STA BIP,2
DWND2 LDA+ BIP,2; SKE =0; BRU DWND3
      SKR BIP,2; NOP; BRU DWND2
DWND3 MIN BIP,2; BRU DBSR
      ENDF
*
IF      V3

```

```

* BRS BE+5
* SET DISC BIT MAP
* INPUT: SS01=DISC ADDR. OF X-BLOCK RIGHT SHIFTED 2 PLACES.
SDBMA SKN SDBM8; BRM NTRPB
      SKN SS01; BRU SDBM9; MIN SDBM8; MIN 0; LDA =-1; STA N0ACT
      STB N0VP; LDX SS03; STX ACA; BRU NP0PX
SDBM9 BRM BGET; BRM M0NCR; BRM BSET (BUFFER ADDR. IN T)
      LDA BUFF; ADD =27600000B+NDXWC+BX0; STA SDBM3+1
SDBM2 LDA SS01; CLB; LSH 2; STA SS01; BRM L0CBIT (FIND X-BLOCK)
      BRU SDBM5 (INVALID ADDR.); BRU SDBM12 (OUTSIDE CYLINDER)
      E0R =-1; ETR TABLE,2; SKE TABLE,2; BRU ++2
      BRU SDBM5; STA TABLE,2
SDBM12 LDB SS01; LDA T; ADD =BX0; LDX =NDXWR; BRM DTC
      BRM DST; BRU ++2; BRU SDBM4
*          REWRITE CLEANED UP XBLOCK IF SWITCH SET >=1
      SKN XCLEAN; BRU ++2; BRU SDBM3-1
      LDA BUFF; ADD =BX0; STA SDBM13
      LDX =-NDXWR; CLB
      LDA* SDBM13; MIN SDBM13; SKG X1; BRX *=3
      CXA; SKA ADMSK; BRU ++2; BRU SDBM3-1
      STB* SDBM13; MIN SDBM13; BRX *=2
      LDA T; ADD =BX0; LDB SS01; LDX =NDXWR
      BRM DTW; BRM DST; BRU ++2; MIN SD4
      LDX =-NDXWC
SDBM3 STX SDBM10; LDA NDXWC+BX0,2
      SKE =0; BRU SDBM6
SDBM7 LDX SDBM10; BRX SDBM3
      LDX BUFF; LDA NDXWC+BX0-2,2; STA SS01; SKE =0; BRU SDBM2
      MIN 0; BRU SDBM11
SDBM6 CLB; LSH 2; ETR DMSK; STA SS01
      BRM L0CBIT; BRU SDBM4; BRU SDBM7
      E0R =-1; ETR TABLE,2; SKE TABLE,2; BRU ++2
      BRU SDBM4; STA TABLE,2; BRU SDBM7
SDBM4 MIN XBERR; BRU SDBM11
SDBM5 MIN FDERR; BRU SDBM11
SDBM11 LDA BUFF; BRM BPUT; BRU NP0PX
SDBM8 DATA =1
SDBM13 ZR0
SDBM10 ZR0 0

```

SD4 ZR0 0  
ENDF  
ENDDSC BSS 0  
END

WRITE FAILURES

3100 IDENT 7/02/67

\* ENTRY POINTS

ENTRY PTAPE  
ENTRY TXC1, TXC2, TXS1, T2JFM, CKBUF  
ENTRY FTIME, ED, INTR, I0B, I0RW, I0SW, WBE, WBUF  
ENTRY MTDI, TGET, TREL, TN0, PDSW1  
ENTRY BPTST, NI0DMS, NI0QDS, NFI0DS  
ENTRY P0SL1A, P0SL1, P0SL2, RSTBUF, DADMSK, RLDMSK  
ENTRY I0DMS, I0QDMS, FI0DMS, ACTR, AIRWD  
ENTRY BGET, BSET, BPUT, UNIT  
ENTRY XX, X0, X1, X2, X3, X4, X5, X6, X7  
ENTRY XN1, XN2, XN3, XN4, XN5, XN6, XN7  
ENTRY 0UTBIT, CHRBIT, DRMBIT  
ENTRY DRXBIT, DR0BIT  
ENTRY E0RBIT, E0FBIT, ERRBIT, E0TBIT  
ENTRY FILE, FFLST, FA, FD, FC, FW  
ENTRY BUFF, DEV, DIU, 0PNDEV, BUFS, ADIU, EDIU  
ENTRY I0I, GPW, GPWF, CLS, DCLS3, CBF, I0IE  
ENTRY M0N0PN, M0NCLS, I0H, I0RET, RDU  
ENTRY BLK31, INT31, INT33, WRRL3

E0RBIT DATA 00100000B  
E0FBIT DATA 00200000B  
ERRBIT DATA 00400000B  
E0TBIT DATA 01000000B  
XX DATA 37777777B  
X0 DATA 00000000B  
X1 DATA 10000000B; XN7 EQU X1  
X2 DATA 20000000B; XN6 EQU X2  
X3 DATA 30000000B; XN5 EQU X3  
X4 DATA 40000000B; XN4 EQU X4  
X5 DATA 50000000B; XN3 EQU X5  
X6 DATA 60000000B; XN2 EQU X6  
X7 DATA 70000000B; XN1 EQU X7  
WBDBIT DATA 10000000B  
SBFBIT DATA 00400000B  
CHRBIT DATA 10000000B  
0UTBIT DATA 00100000B  
DRMBIT DATA 04000000B  
DRXBIT DATA 02000000B



```

DR0BIT DATA 01000000B
* FILE CONTROL TABLES
IF FCB
FA DATA 0,0,0
ELSF 1
FA DATA 7700000B,7700000B,7700000B
ENDF
BSS NFILE=3;* INDEX BLOCK OR SUBROUTINE ADDRESS, JOB NUMBER
$EFA EQU FA+NFILE
$EFA1 EQU FA+NFILE-1
FW BSS NFILE;* WORD BEING PACKED OR UNPACKED
FD ZR0 NDEV,1; BRU NDEV+1,1; BRU NDEV+4
BSS NFILE=3;* DEVICE NUMBER AND INDICATORS
IF FCB
* MAKES JOB NUMBER 77B UNAVAILABLE
FC DATA 7700000B,7700000B,7700000B
ELSF 1
FC ZR0; ZR0; ZR0
ENDF
BSS NFILE=3;* CHAR COUNT, DRUM BUFFER ADDRESS
* DEVICE DISPATCHER
DEV ZR0 0 USED TO HOLD DEV[N]
6B GPWN PAPER TAPE INPUT
7B GPWN PAPER TAPE OUTPUT
0B TRAP WAS CARD INPUT 2/14/67
6B GPWN MAG TAPE INPUT
7B GPWN MAG TAPE OUTPUT
1B NTRAP,1 CARD PUNCH
1B NTRAP CARD PUNCH BINARY
44B GPWSI DRUM INPUT
45B GPWS0 DRUM OUTPUT
64B NTRAP RANDOM DRUM FILE
5B GPWLP,1 LINE PRINTER OUTPUT
0B NTRAP CARD INPUT
0B NTRAP CARD INPUT BINARY
NDEV EQU *-DEV
NNDEV EXT -NDEV
ZR0 FTCI,1 TELETYPE INPUT
BRU FTC0,1 TELETYPE OUTPUT

```

```

IF      V8
ZR0    TRAP
BRU    TRAP
ELSF   1
ZR0    ISTD,1      SPECIFIED TELETYPE INPUT
BRU    OSTC,1     SPECIFIED TELETYPE OUTPUT
ENDF
BRU    GPWX      'NOTHING'
ZR0    GPWI0S    SUBROUTINE FILE

```

\* BUFFER ADDRESSES

```

IF      V2
BUF    EQU      34000B
RPT    NBUFA
BUF    EQU      BUF+NDBW
DATA   BUF
ENDR
ENDF

```

```

IF      V2
BUF    EQU      FBADR
ELSF   1
BUF    EQU      DBTOP+5
ENDF

```

```

RPT    NBUFX
DATA   BUF
BUF    EQU      BUF+NDBW
ENDR

```

```

IF      V2
BUF    EQU      *NBUFX
ELSF   1
BUF    EQU      *NBUFA-NBUFX
ENDF

```

\* BUFFER LENGTHS AND UNIT COUNTS

```

NTAPE1 EQU      NTAPE*100000B+40000B
NDDW1  EQU      NDDW+10000000B
BUFS   DATA    0,RTCNT,PNCNT,0,NTAPE1+TCNT,NTAPE1+TCNT
        DATA    CPCNT,CPCNTB,NDDW1,NDDW1,NDDW1,LPCNT,CRCNT,CRCNTB
$EBUFS EQU      BUFS+NDEV

```

\* DRIVER DISPATCHER

\* 0P CODE IS PU ACT. ROUTINE INDEX

\* X1 PROVIDES COMPUTATION OF IORW, IOSW BY ED.

```
SEL  ZR0  0
      3    RTX,1
      3    PNX,1
      0    TRAP
      2    TRX,1
      2    TWX,1
      4    CPX,1
      4    CPXB,1
      0    DRMSI
      0    DRMS0
      ZR0  0
      8    LPX,1
      2    CRX,1
      2    CRXB,1
```

\* B10 DRIVER DISPATCHER

```
BDEV  DATA 0,BIS,BIS,BIG,BIS,BIS
      DATA TRAP,TRAP
      DATA SBI,SB0,TRAP,BIG
      DATA BIG,BIG,BIG,BIG,BIG,TRAP
```

\* DEVICE-IN-USE TABLES

\* ENTRY SET TO \*1 WHEN FREE

```
DIU   BSS 4; DATA ADIU,ADIU; BSS NDEV-6
```

```
ADIU  BSS NTAPE+NLINK*2
```

```
EDIU  EQU *
```

```
$LADIU EQU ADIU-EDIU
```

\*

\*

\* FILE OPENING DRIVERS

\* 3 ARGUMENT 0PNT MACRO

\* A(1): DEVICE ACCESS TIME IN MILLISECONDS

\* A(2): DRIVER ADDR,

\* A(3): 1=EXEC ONLY ALLOWED TO OPEN.

\* MACRO CONVERTS A(1) TO CLOCK TICKS. PUTS RESULT IN 0P CODE FIELD.

\* A(3) IS PUT INTO BIT 2.

\*

\* 2 ARGUMENT 0PNT MACRO

\* A(1): DEVICE ACCESS TIME

\* A(2): 1=EXEC ONLY ALLOWED TO OPEN.

```

0PNT  MACR0  A
0PNDEV NARG
      IF      0PNDEV#2
      0PNT   A(1),0,A(2)
      ELSF   1
0PNDEV EQU  A(1)*6/100(AND)77B+A(3)*100B
          ($0PNDEV)  A(2)
      ENDF
      ENDM

```

```

ZR0
0PNT  RTWT,0
0PNT  PNWT,0
0PNT  0,TRAP,0
0PNT  TXWT,MTR0PN,1
0PNT  TXWT,MTW0PN,1
0PNT  CPWT,NTRPB,1
0PNT  CPWT,NTRPB,1
0PNT  34,DRM0PN,1
0PNT  34,DRM0PN,1
0PNT  34,NTRPB,1
0PNT  LPWT,LP0PN,1
0PNT  CRWT,NTRPB,1
0PNT  CRWT,NTRPB,1
0PNDEV EQU  *=NDEV

```

```

      IF      V6
* 'CIT' 12/15/66  V. VAN VLEAR
* CHARACTER INPUT AND TEST
* INPUT: CHARACTER TO TEST IN A
*      ADDR. #FILE NO.
* RETURN: NO SKIP: NO COMPARE. LEAVE CHAR. IN BUFFER, AND
*      RETURN CHAR. IN A.
*      SKIP: COMPARES. TAKE CHAR. OUT OF BUFFER.

```

```

CIT  P0PD  13400000B,1,1,0,1
CITS STA SS01; STB SS02; STX SS03
LDA*  0      GET FILE NO.
SKA   *(N0T)1  CK FILE TYPE

```

	BRU	CIT1	
	SKE	=0	TELETYPE
	BRM	TRPB	OUTPUT, NOT VALID
	LDB	=I0X-1	
	STB	GPW	SET RETURN
	LDX	UTTY	
	LDA	TIS2,2	CK CHAR COUNT
	SKG	=0	
	BRU	CTI3	NO INPUT CHARS IN BUFF
	LDX	TIS5,2	
CTI1	EAX	1,2	INC. INPUT BUFF PTR TO GET CHAR
	LDB	0,2	GET CHAR
	SKB	=200B	CK IF END OF BUFF
	BRU	CTI2	YES, GO END AROUND
	CLA		
	LCY	8	POS CHAR IN A
	SKE	SS01	CK IF EQUAL TO CHAR IN A
	BRU	++3	
	MIN	0	
	BRU	FTCI	COMPARES SO TAKE OUT OF BUFF
	STA	T	PUT THIS CHAR IN A BUT DONT TAKE OUT OF BUFF
	BRR	GPW	
CTI2	STX	T	
	CBA		
	ADM	T	GO BACK TO START OF BUFF
	LDX	T	
	BRU	CTI1	
CTI3	LDB	XX	SET FLAG, NEED A CHAR.
	STB	TTYBRK,2	
	CXA		
	ADD	=600000B	
	ADD	=TTYBRK	
	CAB		
	BRU	TIDMS	SET ACTIVATION FOR NEXT INPUT CHAR.
CIT1	BRM	I0I	PROCESS FILE NO
	SKB	OUTBIT	CK IF OUTPUT

```

BRM      TRPB      YES, NOT VALID FOR CIT
SKB      CHRBIT
BRU      CIT2
SKN      FC,2
BRU      CIT3
BRM      GPW
LOA      T
LDX      FILE
STA      FW,2
LDA      X3
ADM      FC,2
CIT3     CLA
LDB      FW,2
LCY      8
SKE      SS01      CK IF EQUAL TO CHAR IN A
BRU      I0X+1
MIN      0
LDB      FD,2
BRU      CI3

```

```

CIT2     CLA
STA      SWCIT
BRU      CI2

```

```

$SWCIT DATA =1
ENDF

```

```

*
* 'CI0' 2/3/66
*

```

```

CI0      P0PD      16100000B,1,1,0,1
CIE      STA SS01; STB SS02; STX SS03
LDA* 0; SKA =(NOT)1; BRU CI1; LDB =I0X=1; STB GPW
SKE =0; BRU FTC0; BRU FTCI
CI1      BRM I01; SKB CHRBIT; BRU CI2; SKN FC,2; BRU CI3
LDA FW,2; STA T; BRM GPW
LDA T; LDX FILE; STA FW,2; LDA X2; LDB FD,2; BRU CI5
CI2      LDA SS01; ETR =377B; STA T; BRU W12
CI3      LDA XN1

```

CI5 ADM FC,2; SKB OUTBIT; BRU CI4  
CLA; LDB FW,2; LCY 8; STB FW,2; BRU I0X+1  
CI4 LDA SS01; RCY 8; LDA FW,2; LCY 8; STA FW,2; BRU P0PXI0

\*  
\* 'WI0' P. DEUTSCH 9/13/65  
\*

WI0 P0PD 16000000B,1,1,0,1  
WIE STA SS01; STA T; STB SS02; STX SS03  
LDA\* 0; BRM I0I; SKB CHRBIT; BRM NTRPB  
WI2 BRM GPW; LDB DEV; SKB OUTBIT; BRU P0PXI0

\*  
\* I0 EXIT

I0X LDA T; STA SS01; BRU P0PXI0  
WI1 BRM WIT; BRU CI2+1

\*  
WIT ZR0; SKB OUTBIT; BRU WI3; LDA FC,2  
WI5 LDB X7; SKM X2; BRU WI4  
LDA XN3; ADM FC,2; CLA; XMA FW,2; STA T; BRR WIT  
WI4 LDB DEV; BRM GPW  
LDA T; RCY 8; LDX FILE; LDA FW,2; LCY 8; STA FW,2  
LDA FC,2; ADD X1; STA FC,2; BRU WI5  
WI3 LDA FC,2; ETR X7; LDB T; SKE X2; LDB FW,2  
WI7 STB FW,2; CLA; LCY 8; LDB DEV; BRM GPW  
LDX FILE; CLA; LDB FW,2; LCY 8  
LDA XN1; ADM FC,2; SKN FC,2; BRU WI7  
LDA X3; ADM FC,2; BRR WIT

\*  
\* 'BI0' 12/15/65  
\*

\* BLOCK I/O, KLUDGE VERSION  
\* INPUT: A=NUMBER OF WORDS, X=1ST LOCATION  
\* OUTPUT: A=1ST MEMORY LOC. NOT READ INTO.

\*  
BI0 P0PD 17600000B,1,1,0,1  
BI1 STA SS01; STB SS02; STX SS03  
LDA\* 0; BRM I0I; SKB CHRBIT; BRM <sup>15-12-65</sup>NTRAP; CBX; BRU\* BDEV,2  
SB0 CAX; MIN BDC,2 (TEMP.)  
SB1 EQU \* (TEMP.)  
BIS CAX; LDA SS01; SKG =0; BRU BI3

```

LDA 1,2; SUB 0,2; ETR ADMSK; SKG =0; BRU BIG
SKG SS01; BRU **2; LDA SS01; STA T
ADD 0,2; STA BI4; LDA T; CNA
COPY AX,N; ADD SS03; ETR ADMSK
SKB OUTBIT; BRU BI6; MRG BIC2; STA BI5; LDA BIC1; ADM BI4
BI4 LDA 0,2
BI5 STA 0,6; BRX BI4; LDA T; ADM SS03; ADM* BUFF
CNA; ADM SS01; LDB DEV; LDA BUFF; BRU BIS
BI6 MRG BIC3; XMA BI4; MRG BIC4; STA BI5; BRU BI4
BIC1 LDA 0,2
BIC2 STA 0,6
BIC3 LDA 0,6
BIC4 STA 0,2
BIG BSS 0
BI2 LDX SS03; LDA 0,6; STA T
SKB CHRBIT; BRU BI1; BRM GPW
BI7 LDX SS03; LDA* 0; SKA #77740000B; BRU BI3A
LDA T; STA 0,6; LDB DEV
MIN SS03; LDA SS01; SKG =1; BRU BI3; SKR SS01; BRU BI2
BI3 MIN 0
BI3A LDA SS03; STA SS01; BRU POPXI0
BI1 BRM WIT; BRU BI7
*
POPXI0 LDA SS01; LDB SS02; LDX SS03; SKN TIME; BRR 0
MIN 0; MIN ACTR; LDB PACDMB; LDA QUTAB; STA TTIME
LDX =QTI; BRU POPDMS
*
*
* !I0! 2/3/66
*
* THIS ROUTINE IS CALLED BY 'IC0' AND 'WI0' TO CHECK THE VALIDITY
* OF THE FILE NUMBER AND TO SET UP 'FILE', 'BUFF', AND 'DEV'.
* INPUT: A=FILE NO.
* OUTPUT: B=DEV
*
FILE ZR0;* FILE NUMBER
BUFF ZR0;* BUFFFER ADDRESS
UNIT ZR0;* UNIT NUMBER
ACTR ZR0;* ACTIVATION COUNT

```



```

AIRWD  IF      V1
      DATA 770000B
      ENDF

*
I0IC1 ZR0 NDEV+2,1  INPUT TO SPECIFIED TELETYPE
I0IC2 BRU NDEV+3,1  OUTPUT TO SPECIFIED TELETYPE
*
*
I0I   ZR0; ETR ADMSK; STA FILE; SKG =NFILE-1; BRU I0I4
* CHECK FOR SPECIFIED TELETYPE
      SKG =777B+NTTY; SKG =777B; BRU I0I5
* INPUT TO SPECIFIED TELETYPE
      LDB I0IC1
I0I6  ETR =77B; STA UNIT; STB DEV; BRR I0I
I0I5  SKG =1777B+NTTY; SKG =1777B; BRM NTRPB
* OUTPUT TO SPECIFIED TELETYPE
      LDB I0IC2; BRU I0I6
* REGULAR FILE (NOT SPECIFIED TELETYPE)
      IF      FCB
I0I4  AXC; LDB FC,2; LCY 9; ETR =77B; SKE JOB; BRU I0I1
      ELSF  1
I0I4  AXC; LDB FA,2; LCY 9; SKE JOB; BRU I0I1
      ENDF
I0I2  LDB FD,2; SKB X6; BRU I0I7; STB DEV
      LDA FC,2; ETR ADMSK; STA BUFF; BRR I0I
* CHECK FOR TELETYPE OR NOTHING
I0I1  SKE =77B; BRM NTRPB; BRU I0I2
I0IE  CAB
* ERROR OF BUFFER BUSY
I0I7  SKB X2; BRU NI0DMS; LDA XN4; ADM FD,2
      LDA =I0I8-1; STA GPW; BRM MPPACT; LDA ERRBIT; CLB; BRU GPW2
I0I8  BRU POPX
*
*
*      'INTERRUPT 31'
*
* INTERRUPT 31. W BUFFER INTERRUPT ROUTINE.
BLK31 ZR0;* I31/I33 SUBROUTINE. SET=0 WHEN FREE. SET GT 0 WHEN BUSY
ISA   ZR0;* SAVE (A)

```

```

ISB      ZR0;* SAVE (B)
ISX      ZR0;* SAVE (X)
TNO      ZR0;* TAPE NUMBER
WFILE    ZR0;* FILE NUMBER
WBUFF    ZR0;* BUFFER ADDRESS
WBE      ZR0;* END OF WBUFF
WSRRL3   ZR0;* SAVE RRL3
WRRL3    ZR0;* SET UP RRL3
INTR     ZR0;* INTERRUPT ROUTINE ADDR
WBSV     ZR0;* TEMP STORAGE
*
INT31    ZR0; STA ISA; STB ISB; STX ISX
INT33    EQU INT31
          LDA WRRL3; XMA RRL3; STA WSRRL3
          LRR3; PBT RRL3; LDA WBUFF; AXC
          XMA BLK31; STA INTR; BRM* INTR
          BRU INTF (DRIVE DEVICE)
          STB WBSV (DEVICE NOT FREE)
          LDX WFILE; LDX FC,2; COPY XA,AX,B; RSH 15
          ETR #77B; XXA; LDB WBSV; LDX TTN0,2
          DIR; BRM EPU
INTF     LDA BLK31; SKG #0; BRU INTU
INTX     LDA WSRRL3; STA RRL3; LRR3; PBT RRL3
          LDA ISA; LDB ISB; LDX ISX; BRI INT31
INTU     LDX WFILE; LDA XN2; ADM FD,2; MIN ACTR
          LDA WRRL3; LDB WBUFF; SKB #4000B; LCY 6
          RCY 6; ETR #77B; CAX; SKR RMC,2; NBP; BRU INTX

```

```

*
*
*
* !GPW! P. DEUTSCH 9/13/65
*
* THIS ROUTINE GETS OR PUTS ONE WORD ON THE DEVICE SPECIFIED IN (B).
* IT WILL DISMISS THE USER IF THIS IS NOT IMMEDIATELY POSSIBLE.
* IT TAKES CARE OF SUBROUTINE FILES AS WELL.
* INPUT; A=T=WORD, B=DEVICE.
* OUTPUT; T=WORD
*
GPW      ZR0; CBX; BRU* DEV,2

```

```

* BUFFER BUSY
NI0DMS BRM MPPACT; BRU I0DMS
NI0QDS BRM MPPACT; BRU QT0DMS
NFI0DS BRM MPPACT
FI0DMS BSS 0
I0DMS LDA FILE; ADD =FD+1100000B; CAB
QT0DMS LDX =QT1; BRU P0PDMS
I0QDMS LDX =QI0; BRU P0PDMS
* SUBROUTINE FILE
GPWI0S LDX PACPTR; LDA PL,2; ETR =7700000B
      LDX FILE; CXB; LDX FA,2; E0R 0; E0R =3240000B; STA* 0,6
      ADD =-1 (SET X0); STX 0; LSH 39; LDX PACPTR; STA PL,2
      LDX FILE; LDA GPW; ETR ADMSK; XMA FC,2; ETR X7; ADM FC,2
      LDA T; LDB SS02; LDX SS03; SKN TIME; BRR 0; BRU XP0P
*
* BRS 41 RETURN FROM I/O SUBROUTINE
I0RET STA T; LDA PL,2; ETR =7700000B; RSH 15
      SKG =0; BRM TRPB; BRM I0I
      LDA FC,2; ETR ADMSK; STA GPW; LDX FA,2; LDA* 0,6
      LDX PACPTR; STA PL,2; STA 0; BRR GPW
* DISC I/O
GPWS0 LDX BUFF; MIN BDC,2
GPWSI BSS 0
* NORMAL CASE, NOT BUSY
GPWN LDX BUFF; LDA 1,2; SUB 0,2; SKA ADMSK; BRU GPW6
      LDB =-1
      STB SWCIT
* BUFFER EMPTY
GPW5 SKE =0; BRU GPW8
* NO FLAGS, START DEVICE
GPW7 LDA FILE; LDX SS03; LDB 0; SKB =40000B; STA* 0
GPWD LDX DEV; LDB SEL,2; LDX BUFF
      BRM ED; BRU NFI0DS (DR0VE DEVICE)
      BRU NI0QDS (CHANNEL NOT READY=BLK31)
      BRU NI0DMS (DEVICE NOT READY)
* BUFFER EMPTY BUT FLAGS TO REPORT
* E0R,E0F,ERROR ALWAYS TURN ON X4 IN 'FILE'
GPW8 MIN BUFF; SKA ERBIT; BRU GPW8A; SKA E0RBIT; BRU GPW8B
GPWF LDB =27657537B (137B)

```

```

GPW12  CNA; ADM* BUFF; CNA
* INPUT; A=FLAG BIT. B=RETURN STATE.
* STATE; 0=NØ DISC SPACE LEFT, -1=DEVICE IN USE, -2=TØØ MANY FILES ØPEN
GPW2   STB T; MRG FILE; MRG X4; LDX SSØ3; LDB 0
      SKB #4000ØB; STA* 0; CAX; LDA FC,2; LDB FD,2
      MRG X7; SKB ØUTBIT; ØØR X5; STA FC,2
      LDA #20000ØB; LDX PACPTR; BRM IIR; BRR GPW
      LDA T; STA SSØ1; BRU PØPINT
GPW8B  ETR ØØRBIT; LDB #27Ø56134B (134B); BRU GPW12
GPW8A  LDA ERRBIT; CLB; BRU GPW12
* BUFFER NØT EMPTY
GPW6   SKB ØUTBIT; BRU GPW9
      LDA*   0,2
      SKN    SWCIT
      BRU    CGPW6
CGPW6A MIN    0,2; STA T; BRR GPW
* ØUTPUT, NØT INPUT
GPW9   LDA T; STA* 0,2; MIN 0,2
GPWX   BRR GPW

*
*
CGPW6  IF      V6          ENDF BEFORE 'CTRL'
      LDB     #=1
      STB     SWCIT
      SKE     SSØ1
      BRU     CGPW6A+1
      MIN     0
      BRU     CGPW6A
      ENDF

*
*
* PRINTER PUT ROUTINE, TRANSLATES ASC TO 6 BIT SDS CHARS, FILLS
* BFR , HANDLES MLTPLE BLANKS AND NØRMAL PRINTER CØNTRØLS
GPWLP  LDA T; CXB; LDX BUFF; SKN PDSW; BRU PD1          (GET BLANKS CØUNT)
      SUB #1; STA PDCNT; CLA; STA T
PD2    LDA 1,2; SUB 0,2; SKA ADMASK; BRU PD3 (RØØM IN BUFFER)
PD7    MIN PDSW; * BUFFER FULL; BRU GPW
PD3    LDA T; SKG #77B; BRU PD4 (PUT CHAR. IN BUFFER)

```

```

SKE =155B; BRU **2; BRU PD6; SKE =152B; BRU PD5 DRIVE IF CR OR LF
PD6  CNA; STA PDSW1
PD11 CLA; SKE BLK31; BRU PD7; MIN 0; BRU PD7
PD5  SKE =137B; BRU PDEX; BRU PD6
PD1  BRU PD10
PD10 SKN PDSW2; BRU PD12; LDA P0SL1A; STA P0SL1; STA PDSW2
PD12 LDA T; SKE =135B (MULT. BLANK CHAR.); BRU PD9
      LDA =-1; STA PDSW
PDEX BRR GPW
PD9  SKE =133B; BRU PD2; CNA; STA PDSW2; BRU PD11
PD4  LDA T; RSH 2; COPY AX,A (CHARACTER NUMBER)/4
      LSH 2; MUL =3; LDA PRCHRS,2 (LDA WITH CHR WRD,CMPTE SHIFT)
      CBX; LSH 0,2; ETR =77000000B (LEFT ADJUST CHAR)
      LDX BUFF; STA* 0,2; MIN 0,2 (PUT CHAR IN BUFFER)
      SKR PDCNT; BRU PD2; MIN PDSW; MIN PDCNT; BRU PDEX (EXT NOT BLKS)
PDSW ZR0      0          -1 WHEN LOOKING FOR CHAR. COUNT
PDCNT ZR0      0
PDSW1 ZR0      0          IGNORE NEXT CHAR
PDSW2 ZR0      0          SKIP TO TOP OF PAGE
P0SL1 SKIPT0   1
P0SL2 SPACE    1
P0SL1A SKIPT0  1
PRCHRS DATA   60571417B,53575714B,74345420B,73403361B
      DATA   00010203B,04050607B,10111556B,36131632B
      DATA   25212223B,24252627B,30314142B,43444546B
      DATA   47505162B,63646566B,67707135B,76555717B

```

```

*
*
*
*
*

```

```

* 'IED' 6/27/66

```

```

* EXECUTE DRIVER OR CONTROL ROUTINE.

```

```

*
FTIME ZR0;* EXPECTED WAIT TIME
EDD   ZR0;* DRIVER ADDRESS
I0RW  ZR0;* I0RD INSTRUCTION
I0SW  ZR0;* I0SD INSTRUCTION
I0B   ZR0;* REAL BUFFER ADDRESS\ADMSK
TXC1  DATA TCNT*40000B+40001B;* BUFFER CONTROL WORD
TXC2  DATA 17000000B;* FILE MARK

```

```

TXS1  ZR0;* ASCW WORD FROM BUFFER
T2JFM HLT* TXC2          PBT WORD FOR END OF FILE
*
* INPUT; B=DRIVER ADDRESS.
* OUTPUT; N0 SKIP= DRIVE DEVICE
*       SKIP * CHANNEL NOT READY (BLK31 NE 0)
*       2 SKIP = UNSUCCESSFUL OF DEVICE NOT READY
*
ED    ZR0; STB EDD
      IF      FCB
      LDX FILE; LDA FA,2; ETR =77B; STA UNIT
      ELSF   1
      LDX FILE; LDA FC,2; RCY 15; ETR =77B; STA UNIT
      ENDF
      LDX FD,2; LDA 0PNDEV,2; RCY 15; ETR =77B; STA FTIME
      LDX BUFF; BRM BSET; LDB EDD; SKB =177000Q0B; BRU EDWMP
      BRM MPDSC; BRM* EDD; BRU EDF
EDW   CLA; SKE BLK31; BRU EDWI; LDA UNIT; STA TN0
      LDA T; RCY 9; ETR =140B; MRG =216000B; STA I0RW
      MRG =200B; STA I0SW; LDA T; ETR ADMSK; STA I0B
      LDA RRL3; STA WRRL3; LDA BUFF; STA WBUFF
      ADD =2; LDB DEV; CBX; SKB 0UTBIT; ADD BUFS,2
      ETR ADMSK; STA WBE; LDA FILE; STA WFILE
      LDX BUFF; BRM* EDD; BRU EDWB; BRU EDWS
EDWB  LDA BLK31; SKG =0; BRR ED
      LDA T; LRSH 11; CAX; MIN RMC,2
EDF   LDX FILE; LDA X2; ADM FD,2; BRR ED
EDWI  LDB EDCI; MIN ED; BRR ED
EDCI  1 BLK31
EDWMP BRM MPWB; BRU EDW
EDWS  LDX FILE; LDA FC,2; RSH 15; ETR =77B; CAX; LDA TTN0,2
      ETR =77B; CAX
      LDB FILE; LDA EDD; DIR; BRM EPU
      MIN ED; MIN ED; BRU EDF
PTAPE ZR0 0  SEE PAPER TAPE DRIVER
*
*
*
* !CTRL! 10/31/65

```

```

*
* THIS HANDLES I/O CONTROL FUNCTIONS
*
* INPUT: A=CONTROL NUMBER, ADDRESS =FILE NUMBER.
CTRL  P0PD      17200000B,1,1,0,1
CTL   STA SS01; STB SS02; STX SS03
      LDA* 0; BRM I0I
      COPY BA,BX; ETR ADMSK; SKG =NDEV=1; BRU **2; BRM TRPB
      LDA SS01; ETR ADMSK; LDB 0PNDEV,2; SKB X1; BRU CTL1
CTL2  SKG* CTLT,2; SKG =0; BRM TRPB; ADD CTLT,2; CAX
      LDB DEV; SKB DRMBIT; BRU **3; BRM MPWB
      BRU **2; BRM MPDSC
      LDA 0,2; STA CTLW; LDX FILE
      ETR =7777777B; XMA CTLW; SKE CTLW; BRM CBF; N0P
      LDB CTLW; SKB =4000000B; BRU* CTLW
      BRM ED; BRU NP0PX; BRU NI0QDS; MIN 0; BRU NI0DMS
CTL1  LDX PACPTR; SKN PQU,2; BRM TRPB; LDX DEV; BRU CTL2
* CONTROL DISPATCHER
CTLW  ZR0      0 CONTROL ROUTINE ADDRESS.
CTLT  DATA   CTT0,CTT1,CTT2,CTT3,CTT4,CTT5,CTT6,CTT7
      DATA   CTT8,CTT9,CTT10
*
* BIT 2=1 MEANS DUMP THE BUFFER FIRST.
* BIT 3=0 MEANS GO TO ED FIRST.
* BIT 3=1 MEANS GO TO CTRL ROUTINE DIRECTLY.
CTT0  DATA 0
CTT1  EQU CTT0
CTT2  DATA 1; 102B PNE0R
CTT3  EQU CTT0
CTT4  DATA 6; 40B NTRAP; 2 MBSR; 2 MFSF; 2 MBSF
      40B NTRAP; 2 MREW
CTT5  DATA 8; 102B ME0R; 102B MBSR; 102B MFSF; 102B MBSF
      102B MERS; 102B MREW; 102B ME0F; 102B MERL
      IF      V1
CTT6  EQU     CTT0
CTT7  EQU     CTT0
      ENDF
      IF      V1
CTT8  DATA 6; 40B NTRAP; 40B NTRAP; 40B NTRAP; 40B NTRAP

```

```

      40B NTRAP; 40B NTRAP
CTT9  DATA 8; 140B NTRAP; 140B NTRAP; 140B NTRAP; 140B NTRAP
      140B NTRAP; 140B NTRAP; 140B NTRAP; 140B NTRAP
      ELSF      1
CTT8  DATA 6; 40B DIEOR; 40B DBSR; 40B DFSR; 40B TRAP
      40B DWND; 40B DREW
CTT9  DATA 8; 140B D8EOR; 140B DBSR; 140B DFSR; 140B DDLR
      140B DWND; 140B DREW; 140B DISR; 140B D8E0F
      ENDF
CTT10 EQU CTT0

```

```

*
* 'M0N0PN' 10/17/65
*
* THIS ROUTINE OPENS PHYSICAL FILES (BRS 1)
* INPUT: A=INDEX BLOCK ADDRESS OR UNIT.
*        X=DEVICE
* OUTPUT: A=FILE NO.
*         X=X BLOCK POINTER.
*
FFLST ZR0;* AVAILABLE FCB LIST
*
M0N0PN LDB SS03; LDA FFLST; SKG =0; BRU 0PN12
      COPY BX,BA,B; STB UNIT; SKA =77740000B; BRU 0PN2
      SKG =NDEV=1; SKG =0; BRM TRPB; STA DEV; ADD =DIU; STA DIU
* OPEN NORMAL FILE
      LDA DEV,2; SKA 0UTBIT; LDB BUFS,2; STB T
      LDA BUFS,2; SKA =40000B; BRU **2; BRU 0PN7
* MAG TAPE
      LRSH 15; ETR =77B; SKN SS01; SKG SS01; BRM TRPB
      IF      FCB
      LDB SS01; ETR =77B; STB UNIT; LDA SS01; ADD* DIU; STA DIU
      ELSF      1
      LDB SS01; LSH 15; STB UNIT; LSH 9; ADD* DIU; STA DIU
      ENDF
0PN7  LDA BUFS,2; SKA X1; BRU 0PN4; SKN* DIU; BRU 0PN11
0PN4  LDA DEV,2; ETR =16300000B; ADM DEV
      LDB DEV,2; LDA 0PNDEV,2; STA 0PNDEV; SKG X1; BRU 0PN6

```



```

LDX PACPTR; SKN PQU,2; BRM TRPB
0PN6 SKB SBFBIT; BRU **2; BRU 0PN8
BRM BGET; BRU 0PN12 (NO BUFFER); CXA; ADD #2; STA 0,2
ADD T; ETR ADMSK; STA 1,2
0PN8 CLA; STA T
LDA 0PNDEV; SKA ADMSK; BRU 0PN5
0PN9 LDA FFLST; STA* DIU; CAX; LDB DEV; STB FD,2
IF FCB
LDA JOB; SKB OUTBIT; BRU **2; MRG #700B; MRG #200B; CLB; LSH 15
ELSF 1
LDA X7; SKB OUTBIT; LDA X2
ENDF
MRG BUFF; IF #FCB; MRG UNIT; ENDF; STA FC,2; BRU 0PN3
0PN5 LDB DEV; XAB; SKA DRMBIT; BRU **3; BRM MPWB
BRU **2; BRM MPDSC; XAB; BRM* 0PNDEV; BRU 0PN13; STA SS01
* ERROR RETURN
BRM MPPACT; LDX DEV; LDB DEV,2; LDA BUFF
SKB SBFBIT; BRM BPUT; BRU P0PX
0PN13 BRM MPPACT; BRU 0PN9
0PN12 LDA #-2; BRU 0PN10
0PN11 LDA #-1
0PN10 STA SS01; BRU NP0PX
* OPEN SUBROUTINE FILE
0PN2 STA T; ETR ADMSK; XMA T; LDX FFLST
ETR #10100000B; ADD #NDEV+5; STA FD,2
LDB X7; SKA OUTBIT; LDB X2; STB FC,2
IF FCB
0PN3 LDA T; MRG UNIT; XMA FA,2; EOR X1
ELSF 1
0PN3 LDB JOB; LSH 39; MRG T; XMA FA,2; EOR X1
ENDF
XMA FFLST; ETR ADMSK; STA SS01; MIN 0; BRU P0PX
*
*
* OPEN PRINTER, INITIALIZE SWITCHES
LP0PN ZR0; PRT; BRU 0PN11; LDB P0SL1A; STB P0SL1
CLB; STB PDSW1; STB PDSW2; BRR LP0PN
* 'MONCLS', 'IOH' 10/16/65
*

```

\* THESE ROUTINES CLOSE PHYSICAL FILES (BRS 2, BRS 8)

\*

\* BRS 2

M0NCLS BRM CLS; BRU P0PX

\* BRS 8

IF FCB  
I0H LDB J0B; LSH 39; LDB =7700000B; LDX =-NFILE  
IH1 SKM FC+NFILE,2; BRU IH2; CXA; ADD =NFILE  
~~ELSF 1~~

~~I0H LDB J0B; LSH 39; LDB =17700000B; LDX =-NFILE  
IH1 SKM FA+NFILE,2; BRU IH2; CXA; ADD =NFILE  
ENDF~~

BRM CLS; BRU I0H

IF V5

IH2 BRX IH1; LDX J0B; LDA TTN0,2; ETR =77B; STA TTN0,2; BRU P0PX  
ELSF 1

IH2 BRX IH1; BRU P0PX  
ENDF

\*

\* DUMP THE BUFFER IF NECESSARY

\* INPUT; B=DEVICE.

CBF ZR0; SKB 0UTBIT; SKB CHRBIT; BRU CBF2

\* OUTPUT FILE, BUT NOT CHARACTER ORIENTED.

LDA FC,2; ETR X7; SKE X2; BRU CBF1; BRU CBF2

\* OUTPUT FILE WITH CHARACTERS IN FW.

CBF1 RSH 18; LDB FW,2; COPY AX,BA; LDB =27657537B (137B)  
LCY 8,2; STA T

LDX FILE; LDA FC,2; ETR =7777777B; MRG X2; STA FC,2  
LDB DEV; BRM GPW; LDA FILE; BRM I0I

CBF2 COPY BA,BX; ETR ADMSK; SKG =NDEV-1; BRU \*\*2; BRR CBF  
SKB DRMBIT; BRU CBF3; SKB 0UTBIT; BRU CBF4

CBF5 MIN CBF; BRR CBF

\* OUTPUT FILE, CHECK FOR DATA IN BUFFER.

CBF4 LDB DEV,2; SKB SBFBIT; BRU \*\*2; BRU CBF5  
LDA\* BUFF; SUB BUFF; SKG =2; BRU CBF5  
BRU GPWD

\* DISC FILE

CBF3 LDX BUFF; LDA BDC,2; SKB 0UTBIT; SKG =-1; BRU CBF5  
BRM BSET; BRM MPDSC

```

BRM DTD; LDA X1; ADM* BIP,2; BRU DTX
*
*
* THIS ROUTINE CLOSES ONE PHYSICAL FILE
* INPUT: A=FILE NUMBER.
CLS   ZR0; ETR ADMSK; SKG =2; BRR CLS; BRM I0I
      BRM CBF; BRU CLS3; SKB DRMBIT; BRU DRMCLS
CLS4  LDX DEV; LDA BUFS,2; LDB DEV,2; SKA =40000B; BRU CLS5
CLS6  LDA =-1; STA DIU,2; LDA BUFF; SKB SBFBIT; BRM BPUT
CLS3  LDX FILE; CLA; STA FC,2; LDA FFLST; MRG X1; STA FA,2
      STX FFLST; BRR CLS
* MAG TAPE
      IF      FCB
CLS5  LDX FILE; LDA FA,2; ETR =77B; STA UNIT
      ELSF    1
CLS5  LDX FILE; LDA FC,2; RCY 15; ETR =77B; STA UNIT
      ENDF
      LDX DEV; ADD DIU,2; SUB =DIU; LDB DEV,2; CAX; BRU CLS6
* DISC
DRMCLS BRM MPDSC; BRU DCLS4
DCLS3  BRM MPPACT; BRU CLS4
*
*
* !RDU! 6/30/66
*
* READ DEVICE AND UNIT
*
RDU    BRM I0I; CBA; ETR ADMSK; STA SS03; SKG =NDEV-1; BRU RDU1
RDU2   LDA UNIT; STA SS01; BRU P0PX
      IF      FCB
RDU1   LDA FA,2
      ELSF    1
RDU1   LDA FC,2; RCY 15
      ENDF
      ETR =77B; BRU RDU2+1
*
*
* W BUFFER BR'S
*

```

```

* TURN OFF RUN-AWAY TAPE
* BRS 114   CALLED BY 'ABT' AND PU
MTDI   CLA; SKE BLK31; SKN PQU,2; BRM TRPB; DISW
      LDX WFILE; LDA FD,2; SKA X2; BRU *+2; BRU P0PX
      LDA =T2K; STA BLK31; LDA =P0PX; STA* 31B; BRR 31B
* LOCK TAPE
* BRS 118
TGET   SKN PQU,2; BRM TRPB; SKG =NTAPE-1; SKG =-1; BRM TRPB
      CAX; SKN TJN0,2; BRU TGET1
      LDA J0B; STA TJN0,2; MIN 0; BRU P0PX
TGET1  LDA TJN0,2; SKE J0B; BRU P0PX; BRU TGET1-2
* UNLOCK TAPE
* BRS 119
TREL   SKN PQU,2; BRM TRPB; SKG =NTAPE-1; SKG =-1; BRM TRPB
      CAX; LDA =-1; STA TJN0,2; BRU P0PX
*
*
* 'BGET', 'BSET', 'BPUT' 6/28/66
*
* BUFFER MANIPULATION ROUTINES
RSTBUF ZR0; CXA; ETR ADMSK; ADD =2; STA 0,2; STA 1,2; BRR RSTBUF
*
BGET   ZR0; LDA FBWRD; SKG =0; BRR BGET; MIN BGET
      LDX =NBUF-1; N0D NBUF-1; E0R X2; RCY NBUF-1; LCY 0,2
      STA FBWRD; LDX BUF,2; STX BUFF; BRR BGET
*
*
* PUT REAL BUFFER ADDR. IN A AND T
BSET   ZR0; COPY XA,B; RSH 11; LDA RRL3; ETR =77B
      LCY 11; STA T; BRR BSET
*
BPUT   ZR0; ETR ADMSK; LDX =-NBUF; SKE BUF+NBUF,2; BRX *-1
      LDB =1; LSH 47,2
      MRG FBWRD; STA FBWRD; BRR BPUT
*
*
* BRS BE+17   TEST FOR LAST BUFFER FREE
CKBUF  LDA =4000000B; SKA FBWRD; BRU P0PX; BRM TRAPB
*

```

```

*
*
  IF      V6
* BRS BE+7   BREAK POINT TEST
* SKIPS IS BREAK POINT SWITCH IS DOWN.
* INPUT; X=SWITCH NUMBER.
BPTEST SKN PQU,2; BRM TRPB; LDX SS03
      LDA =1000B; RSH 0,2; XMA BPT2; ETR =77777000B
      ADM BPT2
BPT2   BPT; MIN 0; BRU PBPX
      ENDF
*
*
*
*
* ENTRY POINTS
      ENTRY  DBIA, DBBA, LASA, SASA, DWIA, DWBA
      ENTRY  DTXS1, IDCL, IDMET, IDRS, IDM2
      ENTRY  DFDL, DFRX, SDBM, SWSF, DFCD, DFER, SSMF
      ENTRY  IDCL1, CBRF
      ENTRY  NDCL, DTH, DTP
      ENTRY  DRQ, DRQU, EDCL, DTXS2
      ENTRY  IDM, DCWBIT
* ENTRIES (FROM MDBG)
      ENTRY  SMIFIL, SMBA, SMDRN, FBWRD, SMFIL, FBADR, IXC
      ENTRY  BX0, BBP, BFP, BIN, BIC, BDN, BDC, BIP, BIA
DCWBIT DATA 40000000B
* DRQ MAP
* WORD 0: DISC ADDR.
* WORD 1: 0-7=FILE NO. 8-23=CORE ADDR.
* WORD 2: 0=R/W 1-5=FTIME 6-8=INT. ROUTINE 9-23=WORD COUNT
* WORD 2 NEGATIVE FOR WRITE.
DRQ   BSS NDRQ*3
DRQU  EQU *
*
*
*
* 'LAS', 'SAS' 10/18/65

```

```

*
* THIS ROUTINE IMPLEMENTS THE LOAD AND STORE TO SECONDARY MEMORY.
*
LAS    POPD    14600000B,1,1,0,1
LASA   BRM MPDSC; BRU LSM
*
SAS    POPD    14700000B,1,1,0,1
SASA   BRM MPDSC; BRU SSM
*
* 'IDM'  4/16/66
*
* THIS IS THE DRUM INTERRUPT ROUTINE FOR ALL DRUM I/O
*
NDCL   ZR0;* COUNT OF DRUM COMMANDS IN LIST
EDCL   ZR0;* CURRENT END OF LIST
      IF V2
IDCL   ZR0;* CURRENT INTERRUPT POINTER
IDCL1  ZR0;* CURRENT COMMAND EXECUTING POINTER.
IDMRET ZR0;* BRR/BRI RETURN SWITCH
IDRS   ZR0;* SAVE REGISTERS ONE TIME SWITCH.
IDWD31 BRM     INT33
IDWD33 EQU     IDWD31
      ENDF
RLDMSK DATA   NDISCS*20000B-1 REAL DISC ADDRESS MASK
DADMSK DATA   NDISCS*32*64*4-1 DISC ADDRESS MASK RIGHT SHIFTED 2
DRMTRY ZR0;* TRY-AGAIN COUNTDOWN
IDSA   ZR0;* SAVE (A)
IDSB   ZR0;* SAVE (B)
IDSX   ZR0;* SAVE (X)
*
INTN0P ZR0; BRR INTN0P
IDT    DATA INTN0P, IDR, IDP, IDF, IDB
*****INSERT NEW INTERRUPT ROUTINE HERE*****
IDM    ZR0    0
      DRT; BRU *-1; DET; BRU ID7; DCT; BRU ID7 (ERROR CHECK)
IDSW1  BRU    ID3          ALERT DISK OR BRU ID3
      P0T*   IDCL         DISK ADDRESS
      E0M*   10000B       ALERT CHNL
IDE2   BRU    *           I/O CONTROL E0M

```

```

PBT      IDCADD      CORE ADDRESS
IDE1     BRU      *      R/W BUC
BRM IDSAV; SKR NDCL; NOP 0      (SAVE REG., DECR. COMMAND COUNT)
LDA IDCL; ADD =3; SKE =DRQU; BRU **2; LDA =DRQ; XMA IDCL (NXT CMD
ID4      XMA IDCL1; SKG =-1; BRU ID5 (IDCL1 IS EXECUTING, SKIP TO POST PROC
CAX; LDA 1,2; LRSB 16; STA ID01      (FILE NUMBER)
CLA; LSH 5; XXA; SKR RMC,2; NOP 0 (BUMP PAGE LOCK)
CAX; LDA 2,2; LRSB 18; ETR =37B; CNA
CLA; LSH 3; COPY AX, XB; BRM* IDT,2 (XEQ INTRPT ROUTINE)
LDA =NDTRY=1; STA DRMTRY      (DRUM ERROR COUNTER)
ID5      LDA IDCL; SKN NDCL; BRU ID6; BRU ID2 (SETUP NXT CMND OR WINDUP)
ID6      STA IDCL; CAX; LDA 1,2; LRSB 14 (SETUP ROUTINE, GET CORE ADRS)
ETR =3; STA ID01      (SAVE TWO HIGH ORDER BITS)
LDA 2,2; LRSB 10; STB IDCADD; ETR =37B; ADD ID02 (GENERATE I/O)
SKN 2,2; ADD ID03; STA IDE2      (STORE EOM)
LDA ID01; CLB; LSH 5; ADM IDE2      (ADD TO EOM)
LDA IDSW1A; STA IDSW1      (SET SWITCH TO XEQ COMMANDS)
LDA ID05; SKN 2,2; ADD ID06; STA IDE1 (MAKE READ/WRITE EOM)
LDA IDCL1; SKG =-1; BRU IDSW1      (EXIT IF CMND ACTIVE)
ID0      SKN IDCL1; BRU ID1; LDA IDWD31; STA 31B
ID1      LDA IDSA; LDB IDSB; LDX IDSX; MIN IDRS (RESTORE REGS.)
SKN IDMR; SKR IDMR; BRI IDMR; BRR IDMR (INTRPT OR SUBR RETURN)
ID2      LDA IDSW1B; STA IDSW1; BRU ID0 (LIST EMPTY, DONT XEQ ANYMORE)
ID3      BRM IDSAV; LDA =-1; BRU ID4 (SETUP TO INDICATE NO CMND EXECUTING)
IDSAV    ZR0; SKN IDRS; SKR IDRS; BRR IDSAV (SAVE ONLY ONCE)
STA IDSA; STB IDSB; STX IDSX; BRR IDSAV
ID7      SKN IDCL1; BRU **2; BRU ID3; BRM IDER; DET; MIN DRMERR; DCT
MIN CHNERR; SKR DRMTRY; BRU **3; MIN DRUERR; BRU IDSW1
MIN NDCL; BRM IDSAV; LDA =-1; XMA IDCL1; BRU ID6
IDR      ZR0; BRM IDF      (CHECK DISK ERRORS)
LDA XN2; ADM FD,2; MIN ACTR; LDA FC,2; CBX
IF      C181
ADD =2
ELSF    1
ADD =1
ENDF
LDB =3777B; SKM 1,2; BRR IDR
LDA 1,2; LRSB 11; ETR =37B; STA ID01      (PAGE NUMBER)
LDA =7; LSH 11; CAX

```

DIR; EOM 21400B; PBT ID01 (SET RELABELLING)

IF C181

LDA -2,2; ADD NDDW,2; STA -1,2

ELSF 1

LDA -1,2; ADM Q,2

ENDF

EOM 21400B; PBT RRL3 (ADD 1ST BFR WRD TO CNT)

EIR; BRR IDR

IDP ZR0; BRM IDF; LDA =-DBB; ADM FD,2; BRR IDP (CK DK ERR,MRK BFR)

IDF ZR0; LDX ID01; SKN DRMTRY; BRR IDF (RETURN IF NOT UNRECOVRD ER)

LDA FD,2; MRG X4; STA FD,2; BRR IDF (MARK ERROR FLAG)

\* INT. ROUTINE FOR NON-FILE DISC I/O. BRS BE+1, BE+2.

IDB ZR0; LDX ID01; LDA X2; EOR TTN0,2; STA TTN0,2

LDA =200000B; SKN DRMTRY; LDA =100000B (NO ERRORS)

ADM TTN0,2; MIN ACTR; BRR IDB

IDM2 ZR0; SKN IDCL1; BRU IDM22

STA BLK31; LDA IDWD33; STA 33B; CLA; XMA BLK31; BRI IDM2

IDM22 DCT; BRU \*\*4; DET; BRU \*\*2; BRI IDM2

MIN IDMRET; BRM IDM; BRU IDM2+1

\* ROUTINE TO SAVE DISC ADDRESS OF ERROR

IDER ZR0; STA IDER3; LDA\* IDCL1; STA\* IDER1; MIN IDER1

LDA IDER2E; SKE IDER1; BRU \*\*3; LDA =IDER2

STA IDER1; LDA IDER3; BRR IDER

IDER1 ZR0 IDER2

IDER2 BSS 10

IDER2E ZR0 \*

IDER3 ZR0 0

ID01 ZR0 0

WORKING CELL

ID02 EOM 17200B

I/O CONTROL EOM

ID03 DATA -200B

READ DIFFERENCE

ID05 EOM 3666B

WRITE EOM

ID06 DATA -1040B

READ DIFFERENCE

IDSW1A EOM 10026B

ALERT DISK

IDSW1B BRU ID3

IDCADD ZR0 0

\*\*\*\*\*

\*

\*

\* 'IDTP', 'IDTH'



\*  
\* GENERALLY USEFUL DRUM ROUTINES

\*  
DTXS1 ZR0;\* COMMAND COUNT  
DTXS2 ZR0;\* LAST COMMAND LOC

\*  
\* RESET COMMAND COUNT

DTP ZR0; LDA #1; STA DTXS1; BRR DTP

\* INITIALIZE CORE ADDRESS

DTH ZR0; RSH 11; STB T; MUL #3; CBX  
LDA RRL2; LDB RRL1; LCY 6,2; ETR =77B  
LDB T; LCY 11; STA T; BRR DTH

\*  
DWI P0PD 14400000B,1,1,0,1

DWIA BRM MPDSC; BRU DIW

DW0 P0PD 14500000B,1,1,0,1

DW0A BRM MPDSC; BRU D0W

DBI P0PD 14200000B,1,1,0,1

DBIA BRM TRPB

DB0 P0PD 14300000B,1,1,0,1

DB0A BRM TRPB

\*  
\* DISC BRS'S

\* BRS 58

SSMF BRM MPDSC; LDA SS01; BRU SSMFA

\* BRS 59

CBRF BRM MPDSC; BRU CBRFA

\* BRS 66

DFDL SKN EXEC1; BRM TRAPB; BRM MPDSC; LDA SS01; BRU DFDLA

\* BRS 67

DFER SKN PQU,2; BRM TRPB; BRM MPDSC; LDA SS01; BRU DFERA

\* BRS 82

SWSF BRM MPDSC; LDA SS01; BRU SWSFA

\* BRS 87

DFRX SKN PQU,2; BRM TRPB BRM MPDSC; BRU DFRXA

\* BRS 113

DFCD BRM MPDSC; LDA SS01; BRU DFCD A

\* BRS BE+5  
SDBM SKN PQU,2; BRM TRPB; BRM MPDSC; LDA SS01; BRU SDBMA  
END

3MDBG IDENT 7/02/67  
\* VERSION 12 6/18/67

\*  
\* SYSTEM FLAGS, OPDS, PARAMETERS, AND MACROS  
\*

\* ASSEMBLY FLAGS

CRXF EQU =1 (NO CARD READER)  
PNXF EQU 1 (PAPER TAPE PUNCH)  
LPXF EQU =1 (NO LINE PRINTER)  
RELCHN EQU =1 (OLD DRUM CHANNEL)  
940M EQU =1 (=1 FOR BERKELEY, 1 FOR 940)  
ARMF EQU 1 (ARMING FEATURE)  
V1 EQU 1 (VERSION 1.85=1)  
V2 EQU 1 (VERSION 1.85=2)  
V3 EQU 1 (VERSION 1.85=3)  
V4 EQU 1 (VERSION 1.85=4)  
V5 EQU 1 (VERSION 1.85=5)  
V6 EQU 1 (VERSION 1.85=6)  
V7 EQU 1 (VERSION 1.85=7)  
V8 EQU 1 (RELABELED VERSION)  
FCB EQU 1 (FCB CHANGES)  
C181 EQU 1 (1.81 COMPATIBLE FILES)

\* OPDS

SBR5 OPD 17300000B,1,1 SYSTEM MODE BRS  
TSN OPD 00222000B,2 GO FROM NORMAL TO MONITOR MODE  
CKN OPD 00220100B,2 TURN ON THE CLOCK  
CKF OPD 00220200B,2 TURN OFF THE CLOCK  
LRR1 OPD 00220400B,2 LOAD RELABELLING REGISTER 1  
LRR2 OPD 00221000B,2 LOAD RELABELLING REGISTER 2  
LRR3 OPD 00221400B,2 LOAD RELABELLING REGISTER 3

IF V5

\* PRINTER COMMANDS

EPT MACRO  
DATA 4014060B END OF PAGE TEST  
ENDM  
PFT MACRO  
DATA 4011060B SKIP IF NO PRINTER ERROR

```

PRT      ENDM
        MACR0
        DATA      4012060B          SKIP IF PRINTER READY
        ENDM
SKIPT0   MACR0      D
        DATA      210460B+D(1)*1000B
        ENDM
SPACE    MACR0      D
        DATA      210660B+D(1)*1000B
        ENDM
PRINT    MACR0      D
        DATA      242060B          ALERT INTERLACE, 1 CHAR. PER WORD
        DATA      215200B          ARM I31, DISCONNECT WHEN
        P0T        D(1)            TRANSMISSION IS COMPLETE.
        ENDM
CAT      0PD        04014000B,2
*
* BREAKPOINT TEST
BPT      0PD        04020000B,2
        ENDF
*
* I/O DEVICE 0PD'S
TTYS     MACR0; DATA 20277777B; ENDM
TTYSK$   EQU       24077000B
E0D      0PD       00600000B,1,1
ALR      0PD       00610026B,2      ALERT RAD
RRF      0PD       00602226B,2      READ RAD
WRF      0PD       00602266B,2      WRITE RAD
RIN      0PD       00616200B,2      RAD I0SD
I0SDE    EQU       00617200B
I0RDE    EQU       00614000B
RSR      MACR0; SKS* 10026B; ENDM
RSE      MACR0; SKS* 11026B; ENDM
CETE     MACR0; SKS* 11000B; ENDM
I0SDW    EQU       214200B
I0RDW    EQU       214000B
ALD      0PD       00210026B,2      ALERT DISC
DSR      0PD       00202626B,2      DISC READ
DRT      0PD       04010026B,2      DISC READY TEST

```

DET 0PD 04011026B,2  
DCT 0PD 04011000B,2

DISC ERROR TEST  
DISC CHANNEL ERROR TEST

\*  
\*

\* PARAMETERS

BE EQU 123  
NP0P EQU 44B

LAST BERKELEY BRS.  
NUMBER OF SYSP0PS IN USE.

\* W BUFFER DEVICE PARAMETERS

RTCNT EQU 64  
PNCNT EQU 40

PAPER TAPE READER BUFFER LENGTH  
PAPER TAPE PUNCH BUFFER LENGTH  
CARD READER BUFFER LENGTH

CRCNT EQU 40  
CRCNTB EQU 40

CARD PUNCH BUFFER LENGTH

CPCNT EQU 40  
CPCNTB EQU 40

NUMBER OF MAG TAPE UNITS

NTAPE EQU 2  
NLINK EQU 0

LENGTH OF MAG TAPE BUFFER  
LINE PRINTER BUFFER LENGTH

TCNT EQU 199  
LPCNT EQU 132

RTWT EQU RTCNT\*40/3  
PNWT EQU PNCNT\*400/6

PAPER TAPE READ TIME  
PAPER TAPE PUNCH TIME

CRWT EQU 300  
CPWT EQU 150

CARD READ TIME  
CARD PUNCH TIME

TXWT EQU 20\*TCNT/10  
LPWT EQU 133

MAG TAPE TIME  
LINE PRINT TIME

NTRTRY EQU 10  
NTWTRY EQU 3

NUMBER OF REREADS  
NUMBER OF REWRITES

\* FILE PARAMETERS

NFILE EQU 40  
MBUGX EQU 34000000B

NUMBER OF FILES  
FBWRD FOR EXEC BLOCK  
PROTECTED FILE BUSY BIT

DBB EQU 00400000B

\* TTY PARAMETERS  
NTTY EQU 32 NUMBER OF TTYS  
NTTB EQU NTTY+NLINK

TOTAL TTY BUFFERS

NLTTC EQU 0  
NTTYC EQU 70

NUMBER OF CHARS IN TTY BUFFER  
TTY EARLY WARNING (2 SEC)

TTYEWM EQU 20  
AMB EQU 40000B

ACCEPT MESSAGE BIT  
ACCEPT INPUT BIT

AIB EQU 100000B  
APB EQU 10000000B

ACCEPT PRINTER LINK BIT

```

AKB      EQU      200000B      ACCEPT KEYBOARD LINK BIT
8RB      EQU      4000000B     8-LEVEL INPUT BIT
8PB      EQU      2000000B     8-LEVEL OUTPUT BIT
ILB      EQU      1000000B     INPUT LINK BIT
OLB      EQU      400000B      OUTPUT LINK BIT
* PAC TABLE PARAMETERS
NSQU     EQU 12  NUMBER OF CLOCK CYCLES IN SHORT QUANTUM.
NFQU     EQU      36          FULL QUANTUM SIZE
NPAC     EQU      144         NUMBER OF PACT SLOTS
NPPAR    EQU 10  LENGTH OF PACT ENTRY
* JOB AND MEMORY PARAMETERS
NJOB1    EQU      32          NUMBER OF JOBS WITHOUT P.U.
NJOB     EQU      NJOB1+1     NUMBER OF JOBS
UMSZ     EQU      15          INITIAL MACHINE SIZE
NMEM     EQU 32  NUMBER OF PAGES
NSMEM    EQU 7   NUMBER OF PAGES USED BY SYSTEM
NCMEM    EQU      60B        COMMON PART OF USER MACHINE
NSMT     EQU      100B       SIZE OF SMT
NUMEM    EQU 100B*NCMEM      NUMBER OF PRIVATE USER PAGES
NPUQ     EQU 16  NUMBER OF PUCT ENTRIES
* RAD AND SWAPPING PARAMETERS
NRDQ     EQU      20          MUST BE GT USER'S PAGES*2
NRTRY    EQU      1          NO. OF READ TRIES FOR RAD.
NRAD     EQU      4          NO. OF RADS
NSEC     EQU      2*NRAD
L2NSEC   EQU      1
NSBND    EQU      18         NUMBER OF 16K BANDS RESERVED FOR SWAPPING
NSSP     EQU      4000B      LBC. OF 1ST SWAPPING AREA. MUST BE A
*                               MULTIPLE OF 1000B.
NSAM     EQU 16  SIZE OF SWAPPER ASSOCIATIVE MEMORY
* DISC PARAMETERS
NDTRY    EQU      4          NUMBER OF READ TRIES FOR DISC.
NDRQ     EQU      30         NO. OF JOBS IN DISC QUEUE.
NDISCS   EQU      32
          IF      NDISCS=32
NPBS     EQU      20; ELSF 1; NPBS EQU 40; ENDF
MAXP     EQU NPBS/2*200B+31*200B
MINP     EQU *NPBS/2*200B+32*200B
TABLEN   EQU NPBS/2*2*NDISCS*32+23

```

TABLLEN EQU TABLLEN/24

\* BUFFER PARAMETERS

NBUFX EQU 3 NUMBER OF BUFFERS IN THE EXEC BLOCK  
NBUF EQU NBUFX TOTAL NUMBER OF DISC BUFFERS.  
NDDW EQU 255 LENGTH OF DATA BLOCK  
BIN EQU NDDW+2 INDEX BLOCK NUMBER  
BIC EQU NDDW+3 INDEX CHANGED FLAG  
BDN EQU NDDW+4 DATA BLOCK NUMBER  
BDC EQU NDDW+5 CHANGED DATA FLAG  
BIP EQU NDDW+6 INDEX BLOCK POINTER  
BIA EQU NDDW+7 INDEX BLOCK DRUM ADDRESS  
NDXW EQU 124 LENGTH OF INDEX BLOCK

IF C181

NDXWC EQU 78 ~~MAX NUMBER OF DATA BLOCKS PER FILE~~

ELSF 1;NDXWC EQU NDXW; ENDF

NDXWCR EQU NDXWC+1

NDXWR EQU 128 NUMBER OF WORDS TO READ/WRITE

NDBW EQU NDDW+8+NDXWR LENGTH OF DISC BUFFER

NDBS EQU NBUF\*NDBW SIZE OF BUFFER AREA

BX0 EQU NDBW-NDXWR INDEX BLOCK ORIGIN REL TO BUFF

BBP EQU BX0+NDXW-1 BACKWARD CHAIN WORD

BFP EQU BX0+NDXW+2 FORWARD CHAIN WORD

IXC EQU BX0+NDXW INDEX BLOCK CHECK WORD

\* TS BLOCK MAP

DBTOP EQU 37777B-NDBS-5-1-17-1 1ST WORD AFTER PRSYMS

SM0FIL EQU DBTOP SECONDARY MEMORY OUTPUT FILE

SMIFIL EQU DBTOP+1 SECONDARY MEMORY INPUT FILE

SMBA EQU DBTOP+2 SECONDARY MEMORY BUFFER ADDRESS

SMDRN EQU DBTOP+3 SECONDARY MEMORY BDN ADDRESS

FBWRD EQU DBTOP+4 BUFFER AVAILABILTY BIT WORD

RMAP EQU DBTOP+5 RAD BIT MAP FOR FILES AND SWAPPING

PB EQU DBTOP+6

PX EQU PB+8

PPB EQU PX+8 POINTER TO PB CHAIN

NFORK EQU PPB+1 NUMBER OF FORKS COUNTER

FBADR EQU DBTOP+5+1+17+1 FIRST BUFFER ADDRESS

\* MONITOR AND EXEC LOCATIONS

RAW EQU 100B RAD ADDRESS OF W

CAW EQU 44000B CORE ADDRESS OF W

```

DAW      EQU      0          DISC ADDRESS OF W
RADSC    EQU      0          RAD ADDRESS OF DISC
CADSC    EQU      40000B     CORE ADDRESS OF DISC
DADSC    EQU      300B      DISC ADDRESS OF DISC
CASET    EQU      50000B     CORE ADDRESS OF SET
DASET    EQU      340B      DISC ADDRESS OF SET
DAEXEC   EQU      100B      DISC ADDRESS OF EXEC
RAEXEC   EQU      600B      RAD ADDRESS OF EXEC
* MACRO$
A        EQU      1
B        EQU      2
AB       EQU      4
BA       EQU      10B
BX       EQU      20B
XB       EQU      40B
E        EQU      100B
XA       EQU      200B
AX       EQU      400B
N        EQU      1000B
X        EQU      20000000B
COPY     MACRO D
K        NARG
L        EQU      0
M        EQU      1
        RPT      K
L        EQU      L+D(M)
M        EQU      M+1
        ENDR
        DATA   4600000B+L
        ENDM
        IF     ARMF
ARMJ     MACRO D; AIR; PBT D(1); ENDM
        ELSE  1
ARMJ     MACRO; ENDM
        ENDF
ENTRY   MACRO L; ENT CNT NARG; RPT ENT CNT; L(ENT CNT) EXT
ENT CNT EQU ENT CNT-1; ENDR; ENDM
SET INT MACRO A; LDA =A(1); STA BLK31; ENDM
TDT     MACRO L; L(1)*W EQU *; RPT NTAPE; L(2) L(3)*B+*=L(1)*W; ENDR; ENDM

```



```

IF      V1
RMFF   MACR0; ENDM
SMFF   MACR0; ENDM
READ   MACR0 D,G,1;G(1) RSR; BRU *=1; ALR; P0T =D(3)/100B
      E0D* 10000B; DATA I0SDE+D(1)/2000B(AND)37B+D(2)/40000B(AND)3*40B
      P0T =D(1)(AND)1777B*40000B+D(2)(AND)37777B; RRF; RSR; BRU *=1
      RSE; BRU G(1); CETE; BRU G(1); ENDM
      ELSF 1
SMFF   MACR0 D; DATA 234006B+D(1)*40B; ENDM
RMFF   MACR0 D; DATA 230006B+D(1)*40B; ENDM
      ENDF
ECHR   MACR0 N;ECHRWD EQU ECHRWD*400B+N(1)*B;NECHR EQU NECHR+1
      IF NECHR=2; DATA ECHRWD;ECHRWD EQU 0;NECHR EQU 0; ENDF; ENDM
ECHB   MACR0 N;ECHVB EQU N(1)B; RPT N(2); ECHR ECHV
ECHVB  EQU ECHVB+N(3); ENDR; ENDM
TRP    MACR0 L;ENTCNT NARG; RPT ENTCNT;L(ENTCNT) EQU TRAP
      FRGT L(ENTCNT);ENTCNT EQU ENTCNT-1; ENDR; ENDM
CACR   MACR0 D; D(2)
      IF D(1); BRU PACACT; BRU PEST
      ELSF 1; BRU PEST; BRU PACACT; ENDF; ENDM
LBL    MACR0 D;1LBL EQU D(2); RPT D(2); LDA D(1)+1LBL-1; LRSH 6
1LBL   EQU 1LBL-1; ENDR; ENDM
*
*
*
* EXEC ENTRY POINTS
EXECI  EQU      10000B
EXECP  EQU      10001B
OFFINT EQU      10002B
*
*
*
DB     MACR0 D;ENTCNT EQU D(1)*B*200B+D(2)*100B
      **400000000B+D(4)(AND)1*40B+D(4)(AND)2*10000000B
      RPT D(3); DATA ENTCNT;ENTCNT EQU ENTCNT+100B; ENDR; ENDM
F0RGT  MACR0 D;ENTCNT NARG; RPT ENTCNT; FRGT D(ENTCNT)

```

ENTCNT EQU ENTCNT-1; ENDR; ENDM

FØRGT CRXF,AIB,8PB,8RB,AMB,APB,AKB  
FØRGT MØUFØ,PNXF,LPXF  
FØRGT RTCNT,PNCNT,TCNT,LPCNT  
FØRGT RTWT,PNWT,TXWT  
FØRGT NTAPE,NLINK,NØUFØ,NØUF  
FØRGT NTRTRY,NTWTRY,NDTRY  
FØRGT NDDW,NDXW,NDBW  
FØRGT NFILE,UMSZ,NTTYC,TTYEWM  
FØRGT NPAC,NPPAR,NJØB,NJØB1,NFØU,NSØU  
FØRGT NPØP,NMEM,NSMEM,NØMEM,NØMEM  
FØRGT NØUØ,NSEC,L2NSEC,NDRØ,NSØND,NSAM,NSMT  
FØRGT ØBB,ENTCNT  
FØRGT SMIFIL,SMBA,SMØRN,FBWRD,SMØFIL  
FØRGT BØØ,BØP,BØP,BIN,BIC,BØN,BØC,BØP,BØA  
FREEZE  
END

3PAC IDENT 7/02/67

\*  
\* SCHEDULER AND SWAPPER

\*  
PACDMB ZR0 \*

PRMSK DATA 7777B

PLMSK DATA 77770000B

ADMSK DATA 37777B

IF V1

NADMSK DATA (N0T)37777B

ENDF

SECMSK DATA NSEC=1 = 7

SGCI 0PD 16500000B,1,1

S0ST 0PD 15100000B,1,1

\* ENTRY POINTS

IF V1

ENTRY CLINTC, GCRL, GCRL1, GCRL2, GCRL3, DISCA

ENTRY RSYBT1, MPPACT, RLTS, ACTPU, PUPACP

ENTRY ARD, AWD, RDSYB, WDSYB, RST, NADMSK, PUDEAD, PWN I

ENTRY PUTIM, NP0PX, NXP0P

ENTRY RSYB2, MPDSC, MPWB

ENDF

ENTRY RMT, RMC, RMA, PLMSK, PRMSK, ADMSK, SECMSK, SRT, SRTE

ENTRY PACDMB, P0PX, XP0P, PACQE, P0PDMS, SETSET, GDC

ENTRY PACG0, PEST, P0PINT, P0PST, PUG0, NPUG0, QGEDMS, PACG01

ENTRY FPULST, PUCT, PUEPTR, PUBPTR, PUPAC

ENTRY SWAP, UPRL, PTRL, LABEL, CHRL, 0MW, RSAM, 0MR, SWMPTR

ENTRY RRL1, RRL2, RRL3, BRRL3, QUTAB, TIME, TTIME, PUCTR

ENTRY SACNT, SAM1, SAM2, SAM3, SAR1, SAR2, SAR3

ENTRY RTEX, RREAL, WREAL, CQ0, SQ0, EXDMS, RSYB, WSYB, DMS

ENTRY EPU, FPLST, QSCH, QPUT, IIR, SIR, PGET

ENTRY CLINT, CLOCK3, PWF I, TRAPT, M0NCR, CRASH

\* ENTRIES FROM MDBG

ENTRY RMAP

\*

```

*
*   SCHEDULER
*
* BRS 72   EXEC DISMISS
* INPUT:  SS03=QUEUE NUMBER.  X=PACPTR.  B=PTEST.
EXDMS   SKN PQU,2; BRM TRAPB; LDX SS03
        RCH 220B (CXA,CBX); MUL =3; LSH 23; ADD =Q0
        RCH 440B (CXB,CAX); MIN 0; BRU P0PDMS
* DISMISS UNTIL INTERRUPT
* BRS 109
DMS     LDB =700005B; BRU P0PST
*BRS 45 (SIM QUANTUM 0FL0)
SQ0     MIN 0; LDB =-1; STB TTIME; BRU PACQE
* RELABEL PAC TABLES BEFORE EXIT
NXP0P   STA SS01; BRM MPPACT; LDA SS01; BRU XP0P
NP0PX   BRM MPPACT
*SYSP0P EXIT TO RESTORE CENTRAL REGISTERS
P0PX    LDA SS01; LDB SS02; LDX SS03
*SYSP0P EXIT IF CENTRAL REG 0K -- ABOLISH WHEN UMT PUT IN
XP0P    SKN TIME; BRR 0; SKN TTIME; SKN ACTR; SKN 0; BRR 0
*QUANTUM 0VERFLOW 0CCURED
        STA SS01; STB SS02; STX SS03; MIN 0
PACQE   LDB PACDMB
QGEDMS  LDX =QGE; SKN TTIME; LDX =QSQ
*P0P DISMISS ENTRY P0INT
P0PDMS  STB PACLVL; LDA PACPTR; BRM QPUT (PUT PAC 0N QUEUE)
        LDB PACLVL (PICK UP DISMISS C0ND)
P0PST   LDX PACPTR; STB PTEST,2
P0PINT  LDA 0; ETR =5003777B; XMA PL,2; ETR =7700000B
        ADM PL,2 (SAVE START L0C)
*SAVE CENTRAL REGISTERS
        LDA SS01; STA PA,2; SKN XPB; BRU PAC2C
        LDX XPB; LDA SS02; STA PB,2
        LDA SS03; STA PX,2; LDX PACPTR
*SET UP P0PER QUANTUM REMAINING
PAC2C   LDA PQU,2; LRSB 12; ETR =7; RCH 402B (CAX+CLB)
        LDA TTIME; SKG =-1; LDA QUTAB,2; LDX PACPTR; LSH 15
        XMA PQU,2; ETR =6007777B; ADM PQU,2

```

```

PACG01 LDA #SETTB; STA TJOB (SET ACCOUNT TO SYSTEM)
      SKN ACTPU; BRU PACPUC
* START SCHEDULER
PACG0  LDX #QTIQ; STX CL0CK3 (SAVE NEG NO FOR SDS CLOCK); RMFF #0B
      LDA RRL3; ETR #7700B; SKE #600B; BRM M0NCR
PACSCN STX PACLVL (SAVE QUEUE); STX PACPTR (SET PACPTR); BRU PEST+1
* MOVE PU FROM QGE TO QTI IF NECESSARY
PACPUC LDX #QGEQ
PACPUB STX PPREV; LDA PNEXT,2; CAX; SKE PUPACP; BRU PACPUA
      LDA PNEXT,2; LDX PPREV; LDB #QGEQ; BRM QGET
      LDA PUPACP; LDX #QTI; BRM QPUT; BRU PACG0
PACPUA SKG #-1; BRU PACPUB; BRU PACG0
*SCAN PACT
PEST  LDX PACPTR; SKN PNEXT,2; BRU PACNXQ (NEW QUEUE); STX PPREV
      LDX PNEXT,2; STX PACPTR
      LDA PTEST,2; LRSB 15; CAX
      LDA CACLST,2; STA T; LDX PACPTR; LDA* PTEST,2; BRU* T
*ACTIVATE PROGRAM
PACACT LDX PACPTR; BRM PGET; BRU PEST (READ ERROR)
      LDA #700001B; XMA PTEST,2; STA PPTST (SAVE PTEST FOR INT SIM)
      LDA PNEXT,2; LDB PACLVL; LDX PPREV; BRM QGET (GET PR0G OFF QUEUE)
      LDX PACPTR; LDA PQU,2; LRSB 15; ETR #77B; STA TTIME
*SET UP TIME, TTIME, AND ACTR
      LDA #NSQU; STA TIME; LDA #1; STA UMTF
*CHECK FOR INTERRUPT AND SET UP START L0C (0)
      LDA T; LRSB 15; ADD #ACTLST; XMA T; BRU* T
PACSRT LDA PL,2; STA 0
PAC0VF R0V; LCY 1; LSH 1 (SET UP 0VERFLOW)
      CLB; STB EXEC1; LDA PTAB,2; LDB #-1; SKA X2; STB EXEC1
      LDA PA,2; COPY B,X; SKN XPB; BRU PAC2B
      LDX XPB; LDB PB,2; LDX PX,2
PAC2B SMFF #0B; BRU* 0
PACNXQ LDA PNEXT,2; SUB #PNEXT; CAX; SKE #QSQQ; BRU *+3
      LDB #-1; STB ACTR; \BRU PACSCN
*CAUSE A PR0G INTERRUPT
PACINT LDA PPTST; ETR #37B; LDB PL,2
      SKN PL,2; LDB SBRST; CAX; LDX 200B,6; STB* 0,6
      CXA; ETR ADMSK; ADD #40000001B; STA 0; LDX PACPTR; BRU PAC0VF
*ACTIVATION TEST ROUTINES

```

CACLST 0 CAC0; 0 CAC1; 0 CAC2; 0 CAC3  
0 CAC4; 1 PACACT; 0 CAC6; 0 PEST  
0 PEST; 0 CAC9; 0 CAC10  
\* SPECIAL ACTIVATION ROUTINES  
ACTLST BRU PACSRT; BRU PACINT

CAC0 CACR 0, (SKG =0)  
CAC1 CACR 1, (SKG =0)  
CAC2 CACR 0, (SKG =-1)  
CAC3 CACR 1, (SKG TTYEMG)  
CAC6 CACR 1, (SKG REAL)  
CAC4 BRU\* PTEST,2  
CAC9 CACR 0, (SKA X2)  
CAC10 CACR 1, (SKG =-1)

\*  
\*  
PACLVL ZR0  
PPREV ZR0  
PPTTEST ZR0

FPLST ZR0

\*  
\*

\* PHANTOM USER

\*  
PUDMS CLX; CLA; STA XPB; LDA =-1; STA ACTPU; EIR  
LDA REAL; LDB PUAT; SBRS 72

\* MAG TAPE CLEANUP. CHECKS EVERY 3 SECONDS.

LDA BLK31; SKG =0; BRU PUG03  
LDA =-1; XMA\* 31B; SKE =-1; BRU PUG0  
SBRS 114 (MTDI); BRU PUG02

PUG03 STA\* 31B

\* P.U. SCHEDULER

PUG0 LDA PUCPTR; DIR; SKG =0; BRU PUDMS; EIR; LDX =PUBPTR; STA PUCTR1  
PUSCN STX PUCPTR; LDX 0,2; LDA 2,2; RSH 12; STA PUPAC

```

        CLA; LCY 12; STA FILE; LDA 1,2; STA PUTST
        LRSR 15; ETR =77B; AXC; BRU* PUCLST,2
* SET PU RELABELLING BACK
SPURL  ZR0; BRM MPPACT; CLA; STA J0B; LDX PACPTR; BRM CHRL
        BRU *=2; LDA RRL3; ETR =77B; STA RLTS; BRR SPURL
NPUG0  BRM SPURL; BRU PUG0
NPUNXT BRM SPURL
* CAN'T PROCESS THIS ENTRY
PUNXT  LDX* PUCPTR; CXA; SKE PUEPTR; BRU PUSCN
        LDA PUCTR1; DIR; SKE PUCTR; BRU PUG0; LDX =3; BRU PUDMS+1
* SPECIAL REACTIVATION TEST
PUAT   + **1; LDA PA,2; ADD =180; SKG REAL; BRU PACACT
        LDA PUCTR; SKG =0; BRU PEST; LDA =PUG0; STA PL,2; BRU PACACT
* PHANTOM USER ACTIVATION TESTS
PUCLST DATA PUDI0,PURBT,PUTRTW,PUBRTW,PUCRTW,PUFK
        DATA PUL0,PUTIM,PULPTW

```

```

        IF          V5
* TEST PROGRAM INTERRUPT TIME OUT
PUTIM  LDX* PUCPTR; LDA REAL; SKG 2,2; BRU PUNXT
        LDA 3,2; CAB; ETR =77B; CAX; LDA WERIS,2; SKG =-1
        BRU PUACTION; CBA
        RSH 12; MRG PLMSK; STA PUTIM1; CAX
        LDA PIM,2; SKA =3; BRU **2; BRM MONCR
        SUB =1; STA PIM,2
        LSH 6; ETR =37B (INT. NO.); COPY AX,B
        LDA =4000000B; RSH 0,2; LDX PUTIM1; BRM IIR
        BRU PUACTION (NO INT.); BRU PUACTION (INT.)
PUTIM1 ZR0 0
* REMOVE EXTRA PU ENTRIES
PUL0  BRU PUACTION
        ELSF      1
* TEST DATA-SET TIME OUT
PUDHU LDX FILE; LDA REAL; SKG TTYTIM,2; BRU PUNXT; BRU PUACTION
        ENDF
* TEST IF TAPE READY
        IF          FCB
PUTRTW LDX FILE; LDA FA,2; ETR =77B; COPY AX,B
        ELSF      1

```

```

PUTRTW LDX FILE; LDA FC,2; RSH 15; ETR =77B; CAX
      ENDF
      BRM MPWB
      EXU TRTW,2; BRU **2; BRU NPUNXT; BRM MPPACT
* TEST W-BUFFER
PUBRTW CLA; SKE BLK31; BRU NPUNXT
      IF      FCB
      LDX FILE; LDA FC,2; LRSB 15; ETR =77B; COPY AX,B; STX JOB
      ELSF    1
      LDX FILE; LDA FA,2; LRSB 15; COPY AX,B; STX JOB
      ENDF
      LDA PMTP,2; ADD =20000000B-NCMEM; STA PMTJOB; CLA
      LDX RL3,2; BRM SWAP; BRU NPUNXT; BRM LABEL
      LDA RRL3; ETR =77B; STA RLTS; BRM MPWB; BRU PUACTION1
PUACTION LDX FILE; LDA =57777777B; ETR FD,2; STA FD,2
      LDB FD,2; STB DEV
      LDA FC,2; ETR ADMSK; STA BUFF
      LDB PUTST; BRM ED; BRU NPUGO; BRM M0NCR; BRM M0NCR
* TEST CARD READER READY
PUCRTW BRM MPWB; EXU CFTW; BRU PUACTION; EXU CRTW; BRM M0NCR
      BRM MPPACT; BRU PUBRTW
PULPTW PRT; BRU NPUNXT; CLA; SKE BLK31; BRU NPUNXT
      BRM MPPACT; BRU PUBRTW
* TEST IF DRUM FILE READY
PUDI0  LDX FILE; LDA FD,2; SKA =DBB; BRU PUNXT; BRU PUACTION
* TEST IF RUBOUT APPLICABLE
PURBT  LDA PUPAC; LRSB 3; STA TIPIX; LDX FILE; SKG =1; BRU PURBT2
      SKN TTYASG,2; BRM M0NCR; LDX TTYASG,2
      LDA =PURBTA; BRM SCFK; STX PUPAC
      BRU PUACTION
PURBTA ZR0; LDA PIM,2; SKA X1; BRU PURBT1; BRR PURBTA - See
PURBT1 MRG X2; STA PIM,2; BRU PUNXT
      IF      V5
PURBT2 SKN TTYASG,2; BRU PUACTION1; BRM M0NCR
      ENDF
* TEST IF P.U. FORK POSSIBLE
PUFK   LDA FPLST; SKG =0; BRU PUNXT
      LDA PUTST; STA PUFT+1; LDA FILE; STA PUFT+3
      LDX PUPAC; LDA PUFR,2; STA PUFT; BRU PUACTION1

```



```

PUFTA ZR0 PUFT,4
* ACTIVATE P.U. REQUEST ROUTINE
PUACT LDX PUPAC; BRM PGET; BRU PUNXT (READ N0 G00D)
PUACT1 LDX PUCPTR; DIR; LDA FPULST; XMA* 0,2
      XMA 0,2; STA FPULST; SKE PUEPTR; BRU **2; STX PUEPTR
      SKR PUCTR; MIN PULIM; EIR
      LDA PUTST; LRSR 15; ETR =77B; CAX; BRU* PUCSET,2
* PHANTOM USER REQUEST PROCESSING PREPARATION
PUCSET ZR0 PUDI0S; ZR0* PUTST; DATA PUACTION,PUACTION,PUACTION,PUFP
      DATA PUNXT,PUNXT,PUACTION
* CONTINUE DRUM BR5
PUDI0S LDX FILE; LDA XN2; ADM FD,2; BRM MPDSC; LDA FC,2
      ETR ADMSK; STA BUFF; CAX; BRR PUTST
* RUN P.U. FORK
PUFP LDA PUFTA; SBR5 9; BRU PUNXT
PUFR DATA PUMSG
* PRINT MESSAGE
PUMSG ETR ADMSK; MUL =3; LSH 23; STA PUFL; SKR PUFL
PUMC SGC1 PUFL; HLT; SKE =17B; BRU **2; SBR5 10; LDB PUFL
      SKE =4; BRU PUMC1; LDA =155B; S0ST SS03; LDA =152B
PUMC1 S0ST SS03; STB PUFL; BRU PUMC

* MAKE P. U. ENTRY
* INPUT: A=1,2. B=2,2. X=3,2. LOW 6 BITS OF X=TTY N0.
* OUTPUT: X=PU PTR.
EPU ZR0; XMA* FPULST; SKR PULIM; BRU **2; BRM M0NCR
      MIN PUCTR; XMA FPULST
      STX PUXSV; CAX; XMA* PUEPTR
      STX PUEPTR; XMA 0,2; STA 1,2; STB 2,2
      XMA PUXSV; STA 3,2; XMA PUXSV; EIR; BRR EPU
PUXSV ZR0 0
*
*
      IF V5
PUCT BSS NPUQ*4
$EPUCT3 EQU PUCT+NPUQ*4+4
$EPUCT EQU PUCT+NPUQ*4
$EPUCM3 EQU PUCT+NPUQ*4+4
      ELSF 1

```

```

PUCT BSS NPUQ*3
$EPUCT3 EQU PUCT+NPUQ*3+3
$EPUCT EQU PUCT+NPUQ*3
$EPUCTM3 EQU PUCT+NPUQ*3+3
      ENDF
FPULST ZR0 0 1ST FREE ENTRY
PUBPTR ZR0 0 PTR. TO 1ST ACTIVE ENTRY. LAST ENTRY POINTS TO PUBPTR
PUCTR ZR0 0 NO. OF ENTRIES.
PUCTR1 ZR0 0 COUNTER DURING PU PROCESSING.
PUCPTR ZR0 0 IND. PTR. TO ACTIVE ENTRY DURING PU PROCESSING.
PUEPTR ZR0 0 LAST ACTIVE ENTRY.
PUPAC ZR0 0 PACPTR OF ENTRY BEING PROCESSED BY PU.
PUDEAD 6 0 NULL PU ENTRY
PUFL ZR0; DATA 137777B
PUFT BSS 7
PUTST ZR0
PULIM ZR0 NPUQ COUNTS NO. OF PU ENTRIES.
PUPACP ZR0 0 PACPTR FOR PU
ACTPU DATA =1 ACTIVATE PU SWITCH
*
*
```

\* GET USER TO CORE

```

PGET ZR0; STX PUPAC; CLA; STA XPB; SKN PUPAC; BRU PGET1
      LDA PTAB,2; LRSB 15; ETR =77B; STA J0B
      CAX; ADD =ETT0; STA TJ0B (SET UP TIME CHARGING)
      LDA PMTP,2; ADD =20000000B+NCMEM; STA PMTJ0B (SET UP PMT PTR)
      LDA TTN0,2; ETR =77B; STA UTTY (SET UP USER TELETYPE)
      LDX PUPAC; BRM CHRL; BRR PGET
      LDA RRL3; ETR =77B; STA RLTS; LDX PUPAC
      LDA PTAB,2; SKA X1; BRU **2; BRU PGET1 (NO TS BLOCK)
      LDA PIM,2; ETR =70B; LRSB 3; MRG X4; STA XPB
PGET1 MIN PGET; BRR PGET
```

```

RMT BSS NMEM REAL MEMORY ASSIGNMENT TABLE
$ERMTEQU RMT+NMEM
RMC BSS NMEM REAL MEMORY LOCKUP TABLE
```

```

$ERMC EQU RMC+NMEM;$ESRMC EQU RMC+NSMEM
RMA BSS NMEM REAL MEMORY AGE TABLE
$ERMA EQU RMA+NMEM

```

- \* SWAPPER- CALLED WITH RELABELLING IN A,B,X
- \* NO SKIP EXIT- READ ERROR OR INSUFFICIENT MEMORY
- \* SKIP EXIT- REAL RELABELLING IN A,B,X
- \* THE SWAPPER FIRST UNPACKS THE RELABELLING REGISTERS,
- \* THEN DECODES THE BYTES PRODUCING READ COMMANDS
- \* OR INCREMENTING THE LOCK. IF THERE ARE NO READ COMMANDS, THEN IT
- \* COMPUTES THE REAL RELABELLING, UNLOCKS ALL PAGES REFERENCED,
- \* AND EXITS SKIPPING. OTHERWISE IT BUILDS A LIST OF
- \* WRITE COMMANDS. THE SWAPPER THEN PUTS OUT DRUM COMMANDS.
- \* IT WILL ASSIGN A BLOCK TO A READ, AND READ IN A PAGE IF
- \* THERE IS AN OPTIMAL READ COMMAND. OTHERWISE IT WRITES OUT THE
- \* OPTIMAL FREE PAGE, AND MAKES THE BLOCK FREE FOR READING.
- \* WHEN ALL COMMANDS HAVE BEEN COMPUTED, THE DRUM IS
- \* STARTED, AND THE REAL RELABELLING INFORMATION IS COMPUTED. AFTER THE
- \* SWAP IS COMPLETE, THE SWAPPER EXITS SKIPPING.

```

SWAP ZR0; MIN SWAP; STA SWR1; STB SWR2; STX SWR3
      LDB J00; LSH 36; COPY AB,XB; STB SWR4
      LDA SWR1; LDB SWR2; LDX SWR3; SKN SACNT; BRU SWP0
      LDB SWR4; LDX SACNT
SWP22 SKE SAM1,2; BRU SWP24; LDA SWR2; SKE SAM2,2; BRU SWP26
      CBA; SKE SAM3,2; BRU SWP26
      LDA SAR1,2; LDB SAR2,2; LDX SAR3,2; MIN SWAP; BRR SWAP
SWP26 LDA SWR1
SWP24 BRX SWP22; LDB SWR2; LDX SWR3
SWP0 BRM DCRL; SKN DCRT3; BRU SWP1
* COMPUTE AND RETURN REAL RELABELLING IN A,B,X
SWP38 BRM PTRL; LDA SACNT; SKG =NSAM; BRU SWP37; SUB =1; STA SACNT
SWP39 STA SAPTR; XXA; ETR PRMSK; STA SAR3,2
      STB SAR2,2; LDA SWR4; STA SAM3,2
      LDA SWR1; STA SAM1,2; LDA SWR2; STA SAM2,2
      LDA SWD1; STA SAR1,2; LDX SAR3,2; MIN SWAP; BRR SWAP

```

\*

```

SWP37 LDA SAPTR; SKG =NSAM; CLA; SUB =1; BRU SWP39
SWP1 LDA =SWT5; STA SWT6; BRM RSAM
* FINISH DECODING USER RELABELLING
LDX =NSEC; STX SPT+NSEC,2; BRX *=1; LDX =-10
* LOCK PAGE OR SAVE DRUM COMMAND
SWP3 LDA SRTE,2; SKA X4; BRU SWP4; ETR =37B; COPY XB,AX
MIN RMC,2 (LOCK PAGE); LDA RMA,2; MRG X2; STA RMA,2
CBX; BRX SWP3; BRU SWP2
* ON DRUM, CHECK FOR DUPLICATE READ
SWP4 STX SWT13; LDX SWT6; STA SWT14=SWT5,2; CAX
LDA 0,2; ETR =177740B; LDX =SWT5
SWP40 XXA; SKE SWT6; BRU SWP41; STX* SWT6 (NOT DUP)
CXA; RSH 6; CBX; RSH 8; CXB; LSH 1; ETR SECMSK; CAX
LDA SWT6; ADD =SAT=SWT5; XMA SPT,2; STA* SPT,2
SWP41 XXA; EAX 1,2; SKE =1,2; BRU SWP40
LDX SWT13; BRX SWP3 (DUP)
* CHECK FOR SUFFICIENT MEMORY AVAILABLE
SWP2 LDA SWT6; STA SWT7; SUB =SWT5; STA SWT12; CNA
LDB X4; LDX =NSMEM-NMEM
SKB RMC+NMEM,2; ADD =1; BRX *=2; SKG =-1; BRU SWP5
* SET UP BLOCKS FOR WRITE, RELEASE READ-ONLY BLOCKS
SWP6 SKR SWT12; BRU *=2; BRU SWP12; LDA XX; LDX =NSMEM-NMEM
SWP7 SKN RMC+NMEM,2; BRU SWP8; SKG RMA+NMEM,2; BRU SWP8
LDA RMA+NMEM,2; STX SWMPTR
SWP8 BRX SWP7; LDX SWMPTR; LDA XX; STA RMA+NMEM,2
SKN RMT+NMEM,2; BRU SWP6
LDA RMT+NMEM,2; SKA X2; BRU SWP10 (READ-ONLY)
LDX SWT7; STA SWT14=SWT5,2; XXA; LDX 0,2
COPY XA,AX,B; STA 0,2
RSH 6; STB SWT13; RSH 8; LDB SWT13; LSH 1;
ETR SECMSK; XXA
ADD =SAT=SWT5; XMA SPT,2; STA* SPT,2
MIN SWT7; BRU SWP6
* RELEASE READ-ONLY PAGE
SWP10 CLB; STB RMT+NMEM,2; CAX
LDA 0,2; MRG X4; STA 0,2; BRU SWP6
* NOT ENOUGH, RELEASE LOCKS AND EXIT
SWP5 LDX =-10

```

```

SWP46 LDA SRTE,2; SKA X4; BRU SWP47
      ETR =37B; XXA; SKR RMC,2; NOP; CAX
SWP47 BRX SWP46; BRU SWP50 (FAILURE)
* FIND CURRENT DRUM POSITION
SWP12 CLB; SKN NRCL; BRU SWP35
      LDA RDRAD; ADD =1; ETR RADMSK (1ST WRITE RAD)
      LSH 13; STA WRRAD; RSH 3; MRG PINE0D
      STA **+1; E0D 0; PIN TSWP1; LDA TSWP1; MRG WRRAD; BRU SWP36
SWP35 LDA ERCL; SKE =RADQ; BRU **+2; LDA =RDQU; SUB =4; CAX; LDA 0,2
SWP36 LRSR 6; CBX; RSH 8; ETR RADMSK; CXB
      LSH 1; STA SWT12 (OPTIMAL DRUM POSITION)
      LDA SWT7; SUB =SWT5; STA SWT10; ADM PAGES
* CHECK FOR REMAINING COMMANDS
SWP15 CLA; SKE SWT10; BRU SWP14; MIN DSWAPC
      BRM RST; BRU SWP38 (DONE); BRM PTRL
SWP50 LDA SWR1; LDB SWR2; LDX SWR3; BRR SWAP
* FIND OPTIMAL COMMAND
SWP14 LDX SWT12; SKN SPT,2; BRU SWP16
SWP17 LDA SWT12; ADD =1; ETR SECMSK; STA SWT12; BRU SWP15
SWP16 LDA SPT,2; ADD =SWT5-SAT+1; SKG SWT6; BRU SWP19
* REMOVE COMMAND FROM STACK
SWP20 LDA* SPT,2; XMA SPT,2; ADD =SWT5-SAT; CAX; STA SWT11
      ADD =1; SKG SWT6; BRU SWP27 (READ); BRU SWP48 (WRITE)
* ASSIGN AND LOCK BLOCK, OUTPUT READ, UPDATE PMT/SMT
SWP19 LDX =NSMEM-NMEM; SKN RMT+NMEM,2; BRU SWP21
      BRX **=2; BRU SWP17
SWP21 CXA; ADD =NMEM; STA SWMPTR; LDX SWT12; BRU SWP20
SWP27 LDX SWT14-SWT5,2; BRM 0MR; BRU SWP23
* OUTPUT WRITE, RELEASE BLOCK, AND UPDATE PMT/SMT
SWP48 BRM 0MW; LDA* SWT11; ETR =37B; AXC; STA RMT,2; STA RMA,2
      LDA SWT11; LDX SWT14-SWT5,2; LDA 0,2; MRG X4; STA 0,2
SWP23 SKR SWT10; BRU SWP48 LDA SWT12; EOR =1; STA SWT12
      BRM 0MNCR
15
* UNPACK RELABELLING REGISTERS
* INPUT; RELABELING IN A,B,X
* OUTPUT; ALL REGISTERS CLOBBERED
UPRL ZR0; STB SWT3; LRSR 18; STA SRT
      CLA; LCY 6; STA SRT+1; CLA; LCY 6; STA SRT+2
      CLA; LCY 6; STA SRT+3; LDA SWT3; LRSR 18; STA SRT+4

```

```

CLA; LCY 6; STA SRT+5; CLA; LCY 6; STA SRT+6
CLA; LCY 6; STA SRT+7; CXA; RSH 6; ETR =77B; STA SRT+8
CLA; LCY 6; STA SRT+9; LDA =-1; STA DCRT3; BRR UPRL
* PACK RELABELING WORD
PKRL4 ZR0
PKRL ZR0; STA PKRL1
PKRL1 LDA SRTE,2; COPY XB,AX; SKA X4; BRU PKRL2; ETR =37B; XXA
SKN DCRT3; SKR RMC,2; NOP; CBX
PKRL3 RCY 6; LDA PKRL4; LCY 6; STA PKRL4; BRX PKRL1; BRR PKRL
PKRL2 LDA 0,2; CBX; BRU PKRL3
* PACK TOTAL RELABELING
PTRL1 LDA SRT+4,2
PTRL2 LDA SRT+8,2
PTRL3 LDA SRTE,2
PTRL ZR0; LDA PTRL1; LDX =-4; BRM PKRL; STA SWD1
LDA PTRL2; LDX =-4; BRM PKRL; STA SWD2
LDA PTRL3; LDX =-2; BRM PKRL; ETR PRMSK
CAX; LDA SWD1; LDB SWD2; BRR PTRL
* DECODE RELABELLING
DCRL ZR0; BRM UPRL; LDX =-10
DCRL1 LDA SRTE,2; SKE =0; BRU DCRL2; LDA =40B
* STORE RESULTING SRT ENTRY
DCRL9 STA SRTE,2; BRX DCRL1; BRR DCRL
* STORE PMT/SMT ADDRESS IN SRT
DCRL2 COPY XB,AX; SKG =NCMEM-1; BRU DCRL3+1; EAX* PMTJOB
LDA 0,2; SKA X4; BRU DCRL5; SKA X1; BRU DCRL3
DCRL4 ETR =77B; CBX; BRU DCRL9
DCRL3 CAX; LDA SMT,2; SKN SMT,2; BRU DCRL4
COPY XA,BX; ADD DCRL8; MIN DCRT3; BRU DCRL9
DCRL5 COPY XA,BX; MRG X4; MIN DCRT3; BRU DCRL9
DCRL8 ZR0 SMT,5
* RESET SAM
RSAM ZR0; CLA; STA SACNT; STA SAPTR; LDX =NSMEM-NMEM
RSAM1 LDA RMA+NMEM,2; RSH 1; STA RMA+NMEM,2; BRX RSAM1; BRR RSAM
* OUTPUT MEMORY WRITE
0MW ZR0; LDA 0,2; CAX; ETR =177700B; RSH 1
XXA; LSH 11; ETR =174000B
CXB; LDX =4000B; BRM RTW; BRR 0MW

```

\*

```

* OUTPUT MEMORY READ
* INPUT: X=PMT ADDR., SWMPTR=PAGE NUMBER.
0MR ZR0; STX 0MR1
    LDA 0,2; ETR =37B; STA 0MR2; CXA; ETR ADMSK
    SKG =PMTM1; STA 0MR2
    LDA 0,2; ETR =37777740B; MRG SWMPTR; STA 0,2; CAB
    ETR =37B; XXA; ETR ADMSK; MRG X4; SKB =40B; MRG X2
    STA RMT,2; LDA X2; STA RMA,2
    CLA; SKE 0MR2; BRU **2; BRR 0MR
    LDX 0MR1; LDA SWMPTR; LSH 11; ETR =174000B
    LDX 0,2; XXA; ETR =177700B; RSH 1; COPY AB,XA; LDX =4000B
    BRM RTC; LDX RTXS2; LDA =3; ADM 3,2; BRR 0MR
0MR1 ZR0
0MR2 ZR0 0
*
* SET UP REAL RELABELING. KEEPS PAGE 6 RELABELLING FROM RRL3.
* INPUT: REAL RELABELING IN A,B,X
LABEL ZR0; STA RRL1; STB RRL2; CXA; SKA =3700B; BRM M0NCR
    ETR =77B; XMA RRL3; ETR =7700B; ADM RRL3
    LRR1; POT RRL1; LRR2; POT RRL2; LRR3; POT RRL3; BRR LABEL
* SETS UP RELABELING FOR A FORK. SAVES RRL3 IN BRRL3
* INPUT: X=PACPTR
* SKIPS IF SUCCESSFUL
CHRL ZR0; LDA RL1,2; LDB RL2,2; LDX J0B; LDX RL3,2
    BRM SWAP; BRR CHRL; STX BRRL3; BRM LABEL; MIN CHRL; BRR CHRL

DCRT3 ZR0
SWR1 ZR0 0 PSEUDO=RELABELLING INPUT
SWR2 ZR0
SWR3 ZR0
SWR4 ZR0 0 SWR3 + J0B
SWD1 ZR0 0 REAL SWR1
SWD2 ZR0 0 REAL SWR2
SACNT ZR0 0 SAM COUNT. COUNTS NEGATIVE
SAPTR ZR0 0 SAM POINTER
SAM1 BES NSAM SWAPPER ASSOCIATIVE MEMORY
SAM2 BES NSAM
SAM3 BES NSAM
SAR1 BES NSAM

```

```

SAR2  BES NSAM
SAR3  BES NSAM
SRT   BSS 10  TABLE FOR DECODING PSEUDO-RELABELLING
SRTE  EQU *    CONTAINS RRL BYTES OR POINTERS TO PMT/SMT
SWT3  ZR0
SWT5  BSS 20  PMT/SMT ENTRIES TO BE READ OR WRITTEN
SWT14 BSS 20  PMT/SMT ADDRESS FOR ENTRIES IN SWT5
SAT   BSS 20  POSITION STACKS
SWT6  ZR0 0   FIRST WRITE COMMAND
SWT7  ZR0 0   END OF WRITE COMMANDS
SWT10 ZR0 0   COMMAND COUNT
SWT11 ZR0 0   OPTIMAL READ/WRITE
SWT12 ZR0 0   CURRENT DRUM POSITION
SWT13 ZR0 0   TEMP STORAGE
RDRAD ZR0 0   NO. OF READ RAD
WRRAD ZR0 0   NO. OF RAD IN THIS SET OF WRITES
PINE00 DATA 0610226B  PIN E00
SWMPTR ZR0 0   POINTER FOR SCAN FOR FREE MEMORY
SPT   BSS NSEC POSITION STACK POINTERS
TSWP1 ZR0 0   RAD SECTOR NUMBER.
RADMSK DATA  NRAD=1      RAD MASK
RRL1  ZR0
RRL2  ZR0
RRL3  ZR0 0   CURRENT CONTENTS OF RR3
BRRL3 ZR0 0   RR3 FOR RUNNING FORK
RLTS  ZR0 0   TS BLOCK RELABELLING FOR RUNNING FORK
RLPACT DATA 600B  PAC TABLE REAL RELABELLING
RLDSC DATA 1000B  DISC REAL RELABELLING
RLWB  DATA 1100B  W BUFFER DRIVERS REAL RELABELLING
*
*
* ROUTINES TO RELABEL EXTRA MONITOR PAGES.
*
* PAC TABLE RELABELING
MPPACT ZR0; LDA RLPACT; MRG RLTS; STA RRL3
      LRR3; PGT RRL3; BRR MPPACT
*
* DISC RELABELING
MPDSC  ZR0; LDA RLDSC; MRG RLTS; STA RRL3

```



LRR3; PBT RRL3; BRR MPDSC

\*

\* W BUFFER ROUTINE RELABELING

MPWB ZR0; LDA RLWB; MRG RLTS; STA RRL3

LRR3; PBT RRL3; BRR MPWB

IF V1

\*

\* READ OR WRITE ON DISC (BE+1, BE+2)

\* INPUT: A=CORE ADDR. B=DISC ADDR. X=WORD COUNT.

\* FLAGS KEPT BY JOB IN TTN0. X2 IS BUSY BIT.

\* OP CODE FIELD IS: 00 START READ, 01 OPERATION IN PROGRESS,

\* 10 READ DONE=NO ERRORS, 11 READ DONE=ERRORS.

AWD LDA =77B; LDB =DTW; BRU ARD4

ARD LDA =17B; LDB =DTC

ARD4 SKA SS03; BRM TRAPB DISC BLOCKED IO REQUIREMENT

LDA SS03; SKG =0; BRM TRAPB; SKN PQU,2; BRM TRAPB

STB ARD1; LDX JOB; LDA TTN0,2

SKA =300000B; BRU ARD2 (JOB FINISHED); LDA SS01

ETR ADMSK

STA T; ADD SS03; SUB =1

ETR NADMASK; SKE =0; BRM TRAPB; LDA T; CAX; LDB 0,6

BRM DTH; LDA SS02; SKG RLDMSK; SKG =-1; BRM TRAPB

CAB; BRM MPDSC

LDA T; LDX SS03; BRM\* ARD1; LDA =20100000B

LDX JOB; ADM TTN0,2; LDA =400000B; LDX DTXS2

ADM 2,2; LDB JOB; LSH 40; ADM 1,2; BRM DTS

ARD3 LDA =TTN0; ADD =1100000B; ADD JOB; CAB

BRU NI0QDS

ARD2 SKA =20000000B; BRU ARD3; CAB; ETR =77B; STA TTN0,2; CBA

SKA =100000B; BRU \*\*2 (ERRORS); BRU POPX

LDA =1000B; LDX PACPTR; BRM IIR; BRM TRAPB; BRU POPINT

ARD1 ZR0 0

\*

\* READ OR WRITE 2K ON DISC (BRS BE+9, BE+10)

\* AVAILABLE ONLY TO EXEC. THIS BRS SHOULD NOT BE GENERALLY USED.

WDSYB LDB =DTW; BRU RDSYB+1

RDSYB LDB =DTC; STB RDRS1; SKN PQU,2; BRM TRAPB

```

LDA SS01; ETR =34000B
CAX; LDB 0,6; BRM DTH
BRM MPDSC
LDB SS02; LDA T; LDX =4000B; BRM* RDBRS1
BRM DST; BRU NP0PX; BRM NTRPB
RDBRS1 ZR0
*
* READ OR WRITE 2K ON RAD (BRS 104,105)
WSYB LDB =RTW; BRU RSYB+1
RSYB LDB =RTC; STB RSYBT1; ETR =34000B; SKN PQU,2; BRM TRAPB
CAX; LDB 0,6; BRM DTH
RSYB2 LDB SS02; LDX =4000B; BRM* RSYBT1
BRM RST; BRU P0PX; BRM TRAPB
RSYBT1 ZR0
ENDF
*
*
IF V1
* START RAD AND WAIT
RST ZR0; LDA RUERR; STA RSTE
BRM RTS; SKN NRCL; BRU *-1
LDA RUERR; SKE RSTE; MIN RST; BRR RST
$RSTE ZR0
ENDF

*
* CLOCK ROUTINES AND TABLES
*
QUTAB ZR0 NFQU NUMBER OF CLOCK CYCLES IN QUANTUM
TIME ZR0 0 SHORT
TTIME ZR0 0 LONG
UMTF ZR0
CLOCK3 ZR0
* CLOCK INTERRUPT ROUTINE AVG. TIME = .047 MS
CLINT ZR0; MIN REAL; MIN* TJOB; SKN CLINT; MIN STIME
BPT4
CLINTC BRM M0NCR

```

```

        SKR TTIME; BRU **2; BRU CL0UT; SKR TIME; BRI CLINT
        SKN ACTR; BRU CL0UT; BRI CLINT
CL0UT  SKR UMTF
UMTN   N0P; SKN CLINT; BRU CKOI
        XMA CLINT; STA TRAPT; LDA CLINT; BRI =TRAPT+1
CKOI   STA SA; LDA* CLINT; ETR =20077777B; SKE =40000B
        BRU 0K; LDA =100; STA C0UNT; STX SX
        SKN 0; BRU 0KX; LDX 0
LPT    LDA 0,6; SKA =40000B; BRU IND; BRU 0KX
IND    SKA X2; ADD SX; CAX; SKR C0UNT; BRU LPT
        BRI =TRAP
0KX    LDX SX
0K     LDA SA; BRI CLINT
SA     ZR0 0
SX     ZR0 0
C0UNT  ZR0 0
* USER MODE TRANSITION TRAP
TRAPT  ZR0; STA SS01; STB SS02; STX SS03
        LDA UMTF; SKE =1; BRU TRAPT1; LDA SS01; BRU* TRAPT
TRAPT1 LDA TRAPT; STA 0; BRU PACQE
* FAST CLOCK (POWER OFF INTERRUPT)
        IF      V5
PWF1   ZR0; STA P0FFA; STB P0FFB; STX P0FFX; MIN PWFL
PWF11  BRU *; BRI PWF1
*
* POWER ON INTERRUPT
PWN1   ZR0; BRU* *
P0FFA  ZR0 0
P0FFB  ZR0 0
P0FFX  ZR0 0
        ELSEF  1
PWF1   ZR0; BRI PWF1
        ENDF
*
*
* BR'S FOR TIMING
* BR 88 READ COMPUTE TIME.
        IF      V5
RTEX   SKN PQ0,2; BRM TRAPB; LDX J0B; LDA ETTB,2; STA SS01; BRU P0PX

```

```

RTX      ELSF      1
        LDX J0B; LDA ETTB,2; STA SS01; BRU P0PX
        ENDF

*
* BRS 42
        IF          V6
RREAL   LDA REAL; STA SS01; LDB DMIN; STB SS02; BRU P0PX
        ELSF      1
RREAL   LDA REAL; STA SS01; BRU P0PX
        ENDF

*
* BRS 81  DISMISS FOR SPECIFIED TIME. TIME IN MS.
WREAL   MUL =1727024B; SKB X4; ADD =1; ADD REAL; STA SS01
        LDX =0T; LDB WREC1; MIN 0; BRU P0PDMS
WREC1   + **1; LDA PA,2; CACR 1,(SKG REAL)

*
* BRS 57  GUARANTEE'S 16MS.
C00     LDA TIME; SKG =1; BRU S00; BRU P0PX

*

* SYSTEM START
SETSET  DISW; TSN; LDA 22B; STA DISCA
SETST1  LDA =DASET; ADD DISCA; ADD DRC; LDB =CASET
        LDX =4000B; BRM GDC; BRU SETST1
        LDA MSETRL; STA RRL3; LRR3; P0T RRL3; BRU SETSA
MSETRL  DATA      612B          RELABELLING FOR SET
DISCA   ZR0        0            DISC ADDRESS OF SYSTEM

*

* GET DISC BLOCK TO CORE. A=DISC ADDR, B=CORE ADDR, X=COUNT
* SKIPS IF NO ERRORS
GDC     ZR0 0; ARMI =600000B
        STB GDCL; STA GDCL+1; LDA GDCL; ETR ADMSK; RSH 14
        CXA; ETR =77777B; LCY 14; STA GDCL+2; LSH 19
        LDA GDCL; ETR =140000B; CBX; RSH 14
        CXB; LSH 5; MRG =I0RDW; STA GDCL+3
        DRT; BRU *=1; ALD; P0T GDCL+1

```

```

EOM* 10000B; EXU GDCL+3; PBT GDCL+2
DSR
DRT; BRU **1; DET; BRU GDC2; DCT; BRU GDC2
MIN GDC
GDC2 BRR GDC
GDCL DATA 0 REAL CORE ADDRESS
DATA 0 DISC ADDRESS
DATA 0 WC+CORE ADDRESS
DATA 0 IBSD WITH EXTENDED ADDRESS AND WORD COUNT
GCRL DATA 0
GCRL1 DATA 0
GCRL2 DATA 0
GCRL3 DATA 0

```

```

*
*   QUEUE ROUTINES
*
* INPUT: A=PACPTR, X=QUEUE,
QPUT ZR0; LDB 2,2; XXA; STB PNEXT,2; XXA; LDB 1,2; STA 1,2
      CBX; STA PNEXT,2; BRR QPUT
*
* INPUT: A=PNEXT OF FORK TO GET, B=QUEUE (QI0Q), X=PREVIOUS PACPTR
QGET ZR0; STA PNEXT,2; XXB; SKG =0; BRR QGET; STB PNXT1,2; BRR QGET
*
* INPUT: A=NEW ACT. COND. X=PACPTR,
* QSCH CHANGES ACTIVATION CONDITION AND TAKES PAC ENTRY OFF
*   THE QUEUES.
QSCH ZR0; XMA PTEST,2; EBR =700000B; SKA =7700000B; BRU **2
      BRR QSCH (OLD ACT. COND. WAS 7. FORK NOT ON QUEUES.)
      LDA PNEXT,2; STA QSCH1; STX QSCH4; LDA =QTI
QSCH2 SKG =0; BRU QSCH3; SUB =PNEXT; CAB
QSCH3 CAX; LDA PNEXT,2; SKE QSCH4; BRU QSCH2
      LDA QSCH1; BRM QGET; LDX QSCH4; BRR QSCH
QSCH1 ZR0
QSCH4 ZR0
*
*   INTERRUPT LOGIC
*

```

```

* IIR      MAKES ACT. COND. A PRGR. INT. AND PUTS FBRK ON Q10
* INPUT; A=INT. MASK.  X=PACPTR.
IIR      ZR0; SKA PIM,2; BRU **2; BRR IIR
          STA IIR1; STX IIR2; EOR PIM,2; STA PIM,2
          CLEAR; LDA IIR1; NOD 24; CXA; SUB =500000B=2; CNA
          CAB; LDX IIR2; LDA PTEST,2; SKE =700004B; BRU IIR3
          LDA PIM,2; SKA X1; BRR IIR (NON-TERMINABLE BRS)
          STB IIR1; LDA PPTR,2; RSH 12; CAX; BRM DFK; LDB IIR1
IIR3     CBA; LDX IIR2; BRM QSCH; CXA; MRG X7
          LDX =Q10; BRM QPUT; LDX IIR2; MIN IIR; BRR IIR
IIR1     ZR0
IIR2     ZR0
*
* SEARCH FBRK STRUCTURE AND INTERRUPT
* CHECK PARALLEL FBRKS FIRST. THEN GO UP FBRK STRUCTURE.
* INPUT; A=INT. MASK.  X=PACPTR.
SIR      ZR0; STA SIR1; LDA PPTR,2; MRG PLMSK; CAX
          LDA PPTR,2; LRSR 12
SIR2     SKA PRMSK; BRU SIR3
SIR4     LDA PPTR,2; SKA PRMSK; BRU **2; BRR SIR
          MRG PLMSK; CAX; LDA SIR1; BRM IIR; BRU SIR4
SIR5     MIN SIR; BRR SIR
SIR3     MRG PLMSK; CAX; LDA SIR1; BRM IIR; BRU **2; BRU SIR5
          LDA PQU,2; BRU SIR2
SIR1     ZR0
*
      IF      V1
* MONITOR CRASH
M0NCR   ZR0; SKN M0NCR; BRU CRA2; MIN MCRC; BRU CRA2
CRA2    CKF; DIR; STA MCRA; STB MCRB; STX MCRX; MIN MCRC
          LDA 0; STA MCRO; CLA; SKE JOB; BRU **2; BRM ST0P
          LDA =1; SKE MCRC; BRM ST0P
* SAVE USER'S TS BLOCK
          LDA RLTS; ETR =77B; SKE =0; BRU **2; BRM ST0P
          SKE =40B; BRU **2; BRM ST0P
          MRG =700B; STA CRA5; LRR3; POT CRA5
          BRM CRA3
$ST0P   ZR0; BRU *
CRA3    ZR0; LDX =-3777B; LDA 3777B,2; STA 33777B,2; BRX **2

```

```
LDA 37777B; STA 33777B; BRR CRA3
CRASH SKN PQU,2; BRM TRAPB; BRU CRA2
CRA5 ZR0 0 TS BLOCK RELABELLING
MCRA ZR0 0
MCRB ZR0 0
MCRX ZR0 0
MCRO ZR0 0
MERC ZR0 0
ENDPAC BSS 0
      ENDF
      END
```

3PMT IDENT 7/01/67

\*

ENTRY SMT, SMTE, PMT, PMTM1  
ENTRY Q0, Q1, Q1N, Q1E, NLQ  
ENTRY PNEXT, PL, PA, RL1, RL2, PPTR, PTEST, PQU, PTAB, PIM  
ENTRY PNXP1, PL4, PACT1, PPTR2, PPTRU

\*

\*

\* THE FIRST ENTRIES MUST BE SET TO THE EXEC AND COMMON SUBSYSTEMS

SMT DATA 40B,20000001B,20000002B,20000003B,20000004B,20000005B  
DATA 20000006B,20000000B  
DB 6,0,2,3; DB 22,0,1,3; DB 7,0,2,3 (EXEC-10)  
DB 35,0,2,1 (XFTC=15); BSS 1  
DB 20,0,2,3 (DDT=20); DB 5,0,2,1 (QED=22)  
DB 31,0,5,1 (CAL=24); DB 23,0,4,1 (SN0B0L-31)  
DB 12,1,4,1 (CCS1=35); DB 17,0,2,1 (FTC=41)  
DB 14,1,2,1 (CCS5=43); DB 33,0,3,1 (XBASIC=45)  
DB 15,1,3,1 (CCS7=50); DB 25,1,3,1 (XF0S=53)  
DATA 20002110B (DISC=56); DATA 20000311B (W=57)

\$NMSMT EQU \*-SMT  
BSS NSMT+SMT-\* SHARED MEMORY TABLE  
SMTE EQU \*  
PMTM1 EQU \*-1  
PMT BSS NUMEM\*NJ0B PROGRAM MEMORY TABLE. NUMEM=16.

\* QUEUES

QUEUE MACRO N  
\$Q.N(1) DATA Q.N(2),Q.N(1)-PNEXT,Q.N(2)  
\$Q.N(1),Q EQU Q.N(1)-PNEXT  
ENDM  
Q0 EQU \*  
\$QTI DATA QI0,QTI-PNEXT,QI0  
\$QI0 DATA QSQ,QI0-PNEXT,QSQ  
\$QSQ DATA QQE,QSQ-PNEXT,QQE  
\$QQE DATA QTI,QQE-PNEXT,QTI  
Q1 EQU \*  
Q1N EQU Q1+1



Q1E EQU Q1+2  
NLQ DATA Q0-Q1

\* PACT TABLE  
PACT BSS NPAC\*NPPAR  
PACT1 EQU PACT+NPPAR  
PDATA EQU \*  
PNEXT EQU PDATA+0  
PL EQU PDATA+1  
PA EQU PDATA+2  
RL1 EQU PDATA+3  
RL2 EQU PDATA+4  
PPTR EQU PDATA+5  
PTEST EQU PDATA+6  
PQU EQU PDATA+7  
PTAB EQU PDATA+8  
PIM EQU PDATA+9

\*  
IF V1  
\$QTIQ EQU QTI-PNEXT  
\$QIQ EQU QI-PNEXT  
\$QSQQ EQU QSQ-PNEXT  
\$QQEQ EQU QQE-PNEXT  
ENDIF

PNXTP1 EQU PNEXT+1  
PL4 EQU PL+4  
PPTR2 EQU PPTR+PACT-PDATA+NPPAR  
PPTRU EQU PPTR-NPPAR

END

```

3RAD IDENT 6/30/67
* ENTRY POINTS
RC EQU 1 CHECK ON RAD TIMING
ENTRY NRCL,RTS,RTC,RTW,RTP,RTXS1,RADI2
ENTRY RDR,RADI,RLIS,RADQ,RDQU,ERCL,RTXS2
RADQ BSS NRDQ*4;RDQU EQU *)* RAD I/O REQUESTS
*
*
* THIS IS THE RAD INTERRUPT ROUTINE FOR ALL RAD I/O
*
NRCL ZR0;* COUNT OF RAD COMMANDS IN LIST
ERCL ZR0;* CURRENT END OF LIST
RADTRY ZR0;* TRY-AGAIN COUNTDOWN FOR IO PARITY ERRORS
*
RDR ZR0 0
LDA =1; STA RDREX; LDA RRT; STA RADI+1
CLA; STA RLISE; MIN RCC3
LDX RLIS; LDA 2,2; STA RADIF; LDA 0,2; STA RADDR; LDA 1,2
STA CADD; LDA RRD; SKN 3,2; LDA RWR; STA RADIG
SKS* 14000B; BRU *=1 (CHANNEL READY)
DIR
EOD 10026B; PBT RADDR (RAD ADDR.)
BRU RADIA
RLIS ZR0 0 POINTER TO NEXT COMMAND
*RLISE ZR0 0 POINTER TO PREVIOUS COMMAND
RADI ZR0
SKS* 10026B; BRU *=1 (READY TEST)
EOD 10026B; PBT RADDR (RAD ADDRESS)
RADIK SKS* 11026B; BRM RDER (RAD ERROR)
SKS* 11000B; BRM RCER (CHANNEL ERROR)
*RADIA EOD* 10000B;
RADIF EOD 17200B; PBT CADD (CORE ADDR.)
RADIG EOD 2266B
STA RIA; STB RIB; STX RIX
IF RC
LDA REAL; STA* PRC1; MIN PRC1; LDA PRC1
SKE ERC1; BRU **2; LDA =RC1; STA PRC1
ENDF
CLA; SKE RLISE; BRM RFIX

```

```

LDX RLIS
STX RLISE SAVE PTR. TO THIS COMMAND
EAX 4,2 GET POINTER TO NEXT COMMAND
CXA
SKE =RDQU IS THIS THE END OF THE QUEUE
BRU **2 NO
LDX =RADD WRAP AROUND QUEUE
STX RLIS SAVE PTR. TO NEXT COMMAND
CLA; SKE NRCL; BRU **2; BRU REXE
LDA 0,2; STA RADDR; LDA 1,2; STA CADD
LDA 2,2; STA RADIF
LDA RRD; SKN 3,2; LDA RWR; STA RADIG
REXIT SKR RDREX; BRU DREX (DRIVER EXIT)
LDA RIA
LDB RIB
LDX RIX
BRI RADI
RCER ZR0; BRM RCHK; MIN RCERR; MIN RUERR; BRR RCER
RDER ZR0; BRM RCHK; MIN RADERR; MIN RUERR; BRR RDER
RCHK ZR0; STA RIA2; LDA* RLISE; STA* RD2A
MIN RD2A; LDA =RD2E; SKE RD2A; BRU **3; LDA =RD2; STA RD2A
LDA RIA2; BRR RCHK
$RD2A ZR0 RD2
$RD2 BSS 10 ADDRESS OF RAD ERRORS
RD2E EQU *
RIA2 ZR0 0
REXE LDA NRRT; STA RADI+1; BRU REXIT
DREX MIN RCC4; EIR; BRR RDR
NCMND SKS* 10026B; BRU *=1 (RAD READY TEST)
SKS* 11026B; BRM RDER (RAD ERROR TEST)
SKS* 11000B; BRM RCER (CHANNEL ERROR)
STA RIA; STB RIB; STX RIX
BRM RFIX
BRU REXIT
RFIX ZR0; LDX RLISE; LDA 1,2; RSH 9; EOR 2,2; ETR =37B
EOR 2,2; RSH 2; ETR =37B; COPY AX,XA,AB; SKR RMC,2; NOP
MIN RCC1
CAX; LDA 3,2; ETR =77B; CAX; BRM* IRT,2
SKR NRCL; NOP; BRR RFIX

```

```

RIA      ZR0      0
RIB      ZR0      0
RIX      ZR0      0
$RADDR   ZR0      0          RAD ADDR,
$CADD    ZR0      0          CORE ADDR.
RDREX    ZR0      0          0 FOR DRIVER EXIT
RRT      SKS*     10026B
RER      SKS*     11026B
RRD      E0D     2226B
RWR      E0D     2266B
NRRT     BRU      NCMND
          IF      RC
PRC1     ZR0      RC1
RC1      BSS      30
ERC1     ZR0      *
          ENDF
IRT      DATA RIN0P,RIN0P,RIN0P,IRS
RIN0P    ZR0; BRR RIN0P
* CHECK FOR SWAP ERRORS
IRS      ZR0; LDA RUERR; SKG RSTE; BRR IRS; COPY BX,A; STA RMA;2
          XMA RMT;2; CAX; LDA 0;2; MRG X4; STA 0;2; BRR IRS
*
*
* I2 INTERRUPT ROUTINE
RADI2    ZR0 0; STA RIA
          MIN RCC2; LDA RCC2; SKE RCC1; BRU N0I1; LDA RIA; BRI RADI2
N0I1     MIN I2ER;
          LDA RADI2; STA RADI; LDA RIA; BRU RADI+1
RCC1     ZR0      0      N0. 0F I1 INTERRUPTS
RCC2     ZR0      0      N0. 0F I2 INTERRUPTS
RCC3     ZR0      0      N0. 0F TIMES THROUGH DRIVER
RCC4     ZR0      0      N0. 0F TIMES THRU DRIVER EXIT
*
*
* GENERALLY USEFUL RAD ROUTINES
*
RTXS1    ZR0;* COMMAND COUNT
*

```

\* RESET COMMAND COUNT

RTP ZR0; LDA =1; STA RTXS1; BRR RTP

\* ENTER COMMAND IN LIST AND LOCK MEMORY BLOCK

\* A = ABS CORE ADDR

\* B = RAD ADDR

\* X = WORD COUNT

```
RTC   ZR0
      MIN      RTXS1      INCREMENT NO. OF RAD COMMANDS
      STA      RTTMP      SAVE CORE ADDR. Δ R/W BIT
      ETR      =37777B    GET CORE ADDR. = 16 BITS
      XAB
      STA*     ERCL       STORE RAD ADDR.
      LSH      10         LEFT JUSTIFY CORE ADDR. IN B MINUS 2 HIGH
                          ORDER BITS
*
      CXA
      ETR      =77777B
      LCY      14         WORD CNT + CORE ADDR IN A
      LDX      ERCL
      STA      1,2
      LSH      19         LEFT JUSTIFY 4 BITS OF WORD CNT LEAVE BL
RTC1  LDA      RTTMP      CORE ADDR.
      ETR      =140000B
      CBX
      RSH      14
      CXB
      LSH      5
      MRG      =00617200B  I0SD
      SKN      RTTMP      SKIP IF WRITE
      MRG      X4         CHANGE TO NOP TO DELETE EARLY I1
RTC2  LDX      ERCL
      STA      2,2
      CLA
      SKN      RTTMP; MRG X4
      STA      3,2
      LDA      RTTMP; RSH 11; ETR =37B; CAX; MIN RMC,2
      LDA      ERCL
      STA      RTXS2
      ADD      =4
      SKE      =RDQU      IS THIS THE END OF THE QUEUE
```

```

BRU      **2          NO
  LDA    =RADQ      YES - WRAP AROUND QUEUE
STA      ERCL
LDA      #NRDQ=2
SUB      RTX$1
SKG      NRCL
BRU      **1

*
          WAIT FOR QUEUE TO EMPTY A BIT
RTW      BRR        RTC
          ZR0
          MRG        X4      SET FOR WRITE
          BRM        RTC
          BRR        RTW
RTXS2    ZR0        0          PREVIOUS COMMAND POINTER.
RTTMP    ZR0        0
* RAD START
RTS      ZR0; LDA RTX$1; SKA X4; BRU RTS1; ADD =1; DIR; XMA NRCL; ADM NRCL
          EIR; SKE =1; BRU RTS1; LDA #NRTRY=1; STA RADTRY;
          BRM RDR
RTS1     BRM RTP; BRR RTS

          END

```

3RUP IDENT 6/30/67  
\* STRING AND FLOATING POINT SYSPOPS 7/30/66

IF V6  
SCIT 8PD 13400000B,1,1  
ENDF  
SWCI 8PD 15700000B,1,1  
SCI0 8PD 16100000B,1,1  
SBR5 8PD 17300000B,1,1  
SSKSE 8PD 16300000B,1,1  
UGCI MACR0 A  
EAX A(1)  
BRM GCU  
ENDM

SYSPOPS WITH BIT 0 REMOVED ...  
... FOR USE BY SYSTEM MODE ROUTINES

\* UTILITY BRS'S (33,34,35,37)

\* BRS 33

\*GETSTR STB GSIN1  
STX GSIN3  
ETR =40037777B  
CAX  
LDB 1,6  
SKA =40000000B  
STB 0,6  
EAX 0,6  
STX GSIN2  
SCI0 GSIN3  
GSIN4 SWCI\* GSIN2  
SCI0 GSIN3  
SKE GSIN1

BRU GSIN4  
LDA 0,6  
LDB 1,6  
LDX GSIN3  
BRU EP0PX

\* BRS 34

\*OUTMSG STB STR03  
MUL #3  
LSH 23  
SUB #1  
SKN STR03  
BRU TYM2  
LDB #137777B  
BRU STR06  
TYM2 CAB  
ADD STR03  
XAB  
BRU STR06

NOT SPECIAL MODE

\* BRS 35

\*OUTSTR ADD #0 CLEAR X\0  
STX STR03  
STR06 ADD #0 CLEAR X\0  
STX STR01  
ETR #177777B  
STA STR02  
CBA  
ETR #177777B  
STR04 STA STR021  
UGCI STR02  
BRU STR08  
SKN STR03  
BRU STR07



	SKE	=17B
	BRU	STR05
STR08	L <sub>DX</sub>	STR01
	BRU	EP0PX
STR05	SKE	=4
	BRU	STR07
	L <sub>DA</sub>	=155B
	IF	V6
STR07	SKG	=77B
	BRU	STR07A
	SKE	=147B
	BRU	**2
	BRU	STR07A
	SKE	=155B
	BRU	**2
	BRU	STR07A
	SKE	=152B
	BRU	**2
	BRU	STR07A
	SKN	UEXFLG
	BRU	STR04
	CAB	
	L <sub>DA</sub>	=6
	SCI0	STR01
	CBA	
	SUB	=100B
STR07A	SCI0	STR01
	BRU	STR04
	ELSF	1
	SCI0	STR01
	L <sub>DA</sub>	=152B
STR07	SCI0	STR01
	BRU	STR04
	ENDF	

BELL

AMPERSAND

```

$GSL00K STX      WR13
  ABC
  STA      WR7          SET 'FIRST MATCH'
  LCY      12
  SKG      =0
  LDA      UC0UT
  STA      WR1
  CLA
  LCY      12
  STA      WR2
  LDA      SS02
  ETR      ADMSK
  MRG      X4
  STA      WR3
GSL14A LDA      1,6
  SUB      0,6
  STA      WR5

  LDX      WR13        PICK UP HASH TABLE BOUNDS
  LDA      0,6
  LDB      1,6
  STB      WR6
  CAX
  BRU      **3

GSL19 EAX      3,2
  CXA
  SKE      WR6          END OF HASH TABLE SCAN
  BRU      GSL18       N0

  SKN      WR7          UNIQUE MATCH FOUND
  BRU      GSL14       N0

  LDX      WR10        TRACE AND 0/P REMAINDER OF STRING
  LDA      1,6
  ETR      =177777B
  CAB

```

```

LDA WR9
STA WR11
STB WR12
GSL19A UGCI WR11
BRU GSL15
SCIT WR2 CHARACTER INPUT IF TESTS EQUAL
BRU **2 DOESN'T COMPARE
BRU GSL19B
SKG =72B CK IF ALPHA CHAR.
SKG =40B
BRU GSL15 NON-ALPHA, SO PROPER TERMINATOR
LDA WR5; SKE =0; BRU GSL14; BRU GSL15

GSL19B STA WR5; BRU GSL19A
GSL15 LDX WR10 EXIT VIA HERE IF MATCH FOUND
LDB 2,6
CXA
MIN SBRST SET SKIP ON RETURN
BRU GSL22

GSL14 LDX WR3 EXIT VIA HERE IF NO MATCH FOUND
LDA 0,6
LDB 1,6
GSL22 LDX WR13
BRU EP0PX

GSL16 CLA
STA WR7 SET 'FIRST MATCH
GSL18 LDA 0,6
SKG =0 HASH TABLE ENTRY EMPTY
SKG **2
BRU **2
BRU GSL19 YES, TRY NEXT ONE

LDA 1,6 NO, IS HASHT STR. SHORTER THAN GIVEN STRING
SUB 0,6
ETR =177777B
ADD =1
SKG WR5

```

	BRU	GSL19	YES, MATCH IMPOSS
	LDA	0,6	YES
	ETR	=177777B	
	CAB		
	ADD	WR5	
	XAB		
	SSKSE*	WR3	COMPARE PROPER INITIAL PARTS
	BRU	GSL19	NO MATCH
	SKN	WR7	MATCH, FIRST SO FAR
	BRU	GSL17	YES
	SCIO	WR2	NO, APPEND CHARS UNTIL ONE ELIMINATED
	SWCI*	WR3	
	MIN	WR9	
	MIN	WR5	
	LDA	WR8	
	LDB	WR9	
	SSKSE*	WR3	
	BRU	GSL16	PREVIOUS MATCH ELIMINATED
	BRU	GSL18	
GSL17	STA	WR8	
	STB	WR9	
	STX	WR10	
	LDA	=40000000B	RESET 'FIRST MATCH'
	STA	WR7	
	BRU	GSL19	

\*930 STRING PROCESSING SYSTEM. MANUAL IS DOCUMENT NUMBER 30.10.20

\*WRITE CHARACTER AND INCREMENT. THE POP ADDRESSES A STRING POINTER, AND  
 \*THE CHARACTER IN A IS WRITTEN ONTO THE END OF THE SPECIFIED STRING.

\*THE SECOND POINTER IS INCREASED BY ONE, SO THAT THE POINTER NOW  
\*POINTS TO THE NEW STRING. X AND A ARE PRESERVED, B DESTROYED.

WCI PBPB 15700000B,1,1,0,1

ETR =377B

STA WCH1

SAVE CHARACTER TO BE WRITTEN

STX WCH2

SAVE X REGISTER

EAX\* 0

GET EFFECTIVE ADDRESS OF PBP

MIN 1,6

INCREMENT SECOND WORD OF STRING POINTER

LDA 1,6

AND PICK IT UP.

BRU WCH5

GO TO WCH CODE, WHICH IS IDENTICAL FROM

\*WRITE CHARACTER AND DECREMENT: DECR. 1ST PTR.

\*AND WRITE CHAR. ON BEGINNING OF STRING

WCD PBPB 13500000B,1,1,0,1

ETR =377B

STA WCH1

STX WCH2

EAX\* 0

LDA 0,6; SKR 0,6; NOP

BRU WCH5

\* THIS POINT ON.

\*STORE POINTER. THE A AND B REGISTERS ARE STORED IN THE WORD ADDRESSED

\*BY THE PBP AND THE NEXT WORD

STP PBPB 16700000B,1,1,0,1

STX STP1

SAVE INDEX REGISTER

EAX\* 0

GET EFFECTIVE ADDRESS

STA 0,6

DO THE

STB 1,6

STORE

LDX STP1

RESTORE X

BRR 0

RETURN

\*LOAD POINTER. THE A AND B REGISTERS ARE LOADED FROM THE WORD ADDRESSED BY THE P

\*AND THE NEXT WORD.

LDP PBPB 16600000B,1,1,0,1

STX STP1

EAX\* 0

LDA 0,6

LDB 1,6

LDX STP1

BRR 0



```

      ETR =377B      REMOVE SUPERFLUOUS BITS
      LDX GCI1      RESTORE X
      BRR 0         AND RETURN
GC12  LCY 8         SHIFT TABLE FOR EXTRACTING A CHARACTER
      LCY 16        FROM A WORD
      CBA
      BRU TRAP      ILLEGAL STRING POINTER
* GCI WITH POINTER IN T.S, BLOCK
GCU   ZR0
      LDA GCU
      STA 0
      MIN 0,2
      LDA 0,2
      SKG 1,2
      BRU GCI8
      SKR 0,2
      NOP
      BRR GCU
*WRITE CHARACTER ON THE END OF STRING STORAGE, THIS IS THE ONLY POP WHICH
* CAN CAUSE A GARBAGE COLLECTION, IT EXPECTS THE ADDRESS OF THE
* PARAMETER TABLE IN X AND THE CHARACTER TO BE WRITTEN IN A AND
* RESTORES BOTH, B IS DESTROYED.
WCH POPD 16400000B,1,1,0,1
      ETR =377B
      STA WCH1      SAVE CHARACTER
      STX WCH2      AND INDEX (ADDRESS OF PARAMETER TABLE)
      EAX* 0
      MIN 0,6
*
*
*
      LDA 0,6
      SKE 1,6
*
*
      BRU WCH5
      LDA =1
      ADM 0,6
      LDA WCH1
      LDB 0
      BRU 2,6
WCH5  MUL =12525253B      COMPUTE WORD AND CHARACTER ADDRESSES

```

*		SEE GCI CODE
	CAX	SAVE WORD ADDRESS
	LCY 5	CHARACTER ADDRESS SHIFTED 3 IN AC
	ETR =30B	REMOVE EXTRANEIOUS BITS
	LDB 0,6	PICK UP WORD
	XXA	SHIFT COUNT TO X
	LCY 8,2	SHIFT IT TO PUT CHARACTER TO BE ALTERED
*		AT BOTTOM
	ETR =77777400B	MASK OUT THIS CHARACTER
	MRG WCH1	AND INSERT THE NEW ONE
	RCY 8,2	SHIFT WORD BACK
	CAX	RESTORE WORD ADDRESS
	STB 0,6	STORE WORD
	LDA WCH1	RESTORE A
	LDX WCH2	AND X
	BRR 0	AND RETURN

\*SKIP ON STRING EQUAL, THIS POP COMPARES THE STRINGS WITH POINTERS IN  
 \* AB AND IN THE TWO WORDS ADDRESSED BY THE POP. IT IS  
 \* EXACTLY IDENTICAL TO THE MACHINE COMMAND SKE. ALL REGISTERS ARE  
 \* PRESERVED. THIS POP CHECKS FOR EQUAL LENGTH STRINGS FIRST  
 \* RETURNING WITHOUT A SKIP IF THE STRINGS ARE NOT EQUAL. IF THEY  
 \* ARE, IT CALLS SKCB TO FIND OUT ABOUT EQUALITY OF THEIR CONTENTS

SKSE POPD 16300000B,1,1,0,1

STA SKS1	
STB SKS1+1	
STA SKS5	SAVE TWO COPIES
STB SKS5+1	OF FIRST STRING POINTER
STX SKS2	SAVE X
SUB SKS1+1	COMPUTE
STA SKS6	AND SAVE LENGTH
EAX* 0	GET ADDRESS OF SECOND STRING POINTER
LDA 0,6	COMPUTE
SUB 1,6	ITS LENGTH
SKE SKS6	AND COMPARE WITH FIRST
BRU *+2	NOT EQUAL
BRU SKSE2	EQUAL
LDX SKS2	NOT EQUAL. RESTORE X,
LDA SKS5	A AND
LDB SKS5+1	B



```

SKSE2  BRR 0          AND RETURN WITHOUT SKIPPING
      LDB 1,6        LENGTH EQUAL, PICK UP SECOND
      LDA 0,6        STRING POINTER
      LDX SKS2       AND ORIGINAL X
      BRM SKC0       COMPARE STRING IN AB WITH THE ONE IN SKS1
      BRU **2        AB LESS
      MIN 0          EQUAL, INCREMENT RETURN ADDRESS FOR SKIP
SKSE1  LDA SKS5       GREATER, RESTORE A
      LDB SKS5+1     AND B
      BRR 0          AND RETURN
*SKIP ON STRING GREATER. THIS POP IS TO SKG AS SKSE IS TO SKE. IT CALLS
*   SKC0 IMMEDIATELY WITHOUT COMPUTING LENGTHS
SKSG POPD 16200000B,1,1,0,1
      STA SKS1       SAVE
      STB SKS1+1     TWO COPIES
      STA SKS5       OF FIRST STRING.
      STB SKS5+1
      STX SKS6       SAVE X
      EAX* 0         GET
      LDA 0,6        SECOND STRING
      LDB 1,6        POINTER
      LDX SKS6       RESTORE X
      BRM SKC0       DO COMPARISON
      MIN 0          SECOND STRING LESS, I.E. FIRST GREATER(=.)
*                               INCREMENT RETURN ADDRESS FOR SKIP
      NOP 0          EQUAL
      LDA SKS5       FIRST LESS, RESTORE A
      LDB SKS5+1     AND B
      BRR 0          AND RETURN
*THIS ROUTINE COMPARES THE STRINGS IN AB (FIRST STRING) AND IN SKS1
* (SECOND STRING). IT RETURNS WITHOUT SKIPPING IF THE FIRST IS
* LESS, SKIPS ONCE IF THEY ARE EQUAL, AND SKIPS TWICE IF THE FIRST
* IS GREATER. NOTE THAT SEVERAL OF THE NECESSARY GCI'S ARE WRITTEN
* OUT INSTEAD OF BEING CALLS ON THE POP.
SKC0 ZR0 0
      STA SKS4       SAVE FIRST
      STB SKS4+1     STRING POINTER
      STX SKS7       AND X
SKC01 MIN SKS1      BEGIN LOOP. GET A CHARACTER

```

LDA SKS1	FROM SECOND STRING
SKG SKS1+1	(IS IT NULL\$\$
BRU SKC04	NO
SKC02 MIN SKC0	YES.) FIRST STRING CANNOT BE LESS, SO
*	INCREMENT RETURN ADDRESS
MIN SKS4	IS FIRST STRING NULL
LDA SKS4	
SKG SKS4+1	
MIN SKC0	NO, THEREFORE IT IS GREATER
BRU SKC05	RETURN
SKC04 MUL =12525253B	SECOND STRING NOT NULL, GET CHARACTER
RCH 401B	AS
LRSH 22	IN
LDA 0,6	GCI
RCH 24B	CODE
EXU GCI2,2	ABOVE
ETR =377B	
STA SKS3	GET IT
MIN SKS4	TRY TO GET CHARACTER
LDA SKS4	FROM FIRST STRING
SKG SKS4+1	IS IT NULL\$\$
BRU **2	NO
BRU SKC05	YES SO IT MUST BE LESS, RETURN WITHOUT
*	SKIPPING
MUL =12525253B	NOT NULL, GET
RCH 401B	CHARACTER
LRSH 22	AS
LDA 0,6	IN
RCH 24B	GCI
EXU GCI2,2	CODE
ETR =377B	ABOVE
SKE SKS3	AND COMPARE WITH CHARACTER OF 2ND STRING
BRU SKC03	NOT UAL
BRU SKC01	EQUAL, LOOP
SKC03 SKG SKS3	COMPARE FURTHER
BRU SKC05	LESS, RETURN WITHOUT SKIPPING
MIN SKC0	GREATER SKIP
MIN SKC0	TWICE
SKC05 LDX SKS7	RETURN POINT, RESTORE X

```

BRR SKC0                AND RETUR
* THIS ROUTINE, CALLED WITH BRS 5, SEARCHES A HASH TABLE FOR AN
* APPROPRIATE ENTRY FOR THE STRING IN AB. IT SKIPS IF THE STRING
* IS ALREADY IN THE TABLE. OTHERWISE IT SAVES THE INDEX OF THE
* PARAMETER TABLE AND DOES NOT SKIP. X IS PRESERVED. A AND B ARE
* PRESERVED IF THERE IS NO SKIP, BUT B IS THE INDEX IN THE HASH
* TABLE AND A THE VALUE (THIRD WORD) IF A MATCHING ENTRY IS FOUND
$SSCH STA SCH1                PUT STRING
      STB SCH1+1            POINTER AWAY
      STA SCH5                (TWO COPIES)
      STB SCH5+1
      CBA
      SUB SCH5                LENGTH OF STRING
      STA SCH9                AND SAVE IT
*
      RCH 402B                CAX + CLB
      SKG #3                  FIGURE OUT MASKS FOR FIRST 3 AND LAST 3
*                               CHARACTERS
      BRU SCH12                IF STRING IS LESS THAN 3 CHARACTERS LONG
      LDA #77777777B          NOT LESS. THE MASK
      STA SCH10                IS ALL
      STA SCH11                ONES
      BRU SCH13
SCH12 LDA SCH14,2            LESS. GET THE FIRST
      STA SCH10                CHARACTERS MASK
      LDA SCH15,2            AND THE LAST
      STA SCH11                CHARACTER'S MASK
SCH13 CXA                    RECOVER LENGTH
      LSH 3                    MULTIPLY BY 8
      STA SCH4                AND USE IT TO START THE HASH CODE
      LDA SCH5
      ADD #1
      MUL #12525253B          COMPUTE WORD ADDRESS FOR FIRST CHARACTER
*                               OF STRING
      RCH 401B                CAX + CLA. WORD ADDRESS TO X
      LRSB 22                 CHARACTER ADDRESS TO B
      LDA 0,6                 GET FIRST WORD OF STRING
      LDX 1,6
      XXB                     GET SECOND WORD

```

```

EXU SCH16,2
ETR SCH10
LRSH 12
ADM SCH4
CLA
LSH 12
ADM SCH4
LDA SCH5+1
MUL =12525253B
RCH #01B
LRSH 22
LDA 0,6
LDX =1,6
XXB
EXU      SCH17,2
ETR SCH11
LRSH 12
ADM SCH4
CLA
LSH 12
ADM SCH4
LDX SS03
LDA      1,6
SUB      0,6
XMA      SCH4
LRSH 23
DIV SCH4
CLA
LSH 1
DIV =3
MUL =3
LSH 23
ADD      0,6
CAX
STA      SCH20
LDA      =1
*
STA      SCH52
SCH50 LDA 0,6

```

```

SHIFT SO THAT FIRST 3 CHARACTERS ARE IN A
GET FIRST 3 CHARACTERS
FOLD THIS
WORD IN HALF, ADDING THE
HALVES
TO THE HASH
CODE BEING ACCUMULATED
REPEAT
THE
ABOVE
FOR
THE
LAST
THREE
CHARACTERS
OF
THE
STRING

```

```

RECOVER ADDRESS OF CONTROL TABLE
COMPUTE LENGTH
OF HASH TABLE
AND GET
HASH CODE
MOD

```

```

THIS LENGTH,
AND AN
EVEN MULTIPLE
OF 3

```

TURN IT INTO A HASH TABLE ADDRESS

```

SET THE 'DID WE PASS OVER A -1 (DELETED)
ENTRY' FLAG
TO IN019
BEGIN LOOP TO SCAN HASH TABLE9 NOTE

```



	CAB		B
	LDA	2,6	AND VALUE INTO A
	LDX	SS03	RESTORE X
	BRU	XP0P	AND EXIT
SCH7A	CBX		
SCH7	CXA		ENTRY DOES NOT MATCH, GO ON TO NEXT ONE.
*			INDEX OF CURRENT ENTRY TO A
	LDX	SS03	GET ADDRESS OF CONTROL TABLE
	ETR	=777777B	
	SKE	0,6	HAVE WE REACHED THE BEGINNING OF THE
*			TABLE IN OUR BACKWARDS SCAN
	BRU	++2	NO
	LDA	1,6	YES, CYCLE TO END
	SUB	=3	MOVE DOWN TO THE NEXT ENTRY (EACH
*			ENTRY IS 3 WORDS)
	CAX		
	SKE	SCH20	HAVE WE MADE A FULL CIRCLE
	BRU	SCH50	AND LOOP
	LDX	SS03	
	LDB	=1	
SCH51	BRU	SCH18	GIVE UP
	SKN	SCH52	FOUND A =1 ENTRY, IS THE FIRST ONE
	BRU	SCH7A	NO, DO NOTHING
	STB	SCH52	YES, SET FLAG POSITIVE ADD TO ADDRESS OF
*			ENTRY FOUND
SCH14	BRU	SCH7A	
	DATA	0,77600000B,77777400B	MASKS FOR FIRST 3 CHARACTERS OF STRING
*			NOT 3 CHARACTERS LONG
SCH15	DATA	0,377B,177777B	MASKS FOR LAST 3 CHARACTERS IF STRING IS
*			NOT 3 CHARACTERS LONG
SCH16	N0P	0	SHIFT INSTRUCTIONS TO GET FIRST 3
*			CHARACTERS OF STRING IN A REGARDLESS OF
*			CHARACTER ADDRESS
	LCY	8	OF THE FIRST CHARACTER
SCH17	LCY	16	
	RCY	16	SAME FOR LAST 3 CHARACTERS
	RCY	8	
	N0P	0	
*	BRS	6	INSERTS THE MISSING STRING

```

$SSIN  LDX  SS03
        LDA  2,6
        SKA  =40000000B
        BRU  TRAP
        CAX
        LDA  SS01
        STA  0,6
        STB  1,6
        CXB
        LDA  2,6
        LDX  SS03
        BRU  XP0P

```

```

WAS TABLE FULL
YES, CANIT FIT NEW ENTRY
NO

```

```

STP1   ZR0   0
GCI1   ZR0   0
GCI7   ZR0   0
WCH1   ZR0   0
WCH2   ZR0   0
SKS1   BSS   2
SKS2   ZR0   0
SKS3   ZR0   0
SKS4   BSS   2
SKS5   BSS   2
SKS6   ZR0   0
SKS7   ZR0   0
SCH1   BSS   2
SCH5   BSS   2
SCH4   ZR0   0
SCH9   ZR0   0
SCH10  ZR0   0
SCH11  ZR0   0
SCH20  ZR0   0
SCH52  ZR0   0

```

```

MASK FOR FIRST 3 CHARACTERS OF STRING
MASK FOR LAST 3 CHARACTERS OF STRING
STARTING LOCATION FOR SCAN OF HAST TABLE
FLAG IS -1 IF NO -1 ENTRY HAS BEEN
ENCOUNTERED DURING SCH SCAN, THE INDEX
OF THE FIRST SUCH ENTRY ENCOUNTERED
OTHERWISE

```

```

*
*
*

```

\* FLOATING POINT ARITHMETIC

* FAD	FLAATING ADD	
FAD	P8PD	15600000B,1,1,0,1
\$FADE	STA	SS01
	CLA	
FASE	STX	SS03
	STA	FLAG
	STB	ZM
	STE	
	STX	ZE
	LDX	SS03
	EAX*	0
	CLA	
	SKE	0,6
	BRU	FAN
	CLAB	
FAN	BRU	FLAD
	SKE	SS01
	BRU	FAX
	CLAB	
FAX	BRU	FLAC
	CXA	
	LDB	1,6
	STE	
	XXA	
	SUB	ZE
	SKG	=1
	BRU	FLAGM
	SKA	=100B
	LDA	=39
	XMA	SS01
	LDB	ZM
	RSH*	SS01
	XAB	
FLAC	SKN	FLAG



	BRU	FAS
	MRG	=777B
	SUB	1,6
	EOR	=777B
	XAB	
	SUC	0,6
FAS	BRU	FLAF
	COPY	A,E
	ADD	1,6
	XAB	
	ADC	0,6
FLAF	STE	
FLAOT	OV	
	BRU	FLAOS
	NOD	38
	SKA	=-1
	BRU	FLANZ
	CLX	
FLANZ	XXA	
	SKG	=-257
FLAOF	BRR	FLAOA
FLAOK	XXA	
FLAX	OV	
	BRU	FLOF
FLAX1	LDE	
	LDX	SS03
	BRU	XPBP
FLAOA	DATA	10000000B+FLAOK-1
FLAGM	CNA	
	SKA	=-100B
	LDA	=39
	LDX	0,6
	XXA	
	RSH	0,2
	COPY	B,E
FLAD	SKN	FLAG
	BRU	FLSS
	XMA	SS01
	XAB	

SET OVERFLOW DEVIUSLY

```

XMA      ZM
SUB      ZM
XAB
SUC      SS01
BRU      FLAF
FLSS    XAB
ADD      ZM
XAB
ADC      SS01
BRU      FLAF
FLA0S   RSH      1
EOR      =40000000B
BRX      FLAX
XXA
SKG      =255
BRU      FLA0K
BRR      FLA0A
FL0F    BRX      +=2
BRU      FL0J
CLEAR
LDX      SS03
BRU      XP0P
FL0J    EOR      =40000000B
RSH      39
EOR      =40000000B
EAX      255
BRR      FL0J1
FL0J1   ZR0      FLAX1=1,1
* FSB   FLOATING SUBTRACT
FSB     POPD     15500000B,1,1,0,1
$FSBE   STA      SS01
LDA     =1
BRU     FASE
* FNA   FLOATING CNA (BRS 21)
$FNA    LDX      SS03
BRM     FNAS
BRU     XP0P
FNAS    ZR0
STX     FNASX

```

SET OVERFLOW DEVI0USLY

	SKB	=1000B
	BRU	FLNA
	CNA	
	SKE	=0
	SKA	=17777777B
	BRR	FNAS
	STE	
	SKE	=40000000B
	BRU	FLCOM
	RCY	1
	BRX	FLCA
FLCOM	N0D	4
FLCA	XXA	
	SKG	=255
	SKG	=257
	BRR	FNA8FL
	BRU	FLNB1
FNA8FL	ZR0	FNA8F2=1,1
FNA8F2	XXA	
	E0R	=40000000B
	RSH	39
	E0R	=40000000B
	BRX	**3
	EAX	255
	BRU	FLNB
	CLEAR	
	BRU	FLNB
FLNB1	XXA	
	R0V	
FLNB	LDE	
FLMX	LDX	FNASX
	BRR	FNAS
FLNA	STE	
	XAB	
	CNA	
	XAB	
	E0R	=-1
	BRU	FLNB

\* ISC INTERNAL TO STRING CONVERSION

```

ISC2  N8P      1000B
ISC   P8PD     14000000B,1,1,0,1
*ISC1 STA      SS01
      LDA      ISC2
      BRU      SC1
* SIC  STRING TO INTERNAL CONVERSION
SIC2  N8P      1001B
SIC   P8PD     14100000B,1,1,0,1
*SIC1 STA      SS01
      LDA      SIC2
SC1   STA      BSX
      STB      SS02
      STX      SS03
      EAX*     0
      CXA
      ETR      ADMSK
      STA      UBE
      BRU      BSE
* FMP  FLOATING MULTIPLY
FMP   P8PD     15400000B,1,1,0,1
*FMPE STX      SS03
      STA      SS01
      STE
      STX      ZE
      BAC
      LDX      SS03
      EAX*     0
      LRSB     2
      MUL      0,6
      STA      ZM
      LDA      1,6
      CXB
      COPY     AX,A,E
      XXA
      ADM      ZE
      COPY     XA,BX,B
      LRSB     2
      MUL      SS01
      ADD      ZM

```

	MUL	#2
	STB	ZM
	XMA	SS01
	MUL	0,6
	XAB	
	ADD	ZM
	XAB	
	ADC	SS01
	LDX	ZE
	SKA	#37777777B
	BRU	FLEND
	SKB	#1000B
	BRU	FLEND
	SKE	#40000000B
	BRU	FLND2
	RCY	1
FLEND	BRX	FLEND
	N8D	4
	R8V	
	XXA	
	SKG	#255
	SKG	#257
	BRR	FLA8A
	XXA	
FLND2	LDE	
	LDX	SS03
	BRU	XP8P
* FDV	FL0ATING DIVIDE	
FDV	P8PD	15300000B,1,1,0,1
*FDVE	STX	SS03
	STA	SS01
	STE	
	STX	ZE
	LDX	SS03
	EAX*	0
	RSH	2
	DIV	0,6
	8VT	

	BRU	FDV8
	STA	ZM
	BAC	
	RSH	1
	STA	SS01
	LDB	1,6
	CXA	
	STE	
	XXA	
	CNA	
	ADD	#2
	ADM	ZE
	BAC	
	RCY	2
	CNA	
	MUL	ZM
	ADD	SS01
	DIV	0,6
	MUL	#2
	ADD	ZM
	LDX	ZE
	SKA	#1
	BRU	FLND
	BRU	FLND2
FDV8	LDA	SS01
	E8R	0,6
	BRU	FL8J

\* STORAGE  
ZE ZR0  
ZM ZR0  
FLAG ZR0  
FNASX ZR0

\* PRINT UNSIGNED INTEGER  
\* BRS 36  
\* INPUT: A=NUMBER. B=RADIX. X=FILE NO.  
\*

\$OUTNUM STA OUTN02  
STB OUTN03

	STX	0UTN05
	CBA	
	SKG	#1
	BRU	TRAP
	CLA	
0UTN06	STA	0UTN04
	LDA	0UTN02
0UTN07	STA	0UTN01
	LRSH	23
	DIV	0UTN03
	SKE	0UTN04
	BRU	0UTN07
	CBA	
	ADD	#20B
	SCI0	0UTN05
	LDA	0UTN01
	SKE	0UTN02
	BRU	0UTN06
	LDB	0UTN03
	LDX	0UTN05
	BRU	EP0PX

\*  
 \* DEMAND SIGNED INTEGER  
 \* BRS 38  
 \* INPUT: B=RADIX. X=FILE NO.  
 \* OUTPUT: A=NUMBER  
 \*

*GETNUM	STX	GETN01
	BAC	
	STA	GETN02
	STB	GETN03
	STB	GETN07
GETNM	SCI0	GETN01
	SKE	#13B PLUS
	BRU	*+2
	BRU	GETN06
	SKE	#15B MINUS
	IF	V6
	BRU	GETNM1

```

      ELSF      1
      BRU      GETN06+1
      ENDF
      LDA      =-1
      STA      GETN07
GETN06 SCI0    GETN01
      SUB      =20B
      SKG      =-1
      BRU      GETN04
      SKG      GETN02
      BRU      GETN05
GETN04 ADD      =20B
      CAB
      LDA      GETN03
      SKN      GETN07
      BRU      EP0PX
      CNA
      BRU      EP0PX
GETN05 SKE      GETN02
      BRU      **2
      BRU      GETN04
      XMA      GETN03
      MUL      GETN02
      LSH      23
      ADM      GETN03
      BRU      GETN06

      IF      V6
GETNM1 SKE      =0
      BRU      **2
      BRU      GETNM
      SKE      =155B
      BRU      GETN06+1
      BRU      GETNM
      ENDF

```

IGNORE LEADING SPACES

IGNORE LEADING C.R.

```

*
* FIX AND FL0AT
*

```

```

$FFIX SKA X4

```



	BRM	FNAS
	SKD	#23
	BRU	FIXBIG
	COPY	B,E
	RSH	0,2
FIXEND	SKN	SS01
	BRU	FEND1
	SKB	#-1
	BRU	**3
	CNA	
	BRU	FEND1
	XAB	
	EOR	#-1
FEND1	LDX	SS03
	BRU	XPBP
FIXBIG	COPY	B,E
	LSH	0,2
	ETR	XX
	BRU	FIXEND
*		
\$FFLT	CLB	
	SKE	#0
	BRU	**2
	BRU	XPBP
	LDX	#23
	NBD	48
	LDE	
	LDX	SS03
	BRU	XPBP
	END	

DOUBLE PRECISION NEGATE BEGINS

DOUBLE PRECISION NEGATE ENDS

```

3SET IDENT 6/30/67
* MONITOR MODULE 0
  ENTRY SETSA
*
*
* !IOPSET! 9/27/65
*
* THIS ROUTINE INITIALIZES THE FILE CONTROL TABLE AND ALL
* NON-TELETYPE I/O.
*
IOPSET ZR0; IF =V1; BRM LCLRS; ENDF; LDA =10000003B; LDX =-NFILE+3
ISET1 ADD =1; STA EFA,2; BRX ISET1; LDA =3; STA FFLST
  IF V3
  LDA =2000B; STA LOGFIL (LOG CONSOLE=0)
  ELSF 1
  LDA =2013B; STA LOGFIL (LOG CONSOLE = 11)
  ENDF
  LDX =LADIU; LDB =-1; STB EFA1; STB LOGFLG
  IF V6; STB SWCIT; ENDF
ISET2 STB EDIU,2; BRX ISET2; LDX =NNDEV
ISET3 LDA EBUFS,2; SKA =40000B; BRU =+2; STB ADIU,2; BRX ISET3
ISET5 CXA; SKE =NTAPE; BRU =+2; BRU ISET4
  STB TJN0,2; CLA; STA TSTATE,2; STB TAPREL,2
  STA TDIBA,2; STA TDFN0,2; EAX 1,2; BRU ISET5
ISET4 CLA; STA BLK31; ARMI AIRWD; BRR IOPSET
*
*
* !DRMSET!
* THIS ROUTINE INITIALIZES THE DISC I/O.
*
DRMSET BSS 0
DRMRST ZR0; LDA =-1; STA NDCL; STA IDCL1; STA DTXS1
  STA IDMRST; CLA; STA IDRS
  LDA =DRQ; STA EDCL; STA IDCL; BRR DRMRST
*
*
  IF V1
* !RADSET!
*

```

\* THIS ROUTINE INITIALIZES THE RAD I/O

```
*  
RADSET ZR0; LDA =-1; STA NRCL; STA RTXS1  
      LDA =RADQ; STA ERCL; STA RLIS; BRR RADSET  
      ENDF
```

\* '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 ZR0 0; STA SRFLG  
      LDA =LTTYM1; STA T; COPY A,X; STA TTYBUF,2; EAX 1,2; SKR T; BRU *-3  
      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 ADMSK; STA TTYASG,2  
TSET3  CLA; STA TIS2,2; STA TTYFLG,2; STA TTYBRK,2; STA TTYTIM,2  
      LDA TOS5,2; ADD TTYLNG,2; ADD =1; COPY AB,XA  
      IF =V5  
      MRG =100000B; STA TTCH; TTYS; PBT TTCH  
      ENDF  
      LDA =-1; STA TOS2,2; STA TOS3,2  
      LDA TTYLNG,2; CNA  
      XXB; STA =1,2; XXB; EAX 1,2; SKR T; BRU TSET1  
      LDA =-1; STA ATIS2; LDA =ATTBUF; STA ATIS4; STA ATIS5  
      SKN SRFLG; BRR TTYSET; LDX =-NJ0B+2; STX ETTN0  
TSET2  CXA; ADD =NJ0B; STA ETTN0,2; BRX TSET2  
      IF =V5  
      LDA =1; STA FULST; LDA =100000B; STA TTCH  
      TTYS; PBT TTCH; MIN TTCH; TTYS; PBT TTCH  
      LDA =20001B; ADM TTCH; TTYS; PBT TTCH; MIN TTCH  
      SKR TSET4; BRU *-4  
      BRR TTYSET  
      ELSEF 1  
      LDA =1; STA FULST; BRR TTYSET  
      ENDF  
SRFLG ZR0 0
```

```

TTCH   ZR0 0
TSET4  DATA NTTY=3
*
*      SYSTEM INITIALIZE ROUTINE
*
SETSA  LDA X4; STA REAL; BRM SSET
        BRM DRMSET; BRM RADSET; CKN; EIR; BRU PACG01

* RESET SYSTEM
SSET   ZR0; LDX =-77B; LDA CLINTC; STA 100B,2; BRX *-1
        STA SETSET
        LDX =-NPAC*NPPAR; STX CLOCK3; CLB
        LDA =SMT; STA ADRSMT
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 =-NPAC*NPPAR; STX PACPTR; STX PUPACP; LDX =-NPPAR
SET2   LDA EPUPD,2; STA PACT1,2; BRX SET2
        LDA PACPTR; LDX =QTI; BRM QPUT
        LDX =-NPUQ*4
SET3   CXA; ADD =EPUCT3; STA EPUCT,2
        EAX 3,6; BRX SET3; CLB; STB EPUCM3
        LDA =PUCT; STA FPULST; LDA =PUBPTR; STA PUBPTR; STA PUEPTR
        LDA =25; LDX =-NSMEM; STA ESRMC,2; BRX *-1
        LDA =-1
        IF      V3
        STA DMIN
        LDX =-NTTY; STA EWERIS,2; BRX *-1
        ENDF
        LDX =NSMEM-NMEM; STA ERMCM,2; BRX *-1

BRM RSAM; LDX =-NMEM; CLA
STA ERMT,2; STA ERMA,2; BRX *-2
LDX =-NJ0B; CLA; STA EPMT,2; BRX *-1
IF V5; STA N0ACT; ENDF

```

```

SKN M48K; BRU SET4
LDX =8; LDA =25; STA ERMC,2; BRX *-1
LDX =8; LDA X6; STA ERMT,2; BRX *-1
SET4 LDA BST; LDX =100B+NP0P; STA 200B=NP0P,2; BRX *-1
STA 177B; LDX =WDTB=WDTE
LDA WDTE,2; STA* WDTE+1,2; BRX **1; BRX *-3
IF V6; LDA =ACST; STA PACST; ENDF
LDA =DAW; ADD DISCA; ADD DRC; LDB =CAW
LDX =4000B; BRM GDC; BRU **6
LDA =DADSC; ADD DISCA; ADD DRC; LDB =CADSC; LDX =4000B
BRM GDC; BRU **6
*SET5 LDA =DAEXEC; ADD DISCA; ADD DRC; LDB =CAEXEC; LDX =4000B
BRM GDC; BRU **6
LDA =RAEXEC; LDB =CAEXEC; LDX =4000B; BRM GCR; BRU **4
LDA =40B; ADM =DAEXEC; ADM =RAEXEC; SKR EXP; BRU SET5
LDB =25; LDA SMTAD; ADD X4; LDX =8; ADD =56B; STA RMT,2
STB RMC,2; EAX 1,2; ADD =1; STA RMT,2; STB RMC,2
BRM I0PSET; LDA =-1; BRM TTYSET; BRR SSET

```

```

SMTAD DATA SMT
CAEXEC EQU 54000B CORE ADDRESS FOR EXEC
EXP DATA 3 NO. OF EXEC PAGES =1.

```

\*  
\*

```

*GCR ZR0;
STB GCRL; STA GCRL1; LDA GCRL; ETR ADMSK; RSH 14
CXA; ETR =77777B; LCY 14; STA GCRL2; LSH 19
LDA GCRL; ETR =140000B; CBX; RSH 14
CXB; LSH 5; MRG =614200B; STA GCRL3
RSR; BRU **1; ALR; P0T GCRL1
E0D* 10000B; EXU GCRL3; P0T GCRL2
WRF
RSR; BRU **1; CETE; BRU GCR2; RSE; BRU GCR2

```

```

MIN GCR
GCR2 BRR GCR

```

\*  
\*

```

PUPD DATA 0,PUG0,0,07010203B,04050600B

```

EPUPD DATA 0,PACDMB,47700000B,0,0  
EQU \*

WDTB EQU \*

WDPII BRM TRAPI; 0 40B  
WDIMI BRM TRAPM; 0 41B  
WDRMT BRM TRAPR; 0 43B  
WDUMT BRM TRAPT; 0 44B  
WD31 BRM INT31; 0 31B  
WD33 BRM INT33; 0 33B  
WDPWNI BRM PWNI; 0 36B  
WDPWFI BRM PWFII; 0 37B  
WDCP BRM CPUP; 0 56B  
WDIOP BRM IOP; 0 57B  
BRM RADI; 0 64B  
WDIDM BRM RADI2; 0 65B  
CLOCK1 BRM CLINT; 0 74B  
CLOCK2 SKR CLOCK3; 0 75B  
WDTI BRM TII; 0 200B  
WDT0 BRM T0I; 0 201B  
WDTN BRM TNI; 0 202B  
WDTF BRM TFI; 0 203B  
BRJ 204B; 0 204B  
BRM ATII; 0 205B  
BRJ 206B; 0 206B  
IF =V1  
WDATI BRM ATII; 0 207B  
ENDIF

WDTE EQU \*

\*  
\*

WRITE LDA =34000B; LDB =2100B; LDX =4000B; BRS 105; BRS 10

END

3SYMS IDENT 7/01/67

ENTRY VERS  
IF V1  
ENTRY DFAIL,DRC  
ENTRY XBERR,FDERR,TNICTR,TFICTR,ARM0T,N0VP,ECNTR  
ENTRY TNIAC,TFIAC,EXTPU,NB112,D0WN  
ENTRY DSWAPC,SWAPC,PAGES,TIICTR,T0ICTR  
ENTRY CHNERR,RCERR,RADERR,RUERR,I2ER,CPUPC,I0PC  
ENDF  
ENTRY TNERR,TURERR,TRERR,TWERR,DRMERR,DRUERR,WERIS  
ENTRY TAPREL,BRSTV,PWFL,M48K  
ENTRY CTP,TSTATE,MTFL,TJN0,WR4X3,XWR4X3  
ENTRY PACPTR,J0B,UTTY,TJ0B,PMTJ0B  
ENTRY PMA,WERIS  
ENTRY STIME,REAL,SETTB,ETTB  
ENTRY PMTP,RL3,ADRSMT  
IF V3  
ENTRY CPARW,AUNN,EAUNN,DMIN,ACST,PACST  
ENTRY XPB,EXEC1  
ENTRY XCLEAN  
ENDF

\*  
\* CONSTANTS AND TABLES  
\*

PACPTR ZR0  
J0B ZR0  
UTTY ZR0  
XPB ZR0 0 PB POINTER  
EXEC1 ZR0 0 NEGATIVE FOR SUBSYSTEM STATUS  
TJ0B ZR0 0 ELAPSED TIME POINTER  
PMTJ0B ZR0 0 POINTER TO CURRENT PMT  
ADRSMT ZR0 0 ADDRESS OF SMT  
M48K DATA =1 =1 FOR 48K. 0 FOR 64K.  
DRC DATA 20000B CONSTANT FOR LOCATING EXEC ON DISC

VERS ASC '12.1'  
ASC '000'

\* JOB INDEXED TABLES

PMA BSS NJOB MACHINE SIZE INFO  
PMTB BSS NJOB POINTER TO FIRST PMT ENTRY  
\*EPMTP EQU PMTB+NJOB  
RL3 BSS NJOB MONITOR MAP, SHOULD CONTAIN ONLY TS BLOCK MAP  
\*ERL3 BSS 0  
WERIS BSS NTTB USER ASSIGNED TO TTY  
\*EWERIS EQU \*

IF V3

\*

\* EXEC PARAMETERS

CPARW BSS NJOB USER'S CONTROL PARAMETERS,  
\* BITS: 0=SUBSYSTEM, 1=DEVICE, 2=OPERATOR, 3=SYSTEM  
AUNN BSS NJOB ACCT/USER NOS.

EAUNN EQU \*

ACST BSS 64

\*EACST EQU \*

\* EXEC SWITCHES, SET BY BRS BE+13

\*SWEX BSS 0

\*SWLET DATA 0 LETTER ROUTINE SWITCH

\*PACST ZR0 ACST

\*NOVP ZR0 0

NEW OVERFLOW POINTER

\*ACA ZR0 0

ACCOUNTING AREA

DOWN DATA 0

SET TO -1 IF SYSTEM GOING DOWN

\*NOACT DATA 0

ACCOUNTING SWITCH, -1 FOR ACCOUNTING.

\*SWMBL DATA =1

MAIL BOX LIST BUSY

\*SWMBM DATA =1

MAIL BOX MESSAGES BUSY

\*ENMSK DATA 77377777B

SUBSYSTEM ENABLE MASK.

\*SWFD DATA 0

FILE DIR. BUSY FOR USER SPECIFIED.

ENDF

\* SYSTEM PARAMETERS

ECNTR DATA =27

NO. OF PARAMETERS



TNERR	ZR0	0		NOISE RECORDS READ
TURERR	ZR0	0		PERMANENT READ ERRORS
TRERR	ZR0	0		TOTAL READ ERRORS
TWERR	ZR0	0		TOTAL WRITE ERRORS
PWFL	ZR0	0		NO. OF POWER FAILURES.
CHNERR	ZR0	0		TOTAL DISC CHANNEL ERRORS.
DRMERR	ZR0	0		TOTAL DISC ERRORS
DRUERR	ZR0	0		UNCORRECTABLE DISC ERRORS
DFAIL	ZR0	0		DISC FAILURES FOUND BY SOFTWARE CKSUM.
RCERR	ZR0	0		RAD CHANNEL ERRORS.
RADERR	ZR0	0		RAD ERRORS.
RUERR	ZR0	0		UNCORRECTABLE RAD ERRORS.
I2ER	ZR0	0		I2 RAD ERRORS WITH NO I1.
CPUPC	ZR0	0		CPU PARITY COUNT
I0PC	ZR0	0		I0 PARITY COUNT
REAL	ZR0	0		NO. OF CLOCK TICKS
STIME	ZR0	0		TIME IN SYSTEM MODE.
DSWAPC	ZR0	0		NO. OF TIMES SWAP CALLS RST
SWAPC	ZR0	0		SWAP COUNT
PAGES	ZR0	0		NO. OF PAGES SWAPPED
ARM0T	ZR0	0		DISC ARM MOTION
TIICTR	ZR0	0		TTY INPUT INTERRUPTS
TOICTR	ZR0	0		TTY OUTPUT INTERRUPTS
XBERR	ZR0	0		NO. OF X-BLOCK ERRORS WHILE SETTING MAP
FDERR	ZR0	0	NO. OF FILE	DIR. ERRORS WHILE SETTING DISC BIT MAP
TNICTR	ZR0	0		TTY 0N INTERRUPTS
TFICTR	ZR0	0		TTY 0FF INTERRUPTS
TNIAC	ZR0	0		0N INTERRUPT ACCEPTED
TFIAC	ZR0	0		0FF INTERRUPT ACCEPTED
EXTPU	ZR0	0		EXTRA PU ENTRIES AT BRS 112
NB112	ZR0	0		NO. OF TIMES THRU BRS 112.
*				
XCLEAN	DATA	0		SWITCH USED IN SDBM, SEE :SDBM12:+2
CTP	DATA	2,2		CURRENT MAG TAPE POSITION
TSTATE	DATA	0,0		STATE OF MAGTAPE (0=FREE,1=READ,2=WRITE)
TAPREL	DATA	0,0		HIGHER PRIORITY TAPE REQUEST FLAG
MTFL	DATA	0,0		LENGTH OF TAPE FILE IN RECORDS
TJNB	DATA	0,0		JOB WITH TAPE LOCKED OR =1
*TDIBA	DATA	0,0		TAPE DRUM FILE BLOCK ADDRESS

```

*TDFN0 DATA 0,0 TAPE DRUM FILE NUMBER
*WTDFLG ZR0 0 WRITE TAPE ON DRUM IF <0
*LOGFIL ZR0 0 MESSAGE LOGGING FILE
*LOGFLG ZR0 0 MESSAGE BEING LOGGED FLAG
*NFENFLG ZR0 0 ALLOW NO NEW USERS IF +
WR4X3 ZR0 0 PTR. TO STRING BUFFER, PAGE 7, SET BY 'SIT'
XWR4X3 ZR0 0 PTR. TO STRING BUFFER, PAGE 0, SET BY 'SIT'

```

```

* BRS MODULE TRANSFER VECTOR
BRSTV DATA 10000003B,12000000B

```

\* SPS CONTROL TABLES FOR EXEC

```

*CRTAB ZR0 CHT
ZR0 SYSEHT
*CTABIT DATA CIT,0,0
*CTAB DATA CHT,ECHT,0,CSS,ECSS,0
*SYSIT DATA SIT,0,SYST
*SYSTAB DATA SYST,SYSEHT,0,SYSSS,SYSESS,0

```

\* TIMING TABLES

```

DMIN IF V1
ZR0 0 SYS. START-UP DATE/TIME IN MINS.
ELSF 1
REAL ZR0
ENDF
SETTB ZR0 0 SYSTEM ETTB ENTRY
ETTB BSS NJOB COMPUTE TIME COUNTER

```

END

```

3TS IDENT 6/30/67
DELSYM
10RG EQU *
LOCORG MACRO D;20RG EQU *-10RG; BSS D(1)-20RG; ENDM
USYM MACRO D;D(1) BSS 0;D(2) EXT D(1)+X;20RG NARG
IF 20RG=2; BSS D(3); ELSF 1; ZR0; ENDF; ENDM
XSYM MACRO D;D(1) EQU **X;20RG NARG
IF 20RG=1; BSS D(2); ELSF 1; ZR0; ENDF; ENDM
$X EQU 34000B

```

\* TEMP STORAGE FOR CLASS 2 BR\$'S

```
LOCORG 2
```

```

XSYM SBR$RT
XSYM GSIN1 BRS 33
XSYM GSIN2
XSYM GSIN3
XSYM STR01 BRS 34,35
XSYM STR02
XSYM STR021
XSYM STR03
XSYM OUTN01 BRS 36
XSYM OUTN02
XSYM OUTN03
XSYM OUTN04
XSYM OUTN05
XSYM GETN01 BRS 38
XSYM GETN02
XSYM GETN03
XSYM GETN07
USYM XWR1,WR1 BRS 37
USYM XWR2,WR2
USYM XWR3,WR3
USYM XWR31,WR31

```

```

IF V3
XWR42 USYM XWR4,WR4,30
EXT XWR4+2

```

```
ELSF      1
XSYM     WR4,30
ENDF
XSYM     WR5
XSYM     WR6
XSYM     WR7
XSYM     WR8
XSYM     WR9
XSYM     WR10
XSYM     WR11
XSYM     WR12
XSYM     WR13
```

```
LOCORG 225B
```

```
USYM     UBRST,UPL
USYM     UA,UBA
USYM     UB,UBB
USYM     UX,UBX
USYM     CMRL1,UBRL1
USYM     CMRL2,UBRL2
USYM     UE,UBE
$UIDX   ZR0      0
EXRL    EXT     EXRL1
$EXRL1  ZR0      0
$EXRL2  ZR0      0
        USYM     PRGRL1,UPRRL1
        USYM     PRGRL2,UPRRL2
PRGRL   EXT     PRGRL1
$SSRL1  ZR0      0
$SSRL2  ZR0      0
SSRL    EXT     SSRL1
$FDCTL  DATA   FDHT,EFDHT,0,(R)FDSS=1,(R)EFDSS=1,0
FDCTL2  EXT     FDCTL+2
FDCTLC  EXT     FDCTL+3
```

FDCTLE	EXT	FDCTL+4
LFDSS	EXT	EFDSS=FDSS+EFDSS=FDSS+EFDSS=FDSS
LGFD	EXT	FDCTL=EFDSS
FDHTM1	EXT	FDHT=1
EFDHM1	EXT	EFDHT=1
\$FDHT	BSS	144
\$EFDHT	NOP	0

	IF	V3
\$DUMHT	BSS	3
BCHT	BSS	144
	BSS	4
SZH	EXT	BCHT=FDHT
	ENDF	

\$FDSS	BSS	120
\$EFDSS	NOP	0
TFDSS	EXT	EFDSS=1
\$CHFD	ZR0	0
\$INFIL	ZR0	0
ECRFIL	EXT	INFIL
\$BUTFIL	ZR0	0
ECDFIL	EXT	BUTFIL
\$MFD2	ZR0	0
\$MFD21	ZR0	0
\$UTN0	ZR0	0
\$RFDFLG	ZR0	0

=1 FOR READING CFD ON ENTER, 0 FOR READING IT ON LOGOUT

*		
\$RFD1	ZR0	0
\$CFI1	ZR0	0
\$CFI2	ZR0	0
\$PDN1	ZR0	0
\$SAVELL	ZR0	0
\$SAVEFL	ZR0	0
\$SAVE0R	ZR0	0
\$SAVESL	ZR0	0
\$COPY1	ZR0	0
\$COPY2	ZR0	0
\$COPY4	ZR0	0

\$COPY7	ZR0	0
\$COPYT	ZR0	0
\$CR3	ZR0	0
	USYM	WAIT,UWAIT
	USYM	IDIOT,UIDIOT
	USYM	CIN,UCIN
	USYM	C0UT,UC0UT
\$UNPTR	BSS	2
UNPTR1	EXT	UNPTR+1
	USYM	UN0,UUN0
	USYM	EXFLAG,UEXFLG
\$RELST4	ZR0	0
\$OK1	ZR0	0
\$EXECL	BSS	7
EXECL6	EXT	EXECL+6

\* PANIC TABLES FOR EXEC FORKS

\$EXSL	ZR0	0
\$EXSA	ZR0	0
\$EXSB	ZR0	0
\$EXSX	ZR0	0
\$EXSR1	ZR0	0
\$EXSR2	ZR0	0
\$EXSM	ZR0	0
\$SSL	ZR0	0
\$SSA	ZR0	0
\$SSB	ZR0	0
\$SSX	ZR0	0
\$SSR1	ZR0	0
\$SSR2	ZR0	0
\$SSM	ZR0	0

\* WORKING SPACE FOR EXEC BR'S

\$GFN1	ZR0	0
--------	-----	---

\$GBFN1	ZR0	0
\$GBFN2	ZR0	0
\$GBFN3	ZR0	0
\$GBFNFL	ZR0	0
\$GBFNFT	ZR0	0
\$GFNRET	ZR0	0
\$GBFNS	BSS	2
GBFNS1	EXT	GBFNS+1
\$NTRY1	ZR0	0
\$TMSG1	ZR0	0
\$FN1	ZR0	0
\$FN2	ZR0	0
\$FNPTR	BSS	2
FNPTR1	EXT	FNPTR+1
IOPTR	EXT	FNPTR
IOPTR1	EXT	FNPTR+1
\$INSFN1	ZR0	0
\$LFDC1	ZR0	0
\$DF1	ZR0	0
\$DF2	ZR0	0
\$CF1	ZR0	0
\$KILL1	ZR0	0
\$KILL2	ZR0	0
\$KILL3	ZR0	0
\$GETPL	ZR0	0
\$GETSTL	ZR0	0
\$XM1	BSS	2
XM11	EXT	XM1+1
\$DEL1	ZR0	0
\$DEL2	ZR0	0
\$DEL3	ZR0	0
\$DEL4	ZR0	0
\$FDEL1	ZR0	0
\$FDEL2	ZR0	0
\$FDEL3	ZR0	0
\$FDEL4	ZR0	0
\$FDEL41	ZR0	0
\$MSZ2	ZR0	0
\$PRMA1	ZR0	0

CSYS2	EXT	PRMA1
\$PRMA2	ZR0	0
CDEF2	EXT	PTR1
CDEF4	EXT	PTR2
CDEF5	EXT	PTR5
\$PTR1	ZR0	0
\$PTR2	ZR0	0
\$PTSTAT	ZR0	0
\$PTR3	ZR0	0
\$PTR4	ZR0	0
\$PTR5	ZR0	0
\$PMTFL	ZR0	0
\$MAGTFT	ZR0	0
\$MAGT	ZR0	0
\$MTFL1	ZR0	0
\$TEND	ZR0	0
\$MTPFN	ZR0	0
\$UN01	ZR0	0
\$DUMP1	ZR0	0
\$ECDMP6	ZR0	0
\$ECRV6	ZR0	0
\$REC2	ZR0	0
\$REC3	ZR0	0
\$ERB1	ZR0	0
\$ERB2	ZR0	0
\$ERB3	ZR0	0
\$DRSW1	ZR0	0
\$DRSW2	ZR0	0
\$DRSW3	ZR0	0
\$DRSW4	ZR0	0
\$DRSW9	ZR0	0
\$SYSTL	ZR0	0
\$SYSRRL	ZR0	0
\$RSYS1	ZR0	0
\$RSYS2	ZR0	0
\$GSYS1	ZR0	0
\$GBI01	ZR0	0
\$GBI0PT	ZR0	0
\$GBI0FN	ZR0	0

STATUS OF USERS TAPE REQUEST (DURING TAPEWAIT)



\$GBIOSW ZR0	0
\$GBIO2 ZR0	0
\$GBIOCR ZR0	0
\$GBIOC2 ZR0	0
\$GBIO4 ZR0	0
\$GET1 ZR0	0
IOSP1 EXT	GET1
\$GET2 ZR0	0
IOSP2 EXT	GET2
\$GET3 ZR0	0
IOSP3 EXT	GET3
\$GET4 ZR0	0
IOSP4 EXT	GET4
\$GET5 ZR0	0
IOSP5 EXT	GET5
\$GET6 ZR0	0
\$GETFN ZR0	0
IOWFILE EXT	GETFN
\$SUSR1 ZR0	0
\$SUSR2 ZR0	0
\$SUSR3 ZR0	0
\$SRSU1 ZR0	0
\$SRSU20 ZR0	0
\$SRSU21 ZR0	0
\$SRSU3 ZR0	0
\$SRSU4 ZR0	0
\$UREAL ZR0	0
\$TIME1 ZR0	0
\$TIME2 ZR0	0
\$TIME3 ZR0	0
\$TFILE ZR0	
\$TWMSG ZR0	0
\$DGCNT ZR0	
\$SPACES ZR0	
\$OVFL ZR0	
\$ERRFLG ZR0	
\$VDGTS ZR0	
\$I0D ZR0	
\$I0W ZR0	

\$IOFMT ZR0  
\$DEXP ZR0  
\$IODEXP ZR0  
\$SIGN ZR0  
\$NCONV ZR0  
\$RLI01 BSS  
\$RLI02 ZR0  
\$RLI04 ZR0  
\$RLI05 ZR0  
\$RLI06 ZR0  
\$RLI07 ZR0  
\$RNDFLG ZR0  
\$EXPFLG ZR0  
\$DFLG ZR0  
\$PLUS ZR0  
\$MINUS ZR0  
\$PERIOD ZR0  
\$EEE ZR0  
\$ENDCHR ZR0  
\$FFL ZR0  
\$FFA ZR0  
\$FFX ZR0  
\$OUTSX ZR0  
\$RNDX ZR0  
\$RLITEX ZR0  
\$FIOW ZR0  
\$A1 ZR0  
\$A2 ZR0  
\$A3 ZR0  
\$IOLINK ZR0  
\$ERRNUM ZR0

2

IF  
\$RLV1 ZR0  
\$TLV1 ZR0  
\$TLV11 ZR0  
\$RLV2 ZR0  
\$TLV2 ZR0  
\$TLV21 ZR0

V3

```

$TLV22 ZR0
$TLV23 ZR0
$RLV3 ZR0
$TLV3 ZR0
$RLV4 ZR0
$TLV4 ZR0
$SWOFF ZR0
$SWTM ZR0
$OVFP ZR0
$PJ BSS 2
$PJ2 BSS 2
$EPJ EQU *
$MTIME ZR0 0 USER'S VALID TIME PARAMETER
$CTIME ZR0 0 ACCUM. COMPUTER TIME.
$RTIME ZR0 0 ACCUM. REAL TIME
$LETP ZR0 0 LETTER POINTER

```

```

$DTEMP EQU FBADR+2-34000B
$DTEMPE EQU DTEMP+64
$DTIME EQU DTEMP+127
$DTIME1 EQU DTEMP+128
$LETEN EQU DTEMP+256

```

ENDF

END

```

3TTY IDENT 6/30/67
* ENTRY POINTS
ENTRY TTYEMG,TTYASG,TTYTBL,ETO,TTN0,FULST
ENTRY TIS2,TIS4,TIS5,T0S2,T0S3,T0S4,T0S5
ENTRY TTYFLG,TTYBRK,TTYTIM,TTYLNG
ENTRY TTYBUF
ENTRY ATIS2,ATIS4,ATIS5,ATTBUF
ENTRY TS0FF,TII,T0I,ATII,TS0NM,TS0NI
ENTRY FTCI,FTC0
ENTRY CIB,C0B,SKI,D0B,CET,RDET
ENTRY SET8P,CLR8P,GATX
IF V1
ENTRY CARRY
ENTRY TFI,TNI,TIDMS,CRSW,TREC,TIPIX,TTY0N
ELSF 1
ENTRY LTIS2,LTIS4,LT0S2,LT0S4,LTTBRK,LTIS5,LT0S5,LF1,LF0
ENDF

```

```

* TABLES INDEXED BY TELETYPE NUMBER
TIS2 BSS NTTB;* CHAR COUNT FOR TTY INPUT BUFFER
TIS4 BSS NTTB;* INTERRUPT INPUT BUFFER POINTER
TIS5 BSS NTTB;* TIS INPUT BUFFER POINTER
T0S2 BSS NTTB;* CHAR COUNT FOR TTY OUTPUT BUFFER, -1 = INACTIVE
T0S3 BSS NTTB;* MULTIPLE BLANK COUNTER. X2 IF NEXT CHAR IS COUNT
T0S4 BSS NTTB;* INTERRUPT OUTPUT BUFFER POINTER
T0S5 BSS NTTB;* T0S OUTPUT BUFFER POINTER
TTYTBL BSS NTTB;* ECHO TABLE AND ASSORTED FLAGS
$ETTYTB EQU TTYTBL+NTTY
TTYFLG BSS NTTB;* + IF BUFFER FULL OR TELETYPE UNUSED
TTYBRK BSS NTTB;* =1 IF WAITING FOR BREAK CHAR, + IF NOT
TTYASG BSS NTTB;* TTY OWNERSHIP STATUS
* FORK TO TERMINATE IN CASE OF RUBOUT.
TTYREC BSS NTTB;* STATE CHANGE RECORD
IF -V8
RILCW BSS NTTB;* REL. INPUT LCW
$ERILCW EQU RILCW+NTTY
R0LCW BSS NTTB;* REL. OUTPUT LCW
$ER0LCW EQU R0LCW+NTTY
AILCW BSS NTTB;* ABS. INPUT LCW
$EAILCW EQU AILCW+NTTY

```

```

A0LCW BSS NTTB;* ABS. 0UTPUT LCW
$EA0LCW EQU A0LCW+NTTY
      ENDF
TTYTIM BSS NTTB;* TIME AT LAST RUBOUT
TIIS5 BSS      NTTB  PREVIOUS CHAR. AND FLAGS FOR SPEC. CR/LF CNT.
*      NEG. IF AT LEFT EDGE. X2=N0 CR SENT.
* JOB = TELETYPE INDEX
TTN0 BSS NJ0B
$ETT0 EQU TTN0+NJ0B-1
FULST ZR0
* TELETYPE ECH0 TABLES
* ECH0 PARAMETERS: A(1)=8 BIT BYTE, A(2)=N0. 0F REPEATS, A(3)=INCREMENT
* 8 BIT BYTE: BREAK IF BIT 0=1, N0 ECH0 IF CHAR.=1
NECHR EQU 0
ECHRWD EQU 0
ET0 BSS 0;* ECH0 EVERYTHING, BREAK 0N EVERYTHING
      ECHR 0
      ECHB 201,6,0
      ECHR 207
      ECHB 201,2,0
      ECHR 212
      ECHB 201,2,0
      ECHR 215
      ECHB 201,18,0
      ECHB 240,64,1
ET1 BSS 0;* ECH0 EVERYTHING, BREAK 0N PUNCS AND CTRLS
      ECHR 0
      ECHB 201,6,0
      ECHR 207
      ECHB 201,2,0
      ECHR 212
      ECHB 201,2,0
      ECHR 215
      ECHB 201,18,0
      ECHR 040
      ECHB 241,15,1
      ECHB 060,10,1
      ECHB 272,7,1
      ECHB 101,26,1

```

```

ET2      ECHB      333,5,1
        BSS 0;* ECH0 EVERYTHING, BREAK 0N CTRLS 0NLY
        ECHR      0
        ECHB      201,6,0
        ECHR      207
        ECHB      201,2,0
        ECHR      212
        ECHB      201,2,0
        ECHR      215
        ECHB      201,18,0
        ECHR      040
        ECHR      241
        ECHB      042,62,1
ET3      BSS 0;* ECH0 N0THING, BREAK 0N EVERYTHING
        ECHR      0
        ECHB      201,95,0

```

\* TELETYPE BUFFERS

TTYEMG DATA TTYEWM

TTYLNG RPT NTTY, 0 NTTYC, ENDR

LTTY EQU NTTY\*NTTYC+NTTY+2

LTTYM1 EXT LTTY-1

ITTBLEXT 40000000B+AMB+APB+ETO

\*

\* INPUT AND ECH0 CHAR'IS: BIT 0=0

\* 0UTPUT CHAR'IS: BIT 0=1

TTYBUF BSS LTTY;\* TELETYPE I/O BUFFER

\*

\* !TII! 1/30/66 = 49 CYCLES

\*

\* THIS ROUTINE PROCESSES TELETYPE INPUT INTERRUPTS, AND PUTS THE

\* CHARACTERS INTO A BUFFER FOR THE 205 INTERRUPT TO PROCESS

\*

TIIA ZR0;\* SAVE (A)

IF =ARMF

TIIIB ZR0

ENDF

TIIIX ZR0;\* SAVE (X)

TIIIS1 ZR0;\* TELETYPE 'PIN' WORD

TIIIS2 ZR0;\* INPUT CHAR AND TELETYPE N0.

```

TIIS4  ZR0;* ECH0
      IF      V6
TIISV  ZR0      0          SAVED CHAR FOR SPEC. CR/LF CONTROL
      ENDF
TIC1   1 TIP
*
TII    ZR0; TTYS; PIN TIIS1
      IF      V1
      MIN TIICTR
      ENDF
      STA TIIA; STX TIIX
      LDX TIIS1; CLA; COPY XA,E
      SKG =NTTY-1; BRU *+2; BRU TII4 (ILLEGAL TTY INP)
      LDA WERIS,2; SKG =-1; BRU TII4 (TTY NOT ON)
* FILE CHARACTER FOR ATII
      MIN ATIS5; LDA* ATIS5; SKA #200B; ADM ATIS5
      STX* ATIS5; MIN ATIS2
      IF ARMF; ARMI SATIW; ELSF 1; BRM ATII; ENDF
TII4   LDA TIIA; IF =ARMF; LDB TIIB; ENDF; LDX TIIX; BRI TII
*
* 'ATII' 5/18/66
*
* THIS ROUTINE IS STARTED BY A PERMANENTLY ACTIVE
* LOW-PRIORITY INTERRUPT TO FINISH UP FOR TII
*
ATIS2  ZR0;* ATII RING BUFFER COUNTER
ATIS4  ZR0;* ATII READOUT POINTER
ATIS5  ZR0;* WRITEIN POINTER
ATTBUF BSS 30;ATTBE DATA #30;* RING BUFFER
      IF      ARMF
ATIIA  ZR0;* SAVE (A)
ATIIB  ZR0;* SAVE (B)
ATIIX  ZR0;* SAVE (X)
      IF      V1
SATIW  DATA 202000B;* ARM ATI INT
RATIW  DATA 575777B;* DISARM ATI INT
      ELSF    1
SATIW  DATA 200400B;* ARM ATI INT
RATIW  DATA 577377B;* DISARM ATI INT

```

```

        ENDF
        ENDF
*
        IF      ARMF
ATII   ZR0; ARMI RATIO; SKN ATIS2; BRU **2; BRI ATII
        STA ATIIA; STB ATIIB; STX ATIIIX
        ELSF   1
ATII   ZR0; SKN ATIS2; BRU **2; BRR ATII
        ENDF
ATII1  ARMI RATIO; MIN ATIS4
ATII1A LDA* ATIS4; SKE ATTBE; BRU ATII2
        ADM ATIS4; BRU ATII1A
ATII2  STA TIIS2; COPY AX,AB,A; LCY 8; SKN TTYTBL,2; BRU TII6
        IF      V3
ATII3  ETR =177B; SKG =37B; SKG =32B; BRU **2
        BRU ATII6 (CONTR0L=SHIFT K, ETC.)
        SKG =137B; BRU TII1
        SKG =176B; SKG =173B; BRU ATIX
        ELSF   1
ATII3  ETR =177B; SKE =177B; BRU TII1
        ENDF
ATII6  LDA TTYASG,2; SKA =20000B; BRU **2; BRU ATIX
* PROCESS RUBOUT
        LDA REAL; MRG =40000000B; XMA TTYTIM,2; SKA =40000000B
        SUB TTYTIM,2; CNA
        SKG =10; SKG =-1; BRU **3; LDA =-1; STA TTYTIM,2
* SEARCH P.U. QUEUE
        CXA; ETR =77B; STA TIIS2; LDX PUBPTR
ATR1   CXA; SKE =PUBPTR; BRU ATR2
        IF      V5
        LDA TIIS2; CAX; MRG =300000B; CAB; LDA TIC1
        ELSF   1
        LDB TIIS2; LDA TIC1
        ENDF
        DIR; BRM EPU; MIN ACTR; MIN ACTPU
ATIX   SKR ATIS2; BRU ATII1
        IF      ARMF
        LDA ATIIA; LDB ATIIB; LDX ATIIIX; BRI ATII
        ELSF   1

```



```

BRR ATII
ENDF
ATR2 LDA 1,2; LDB 2,2; LDX 0,2; SKE TIC1; BRU ATR1
CBA; ETR =77B; SKE TIIS2; BRU ATR1; BRU ATIX (GOT ONE ALREADY)

```

\* NOT SPECIAL

```

TII1 SKN TTYFLG,2
BRU ATIX
STA TIISV SAVE INPUT CHAR.
SKG =15B CK IF POS. LF OR CR.
BRU TII1F YES
TII1E LDA TIIS5,2 SAVE CURRENT CHAR AS PREVIOUS.
ETR =70000000B SAVE ANY FLAGS
MRG TIISV
STA TIIS5,2
LDA TIISV RECOVER INPUT CHAR.
MUL =12525253B DIV. BY THREE
ADD TTYTBL,2

```

```

CAX; LDX 0,2 (ECHO WORD IN X); LCY 5; ETR =30B (SHIFT IN A)
COPY XB,AX; LCY 8,2; CLB; RCY 8 (ECHO IN B)
LDX TIIS2; CXA; ADD X3 (SUBTRACT 240B); ETR =37600000B
SKB =37400000B; BRU TII1B (ECHO AND FILE)
SKB =37600000B; BRU TII0 (FILE W/O ECHO)
BRU TII1A (IGNORE EXCEPT FOR BREAK)

```

```

TII1F IF V6
SKG =11B CK FOR POS CR OR LF.
BRU TII1E NO
LDA X1 CHECK FOR FLAG TO INHIBIT CR/LF SUPPRESS.
SKA TIIS5,2
BRU TII1E FLAG SET
LDA TIISV
SKE =12B CK FOR LF.
BRU =+3 NO
LDA =15B PREPARE TO SEE IF PREV.CHAR WAS CR.
BRU TII1G
SKE =15B CK IF CR.
BRU TII1E NOT EITHER CR OR LF
LDA =12B

```

```

TIIIG  LDB      =377B      GET MASK TO IGNORE FLAGS
        SKM      TIIS5,2   CK IF PREV. CHAR WAS OPPOSITE CURRENT
        BRU      TIIIE     NO, USE CURRENT
        LDA      TIIS5,2   YES, SAVE CURRENT
        ETR      =70000000B  SAVE FLAGS
        MRG      TIISV
        STA      TIIS5,2
        BRU      ATIX      IGNORE
        ENDF

```

\* NEEDS ECHO

\* INPUT; A=CHAR, LEFT IN INTERNAL CODE, B=ECHO LEFT

```

TIIIB  STB TIIS4; SKN TTYTBL,2; BRU TII3 (DEFER ECHO)
TIIID  SKN TOS2,2; BRU TII3B (TYPING, DEFER); STA TIIS2
        LDA TIS2,2; SKE =0; BRU TII3A (CHECK PREV DEFER)
TIIIC  MIN TOS5,2; LDA* TOS5,2; SKA =200B; BRU TII8
TII9   RCY 16; LDB TIIS4; LCY 16; STA* TOS5,2; MIN TOS2,2
        MIN TOS2,2; LDA =140000B; COPY XA,E; XMA TIIS4
        TTYS; POT TIIS4; CAB; BRU **2

```

\* FILE IN BUFFER AS OUTPUT CHAR.

\* INPUT; A=CHAR, LEFT IN INTERNAL CODE, B=ECHO LEFT.

```

TII0   STA TIIS2; MIN TIS4,2; LDA* TIS4,2; SKA =200B; BRU TII5
        ETR =77600377B; CNA; ADD TIIS2; ADM* TIS4,2 (SAVE OUTPUT CHAR)
        MIN TIS2,2; LDA TTYLNG,2; SKG TIS2,2; STA TTYFLG,2 (INP BUF OFL0)
        LDA TIS2,2; SKG =NTTYC=TTYEWM; BRU TII1A
        CLA; STA TTYBRK,2; MIN ACTR; BRU TII7A

```

TII1A SKB X4; BRU TII1K (BREAK CHAR)

LDA TTYBRK,2; SKE XX; BRU TII1K

TII7A LDA REAL; ETR XX; STA TTYTIM,2; BRU ATIX

TII1K LDA REAL; SKG TTYBRK,2; BRU \*\*2; BRU TII7A
 ADD =13; STA TTYBRK,2; MIN ACTR; BRU TII7A

\* RECORD BREAK

TII7 MIN TTYBRK,2; MIN ACTR; BRU TII7A

\* ECHO WRAPAROUND

TII8 ADM TOS5,2; LDA\* TOS5,2; BRU TII9

\* 8=LEVEL MODE MAYBE

TII6 LDB TTYTBL,2; SKB =8RB; BRU \*\*2; BRU ATII3 (NO 8=LEVEL IN)
 RCY 8; LDB =77600000B; SKM TIIS2; BRU TII2

\* 8=LEVEL EOF

```

        LDA TTYTBL,2; ETR =8PB+ILB+0LB+AMB+AIB
        SKA =8PB+ILB+0LB; BRU **2; EOR X4; MRG =ETO; STA TTYTBL,2
TII2   LDA TIIS2; ETR =77600000B; CLB; BRU TII0 (NO CONV OR ECH0)
* TAKE CARE OF WRAPAROUND
TII5   ADM TIS4,2; BRU TII0+2
* FILE ECH0
TII3A  LDA TIIS2; LDB* TIS4,2
        SKB =177B; BRU TII3B (PREV DEFER); BRU TII1C
TII3   LDB TTYTBL,2; SKB =ILB+0LB; BRU TII3B; BRU TII1D
TII3B  RCY 8; LDB TIIS4; LCY 8; ETR =37600177B; LDB X4; BRU TII0
*
* 'TSON', 'RPAN' 3/20/66
*
* THIS ROUTINE, ENTERED FROM THE PHANTOM USER, PROCESSES RUBOUTS
* TIPIX VALUES: 0=BRS 112. 1=ON INT. 2=OFF INT. 3=RUBOUT
*
TIP    LDX FILE; STX UTTY
        IF      V5
        LDX TIPIX; BRU* TIPL,2
TIPL   DATA TSON,TSON,TFIP,RPAN
        ELSF    1
        SKN TTYASG,2; BRU TSON
        ENDF
* NORMAL RUBOUT
* INPUT; PUPAC=TTYASG
RPAN   LDX UTTY; LDA TTYTIM,2; SKE =-1; BRU TIP6
* EMERGENCY == RETURN TO EXEC
        LDA REAL; STA TTYTIM,2; LDX TTYASG,2; BRM HFK; STX PUPAC
TIP6   LDX PUPAC; LDA =2000000B
        BRM IIR; BRU **2; BRU NPUG0 (INT ARMED AND CAUSED)
* RUBOUT INTERRUPT NOT ARMED
* CLEAR INPUT BUFFER
        LDX UTTY; DIR; BRM CLIB; EIR
        LDX PUPAC; BRM RFK; BRM TFK (TERM AFFECTED FK STRUCTURE)
* CLEAR OUTPUT BUFFER FOR EXEC PR0GS
        LDX PUPAC; SKN PQU,2; BRU NPUG0
        LDX UTTY; BRM CL0B; BRU NPUG0
TIPIX  ZR0      0
*

```

\* OFF INTERRUPT PROCESSING

```
IF V5
TFIP LDX UTTY; LDA =-3; SKE WERIS,2; BRM M0NCR
      LDX TTYASG,2; STX TFIP3; BRM HFK; STX PUPAC
      LDX UTTY; DIR; BRM CLIB; MIN WERIS,2; EIR
      LDX PUPAC; BRM RFK; BRM TFK; LDA PUPAC; LDB TFIP1
      SKE TFIP3; LDB TFC02; STB PL,2
      LDX UTTY; BRM CL0B; LDA =-1; STA T0S2,2; BRU NPUG0
TFIP1 ZR0 0FFINT,4
TFIP3 ZR0 0
      ENDF
```

\* CLEAR INPUT BUFFER

```
CLIB ZR0; LDB TIS5,2; STB TIS4,2; LDB =-1; STB TTYFLG,2
      CLB; STB TIS2,2; BRR CLIB
```

\* UNUSED TELETYPE

\* START UP NEW USER

```
TS0N LDA FPLST; SKG =0; BRU TS0N1
      LDA FULST; SKG =-1; BRU TS0N1
      STA J0B; CAX; LDA FILE; XMA TTN0,2; STA FULST
      LDX FILE; STX T0S2,2; BRM CL0B; LDA =-1; STA T0S2,2
      BRM CLIB
```

\* START UP EXEC FOR THIS USER

```
BRM GFK; BRU TS0N2; LDA =NUMEM; MUL J0B; LSH 23; STX PUPAC
LDX J0B; ADD TS0NC4; STA TS0NT1
SUB =NCMEM+NUMEM; STA PMTJ0B; ADD =-20000000B+NCMEM; STA PMTP,2
LDA =UMSZ*100000B+NUMEM; STA PMA,2; CLA; STA RL3,2; LDX =-NUMEM
STA* TS0NT1; BRX =-1; LDX PUPAC; BRM SETPAC
LDA =NCMEM; LDX J0B; STA RL3,2; BRU NPUG0
```

```
TS0N2 LDX J0B; CAX; XMA FULST; STA TTN0,2
```

```
TS0N1 LDA TS0NM; LDB FILE; LDX FILE; DIR; BRM EPU; BRU NPUG0
```

```
TS0NC2 DATA 00011011B
```

```
TS0NC3 DATA 13000000B ○
```

```
TS0NC4 ZR0 PMT,2
```

```
TS0NT1 ZR0
```

```
TS0NC1 ZR0 TS0N1
```

\* SET UP NEW PAC SLOT

```
SETPAC  ZR0; STA RL2,2; STA RL1,2
        LDB J0B; LSH 39; ADD =EXECL; MRG X6; STA PTAB,2
        LDA X1; STA PIM,2
        LDA PUPAC; LDX FILE; STA TTYASG,2
        LDB =-1; STB TTYFLG,2; LDX =GTI; BRM QPUT; LDX PUPAC
        LDA TS0NC1; STA PL,2; LDA TS0NC2; STA RL1,2; LDA TS0NC3; STA RL2,2
        CLA; STA PA,2
        LDA PACDMB; STA PTEST,2; LDA =40000000B+NFQU*100000B
        STA PQU,2; BRR SETPAC
```

\*

```
TS0NM 5  *+1; ASC '$N0 R00M.$/!'
TS0NI  CLAB; LDX PACPTR; STA MGTS5; BRM MGET; BRU TS0N3
        LDA =-1; STA MGTS5
        LDX PACPTR; LDA X1; ADM PTAB,2; LDA X4; STA XPB
        LDA RRL3; ETR =77B; STA RLTS
        LDA X2; LDX =NCMEM; ADM* PMTJ0B
        LDA =-1; STA SMIFIL; STA SM0FIL
        LDA RMPM; STA RMAP
        LDA =7; STA NF0RK
        LDX =-7; LDA =2; STA PX,2; ADD =1; BRX *-2
        LDA =1; STA PPB
        LDA =MBUF; STA FBWRD; BRU EXECI,4
TS0N3  LDA =-1; STA MGTS5
        LDA TS0NM; LDB UTTY
        IF V5; CBX; ENDF
        DIR; BRM EPU; LDX J0B; SBRS 112
RMPM  DATA 37777400B
```

\*

\*

```
IF V6
* BRS BE+11 LF/CR IGNORE SWITCH
* IF A LESS THAN 0, DON'T IGNORE LF/CR
CRSW  LDA X1; LDX UTTY; SKN SS01; BRU CRSW1; MRG TIIS5,2
CRSW2 STA TIIS5,2; BRU P0PX
CRSW1 E0R =-1; ETR TIIS5,2; BRU CRSW2
        ENDF
```

\*

```

*
*
* !TSOFF! 3/8/66
*
* BRS 112
* TURN OFF A TELETYPE STATION
* INPUT: X=JOB NO.
*
TSOFF SKN PQU,2; BRM TRAPB; LDA SS03; COPY AX,B; STB PMTP,2
      XMA FULST; XMA TTN0,2
      IF V5; ETR =77B; ENDF; COPY AX,AB
      LDX TTYASG,2; CXA; ADD =PPTR; XMA FPLST; STA PPTR,2; STX SS01
      CBX; BRM TTSOFF; CLX
* CLEAR TAPES
TSOFF3 LDA TJN0,2; SKE SS03; BRU TSOFF4; LDA =-1; STA TJN0,2
TSOFF4 EAX 1,2; CXA; SKE =NTAPE; BRU TSOFF3
      LDA SS03; SKE JOB; BRM M0NCR
      IF V5
* CLEAR PU
      LDX PUBPTR
TSOFF6 CXA; SKE =PUBPTR; BRU TSOFF5
      MIN NB112
      LDX JOB; CLA; STA AUNN,2
      LDA UTTY; ETR =77B; COPY AB,AX; MRG =100000B
      STA SS02; MRG =200000B; STA SS01; LDA =-1
      DIR; SKE WERIS,2; BRU **2; BRM M0NCR; SKG WERIS,2
      BRU TSOFF9 (0N);
* OFF INTERRUPT PROCESSING
      STA WERIS,2; EIR; SKN DOWN; BRU **2; BRU PACG0 (SUICIDE)
      TTYS; P0T SS02; LDA =4; DIR; BRM TREC; BRU PACG0 (SUICIDE)
* LOGOUT ONLY. TTY STILL CONNECTED.
TSOFF9 STA TTYTIM,2; LDA =NPU0-1
      SKG PUCTR; BRU TSOFF10; CLA; STA WERIS,2; LDA TIC1
      BRM EPU; MIN ACTPU; BRU PACG0
TSOFF10 EIR; SKN DOWN; BRU **2; BRU PACG0 (SUICIDE)
      TTYS; P0T SS01; MIN B112; LDA =-5; DIR
      BRM TREC; BRU PACG0 (SUICIDE)
B112 ZR0 0

```

```

* CLEAR PU
TS0FF5 LDA 3,2; ETR =77B; SKE UTTY; BRU TS0F11
      LDA PUDEAD; STA 1,2; MIN EXTPU
TS0F11 LDX 0,2; BRU TS0FF6
      ELSF 1
      BRU PACG0
      ENDF

```

```

*
* 'T01' P. DEUTSCH 9/13/65
*
* THIS ROUTINE PROCESSES TELETYPE OUTPUT INTERRUPTS.
*

```

```

T01A  ZR0;* SAVE (A)
T01B  ZR0;* SAVE (B)
T01X  ZR0;* SAVE (X)
T0IS2 ZR0;* TELETYPE NO.
T018  ZR0 0

```

```

*
T01   ZR0; STA T01A; STX T01X; TTYS; PIN T0IS2; LDX T0IS2
      SKR T0S2,2; BRU *+2; BRU T011A
      MIN T0ICTR
T012  MIN T0S4,2; LDA* T0S4,2; STA T018
      SKA =200B; BRU T011; STB T01B
      LCY 8
      ETR =377B*200000B; LDB TTYTBL,2; SKB =8PB; BRU T012B
      ETR =37600000B
      SKG =140B*200000B
      SKG =15B*200000B
      BRU T013 POSSIBLE LF OR CR
T012B SKN TIIS5,2 CK FLAG FOR LEFT EDGE PAPER
      BRU T012A NOT AT EDGE
      CAB
      LDA X4 REMOVE LEFT EDGE FLAG
      ADM TIIS5,2
      CBA
T012A MRG =40000B BIT 9. 0 FOR NO INTERRUPT.
      ADM T0IS2; TTYS; POT T0IS2
      LDA T0S2,2; SKG TTYEMG; MIN ACTR
      LDA REAL; ETR XX; STA TTYTIM,2

```

```

LDB T01B
T011A LDA T01A; LDX T01X; BRI T01
T011 ADM T0S4,2; BRU T012+1

IF V6
T013 SKG #11B*200000B
BRU T012B NOT POS. CR/LF
SKE #12B*200000B
BRU T014
LDB X2 LINE FEED
SKN TIIS5,2 CK IF AT LEFT EDGE
BRU T015 NO
SKB TIIS5,2 CK IF C.R. HAS BEEN SUPPLIED YET
BRU **2
BRU T012A YES, TYPE THE LINE FEED
LDA #20000000B REMOVE X2, LEAVE SIGN ON
ADM TIIS5,2
LDA #177B*200000B TYPE A R.O.
BRU T012A
T014 SKE #15B*200000B CK IF C.R.
BRU T016 NO
SKN TIIS5,2 CK IF AT LEFT EDGE
BRU **3 NO
LDA #12B*200000B AT EDGE SO TYPE LINE FEED ONLY
BRU T012A
CAB SAVE CHAR
T014A LDA X4 SET FLAG FOR LEFT EDGE
ADM TIIS5,2
LDA #NTTYC=1
SKN T018
LDA #=1
ADM T0S4,2 BACK UP POINTER TO POINT TO CURRENT CHAR
CBA
MIN T0S2,2
BRU T012A
T015 SKB TIIS5,2 CK IF NEW LINE FEED
BRU **3 NO, GO SUPPLY C.R. FOR PREV. L.F.
XAB
BRU T014A+1

```



```

LDB      =15B*200000B      TYPE C.R.
BRU      T0I4A
T0I6    SKG      =140B*200000B
BRU      T0I2B
SKE      =142B*200000B      SPECIAL LINE FEED
BRU      **2
BRU      **3
SKE      =145B*200000B      SPECIAL CARRIAGE RET.
BRU      T0I2B
SUB      =130B*200000B
BRU      T0I2A
ENDF

```

```
IF      V1
```

```

*
* !TNI! 10/12/66
* THIS ROUTINE PROCESSES CARRIER ON INTERRUPTS
*
TNIS     0
TNIA     0
TNIB     0
TNIX     0
TNI      ZR0; TTYS; PIN TNIS; STA TNIA; STB TNIB; STX TNIX
LDA TNIS; ETR =77B; SKG =NTTY-1; BRU **2; BRU TNI1; CAX
IF      V5
MIN TNICTR
LDA =2; DIR; BRM TREC
LDA WERIS,2; SKE =-1; BRU TNI1; CLA; STA WERIS,2
MIN TNIA
CXA; MRG =100000B; CAB; LDA TIC1
DIR; BRM EPU; MIN ACTR; MIN ACTPU
ELSF    1
LDA TTYASG,2; SKE ADMSK; BRU TNI1
LDA TIC1; CXB; DIR; BRM EPU; MIN ACTR
ENDF
TNI1    LDA TNIA; LDB TNIB; LDX TNIX; BRI TNI
*
* !TFI! 10/12/66

```

```

*      THIS ROUTINE PROCESSES CARRIER OFF INTERRUPTS
*
TFIS  0
TFIA  0
TFIB  0
TFIX  0
TFI   ZR0; TTYS; PIN TFIS; STA TFIA; STB TFIB; STX TFIX
      LDA TFIS; ETR #77B; SKG #NTTY-1; BRU #+2; BRU TFI1
      IF V5
      MIN TFICTR; CAX
      MRG #20000B; STA TFIS; DIR; LDA #0; SKG WERIS,2; BRU TFI2
      EIR; BRU TFI3
TFI2  LDA #3; EIR; STA WERIS,2; LDA REAL; STA TTYTIM,2
      LDA #1; STA TOS2,2; MIN TFIAC; MIN ACTR; MIN ACTPU
      LDA #NPUQ-1; SKG PUCTR; BRM M0NCR; CAX; MRG #20000B
      CAB; LDA TIC1; DIR; BRM EPU
TFI3  LDA TFIS; ETR #77B; CAX; LDA #8
      DIR; BRM TREC; TTYS; P0T TFIS
      ELSF
      CAX; LDA TTYASG,2; LDB #100000B; SKE ADMSK; LDB #120000B
      COPY XB,E; STB TFIS; TTYS; P0T TFIS; SKE ADMSK; BRU #+2; BRU TFI1
      LDA REAL; STA TTYTIM,2; CXB; LDA TFC1; DIR; BRM EPU
      ENDF
TFI1  LDA TFIA; LDB TFIB; LDX TFIX; BRI TFI
*
*      IF V5
* 'TREC' 1/3/67
*      THIS ROUTINE RECORDS THE STATE CHANGES IN TTY LINES.
*      LOWEST OCTAL DIGIT CHANGES.
*      2=ON INTERRUPT. 0=OFF INTERRUPT. 4=ANSWERED. 1=CARRIER ON.
*      INPUT: A=STATE OF LINE. X=TTY NUMBER.
*
TREC  ZR0; STA TRECA; STB TREC B; LDA TREC1; SKN TRECA
      ADD TREC2; STA TREC3; LDA TTYREC,2; LRSB 3
      LDA TTYREC,2; LSH 3
TREC3 MRG TRECA; STA TTYREC,2; LDB TREC B; EIR; BRR TREC
TREC A ZR0 0
TREC B ZR0 0

```

TREC1 ETR TRECA  
TREC2 ERM 0

\*  
\*

\* BRS BE+6 TTY0N

\* INPUT: A=TTY NO. IF B LESS THAN ZERO, TURN TTY 0N.

TTY0N SKN PQU,2; BRM TRAPB; ETR =77B; MRG =100000B  
SKN SS02; BRU TTY0FF; SKN DOWN; BRU \*+2; BRM TRAPB  
STA TT09; CAX; SKN WERIS,2; BRU P0PX  
TTYS; P0T TT09; LDA =4; DIR; BRM TREC; BRU P0PX

TTY0FF MRG =200000B; STA TT09  
TTYS; P0T TT09; ETR =77B; CAX; LDA =-5  
DIR; BRM TREC; BRU P0PX

TT09 ZR0 0

\*  
\*

\* BRS 126 (BE+3) CARRIER TEST

\* INPUT: A=TTY NO.

\* RETURN: NO SKIP; NO CARRIER

\* SKIP: CARRIER PRESENT

CARRY SKN PQU,2; BRM TRAPB; ETR =77B; CAX; MRG CRY2; STA CRY3

CRY3 SKS\* 36200B,2; BRU P0PX (NO CARRIER)

LDA =1; DIR; BRM TREC; MIN 0; BRU P0PX

CRY2 SKS\* 36200B,2

\*  
\*

ENDF

\*  
\*

\* ITI! 9/26/65 = 31 CYCLES NORMALLY

\*  
\*

\* THIS ROUTINE GETS A CHARACTER FROM TELETYPE (X), OR CREATES  
\* A 'PTEST' WORD IF THIS IS NOT POSSIBLE. IT ALSO PROCESSES  
\* DELAYED ECHOS.

\*  
\*

TIS1 ZR0;\* SAVE CHAR WHILE TYPING ECHO

\*  
\*

TI ZR0; LDA TIS2,2; SKG =0; BRU TI3

TIS MIN TIS5,2; LDB\* TIS5,2; SKB =200B; BRU TI2

CLA; LCY 8; SKB =77400B; BRU TI1

```

TI7   SKR TIS2,2; LDB =-1; STB TTYFLG,2; BRR TI
TI2   CBA; ADM TIS5,2; BRU TIS+1
* DISMISS
TI3   CXA
      LDB XX; STB TTYBRK,2; ADD =600000B+TTYBRK; CAB
      MIN TI; BRR TI
* PRINT DELAYED ECHO
TI1   STA TIS1; LCY 16; LDB =1; BRM T0; BRU TI6 (BUFFER NOT FULL)
      LDX T0S8; SKR TIS5,2; MIN TI; BRR TI
TI6   LDA TIS1; BRU TI7
*
* 'TCI','FTCI','IST','ISTC','LFU','LFI' 6/28/66
*
* THIS IS THE USER INTERFACE FOR TELETYPE INPUT
*
TCI   POPD      17400000B,1,1,0,1
TCIP  STB SS02; STX SS03; LDX UTTY; BRM TI; BRU TCIP1
TIDMS LDA =-1; STA TIME; STA TTIME
TI0DMS LDX =QT1; BRU POPDMS
TCIP1 LDX SS03; STA* 0; LDB SS02; BRU XP0P
FTCI  LDX UTTY
FTI   BRM TI; BRU **2; BRU TIDMS; STA T; BRR GPW
IST   POPD      15000000B,1,1,0,1
      IF V8; BRM TRAPB; ELSEF 1
ISTP  STB SS02; STX SS03; LDA* 0; BRM GATX; BRM TRAPB
      BRM TI; BRU **2; BRU TIDMS; LDB SS02; LDX SS03; BRU XP0P
ISTC  LDA UNIT; BRM GATX; BRM TRAPB; BRU FTI
      ENDF
*
* 'STI' 2/6/66
*
* SIMULATE TELETYPE INPUT
*
STI   POPD      13600000B,1,1,0,1
STIP  STA SS01; STB SS02; STX SS03
      LDA* 0; SKG =-1; LDA UTTY; STA UNIT; CAX
      SKE UTTY; BRU **2; BRU STIP1
      LDX PACPTR; SKN PQU,2; BRM TRAPB; LDX UNIT
STIP1 LDA TIS2,2; SKE =0; BRU STIP2

```

```
LDB SS01; LSH 40; MRG UNIT; ADD X1; CAX; DIR
MIN ATIS5; LDA* ATIS5; SKA =200B; ADM ATIS5
EIR; STX* ATIS5; MIN ATIS2
IF ARMF; ARMI SATIW; ELSF 1; DIR; BRM ATII; EIR; ENDF; BRU P0PX
STIP2 CXA; ADD =100000B+TIS2; CAB; BRU TI0DMS
```

```
*
* 'T0' 9/26/65
```

```
* THIS ROUTINE TYPES OUT THE CHARACTER (A) ON TELETYPE (X),
* CREATING A 'PTEST' WORD IF THIS IS NOT POSSIBLE. IT TAKES
* CARE OF LINKED TELETYPES AS WELL.
```

```
*
T0S1 ZR0;* OUTPUT CHAR FOR STARTING UP TELETYPE
T0S8 ZR0;* TELETYPE 'T0' WAS CALLED FOR
T0S9 ZR0;* NUMBER OF CHARACTERS TO PRINT
```

```
*
* INPUT: A=CHAR. IN TTY CODE, B=NO. OF CHAR'S, X=TTY NO.
* OUTPUT: B=PTEST IF BUFFER IS FULL.
* SKIP: BUFFER FULL
* NO SKIP: BUFFER NOT FULL.
```

```
T0 ZR0; STB T0S9; STX T0S8; ETR =377B; MUL =200B
LDA TTYLNG,2; SUB T0S9; SKG T0S2,2; BRU T04
```

```
* NOT FULL BUFFER
```

```
DIR; SKN WERIS,2; BRU **2; BRU T06; MIN T0S5,2
LDA* T0S5,2; SKA =200B; BRU T05
```

```
T02 ETR =77600377B; CNA; COPY BA,N; STA* T0S5,2
SKN T0S2,2; BRU T03; CXA; ETR =77B; MRG =140000B (BITS 8,9)
STA T0S1; ETR =77B; MRG =TTYSKS; STA **1
SKS* 37000,2; BRM M0NCR; TTYS; P0T T0S1; MIN T0S2,2
```

```
T03 MIN T0S2,2
```

```
T06 EIR; BRR T0
```

```
T05 ADM T0S5,2; LDA* T0S5,2; BRU T02
```

```
* FULL BUFFER
```

```
T04 CXA; ETR =77B; ADD =300000B+T0S2; CAB
MIN T0; BRR T0
```

```
*
*
IF -V8
* CHECK FOR LINKS
```

```

T06 LDA TTYTBL,2; SKA =0LB; BRU T0L1; BRU T07
T0L1 STB TLMB; LDA =T0L2; BRM TL0M; LDX T0S8; LDA =T0L3; BRM TL0M
LDX T0S8; BRR T0
T0L2 ZR0; LDA TTYLNG,2; SUB T0S9; SKG T0S2,2; BRU T04; BRR T0L2
T0L3 ZR0; LDB TLMB; BRU T07

```

```

*
* 'TL0M', 'TLIM' 1/30/66

```

```

* MAP LINK WORDS

```

```

*
TLMA ZR0; * SUBROUTINE ADDRESS
TLMB ZR0; * SAVE B
TLMW ZR0; * SAVE LINK WORD

```

```

*
TL0M ZR0; LDX A0LCW,2
TLM1 STA TLMA; CXA
TLM2 SKE =0; BRU TLM3; BRR TL0M
TLM3 LRSB 1; LDX =23; N0D 23; E0R X2
LCY 2,2; STB TLMW; BRM* TLMA; LDA TLMW; BRU TLM2
TLIM ZR0; LDB TLIM; STB TL0M; LDX AILCW,2; BRU TLM1
ENDF

```

```

*
* 'T0F', 'TC0', 'FTC0', '0ST', '0STC', 'LF0' 6/28/66

```

```

* THIS IS THE USER INTERFACE FOR TELETYPE OUTPUT

```

```

* INPUT: A=CHAR., X=TTY NO.

```

```

T0F ZR0; SKN TTYTBL,2; BRU T0F4
T0F5 LDB T0S3,2; SKB X4; BRU T0F1; SKB X2; STA T0S3,2
T0F2 SKR T0S3,2; BRU **2; BRR T0F
LDB =1; LDA =240B; BRM T0; BRU T0F2
MIN T0S3,2; BRU I0QDMS
T0F1 SKE =135B; BRU T0F3; LDA X2; STA T0S3,2; BRR T0F
T0F4 LDB TTYTBL,2; SKB =8PB; BRU T0F3A; BRU T0F5
T0F3 ADD =240B
T0F3A LDB =1; BRM T0; BRR T0F; BRU I0QDMS

```

```

*
TC0 P0PD 17500000B,1,1,0,1
TC0P STA SS01; STB SS02; STX SS03; LDA* 0; LDX UTTY

```

```

FTC0  BRM T0F; BRU P0PX
      LDX UTTY
      LDA SS01; ETR =377B; STA T; BRM T0F; BRR GPW
0ST   P0PD 15100000B,1,1,0,1
      IF V8; BRM TRAPB; ELSF 1
0STP  STA SS01; STB SS02; STX SS03; LDX* 0; BRM GLTT
      LDA SS01; BRM T0F; BRU P0PX
0STC  LDX UNIT; BRM GLTT; BRU FT0
      ENDF

```

```

*
*
* 'CIB','C0B','SKI','D0B','CET','RDET' 10/18/65
*
* THESE ROUTINES MANIPULATE TELETYPE BUFFERS AND ECH0 TABLES.
*

```

```

* BRS 11 CLEAR INPUT BUFFER
CIB   BRM GATT; DIR; LDA TIS5,2; STA TIS4,2
      LDA =-1; STA TTYFLG,2; CLA; EIR; STA TIS2,2; BRU P0PX

```

```

* BRS 29 CLEAR OUTPUT BUFFER
C0B   BRM GATT; BRM C0B; BRU P0PX

```

```

* BRS 13 SKIPS IF INPUT BUFFER EMPTY
SKI   BRM GATT; LDA TIS2,2; SKG =0; MIN 0; BRU P0PX

```

```

* BRS 14 DISMISS UNTIL OUTPUT BUFFER EMPTY.
D0B   BRM GATT; SKN T0S2,2; BRU +=2; BRU P0PX
      CXA; ADD =1200000B+T0S2; CAB; LDX =QTI; BRU P0PDMS

```

```

* BRS 12 CHANGE ECH0 TABLE
CET   BRM GATT; LDA SS01; SKG =-1; BRU CET1; SKA =(N0T)3; BRM TRAPB
CET4  CLB; LSH 5; ADD =ETO
CET2  DIR; XMA TTYTBL,2; ETR =37740000B-8RB-8PB
      ADD TTYTBL,2; EIR; STA TTYTBL,2; SKA =8PB+8RB+ILB+0LB; BRU P0PX
      LDA X4; ADM TTYTBL,2; BRU P0PX
CET1  ETR =377B; MRG =8RB; BRU CET2

```

```

* BRS 40 READ ECH0 TABLE NUMBER
RDET  BRM GATT; LDA TTYTBL,2; SKA =8RB; BRU RDET1

```

```

SUB =ETO; ETR ADMSK; RSH 5
RDET2 STA SS01; BRU P0PX
* 8=LEVEL INPUT, GET TERMINAL CHARACTER.
RDET1 ETR =377B; MRG X4; BRU RDET2
*
* 'MSGS', 'ASTT', 'RSTT' 1/30/66
*
* THESE ROUTINES CHANGE THE ACCEPT MESSAGES BIT AND ASSIGNMENT
* STATUS OF TELETYPES
*
IF      =V8
MSGS   BRM GATT; CLA; LDB SS01; SKB =1; LDA =AIB
      SKB =2; MRG =AMB; DIR; XMA TTYTBL,2
      ETR =(NOT)AMB(AND)(NOT)AIB; EIR; ADM TTYTBL,2; BRU P0PX
*
*
* NOT IMPLEMENTED
* BRS 27 ATTACH TELETYPE
ASTT   SKG =NTTY=1; SKG =-1; BRM TRAPB
      COPY XB,AX; LDA TTYASG,2; SKE ADMSK; BRU P0PX
      LDA JOB; STA TTYASG,2; LDA =40000000B+AMB+APB+ETQ; STA TTYTBL,2
      LDA =-1; STA TTYFLG,2; MIN 0; BRU P0PX
* BRS 28 RELEASE TELETYPE
RSTT   SKG =NTTY=1; SKG =-1; BRM TRAPB
      COPY XB,AX; LDA TTYASG,2; SKE JOB; BRM TRAPB
      BRM TT0FF; BRU P0PX
      ENDF
*
*
* 'TT0FF', 'GATT', 'GLTT', 'CL0B' 1/30/66
*
* THESE ROUTINES TURN TELETYPES OFF, AND CHECK THE LEGALITY OF
* I/O AND LINK REQUESTS
*
TT0FF  ZR0; CLA; STA TTYFLG,2; LDA ADMSK; STA TTYASG,2
      LDA =40000000B+AMB+ETO; STA TTYTBL,2; BRM CL0B
      BRR TT0FF
*

```



```

*
* INPUT; SS03=TTY NUMBER
GATT  ZR0; LDA SS03; BRM GATX; BRM TRAPB; BRR GATT
*
* CHECKS FOR LEGAL TTY NO. CHANGES =1 TO UTTY.
GATX  ZR0; SKG =NTTY-1; SKG =-2; BRM TRAPB; SKG =-1; LDA UTTY
      LDX PACPTR; LDB PQU,2; CAX
      SKE UTTY; SKB X4; BRU GATX1
      LDA TTYASG,2; SKE JOB; BRR GATX
GATX1  MIN GATX; BRR GATX
*
*
      IF      =V8
GLTT  ZR0; CXA; BRM GATX; BRU *+2; BRR GLTT
      LDA TTYTBL,2; SKA =AMB; BRR GLTT; BRM TRAPB
      ENDF
*
CL0B  ZR0; LDA =-1; STA T0S3,2; DIR; SKN T0S2,2; BRU CL0B1
      EIR; BRR CL0B
CL0B1  CLB; STB T0S2,2; EIR; LDA T0S4,2; STA T0S5,2
      BRR CL0B
*
* !SET8P!, !CLR8P! 10/18/65
*
* THESE ROUTINES IMPLEMENT THE SPECIAL, FULL 8 LEVEL TELETYPE
* OUTPUT
*
* BRS 85
SET8P  BRM GATT; LDA =8PB; SKA TTYTBL,2; BRU P0PX
      ADM TTYTBL,2; SKN TTYTBL,2; BRU P0PX
      LDA X4; ADM TTYTBL,2; BRU P0PX
* BRS 86
CLR8P  BRM GATT; LDA =-8PB; LDB TTYTBL,2
      SKB =8PB; ADM TTYTBL,2; SKB =8RB+ILB+0LB; BRU P0PX
      LDA X4; SKN TTYTBL,2; ADM TTYTBL,2; BRU P0PX
*
* !LNKC!, !CLCW!, !CLCB!, !LNKS! 2/6/66
*
* CLEAR, RECOMPUTE, AND SET LINKS

```

```

*
IF      =V8
RLCPTR  ZR0;* POINTS TO REL. LCW TABLE
ALCPTR  ZR0;* POINTS TO ABS. LCW TABLE
CLCWI   ZR0;* BIT COUNTER
CLCWJ   ZR0;* WORD COUNTER
*
LNKC    BRM GATT; STX T; LDB =1; LSH 24,2
        COPY AB,N; LDX =-NTTY
LNKC1   SKB ERILCW,2; ADM ERILCW,2
        SKB ER0LCW,2; ADM ER0LCW,2; BRX LNKC1
        LDX T; STB RILCW,2; STB R0LCW,2
        EAX CLWI; BRM CLCW; EAX CLW0; BRM CLCW
        BRM CLCB; BRU POPX
*
CLWI    ZR0 RILCW,2; ZR0 AILCW,2; DATA AIB,AKB
CLW0    ZR0 R0LCW,2; ZR0 A0LCW,2; DATA AMB,APB
CLCW    ZR0; LDA 0,2; LDB 1,2; STA RLCPTR; STB ALCPTR; LDX =NTTY=1
CLCW5   LDA* RLCPTR; STA* ALCPTR; CXA; EAX =1,2; SKE =0; BRU CLCW5
        LDX =NTTY=1; STX CLCWI
CLCW3   LDX =NTTY=1; STX CLCWJ
CLCW4   LDX CLCWJ; LDA* ALCPTR; LDX CLCWI; RCY 0,2; SKA =1; BRU CLCW1
CLCW2   SKR CLCWJ; BRU CLCW4; SKR CLCWI; BRU CLCW3; BRR CLCW
CLCW1   LDA* ALCPTR; LDX CLCWJ; MRG* ALCPTR; STA* ALCPTR; BRU CLCW2
*
CLCB    ZR0; LDX =-NTTY
CLCW6   LDA ETTYTB,2; ETR =ILB+0LB; CNA; ADM ETTYTB,2
        LDB =1; LSH 24+NTTY,2; SKE EAILCW,2; BRU CLCW6A
CLCW7   SKE EA0LCW,2; BRU CLCW7A
CLCW8   LDA X4; LDB ETTYTB,2; SKB X4; ADM ETTYTB,2
        SKB =8RB+8PB+ILB+0LB; BRU +=2; ADM ETTYTB,2
        BRX CLCW6; BRR CLCB
CLCW6A  CAB; LDA =ILB; ADM ETTYTB,2; CBA; BRU CLCW7
CLCW7A  LDA =0LB; ADM ETTYTB,2; BRU CLCW8
*
LNKS    BRM MSP; DATA 0; BRU LNKSA

        ENDF
ENDTTY  BSS 0

```

END

3W IDENT 6/30/67

\* ENTRY POINTS

ENTRY RTX,PNX,TRX,TWX,T2K,CPX,CPXB,LPX,CRX,CRXB  
ENTRY MTR0PN,MTW0PN  
ENTRY PNE0R,MREW,MFSF,MBSF,MBSR  
ENTRY ME0R,ME0F,MERS,MERL  
ENTRY TRTW,CFTW,CRTW

\*  
\* 'INT31','INT33' P, DEUTSCH 9/16/65

\*  
\* W BUFFER INTERRUPT ROUTINES

\*  
\* THE W BUFFER DRIVERS AND INTERRUPT ROUTINES HAVE THE FOLLOWING  
\* CONVENTIONS. IF THE ROUTINE SKIPS UPON RETURN, IT  
\* HAS FAILED. A FAILURE USUALLY OCCURS IF THE DEVICE ADDRESSED  
\* IS NOT READY FOR SOME REASON. IF THE ROUTINE RETURNS WITHOUT  
\* SKIPPING, THEN IT HAS SUCCEEDED IN PERFORMING THE TASK ASSIGNED  
\* TO IT.

\*  
\* 'RTX','PNX' 10/18/65

\*  
\* PAPER TAPE READER AND PUNCH DRIVERS

\*  
RTX ZR0; SETINT RTI; LDA I0B; ADD =RTCNT\*40000B+2; STA PTAPE  
E0M\* 2604B; EXU I0SW; P0T PTAPE; BRR RTX

\*  
RTI ZR0; ASCW; PIN PTAPE; LDA PTAPE; STA 1,2; EAX 2,2; STX =2,2  
LDA =1,2; SUB I0B; ETR ADMSK; ADD WBUFF  
BETW; MRG ERB0IT; CZTW; MRG E0RBIT; STA =1,2  
BRR RTI

\*  
IF PNX  
PNX ZR0; LDA 0,2; SUB WBUFF; SUB =2; CLB; LSH 14  
ADD I0B; ADD =2; STA PTAPE; SETINT PNI  
E0M\* 3644B; EXU I0SW; P0T PTAPE; BRR PNX

\*  
PNE0R ZR0; LDA E0RBIT; ADM 1,2; CXA; ADD =2

```

PNX1  SKE 0,2; BRU PNX1; BRM PNI; BRR PNE0R
      BRM PNX; BRR PNE0R
*
PNI   ZR0; LDA WBE; XMA 1,2; SKA E0RBIT; BRU PNI1
      EAX 2,2; STX =2,2; BRR PNI
PNI1  LDA PNI2; XMA 31B; E0M 1644B; BRU *
PNI2  BRU **1; T0PW; STA 31B; SETINT PNI; MIN PNI; BRI PNI
      ELSF 1
PNX   ZR0
PNE0R BRU TRAP
      ENDF
*
      IF V6 UP TO 'TRX'
LPX   ZR0; * ***** PRINTER DRIVER 4/2/67
      MIN LPX; CAT; BRR LPX; PRT; BRR LPX; PFT; N0P
      LDA P0SL1; EPT; STA P0SL1; EXU P0SL1; SETINT P0I (SLEW MAYBE
      LDB =60000000B; LDA 1,2; SKG 0,2; BRU **4
      STB* 0,2; MIN 0,2; BRU **4; LDA =LPCNT*40000B+2; ADM I0B
      LDA P0SL2; STA P0SL1; E0M* 2060B; EXU I0RW; P0T I0B; BRU* LPX
LPERRS BRM RSTBUF; LDA ERBBIT; ADM 1,2; BRU* LPX (ERR0R)
* PRINTER INTERRUPT ROUTINE
P0I   ZR0; BRM RSTBUF; LDA =LPCNT; ADM 1,2; CLA; STA PDSW1; BRR P0I
* ***** END OF PRINTER DRIVER
* CARD PUNCH DRIVER *****
CPX   ZR0; * OUTPUT BINARY OR H0LLERITH
      MIN CPX; CATW; BRR CPX (ERR EXIT IF CHNL ACTIVE)
      LDA =10; STA CPECNT (ERR0R REPUNCH MAX)
      BRM CPUNCH; BRR CPX; BRU* CPX (BRR N0T READY, BRU 0K)
CPUNCH ZR0; * PUNCH 12 ROWS, RETURN SKIPS IF PUNCH IS INITIATED
      SKS 14046B; BRR CPUNCH (BRR N0T READY)
      LDA =11; STA CPRCNT (ROW COUNTER)
      MIN CPBB; LDA =1; STA CPR0WI (BFR BUSY, N0T AN INTRPT)
* OUTPUT A ROW
CPR0W SETINT CPR0WI; SKN CPM0DE; BRU CPP (CLEAR UNUSED PART 0F BIN BFR
      SETINT CPR0WI; SKN CPM0DE; BRU CPP; STB* 0,2; MIN 0,2; BRU **4
CPP   LDX CPM0DE; LDA CPBUFF; ADD CPSIZE,1; STA CPP0T (CREATE P0T WRD)
      EXU CPE0M,2; E0M 16000B; P0T CPP0T (I0RD)
      SKN CPR0WI; BRR CPR0WI; MIN CPUNCH; BRR CPUNCH
CPFIX MIN CPECNT; SKN CPECNT; BRU CPF0RK (BRU TO END ERR0R LOOP)

```



```

COPY AX,A; LSH 2; MUL =3 X= WRD POSITION B=SHIFT
LDA CRCHRS,2; COPY BX,AB,A; LCY 6,2; ETR =77B ASC AT A16-23
SKE =60B; BRU CRT2; MIN CRBCNT; BRU CRT3 MIN IF BLANK
CRT2 COPY AB,A; SKE CRBCNT; BRU CRTBLK; BRM CRINSERT BRU 1ST NONBLANK
CRT3 SKR CRCC; BRU CRT1 END OF CARD
CRT4 LDB =155B; BRM CRINSERT; LDA CRTEMP; SKE =0; BRU CRT4 LST BFR WRD
LDX =CR2BUF; STX =2,2; BRR CRI
CRTBLK STA CRX; LDB =135B; BRM CRINSERT INSERT MULTPL BLK CHAR
LDB CRBCNT; BRM CRINSERT; LDA CRX; BRU CRT2
CRINSERT ZR0;* LOAD 8 BIT ASC IN B INTO NEXT BUFFER POSITION
LDX CRSFT2; RCY 32,2; ADM CRTEMP; EAX 7,2 POSITION ASC
BRX CRT5; CLA; XMA CRTEMP; STA* CR1BUF INSERT ASC WORD
MIN CR1BUF; STX CRTEMP; LDX =-24
CRT5 STX CRSFT2; BRR CRINSERT
CRTW SKS 12006B CARD READER READY
CFTW SKS 11006B CARD READER END OF FILE
CRE0FH DATA 40404000B EEE
CRE0FB DATA 77777777B
CRCC ZR0 0 CHR COUNTER
CRSFT1 DATA =3*6 SHIFT WORD TO POSITION SDS INPUT CHAR
CRSFT2 DATA =3*8 SHIFT WORD TO POSITION ASC CHAR
CRBCNT ZR0 0 CONSECUTIVE BLANK COUNT
CRTEMP ZR0
CRCHRS DATA 20212223B,24252627B,30314035B,07323676B
DATA 13414243B,44454647B,50513716B,11733403B
DATA 15525354B,55565760B,61620104B,12753302B
DATA 00176364B,65666770B,71720614B,10777405B
*
BINARY CARD READER DRIVER
CRXB ZR0; EXU CRTW; BRU CRNR (READY )
LDA =CRBUF; ADD =2+40*40000B; STA CRIB (PBT WORD)
SETINT CRIB; E0M* 3606B; E0M 16000B; PBT CRIB; BRR CRXB
CRIB ZR0; LDA CRIB; STA CRI; CETW; BRU CRERRS
LDA CR2BUF; SKE CRE0FB; BRU *+2; BRU CRE0F2
BRM RSTBUF; LDA =CRCNTB; ADM 1,2; BRR CRIB
*
LAST CRX,CRI *****
CPBB ZR0
CPBUFF ZR0
CPFLIP ZR0
CPMODE ZR0

```

CR1BUF ZR0  
CR2BUF ZR0  
CRBUF ZR0

\*  
\* 'TRX', 'TWX' 10/24/65

\* THESE ARE THE BASIC TAPE READING AND WRITING ROUTINES

\*  
TDT TRT,SKS,10410 SKIP IF TAPE NOT READY  
TDT BTT,SKS,12010 SKIP IF NOT BEGINNING OF TAPE  
TDT ETT,SKS,11010 SKIP IF NOT END OF TAPE  
TDT TFT,SKS,13610 SKIP IF NOT END OF FILE  
TDT FPT,SKS,14010 SKIP IF TAPE NOT FILE PROTECTED  
IF V1  
TDT D8T,SKS,17210 SKIP IF NOT 800 DENSITY  
ENDF  
TDT D2T,SKS,16210 SKIP IF NOT 200 DENSITY  
TDT D5T,SKS,16610 SKIP IF NOT 556 DENSITY  
TDT RTB,E0M\*,3610 READ TAPE BINARY  
TDT WTB,E0M\*,3650 WRITE TAPE BINARY  
TDT WFM,E0M\*,2050 WRITE FILE MARK  
TDT SFB,E0M,3630 SCAN FORWARD IN BINARY  
TDT SRB,E0M,7630 SCAN REVERSE IN BINARY  
TDT ETF,E0M\*,3670 ERASE TAPE FORWARD  
TDT ETR,E0M\*,7670 ERASE TAPE IN REVERSE  
TDT REW,E0M,14010 REWIND  
TXS2 ZR0;\* SPACING CTRL FILE COUNT  
TXS3 ZR0;\* TRY AGAIN COUNT  
TXS4 ZR0;\* INTERLACE WORD

\*  
\* OPEN MAG TAPE  
MTW0PN ZR0; LDA MTW0PN; STA MTR0PN; LDX SS01; EXU FPTW,2; BRU T0PN2  
BRU MTR0PN+1  
MTR0PN ZR0; LDX SS01; LDA TJN0,2  
SKE J0B; SKG \*=1; BRU \*+2; BRU T0PN1  
EXU TRTW,2; BRU T0PN3  
IF V1  
EXU D8TW,2; BRU T0PN3  
ENDF



```

        EXU D2TW,2; BRU T0PN3; EXU D5TW,2; BRU T0PN3
        CLA; MIN MTR0PN; BRR MTR0PN (NOT READY)
T0PN3  LDB X2; STB MTF,2; BRR MTR0PN
T0PN2  LDA =1; MIN MTR0PN; BRR MTR0PN
T0PN1  LDA =2; BRU T0PN2+1
*
* TAPE READ DRIVER
TRX    ZR0; LDX TN0; LDA MTF,2; SKG =0; BRU TR5
        EXU ETTW,2; BRU TR4; EXU TRTW,2; BRU TR3
* EXIT SINCE TAPE NOT READY
        MIN TRX; BRR TRX
* NO READ BEYOND EOT
TR4    LDA E0TBIT; BRM T0I; BRR TRX
* PSEUDO EOF
TR5    LDA E0FBIT; BRU TR4+1
* START READ
TR3    SKR MTF,2; LDA =NTRTRY=1; STA TXS3; BRM T2I; BRR TRX
* SIGNAL EOT (NOT INT. RTN.)
T0I    ZR0; LDX WBUFF; EAX 2,2; STX =2,2
        MRG WBE; STA -1,2; BRR T0I
* READ BUFFER
T2I    ZR0; LDA I0B; ADD TXC1; STA TXS4; SETINT T1I
        LDX TN0; EXU RTBW,2; EXU I0RW; P0T TXS4; BRR T2I
* CLEAN UP AFTER READ
T1I    ZR0; LDX WBUFF; EAX 2,2; STX =2,2; COPY XA,XB; ADD -1,2; LDX TN0
        EXU TFTW,2; BRU T1I3; BETW; BRU T1I1
T1I2   EXU ETTW,2; MRG E0TBIT; LDX WBUFF; STA 1,2; BRR T1I
* EOF WAS READ
T1I3   CBA; MRG E0FBIT; BRU T1I2
* TAPE ERROR ON READ
T1I4   ASCW; PIN TXS1; XMA TXS1; SUB I0B; ETR ADMSK; SKG =150; BRU T1I4
        EXU BTTW,2; BRU T1I4; MIN TRERR; SKR TXS3; BRU T1I5
        MIN TURERR; LDA TXS1; MRG ERBIT; BRU T1I2 (PERM ERROR)
* NOISE= READ AGAIN
T1I4   MIN TNERR; BRM T2I; BRR T1I
* TRY RE-READ, SCAN BACK
T1I5   SETINT T3I; EXU SRBW,2; BRR T1I
* PROCESS I31 WHEN BACKING OVER READ ERROR
T3I    ZR0; WIM TXS1; SETINT T2I; BRR T3I

```

```

*
* TAPE WRITE DRIVER
TWX   ZR0; LDX TN0; LDA MFL,2; SKG =0; BRM NTRPB
      EXU ETTW,2; BRU TW4; EXU TRTW,2; BRU TW3
      MIN TWX; BRR TWX (NOT READY)
* NO WRITE BEYOND EOT
TW4   LDA EOTBIT; BRM T0I; BRR TWX
* START WRITE
TW3   LDB =NTWTRY-1; STB TXS3; SKR MFL,2; LDX WBUFF
      LDA 1,2; ETR EORBIT; ADD 0,2; SUB WBUFF; SUB =2; STA 1,2
      LDX TN0; EXU BTTW,2; BRU TW2; BRM T0J; BRR TWX
* WRITE 3-INCH GAP AT LD PT
TW2   LDA =T0J; BRM T1J; BRR TWX
* WRITE 3 INCH GAP (NOT INTERRUPT ROUTINE)
T1J   ZR0; EXU ETFW,2; EXU IORW; P0T X4; STA BLK31; BRR T1J
* WRITE BUFFER
T0J   ZR0; LDX WBUFF; LDA 1,2; SKG =0; BRU T0J1
      LDA I0B; ADD TXC1; STA TXS4; SETINT T3J
      LDX TN0; EXU WTBW,2; EXU IORW; P0T TXS4; BRR T0J
T0J1  BRM T2K; BRR T0J
* WRITE FILE MARK. INTERRUPT AFTER 3' GAP.
T2J   ZR0; LDX TN0; EXU WFMW,2; E0M 16000B; P0T T2JFM
      SETINT T2K; BRR T2J
* CLEAN UP AFTER WRITE
T3J   ZR0; LDX TN0; LDA WBE; BETW; BRU T3J1
T3J2  EXU ETTW,2; MRG EOTBIT; LDX WBUFF; STA 1,2
      EAX 2,2; STX =2,2; BRR T3J
T3J1  MIN TWERR; SKR TXS3; BRU T3J3; MRG ERRBIT; BRU T3J2 (PERM ERR)
* TRY RE-WRITE, ERASE BACKWARDS
T3J3  ASCW; PIN TXS1; LDA I0B; XMA TXS1; SUB I0B
      CLB; LSH 14; STA TXS1
      EXU ETRW,2; EXU IORW; P0T TXS1; SETINT T0J; BRR T3J
* ERASE BEFORE FILE MARK
T4J   ZR0; LDX TN0; LDA =T2J; BRM T1J; BRR T4J
*
* 'ME0R', 'MBSF', 'MFSF', 'MBSR', 'ME0F', 'MERS', 'MREW', 'MERL' 12/15/65
*
* MAG TAPE CONTROL ROUTINES
*

```

```

* CLEAN UP AFTER SPACING (I33)
T2K   ZR0; LDX WBUFF; EAX 2,2; STX -2,2
      LDA WBE; STA -1,2; BRR T2K
* FORWARD SPACE FILE I31
T5K   ZR0; LDX TN0; EXU TFTW,2; BRU T5K1
* NOT EOF, CONTINUE
T5K2  SETINT T5K; EXU SFBW,2; BRR T5K
* EOF SENSED, DONE
T5K1  MIN TXS2; SKN TXS2; BRU **2; BRU T5K2
      WIM TXS1; SETINT T2K; BRR T5K
* BACKSPACE FILE FORWARD SPACE I31
T4K   ZR0; WIM TXS1; SETINT T2K; BRR T4K
* BACKSPACE RECORD I31
T1K   ZR0; LDA -1000B
T1K0  LDX TN0; MIN TXS2; WIM TXS1; SKA TXS1; BRU T1K2
      EXU BTTW,2; BRU T1K4; EXU TFTW,2; BRU T1K2
* NOT EOR, CONTINUE
T1K1  LDA INTR; STA BLK31; EXU SRBW,2; BRR T1K
T1K2  SKN TXS2; BRU **2; BRU T1K1
* END SENSED, GO FORWARD
T1K3  SETINT T3K; BRR T1K
* BOT SENSED (ACTUALLY I33)
T1K4  BRM T2K; BRR T1K
* BACKSPACE FILE I33
T3K   ZR0; LDX TN0; EXU TRTW,2; BRU T3K2
      MIN T3K; BRR T3K (NOT READY)
T3K2  EXU BTTW,2; BRU T3K1; SETINT T4K; EXU SFBW,2; BRR T3K
T3K1  BRM T2K; BRR T3K
* BACKSPACE FILE I31
T6K   ZR0; LDA T6K; STA T1K; CLA; BRU T1K0
*
* PREPARE FOR MAG TAPE CTRL
MTTR  ZR0; MIN MTTR; LDX TN0; EXU TRTW,2; BRU MTT1
      CXB; LDX MTTR; MIN -3,2; BRR -3,2 (NOT READY)
* SET UP INTERRUPT, EXIT
MTT1  EXU* MTTR; BRU MTT2; STA BLK31; BRR MTTR
* EOT/BOT SENSED
MTT2  LDA EOTBIT; BRM T01; LDX MTTR; BRR -3,2
* MAG TAPE CTRLS

```

\* REWIND

MREW ZR0; LDX TN0; EXU TRTW,2; BRU MREW1  
CXB; MIN MREW; BRR MREW (NOT READY)  
MREW1 EXU REWW,2; BRM T2K; BRR MREW

\* WRITE EOR

MEOR ZR0; LDX WBUFF; LDA 1,2; MRG EORBIT; STA 1,2  
BRM TWX; BRR MEOR; MIN MEOR; BRR MEOR

\* ERASE

MERS ZR0; LDA =T2K; BRM MTTR; EXU ETTW,2  
MERS1 BRM T1J; BRR MERS

\* LONG ERASE

MERL ZR0; LDA IORW; MRG =2; STA IORW  
LDA MERL; STA MERS; BRU MERS+1

\* WRITE EOF

MEOF ZR0; LDA =T4J; BRM MTTR; EXU ETTW,2  
XMA MEOF; STA MERS; XMA MEOF  
EXU BTTW,2; BRU MERS1 (ERASE IF EOF AT BOT); BRM T4J; BRR MEOF

\* FORWARD SPACE FILE

MFSF ZR0; LDA =T5K; BRM MTTR; EXU ETTW,2  
LDA =1; STA TXS2; EXU SFBW,2; BRR MFSF

\* BACKSPACE FILE

MBSF ZR0; LDA =T6K  
MBSP BRM MTTR; EXU BTTW,2  
LDA =2; STA TXS2; EXU SRBW,2; BRR MBSF

\* BACKSPACE RECORD

MBSR ZR0; LDA MBSR; STA MBSF; LDA =T1K; BRU MBSP  
END

4MDBG IDENT 7/02/67  
\* VERSION 12 6/18/67

\*  
\* SYSTEM FLAGS, BPDS, PARAMETERS, AND MACROS  
\*

\* ASSEMBLY FLAGS

CRXF EQU -1 (NO CARD READER)  
PNXF EQU 1 (PAPER TAPE PUNCH)  
LPXF EQU -1 (NO LINE PRINTER)  
RELCHN EQU -1 (OLD DRUM CHANNEL)  
940M EQU -1 (-1 FOR BERKELEY, 1 FOR 940)  
ARMF EQU 1 (ARMING FEATURE)  
V1 EQU 1 (VERSION 1.85-1)  
V2 EQU 1 (VERSION 1.85-2)  
V3 EQU 1 (VERSION 1.85-3)  
V4 EQU 1 (VERSION 1.85-4)  
V5 EQU 1 (VERSION 1.85-5)  
V6 EQU 1 (VERSION 1.85-6)  
V7 EQU 1 (VERSION 1.85-7)  
V8 EQU 1 (RELABELLED VERSION)  
FCB EQU 1 (FCB CHANGES)  
C181 EQU 1 (1.81 COMPATIBLE FILES)

\* BPDS

SBRS BPD 17300000B,1,1 SYSTEM MODE BRS  
T\$N BPD 00222000B,2 GO FROM NORMAL TO MONITOR MODE  
CKN BPD 00220100B,2 TURN ON THE CLOCK  
CKF BPD 00220200B,2 TURN OFF THE CLOCK  
LRR1 BPD 00220400B,2 LOAD RELABELLING REGISTER 1  
LRR2 BPD 00221000B,2 LOAD RELABELLING REGISTER 2  
LRR3 BPD 00221400B,2 LOAD RELABELLING REGISTER 3  
IF V5

\* PRINTER COMMANDS

EPT MACRO  
DATA 4014060B END OF PAGE TEST  
ENDM  
PFT MACRO  
DATA 4011060B SKIP IF NO PRINTER ERROR

```

PRT      ENDM
        MACR0
        DATA      4012060B          SKIP IF PRINTER READY
        ENDM
SKIPT0   MACR0      D
        DATA      210460B+D(1)*1000B
        ENDM
SPACE    MACR0      D
        DATA      210660B+D(1)*1000B
        ENDM
PRINT    MACR0      D
        DATA      242060B          ALERT INTERLACE, 1 CHAR. PER WORD
        DATA      215200B          ARM I31, DISCONNECT WHEN
        P0T        D(1)            TRANSMISSION IS COMPLETE.
        ENDM
CAT      0PD        04014000B,2
*
* BREAKPOINT TEST
BPT      0PD        04020000B,2
        ENDF
*
* I/O DEVICE 0PD'S
TTY5     MACR0; DATA 20277777B; ENDM
TTY5SKS EQU        24077000B
E0D      0PD        00600000B,1,1
ALR      0PD        00610026B,2     ALERT RAD
RRF      0PD        00602226B,2     READ RAD
WRF      0PD        00602266B,2     WRITE RAD
RIN      0PD        00616200B,2     RAD I0SD
I0SDE    EQU        00617200B
I0RDE    EQU        00614000B
RSR      MACR0; SKS* 10026B; ENDM
RSE      MACR0; SKS* 11026B; ENDM
CETE     MACR0; SKS* 11000B; ENDM
I0SDW    EQU        214200B
I0RDW    EQU        214000B
ALD      0PD        00210026B,2     ALERT DISC
DSR      0PD        00202626B,2     DISC READ
DRT      0PD        04010026B,2     DISC READY TEST

```

DET	0PD	04011026B,2	DISC ERROR TEST
DCT	0PD	04011000B,2	DISC CHANNEL ERROR TEST
*			
*			
* PARAMETERS			
BE	EQU	123	LAST BERKELEY BRS.
NP0P	EQU	44B	NUMBER OF SYSP0PS IN USE.
* W BUFFER DEVICE PARAMETERS			
RTCNT	EQU	64	PAPER TAPE READER BUFFER LENGTH
PNCNT	EQU	40	PAPER TAPE PUNCH BUFFER LENGTH
CRCNT	EQU	40	CARD READER BUFFER LENGTH
CRCNTB	EQU	40	
CPCNT	EQU	40	CARD PUNCH BUFFER LENGTH
CPCNTB	EQU	40	
NTAPE	EQU	2	NUMBER OF MAG TAPE UNITS
NLINK	EQU	0	
TCNT	EQU	199	LENGTH OF MAG TAPE BUFFER
LPCNT	EQU	132	LINE PRINTER BUFFER LENGTH
RTWT	EQU	RTCNT*40/3	PAPER TAPE READ TIME
PNWT	EQU	PNCNT*400/6	PAPER TAPE PUNCH TIME
CRWT	EQU	300	CARD READ TIME
CPWT	EQU	150	CARD PUNCH TIME
TXWT	EQU	20+TCNT/10	MAG TAPE TIME
LPWT	EQU	133	LINE PRINT TIME
NTRTRY	EQU	10	NUMBER OF REREADS
NTWTRY	EQU	3	NUMBER OF REWRITES
* FILE PARAMETERS			
NFILE	EQU	40	NUMBER OF FILES
MBUFX	EQU	34000000B	FBWRD FOR EXEC BLOCK
DBB	EQU	00400000B	PROTECTED FILE BUSY BIT
* TTY PARAMETERS			
NTTY	EQU	32	NUMBER OF TTYS
NTTB	EQU	NTTY+NLINK	TOTAL TTY BUFFERS
NLTTC	EQU	0	
NTTYC	EQU	70	NUMBER OF CHARS IN TTY BUFFER
TTYEWM	EQU	20	TTY EARLY WARNING (2 SEC)
AMB	EQU	40000B	ACCEPT MESSAGE BIT
AIB	EQU	100000B	ACCEPT INPUT BIT
APB	EQU	10000000B	ACCEPT PRINTER LINK BIT

```

AKB EQU 200000B ACCEPT KEYBOARD LINK BIT
8RB EQU 4000000B 8-LEVEL INPUT BIT
8PB EQU 2000000B 8-LEVEL OUTPUT BIT
ILB EQU 1000000B INPUT LINK BIT
OLB EQU 400000B OUTPUT LINK BIT
* PAC TABLE PARAMETERS
NSQU EQU 12 NUMBER OF CLOCK CYCLES IN SHORT QUANTUM.
NFQU EQU 36 FULL QUANTUM SIZE
NPAC EQU 144 NUMBER OF PACT SLOTS
NPPAR EQU 10 LENGTH OF PACT ENTRY
* JOB AND MEMORY PARAMETERS
NJOB1 EQU 32 NUMBER OF JOBS WITHOUT P.U.
NJOB EQU NJOB1+1 NUMBER OF JOBS
UMSZ EQU 15 INITIAL MACHINE SIZE
NMEM EQU 32 NUMBER OF PAGES
NSMEM EQU 7 NUMBER OF PAGES USED BY SYSTEM
NCMEM EQU 60B COMMON PART OF USER MACHINE
NSMT EQU 100B SIZE OF SMT
NUMEM EQU 100B*NCMEM NUMBER OF PRIVATE USER PAGES
NPUQ EQU 16 NUMBER OF PUCT ENTRIES
* RAD AND SWAPPING PARAMETERS
NRDQ EQU 20 MUST BE GT USER'S PAGES*2
NRTRY EQU 1 NO. OF READ TRIES FOR RAD.
NRAD EQU 4 NO. OF RADS
NSEC EQU 2*NRAD
L2NSEC EQU 1
NSBND EQU 18 NUMBER OF 16K BANDS RESERVED FOR SWAPPING
NSSP EQU 4000B LOC. OF 1ST SWAPPING AREA. MUST BE A
* MULTIPLE OF 1000B.
NSAM EQU 16 SIZE OF SWAPPER ASSOCIATIVE MEMORY
* DISC PARAMETERS
NDTRY EQU 4 NUMBER OF READ TRIES FOR DISC.
NDRQ EQU 30 NO. OF JOBS IN DISC QUEUE.
NDISCS EQU 8
IF NDISCS*32
NPBS EQU 20; ELSF 1; NPBS EQU 40; ENDF
MAXP EQU NPBS/2*200B+31*200B
MINP EQU *NPBS/2*200B+32*200B
TABLEN EQU NPBS/2*2*NDISCS*32+23

```



TABLEN EQU TABLEN/24

\* BUFFER PARAMETERS

NBUF<sub>FX</sub> EQU 3 NUMBER OF BUFFERS IN THE EXEC BLOCK  
NBUF EQU NBUF<sub>FX</sub> TOTAL NUMBER OF DISC BUFFERS.  
NDDW EQU 255 LENGTH OF DATA BLOCK  
BIN EQU NDDW+2 INDEX BLOCK NUMBER  
BIC EQU NDDW+3 INDEX CHANGED FLAG  
BDN EQU NDDW+4 DATA BLOCK NUMBER  
BDC EQU NDDW+5 CHANGED DATA FLAG  
BIP EQU NDDW+6 INDEX BLOCK POINTER  
BIA EQU NDDW+7 INDEX BLOCK DRUM ADDRESS  
NDXW EQU 124 LENGTH OF INDEX BLOCK

IF C181

NDXWC EQU 78 MAX NUMBER OF DATA BLOCKS PER FILE

ELSE 1; NDXWC EQU NDXW; ENDF

NDXWCR EQU NDXWC+1

NDXWR EQU 128 NUMBER OF WORDS TO READ/WRITE

NDBW EQU NDDW+8+NDXWR LENGTH OF DISC BUFFER

NDBS EQU NBUF\*NDBW SIZE OF BUFFER AREA

BX0 EQU NDBW-NDXWR INDEX BLOCK ORIGIN REL TO BUFF

BBP EQU BX0+NDXW-1 BACKWARD CHAIN WORD

BFP EQU BX0+NDXW+2 FORWARD CHAIN WORD

IXC EQU BX0+NDXW INDEX BLOCK CHECK WORD

\* TS BLOCK MAP

DBTOP EQU 37777B-NDBS-5-1-17-1 1ST WORD AFTER PRSYMS

SMOFIL EQU DBTOP SECONDARY MEMORY OUTPUT FILE

SMIFIL EQU DBTOP+1 SECONDARY MEMORY INPUT FILE

SMBA EQU DBTOP+2 SECONDARY MEMORY BUFFER ADDRESS

SMORN EQU DBTOP+3 SECONDARY MEMORY BDN ADDRESS

FBWRD EQU DBTOP+4 BUFFER AVAILABILITY BIT WORD

RMAP EQU DBTOP+5 RAD BIT MAP FOR FILES AND SWAPPING

PB EQU DBTOP+6

PX EQU PB+8

PPB EQU PX+8 POINTER TO PB CHAIN

NFORK EQU PPB+1 NUMBER OF FORKS COUNTER

FBADR EQU DBTOP+5+1+17+1 FIRST BUFFER ADDRESS

\* MONITOR AND EXEC LOCATIONS

RAW EQU 100B RAD ADDRESS OF W

CAW EQU 44000B CORE ADDRESS OF W

DAW	EQU	0	DISC ADDRESS OF W
RADSC	EQU	0	RAD ADDRESS OF DISC
CADSC	EQU	40000B	CORE ADDRESS OF DISC
DADSC	EQU	300B	DISC ADDRESS OF DISC
CASET	EQU	50000B	CORE ADDRESS OF SET
DASET	EQU	340B	DISC ADDRESS OF SET
DAEXEC	EQU	100B	DISC ADDRESS OF EXEC
RAEXEC	EQU	600B	RAD ADDRESS OF EXEC
* MACR0S			
A	EQU	1	
B	EQU	2	
AB	EQU	4	
BA	EQU	10B	
BX	EQU	20B	
XB	EQU	40B	
E	EQU	100B	
XA	EQU	200B	
AX	EQU	400B	
N	EQU	1000B	
X	EQU	20000000B	
COPY	MACR0	D	
K	NARG		
L	EQU	0	
M	EQU	1	
	RPT	K	
L	EQU	L+D(M)	
M	EQU	M+1	
	ENDR		
	DATA	4600000B+L	
	ENDM		
	IF	ARMF	
ARM!	MACR0 D; AIR; P0T D(1);	ENDM	
	ELSF	1	
ARM!	MACR0;	ENDM	
	ENDIF		
ENTRY	MACR0 L; ENT CNT NARG; RPT ENT CNT; L(ENT CNT) EXT		
ENT CNT	EQU ENT CNT-1; ENDR; ENDM		
SET INT	MACR0 A; LDA =A(1); STA BLK31; ENDM		
TDT	MACR0 L; L(1).W EQU *; RPT NTAPE; L(2) L(3).B+*-L(1).W; ENDR; ENDM		

```

IF      V1
RMFF   MACR0; ENDM
SMFF   MACR0; ENDM
READ   MACR0 D,G,1;G(1) RSR; BRU *-1; ALR; P0T =D(3)/100B
      E0D* 10000B; DATA I0SDE+D(1)/2000B(AND)37B+D(2)/40000B(AND)3*40B
      P0T =D(1)(AND)1777B*40000B+D(2)(AND)37777B; RRF; RSR; BRU *-1
      RSE; BRU G(1); CETE; BRU G(1); ENDM
      ELSF 1
SMFF   MACR0 D; DATA 234006B+D(1)*40B; ENDM
RMFF   MACR0 D; DATA 230006B+D(1)*40B; ENDM
      ENDF
ECHR   MACR0 N;ECHRWD EQU ECHRWD*400B+N(1)*B;NECHR EQU NECHR+1
      IF NECHR=2; DATA ECHRWD;ECHRWD EQU 0;NECHR EQU 0; ENDF; ENDM
ECHB   MACR0 N;ECHVB EQU N(1)B; RPT N(2); ECHR ECHV
ECHVB  EQU ECHVB+N(3); ENDR; ENDM
TRP    MACR0 L;ENTCNT NARG; RPT ENTCNT;L(ENTCNT) EQU TRAP
FRGT L(ENTCNT);ENTCNT EQU ENTCNT-1; ENDR; ENDM
CACR   MACR0 D; D(2)
      IF D(1); BRU PACACT; BRU PEST
      ELSF 1; BRU PEST; BRU PACACT; ENDF; ENDM
LBL    MACR0 D;1LBL EQU D(2); RPT D(2); LDA D(1)+1LBL-1; LRSH 6
1LBL   EQU 1LBL-1; ENDR; ENDM
*
*
*
* EXEC ENTRY POINTS

EXECI  EQU      10000B
EXECP  EQU      10001B
0FFINT EQU      10002B
*
*
*
DB     MACR0 D;ENTCNT EQU D(1)*B*200B+D(2)*100B
      ++40000000B+D(4)(AND)1*40B+D(4)(AND)2*10000000B
      RPT D(3); DATA ENTCNT;ENTCNT EQU ENTCNT+100B; ENDR; ENDM

F0RGT  MACR0 D;ENTCNT NARG; RPT ENTCNT; FRGT D(ENTCNT)

```

ENTCNT EQU ENTCNT-1; ENDR; ENDM

F0RGT CRXF,AIB,8PB,8RB,AMB,APB,AKB  
F0RGT MBUF,PNXF,LPXF  
F0RGT RTCNT,PNCNT,TCNT,LPCNT  
F0RGT RTWT,PNWT,TXWT  
F0RGT NTAPE,NLINK,NBUF,NBUF  
F0RGT NTRTRY,NTWTRY,NDTRY  
F0RGT NDDW,NDXW,NDBW  
F0RGT NFILE,UMSZ,NTTYC,TTYEWM  
F0RGT NPAC,NPPAR,NJOB,NJOB1,NFQU,NSQU  
F0RGT NPOP,NMEM,NSMEM,NCMEM,NUMEM  
F0RGT NPUQ,NSEC,L2NSEC,NDRQ,NSBND,NSAM,NSMT  
F0RGT DBB,ENTCNT  
F0RGT SMIFIL,SMBA,SMDRN,FBWRD,SM0FIL  
F0RGT BX0,BBP,BFP,BIN,BIC,BDN,BDC,BIP,BIA  
FREEZE  
END

5MDBG IDENT 7/02/67  
\* VERSION 12 6/18/67

\*  
\* SYSTEM FLAGS, θPDS, PARAMETERS, AND MACROS  
\*

\* ASSEMBLY FLAGS

CRXF EQU =1 (NO CARD READER)  
PNXF EQU 1 (PAPER TAPE PUNCH)  
LPXF EQU =1 (NO LINE PRINTER)  
RELCHN EQU =1 (OLD DRUM CHANNEL)  
940M EQU =1 (=1 FOR BERKELEY, 1 FOR 940)  
ARMF EQU 1 (ARMING FEATURE)  
V1 EQU 1 (VERSION 1.85=1)  
V2 EQU 1 (VERSION 1.85=2)  
V3 EQU 1 (VERSION 1.85=3)  
V4 EQU 1 (VERSION 1.85=4)  
V5 EQU 1 (VERSION 1.85=5)  
V6 EQU 1 (VERSION 1.85=6)  
V7 EQU 1 (VERSION 1.85=7)  
V8 EQU 1 (RELABELLED VERSION)  
FCB EQU 1 (FCB CHANGES)  
C181 EQU 1 (1.81 COMPATIBLE FILES)

\* θPDS

SBR5 θPD 17300000B,1,1 SYSTEM MODE BRS  
TSN θPD 00222000B,2 GO FROM NORMAL TO MONITOR MODE  
CKN θPD 00220100B,2 TURN ON THE CLOCK  
CKF θPD 00220200B,2 TURN OFF THE CLOCK  
LRR1 θPD 00220400B,2 LOAD RELABELLING REGISTER 1  
LRR2 θPD 00221000B,2 LOAD RELABELLING REGISTER 2  
LRR3 θPD 00221400B,2 LOAD RELABELLING REGISTER 3  
IF V5

\* PRINTER COMMANDS

EPT MACRO  
DATA 4014060B END θF PAGE TEST  
ENDM  
PFT MACRO  
DATA 4011060B SKIP IF NO PRINTER ERROR

```

PRT      ENDM
        MACR0
        DATA      4012060B          SKIP IF PRINTER READY
        ENDM
SKIPT0   MACR0      D
        DATA      210460B+D(1)*1000B
        ENDM
SPACE    MACR0      D
        DATA      210660B+D(1)*1000B
        ENDM
PRINT    MACR0      D
        DATA      242060B          ALERT INTERLACE, 1 CHAR. PER WORD
        DATA      215200B          ARM I31, DISCONNECT WHEN
        PBT        D(1)             TRANSMISSION IS COMPLETE.
        ENDM
CAT      BPD        04014000B,2
*
* BREAKPOINT TEST
BPT      BPD        04020000B,2
        ENDF
*
* I/O DEVICE BPD'S
TTYS     MACR0; DATA 20277777B; ENDM
TTYSKS   EQU        24077000B
E0D      BPD        00600000B,1,1
ALR      BPD        00610026B,2    ALERT RAD
RRF      BPD        00602226B,2    READ RAD
WRF      BPD        00602266B,2    WRITE RAD
RIN      BPD        00616200B,2    RAD I0SD
I0SDE    EQU        00617200B
I0RDE    EQU        00614000B
RSR      MACR0; SKS* 10026B; ENDM
RSE      MACR0; SKS* 11026B; ENDM
CETE     MACR0; SKS* 11000B; ENDM
I0SDW    EQU        214200B
I0RDW    EQU        214000B
ALD      BPD        00210026B,2    ALERT DISC
DSR      BPD        00202626B,2    DISC READ
DRT      BPD        04010026B,2    DISC READY TEST

```

DET	OPD	04011026B,2	DISC ERROR TEST
DCT	OPD	04011000B,2	DISC CHANNEL ERROR TEST
*			
*			
* PARAMETERS			
BE	EQU	123	LAST BERKELEY BR\$.
NP0P	EQU	44B	NUMBER OF SYSP0PS IN USE.
* W BUFFER DEVICE PARAMETERS			
RTCNT	EQU	64	PAPER TAPE READER BUFFER LENGTH
PNCNT	EQU	40	PAPER TAPE PUNCH BUFFER LENGTH
CRCNT	EQU	40	CARD READER BUFFER LENGTH
CRCNTB	EQU	40	
CPCNT	EQU	40	CARD PUNCH BUFFER LENGTH
CPCNTB	EQU	40	
NTAPE	EQU	2	NUMBER OF MAG TAPE UNITS
NLINK	EQU	0	
TCNT	EQU	199	LENGTH OF MAG TAPE BUFFER
LPCNT	EQU	132	LINE PRINTER BUFFER LENGTH
RTWT	EQU	RTCNT*40/3	PAPER TAPE READ TIME
PNWT	EQU	PNCNT*400/6	PAPER TAPE PUNCH TIME
CRWT	EQU	300	CARD READ TIME
CPWT	EQU	150	CARD PUNCH TIME
TXWT	EQU	20+TCNT/10	MAG TAPE TIME
LPWT	EQU	133	LINE PRINT TIME
NTRTRY	EQU	10	NUMBER OF REREADS
NTWTRY	EQU	3	NUMBER OF REWRITES
* FILE PARAMETERS			
NFILE	EQU	40	NUMBER OF FILES
MBUFx	EQU	34000000B	FBWRD FOR EXEC BLOCK
DBB	EQU	00400000B	PROTECTED FILE BUSY BIT
* TTY PARAMETERS			
NTTY	EQU	32	NUMBER OF TTYS
NTTB	EQU	NTTY+NLINK	TOTAL TTY BUFFERS
NLTTC	EQU	0	
NTTYC	EQU	70	NUMBER OF CHARS IN TTY BUFFER
TTYEWM	EQU	20	TTY EARLY WARNING (2 SEC)
AMB	EQU	40000B	ACCEPT MESSAGE BIT
AIB	EQU	100000B	ACCEPT INPUT BIT
APB	EQU	10000000B	ACCEPT PRINTER LINK BIT

```

AKB      EQU      200000B      ACCEPT KEYBOARD LINK BIT
8RB      EQU      4000000B     8-LEVEL INPUT BIT
8PB      EQU      2000000B     8-LEVEL OUTPUT BIT
ILB      EQU      1000000B     INPUT LINK BIT
OLB      EQU      400000B      OUTPUT LINK BIT
* PAC TABLE PARAMETERS
NSQU     EQU 12  NUMBER OF CLOCK CYCLES IN SHORT QUANTUM.
NFQU     EQU      36          FULL QUANTUM SIZE
NPAC     EQU      144         NUMBER OF PACT SLOTS
NPPAR    EQU 10  LENGTH OF PACT ENTRY
* JOB AND MEMORY PARAMETERS
NJOB1    EQU      32          NUMBER OF JOBS WITHOUT P.U.
NJOB     EQU      NJOB1+1     NUMBER OF JOBS
UMSZ     EQU      15          INITIAL MACHINE SIZE
NMEM     EQU 32  NUMBER OF PAGES
NSMEM    EQU 7   NUMBER OF PAGES USED BY SYSTEM
NCMEM    EQU      60B        COMMON PART OF USER MACHINE
NSMT     EQU      100B       SIZE OF SMT
NUMEM    EQU 100B*NCMEM      NUMBER OF PRIVATE USER PAGES
NPUQ     EQU 16  NUMBER OF PUCT ENTRIES
* RAD AND SWAPPING PARAMETERS
NRDQ     EQU      20          MUST BE GT USER'S PAGES*2
NRTRY    EQU      1          NO. OF READ TRIES FOR RAD.
NRAD     EQU      4          NO. OF RADS
NSEC     EQU      2*NRAD
L2NSEC   EQU      1
NSBND    EQU      18          NUMBER OF 16K BANDS RESERVED FOR SWAPPING
NSSP     EQU      4000B      LOC. OF 1ST SWAPPING AREA. MUST BE A
*                               MULTIPLE OF 1000B.
NSAM     EQU 16  SIZE OF SWAPPER ASSOCIATIVE MEMORY
* DISC PARAMETERS
NDTRY    EQU      4          NUMBER OF READ TRIES FOR DISC.
NDRQ     EQU      30         NO. OF JOBS IN DISC QUEUE.
NDISCS   EQU      16
        IF      NDISCS=32
NPBS     EQU      20; ELSF 1; NPBS EQU 40; ENDF
MAXP     EQU      NPBS/2*200B+31*200B
MINP     EQU      =NPBS/2*200B+32*200B
TABLEN   EQU      NPBS/2*2*NDISCS*32+23

```



TABLÉN EQU TABLÉN/24

\* BUFFER PARAMETERS

NBUFX	EQU	3	NUMBER OF BUFFERS IN THE EXEC BLOCK
NBUF	EQU	NBUFX	TOTAL NUMBER OF DISC BUFFERS.
NDDW	EQU	255	LENGTH OF DATA BLOCK
BIN	EQU	NDDW+2	INDEX BLOCK NUMBER
BIC	EQU	NDDW+3	INDEX CHANGED FLAG
BDN	EQU	NDDW+4	DATA BLOCK NUMBER
BDC	EQU	NDDW+5	CHANGED DATA FLAG
BIP	EQU	NDDW+6	INDEX BLOCK POINTER
BIA	EQU	NDDW+7	INDEX BLOCK DRUM ADDRESS
NDXW	EQU	124	LENGTH OF INDEX BLOCK
	IF	C181	
NDXWC	EQU	78	MAX NUMBER OF DATA BLOCKS PER FILE

ELSF 1;NDXWC EQU NDXW; ENDF

NDXWCR	EQU	NDXWC+1	
NDXWR	EQU	128	NUMBER OF WORDS TO READ/WRITE
NDBW	EQU	NDDW+8+NDXWR	LENGTH OF DISC BUFFER
NDBS	EQU	NBUF*NDBW	SIZE OF BUFFER AREA
BX0	EQU	NDBW*NDXWR	INDEX BLOCK ORIGIN REL TO BUFF
BBP	EQU	BX0+NDXW*1	BACKWARD CHAIN WORD
BFP	EQU	BX0+NDXW*2	FORWARD CHAIN WORD
IXC	EQU	BX0+NDXW	INDEX BLOCK CHECK WORD

\* TS BLOCK MAP

DBTOP	EQU	37777B=NDBS-5-1-17-1	1ST WORD AFTER PRSYMS
SMOFIL	EQU	DBTOP	SECONDARY MEMORY OUTPUT FILE
SMIFIL	EQU	DBTOP+1	SECONDARY MEMORY INPUT FILE
SMBA	EQU	DBTOP+2	SECONDARY MEMORY BUFFER ADDRESS
SMDRN	EQU	DBTOP+3	SECONDARY MEMORY BDN ADDRESS
FBWRD	EQU	DBTOP+4	BUFFER AVAILABILITY BIT WORD
RMAP	EQU	DBTOP+5	RAD BIT MAP FOR FILES AND SWAPPING
PB	EQU	DBTOP+6	
PX	EQU	PB+8	
PPB	EQU	PX+8	POINTER TO PB CHAIN
NFORK	EQU	PPB+1	NUMBER OF FORKS COUNTER
FBADR	EQU	DBTOP+5+1+17+1	FIRST BUFFER ADDRESS

\* MONITOR AND EXEC LOCATIONS

RAW	EQU	100B	RAD ADDRESS OF W
CAW	EQU	44000B	CORE ADDRESS OF W

DAW	EQU	0	DISC ADDRESS OF W
RADSC	EQU	0	RAD ADDRESS OF DISC
CADSC	EQU	40000B	CORE ADDRESS OF DISC
DADSC	EQU	300B	DISC ADDRESS OF DISC
CASET	EQU	50000B	CORE ADDRESS OF SET
DASET	EQU	340B	DISC ADDRESS OF SET
DAEXEC	EQU	100B	DISC ADDRESS OF EXEC
RAEXEC	EQU	600B	RAD ADDRESS OF EXEC

\* MACR0S

A	EQU	1
B	EQU	2
AB	EQU	4
BA	EQU	10B
BX	EQU	20B
XB	EQU	40B
E	EQU	100B
XA	EQU	200B
AX	EQU	400B
N	EQU	1000B
X	EQU	20000000B
COPY	MACR0	D
K	NARG	
L	EQU	0
M	EQU	1
	RPT	K
L	EQU	L+D(M)
M	EQU	M+1
	ENDR	
	DATA	4600000B+L
	ENDM	
	IF	ARMF
ARM1	MACR0 D; AIR; P0T D(1);	ENDM
	ELSF	1
ARM1	MACR0;	ENDM
	ENDF	
ENTRY	MACR0 L; ENT CNT NARG; RPT ENT CNT; L(ENT CNT) EXT	
ENT CNT	EQU ENT CNT-1; ENDR; ENDM	
SET INT	MACR0 A; LDA =A(1); STA BLK31; ENDM	
TDT	MACR0 L; L(1).W EQU *; RPT NTAPE; L(2) L(3).B+*.L(1).W; ENDR; ENDM	

```

IF          V1
RMFF       MACR0; ENDM
SMFF       MACR0; ENDM
READ       MACR0 D,G,1;G(1) RSR; BRU **1; ALR; P0T =D(3)/100B
           E0D* 10000B; DATA I0SDE+D(1)/2000B(AND)37B+D(2)/40000B(AND)3*40B
           P0T =D(1)(AND)1777B*40000B+D(2)(AND)37777B; RRF; RSR; BRU *-1
           RSE; BRU G(1); CETE; BRU G(1); ENDM
           ELSF 1
SMFF       MACR0 D; DATA 234006B+D(1)*40B; ENDM
RMFF       MACR0 D; DATA 230006B+D(1)*40B; ENDM
ENDF
ECHR       MACR0 N;ECHRWD EQU ECHRWD*400B+N(1)*B;NECHR EQU NECHR+1
           IF NECHR=2; DATA ECHRWD;ECHRWD EQU 0;NECHR EQU 0; ENDF; ENDM
ECHB       MACR0 N;ECHVB EQU N(1)B; RPT N(2); ECHR ECHV
ECHVB      EQU ECHVB+N(3); ENDR; ENDM
TRP        MACR0 L;ENTCNT NARG; RPT ENTCNT;L(ENTCNT) EQU TRAP
FRGT L(ENTCNT);ENTCNT EQU ENTCNT-1; ENDR; ENDM
CACR       MACR0 D; D(2)
           IF D(1); BRU PACACT; BRU PEST
           ELSF 1; BRU PEST; BRU PACACT; ENDF; ENDM
LBL        MACR0 D;1LBL EQU D(2); RPT D(2); LDA D(1)+1LBL-1; LRSH 6
1LBL      EQU 1LBL-1; ENDR; ENDM
*
*
*
* EXEC ENTRY POINTS
EXECI      EQU      10000B
EXECP      EQU      10001B
OFFINT     EQU      10002B
*
*
*
DB         MACR0 D;ENTCNT EQU D(1)*B*200B+D(2)*100B
           **400000000B+D(4)(AND)1*40B+D(4)(AND)2*10000000B
           RPT D(3); DATA ENTCNT;ENTCNT EQU ENTCNT+100B; ENDR; ENDM
FORGT      MACR0 D;ENTCNT NARG; RPT ENTCNT; FRGT D(ENTCNT)

```

ENTCNT EQU ENTCNT-1, ENDR, ENDM

FØRGT CRXF, AIB, 8PB, 8RB, AMB, APB, AKB  
FØRGT MØUFØ, PNØF, LPØF  
FØRGT RTCNT, PNCNT, TCNT, LPCNT  
FØRGT RTWT, PNWT, TXWT  
FØRGT NTAPE, NLINK, NØUFØ, NØUF  
FØRGT NTRTRY, NTWTRY, NDTRY  
FØRGT NDDW, NDXW, NDBW  
FØRGT NFILE, UMSZ, NTTYC, TTYEWM  
FØRGT NPAC, NPPAR, NJØB, NJØB1, NFØU, NSØU  
FØRGT NPØP, NMEM, NSMEM, NCMEM, NUMEM  
FØRGT NØUØ, NSEC, L2NSEC, NDRØ, NSBND, NSAM, NSMT  
FØRGT DBB, ENTCNT  
FØRGT SMIFIL, SMBA, SMDRN, FBWRD, SMØFIL  
FØRGT BXØ, BBP, BFP, BIN, BIC, BDN, BDC, BIP, BIA  
FREEZE  
END